European Intellectual Property Review

2000

Article

SOFTWARE COPYRIGHT PROTECTION: CAN EUROPE LEARN FROM AMERICAN CASE LAW? PART 1

Estelle Derclaye.

Legislation: Council Directive 91/250 on the legal protection of computer programs
Copyright Act 1976 (United States)
Keywords: Copyright; EC law; Software; United States
Abstract: Comparison of how Software Directive 91/250 and 1976 Act provide protection for computer programs, considering differences in legal text and policy issues in relation to definition of 'computer program' and scope of protection.

*7 Eight years have passed since the adoption of the European [FN1] Software Directive. [FN2] European copyright software case law has, however, not been as abundant as U.S. case law. After having searched for the solution to the copyright protection of software for several years, American courts have now established a body of well-settled law. On the other hand, the few European cases which have addressed the issue have not yet satisfactorily resolved the question of copyrightability of programs. If European courts do not adopt a uniform method of judging these issues, inconsistent decisions will produce great legal insecurity. American precedents may be a useful guide for European judges to utilise in deciding these complex issues. In order to appreciate whether American case law can serve as a model, the policies and the formulation of the legislative texts must be similar in both legal systems and the reasoning underlying American decisions must be adaptable to European law. A comparison between European and American laws is rendered difficult by the rather high obscurity of texts in both systems and by the significant lack of European case law. A review of the statutory differences is important. Indeed, an analysis based primarily on a traditional common law approach (based on the analysis of cases) would be biased, as the Directive evidences significant elements taken from both civil law and common law legal systems.
The first section of this article, after highlighting the reasons for Europe's late initiative to protect software, examines whether the Directive and the 1976 American Copyright Act [FN3] perspectives are similar as regards the choice of protection. This section then attempts to clarify whether the Directive reflects a fully copyright-

orientated approach or a droit d'auteur perspective. The term "copyright" will be used in two ways. Copyright sensu stricto ("copyright s.s.") will refer to the civil law system of droit d'auteur as opposed to the common law system of "copyright". Copyright sensu largo ("copyright s.l.") will refer to intellectual property in the restricted sense of literary and artistic ownership as opposed to other kinds of industrial property rights (patents, trade marks, sui generis protection, etc.). In this sense then, it will include both copyright s.s. and droit d'auteur systems.
The analysis in the second section is confined to two areas: the definition of "computer program" and the copyrightability requirements (expression and originality). This section compares the statutory texts in these two areas. Because the similarities between both texts in these areas have already been widely discussed, a brief overview addresses the common foundations of the definition of "computer program" and the scope of protection in both legal systems. This section therefore compares the differences between particular provisions of European and U.S. statutory laws in two respects: the formulation of the legal text (i.e. the choice of words) and the policy (i.e. the choice of droit d'auteur or copyright s.s.).
The first section of Part 2 of this article (to be published in the next issue of E.I.P.R.) concentrates on the significant American and European decisions which resolved issues concerning the definition of a program and the scope of protection. The U.S. decisions on infringement are reviewed from the perspective of the legal reasoning used in deciding the cases. This angle is undertaken to assist European judges in their search for a satisfactory method of approaching infringement cases. Relevant European decisions on infringement will be reviewed country by country in order to compare the different approaches adopted by the judiciaries of the several Member States. This section further examines whether American precedents have already influenced European case law. Finally, considering the very few and inconstant European cases, the last section incorporates the conclusions drawn in the statutory and case analysis in order to suggest a test, or guideline, for determining whether European judges should consider American precedents.

A Contextual Background to the Protection of Computer Programs in Europe

The reasons for the protection and the influence of U.S. law on the elaboration of the Software Directive

E.C. late initiative

In comparison with the early U.S. initiative, [FN4] the European Community did not progress as swiftly to protect software by copyright. Several reasons explain this delay. Initially, there were discussions concerning *8 whether copyright was within the scope of application of the Treaty of Rome. It was only in the early 1980s that the European Court of Justice resolved this issue holding that copyright s.l. was included to Article 36 of the Treaty. [FN5] In 1985, when the Community began contemplating the need to extend copyright protection to programs, it was primarily preoccupied by the achievement of the single market. [FN6] It was not until 1988, in the Green Paper on Copyright and the Challenge of New Technology, [FN7] that the E.C.'s desire to harmonise software copyright protection materialised. Finally, the adoption of the European Software Directive was delayed by the long and intense pre- legislative debate on the protectability of interfaces and the decompilation issue. [FN8] This long

maturation gives a more important weight to the Directive text compared to the U.S. Copyright Act's more general dispositions on software. [FN9]

The reasons for harmonisation

The Community decided to protect software at the European level, while it was aware that several Member States already protected programs in their copyright s.s. or droit d'auteur general laws. [FN10] The reason underlying harmonisation was the divergence between national intellectual property laws. This divergence had negative consequences: national laws constituted barriers to the free circulation of goods and to the efficient development of the European software industry. In the light of the negative effects that I.P. rights had on the free circulation of goods, [FN11] the Community launched a harmonisation programme. [FN12] The Green Paper on Copyright and the Challenge of New Technology was the first legislative programme in the field of intellectual property undertaken by the Community, and the Software Directive the Community's first legislative effort in the author's rights field. [FN13] While this protection needed to be uniform, it also needed to be strong. The Community's goal was thus twofold: to unify the protection and to reinforce it. The interest in affording an adequate level of protection was aimed at promoting the industrial development of European software companies. [FN14] The growing role of computer technology in the world requires a secure and consistent protection for computer programs. If the protection were low, partial, divided or simply not existing, the consequences for software companies would be disastrous. Programs can be copied so easily and so cheaply that the development of the European software industry, without such protection, would not be fostered. [FN15] Research and investment in the European computer industry, as a consequence, would grow more slowly than in other industrialised countries. Moreover, if the level of software protection in Europe were less than that of other countries, foreign software companies would be reluctant to invest in Europe. Concurrently, it would be a deterrent for European companies that are starting up to remain and develop software in Europe. This situation would have two consequences: no production in Europe and delocalisation in countries where *9 software is afforded good protection, as it is in the United States. [FN16]

Although solid copyright protection was necessary, the Community was careful not to grant an excessive protection to authors. [FN17] Disproportionate protection would allow authors to deny other creators access to the program's ideas and principles, thus stifling competition. A balance between monopoly rights and E.C. competition policy had to be found. The Directive therefore provides for decompilation to achieve inter-operability and the preamble reaffirms that the Directive provisions are without prejudice to the application of Articles 85 and 86 of the Treaty. [FN18]

The choice of protection

Different types of protection were contemplated to protect software. Patent law and contract law were considered to afford insufficient protection. [FN19] Instead, copyright s.l. was deemed more suitable because many Member States and non-E.C. countries had already made that choice, the United States being the first country to start the trend. [FN20] European policy makers were aware that the United States had chosen copyright law and that, since trade in software products was becoming increasingly international, it would be desirable to harmonise national laws on

computer program protection with the protection afforded by the U.S. laws. [FN21] Thus, while an explicit trace of the American influence cannot be found in any of the preparatory texts nor in the Directive preamble, the American choice of copyright s.l. influenced the European Community. [FN22]
Another reason to adopt copyright s.l. was the existence of a high degree of protection by international instruments. [FN23] If copyright s.l. was chosen to protect software, and programs were classified as literary works, they would be within the scope of the Berne Convention. This advice for a copyright s.l. protection came also from another source: the expert group of the WIPO recommended renouncing a specific treaty affording a sui generis protection to computer programs. [FN24] In conclusion, in addition to reflection of U.S. law, international considerations guided the choice of the European Community.

The directive: a "droit d'auteur" or "copyright" perspective?

To discover the perspective in which the Directive was drafted, it is necessary to search the texts in decreasing order of legal importance. Nothing can be found on the policy viewpoint in the Directive Recitals or text, nor in the explanatory, memorandum of the Directive Proposal. The Green Paper then needs to be analysed. The Green Paper is not very explicit about the legal grounds on which it is founded to act in literary ownership, [FN25] nor about the choice of copyright s.s. or droit d'auteur as a perspective. While it stresses economic motivations, [FN26] it nevertheless refers to cultural goals. [FN27] Nowhere does the Green Paper explicitly state whether droit d'auteur or copyright s.s. is preferable.
On some rare issues, the Community's position is more transparent because it wished either to depart completely from the U.S. perspective or to integrate the Anglo-Saxon viewpoint. Two striking examples illustrate this point. For instance, the E.C. tackled the decompilation issue totally differently from the United States. Decompilation is not expressly addressed in the 1976 Copyright Act but is permitted through the fair use doctrine. [FN28] Since the fair use doctrine is not part of *10 E.C. law, [FN29] a very strict and precise provision was adopted for this issue to prevent states from adopting a similar fair use doctrine. [FN30] This is the most striking example of a provision adopted in a precise droit d'auteur perspective. The presumption that the author's employer is the rightholder of the copyright illustrates the opposite choice. [FN31] This provision reflects the influence of the economic orientation which forms the basis of copyright s.s. systems.
Several commentators have attempted to construe the general approach that the Community intended to take. Pamela Samuelson believes that the Directive springs from a civil law tradition, but does not further explore this assumption. [FN32] Other authors find that the Green Paper embraced a copyright-orientated approach and criticise it for taking solely this point of view. [FN33] F. Brison and J.-P. Triaille believe that the "author's right" countries have a tendency to come close to the copyright s.s. system, the objective of which is the protection of investments. For them, this influence explains the low level of originality required for computer programs in the Directive. [FN34] M. Moller believes the original English text of the Green Paper intended copyright s.s.--this seems to be the case at least for the authorship, scope of rights and term of protection questions [FN35]--and questions the relevance of this choice, since the majority of Member States are of an author's right tradition. [FN36] Although these three legal issues (authorship, scope of rights and

term) are not discussed in this article, they show that the Directive is of a mixed nature.
In conclusion, except on very few points, the legal texts do not help one to know whether the Community's intent was to follow the United States or at least a copyright s.s. perspective. This demonstrates in any case how confused the European policy position was and, accordingly, how blurred the influence of both systems is on the directive. It can only be concluded that the texts reflect a unequal mix of both influences.

A Statutory Analysis of the Differences between U.S. and E.U. Law

Before analysing the differences between both texts, some similarities on which everyone agrees have to be pointed out. Concerning the definition of a "computer program", the U.S. and E.C. legal systems have chosen to categorise computer programs as literary works. [FN37] They also protect source and object codes. [FN38] Source code is the code in which the developer writes the program. It is understandable to humans. Object code is composed of ones and zeros and gives the electrical impulses to the computer in order to achieve a particular result. It is only understandable to the machine. Both laws respect the principle of the sole protection of expression, common to copyright s.s. and droit d'auteur systems. Only a low threshold of originality is required in both systems.
As the first section of this article has shown, the Directive's provisions and Recitals do not treat of the chosen perspective. Hence the text formulation is analysed first because it might reveal the masked policy.

The definition of computer program

Text formulation

(1) Europe. The computer program is not defined in the Directive. Several reasons explain this choice. First, framers rightly thought a definition would be quickly outdated because of rapid technological changes. [FN39] The very reason behind this decision was that any definition would by itself be restrictive. Despite the absence of a *11 definition in the Directive itself, the Directive Proposal stated:
Given the present state of the art, the word "program" should be taken to encompass the expression in any form, language, notation or code, of a set of instructions the purpose of which is to cause a computer to execute a particular task or function. [FN40]
Recital 7 of the Directive adds that the term "computer program" shall include programs in any form, including those which are incorporated into hardware, such as firmware. [FN41] It also includes preparatory design work if this can lead to the development of a computer program at a later stage. This latter sentence means that flow charts and other non-literal elements can receive protection only to the extent that they finally permit the building of a computer program. Finally, the Green Paper refers to a more detailed definition adopted for the purposes of the WIPO Model Provisions of the Protection of Computer Software in 1978 [FN42]:
a set of instructions, which can, once transcribed on a medium decodable by a machine, make indicate, accomplish or obtain a particular function, task or result by a machine capable of treating information.

(2) The United States. In the United States, the definition lies in section 101 of the 1976 Copyright Act:
"computer program": a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.
Even if the computer program is defined, the Copyright Act does not state that computer programs are protected. [FN43] It only provides limits to the copyright in a computer program in section 117. The copyrightability of a computer program can be deduced indirectly from this section. [FN44]
(3) Comparison. The very first and fundamental difference between E.U. and U.S. law concerning the definition of "program" is that there is no binding definition in Europe. The repercussions of this will be discussed further. In order to discover if there truly are differences between both legal systems pertaining to the definition, it is necessary to dissect the definition to eliminate the apparent differences.
At first sight and in certain respects, the U.S. definition and terminology seem broader. First, section 101 speaks of "statements" or "instructions", whereas the Proposal only speaks of "instructions". This apparent contradiction is easily resolved. Indeed, it seems that in the 1980 U.S. Copyright Act revision, "statements" referred to source code and instructions, to object code. On the other hand, the European definition says "instructions expressed in any code"; this implies that object and source codes are included. Consequently, both formulations express the same concepts, so that there is no difference between European and U.S. law.
Secondly, the U.S. definition mentions "directly or indirectly". These adverbs refer to the previously discussed words: "statements" and "instructions"; "directly" referring to "instructions" which itself refers to object code and "indirectly" referring to "statements", which itself refers to source code. Indeed, in Apple v. Franklin, [FN45] the court said that "directly" related to the notion of object code. The adverbs relate to the relationships between the languages and the machine. Only object code can be read directly by the machine. Source code must be translated into object code for the machine to be able to read it. [FN46] Nowadays, TRIPs expressly states that both object and source codes are protected in all signatory countries. [FN47] Therefore this second difference is also only apparent and will not raise definition problems in the future.
Thirdly, the Directive Proposal definition states "the expression of a set of instructions ... " while section 101 of the Copyright Act only mentions "instructions". Theoretically, it is possible to interpret that the Proposal definition affords less protection because the phrasing "expression of a set of instructions" might imply that the instructions themselves are not protectable, but only their expression. On the contrary, as the U.S. definition does not differentiate between the expression of the instructions and the instructions themselves, this would imply that protection extends to the instructions themselves. This formulation cannot create a divergence between both laws. Indeed, it is necessary to look for the intent of both legislators and the underlying logic in both texts. Both laws have adopted the idea/expression principle. Thus, in U.S. law, "instructions" must mean "expression of instructions" because U.S. law would not protect instructions that are ideas. In other words, if an instruction merges with an idea, U.S. law would compel its exclusion from protection. In conclusion, both provisions intended the term "instructions" as the expression of an idea, despite their different phrasing.
Fourthly, the term "result" proves the most controversial. Indeed, "result" appears in the U.S. definition but not in the Directive Proposal, which only mentions "tasks or functions". This concept of "result" raises four questions. First, what is a result?

Secondly, is it comprised in the program definition as part of the program or is it a product of the program only and therefore outside its scope? Thirdly, is it included in the European concept of program? Fourthly, does it include "tasks or functions"?

A single program can produce many different results. Some are known as user interfaces "(UIs"). The definition of a UI is broad; it includes "all the ways that the human user interacts with the computer to accomplish the program's tasks". [FN48] Some UIs are invisible to the user, such as the electrical impulses arising inside the computer (switching the computer on, clicking the *12 mouse or converting a file from a version into another version); others are visible and produce a visual or screen display, either literal (e.g. help screen, menus [FN49]) or pictorial (e.g. spreadsheet, video game). Visible UIs are the most important results for the present discussion. Indeed, for results to be protected by copyright s.l., they must not only be functional but must convey information, entertain or portray an appearance; in other words, they must be perceptible to human senses. [FN50]

The question of the inclusion of results in the definition is rather awkward. Dennis Karjala, speaking of user interfaces, argues that "the definition of a computer program in the Copyright Act clearly distinguishes between the set of statements or instructions constituting the program and the 'certain results' that are achieved by execution of those statements and instructions". [FN51] He concludes that "interfaces are not covered by the program copyright". [FN52] The program is simply the code. [FN53] This statement implies that results in general are not comprised in the definition of program and cannot be protected by copyright through it. His argument is logical. The program's aim is to produce a result. But the result is not the program. It is merely the product caused by the program or, to be more precise, the solution of a problem solved by a mathematical operation. The program in itself is materialised by the instructions it gives to the computer only. If Karjala's argument is to be followed, then the whole controversy of the inclusion of the term "result" in the program definition loses its interest. In any case, Karjala's argument aim is to solve and close the debate. The remaining question is whether results generated by computers are protected under other categories of works. This question, which has been under consideration in U.S. case law dealing with user interfaces will be analysed in Part 2 of this article. [FN54]

If Karjala's argument is not adopted, the third question must be considered. The answer is found in interpreting the intent of the European legislator. Recital 10 of the Directive implies that user interfaces can be protected. Furthermore, the WIPO definition to which the Green Paper refers includes the term "result". In conclusion as the intent was to promote a broad concept of "program" and as the WIPO definition and Recital 10 include the result in the scope of protection, results of computer programs must be included in the European concept of a program. In sum, even if Karjala's opinion is not followed, the results of a computer program would be protected in both systems.

The last question which remains is whether "tasks or functions" are included in the U.S. definition. Two arguments can be made. On the first hand, they cannot be said to be part of a result. "Result" leads one to think of an output, independent of the tasks and functions that the computer needs to go through to achieve this output, while "task" or "function" seem to refer to the internal operations that the computer performs to achieve the output. A result is the product of a task or function. The concepts do not overlap. On the other hand, considering the U.S. definition, "result" potentially means more than just the output and could include the tasks or functions

accomplished in order to achieve the result. This can be assumed because the definition says "instructions to be used". Indeed, a contrario, instructions alone would accomplish nothing if not used for the purpose of performing tasks. Using an instruction is performing a task. So it can reasonably be concluded that the U.S. definition incorporates the tasks or functions of the program as well as its result. However, this question whether these tasks or functions are included or not raises another question. Indeed, are they not processes by which a program can run? And processes, in both systems, are excluded from protection by the idea/expression principle. In conclusion, even if they are mentioned in the European definition, they just serve the purpose of defining the program while not being protected elements.
In sum, while not phrased identically, the definitions reflect the same legal meaning in the United States and Europe. Despite their large coincidence, the notions remain extremely broad and, as a consequence, rather imprecise. This relative imprecision and the absence of a binding definition in Europe can cause two types of problems at the level of the courts. First, there can be a lack of uniformity between the interpretation in different Member States' courts. Secondly, a divergence between the U.S. and the European interpretation of a program can arise. The first section of Part 2 of this article will attempt to answer these problems.

Policy

The Green Paper definition, taken up by the Proposal does not really express either a copyright s.s. or a droit d'auteur perspective. It does not stress economic orientation nor does it reflect a typical "personalist" viewpoint. It is a rather neutral notion, compelled by the nature of a computer, which the definition merely describes. The same applies to the American definition. Although it can be alleged that the American definition influenced the European notion, it is unlikely to have any repercussions on the feasibility of case law application, considering its neutrality in policy. Reciprocally, American courts could use European cases on the point.

*13 The scope of protection and the exceptions

To determine the scope of protection, it is necessary to descend into the program's architecture and components to detect which of its elements are protectable. This operation is complex. Thus it is easier to proceed by elimination, i.e. to determine what the non-protectable elements consist of, then to deduce what the protectable elements are.

Text formulation: unprotectable elements

(1) Europe. The Green Paper is silent about unprotectable elements. This is rather surprising as it concerns the idea/expression dichotomy. The Directive and its Recital 14 state the idea/expression principle and define certain non- protectable elements--not in a detailed way, however.
Article 1.2 of the Directive asserts:
Protection in accordance with this Directive shall apply to the expression in any form of a computer program. 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.

Recital 14 of the Directive, relating to Article 1.2 states that "to the extent that logic, algorithms and programming languages comprise ideas and principles, those ideas and principles are not protected under this Directive".
(2) The United States. In the United States, the principle that ideas are not protected lies in section 102 (b) of the Copyright Act:
Ideas, procedures, processes, systems, methods of operation, concepts, principles or discoveries, regardless of the form in which it is described, explained, illustrated, or embodied in such work are not protected by copyright.
(3) Comparison. The U.S. exclusion seems more complete than in Europe. It excludes every kind of "idea or process", stating all the words akin to these terms, which the Directive omits to do. Nevertheless, Member States' laws or decisions might interpret broadly the exclusion and finally converge with all exceptions provided for in the Copyright Act. [FN55] Despite the relative inaccuracy of Article 1.2 of the Directive, Recital 14 asserts precisely what is not protected in relation to software: i.e. logic, algorithms and programming languages. While the Copyright Act does not mention them expressly, they can be considered as included in the general exclusion of section 102 (b).
As interpreted, both texts seem to cover the same exclusions. However, the Directive Recital is obscure because it is phrased in a less clear-cut way than section 102 (b). In fact, Recital 14 reads "to the extent that ...". Thus the Directive Recital could reasonably be construed as meaning that, if logic, algorithms and computer languages are not ideas or do not comprise ideas, they could be protectable. [FN56] This means, in practice, that this assumption is defensible in court. In the United States, on the contrary, as these concepts are categorised as ideas, once and for all, they cannot be arguable in court. Therefore, read literally, the European Directive is less exclusive than the Copyright Act. Considering the varied traditions of European judges, it is likely that one will see the development of a case law divided on the question of copyrightability of certain components of software. Moreover, a growing number of U.S. cases grant patent protection to algorithms and even to software embodied in diskettes, and the European Patent Office has followed this trend of software patentability. [FN57] In view of this case law, courts might be inclined to give copyright protection to ideas, as algorithms.
Nonetheless, the language used in Recital 14 of the Directive should be construed by European courts, as in the United States as excluding all types of "ideas and processes" (meant in the broadest sense). Several arguments support this opinion. First, although recitals intend to enlighten a directive text, the latter prevails over the former. Secondly, if the text is construed as including ideas, there will be a question as to "which ideas" can be protected in a computer program. Yet this question should not even be addressed, as no ideas can ever be copyrighted. Copyrighting ideas goes against the universal copyright s.l. principle of the exclusion of ideas, imposed to WTO Members by Article 9.2 of TRIPs. [FN58] This could mean that the Directive does not conform to TRIPs. If courts do not apply this principle to computer programs, there is a high risk that this tendency will invade the whole field of works protected by copyright law s.l. Consequently, it would destroy the delicate balance that copyright law s.l. seeks to preserve *14 between monopoly on expression and free circulation of ideas. [FN59]
Thirdly, the ambiguity in formulation can be resolved by construing the Directive in view of its legislative history. It was alleged that the Directive should be construed as including the now widely rejected U.S. Whelan [FN60] case. This case's ruling had the effect of protecting a lot of non-protectable elements. [FN61] As a matter of fact,

the Whelan case was the law in the United States at the time the Directive was adopted, and some have said that it influenced E.C. policy-makers. [FN62] This assumption must nevertheless be rejected. First, the Directive is not inspired by Whelan, because, despite its importance, none of the preparatory texts nor the Directive mentioned it. Secondly, at the same time, the Whelan case was highly criticised in the United States. Thirdly, if the framers did not explicitly define the ideas and principles, they intended to leave the question for the courts to decide because of the intense controversy on that issue. [FN63] Finally, to admit the inclusion of the Whelan ruling would go against the E.C.'s general desire to promote competition. [FN64] These four arguments plead for the non- influence of the Whelan case on the Directive. Courts must therefore be guided by the general exclusion common to both copyright s.s. and droit d'auteur systems and beware of protecting ideas embodied in program components. Still, if judges decide to construe the Directive as protecting some ideas through logic, algorithms and programming languages, they can depart from U.S. case law. They would, however, breach Article 9.2 of TRIPs.

Text formulation: protectable elements

Two types of program elements must be distinguished: literal and non-literal. Literal elements are composed of the object and source codes. "A program's non- literal elements encompass all aspects of the program other than the code. [U.S. c]ourts have defined non-literal elements to include the program's structure, sequence, organization, screen displays, general flow charts, and menu structures". [FN65]
(1) Europe. As opposed to the unprotectable elements, the Directive is rather explicit as to the protectable elements of a computer program. First, the definition of the program in Article 1.1 refers to preparatory design material. [FN66] Thus flow charts and all elements designed at previous stages of the conception of a program are within the scope of protection. Next, the Proposal and Recital 7 refer to codes as being protected. Recital 10 implicitly protects user interfaces. Finally, the Proposal also speaks of sub- programs, routines and modules as being protected. If all parts of these texts are combined, a rather comprehensive definition of literal and non-literal elements of a computer is forged.
The Proposal sentence pertaining to some non-literal elements reads as follows:
Sub-programs and routines, which go together to form modules, which in turn form programs, may all qualify for protection, independently of the protection given to the program as a whole, that is as a compilation of such elements. [FN67]
This sentence integrates in the definition of protectable elements a very important conception: computers can be protected under the compilation doctrine. This means that, the program, taken as a structure of its separate elements, can be granted a "compilation-type" copyright protection (if its structure reveals originality). In other words, it is irrelevant whether the elements, taken separately, are copyrightable or not. The protection applies to the structure, not to the elements taken separately. This doctrine can apply because the nature of a computer program fits into its conception. A program is truly nothing more than the arrangement of data that are designed to produce a result. This arrangement is materialised in a structure, composed of these many elements. The structure is therefore an integral part of the program.
(2) The United States. The U.S. Copyright Act is totally silent as to the protectable elements of a program. The absence of any description has led courts to rely only on section 102 (b) to deduce a contrario which elements were protectable. In so doing,

judges also used the traditional copyright s.s. doctrines of merger, scenes a faire and functionality applying to all copyrighted works and elaborated by them throughout the years. [FN68] As will be seen below, U.S. courts have been fairly hesitant in the application of the compilation doctrine to software. [FN69]
(3) Comparison. The Directive is more complete than the U.S. Copyright Act as to protectable elements. This might be one of the advantages of the Directive's longer maturation over the U.S. provisions pertaining to software. However, in both systems, courts have had to detail all protectable elements.
*15 On the compilation theory, the Directive is much more articulate than U.S. law, which does not provide for a specific provision on compilation applying to computer programs. [FN70] Whether the Directive was inspired by the American general compilation doctrine developed by case law as somewhat irrelevant, because this doctrine is inherent to both copyright s.s. and droit d'auteur systems from the start and for all works. Nearly all Member States protect compilations in their national laws. [FN71] Therefore it is just a matter of common sense to apply the "compilation" type protection to a computer structure.

Policy

The droit d'auteur/copyright issue as to the protectable and unprotectable elements is resolved quite easily. As mentioned earlier, both the E.U. and the United States share the same policy as to the scope of copyright s.l. protection: the idea/expression dichotomy. This principle excludes ideas and principles from protection so that they are free for anyone to use. Thus the copyrightability of the expressive elements can be deduced from the exclusion of the ideas. The copyrightable components are the ones remaining after having discovered the ideas and principles underlying the program. Therefore U.S. copyright law cannot have influenced Article 1.2 of the Directive because both texts are based on a prior common principle. If the contrary is alleged, it makes no difference as the policy remains the same. Accordingly, there is no barrier to the application of U.S. case law in the European Union and reciprocally.

The criterion of protection: originality

Text formulation

(1) Europe. Article 1.3 of the Directive states:
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.
The difference between the notions of originality in the Member States and the need for a uniform notion were considered in depth in the Green Paper. The Paper shows a detailed explanation of the situation of the originality requirement in the Member States and makes a review of European case law in order to try to come to a common acceptable definition of originality. [FN72] This reveals that the Commission did not refer to American law, but focused on all definitions of originality in the different Member States. The Directive defines originality as "the author's own intellectual creation". This definition is nonetheless subject to diverse interpretations. Authors diverge as to the orientation given in the Directive to this standard. Some say this concept of originality comes from the Anglo-Saxon concept; others argue it was inspired by the continental notion. In fact, this definition can mean "originating from"

the author as opposed to originating from an infringer (British perspective), or it can mean "personal to the author" (civil law perspective). Another element can induce a continental inspiration. The preference for the word "creation" instead of "effort" could indicate the choice for a higher level than the British standard. [FN73] In any case, it is agreed that this notion is a compromise between the very high level of originality required in Germany and the very low threshold accepted in the United Kingdom. [FN74] To keep the concept of originality close to a single notion in all Member States in order to achieve uniformity, the Commission insisted that the level of originality had to be weak so that a lot of programs could be copyrightable. [FN75] To conclude, the definition remains vague and provides room for interpretation by the Member States. [FN76] Indeed, between high and low there is a wide range of possible levels. One has to wait for a decision of the European Court of Justice to reveal what the framers intended by originality.

(2) The United States. Originality is not defined in the Copyright Act. Section 101, however, mentions it in the definition of "compilation" and "derivative work". [FN77] Courts have always applied a low threshold of originality. They once were close to the British "skill, judgment and labour" doctrine with their "sweat of the brow" standard of originality. The 1991 Feist case rejected that standard:

Original means only that the work was independently created by the author, (as opposed to copied from others *16 works) and that it possesses at least some minimal degree of creativity. [FN78]

(3) Comparison. The first part of the Feist standard--"the work was independently created by the author"--reflects the traditional Anglo-Saxon view of originality. The second part of the Feist standard--"it possesses at least some minimal degree of creativity"--is along the same lines as the European Directive standard if "creativity" is interpreted as "personal to the author". Personality expresses itself through creativity. Creativity, in the computer programming field, has to do with choices and exclusions of elements that the programmer is led to do while programming, whereby one programmer chooses differently from another because they have different personalities. It can even be argued that, with such a definition, American law grows away from the British standard and is closer to a personalist continental view of originality because it adds a condition (a minimal degree of creativity). This latter concept of originality was nevertheless well-rooted in U.S. copyright law for more than a century. [FN79] In Feist, the Supreme Court seized the opportunity to overrule the "sweat of the brow" line of precedents and to clear things on the point, by reasserting the standard of creativity.

Policy

Quite clearly, the originality criterion is a point where the policy choices pervade the text formulation. This requirement reveals the different conceptions of Anglo-Saxon and continental law on the required degree of originality. Continental laws generally require a higher degree of originality, i.e. the work must be personal to the author. In traditional Anglo-Saxon law, the personality is not as important and, consequently, only effort or skill will be sufficient. However, the American standard since Feist has evolved towards a more continental view. American and European policies have thus converged. European courts can then follow American decisions as long as the Supreme Court does not overrule Feist. In like manner, American jurisdictions can search for inspiration in European rulings on originality.

Conclusion

The texts of both systems mainly converge but their common flaw is to remain either general (idea/expression rule) or very broad (definition of program and originality), which makes them rather vague and therefore subject to many possible interpretations. Thus, although the legislative and policy choices are analogous, only case law can clarify their profound meaning.
Part 2 of this article will be published in the next issue of E.I.P.R.

Note 1. The author swishes to thank Professors Franklin Dehousse, Peter Maggs and Alain Strowel for their helpful comments on an earlier draft of this article.

[FN1]. The terms "Europe", "the European Community", "the E.C.", "the Community", "the European Union" and the "E.U." will be used interchangeably throughout the article for the same concept.

[FN2]. Directive 91/250 on the legal protection of computer programs [1991] O.J. L122/42. The term "Directive" in this article refers to the European Software Directive, unless otherwise stated.

[FN3]. 17 U.S.C. §§ 101-1101, Public Law 94-553, 90 Stat. 2541, as amended up to July 31, 1996 ("the Copyright Act of 1976"). The references to the American Copyright Act in this article will be to this version.

[FN4]. Congress revised the Copyright Act in 1980 to include the CONTU (National Commission on New Technological Uses of Copyrighted Works) report conclusions which advised the inclusion of software in copyrightable works. The CONTU Commission was established by Congress in 1974 to consider the copyright issues raised by photocopying and computer technologies. The CONTU issued its report in 1978 (the "CONTU Report") and concluded, among other things, that computer programs were copyrightable. See R. S. Brown and R. C. Denicola, Cases on Copyright, unfair competition and other topics bearing on the protection of literary, musical and artistic works (6th ed., 1995), pp. 101 and 135.

[FN5]. Art. 36 was the only disposition of the Treaty which could possibly include copyright. See J. Huet and J. C. Ginsburg, "Computer programs in Europe: a comparative analysis of the EC Software Directive" (1992) 30 Colum. J. Transnat'l L. 327 at 329. For the cases in which the Court of Justice stated that copyright was comprised inside the scope of the Treaty, see, e.g., Case C- 78/70, Deutsche Grammophon [1971] E.C.R. 1-487; Case C-62/79 Coditel v. Cine Vog Films [1980] E.C.R. 1-881; Cases C-55 and 57/80 Musik-Vertrieb Membran GmbH and K-Tel Int'l v. GEMA [1981] E.C.R. 1-147.

[FN6]. "The Directive was elaborated in the wake of the White Paper of 1985 on the internal market [COM (85) 310 final], which in turn presaged the adoption of the Single European Act of 1986 [[1987] O.J. L169/1] ...". See Huet and Ginsburg, n. 5 above, at 329.

[FN7]. Communication from the Commission, Green Paper on Copyright and the Challenge of New Technology--Copyright Issues requiring immediate action, COM (88) 172 final, June 7, 1988, at 170.

[FN8]. See A. Strowel and J.-P. Triaille, Le droit d'auteur, du logiciel au multimedia (1997) p. 138. Several industry groups holding divergent interests lobbied intensely on this issue.

[FN9]. See below.

[FN10]. Some states protected programs by copyright in their legislation, others by case law, but in general the protection was not clear. See Recital 1 of the Directive.

[FN11]. The length of protection and the standard of originality were very different among the Member States. This created barriers to the circulation of works. This has occurred in some cases, for these cases, see infra note 12. For example, if a work had fallen in the public domain in Member State A and an importer from that Member State A wanted to import in Member State B where it was still protected, the rightholder in Member State B could block the importation.

[FN12]. The need for harmonisation arose in 1985; see COM (85) 310 final, June 28-29, 1985, COM (88) 172 final, June 7, 1988 and COM (88) 816 final--SYN 183 [1989] O.J. C91/9, at 5, point 1.4. This is recalled in Recitals 4 and 5 of the Directive. The Community decided to harmonise some aspects of copyright law in view of the problems arising in the following cases: Case C-78/70, Deutsche Grammophon, n. 5 above; Case C-158/86 Warner Brothers & Metronome v. Christiaensert [1988] E.C.R. 1-2625; Case C-341/87 EMI Electrola GmbH v. Patricia Im- und Export [1989] E.C.R. 1-79.

[FN13]. There had been another previous attempt in legislating over I.P. rights, but for a sui generis protection; see Council Directive 87/54 on the Legal protection of Topographies of Semiconductor products [1987] O.J. L24/36.

[FN14]. COM (88) 816 final--SYN 183 [1989] O.J. C91/9, at 5, point 1.3; Recital 3 of the Directive. See also Strowel and Triaille, n. 8 above, p. 137. Harmonisation was not only aimed at promoting the free movement of software but also at creating conditions allowing the industry to benefit from the common market. See COM (88) 816 final--SYN 183 [1989] O.J. C91/9, at 5, point 1.4.

[FN15]. See Recital 2 of the Directive. It must nonetheless be noted that this statement is not completely true. One needs to make a distinction between two types of products: worldwide selling products (i.e. Microsoft spreadsheet program "Excel") and those marketable only or mainly in the E.U. (a Dutch spelling checker). Poor copyright protection in the E.U. will indeed discourage investment in the Dutch spelling checker. On the other hand, the maker of the spreadsheet is interested in selling worldwide and in minimising the costs of writing the program. These sales will be the same whether the program is written in Belgium or the United States. But if Belgiums can program it better and more cheaply, the company will have the program written in Belgium to minimise costs, even if Belgium has poor copyright protection.

[FN16]. For a full demonstration, see P. B. Arenas, "Implementation, compliance and enforcement: the European Community Directive for the legal protection of computer software" (1992) 5 Transnat'l Law 803 at 809-811

[FN17]. See Recital 27 of the Directive.

[FN18]. See Huet and Ginsburg, n. 5 above, at 331. See also Arenas, n. 16 above, at 810. The concern regarding a workable competition was aimed at promoting international standardisation in the field of information and communication technologies and promoting open systems, allowing interconnection and compatibility between programs and equipment (See F. Brison and J.-P. Triaille, "La directive C.E.E. du 14 mai 1991 et la protection juridique des programmes d'ordinateur en droit belge" (1991) Journal des Tribunaux 782).

[FN19]. Patent law does not generally protect algorithms, and software can only be patented if integrated into a patentable invention, but not individually. See Munich Convention on the European Patent, October 10, 1973 revised December 17, 1991, Art. 52. Contractual law was considered insufficient to fight against forgery and infringement. See also Arenas, n. 16 above, at 812.

[FN20]. COM (88) 816 final--SYN 183 [1989] O.J. C91/7, point 3.6. See also H. C. Jeroham, '"The EC Copyright Directive, Economics and Authors' Rights" (1994) 25/6 International Review of Industrial Property and Copyright Law 821 at 839; Brison and Triaille, n. 18 above, at 782; L. J. Raskind, "Symposium: the impact of European integration on intellectual properties: protecting computer software in the European Economic Community: the innovative new directive" (1992) 18 Brook. J. Int'l L. 729: By choosing copyright law, "the Directive continues the position taken in the earlier Green Paper and follows the pattern of the United States and most member states of the Community".

[FN21]. P. Samuelson, "Symposium on U.S.-E.C. legal relations, Comparing U.S. and E.C. copyright protection for computer programs: are they more different than they seem?" (1994) 13 J.L. & Com. 279, fn. 4.

[FN22]. As to the choice of protection, the Green Paper affirms its election of copyright s.l. Nonetheless, it also asserts that there remains some uncertainty about that choice which can only be clarified by a more detailed implementation of the Directive into national law and by the superior courts' interpretation. (See COM (88) 172 final, June 7, 1988, point 5.6.2. and 5.6.29.) This uncertainty of language obscures the Commission's intentions. A. Francon, "Conclusions", in Copyright and the Europeart Community, the Green Paper on Copyright and the Challenge of New Technology (F. Gotzen ed., 1989), pp. 88-89.

[FN23]. COM (88) 816 final--SYN 183 [1989] O.J. C91/8, point 4.

[FN24]. Communication from the Commission, Green Paper on Copyright and the Challenge of New Technology--Copyright Issues requiring immediate action, COM (88) 172 final, June 7, 1988, at 177, point 5.3.5. Indeed, if a sui generis protection had been chosen, programs would not have been protected by the Berne Convention.

[FN25]. Neither the title nor the subheading of the Green Paper expounds on which legal ground it is founded to act in literary ownership. The preliminary chapter considers the subject but does not address it very precisely.

[FN26]. chap. 1, 1-1 to 1-3.

[FN27]. chap. 1, 1-4.

[FN28]. s.107 of the Copyright Act states a very broad exception applying to all works: "the fair use of copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by that section, for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship or research, is not an infringement of copyright." s.107 further sets factors to help determine in any particular case if there is fair use (the purpose and character of the use, the nature of the work, the amount of the portion of the work used and the effect of the use on the potential market for the work).

[FN29]. Samuelson, n. 21 above, at 285 and fn. 31 (see International Copyright Law and Practice (Paul E. Geller ed., 1992)). It must, however, be remembered that there is a similar notion in the United Kingdom known as "fair dealing". But this notion has not been integrated into the Directive.

[FN30]. Decompilation is addressed by Art. 6 of the Directive. As Pamela Samuelson explains it, "in line with the civil law tradition from which it largely sprang, the Directive is grounded on the assumption that all uses of programs for which rules are needed have been addressed in the Directive. Thus, the scope of permitted uses under the Directive is likely to be confined largely to that which has been expressly authorized. Moreover, the Directive intends to forbid Member States from using fair use doctrines (or the like) to expand certain privileges beyond that which the Directive established." See Samuelson, n. 21 above, at 285.

[FN31]. Art. 2.3 of the Directive. cf. s.101 (definition of works made for hire) and s.201 of the U.S. Copyright Act.

[FN32]. See Samuelson, n. 21 above, at 285. She points out the European exceptions to authors' rights, which are very detailed, while in the United States the fair use exception is general and applies for all works.

[FN33]. See, e.g. Brison and Triaille, n. 18 above, at 791. M. Moller, "Author's right or copyright?", in Copyright and the European Community, n. 22 above, pp. 11, 13, 17. H. C. Jeroham, "Harmonizing Intellectual Property Law within the European Union" (1992) 23/5 International Review of Industrial Property and Copyright Law 622, at 627, citing Schricker, "Zur Harmonisierung des Urheberrechts in der Europaischen Wirtschaftsge-meinschaft," in Festschrift fur Ernst Steindorf (1990), p. 1437, and H.C. Jeroham, "Inleiding en epiloog tot het Groenboek van de Europese Commissie over Auteursrecht" (1989) 13 Informatier-echt/AMI 3. Raskind, n. 20 above, at 735.

[FN34]. T. Dreier believes, on the contrary, that the definition of originality is somewhat current on the continent. See T. Dreier, "The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs" [1991] E.I.P.R. 320. See also A. Strowel, Droit d'auteur et copyright, Divergences st convergences (1993), pp. 57-58.

[FN35]. See Moller, n. 33 above, at 17, 1.5.1. and Brison and Triaille, n. 18 above, at 791: "The principle under which the author's rights are granted to the author's employer reflects the adoption of a 'copyright' perspective".

[FN36]. She believes the Commission should have clearly stated its choice of copyright s.s. and should not have mixed droit d'auteur and copyright principles; Moller, n. 33 above, at 17, 1.5.1.

[FN37]. Art. 1.1. of the Directive and 17 U.S.C. § 101 (1996) (defining "literary works").

[FN38]. Source code is written in a choice of several languages (Pascal, Cobol, Fortran, C, C++, Scheme, Java, etc.) Source code needs to be compiled in object code in order for the computer to perform its functions. See also TRIPs, Art. 10.1, which compels signatories to classify programs into literary works and protect the object as well as the source codes.

[FN39]. This springs directly from the explanatory memorandum of the Directive. COM (88) 816 final--SYN 183 [1989] O.J. C91/5, point 1.1. See also Strowel and Triaille, n. 8 above, p. 145; Samuelson, n. 21 above, at 282 and fn. 11

[FN40]. COM (88) 816 final--SYN 183 [1989] O.J. C91/9.

[FN41]. Firmware is software that is stored on a chip rather than on punched cards, a diskette, a hard disk or tape.

[FN42]. See Green Paper, n. 7 above, at 5.1.1, note 1.

[FN43]. Indeed s.102 (a) does not mention programs in the list of protected works, although the list is not exhaustive.

[FN44]. If it was not copyrightable, there would not be a need for the s.117 exceptions.

[FN45]. 714 F. 2d 1240 (3d Cir. 1983), cert. dism., 464 U.S. 1033 (1984).

[FN46]. This is compelled by the majority expressed in the CONTU Report. See CONTU Report, n. 4 above, at 21.

[FN47]. Art. 10.1 TRIPs.

[FN48]. M. P. Larvick, "Questioning the necessity of copyright protection for software interfaces" (1994) U. Ill. L. Rev. 187 at 190.

[FN49]. "The program's menus present the user with a list of choices that instruct the computer to perform specific tasks, such as printing or saving a document to disk", ibid., at 190. See also D. Bainbridge, Software copyright law (1994).

[FN50]. They will be copyright-protected to the extent they have no function other than to inform or portray an appearance. Functionality is the realm of patent law protection while copyright law protects non-functional works. This is the basis of the limit between copyright and patent protections. See D. S. Karjala, "The Relative Roles of Patent and Copyright in the Protection of Computer Programs" (1998) 17 J. Marshall J. Computer & Info. L. 41 at 46-47, 70.

[FN51]. Karjala, ibid., at 52 and 55.

[FN52]. Karjala, ibid., at 55. In fact, this was already mentioned, but in a less clear-cut way, in D. S. Karjala, "Recent United States and International Developments in Software Protection" [1994] E.I.P.R. 13-20 and 58-66 at 17. The author said that the scope of protection of computer programs should be tied closely to the instructions or statements, so only to object and source codes; as to interfaces, they should be excluded from the "program protection", i.e. should not be analysed as "non-literal" elements of the program.

[FN53]. Karjala, n. 50 above, at 52.

[FN54]. UIs have been generally considered by courts to be non-literal elements of a program. For an opposite opinion, see Karjala, n. 50 above, at 55.

[FN55]. See Samuelson, n. 21 above, at 299.

[FN56]. Dreier is of that opinion: "Cautiously, the directive does not exclude logic, algorithms and programming languages from copyright protection per se, but only to that extent that they comprise ideas and principles." Dreier, n. 34 above, at 320 (and fn. 18) and T. Dreier, "La Directive du Conseil des Communautes europeennes du 14 mai 1991 concernant la protection juridique des programmes d'ordinateur" (1991) Juris Classeur Periodique 352. See also Strowel and Triaille, n. 8 above, 153.

[FN57]. For the United States see the new Examination Guidelines for computer-related inventions, 61 Fed. Reg. 7478 (1996) and for the cases, e.g. Diamond v. Diehr 450 U.S. 175 (1981); In re Abele 684 F. 2d 902; Arrythmia Research Technology, Inc. v. Corazonix Corp. 958 F. 2d 1053 (Fed. Cir. 1992); In re Alappat 33 F. 3d 1526 (Fed. Cir. 1994) and In re Beauregard 53 F. 3d 1583, 1584 (Fed. Cir. 1995). The EPO admits patentability for computer programs if the software is integrated in an industrial process and provides a technical solution to a problem, see, e.g., Vicom, O.J. EPO 1987, 14; IBM, O.J. EPO 1990, 5, IBM/Editable document form, O.J. EPO 1994, 557. For a discussion about the protectability of software by patents, see Part 2 of this article, in the next issue of E.I.P.R. In Europe, the newly issued Communication on a project of a Directive Proposal on the harmonisation of patentability requirements for computer programs is an attempt at regularisation of the possibility of patenting software. See Communication of the Commission, project of a Directive Proposal on the harmonisation of patentability requirements for computer

programs, COM (99) 42 final. http://
www.europa.eu.int/comm/dg15/en/intprop/indprop/99.htm.

[FN58]. Art. 9.2. reads: "Copyright protection shall extend to expression and not to ideas, procedures, methods of operation or mathematical concepts as such." This provision is much closer to the U.S. definition than to the European exclusion.

[FN59]. Indeed, parties should seek patent protection for the new ideas or processes within their programs, not copyright. Patent and copyright protection are complementary. Patents will protect things that copyright does not, and conversely. Moreover, copyright is a good means of protection against slavish copying, while patents are targeted at protecting against a competitor's "misappropriating" technology.

[FN60]. Whelan Assocs., Inc. v. Jaslow Dental Lab. 797 F. 2d 1222, 1238 (3d Cir. 1986), cert. denied, 479 U.S. 1031 (1987).

[FN61]. Samuelson, n. 21 above, at 293.

[FN62]. Samuelson, ibid., at 293.

[FN63]. This consideration is more sound and is more than suggested in the Green Paper at 5.5.12. See also Samuelson, ibid., at 295.

[FN64]. See Recital 27 of the Directive and the first section of this article.

[FN65]. These elements are similarly protected in Europe. See Larvick, n. 48 above, at 189. See also the definition given by the Altai court in Part 2 of this article.

[FN66]. See "The definition of computer program", above.

[FN67]. COM (88) 816 final--SYN 183 [1989] O.J. C91/9.

[FN68]. Here is an overview of the traditional copyright doctrines:
(1) Elements dictated by efficiency (merger doctrine). When specific instructions are the only and essential means of accomplishing a given task, their use by another will not amount to infringement. If certain tasks can only be done by a very limited number of structures, these are not protected (see Strowel and Triaille, n. 8 above, p. 157 and fn. 65.)
(2) Elements dictated by external factors (scenes a faire doctrine). This doctrine denies protection to those expressions that are standard, stock, or common to a particular topic, as well as well-known industry techniques or that necessarily follow from a common theme or setting. It also excludes those elements of a program that have been dictated by external factors.

[FN69]. See Part 2 of this article, to be published in the next issue of E.I.P.R.

[FN70]. See 17 U.S.C. § 101 (defining "compilations").

[FN71]. Indeed, if the Directive did not afford protection to some computer programs, plaintiffs could still turn to other kinds of protection in the European States such as the general laws on copyright, which protect compilations. Moreover, the Directive leaves room for protection of programs by other types of legal protection (Recital 26). For an example of the protection of computer programs by compilations, see J. Gyngell, "Compilations of Computer Programs: Total Information processing Systems Ltd v. Daman Ltd" [1992] E.I.P.R. 95. It is worth mentioning a British case, however, decided before the implementation of the E.C. Directive, which applies the compilation doctrine to computer programs. The British judges were confronted with the argument that the linking of three application programs was a compilation and therefore a new computer program in itself. The judge decided there was no copyright in this compilation because the arrangement and selection of the programs was purely functional and (applying the originality requirement under British law) there was no evidence of skill and labour in compiling them. For an opposite decision see, e.g., Ibcos Computers Ltd v. Barclays Mercantile Highland Finance Ltd, Chancery Division [1994] F.S.R. 275.

[FN72]. See the explanatory memorandum of the 1989 Directive Proposal.

[FN73]. The British standard of originality provides for "effort, skill and labour" only.

[FN74]. This is illustrated by the second sentence of Art. 1.3. See, e.g., A. Junker, "Die Entwicklung des Computerreschts in den Jahren 1991 und 1992" (1993) 13 New Juristische Wocherischrift 824 at 824-826. Dreier, n. 2 above, at 320: "in view of the legislative history..., it should be clear that the Directive leaves no further room for an exceedingly high originality standard such as the German one." This is recalled in Recital 8 of the Directive which states: "Whereas, in respect of the criteria to be applied in determining whether or not a computer program is an original work, no tests as to the qualitative or aesthetic merits of the program should be applied."

[FN75]. Answer to a parliamentary question, October 6, 1993 [1993] O.J. C350.

[FN76]. Strowel and Triaille, n. 8 above, p. 150. See also Arenas, n. 16 above, at 836-837.

[FN77]. s. 101 states that compilations and derivative works must constitute an "original work of authorship"; see § 101 (1996) (defining "compilations" and "derivative works").

[FN78]. Feist Publications, Inc. v. Rural Telephone Service Co. Inc. 111 S.Ct 1282, 18 U.S.P.Q. 2d 1275 (1991). This definition is general and applicable to all works.

[FN79]. See Burrow-Giles Lithographic, Co. v. Sarony 111 U.S. 53 (1884).
EIPR 2000, 22(1), 7-16

END OF DOCUMENT

European Intellectual Property Review

2000

Article

SOFTWARE COPYRIGHT PROTECTION: CAN EUROPE LEARN FROM AMERICAN CASE LAW? PART 2

Estelle Derclaye.

Keywords: Copyright; European Union; Originality; Software; United States
Abstract: Different approaches taken by US and EU case law relating to definition of computer programs, copyrightability and originality, and suggested approach to help European judges decide whether to use American precedents.

*56 Part I of this article (published in the previous issue) examined the contextual background of the adoption of the Software Directive, and the diverse influences on its text as to the choice of protection of software. Part 1 next showed that there was no evidence of a clear choice between the two systems of droit d'auteur or "copyright s.s.". The conclusion was that the Directive was influenced by a mix of both systems. The subsequent statutory analysis showed virtually no differences in the formulation of the American and European legal texts, except the potential protection of logic, algorithms and programming languages that a certain interpretation of the Directive could allow. It was demonstrated that this option was not to be followed, mainly because it would not conform to the idea/expression principle stated in the Directive and would also be contrary to Article 9.2 of TRIPs. There were no differences in policies. In conclusion, while no divergence was to be found, the relative inaccuracy of both texts has entrusted the courts with the mission to clarify their meaning. Part 2 thus analyses and comments on American and European case law and suggests a test to help European judges determine whether they should use American precedents.

Case Analysis

Program definition cases

The United States

Since the 1980 Amendment of the Copyright Act, there have not been many decisions concerning the definition of a computer program. [FN1] However, one case resolved in itself many questions that had been shelved. In the 1984 case Apple v. Franklin,

[FN2] the Third Circuit reaffirmed that source and object codes are protected because they are comprised in the words "directly and indirectly" of the program definition, as discussed in Part 1. Secondly the court stated that both operating and application programs are protectable for two reasons. First, both types of programs instruct the machine to do something. Secondly, the section 101 definition of "computer program" does not make any difference between operating programs and application programs so that they both fit in the program definition. At first sight, this latter issue seems to have no importance any more because it is now generally accepted that copyright law protects all (current) types of computer programs. Still, Apple v. Franklin provides an important precedent: the computer program definition does not imply any difference in the kinds of programs. In the future, this precedent could be used to include in the definition new types of software now unknown.
Thirdly, the court reminded that only the instructions (expression) are protected and not the method (idea) that instructs the computer to perform its operating functions. This follows directly from the computer program definition of section 101, which only treats of "statements or instructions" and from section 102 (b) which excludes processes from protection. In Computer Associates v. Altai, [FN3] this finding was rephrased. A computer program must be viewed both as text and as behaviour, where the texts are the source and object codes and where the behaviour is what the computer does when the user types and uses commands. [FN4] The behaviour of a program falls within the exclusion of section 102 (b) as a "process", "system" or "method of operation". These cases interpreted the computer program definition in an adequate way.

Europe

Similarly, there are not many European cases on the issue of program definition. One reason might well be that the Directive Proposal definition until now solved all problems concerning the definition. Belgian courts found object and source codes as well as operating systems and application programs copyrightable. [FN5] A French court correctly applied the Directive by deciding that the preparatory design material is protected by copyright law. [FN6] Surprisingly, another French court ruled that only the source code could be afforded protection. [FN7] This decision seems isolated and does not appear to reflect French jurisprudence. It nevertheless *57 shows, several years after the Directive's entry into force, that judges are still not familiar with the concept of a program.
As introduced in the statutory analysis, although definitions mostly coincide, their inaccuracy can cause two types of problems. First, there can be a lack of uniformity in the interpretation given by different Member States' courts. Secondly, a divergence between the U.S. and the European interpretations of a program can arise.
First, it is likely that European judges will endorse different conceptions of computer programs. In this case, uniformity will not be achieved except by the Court of Justice and after a rather long period of legal "insecurity". Until now the interpretation has been rather uniform. The lack of a binding definition should induce the Member States' judiciaries first to construe the concept of computer program broadly and, above all, to look at other Member States' decisions in order to come to a common interpretation.
Secondly, differences between the U.S. and European courts on the interpretation of a program can arise from the lack of a binding definition in Europe, or in other words from the United States' more limited definition. This freedom makes it possible for

European states to protect software or elements of software that would not be protected under U.S. law. Until now, this problem has not occurred but it remains potential. A situation in which this problem could have arisen, but has not, is illustrated by the status of the preparatory design work (such as flow charts) and firmware. In Europe, they are expressly cited in the definition and thus protected, whereas in the United States they are not mentioned. However, firmware falls within the U.S. definition in section 101 because the medium on which the software is embodied is of no significance for protection. [FN8] Moreover, several U.S. courts have recognised the copyrightability of firmware. [FN9] The potential difference in protection as regards flow charts has also been eliminated since the courts decided that non-literal elements of a computer program were protectable. [FN10] Contrasted interpretations in the U.S. and the E.U. can also arise from the vagueness of their definitions. Some programs would be protected in the United States and not in Europe, and vice versa (for instance, new types of software now unknown would not be comprised in one or the other definition). Computer firms could try to find another kind of protection in another legal system for their "new types of software". [FN11] Indeed, national treatment proves to be insecure. [FN12] Another solution would be for Europe and the U.S. to conclude a reciprocity agreement. Better, a common concept of software should be defined in an international arena, in the prospect of a future revision of the Berne Convention and of TRIPs. [FN13] Even if programs are subject to very fast technological changes, the problem of the definition should not be avoided. This is a necessary objective as computer programs gain weight in commerce.
In sum, in the E.U. and the United States, the program definition has not yet engendered problems. It remains to be seen in the future whether with the advent of new types of software on the market, the notions will not appear to be too narrow or ambiguous.

Copyrightability cases

The United States

Two types of decisions will be discussed in this section. The first will focus on infringement of "internal" elements (literal and non-literal) and the second will address infringement of computer user interfaces ("UIs"). These cases are analysed separately because the nature of their elements is substantially different. [FN14] The first series of cases concern internal components, unintelligible to the lay eye, written initially in organigrams and then in a computer language. These have been classified as literary works because they are written in a certain language. These are called "internal" because the user does not see them as he uses the program. The second series of cases involve UIs, the external features, which are visible to the user. UIs are referred to as pictorial works. This distinction between decisions involving internal components and UIs is important as the tests of copyrightability to be used in the two types of cases are different. Courts addressing the internal components issues have made another distinction between cases of non-literal infringement and *58 cases of literal infringement. The relevance of this latter distinction has recently been challenged, as will be further developed. [FN15]

(1) Infringement of internal elements

(a) Non-literal infringement. It must initially be noted that the two types of copying (literal and non-literal infringement) should not be confused with the literal and non-literal elements of a program. "Literal copying denotes verbatim [slavish] copying of all or part of the source code or object code of a computer program." [FN16] Non-literal copying occurs when "the fundamental essence or structure of one work is duplicated" without reproduction at code level. [FN17]

The Second Circuit decision in Computer Associates v. Altai [FN18] was the first to establish a means of determining which non-literal elements of a program can be copyrighted. [FN19] The Circuit court, in reaching this conclusion, applied the test developed for literary works, because the Copyright Act classified computer programs into literary works. The court endorses the following syllogism: "if the non-literal structures of literary works are protected by copyright and if computer programs are literary works, as we are told by the legislature, then the non-literal structures of computer programs are protected by copyright". [FN20] Consequently, the scope of protection of these structures needs to be determined. For that purpose, the court defines non-literal and literal elements. Non-literal elements are (1) program ultimate function or purpose; (2) modules; (3) organisational or flow charts; (4) parameter lists; (5) structure; and (6) macros. [FN21] This determination of the non-literal elements by the courts was important in the United States as the Copyright Act did not address this issue. The literal elements are the translation of the non-literal elements into proper source code.

The Altai court sets forth its "substantial similarity" test for computer program structure, relying on the long-lasting precedents of Nichols v. Universal Picture Corp. and Sheldon v. Metro Goldwyn Pictures Corp., [FN22] decided in literary works. In Nichols, the Second Circuit determined the different degrees of abstraction to find the limit of copyrightability where the expression and the idea meet. The Altai "abstraction-filtration-comparison test" ("AFC test" or "Altai test") consisting of three steps, has the same goal. In the abstraction stage, the court should separate the program into its component parts or in other words dissect the allegedly copied program structure and isolate each level of abstraction contained within it. The court should then proceed from the specific to the general, or, phrased differently, begin with the code and conclude with the program's ultimate function. [FN23] The second factor involves filtration. The court should examine the structural components for each level of abstraction and determine whether the particular inclusion at that level was an idea, was dictated by considerations of efficiency, required by external factors or taken from the public domain. [FN24] This procedure severs the unprotected elements from the protected elements and serves the purpose of defining the scope of plaintiff's copyright. Subsequent to completing filtration, the court is left with the program's kernel of protected expression. In a final step, the court compares this "golden nugget" with the alleged infringing program to determine if there is substantial similarity between the two and decides whether there is infringement.

In summary, the abstraction and filtration steps are tools to do the spadework on the elements and to identify ideas and expressions. Once this is undertaken, a court may apply the substantial similarity standard to the remaining elements conveying expression. The Altai court was nevertheless aware that such a test would have to be adapted to the particular circumstances of each case. [FN25] The value of this test lies in its clarity and easy applicability to software internal components. Furthermore, it carries jurisprudential weight because it is based on the well-accepted Nichols and Sheldon decisions. The Altai decision was also decided by a circuit respected for its decisions in the field of copyright law. These reasons explain why the Altai *59

decision has influenced the subsequential development of software copyright case law. [FN26]

The reasoning of the court in CA v. Altai was further defined by the Tenth Circuit in Gates Rubber v. Bando Chemical Industries. [FN27] The Court of Appeals reconsiders the levels of abstraction set forth by the Second Circuit and the several elements composing them. [FN28] Commencing with abstraction, the court identifies--at least--six levels of declining abstraction that can be found in a computer program: (1) the main purpose; (2) the program structure or architecture; (3) modules; (4) algorithms and date structures; (5) source code; and (6) object code. [FN29]

In the filtration step, the court identifies six different types of unprotectable "data" that can be found within the abstraction levels: ideas, processes, facts, public domain, "merger" and scenes a faire. This helpful description gives some guidelines as to which elements will generally not be protectable. The main purpose or function of a program will always be an unprotectable idea. Likewise, as each module may typically be described by its individual purpose or function, a module's basic function or purpose will nearly always be an unprotectable idea or process. On the contrary, source and object codes will almost always be found protectable. The reason is that a programmer proceeds from the general to the specific, or in other words, from the main idea to the expression of this idea in more complex writing. As for the intermediate levels of abstraction, such as structure, sequence, organisation and the like, it is more difficult to generally predict their protectability or unprotectability. It will depend on each case's facts. The court provides a guide in which it describes in which components, ideas, processes, facts and scenes a faire can generally be found. [FN30] It adds that, while the structure of one program may be unprotectable because it constitutes an idea, the organisation and arrangement of another program may be expressive. [FN31] This introduces the possibility of protecting the structure of a program by the compilation doctrine.

Comparing, for the court, is qualitative rather than quantitative. [FN32] In its opinion, both programs should be looked at in their entirety before being abstracted and filtrated. The underlying justification is to prevent courts from concentrating on particular elements without having a general overall picture of the program itself. However, such a preliminary step does not obviate the ultimate need to compare the program's protected elements with the alleged infringing program components. [FN33] The court concludes with a final remark expounding the Altai's court reserve: the appropriate test to be applied in each case and the order in which the various program components are to be examined may vary depending on the claims involved, the procedural posture of the suit and the nature of the computer programs at issue. [FN34]

The determination of the six levels of abstraction and the adaptation principle have had success in further litigation. [FN35] The Harbor Software v. Applied Systems case is a good example of the typical application of the Altai test revisited in Gates, but it does not add more to Gates's elaboration of the test. [FN36] The court in Engineering Dynamics, Inc. v. Structural Software, Inc. also mentions the idea of the application of the compilation doctrine to the program structure. [FN37]

(b) Literal infringement. Prior to Bateman v. Mnemonics, the application of the AFC test to literal infringement cases was controversial. [FN38] In Bateman, the Eleventh Circuit court faced the issue of literal copying of a *60 program. [FN39] The defendant appealed because, during the examination of "substantial similarity", the District Court's jury jurisdiction limited the AFC test to non-literal copying instances. As a result, "filtration", the most important step of the test, was not applied to the

literally copied part. This implied in the jury's mind that these elements are not filtered because they are always copyrightable.
Initially, the AFC test was created by the Second Circuit to resolve non- literal infringement cases. The Gates Rubber court " 'simply chose' to include the literal question in the filtration analysis ... ". [FN40] The Eleventh Circuit in Bateman follows the Gates court in his choice and asserts that some type of analysis has to be done in instances of literal copying, whatever it is called (filtration, merger doctrine, etc.). If this is not done, some unprotectable elements will unjustifiably be deemed copyrighted. There is no basis to discriminate between the type of copying made by the defendant. Rather logically, the copyrightability of elements in the copied program does not depend on the type of copying. In sum, this case challenges the relevance of the courts' classification (in their application of the AFC test) between cases of literal and non-literal infringement.
In conclusion, the AFC test is now accepted in virtually all circuits. It is even used to test copyrightability in cases not involving computer programs. [FN41] Its success is due to its structured line of argument and its relative flexibility in application. Nonetheless, the application of the AFC test to user interfaces has been more complex and controversial.

(2) Infringement of computer user interfaces

The five cases dealing with UIs can be approached in many different angles. [FN42] Only the angle of the courts' reasoning is relevant to the argument. What European judges need to know is the most suitable way, in other words the most correct legal reasoning, to judge "UI cases". If American judges use a correct reasoning to deal with UIs, nothing should hinder European judges from using it too.
The methods used by the courts can be classified into two categories: application of the AFC test and application of section 102 (b). Three courts decided to apply the AFC test to UIs. In Engineering Dynamics, Inc., v. Structural Software, Inc., [FN43] the plaintiff sought to copyright its input and output formats as a whole. Accordingly, the Fifth Circuit adapts the AFC test and applies it only at this level of abstraction without entering into the singular elements of each format. At the filtration stage, it asserts that the criterion to decide should be the functionality of the UI. If the functional aspect of the UI outweighs its expressive purpose, it is uncopyrightable. [FN44] In the comparison stage, the court points out that the intrinsic functionality of a UI renders its protection thin. In Productivity Software Inc., v. Healthcare Technologies, Inc., [FN45] the District Court of the Southern District of New York, finding no single element protected after having filtrated, applies the compilation doctrine to the elements taken as a whole. The court granted a narrow protection to the arrangement of the display screens because it was not highly expressive. This illustrates the possible combination of the two methods (AFC test and compilation doctrine). [FN46] While stating that it endorses the Altai test, the Court of Appeals of the Ninth Circuit in Apple v. Microsoft, [FN47] however, creates its own method. [FN48] Having dissected the program to find its copyrightable elements, the court then determines the scope of the plaintiff's copyright. [FN49] If it is broad (a lot of elements are protectable), the substantial similarity test can be used; if it is thin (a large number of elements are uncopyrightable), only "virtual identity" will infringe. [FN50] As the court finds many single elements unprotectable, only the overall *61 look and feel of the work (or in other words, the elements seen as a compilation

[FN51]) is protected. This protection is thin because it attaches only to the selection or arrangement of the elements.
In Lotus v. Borland [FN52] and MiTek Holdings, Inc., v. Arce Engineering Co., Inc., [FN53] courts dealt with UIs which were menu command hierarchies [FN54] or structures. The two courts decided that these UIs were methods of operations falling under section 102 (b)'s exclusion. The First Circuit in Lotus found the AFC test was best suited to non-literal infringement cases and refused to apply it because Lotus was a literal infringement case. [FN55] The First Circuit compares the UI to the buttons of a video cassette recorder, which are purely functional. Indeed, if specific words (i.e. file, save) are essential to operating something then they are a part of a "method of operation" and as such are unprotectable. [FN56] In Lotus, the structure as a whole itself qualified as a method of operation, so that the compilation of commands was also uncopyrightable. On the contrary, the MiTek court found the plaintiff's UI as a compilation copyrightable. However, as the defendant's UI was substantially different from the plaintiff's, it was found non-infringing. [FN57]
As a conclusion, although courts all apply the compilation doctrine at some point of their reasoning, they diverge as to the correct way to judge UI infringement. This inconsistency in case law has raised many criticisms.

(3) Criticisms on user interfaces cases

Several authors have expressed dissatisfaction as regards the way courts apply copyright rules to UIs. [FN58] The majority claim that copyright law is not the most adequate legal tool to protect UIs. They suggest several alternative protections to copyright law for UIs, such as trade mark or trade dress protection, patent or even a type of new sui generis protection. [FN59] Two authors, however, deny that there is a problem with the protection of UIs. The reason is that the user interface portion of a program is copyrightable as part of an audiovisual work [FN60] and principles to be applied to this aspect of computer software are fairly well settled.
D. Karjala makes a more elaborated development of this latter argument. [FN61] In his opinion, UIs should be protected as pictorial or graphic (i.e. screen displays), audiovisual, musical, literary (i.e. help text) or similar traditional copyrighted works for their appearance. As *62 for their functional aspect, authors should claim protection as a non-obvious technological advance under the patent standards. Indeed, interfaces, being a result produced by the program, are not part of the program themselves and therefore should not be granted a "computer copyright" protection. [FN62] He therefore rejects the courts' classification of UIs as "non-literal elements" of the program generating the interface. [FN63] As Karjala seems to infer, the appropriate test for a UI case would be first to separate the functional elements from the non-functional ones, using section 102 (b), the AFC test or another method. After this "separation" the remaining elements would be considered under the traditional copyright section 1 for pictorial or literary works. If no non-functional element subsists after dissection, then the UI is uncopyrightable. An alternative or cumulative method to D. Karjala's idea would be to use the Mazer and Kieselstein-Cord rulings. [FN64] When functional and artistic features are fused in the same object, the functional aspect does not preclude copyright protection of the artistic feature as long as the latter can be separated physically or conceptually from the former.
In summary, U.S. doctrinal opinions diverge as to what type of protection, if any, should be afforded to UIs. A solution would be to eliminate the functional elements and protect the remaining ones with the traditional copyright for pictorial or literary

works as opposed to the "program copyright". [FN65] If this question remains unsatisfactorily resolved by courts, the legislature will have to tackle the problem. Anyway, for the time being, copyright is still the only means by which user interfaces are protectable. Precedents are there to show it, and courts do not seem ready to change their rulings.

(4) Is the AFC test still viable?

Despite its relative efficiency, the AFC test has been largely criticised, as it holds several disadvantages. The first criticism relates to the very thin protection that the test tends to give to computer programs. It is important to understand why. [FN66] The filtration step tends to exclude a great number of elements. These elements are generally dictated either by efficiency or by functionality. They constitute the most important components to the functioning of computer programs, [FN67] but paradoxically they will be deemed unprotectable. This is because copyright excludes functional articles from its protection. [FN68] In consequence, the Altai test rewards "inefficient programmers over efficient ones". [FN69] Indeed, because it excludes from protection a program's efficient elements, a programmer would be induced to create an inefficient program to get protection. [FN70] The paradox is grounded in the difficult application of the idea/expression principle: if the limiting doctrines of merger, scenes a faire, etc., are applied to computer program elements, the scope of copyright protection is narrowed. Consequently, the promotion of competition and technological progress are enhanced because elements are freely available for all programmers. But if these limiting doctrines are applied too broadly and "consume the program that they were designed to protect", [FN71] then programmers have no incentive to create because they know they will be refused protection. Thus advancement slows down. Conclusively, the main problem in applying the AFC test is finding the correct balance between idea and expression. The line between the two being extremely hard to draw, courts risk either granting protection to ideas or excluding expression from protection. The end result is that it leaves the software industry insecure.

The second criticism is that the AFC test does not allow compilation protection for programs. Precisely, the compilation doctrine should be applied in order to protect the architecture or "design-level expression". [FN72] This second criticism loses its strength as numerous courts increasingly tend to use the compilation doctrine in internal elements cases as well as in user interface cases. [FN73] Indeed the Feist doctrine does not preclude the application of the AFC test and reciprocally. Altai will find single elements to be copyrighted while Feist will find the overall structure copyrightable independently *63 of the copyrightability of the internal elements. In consequence, the legal protection of a program can be twofold while, most importantly, it respects the idea/expression principle. In summary, courts are nowadays certainly more inclined to combine the AFC text with the compilation theory.

While one commentator believes that copyright is adaptable and capable of protecting programs in a suitable manner, [FN74] several others propose other types of protection, as a remedy to the disadvantages of the AFC test. Some argue that patents can protect software efficiently. [FN75] Indeed, the number of patents granted by the Patent and Trademark Office to computer programs has been increasing since the 1980s. [FN76] Now courts have been as far as to patent software, as long as it is embodied in tangible medium, such as diskettes. [FN77] However, patent protection

holds major inconveniences: the cost and length of procedures in order to be protected, [FN78] which copyright protection does not require. [FN79] Others suggest trade secret protection. [FN80] To be effective, though, trade secret protection requires contracts between parties, which is sometimes not possible or not suitable (for example for widely sold computer programs) and secondly, U.S. trade secrets law being a matter left to state law, it leads to legal uncertainty if programs are sold in several states having different trade secret laws. Another disadvantage is that protection by trade secret might be considered as circumventing federal copyright law. [FN81] Sui generis protection might thus be a more suitable solution in the long term. [FN82]
To conclude, U.S. commentators' views are split over the question of the adequacy of the AFC test to judge of the copyrightability of programs. Some commentators still believe that the AFC test is the best "stopgap solution" to alternative protections, but that it must continuously be adapted to correctly protect software. [FN83] Others severely criticise it and suggest alternative protections. In view of the cases, one must admit that the abstraction-filtration-comparison method is now generally accepted in all circuits and seems to be considered by them as a quite efficient standard based on a structured approach and on a long-lasting ground of precedents. However, commentators' opinions must not be underestimated as they might influence later courts' rulings.

Europe

In contrast with the booming software litigation in the United States, European courts have not been faced with a tremendous number of "copyrightability" cases. A reason might be that companies make licence and employment contracts as well as confidentiality agreements so effective that no licensee nor developer ever dares to breach them. [FN84] Some also do not allow tele-working in order to maintain control over the source codes. [FN85] Another reason is that when there is a violation, few plaintiffs decide to introduce an action owing to the slowness of trials. Others ask for a "summary judgment" which designates an expert to describe the defendant's products. On the expert's conclusions of similarity or literal copying, the parties transact.
French courts have not elaborated a structured reasoning nor a test, but merely rely on experts' opinions. [FN86] Experts seem generally to be implicitly using more or less the same doctrines as merger, scenes a faire, industry standards and public domain, in a somewhat similar manner as AFC test's filtration stage. In CA v. Faster and Altai [FN87] (which involved the same parties as in the *64 American CA v. Altai case and Faster, the distributor in France of CA's products), the French court adopted the reasoning of the U.S. court but without questioning whether it had to be done. Some courts apply a quantitative rather than a qualitative approach to judge infringement cases. [FN88] None of them clearly uses a method, be it similar or not to be AFC test. As a result of the courts' lack of method, they find for infringement of unprotectable elements of the alleged infringed program or find for no infringement because the second program only copies a small amount of the plaintiff's program, while this small amount might be the core of protectable expression. In sum, confusion seems to reign as to which method should be used in order to find for infringement.
In the United Kingdom, the Chancery Division of the High Court tried three cases which cause of action however accrued before the effective date of the Directive. [FN89] In the three cases, judges used the traditional British requirements of

copyrightability ("skill and labour") and of infringement ("reproduction of a substantial part of the work" which must be a result of copying), which they are allowed to do as the facts preceded the entry into force of the Directive. In John Richardson Computers Ltd v. Flanders Chemtec, while Ferris J. purported to rely on Altai, he in fact applied the traditional U.K. requirements for copyrightability and for infringement. In fact, this case was more a "user interface case" because the court did not look at the codes, only at the UIs. [FN90] In applying the U.K. standards, the court protected functional features of the UI. This clearly shows that it did not make a faithful application of Altai. Nevertheless and interestingly, the court found a screen display to be a product of a program and not the program itself.

In Ibcos Computers Ltd v. Barclays Mercantile Highland Finance Ltd, [FN91] Jacob J. elaborated the test of copyright infringement as follows: (1) what are the work or works in which the plaintiff claims copyright? (2) is such work original? (3) was there copying from that work? (4) if there was copying, has a substantial part of that work been reproduced? In this case, the judge looked at the codes. The plaintiff's program was in fact a complicated composite of interrelated programs, routines and subroutines. Owing to a substantial amount of skill and labour, each individual program was found original and the "package" of programs as well. To find copyrightability of the programs as a whole, Jacob J. applied the U.K. Copyright Act, which specifically provides for the protection of original compilations. After finding that the plaintiff's work had been copied, the judge discussed the relevance of applying the American test for infringement. The judge differentiated between U.S. and U.K. copyright law. While the American statute and case law have expressly excluded functional works of the copyright protection, the British have not done so. In other words, while in the United States any idea will be excluded under section 102 (b), in the United Kingdom a "detailed idea" (such as a book's table of contents) can be protected if original. [FN92] Thus, for Jacob J., Altai is not helpful in resolving the case. Accordingly, he applies the U.K. "substantial part of plaintiff's work" standard. This standard seems close to the quantitative method used by the French courts. It is also likely to protect non-original elements.

A step forward has been taken in the recent case Cantor Fitzgerald International and Cantor Fitzgerald Securities v. Tradition (United Kingdom) Ltd, Michael Howard and Christopher Harland. For Pumfrey J., the test elaborated in Ibcos has to be adapted to the circumstances of each case. Substantial parts must not be understood as covering all parts of a program because without any of them a program would not function as desired. Substantial parts are, on the contrary, only those where the author's skill and labour can be found. [FN93] The court also reaffirms that literal copying is not necessary to find infringement, as the structure of a program is protected if it shows enough skill and labour.

A clear conclusion as to the U.K. courts' position cannot be drawn as their rulings are based on the law applicable prior to the Directive. These cases, however, seem to show that British judges are somewhat hesitant to utilise U.S. case law in computer program infringement cases. It remains to be seen if U.K. courts will use the requirement of "creation" instead of skill and labour in future cases. Since Cantor, the "substantial part" standard seems to move towards a qualitative rather than quantitative standard. It seems that, with this qualitative standard, courts can roughly get results close to ones resulting from abstraction and filtration.

In Ireland, one case raised the question of infringement of a software. However, the question of copyrightability was not discussed because the plaintiff did not offer

scientific evidence that the defendant had copied the algorithm used in his software. [FN94]
In Belgium, litigation mainly involved slavish copying. Courts generally presumed that the infringed program was original without inquiring into the *65 copyrightability of the internal elements. [FN95] However, recently the Brussels Court of Appeals confirmed a summary judgment designating an expert to analyse the similarities and differences between the plaintiff's and the defendant's programs. [FN96] Indeed, a dissection method close to the AFC test could be used analogously in Belgium. Courts have in fact adopted a test for literary and audiovisual works which consists of abstracting and dissecting the work and comparing their similarities and differences. [FN97]
Around the time the Software Directive was adopted, German courts had to deal with video games, which are similar to graphical user interfaces. [FN98] The German courts decided that interfaces were protectable independently from the computer program underlying them, as (audio) visual works. This meets the solution singled out earlier in the conclusion on user interfaces in the United States, French and Belgian courts adopted similar rulings. [FN99] It is thus probable that future French, Belgian and German decisions would hold the same reasoning for computer program user interfaces.
In sum, European courts have either not yet fully been faced with copyrightability issues or have decided them without a structured analysis. This makes it urgent to find an appropriate method of deciding cases to restore confidence and legal security.

Originality cases

The United States

In the United States, there is no concept of originality specific to computer programs. This is due to the fact that U.S. copyright law is unified, in all states and for all works, whereas European copyright law is only partially harmonised. As discussed earlier, the clearest explanation of originality was set in Feist Publications, Inc. v. Rural Telephone Service Co., Inc. [FN1]
One case, EDI v. SSI, reiterated the Feist standard of originality and applied it to user interfaces. [FN2] "In general, copyright only protects originality of user interface to the extent that the selection of variable inputs from the universe of potential inputs reflects non-functional judgments." [FN3] Thus the input format of a structural engineering computer program is based on a minimum level of originality if the program's creator exercised judgment in formulating input cards, rather than merely reflecting industry standards and laws of engineering. This reflects the common American and European originality requirements for compilations: selection or choice in the elements amounts to originality, as long as this choice or selection is creative, in other words not dictated by standard techniques.

Europe

Few European fora construed the originality requirement set in the Directive. In Belgium, two decisions, however, trying facts which occurred before the Directive's entry into force, applied a notion of originality similar to the Directive's standard. The commercial court of Charleroi decided there was no originality, as no program such as the ones involved in litigation could function without the rules used. [FN4] Even if

the court did not apply the Directive's notion it would have come to the same conclusion if it had been obliged to do so. The Brussels Court of Appeals, while referring to "the author's own intellectual creation", however, uses other definitions. Therefore, as it does not only and clearly apply the Directive's standard, "it would be incorrect to see in this decision a clear example of the theory of the Member States' obligation of interpreting the national law in conformity with E.U. law". [FN5]
A first German decision in the aftermath of the Directive's coming into force rightly construes the originality requirement. In Buchhaltungsprogramm, [FN6] the *66 Bundesgerichtshof clearly asserts that the Directive has imposed a less high level of originality for computers, and it states that, in the future, German courts will have to respect it. [FN7] The court structures the examination of the originality requirement in two stages: first, one has to examine whether the plaintiff's own intellectual creation in the program has been copied by the defendant and, secondly, whether the program shows some individuality. [FN8] By this, the German Supreme Court means that it must not represent a mechanical stringing together of previously known material or material in the public domain, and must not be based entirely on purely routine programming work.
A year later, in CA v. Altai v. Faster, [FN9] the commercial court of Bobigny found that "originality is to be perceived through the mark of an intellectual contribution, implying a minimum threshold of creativity, a contribution of something new which involves more than the application of automatic and compelling logic". [FN10] This interpretation of originality is very close to Feist's. But it may be permissible to doubt that originality "must be a contribution of something new". Novelty is not a requirement of copyrightability. Perhaps the court meant "novelty" in the sense of expression (as opposed to ideas). In view of the context, it is reasonable to think that it had such an intent. Recently, a French court, while correctly distinguishing between the program internal elements and the UI, decided, however, that if the UI is original, the program as a whole is original too. [FN11] It is incorrect to rule that the entire program is original if the court does not look into the program's internal elements. Nonetheless, except for this case, the originality requirement is similar if not identical in France and the United States. In addition, the French and German decisions converge as to the non-originality of routine mechanical work.
The German and French courts correctly interpreted the Directive's standard. These first European decisions might lead the way for other Member States to follow this conception, which meets the American one. [FN12] Besides, it is desirable that American and European case law converge on the concept of originality to see a start in the international standardisation trend. It would be a relief (and an incentive) for software companies as they would be equally protected in both systems.

Proposition for a Test

After having analysed legislation and cases, the question of the applicability of American precedents in Europe can be tackled. Part 1 has showed that both statutory texts are, on the three matters discussed, very similar. American and European case law on the program definition and originality at present converge. The most controversial and fundamental issue is then the copyrightability of the several and diverse program elements. The case law of the Member States, which has not yet come to maturity, errs because it has not set a structured manner to determine infringement. In order to help courts adapt American decisions to European situations when relevant, a test can be proposed.

Reasons for the test

Three essential reasons dictate why a test should be adopted. First, the texts do not differ in both systems. Secondly, European courts do not have experience in the field of program protection and have difficulties in finding an appropriate method of ruling. Thirdly, the United States has on the contrary acquired a well-settled body of case law. As European judges might be tempted to use U.S. case law on these grounds without further reflection, the test stands as a safeguard against inopportune applications and as a tool to determine how it must be applied, when relevant.

Method to apply the test

This test would consist of three different stages of reasoning: comparison--comprehension--application (or non-application). Before applying an American precedent, European judges should first ask themselves whether the two texts cover the same legal reality and whether policies are similar. In order to discover this, they should try to understand the reasons underlying the legislation under consideration (is it droit d'auteur or "copyright s.s." orientated; what is the legislator's intent?) and underlying the decision itself (why courts decided that way; is it sound reasoning; is the line of argument applicable in Europe or is the rationale general or particular to U.S. law?). Indeed it is not sufficient that text formulation and policies are similar; the line of argument of the cases must also be correct. Finally, if the texts and the decision reflect similar policies and rationales, they should consider applying the case in Europe. If they come to the opposite conclusion, they should decide not to do so.

Application of the test to the different issues

It is now possible to apply the three-tiered test. Unfortunately, the lack of consistent case law confines one to making sheer conjectures. One can at least venture into some general guidelines for future decisions.

As regards the program definition, the formulation of the texts, the policy underlying the texts and the reasons *67 underlying the decisions being analogous, Apple v. Franklin can be used in Europe. The issues of the meaning of "instructions" and of the inclusion of the source and object codes are now settled by TRIPs.

As to the scope of protection, several positive and negative aspects of the AFC test must be considered. First, the AFC test is based on the idea/expression principle common to the Software Directive and the Copyright Act. So, the texts and policies underlying the test being alike, nothing prevents European judges from using the AFC test. For instance, although most non-literal elements are mentioned in the European texts, the determination made in Altai can still be relevant for the non-literal elements that the European texts do not cite (such as parameter lists, structure and macros). Secondly, despite criticism on the AFC test's counter-productive effects, virtually all commentators agree that the AFC test is the best stop-gap solution and, moreover, there is no readily available alternative protection to copyright, be it in the United States or Europe, to protect programs. [FN13] European judges should therefore be aware of these considerations before considering application of the AFC test. Thirdly, the Directive's phrasing provides room to judge more generously than the "Altai-generation courts" in two instances. First of all, the Directive clearly states that a

program can be protected as a compilation whereas, in the United States, it took a long time for courts to use this legal concept. The compilation doctrine thus provides a good palliative to Altai's unfortunate effects. Secondly, when logic, algorithms and programming languages do not comprise ideas or principles, they can be protected in Europe. Courts can then reject the Altai precedent because the legislative texts on which it is based are phrased differently. This is the only point of law on which American and European case law could diverge. However, it would not be advisable to follow this path because the Directive is not in harmony with traditional copyright s.l. doctrines and with the TRIPs on this point. [FN14]

UIs raise the question of the type of protection they should be granted. American doctrine is not unanimous on the type of protection to choose and case law hesitates between various tests. Thus the three-tiered test is difficult to apply. However some things are certain in both systems. Copyright is the only "legislatively" speaking available protection, and the one applied unanimously by American as well as European courts. UIs can enjoy two kinds of protection which can be combined: the copyright protection applicable to (audio)visual and literary works and the compilation doctrine (for which the Directive provides expressly). The combination of these protections can afford UIs a rather secure and strong protection without giving a broad monopoly to the author. There is no need to use U.S. case law to protect UIs by visual and compilation copyright. European courts already have their own body of law regulating this matter. Indeed the visual works protection and compilation doctrines are not defined in the Directive, and until the European Union harmonises law on this point, Member States remain free to judge according to their national copyright s.l. law. Nonetheless, nothing should preclude the application of U.S. law if the "comparison--comprehension--application" test concludes for application. Each judge knows if national case law is sufficiently developed on the point to decide to apply the suggested test or not. Finally, the seemingly most suitable manner to decide UIs issues would be to abstract their functional aspects and protect the remaining artistic features by the visual or literary copyright.

At last, as the U.S. policy underlying the concept of originality is not in contradiction with the one set out in the Directive, and the reason behind the Feist decision is close to the reasons guiding E.U. law, Feist's holding can be used in Europe.

The test can in like manner be used in other domains covered by the Directive, for instance decompilation. The Directive sets a fairly precise and strict provision to determine the limited cases in which the decompilation is permitted. The fair use doctrine application is prohibited by the Directive. Therefore this is a clear case where U.S. precedents cannot be utilised in the E.U. legal system. The texts do not coincide and the policies underlying provisions are diametrically opposed. Another example of the possibility of using the three-tiered test is the presumption of ownership in favour of the author's employer, which reflects an American influence.

Conclusion

The aim of this analysis was to discover whether U.S. precedents can be used in the European Union. On the one hand, the statutory texts analysed show a fundamentally similar drafting and reflect the same policies. For this reason, U.S. case law is readily applicable on the following three points. First, the definitions of "program" are virtually identical in both legal systems and do not reflect either a copyright s.s. or a droit d'auteur policy. Secondly, application by U.S. and European courts has resulted in a convergence of statutory standards for "originality" in the two systems. Thirdly,

U.S. and European texts do not diverge with respect to the general idea/expression principle.
On the other hand, the ambiguous formulation of Recital 14 of the Directive as to logic, algorithms and programming languages could cause future problems if judges interpret Article 1.2 as including certain ideas and principles in the scope of protection. As to the AFC method, it is generally agreed that it is an adequate tool to find copyrightability of elements and that copyright s.l. remains the less improper solution. Since the AFC test does not clash with the droit d'auteur principles, nothing prevents its adoption by European courts. If, for the time being, European judges endorse the AFC test, the transplantation of U.S. precedents will integrate a common pattern of reasoning and create a uniform body of case law. However, the test shows that copyright as applied to software is not the total appropriate solution for protection. Furthermore, copyright *68 and patent protections are complementary but as yet, neither is fully adequate. [FN15] Thus, as the evolution is not yet ripe, European as well as American courts should be encouraged to find other solutions to protect software, be they other tests or sui generis-type protections. However, the American and European judiciaries face the same obstacle: they do not have much room to create alternative protections. Being confined to their copyright and patent laws, they are compelled to await a legislative initiative.
On a broader scale, a more suitable software protection should be fostered not only in Europe and in the United States but also internationally. An international review of the protection of programs should be undertaken to create, if technically possible, a correct common definition of computer program, change the program classification as "literary work" and grant it a tailor-made protection. A sui generis protection, along the same lines as the one set in the E.U. Database Directive, could be the basis for a better understanding and protection of computer programs. As certain aspects are better protected by copyright s.l. or by patent, a combination of copyright s.l. and patent along with a sui generis protection could be considered as well. Such a combination could safeguard the principle that all ideas should be free for anyone to build on. Additionally, it would reward the time and resources invested into the conception of a program, which cannot be protected through classical copyright protection. If such a balanced option materialises in an international arena, a common protection of programs will create easier investment conditions worldwide. Consequently the creation of complex and efficient software, as well as the progress of science, will expand and contribute to the welfare of the society. This is the goal that both Anglo- Saxon copyright and continental droit d'auteur law intend to achieve.

[FN1]. A more recent decision raised the issue of new types of software. However, it was not the main issue. As its rationale is laconic and very obscure, its importance is consequently diminished. See Baystate Technologies Inc., v. Bentley Systems, Inc. 946 F. Supp. 1079 (December 6, 1996).

[FN2]. 714 F. 2d 1240 (3d Cir. 1983), cert. dism., 464 U.S. 1033 (1984).

[FN3]. 775 F. Supp. 544 (E.D.N.Y. 1991).

[FN4]. ibid., at 559. In a word-processing program, for example, text can be deleted, blocks of text can be moved, formatting of documents can be changed; all sorts of operations can be instituted, and these can only be described as behaviour.

[FN5]. See S.P.R.L. AB2M Computers v. S.C. Sogestic, Charleroi commercial court, January 19, 1993 (1993) Jurisprudence Liege Mons Bruxelles 1178; (1993) Ingenieur-Conseil 375; (1993) Revue regionale de Droit 404 and comment, E. Montero. The codes were found uncopyrightable because non-original. See Brussels Court of Appeals, October 14, 1993, (1993) Ingenieur-Conseil 358; (1994) 2 Computerrecht 46. These cases, however, judged facts which occurred before the Directive's entry into force.

[FN6]. 3 V Finance v. FL Software, Patrick L. and Alexandre F., Tribunal de grande instance de Paris, 3d ch., May 31, 1995 (1995) Expertises, 319. See J. Gyngell, "Compilations of Computer Programs: Total Information Processing Systems Ltd v. Damar Ltd" [1992] E.I.P.R. 95, which discusses a British case decided before the implementation of the Directive.

[FN7]. ESX v. Tech Com, commercial court of Creteil, January 17, 1995 (1995) Expertises 161.

[FN8]. § 101 (see definition of "fixed"). To the author's knowledge, no U.S. court decision has made a distinction based on the particular medium in which a program is recorded.

[FN9]. See, e.g., Alcatel USA, Inc. v. DGI Technologies, Inc. 166 F. 3d 772, 49 U.S.P.Q. 2d 1641 (5th Cir. (Tex.), January 29, 1999); DSC Communication Corp. v. DGI Technologies, Inc. 898 F. Supp. 1183, 1995 Copr. L. Dec. P 27,490, 37 U.S.P.Q. 2d 1496 (N.D.Tex., September 1, 1995); DSC Communications Corp. v. DGI Technologies Inc. 81 F. 3d 597, 64 USLW 2723, 1995 Copr. L. Dec. P 27,513, 38 U.S.P.Q. 2d 1699 (5th Cir. (Tex.), April 30, 1996); Compaq Computer Corp. v. Procom Technology, Inc. 908 F. Supp. 1409, 1996-1 Trade Cases P 71,264, 1995 CopR. L. Dec. P 27,491, 37 U.S.P.Q. 2d 1801 (S.D.Tex, December 6, 1995); Soft Computer Consultants, Inc. v. Lalehzarzadeh 1988 WL 161312, 1989 Copr.L. Dec. P 26,403 (E.D.N.Y., July 26, 1988).

[FN10]. See below, discussing Computer Associates v. Altai 982 F. 2d 693, (2nd Cir. 1992); Gates Rubber v. Bando Chemical Industries 9 F. 3d 823 (10th Cir. 1993 (October 19, 1993)); Harbor Software v. Applied Systems 887 F. Supp. 86, 925 F. Supp. 1042, and 936 F. Supp. 167 (S.D.N.Y.); Bateman v. Mnemonics 79 F. 3d 1532 (11th Cir. 1996) (March 22, 1996).

[FN11]. If patent or semi-conductor chip protection is not available, firms could for instance turn to the European Database Directive 96/9 [1996] O.J. L77/20 if they can get over the hurdle of Art. 1.3 (proving that their new type of program is not a program under the Software Directive). Another solution would be to try to protect computer programs under the general U.S. and European states' copyright law which provides for protection of compilations.

[FN12]. It is acknowledged that the principle of national treatment exists in the Berne Convention as to literary works (in which computer programs are included). The problem is that there is no mention nor definition of a computer program in Berne and countries could refuse protection to a program if it does not fit into its definition of program.

[FN13]. Art. 10.1 of TRIPs only says that computer programs are literary works and are composed of source and object codes. But the Article does not define the program nor other protectable elements. For national treatment, see Art. 3 of TRIPs.

[FN14]. L. M. Gable, in "The feasibility of the abstraction-filtration- comparison test for computer software copyrightability (and analysis of Bateman v. Mnemonics)" (1998) 14 Ga. St. U.L.Rev. 447, however, reviews both types of cases in chronological order without making such a distinction.

[FN15]. See this section, Bateman v. Mnemonics.

[FN16]. See J. Cai, "Bateman v. Mnemonics, Inc., 79 F. 3d 1532 (11th Cir. 1996) Abstraction-Filtration-Comparison, Should the Test be 'Literally Copied' in a Computer Program Literal Copying Case?" (1998) 16 Temp. Envtl. L. & Tech. J. 287; and Bateman v. Mnemonics, 79 F. 3d at 1544, n. 25 (quoting Melville B. Nimmer and David Nimmer, Nimmer on Copyright (1995), 13.03[A](1)).

[FN17]. ibid.

[FN18]. Computer Associates v. Altai 982 F. 2d 693 (2nd Cir. 1992). For the District Court judgment, see Computer Associates v. Altai, 775 F. Supp. 544 (E.D.N.Y. 1991). The district court found no non-literal element protected, and the Court of Appeals affirmed the District Court analysis on this point, see CA v. Altai 23 U.S.P.Q. 2d 1241, at 1259-1260. D. S. Karjala, "Recent United States and International Developments in Software Protection" [1994] E.I.P.R. 58, believes a consequence of the Altai decision is that "only literal code or close paraphrasing is protected." See also below, the discussion on data types in Gates Rubber.

[FN19]. The first case to recognise non-literal elements were copyrightable was Whelan Assocs., Inc. v. Jaslow Dental Lab. 797 F. 2d 1222, 1238 (3d Cir. 1986), cert. denied, 479 U.S. 1031 (1987).

[FN20]. See Altai, n. 18 above, at 702. This was already announced in Whelan, n. 19 above.

[FN21]. For definitions of these elements, see Altai, n. 18 above, at Background, I. Computer program design and thereunder; see also J. W. L. Ogilvie, "Defining Computer Program Parts under Learned Hand's Abstractions Test in Software Copyright Infringement Cases" (1992) 91 Mich. L. Rev. 526.

[FN22]. Nichols v. Universal Picture Corp. 45 F. 2d 119, (2d Cir. 1930), cert. denied, 282 U.S. 902, 51 S. Ct 216 (1931); Sheldon v. Metro Goldwyn Pictures Corp. 81 F. 2d 49 (2d Cir. 1936), cert. denied, 298 U.S. 669, 56 S. Ct 835. See also Gable, n. 14 above, and R. V. Baca, "Copyright law; Tenth Circuit Application of the Abstraction-Filtration-Comparison Test to Determine the Scope of Copyright Protection for Computer Programs: Autoskill v. National Educational Support Systems" (1994) 24 N.M.L. Rev. 413 at 423.

[FN23]. A good and synthetic description is given by Gable, ibid. at 463- 464.

[FN24]. For definition of these notions, see n. 68, Part 1 of this article, [2000] E.I.P.R. 7 at 14.

[FN25]. CA v. Altai 982 F. 2d at 712. See also Baca, n. 22 above, at 421.

[FN26]. After the Altai ruling, some circuits showed interest for the AFC test, although the facts involved were dissimilar. See, e.g., Data Gen. Corp. v. Grumman Sys. Support Corp. 803 F. Supp. 487 (D. Mass. 1992) (October 9, 1992) (since the defendant admitted taking copies of the infringed computer program, reproducing and using them without modification (slavish copying), detailed examination of code by the Altai test was not necessary); Bateman v. Mnemonics, n. 16 above (the fact that detailed examination of the code is not necessary when there is literal infringement has been re-questioned); Sega Enters Ltd v. Accolade Inc. 977 F. 2d 1510 (9th Cir. 1992) (October 20, 1992 and amended January 6, 1993); Autoskill v. Nat'l Educ. Support Sys., Inc. 994 F. 2d 1476 (10th Cir.) (May 19, 1993), cert. denied, 114 S. Ct 307 (1993) (upheld the AFC test). See also J. Cai, n. 16 above, and Baca n. 22 above, at 422-423. See also Karjala, n. 18 above, at 61-62.

[FN27]. n. 10 above.

[FN28]. The court makes an important remark before embarking on the explanation of the abstraction step. Abstraction is useful in enabling a court to filter out ideas and processes from protectable expression, but it is not an end in itself and does not itself identify them; see n. 10, above, at 834.

[FN29]. Based on the article by Ogilvie, n. 21 above, the court gives definitions of these elements; see n. 10 above, at 835.

[FN30]. Briefly, processes can be found at any level, except perhaps the main purpose level (which reflects an idea, not a process). Most commonly, processes will be found as part of the system architecture, as operations within modules, or as algorithms. Facts may be found at a number of levels of abstraction, but will most often be found as part of data structures or in the source or object codes. Finally, the scenes a faire doctrine excludes external factors. In the area of computer programs these external factors may include: hardware standards and mechanical specifications, software standards and compatibility requirements, computer manufacturer design standards, target industry practices and demands, and computer industry programming practices (n. 10 above, at 837-838).

[FN31]. ibid., at 836, 839.

[FN32]. ibid., at 839.

[FN33]. ibid., at 841. J. Cai, n. 16 above, agrees with the Gates court views: "during the filtration stage, courts should avoid applying the analysis in an overly restrictive or mechanical manner which ignores a comparison of the two works as a whole." Apple v. Microsoft 35 F. 3d 1435 (9th Cir. 1994) also suggests gaining an overall picture but after dissection; see below.

[FN34]. n. 10 above, at 834, at n. 12.

[FN35]. See, e.g., Engineering Dynamics, Inc. v. Structural Software, Inc. 982 F. 2d at 706 (three-step test can and should be modified when computer technology demands it); 9 F. 3d 823, at 834, n. 12 (same); Control Data Systems v. Infoware 903 F. Supp. 1316, at 1322 (D. Minnesota) (August 17, 1995). This court cites Lotus, Eng'g Dynamics, Gates Rubber Co., and Altai and endorses the AFC test. But as the case was decided under a preliminary injunction, there was no deep analysis.

[FN36]. There were in fact three cases in the Harbor litigation (887 F. Supp. 86, (June 5, 1995), 925 F. Supp. 1042, (May 14, 1996), and 936 F. Supp. 167 (September 9, 1996) (S.D.N.Y.)): the first one is interesting for the facts description, the two other ones for the definition of the scope of protection and for the test application. The court follows the more precise filtration analysis developed in Gates (see 925 F. Supp. 1042, at 1052). In the third decision (936 F. Supp. 167), it proceeds to the comparison and finds certain protected elements copied by defendant are substantially similar to those of plaintiff.

[FN37]. n. 10 above, at 837.

[FN38]. For rejections of the application of the AFC test, see, e.g., Data Gen. Corp. v. Grumman Sys. Support Corp. 803 F. Supp. 487 (D. Mass. 1992); Lotus v. Borland 49 F. 3d 807, (1st Cir. 1995), (S. Ct 1996), in which the court stated that the AFC test was not helpful in deciding literal copying cases. For the Lotus v. Borland decision, see below.

[FN39]. n. 10 above.

[FN40]. Gable, n. 14 above, at 476-477.

[FN41]. See Mitel v. Iqtel 124 F. 3d 1366 (10th Cir. 1997) (Altai test applied to a series of command codes to access the features of a piece of telecommunications hardware (a call controller)).

[FN42]. The five cases can be reviewed under different categories of approaches: chronologically, by circuit, by the type of facts, by the type of infringement (literal: EDI, Lotus, MiTek or non-literal: Apple, Productivity Software, MiTek), by the nature of the UI (pictorial: Apple, MiTek or literary: Lotus, Productivity Software, EDI) or by the kind of reasoning adopted by the courts.

[FN43]. 26 F. 3d 1335; 63 USLW 2066, 1994 CoPr. L. Dec. P 27,300, 31 U.S.P.Q. 2d 1641 (5th Cir. 1994) (July 13, 1994) (further referred to as "EDI v. SSI" or the "EDI case").

[FN44]. See also D. S. Karjala, "The Relative Roles of Patent and Copyright in the Protection of Computer Programs" (1998) 17 J. Marshall J. Computer & Info. L. 41 at 44-48.

[FN45]. 1995 WL 437526 (S.D.N.Y.) (July 25, 1995). Productivity Software program was conceived to "allow a user to increase his or her typing efficiency by typing

shortened forms of words and phrases which are then automatically expanded into corresponding long forms".

[FN46]. The compilation doctrine was mentioned as a complementary protection in the Harbor cases. In addition, the Productivity Software court uses the average lay observer test to decide that the screen is dissimilar. Understanding why is easy: a UI consists of an audiovisual work, which does not require experts to determine similarity. The definition of "substantial similarity" for visual works in the Second Circuit is "whether an average lay observer would recognize the alleged copy as having been appropriated from the copyrighted work". See, e.g., Steinberg v. Columbia Pictures Industries, Inc. 663 F. Supp. 706 (S.D.N.Y. 1987); Ideal Toy Corp. v. Fab-Lu Ltd 360 F. 2d 1021, 1022 (2d Cir. 1966); Silverman v. CBS, Inc. 632 F. Supp. 1344, 1351-1352 (S.D.N.Y. 1986).

[FN47]. 35 F. 3d 1435 (9th Cir. 1994) (Sept. 19, 1994). Apple sued Microsoft for using its desktop metaphor, icons, zooming rectangles, etc.

[FN48]. ibid., at 1445: "Other courts perform the same analysis, although articulated differently." Among other cases, the court cites namely Altai, Gates Rubber and EDI.

[FN49]. Using this method, the court found in this case five basic ideas (embodied in the desktop metaphor) that can generally be copied by any competitor: use of windows to display multiple images on the computer screen and to facilitate user interaction with the information contained in the windows; iconic representation of familiar objects from the office environment; manipulation of icons to convey instructions and to control operation of the computer; use of menus to store information or computer functions in a place that is convenient to reach, but saves screen space for other images; and opening and closing of objects as a means of retrieving, transferring and storing information.

[FN50]. See Gable, n. 14 above, at 471.

[FN51]. The court purports to apply Feist; see Apple, n. 47 above, at 1442, 1446. "Any claim of infringement that Apple may have against Microsoft must rest on the copying of Apple's unique selection and arrangement of all of these features."

[FN52]. 49 F. 3d 807 (1st Cir. 1995) (March 5, 1995); (S. Ct 1995) (January 16, 1996).

[FN53]. 89 F. 3d 1548 (11th Cir. 1996) (August 5, 1996). The software at stake was a wood truss "lay out" program. MiTek's facts involved three issues, among them the copyrightability of internal elements and of user interfacts. MiTek first contended that the District Court did not proceed to the abstraction, and acted directly at the filtration stage. Indeed, as a MiTek expert had already presented to the court the abstracted protectable elements, the court judged this was enough and that it needed not abstract a second time. The Court of Appeals confirmed. In this sense, the Court of Appeals for the Eleventh Circuit might have created a precedent: each time a plaintiff presents his abstracted elements to the court, the court can take them as they stand and work on them for the filtration step. The Court of Appeals bases its reasoning on the fact that the burden of proof is on the plaintiff. The court goes even further in the argument:

"Perhaps the best approach for a District Court in any computer program infringement case, whether involving literal or non-literal elements, is for it to require the copyright owner to inform the court as to what aspects or elements of its computer program it considers to be protectable" (89 F. 3d 1548, at 1555).

[FN54]. Here is an explanation of Lotus's menu command hierarchy. There was a set of commands at the top of the screen, such as: File, Worksheet, etc. If you chose one of these commands, you would be presented with a set of options; e.g., if you chose "FILE", it might be: SAVE, LOAD, COPY, PRINT (all things that you could do with a file). If you chose "SAVE" and there was already a file with that name, you might get the options: REPLACE, CANCEL. Thus there was a hierarchy. You could express the whole thing as an inverted tree with various branches growing out of "FILE" and sub-branches growing out of these branches. To have an idea of the Lotus 1-2-3 UI, see D. Bainbridge, Software Copyright Law (1994), p. 98.

[FN55]. The later Bateman case sees no reason to make that difference. See above, in literal infringement cases.

[FN56]. Although the District Court said that many alternatives existed for each word, when the word signifies a function, it cannot be protected by copyright.

[FN57]. The Court of Appeals endorses the "Apple test". It found no virtual identity because the defendant Arce used icons for its UI as opposed to words used in the MiTek interface. The MiTek ruling is a synthetic application of Lotus and Apple rulings.

[FN58]. The reader can be referred to a few articles:
M. Bergner "Changing views: a comment on intellectual property protection for the computer user interface" (1998) 42 St. Louis L.J. 301.
J. M. Rolling, "No Protection, No Progress For Graphical User Interfaces" (1998) 2 Marq. Intell. Prop. L. Rev. 157.
B. G. Joseph, "Copyright protection of software and compilations a review of significant developments 1991-1998" (1998) 518 PLI/Pat 175. N. M. Tocups and R. J. O'Connell, "Proprietary Right Trademark and trade dress protection for computer software" (1997) 1410 Computer Law 17.
M. D. Carlson, "Intellectual property protection for computer icons: the trademark alternative" (1997) 31 U.S.F. L. Rev. 433.
M. J. Schallop, "Protecting user interfaces: not as easy as 1-2-3" (1996) 45 Emory L.J. 1533.
J. Myers, "Apple v. Microsoft: virtual identity in the GUI wars" (1995) 1 Rich. J.L. & Tech. 5.
N. P. Terry, "GUI wars: the Windows litigation and the continuing decline of 'look and feel' (1994) 47 Ark. L. Rev. 93.
M. P. Larvick, "Questioning the necessity of copyright protection for software interfaces" (1994) U. Ill. L. Rev. 187.
J. Houston, "A unified test for the copyright protection of user interfaces to computer programs" (1993) 32 Duq. L. Rev. 133.
J. C. Phillips, "Sui Genesis Intellectual Property protection for computer software" (1992) 60 Geo. Wash. L. Rev. 997.

[FN59]. M. Bergner points that trade dress and trade marks could be possible ways to afford a better protection of UIs. He then claims patent law or a new sort of petty patent (less lengthy and less costly to acquire) could be a solution to protect UIs. J. Rolling tends to prove that UIs can be adequately protected by trade dress and that it is the best suitable protection for UIs. M. Larvick thinks UIs should not be protected by copyright law because a strong copyright protection decreases innovation (M. Larvick, ibid., at 199, 201 and 202).

[FN60]. See M. A. Hamilton and T. Sabety, "Computer science concepts in copyright cases: the path to a coherent law" (1997) 10 Harv. J. Law & Tec. 230. The authors take user interface in its narrow sense (referring to the screen display), and cite relevant law in fn. 38: Registration and Deposit of Computer Screen Displays, 53 Fed. Reg. 21, 817-23 (1988); see also Atari Games Corp. v. Oman 888 F. 2d 878 (D.C. Cir. 1989); Apple Computer, Inc. v. Microsoft Corp. 799 F. Supp. 1006 (N.D. Cal. 1992); Broderbund Software, Inc. v. Unison World, Inc. 648 F. Supp. 1127 (N.D. Cal. 1986). See also Hamilton and Sabety, n. 36.

[FN61]. Karjala, n. 44 above, at 51 and 55.

[FN62]. The code, producing the end-result (the UI), is protected by the program copyright. Indeed, under this type of protection, it is the code itself generating the UI that will be protected and not the visible UI itself.

[FN63]. However, Karjala does not point out a difficulty inherent in U.S. law which does not exist in Europe: the U.S. system of registration. Under registration rules, a single registration protects the program, the screen display, and the user interface.

[FN64]. Mazer v. Stein 347 U.S. 201, 74 S. Ct 460 (1954). This case involved statuettes for use as bases for table lamps. Kieselstein-Cord v. Accessories by Pearl, Inc. 632 F. 2d 989 (2d Cir. 1980), which involved sculptured design belt buckles cast in precious metals.

[FN65]. The question of the qualification of user interfaces started with video games. Courts have qualified the visual displays as visual works and the code generating the visual display, as computer programs. The same question is raised for multimedia works, which are best qualified as visual works or even as databases, because they can mix text, images and sounds. See, e.g., A. Strowel and J.-P. Triaille, Le droit d'auteur, du logiciel au multimedia (1997), pp. 346-375. It is rather surprising that U.S. courts encounter difficulties with UIs while they did not have any in classifying video games interfaces in audiovisual works.

[FN66]. See, e.g., A. O. Martyniuk (1995), Comment: abstraction-- filtration-- comparison analysis and the harrowing scope of copyright protection for computer programs, 63 U. Cin. L. Rev. 1333.

[FN67]. The most valuable "element" in a software is its architecture, and its development is expensive and time-consuming. See also M. Grewal, "Copyright protection of computer software" [1996] E.I.P.R. 454.

[FN68]. See Karjala, n. 44 above, at 45-47. See Part I [2000] E.I.P.R. 7 at 12, n. 50.

[FN69]. See H. S. Amin, "The Lack of Protection Afforded Software Under the Current Intellectual Property Laws" (1995) 43 Clev. St. L. Rev. 19, 20, at 37. See also J. Cai, n. 16 above, at nn. 168 and 169.

[FN70]. See Martyniuk, n. 66 above, at 1372. See also Karjala, n. 26 above, at 15, and n. 16.

[FN71]. See Martyniuk n. 66 above, at 1374.

[FN72]. Gable, n. 14 above, at 484 and J. S. Wilkins, "Protecting Computer Programs as Compilations Under Computer Associates v. Altai" (1994) 104 Yale L.J. 435 at 455.

[FN73]. Judge Baer in Harbor Software used the compilation doctrine to protect the program structure. In EDI, the court found the input and output structure copyrightable as a compilation. The recent Softel case shows that the AFC test and the compilation doctrine can be reconciled and even applied in combination (Softel, Inc., v. Dragon Medical and Scientific Communications, Inc. 118 F. 3d 955 (2d Cir. 1997)).

[FN74]. Baca, n. 22 above, at 426.

[FN75]. See mainly Karjala, n. 44 above.

[FN76]. See Part 1 of this article [2000] E.I.P.R. 7 at 13, n. 57, and Gable, n. 14 above, at 486.

[FN77]. In re Beauregard 53 F. 3d 1583, 1584 (Fed. Cir. 1995).

[FN78]. Indeed by the time the patent will be issued, the technology claimed might already be outdated. In the United States it takes approximately three years for a patent to be issued.

[FN79]. See, e.g., Baca, n. 22 above, at 426.

[FN80]. Martyniuk, n. 66 above, at 1375.

[FN81]. See Bergner, n. 58, above.

[FN82]. Martyniuk, n. 66 above, at 1376 and R. H. Stern, "Is the Center beginning to hold in US Software copyright law?" [1993] E.I.P.R. 39. A conceivable solution would be a sui generis protection rather similar to the one enacted in the E.U. Database Directive, which rewards the time, efforts and investments put in the creation of a database. This would protect the most efficient ways of running a computer. A shorter and "weaker" protection would be given for certain "creative", non-standard, unusual algorithms.

[FN83]. See Martyniuk n. 66 above, at 1376; Gable, n. 14 above, at 471, however, finding the existence of an effective split among circuits, accepts that the Altai test

remains the leading method. See also Wilkins, n. 72 above, at 469; L. T. Oratz, "User Interfaces: Copyright vs. Trade Dress Protection" (1996) 13 Computer Law 1 at 3.

[FN84]. Based on a conversation with Filip Beernaert, Legal Adviser, Lernour & Hauspie, Speech Products, Ieper, Belgium, http://www.lhsl.com. Microsoft also secures its software licences by strong contracts. Moreover, as its software is developed mainly in the United States, the issues of non- literal copying are more likely to arise there. In Europe and worldwide, Microsoft fights against slavish copying or piracy. Based on a conversation with Didier Vermeersch, Microsoft Benelux and information provided by Ms Sharon Golec-Keniger, Corporate Attorney, Microsoft Europe.

[FN85]. Based on a conversation with Filip Beernaert, Legal Adviser, Lernout & Hauspie, Speech Products, Ieper, Belgium. http://www.lhsl.com.

[FN86]. See, e.g., Sybel Informatique v. Oxalead and Prolepse, Tribunal de Grande Instance de Paris, 3d ch., 2d sect., November 12, 1992 [1993] Expertises 109; Paris Court of Appeals, 4th ch. A, November 23, 1994, [1995] Expertises 36.) The court merely restates the conclusions of the expert, which found some identities between the programs. It does not further question if these identities are found in ideas, elements dictated by efficiency or industry standards, etc. The court finds for infringement on the sole conclusion the program was copied. Other cases illustrate this tendency: Agent judiciaire du Tresor et Trace v. Softmax, Paris Court of Appeals, 4th ch., May 31, 1995 [1995] Expertises 311; Marben GL v. Cao Diffusion, Meaux Commercial Court, December 17, 1996 [1997] Expertises 122; Engineering Systems International (ESI), Agence pour la protection des programmes (APP) v. Mecalog, Francis A. and Joseph Z., Paris Commercial Court, November 22, 1993 [1994] Expertises 32; Mecalog v. Engineering Systems International GmbH (ESI-GmbH), Citroen, Sogedac, PSA-Citroen, Etude et Recherche et l'Agence pour la protection des programmes (APP), Paris Court of Appeals, 4th ch., Sect. B., November 10, 1994 [1995] Expertises 32; Simci, Agence pour la Protection des Programmes and Sondatic v. Digimedia, Paris Court of Appeals, 4th ch., February 16, 1994 [1995] Expertises 240. For comments on Sybel and ESI, see X. Linant De Bellefonds "De la contrefacon de logiciel, enieme !" [1995] Expertises 27 and X. Linant De Bellefonds, note under Engineering Systems International (ESI), Agene pour la protection des programmes (APP) v. Mecalog, Francis A. and Joseph Z., Paris Commercial Court, November 22, 1995 [1994] Expertises 33.

[FN87]. Commercial court of Bobigny, 7th ch., 20 January 1995 (1995) 166 Revue Internationale du droit d'auteur 325. This decision has been affirmed by the Paris Court of Appeals, 4th ch., October 23, 1998 [1999] Expertises 31.

[FN88]. ESI v. Mecalog, Paris Commercial Court, November 22, 1993 [1994] Expertises 32; Simci v. Digimedia, Paris Court of Appeals, 4th ch., February 16, 1994 [1995] Expertises 240.

[FN89]. John Richardson Computers Ltd v. Flanders Chemtec, February 19, 1993 [1993] F.S.R. 497, C.L.Y. 587; Ibcos Computers Ltd v. Barclays Highland Finance Ltd [1994] F.S.R. 275 and Cantor Fitzgerald International and Cantor Fitzgerald Securities v. Tradition (United Kingdom) Ltd, Michael Howard and Christopher

Harland, High Court, Chancery Division, April 15, 1999, as yet unpublished, reviewed in [1999] E.I.P.R. N-142.

[FN90]. Karjala, n. 18 above, at 63-64.

[FN91]. In this case, a former employee was alleged to have developed a software which copied a software he previously elaborated at his former company. See also S. E. Gordon, "The very idea: why copyright law is an inappropriate way to protect computer programs" [1998] E.I.P.R. 10 at 12. For a review of the John Richardson and Ibcos cases, see, e.g., B. B. Sookman, "International differences in copyright protection for software" [1995] C.T.L.R. 142-151.

[FN92]. It is unclear from the decision what Jacob J. meant by "detailed idea". It is not certain that in his opinion only the details (which would amount to expression) are protected, while the more general idea is not.

[FN93]. Cantor Fitzgerald v. Tradition, n. 89 above, at 76.

[FN94]. See News Datacom Ltd, British Sky Broadcasting Ltd and Sky Television plc v. David Lyons trading as Satellite Decoding Systems, Satellite Decoding Systems & Co., Satellite Systems Ltd, Satellite Decoding System (1986) and Marie Molloy, High Court [1994] 1 I.L.R.M. 450 (January 20, 1994).

[FN95]. See, e.g., Novell Inc. v. Crommen, Correctionele Rechtbank (Criminal court), Hasselt, 13th ch., February 16, 1999 [1999] Intellectuele rechten-- Droits intellectuels 34; Microsoft Corp. v. Sycomor, Liege civil court (ref.), July 31, 1995 [1996] Auteurs & Media 151; Apple Computer Inc. v. SA Dobby Yamada Serra and SA, Rama Corp., Brussels Court of Appeals, October 14, 1993 [1993] Ingenieur-Conseil 358; Comm. Charleroi, January 19, 1993 [1993] Jurisprudence Liege Mons Bruxelles 1178.

[FN96]. Brussels Court of Appeal, 9th ch., Consulting & Development v. Argus Integrated Solution, February 4, 1999, unpublished, confirming Civ. Brussels, June 5, 1998, unpublished.

[FN97]. See Civ. Bruxelles, January 22, 1988 [1988] Revue internationale du droit d'auteur 363 and note by M. Buydens. The author points to other decisions (mainly Belgian and French) which use this dissection method, the everlasting problem remaining where to draw the limit between idea and expression.

[FN98]. OLG Koln, jur-pc 1992, 1409; OLG Hamm [1991] Neue Juristische Wochenschrift 2161; OLG Frankfurt a. M., NWJ-RR 1993, 230 = WRP 1993, 32-- Spielmodule; BayObLG [1992] G.R.U.R. 508 = [1993] J.Z. 105 m. Anm. U. Weber-- Verwertung von Computerspielen. Courts agree on the fact the player has the possibility to act on the sequence of images is without relevance as the changes in images are preprogrammed.

[FN99]. Atari v. Valadon, Tribunal de Grande Instance de Paris, December 8, 1982 [1983] Expertises 31; see also Paris Court of Appeals, June 4, 1984 and Court of Cassation, March 7, 1986; Tribunal de Grande Instance de Paris, September 1986 [1987] Expertises 107. Nintendo v. Horelec, Pres. of Tribunal de Premiere Instance,

December 12, 1995 [1996] Intellectuele rechten--Droits intellectuels 89 and Horelec and Sedimex v. Nintendo, Brussels Court of Appeals (9th ch.), April 11, 1997 [1997] Aueurs & Media 265.

[FN1]. 111S. Ct 1282, 18 U.S.P.Q. 2d 1275 (1991).

[FN2]. EDI v. SSI n. 43 above.

[FN3]. ibid., at 409.

[FN4]. n. 95 above. For comments, see J. P. Buyle, L. Lanoye, Y. Poullet and V. Willems, "Chronique de jurisprudence, l'informatique (1987-1994)" [1996] Journal des Tribunaux 205 at 220; A. Strowel and J.-P. Triaille, "Jurisprudence recente et questions d'actualite en matiere de protection de logiciels" (1993) Ingenieur-Conseil at 333: "The formulation used in the writing of the source code was mandatory so that there was no choice and therefore no possible originality".

[FN5]. See Strowel and Triaille, ibid., at 330. Apple Computer Inc. v. S.A. Dobby Yamada Serra and SA Rama Corp., n. 95 above. More recently, a court found that originality was apparent in the copyright registration certificates (see Pres. Tribunal of First Instance, Antwerp, March 2, 1999 [1999] Intellectuele rechten--Droits intellectuels 37).

[FN6]. Buchhaltungsprogramm (Bundesgerichtshof), July 14, 1993 [1995] International Review of Industrial Property and Copyright Law 127. Before the adoption of the Directive, the German Supreme Court had already started to review its originality requirement ("Schopfungshohe") for computer programs in a 1990 decision (Betriebssystem--BGH, Urt. V. 4.10.1990 [1991] Neue Juristische Wochenschrift 1231) which conforms to its posterior ruling in Buchhaltungsprogramm. The court decided that the originality level must be more than the purely routine or mechanical stringing together of material.

[FN7]. Even if the court did not reject explicitly its previous precedents requiring high levels of originality, it is reasonable to think that this is a "leading decision" in the application of the new standards set by the Directive. M. Lehmann, Comment on the Buchhaltungsprogramm decision [1995] International Review of Industrial Property and Copyright Law 133.

[FN8]. To summarise, two elements must be proven to show infringement in Germany: "the existence of an individual intellectual creation and a certain albeit small measure of individuality". See Lehmann ibid., at 134.

[FN9]. CA v. Faster and Altai n. 87 above. This decision has been affirmed by the Paris Court of Appeals, 4th ch., October 23, 1998 [1999] Expertises 31.

[FN10]. As translated in Revue Internationale du droit d'auteur n. 87 above, at 232.

[FN11]. Tribunal de Grande Instance de Creteil, January 13, 1998 [1999] Expertises 113.

[FN12]. As early as 1995 another French court adopted the same notion of originality singled out in CA v. Faster and Altai. See Agent judiciaire du Tresor et Trace v. Softmax, Paris Court of Appeals, 4th ch., Mai 31, 1995, [1995] Expertises 311: "[the author] made a personalised creative effort, going well beyond the simple application of an automatic and compelling logic ... ". See also Marc B., Infico v. Serosi, Jean-Claude D., Douai Court of Appeals, 1st ch., July 1, 1996 [1997] Expertises 155.

[FN13]. This excepts the present trend to protect software by patent, which can be considered as a complement more than an alternative. See above.

[FN14]. Logic, algorithms and programming languages should be protected by patent law or sui generis protection. They should not be granted copyright protection as it gives developers a double monopoly.

[FN15]. See, e.g., Karjala, n. 44 above, and Gordon, n. 91 above.
EIPR 2000, 22(2), 56-68

END OF DOCUMENT