BILETA

17th BILETA Annual Conference

April 5th - 6th, 2002.
Free University, Amsterdam.

The Role of Copyright in Protecting the Creativity of Programmers

(Euan Cameron, De Montfort University, Leicester)

Section 1 Some Thoughts on The Overall Problem

Introduction

This paper will be much concerned with definitions. The impact of globalisation and harmonisation on copyright through the Berne Convention, to European Directives and TRIPS means that the same words are used in many if not most jurisdictions. Yet subtle differences of interpretation can leave one with the impression that the U.K. is out of line with other jurisdictions at the most basic levels. It will be argued that such differences have an inhibiting effect on the creativity of programmers especially as they move from employer to employer in an industry that is a still noted for its rapid turnover of staff. With such an assertion and title, I should assay a definition of the word "creativity" itself. The Concise Oxford Dictionary does not take us far. It defines "creative" as "inventive and imaginative" and creativity merely by reference to that main definition as carrying the sense of the ability to be creative. In this paper I intend the word to carry the additional meaning of the freedom to use the imagination.

The traditional role of Intellectual Property and hence Copyright has been to encourage creativity by rewarding it with a monopoly of varying duration in respect of the product of the creative act. The mechanism for such reward, however is essentially negative - the placing of restrictions on others activities rather than any positive assistance. The question then becomes whether such restrictions interfere with the creative function in copyright. Where on this view is the potential for restriction to conflict with creativity? It is submitted that there are 3 foundations upon which the creative activity of the programmer may be built that could be restricted by the overzealous pursuit of "strong" copyright: 1) the ability to use one's own personal skill and knowledge; 2) restrictions upon the common tools of the trade; and 3) restrictions upon the use of accumulations of knowledge. In the case of 1) above such a right is recognised as a factor to be balanced in adjudging issues of confidentiality in respect of an ex employee but may become lost in the context of a copyright work. In the latter two cases, the issue is one of space and time - to what extent is copyright law going to require creative people to reinvent the wheel as opposed to being allowed to build upon the efforts of the industry. At first sight the reader might regard this as a plea for the use of "open" software or a call for the free use of tools developed by others without payment. But in this context, "tools" is used not only for separately created programmes and proprietarily marketed products but also for routines and "tricks" that are developed either in the industry as a whole or even by the programmer him- or her- self. What might be describes as "techniques" rather than products. At the macro level we know that major producers will attempt to fence off whole areas of programming skill. It happened when Apple laid claim in litigation to the "WIMP" environment for Graphic User Interfaces and recently we have seen the start of the litigation in the U.S. of BT's patent claims to hyperlinks. There are equal dangers at a micro level when an employer lays claim to the individual's skills. In the specific context of copyright where those tools and techniques are developed whilst working for one

employer, the danger comes from the fact that they are tied up within a program produced for the employer. Copyright law must find a way to separate such skill from the proprietary product. Perhaps the same point can be made in the less obvious context of databases especially those that are an integral part of the working of a program. It is after all of the nature of computers that they involve operations on a scale that cannot be carried in the programmer's head. The use of old work as an "aide memoire" to the new is necessary to the freedom of movement of programmers.

Programming Style

These comments need to be placed in the context of the changing nature of programming over the years. The linear model of 2nd and 3rd generation languages meant that it was possible to imagine a relatively straightforward and unified structure to a program as encapsulated in a flow chart. Although the view could be criticised in detail, it would not be entirely unrealistic to imagine a single creative effort flowing from start to finish. Such an image becomes less realistic in more modern languages where overall programs are broken down into a more modular structure. Particular tasks are performed by the coding of self-contained routines and assigned a role within the program as a whole. If concentration is given to such routines then the job of programming becomes one of assembling such routines to carry out the necessary tasks and of providing the means for the overall program to call upon individual modules / routines at the appropriate time.

Such routines may be written afresh by the programmer or may be taken or modified from a "bank" of such routines. That bank may consist of the programmer's own collected store of solutions to performing particular tasks, built up either during employment or even before it. Alternatively it may be part of the common pool of knowledge within a particular company. It is also quite likely to be derived in part from documentation provided by the owner of the programming language itself. This could include routines for particular expected tasks that are included within the languages' manuals or, for example, upon a website containing FAQs or providing a discussion forum for a common interest group of users. Part of the task of the programmer may include the adaptation of such published "common" material to the needs of a particular program. It must be appreciated that such tricks or techniques of the trade available in a "bank" extend not only to the routines themselves but also to the methodologies used to link them and make them work efficiently together.

By far the most realistic of the recent U.K. cases, is the case of Cantor Fitzgerald[1], discussed below. Of present significance is one passage where the judge is criticising an expert relied upon by Cantor Fitzgerald. The example given by the Judge to illustrate his criticism involved approximately 10 lines of code which were very similar to those in the allegedly infringing program and which were used by Cantor's expert to suggest copying. However the similarity could be explained because such coding is shown in the Systems Services manual for the language being used. Thus the evidence did not suggest the copying hypothesised. By suggesting that this represents a realistic view, I mean that it is a case that acknowledges the way in which a program is constructed from modules that may be derived from various pre existing sources rather than from some species of entire immaculate conception.

Creativity

From the above it is submitted that creativity can only flourish if three concepts are promoted:

"reference" - this is an important concept in art and literature. Much if not most great art and literature has progressed by incorporating reference to previous works: a "code" for the knowledgeable. It is one of the worries of the tendency towards "strong" copyright that it ignores such traditions. Likewise a science such as programming develops by building upon the work of others - or, in the context of a former employee, one's own previous work. At the least one should be cautious in cutting off the use of such work as an "aide memoire." This, it is submitted was a key issue in Cantor Fitzgerald International v Tradition (UK) Limited,. The programmers had access to

their work for their previous employer when writing the allegedly infringing programs. Clearly such reference to an earlier work provides a trail for the purposes of causation. But it will be suggested that too much emphasis is put on causation in the U.K. in adjudging copying. There is the risk that the mere fact of referring to one's own previous work will be regarded as sufficient proof of infringement. Is consultation to be equated with copying?

Industry standard solutions should be available to all.

The merely commonplace should not be regarded as personal to a particular program and hence the property of an employer who owns a program incorporating some commonplace idea.

Obviously 2) and 3) are of less import to the individual programmer than 1) except in the sense that the potential for copyright protection being given to such incorporation may have an inhibiting effect on the style of composition and hence creativity. Nevertheless they have been key issues in the great battles for program copyright and the desirability or otherwise of "strong" copyright. It is a battle highlighted in the differing policy approaches taken by the U.S. judges in the Lotus[2] and Apple[3] cases. Judge Keeton in Lotus extolled the virtues of forcing new program developers to seek new ways of expressing the same idea so that progress can be achieved in the search for difference. He argued that such "strong" copyright was the way to strengthen the industry as a whole. By contrast, Judge Walker in Apple argued that this could involve a wasteful duplication of effort which imposes an unnecessary cost upon new creation. And of course, the tenor of the outcome in the Apple litigation was that there must be limits upon the degree to which one company can place a fence around the common heritage of the industry. It is difficult to see how such ideas can impact on U.K. law The case of Ibcos specifically rejected the attempt in John Richardson Computers v Flanders to import such concepts into U.K. law. In a recent article Professor Samuelson[4] has pointed out that the American courts and indeed their copyright structure is much better equipped to engage such policy considerations. It is precisely because of this that one fears the causation-led U.K. approach to infringement.

Before, During and After

The discussion thus far shows that the restrictions on programmers may be divided into the internal and the external. By external influences I mean those that are imposed on the industry as a whole, restricting creativity by fencing off an industry standard or commonplace technique. But this paper is more concerned with the internal - restrictions upon the use of the programmer's own skill and knowledge when someone else has acquired a claim to a work that embodies such skill and knowledge through some relationship with the programmer. It is suggested that such a relationship could take one of three forms: collaboration, commission or employment. The question to be asked in each case is how much can be taken from the relationship when it ceases? Or, to turn it round, how much freedom exists to use knowledge acquired during the relationship? In looking at these questions one might take a simple model of before, during and after the relationship. e.g. Before: how much of that which has been brought to the relationship by the programmer survives until after the relationship and how much is subsumed into work during the relationship? e.g. During: how much credit is to be given for the work that is done during the continuation of the relationship? e.g. After: to what extent is the future employment and method of working of the programmer restricted by the other party to the relationship.

Immediately one sees that these are not mutually exclusive divisions. For example, issues as to what restrictions are created at the end of an employment are mirrored by limits on what can be brought into a new employment. It is particularly difficult to isolate issues peculiar to the continuance of the employment from disputes that occur after the employment. For the commissioned programmer there are questions of ownership that are resolved, subject to contract and assignment, in favour of the programmer as author. For the collaborator, then the issues are those of potential co-authorship or separating an apparently unified work in to the separate efforts that are required for separate

authorship. But for the employee these issues are largely academic unless the contract of employment reverses the presumption of the employer's ownership contained in s11(2) Copyright, Designs and Patents Act 1988 (CDPA). Further the half-hearted attitude of the U.K. legislators to providing moral rights for the actual creators of works is especially prevalent in the case of the employed programmer. Not only have they no right to assert their authorship (s77 CDPA 1988) because of their employment (s79(3)(a) CDPA 1988) but in fact the entire class of "computer programs" are excluded from this moral right (s79(2)(a) & (c) CDPA 1988). Likewise the right to object to alterations and other `derogatory treatment' (s80 CDPA 1988)is removed from all programs by s81(2). It is as if there is a legislative denial of a programmer's creativity. At the least, there is an assertion of the primacy of capital and ownership over creativity in this field.

Again most of issues in all three phases identified above will be adjudged with the benefit of hindsight at the end of a relationship. The three "great" programmer cases[5] in the U.K. [Cantor Fitzgerald v Tradition[6]; John Richardson Computers v Flanders[7]; and Ibcos v Barclay's Mercantile[8]] all involved post employment situations and looked back to the employment or even earlier. The issues in Ibcos include the question of the survival of pre-employment work from a post employment perspective while John Richardson Computers considered work undertaken as an independent contractor for someone who had previously been an employee. From this perspective one might almost regard Ibcos as a stereotypical sequence of events, at least in the 1970s and 80s. The Defendant, Mr Poole, had developed an accounts payroll package whilst working independently. He then became involved in an informal partnership with Mr Clayton to adapt his idea to the agricultural machinery business. This later became company, PK Computers Ltd, for which the copyright work was written incorporating Mr Poole's work - so what had happened to any earlier copyrights that he owned? Poole then drifted away from the company becoming first a consultant and then totally independent at which stage he produced the allegedly infringing program. So one can see a rhythm to the relationship of informality drifting into employment where, perhaps, the terms of employment relating to matters such as copyright are still less than explicit, followed ultimately by a parting of the ways and potential conflict. In John Richardson, the origins of the allegedly copied program were with the company's owner but some of the work was done by independent contractors as well as the defendant during his employment. The question is thus what became of their contributions given that there is no general principle of ownership being given to a commissioner.

By contrast, in Cantor Fitzgerald the independent skill and knowledge was developed by the programmers within the employment relationship and could only be separated from the effort due to the employee if it was in some sense exceptional and thus out of the employment. Even with employees working way over the traditional 40 hour week, the Courts will be unlikely to see such effort as being outside the employment contract. However the case does highlight one important similarity between the three cases of crucial significance to the theme of creativity. In each case when one looks at the skill and knowledge of the programmer, one is talking of specialism not only in programming technique but also in a particular field of application - in Richardson, a program for organising a pharmacy; in Ibcos, programs dealing in agricultural machinery; and in Cantor Fitzgerald, the programs involved systems for creating an automated market in bonds. From the judgement in the latter case, it is clear that one of the programmers, Mr Guppy was to quote the judge "a very good computer programmer ...perhaps outstandingly good." Of equal importance to the present theme Mr Guppy became very knowledgeable about the needs of the brokers and in turn the requirements of programs to meet those needs. Of course, such a programmer has transferable skills, to use the despised jargon, as a programmer but he also had personal skills as an analyser of the computing needs of a broker market. While the acquisition of such knowledge was held to be within his duties to the company, we can see that the boundary between the personal skill and company property is in danger of becoming blurred.

Section 2 An Outline of Some of The Specific Problems

It has been argued thus far that the U.K. approach to copyright has a stifling effect upon the way that ex employee/programmers work. Firstly because the U.K. approach to substantiality and infringement fails to discount sources that are not part of the authorship of the particular program. Secondly because there is a tendency to regard the employee's own skill and knowledge as subsumed within the product owned by the employer. In this section it is intended to summarise some of the factors that create this situation in the U.K.. These include:

The issue of the ownership of material brought to the relationship by the programmer.

The U.K. approach whereby similarity and causation are regarded as sufficient to prove substantial copying and hence infringement.

There is doubt as to what exactly is meant by a "work" especially in the context of a program or a module/routine the "work" is the most basic of all copyright concepts yet there is little discussion of it at a theoretical level.

In program copyright another basic concept that is frequently misused is that of "compilation"

The consequence of these factors is an approach whereby there are conflicting tendencies for the discussion of causation to look at similarities in small portions of the program while for ownership and protection the Courts look to the overall package irrespective of source.

Ownership of material created prior to relationship

The question here is what happens to independently authored material that is brought by the programmer to the relationship. There is no doubt that the U.K. Courts have found some difficulty with this situation. This is mainly because there appears to be a desire to find that an ex-employee/programmer cannot hinder the employer in his exploitation of a venture that has been fulfilled during employment. Various approaches have been suggested to achieve this but some of these, particularly those suggested in John Richardson Computers and in Ibcos, cannot be regarded as satisfactory. In Richardson it will be remembered that the judge was dealing with work brought in by independent contractors. He suggested two ways in which the copyrights could be subsumed within the final product. First it is suggested that there is an implication of assignment or equitable ownership in favour of the producer of the work. Secondly it is suggested that the earlier work's copyright ceases to exist once it is included in a later, larger work. The same two suggestions seem to be made in Ibcos, albeit in a slightly different form and in the context of an employee as opposed to an independent contractor.

The idea of an earlier work being subsumed within a larger work is surely a non starter. The European Software Directive and the CDPA 1988 s 3(1) (c) as amended to take into account the Directive, both make it clear that preparatory work in the creation of a computer program is itself a copyright. This is certainly not the less true because the programmer happened not to be employed by the ultimate owner of the program at the time of the preparations? A fortiore, the same argument would apply if the "preparations" were themselves programs.

Further, the idea that various versions attract separate copyrights has been dramatically affirmed in the recent case of Sweeney v MacMillan. Publishers Ltd[9]. This case arose out of the tortured publishing history of James Joyce's book "Ulysses". Partly because it was banned in various countries during his lifetime and partly because it existed in many different versions because of its manner of composition, some scholars have come to believe that the version published during his lifetime needed to be modified to take into account other versions and manuscripts that remained unpublished at his death. These revisions were first brought together in an edition long after his

death to produce a "correct" version. For present purposes one could regard the version published in his lifetime as out of copyright but it was held that the unpublished material which attracted copyright at the time of publication, remained in copyright[10]. The idea that preparatory writings and manuscripts were to be regarded as a part of the published version was explicitly rejected. There are difficulties with this decision in terms of the stifling of scholarship and of the possibility of the continuos reviving of copyrights but it certainly rejects the view of drafts being subsumed within a later work. Further in Ray v Classic FM plc[11], Lightman J castigated, as a "heresy," the suggestion that where preparatory copyright material was included within the intended jointly owned product then the earlier copyrights could be regarded as subsumed within the later one.

Thus one is left with the issue of an implied equitable assignment or ownership of the previous work. There are inevitably practical difficulties with this sort of idea. . Following the decision of the House of Lords in Liverpool City Council v Irwin [12], the courts must adopt a cautious approach and avoid making agreements with the benefit of hindsight, which the parties would not have contemplated themselves. The evidence accepted in Ibcos to justify such an inference was the agreement to terminate the programmer's employment. To back date that agreement to have an effect at the start of the employment seems to be dubious logic. The reason given by the judge is unconvincing. He states "it would be necessary to imply a licence to use those copyright works. The Court will not, unless there is no other reasonable construction, imply a term to make sense of the contract as a whole" With respect it should be easier to imply a license than a transfer of ownership. As shown in Ray v Classic FM, the implication should represent the minimum necessary to meet the expectations of the parties. Here the *maximum* required is a license that is irrevocable for inclusion of the previous work in the program created for the employer

Similarity Causation and Substantial copying

The approach taken by Jacob J in Ibcos Computers Limited v Barclays Mercantile Highland Finance Limited (supra) has gained a credence beyond the confines computer programming and been cited in a number of later cases .He suggested the following tests:

(1) What are the work or works in which the plaintiff claims copyright?

(2) Is each such work 'original'?

(3) Was there copying from that work?

(4) It there was copying, has a substantial part of that work been reproduced?

There is little doubt that this does encapsulate the typical approach to infringement in the U.K. Courts. However the present author has pointed out on a number of occasions that much of what was said in Ibcos was obiter in the sense that the facts of the case were extremely clear that copying had occurred. The key feature for present purposes is that substantiality is adjudged in these tests by looking at the work as a whole, not just those elements of it that show originality. This allows for protection of non-original elements within the work including those brought to the table from the programmers' own earlier work and general industry knowledge. Thus the usual British emphasis on issues of substantiality is maintained by contrast with the U.S. approach of identifying the original elements in the work as those which are to be protected.

This is connected to the approach to the relationship between causation and substantiality. The recent House of Lords decision in Designers Guild Ltd v Russell Williams (Textiles) Ltd[13], has emphasised the traditional British approach to equating the two in a somewhat explicit manner. It is at least arguable that infringement will be easier to prove in future. The case concerned a fabric design produced by the defendant that allegedly infringed the copyright of the plaintiff. The trial judge, relying primarily on the similarities between the two designs, concluded that there had been

substantial copying. The Court of Appeal concluded that the parts that had been copied did not amount to a substantial part. However the House of Lords reinstated the first instance decision, partly because they regarded issues of substantiality to be intimately connected to issues of evidence and therefore matters in which the appellate Courts should not readily interfere with the decision of the Judge. The attitude is perhaps best summed up in a passage from Lord Scott where he says: "The test proposed in *Laddie* [14] (pp 92-93 (para 2-108)) to determine whether an altered copy constitutes an infringement is: `Has the infringer incorporated a substantial part of the independent skill, labour etc contributed by the original author in creating the copyright work ...?'
My Lords, I think this is a useful test, based as it is on an underlying principle of copyright law, namely, that a copier is not at liberty to appropriate the benefit of another's skill and labour"

If I may paraphrase this, his Lordship is saying that the Laddie test is useful but we (the House of Lords) are much more concerned in looking for copying than in deciding what is original skill and labour. A little bit later in his judgement he states "where the finding of copying is dependant, in the absence of direct evidence, upon the inferences to be drawn from the extent and nature of the similarities between the two works, the similarities will usually be determinative not only of the issue of copying but also of the issue of substantiality".

The possibility of those similarities coming from the programmers own store of knowledge or previous work or from common industry practice are all ignore in this formulation.

What is meant by a "work"?

Logic would suggest that this would be a crucial issue in adjudging substantial taking and therefore by inference the extent of the programmer's effort that is "off limits" to him / her upon leaving employment. Cynicism might suggest that the lack of discussion is due to the British attitude that all that is derived from an earlier program must involve substantial taking. Earlier in this piece I discussed the way in which programs are assembled from a number of separate routines and modules. However the boundary between a module and a program is hard to see. It is essentially a matter of convenience. However if each module is regarded a separate copyright work, then the taking of a substantial part of just one module would amount to infringement even if it was derived from the "bank" of modules available from other sources. If one looks at the programs as a whole then there is room to take into account many factors to judge the "quality" of that which is taken. The present author has previously commented favourably on the Australian case of Powerflex Services Pty Ltd v Data Access Corporation (1997) [15] and the American case of Baystate Technologies. Inc. v. Bentley Systems, Inc. (United States District Court for the District of Massachusetts (1996)) [16]. Those cases reject the idea that a separate copyright attaches to each and every constituent part of a program so that the taking of any one part amounts to an act sufficiently substantial as to amount to infringement in itself. In the U.K. the position is much less clear. Ibcos, as one would expect, seems to take the view that one can separate the various modules into both separate and accumulative copyrights. This is tied to the view that computer programs can be regarded as compilations in the sense of being assembled from separate components (see below). By contrast Cantor Fitzgerald sends out some mixed messages. Pumphrey J concludes: on this issue: "So in my judgement the substantiality of what is taken has to be judged against the collection of modules viewed as a whole." On the other hand his limited findings of infringement are based upon an analysis of various individual modules. There is an ambiguity in the words he uses to conclude that one of the programmers has infringed: "Substantiality is to be judged in the light of the skill and labour in design and coding which went into the piece of code which is alleged to be copied...(Mr Gilbert)[17] still appropriated a substantial part of the skill and labour which went into the CFI modules". The phrase still seems ambiguous. What is "the piece of code which is alleged to be copied."? Is it the coding of each individual module or the whole of the programs written by the team or something in between?

Substantiality and Compilation.

"Compilation" is an inherently ambiguous word given its connection to the equally multi defined "compile". It carries many definitions. In computing it has of course acquired a specialised meaning - to produce a machine coded version of a program from a high level language. In copyright, it is another of the basic terms in U.K. law that is not defined. This has left some potential for using different connotations of the word in new contexts. Generally one would assume that it carries the idea of bringing together disparate individual works. However it is often used in every day language to deal with the bringing together new material as in "compiling a report" or "compiling data" or even "compiling a crossword" In this sense, the word has no precise meaning. Everything creative is thus compiled and it would be hard to think of a copyright work that was not compiled. It is submitted that this cannot be the meaning of the word in s3(1)(a) CDPA 1988. However there was a hint of such a broad meaning in Sweeney v Macmillan where it was suggested of the way that Joyce constructed his novel: "it is a long and extremely complex work, and was written over a lengthy period. Nor was it all written in anything like a straightforward narrative process or style. Especially in the later sections the process seems to have been more one of compilation, from many and various external sources" Fortunately this does not seem to have influenced the ultimate judgement or decision.

It is submitted that the normal meaning does involve the notion of separate works being brought together. It is further suggested that the idea from Ibcos that programs are per se compilations is a dangerous one. The program `compilation' idea seems to refer to the manner in which a program will be created by putting together various modules or subroutines. The compilation idea could be relevant in two ways: either by allowing protection when individual modules are derived from non-original sources, or by giving protection to the structure of the piece. It is its use in this latter sense that seems to predominate in commentaries upon the nature of computer program copyright but it is equally dangerous in the former sense in that encourages a view of each routine as a separate protectable work.

Conclusion

The overall message is that that portion of the programmers skill and knowledge, that is encapsulated in programs written for an employer, is likely to be denied him for future use. In the law of confidentiality, some care is taken to protect the programmer's freedom of employment by allowing the ex employee the use of his / her expertise as long as it does not involve the disclosure of trade secrets (whatever they are) - see Faccenda Chicken v Fowler[18]. But in copyright the opposite approach seems to have held sway. From the very early case of Northern Office Micro Computers Ltd v Rosenstein[19] the attitude has been that it is not proper to allow a programmer to make use of work done for a former employer as a reminder or aide memoire in doing new work. In that case it was demanded that an allegedly lawful copy of a program should be returned to the employer although there was no evidence of any intention to copy. Apparently in the U.K. reference means derivation means causation means substantial copying means infringement. The fact that those similarities come from other causes such as the industry's common heritage or from the programmer's own independently created program or method of working[20], seems to count for naught in the scheme of things.

---

[1] Cantor Fitzgerald International (an unlimited company) and another v Tradition (UK) Limited and others [2000] FSR 95

[2] (1990) 740 F. Supp. 37

[3] Apple v Microsoft (1992) 799 F. Supp 1006

[4] [2001] European Intellectual Property Review 409 "Economic and Constitutional Influences on

Copyright in the United States"

[5] These are normally seen as program copyright cases but the present perspective shows them to have much to say about programmer employment.

[6] See above note i

[7] [1993] FSR 497

[8] [1994] FSR 275

[9] Unreported Nov 2001

[10] The position is more complex than this suggests as it involves transitional arrangements as to unpublished material. Under former U.K. law, such material's copyright term ran from the time of first publication. Eventually all works will be judged from the time of the author's death.

[11] Chancery Division, [1998] FSR 449

[12] [1977] AC 239, [1976] 2 All ER 39

[13] [2001] FSR 113

[14] Laddie, Prescott & Vitoria, The Modern Law of Copyright and Designs (2nd ed. (1995)

[15] [1997] F.C.A. 490

[16] (1996) 946 F. Supp. 1079, both cases are discussed by the present author in (1998) International Review of Law Computers and Technology Vol 12 No1 p161

[17] Mr Guppy, the programmer mentioned earlier had adopted methods of working that meant that he had largely avoided infringing whereas Mr Gilbert was the programmer who had made too much use of reference their previous programming on behalf of Cantor Fitzgerald.

[18] [1986] 3 WLR 288

[19] [1982] FSR 124 (South Africa)

[20] In fairness to the Judge in Cantor Fitzgerald, he did take into such matters in dealing with Mr Guppy who was held not to infringe because similarities in his modules arose not form copying or even "referring" but from his style of working in the light of industry standards.