5th International Conference on Information Law and Ethics 2012
Equity, Integrity and Beauty in Information Law & Ethics

TITLE: Copyright in Computer Programming Languages

Yin Harn Lee
PhD Candidate
Faculty of Law, University of Cambridge

Introduction

The issue of whether programming languages are and ought to be protected by copyright has become much more salient of late. This very question was addressed for the first time by the CJEU in the recent case of *SAS Institute Inc v World Programming Ltd* ('*SAS v WPL*').[1] Its decision, handed down in early May, has had an immediate, international impact: counsel on both sides in the ongoing *Oracle America, Inc v Google Inc* litigation in the US have been ordered to provide submissions on it in the context of whether the vocabulary and grammar of a computer language, as distinct from programs written in that language, be protected by copyright.[2]

Prior to this, although the issue had garnered some degree of academic comment,[3] in practice, it had yet to result in any sort of protracted legal controversy, at least in Europe;[4] the software industry appeared to have operated under the general assumption that any programming language could freely be used by anyone.[5] The validity and prevalence of this assumption must now be doubted, however; while it remains true that no attempts have thus far been made to restrict the use of general-purpose programming languages such as C++ and Java,[6] the assertion of copyright in its

[1] Case C-406/10.

[2] Request for Further Phase One Briefing re Copyrightability of SSO, *Oracle America, Inc v Google Inc* (3 May 2012); Google's May 10, 2012 Copyright Liability Trial Brief, *Oracle America, Inc v Google Inc* (10 May 2012); Oracle's May 10, 2012 Brief Responding to Court's Questions on Copyrightability (10 May 2012).

[3] Following the implementation of the Software Directive, which expressly mentions programming languages, many textbooks on copyright now contain at least some small discussion on the copyright status of programming languages. For examples from the UK, see H Laddie et al, *The Modern Law of Copyright and Designs* (3rd edn, LexisNexis Butterworth, 2008), para. 34.19; D Bainbridge, *Legal Protection of Computer Software* (5th edn, Tottel Publishing, 2008).

[4] In Europe, the only pre-*SAS* case to deal directly with copyright in programming languages appears to be *Navitaire Inc v Easyjet Airline* [2005] ECDR 17, a decision of the UK High Court. In the US, the issue has been canvassed in somewhat larger number of cases: see *Lotus Development Corp v Paperback Software International* 740 F.Supp. 37 (D.Mass 1990); the *Sun Microsystems Inc v Microsoft Corp* litigation, which was commenced in 1997 and settled in 2001; and the ongoing *Oracle America Inc v Google Inc* litigation. The question of whether copyright subsists in programming languages was raised in each of these cases, and has been discussed most extensively in the *Oracle v Google* litigation, though in none of these cases was it directly at issue. For the position in Australia, see *Data Access v Powerflex Services* [1999] HCA 49; (1997) 75 FCR 108; (1996) 63 FCR 336.

[5] See RH Stern, 'Copyright in Computer Programming Languages' (1991) 17 *Rutgers Computer & Technology Law Journal* 321, 322 – 323, 346 (noting the existence of this shared assumption).

[6] cf the *Sun Microsystems Inc v Microsoft Corp* litigation, where counsel for Sun Microsystems, in the course of argument, asserted that Sun Microsystems was indeed claiming copyright in the Java programming language: see the transcript of the proceedings, excerpted in MP Doerr, 'Java: An Innovation in Software Development and a Dilemma in Copyright Law' (1999) 7 *Journal of Intellectual Property Law* 127, 157 – 159. A general-purpose programming language refers to a language designed to be used for writing programs in a wide

programming language by the claimant in *SAS v WPL* may be indicative of a general shift towards a more proprietorial attitude on the part of software developers who have successfully created and marketed programming languages designed for specific purposes and, perhaps more importantly, specific platforms.

Although the decision of the CJEU in *SAS v WPL* has has provided some clarification as to the relationship between programming languages and the scope of the protection conferred on computer programs by the Software Directive,[7] it leaves open the wider question of their status under the law of copyright generally. This, however, is only to be expected, given the circumscribed nature of the questions referred to the CJEU. The aim of this paper, then, is to examine the legal and policy considerations surrounding the issue of copyright protection for programming languages, and to draw out the arguments for and against the conferment of such protection. Part I sets out a definition of the term 'programming language' for the purpose of this paper, and identifies the specific uses which software developers in a similar position to the claimant in *SAS v WPL* are seeking the right to control. Part II discusses the decision of the CJEU in *SAS v WPL* and its implications on the copyright status of programming languages. Part III considers whether programming languages are capable of fulfilling the statutory prerequisites for copyright protection, including whether a programming language can be said to be a 'work' that is the product of its author's 'own intellectual creation', as well as the meaning of 'infringement' in this context. Part IV examines the potential consequences of conferring copyright protection on programming languages, including its possible effects on software users, competing software firms, and the progress of technological development as a whole. The paper concludes that there are sound legal and policy grounds which render doubtful the possibility of copyright protection for programming languages.

Part I Delimiting the scope of the debate

Defining 'programming language'

The term 'programming language' may be taken to refer to a formal language used to express computer programs, and which consists of: (i) a set of vocabulary elements; (ii) a set of syntax rules for combining vocabulary elements into statements; and (iii) a set of semantics, or the assignment of meaning to statements that properly combine vocabulary elements in accordance with syntax rules.[8] This definition is certainly wide

---

variety of application domains, and which for this reason does not include language constructs designed to be used within a specific application domain.

[7] Formerly Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs, now consolidated as Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs.

[8] Stern (n 5), 327. Other authors have made use of similar definitions which refer essentially to the same elements: see P Samuelson, T Vinje and W Cornish, 'Does Copyright Protection under the EU Software Directive Extend to Computer Program Behaviour, Languages and Interfaces?' (2012) 34(3) EIPR 158, 162 (stating that programming languages consist of 'a vocabulary, a set of semantics and a syntax'); D Hunter, 'Mind Your Language: Copyright in Computer Languages in Australia' (1998) 20(3) EIPR 98, 98 (stating that computer languages, like human languages, possess 'a set of words which denote certain things', as well as a grammar or syntax 'which define the way in which these words may correctly be connected to form proper sentences'); MA Hamilton and T Sabety, 'Computer Science Concepts in Copyright Cases: The Path to a Coherent Law' (1997) 10(2) *Harvard Journal of Law & Technology* 239, 265 (describing computer languages as being composed of 'a set of grammar rules and a set of symbols').

enough to encompass languages in which computer programs are written, the best-known of which include BASIC, Fortran, C++ and Java.

Some authors have argued that the term 'programming language' should also extend to so-called 'command languages', namely the keystrokes, input formats and command words (e.g. the commands 'Print', 'Move' and 'Copy' in a spreadsheet program) used for interacting with a computer program. Stern, in particular, argues that these sets of keystrokes and command words fulfil all three elements in the definition of a programming language: the set of all permissible command words comprises its vocabulary, the sequence in which they must be input is its syntax, and the instructions that use the prescribed vocabulary in accordance with the relevant syntax rules have an assigned meaning.[9] The approach taken by Karjala leads to a similar outcome: he argues that '[a]ll [user] interfaces are essentially programming languages', being sets of rules that give semantic meaning to groups of symbols and their syntax and permit a computer to function in the desired manner; based on this reasoning, no functional distinction can be drawn between command languages and programming languages as such.[10] Other authors, however, take the opposing view. Bainbridge, for instance, points out that while a command language does enable a user to interact with a computer program, this interaction does not result in the creation of a separate, discernible program; for this reason, he concludes that while a user command set may be termed a 'computer language', it cannot be appropriately characterised as a 'programming language'.[11] Lowry, meanwhile, states that '[t]here are two basic types of computer languages: programming languages and command languages',[12] thus indicating a conceptual distinction between the two, although both are to be considered as subsets of the wider category of computer languages. The Institute of Electrical and Electronics Engineers also appears to favour the view that programming languages and command languages should be treated as separate concepts. Its *Standard Glossary of Software Engineering Terminology* contains separate entries for each of these terms, with the former being defined as 'a language used to express computer programs'[13] and the latter as 'a language used to express commands to a computer system';[14] 'computer language', meanwhile, is defined as 'a language designed to enable humans to communicate with computers'.[15]

A consideration of the judicial approaches which have been taken in various jurisdictions is inconclusive. In *Navitaire Inc v Easyjet Airline* ('*Navitaire v Easyjet*'),[16] the UK High Court held that a collection of user commands and its syntax amounted to a 'computer language', a concept which it appeared to equate with the term 'programming language' as used in the Software Directive. In contrast, the Massachusetts District Court in *Lotus Development Corp v Paperback Software International* rejected as a 'word-game argument' the defendants' contention that the menu structure of the Lotus 1-2-3

[9] Stern (n 5), 328 – 330.
[10] DS Karjala, 'Copyright Protection of Computer Software in the United States and Japan: Part 1' (1991) 13(6) EIPR 195, 199.
[11] Bainbridge (n 3), 71 – 72.
[12] EG Lowry, 'Copyright Protection for Computer Languages: Creative Incentive or Technological Threat?' (1990) 39 *Emory Law Journal* 1293, 1298.
[13] IEEE, *IEEE Standard Glossary of Software Engineering Terminology* (IEEE Std 610.12-1990, IEEE, 1990), 59.
[14] IEEE (n 13), 17.
[15] IEEE (n 13), 19.
[16] [2005] ECDR 17; [2004] EWHC 1725 (Ch). Bainbridge considers the view taken by the court in this respect to be 'not beyond doubt': Bainbridge (n 3), 71 – 72.

spreadsheet program, in particular the choice of user command terms and the structure and order of those terms, amounted to a 'language' that was not capable of being protected by copyright.[17] As a possible consequence of this, in a subsequent case which also involved the copying of the Lotus 1-2-3 menu command hierarchy, the argument that it might amount to an unprotectable language was not invoked; instead, the case was argued and decided on the basis that the menu command hierarchy, including the user command terms, constituted a user interface.[18]

In this regard, the argument can indeed be made that the command terms used to interact with a computer program would be more aptly characterised as part of its user interface rather than a programming language, particularly when they are considered as a mode of input that is complementary or alternative to the now-ubiquitous mouse or touchpad. Nevertheless, for the sake of comprehensiveness, command languages will be included within the scope of programming languages for the purposes of the present discussion.

The nature of the rights sought

In addition to the broad question of whether programming languages are or ought to be protected by copyright, one also has to address the more practical, related question of the acts which the holder of the copyright in such a work would be protected against. In other words, if copyright does indeed subsist in a programming language, what are the acts which the rightholder is exclusively entitled to carry out and, by the same token, to prevent third parties from carrying out, in respect of that language?

Scholars who have examined this question appear either to conclude or implicitly assume that the developer of a new programming language would have be predominantly interested in preventing unauthorised third parties from: *first*, creating new computer programs which are capable of interpreting and executing programs written in that language; and *second*, writing their own computer programs in that language.[19] The first scenario, as we will shortly see, mirrors precisely the facts which gave rise to the dispute in *SAS v WPL*. The identification of the specific uses which would-be copyright owners of programming languages are seeking to control will prove to be pertinent to two aspects of the present analysis. The first relates to the legal question of whether such uses would amount to infringement in the event that programming languages are demonstrated to be works capable of being protected by copyright law. The second is whether the restriction of these uses in the manner sought would be a desirable outcome in the light of wider policy considerations.

[17] 740 F.Supp. 37 (D.Mass 1990), 72.

[18] *Lotus Development Corp v Borland International Inc* 49 F.3d 807 (1st Cir. 1995) aff'd 516 US 233 (holding that the Lotus 1-2-3 menu command hierarchy was a 'method of operation' by which users were able to operate the program, and consequently that it could not be protected by copyright under §102(b) of the US Copyright Act of 1976).

[19] DE Phillips, 'XML Schemas and Computer Language Copyright: Filling in the Blanks in Blank Esperanto' (2001) 9 *Journal of Intellectual Property Law* 63, 84; Stern (n 5), 347 – 348; Doerr (n 6), 133; Lowry (n 12), 1339.

Part II *SAS Institute Inc v World Programming Ltd*

The claimant in this case, SAS Institute Inc ('SAS Institute'), is the developer of an integrated set of computer programs, known as the SAS System, which enables users to carry out a wide range of data processing and analysis tasks, in particular statistical analysis. The core component of the SAS System, known as Base SAS, enables users to write and run their own application programs in order to manipulate data. These application programs are known as SAS scripts, and are written in a language which is peculiar to the SAS System ('the SAS Language'). Prior to the events giving rise to the present dispute, the customers of SAS Institute had no practical alternative to continuing to license the use of the necessary components in the SAS System in order to be able to run their existing SAS scripts; a customer who wished to change over to another developer's software would be faced with the necessity of having to rewrite its existing application programs in a different language.

The defendant in this case, World Programming Ltd ('WPL'), perceived that there would be a market demand for alternative software capable of interpreting and executing application programs written in the SAS Language. It therefore produced the 'World Programming System' ('the WPS'), which was designed to emulate the functionality of the SAS System as closely as possible in the sense that the same inputs would produce the same outputs, subject to only a few minor exceptions. This would enable users of the SAS System to run their existing SAS scripts on the WPS. There was no suggestion that, in doing so, WPL had access to the source code of the SAS System, or that WPL had copied any of the text or structural design of that source code. Nevertheless, SAS Institute contended that WPL had both committed a series of infringements of copyright and acted in breach of contract in creating the WPS and its accompanying documentation. For the purposes of the present discussion, the most relevant claim in this regard was that WPL had, in copying the manuals for the SAS System published by SAS Institute when creating the WPS, indirectly copied the programs comprising the SAS System, thereby infringing SAS Institute's copyright in the SAS System.[20]

SAS Institute's claim that the WPS infringed its copyrights in the SAS System raised some fundamental issues of copyright law, which were identified by the UK High Court. Of these issues, the one which is most relevant in present context related to the extent to which copyright in a computer program protects the programming language in which it is expressed. The SAS Language in which user scripts for the SAS System were written was held by the court to be a programming language, and it was common ground between the parties that the WPS reproduces certain elements of the SAS Language, and in particular that its parser reproduces keywords of the SAS Language. Counsel for SAS Institute argued that programming languages were not expressly excluded from the scope of the protection conferred by the Software Directive, pointing out to the court that recital 14 to the Directive only states that '*to the extent* that … programming languages comprise ideas and principles, those ideas and principles are not protected under this Directive'.[21] On this basis, he submitted that there was no reason why the

[20] The other principal claims raised by SAS Institute were that WPL had copied the manuals for the SAS System published by SAS Institute in creating the WPS; thereby infringing the copyright in those manuals; that WPL had used a version of the SAS System known as the 'Learning Edition' in breach of the terms of the license relating to that version and of the commitments made under that licence; and that WPL had infringed the copyright in the manuals for the SAS System by creating its own manual.

[21] Now recital 11 to the consolidated version of the Software Directive (emphasis added).

expression of ideas and principles in the form of a programming language should not be protected.[22] While the court acknowledged that recital 14 could be read in the manner contended for by counsel, it found the argument unpersuasive; instead, it relied upon the contrary view taken by a previous court in *Navitaire v Easyjet*, where recitals 13 to 15 of the Software Directive were quoted in support of the principle that computer languages did not fall within the scope of protection afforded to computer programs. However, the court did concede that the point was not *acte clair*, and that a reference to the CJEU was necessary in order to determine it.[23]

The CJEU interpreted the question referred by the UK High Court in this regard as asking whether article 1(2) of the Software Directive 'must be interpreted as meaning that … the programming language … used in a computer program in order to exploit certain of its functions constitute a form of expression of that program and may, as such, by protected by copyright in computer programs for the purposes of that Directive'. Article 1(2) states simply that 'protection in accordance with this Directive shall apply to the expression in any form of a computer program'. The CJEU answered this question in the negative. In doing so, it referred to its own earlier ruling in *Bezpečnostní softwarová asociace v Ministertvo kultury* ('*BSA*'),[24] where it had interpreted article 1(2) to mean that the object of the protection conferred by the Software Directive encompasses the forms of expression of a computer program (such as its source and object code) and the preparatory design work capable of leading, respectively, to the reproduction or the subsequent creation of such a program.[25] On this basis, the programming language used in a computer program to interpret and execute application programs written by users was held to be an element of the program by means of which users exploit certain functions of the program; consequently, it did not constitute a form of expression of that computer program for the purposes of article 1(2), and was therefore not protected under the Software Directive. Crucially, however, the CJEU did go on to state that the SAS Language might still be protected, as a work, by copyright under the Information Society Directive,[26] provided it is its author's own intellectual creation.[27] It now remains for the UK High Court to apply the judgment of the CJEU in determining whether, on the facts of the case, the SAS Language can indeed be protected as a copyright work.

Perhaps the most important aspect of the CJEU's decision is that, while it does not conclusively answer the question of whether programming languages may, as a general rule, be protected by copyright, it categorically denies the possibility of protecting such a language under the Software Directive as a part of the expression of the computer program in which it is used. In doing so, the CJEU has resolved an issue which has been identified in the literature on copyright protection for programming languages. Prior to the decision in *SAS v WPL*, scholars had identified two distinct bases upon which such

[22] This aspect of the Software Directive can be contrasted with the corresponding provision in the Japanese Copyright Act of 1970, art. 10(3) of which states expressly that 'the protection given by this Law to [program works] shall not extend to any *programming language* … used for making such works' (emphasis added). See also DS Karjala, 'The Protection of Operating Software Under Japanese Copyright Law' (1988) 10 EIPR 359, 364 – 367.

[23] The UK High Court referred, in addition, another eight questions which dealt with other aspects of SAS Institute's claims.

[24] [2011] ECDR 3 (Case C-393/09).

[25] [2011] ECDR 3, [35] – [37].

[26] Directive 2001/29/EC of the European Parliament and of the Council of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society.

[27] Case C-406/10, [45].

protection might be claimed: the programming language in question could either be regarded as a work in its own right and thus protected in itself; or, alternatively it could be protected as a non-literal expressive aspect of the computer program in which it was used.[28] In the latter scenario, a subsequent program which was written in, or otherwise made use of, the same programming language as that used in an earlier program would therefore be considered as having copied a part of the non-literal expression of that earlier program. In light of the CJEU's decision, however, this no longer forms a tenable basis upon which to argue that programming languages constitute subject matter that is capable of being protected by copyright. Rather, the remainder of this discussion will proceed on the first premise, namely that programming languages are to be treated as potential works in their own right, independent of any computer program in which they may happen to be used, and the question of whether they can and ought to be protected by copyright will be assessed on this basis.

Part III Legal considerations

As stated earlier, the CJEU in *SAS v WPL* left open the possibility that programming languages might, as works, be protected by copyright, provided they are the product of their author's own intellectual creation. However, as the following analysis will show, it is unlikely that this possibility will be realised. There are two grounds upon which this assertion is made. *First*, while the principal manifestations of a programming language, namely its specification and various implementations, will almost always be 'works' in the copyright sense, the copyright subsisting in them does not entitle their authors or other copyright owners to restrict other programmers from making use of that language in the ways described in Part I. *Second*, a programming language considered as a free-standing concept existing independently of any particular specification or implementation does not constitute a form of expression that can appropriately be characterised as a work. The view that a programming language may properly be regarded as a work on the sole basis that it is the product of its author's own intellectual creation will also be considered; however, it will be argued that such a conclusion is ultimately unsatisfactory.

The specification and implementations of a programming language

A *programming language specification* is a document which contains an explicit description of the programming language concerned, and is often written in a natural human language such as English. An example is the specification for the Java programming language, which exists both in the form of a printed reference manual and as a freely downloadable electronic document.[29] This is one of the primary means through which the developer of a programming language explains to potential users the behaviour and characteristics of the language or, more simply put, teaches the language to potential users. A *programming language implementation*, on the other hand, is a

[28] See e.g. Doerr (n 6), 147 – 155; Lowry (n 12), 1303 – 1306; Stern (n 5), 324. The literal expression of a computer program, in this context, refers to the text of its code. The non-literal aspects of a computer program, at least as identified by courts in the US, include its ultimate function or purpose, its structure or architecture, program modules, organisational or flow charts, algorithms and data structures, parameter lists, and macros: see *Computer Associates v Altai* 982 F.2d 693 (2nd Cir. 1992) and *Gates Rubber v Bando Chemical Industries* 9 F.3d 823 (10th Cir. 1993).

[29] J Gosling et al, *The Java Language Specification* (3rd edn, Addison-Wesley, 2005). The online version is available at <http://docs.oracle.com/javase/specs/jls/se5.0/jls3.pdf> (accessed 30 May 2012).

computer program that is capable of parsing the language concerned or, in other words, interpreting and executing programs written in that language; it often takes the form of a compiler or an interpreter.[30] As an alternative or in addition to a specification, the developer of a programming language may also teach it to potential users through a *reference* or *model implementation*, in whose behaviour the syntax and semantics of the language are made explicit. For example, all versions of the Perl programming language up to Perl 5 use the reference implementation approach, as no specification for the lanaguage exists; however, Perl 6, a planned major revision of the language, is currently under development as a specification.[31]

In almost all cases, both the programming language specification and any and all of its implementations may uncontroversially be treated as 'works' for the purposes of copyright law, the former being a literary work in the general sense and the latter being a computer program or programs.[32] However, the copyright subsisting in these works will not normally be infringed by a user who writes a program in the language to which the specification and implementation relate, nor by a user who develops a subsequent implementation of that language, provided in the latter case that she has not copied the code used in an existing implementation.

In *Infopaq International A/S v Danske Dagblades Forening* ('*Infopaq*'),[33] the CJEU held that the scope of the reproduction right set out in the Information Society Directive encompasses the reproduction of an extract of a protected work, provided that the elements reproduced in the extract are the expression of the intellectual creation of their author.[34] The broader principle to be drawn from this decision is that infringement will have occurred where the alleged infringer has reproduced – or indeed, carried out any of the acts to which the rightholder is exclusively entitled in respect of – any part of a protected work which constitutes the expression of its author's intellectual creation. In the case of a programming language specification, the intellectual creativity of its author can be found in its linguistic expression, in particular the choice, sequence and combination of words;[35] any intellectual creativity which has gone into the design and development of the programming language described in the specification will not form part of the expression of the specification itself. For this reason, a programmer who makes use of the programming language described in a certain specification does not infringe the copyright subsisting in the specification itself, as she is not copying any part of the author's intellectual creation that has gone into the writing of the specification.[36]

---

[30] A compiler is a computer program that translates programs expressed in a high-level programming language (such as the ones under discussion here) into machine language equivalents (which usually consist of a pattern of 0s and 1s) that can be recognised by the processing unit of a computer. In contrast, an interpreter is a computer program that executes programs expressed in a high-level programming language without first translating them into machine language, either directly or by first converting them into an intermediate form (such as bytecode) before execution.

[31] L Wall, 'Official Perl 6 Documentation' (*Perl6.org*) <http://perlcabal.org/syn/> accessed 31 May 2012.

[32] Which are, of course, protected as literary works: see Software Directive, art. 1(1); Agreement on Trade-Related Aspects of Intellectual Property Rights ('TRIPS'), art. 10(1); WIPO Copyright Treaty, art. 4.

[33] [2009] ECDR 16.

[34] [2009] ECDR 16, [48].

[35] This was the language used by the CJEU in *Infopaq* to describe the nature of the intellectual creativity employed by the author of a newspaper article: [2009] ECDR 16, [45].

[36] The conceptual distinction between a specification and its implementation (in this case, the realisation of the specification through a computer program or a component of a program), in the context of server communication protocols, has been discussed by the Court of First Instance in *Microsoft Corp v Commission of the European Communities* [2007] 5 CMLR 11 (Case T-201/04), [192] – [206]. For a similar discussion in the

In this regard, the distinction between a programming language specification and the language itself can be viewed through the lens of the idea-expression dichotomy: from this perspective, the language itself constitutes the 'idea' which is expressed through its specification, and the intellectual creativity that has gone into devising the idea is an entirely separate matter from the intellectual creativity that has gone into the crafting of its expression.[37] The predominantly US-based scholars who have written on this subject have also drawn comparisons between a programming language specification and an instructional work that teaches a reader how to accomplish a certain task or make a certain item, relying most notably on *Baker v Selden*, a landmark Supreme Court decision that is usually seen as having entrenched the idea-expression dichotomy in US copyright law.[38] In *Baker v Selden*, the Supreme Court held that the copyright subsisting in a book which described a new system of book-keeping did not extend to the system itself; in other words, while the author's individual manner of describing his system was protected by copyright, the system itself was not, and the author was not entitled to prevent third parties from making use of the system.[39] On this basis, it has been argued that a programming language specification is comparable to the book in *Baker v Selden*, while the programming language itself is comparable to the book-keeping system described in the book; thus, the copyright subsisting in the former is in no way infringed by the use of the latter.[40]

The same reasoning is equally applicable to any implementation of a programming language. In this case, the relevant intellectual creativity that gives rise to a protectable manifests itself in the manner in which the computer program is put together, including the definition of the tasks to be performed by the program, the selection of the steps to be taken and the way in which those steps are expressed;[41] it does not lie in the design and development of the programming language that the program is capable of parsing. As pointed out earlier, the decision of the CJEU in *SAS v WPL* confirms that the programming language used in a computer program is not a form of expression of that program; consequently, the use in a separate computer program of the same programming language as that used in an existing implementation does not infringe the copyright in that implementation, provided that its actual code has not been copied.

---

context of interfaces, see B Czarnota and R Hart, *Legal Protection of Computer Programs in Europe: A Guide to the EC Directive* (Butterworths, 1991), 35 – 38.

[37] The same point was made by the UK High Court in *SAS v WPL*, though the English formulation of 'skill, labour and judgment' rather than the European formulation of 'author's own intellectual creation' was used: [2010] ECDR 15, [207].

[38] 101 US 99. The principal holding in this case has since been codified in §102(b) of the US Copyright Act of 1976.

[39] 101 US 99, 102 – 104.

[40] Phillips (n 19), 97 – 100; Stern (n 5), 347 – 354; Doerr (n 6), 142 – 143; Lowry (n 6), 1312 – 1313. Comparable cases dealing with instructional works in the UK include *Brigid Foley Ltd v Ellot* [1982] RPC 433, 434 (no infringement of the words and numerals in a knitting guide by the production of knitted garments following the instructions); *Autospin (Oil Seals) Ltd v Beehive Spinning* [1995] RPC 683, 701 (no infringement of the claimant's charts for calculating the dimension of oil seals by using them to make such oil seals). In this respect, an example frequently cited by the UK courts is to that the baking of a cake (or making of a pudding) in accordance with a recipe does not infringe the copyright subsisting in that recipe: see *J&S (Holdings) Ltd v Wright Health Group Ltd* [1988] RPC 403, 414; *Autospin (Oil Seals) Ltd v Beehive Spinning* [1995] RPC 683, 701; *Navitaire v Easyjet* [2005] ECDR 17, [127].

[41] European Commission, 'Proposal for a Council Directive on the Legal Protection of Computer Programs (Explanatory Memorandum)' COM(88) 816 final, pt 1, para. 2.2 – 2.4. This was subsequently cited in the Opinion of the Advocate General in *SAS v WPL*.

The programming language as a free-standing 'work'?

The right to control the specific uses described in Part I cannot therefore be asserted in the copyright subsisting either in the specification or any implementation of the programming language concerned. There remains the possibility of doing so by claiming a copyright in the programming language itself, as a free-standing work which exists independently of any particular specification or implementation. However, it is questionable whether something so abstract as this may appropriately be characterised as a 'work' for the purposes of copyright law.

'Work' is a concept that occupies a central position in modern copyright law.[42] The Berne Convention, for example, is expressed to be for the protection of the rights of authors in their literary and artistic works',[43] and provides that authors are to be given certain moral and economic rights in respect of such protected works.[44] For all its importance, however, the parameters of the term remain singularly ill-defined, both internationally and at the level of the European Union.[45] The Berne Convention does not contain a definition of the term 'literary and artistic works', but merely sets out an illustrative list of the subject matter that is included within its ambit.[46] While the various European Directives require Member States to confer certain exclusive rights for authors in respect of their 'works', again, none of them provide a definition for the term. For this reason, there remains much scope for debate over the precise definition of 'work'. However, from the language of the various instruments dealing with copyright, it seems clear that a 'work' must necessarily be some form of expression, it being a well-established principle that copyright law only protects the expression of ideas, but not the ideas themselves.[47] It also seems fairly clear that the notion of a 'work' in the copyright sense is confined to expressions within the literary, artistic and scientific domains.[48]

It is the first of these two guidelines, the requirement that a 'work' must constitute some form of expression, that is particularly relevant to the present discussion. Can a programming language, considered as a free-standing notion existing independently of any single specification or implementation, be appropriately described as constituting a form of expression? The answer, it is submitted, is in the negative. A fundamental difference between a programming language in itself and the types of subject matter

[42] As a contrast, UK copyright legislation prior to the passing of the Copyright Act of 1911 tended to be subject-specific, with separate pieces of legislation that focused on specific types of creation, rather than a single copyright legislation regulating all types of works. See e.g. the Act for the Encouragement of the Arts of Designing, Engraving, and Etching Historical and Other Prints, by Vesting the Properties Thereof in the Inventors and Engravers, During the Time Therein Mentioned, 1735, 8 Geo. 2, c. 13 (Eng.); the Publication of Lectures Act, 1835, 5 & 6 Will. 4, c. 65 (Eng.); and the Sculpture Copyright Act, 1814, 54 Geo. 3, c. 56 (Eng.). See B Sherman, 'What is a Copyright Work?' (2011) 12 *Theoretical Inquiries in Law* 99, 99 – 103.

[43] Berne Convention, art. 1.

[44] Berne Convention, art. 6*bis*; arts. 9 – 14.

[45] Sherman (n 42), 102 – 103; C Handig, 'The Copyright Term "Work": European Harmonisation at an Unknown Level' (2009) 40(6) IIC 665; C Handig, '*Infopaq International A/S v Danske Dagblades Forening* (C-5/08): Is the Term "Work" of the CDPA 1988 in Line with the European Directives?' (2010) 32(2) EIPR 53. The proposed European Copyright Code drafted by the Wittem Group of academics, however, does contain a definition for the term 'work'.

[46] Berne Convention, art. 2(1).

[47] TRIPS, art. 9; WIPO Copyright Treaty, art. 2; Software Directive, art. 1(2). It is also reflected in the proposed European Copyright Code, art. 1.1(1).

[48] Berne Convention, art. 2(1) ('every production in the literary, scientific and artistic domain'); proposed European Copyright Code, art. 1.1(1) ('any expression within the field of literature, art or science').

listed as examples of 'literary and artistic works' in the Berne Convention – including 'books, pamphlets and other writings', 'dramatic or dramatico-musical works', 'choreographic works and entertainments in dumb show', 'musical compositions with or without words', 'works of drawing, painting, architecture, sculpture, engraving and lithography' – is that the former exists at a much higher level of abstraction and, crucially, is capable of being concretised in the form of detailed expression in which the latter is manifested as a matter of course. To take an example, it is possible to view a literary work, such as a novel, through lenses of increasing specificity, beginning with the mere 'idea' which it contains – such as a general statement of its plot and central themes – and concluding with the actual words and sentences which make up the novel itself. For a programming language, however, there exists no equivalent to the literal words and sentences of a novel beyond its specification and any implementations; considered apart from these, it remains an abstract concept that cannot appropriately be regarded as a form of expression, although it may be described or used in various forms of expression which are themselves works.[49] Consequently, it does not constitute a 'work' for the purposes of copyright law; rather, as pointed out by the Advocate General in *SAS v WPL*, it should be considered as a *means* which permits expression, in the form of computer programs, to be given.[50]

Programming languages were also described in the opinion of the Advocate General as 'devis[ing] specific methods to be used and facilitat[ing] the thinking necessary in order to write and formalise computer programs'.[51] Viewed from this perspective, a programming language has much less in common with the types of literary and artistic works mentioned in the Berne Convention, and is instead more akin to the 'ideas, procedures, methods of operation and mathematical concepts' which are expressly excluded from the scope of various international copyright treaties.[52] Essentially the same point has been raised by scholars in the US to argue against the protection of programming languages by copyright; programming languages, it is contended, constitute 'systems' for communication with computers, and 'systems' are expressly excluded from the scope of the protection conferred by the US Copyright Act of 1976.[53] Although neither the various European Directives on copyright nor the international copyright treaties to which Member States are party contain an express exclusion of 'systems', a similar reasoning applies. Programming languages are the 'building blocks' which permit expression to be given, but are not themselves expression;[54] for this reason, they do not constitute 'works' that are capable of being protected by copyright, but are instead methods which enable human-computer interaction. Thus, they should be considered as falling within those categories of subject matter that, by international consensus, are not appropriate subjects for copyright protection.

---

[49] Stern (n 5), 363 – 364 (arguing that a programming language 'is not a work of authorship. It is something that is embodied or used in one or another kind of work of authorship, such as in a book teaching the language or in a computer program using the language').

[50] Opinion of the Advocate General in *SAS v WPL*, [71].

[51] Opinion of the Advocate General in *SAS v WPL*, [70].

[52] TRIPS, art. 9(2); WIPO Copyright Treaty, art. 2. See also the proposed European Copyright Code, art. 1.1(3) of which excludes from the scope of protectable expressions facts, discoveries, news and data; ideas and theories; procedures, methods of operation, and mathematical concepts.

[53] Phillips (n19), 74 – 78; Lowry (n 12), 1309 – 1315. §102(b) of the US Copyright Act of 1976 expressly excludes from the scope of copyright protection 'any idea, procedure, process, system, method of operation, concept, principle, or discovery'.

[54] Other scholars have also used the metaphor of a programming language as 'building blocks' for expression: see Samuelson, Vinje and Cornish (n 8), 162; Lowry (n 12), 1315.

The programming language as the product of its author's own intellectual creation

This section will briefly consider the opposing argument that, in order to be considered as a 'work' for the purposes of copyright law, a programming language needs only to fulfil the requirement of being its author's own intellectual creation. This is a plausible argument when considered in the light of a line of recent CJEU decisions which have emphasised the requirement of the author's own intellectual creation in discussing the copyright status of various types of works. In *Infopaq*, the CJEU held that copyright 'is liable to apply only in relation to a subject matter which is original in the sense that it is its author's own intellectual creation',[55] a point which was reiterated in the subsequent case of *Painer v Standard Verlags GmbH* ('*Painer*').[56] As we have seen, the CJEU held in *SAS v WPL* itself that the programming language and the format of data files used in a computer program 'might be protected, as works, by copyright under [the Information Society Directive] provided they are their author's own intellectual creation',[57] even though they were not protected under the Software Directive as a form of expression of the computer program to which they related; the same reasoning had previously been applied to a graphic user interface in the *BSA* case.[58] One reading of these decisions is that the current copyright system in Europe requires copyright protection to be conferred upon each and every 'intellectual creation'; applying this to the subject matter of the present discussion leads to the conclusion that a free-standing programming language, in itself, is capable of being protected as a work by copyright, provided it is the product of its author's own intellectual creation.

In construing the requirement of 'author's own intellectual creation', the CJEU has consistently emphasised the importance of the author's ability to make choices; most recently, it has held in *Painer* that an intellectual creation is the author's own if the author was able to express his creative abilities in the production of the work by making free and creative choices.[59] Theoretically, at least, the process of designing a new programming language affords to its developer considerable scope for making free and creative choices, including in selecting the appropriate keywords and devising the relevant syntax rules; indeed, this was taken into consideration in the wording of the actual question referred by the UK High Court to the CJEU, where the SAS Language was described as 'a programming language devised by the author of the [SAS System]' which was said to comprise 'keywords devised or selected by' and 'a syntax devised by' said author. If being an 'intellectual creation' is indeed the sole criterion for being regarded as a 'work', it follows that a programming language, taken by itself and considered independently of any particular specification or implementation, may logically be protected as such.

Such a conclusion, however, is far from satisfying. In the first place, although the process of developing a new programming language may indeed provide sufficient scope for the exercise of free and creative choices, in practice, many 'new' programming

[55] [2009] ECDR 16, [37].
[56] [2012] ECDR 6 (Case C-145/10), [87].
[57] Case C-406/10, [45].
[58] [2011] ECDR 3, [45] – [46].
[59] [2011] ECDR 3, [89]. The element of 'choice' on the part of the author was also referred to in *Infopaq*: [2009] ECDR 16, [45].

languages are in fact built upon predecessor languages.[60] The Java programming language, for example, is modelled upon the C and C++ programming languages, a fact that is explicitly referred to in its specification.[61] As a result, a programming language which is constructed in this manner cannot be said to be entirely the product of its author's *own* intellectual creation, in the sense that it does not originate wholly from that author's free and creative choices alone, but is instead partly derived from the intellectual creation of the author of the predecessor language or languages upon which it is based. In order to assert a copyright claim over such a language, therefore, it will first be necessary to disengage the aspects which have been copied from existing languages from those which have originated wholly from the mind of the present author. This is perhaps particularly evident in the case of command languages, where the functions provided by application programs of the same type will be largely similar, and there is little scope for variety in the range of command terms used for invoking them. Standard commands in a spreadsheet program, for instance, include terms such as 'Print', 'Copy' and 'Move'; while it is theoretically possible to substitute these terms with 'Publish', 'Duplicate' and 'Shift' respectively, the range of possible synonyms and near-synonyms is necessarily very limited.

Secondly, if programming languages, as free-standing, abstract notions existing independently of any associated specification or implementation, were to be regarded as a 'work' capable of being protected of copyright, this could lead to a fundamental divergence in approach between Member States where the relevant legislation requires a work to be fixed in a material form as a prerequisite for copyright protection, such as the UK and Ireland,[62] and Member States where no such legislative requirement is imposed.[63] Programming languages would be protected by copyright in the latter jurisdictions, but not in the former – an outcome that could potentially result in a severe disruption of the functioning of the internal market, running completely counter to the stated purpose of the various European Directives that have, over the years, sought to harmonise certain aspects of copyright law across different Member States.

Thirdly, and perhaps most importantly, even if programming languages may arguably be protected as copyright works under the strict application of existing doctrinal rules, various policy considerations such as the need to foster competition and innovation in the software development industry militate strongly against the conferment of such protection. These will be elaborated upon in Part IV. The preferable approach, therefore, would be to read the line of CJEU decisions beginning with *Infopaq* in a disjunctive manner, such that it requires the adjudicator to determine, as an initial step, whether a particular subject matter is amounts to a 'work', before going on to consider whether it fulfils the criterion of 'author's own intellectual creation', rather than interpreting it to mean all 'intellectual creations' necessarily constitute 'works' that are capable of being protected by copyright. This was arguably the approach taken by the French Court de

[60] Doerr (n 6), 141; Lowry (n 12), 1306 – 1308.
[61] Gosling et al (n 29), 1 – 2.
[62] UK Copyright, Patents and Designs Act of 1988, s 3(2) ('copyright does not subsist in a literary, dramatic or musical work unless and until it is recorded, in writing or otherwise'); Irish Copyright and Related Rights Act of 2000, s 18(1) ('copyright shall not subsist in a literary, dramatic or musical work or an original database until that work is recorded in writing or otherwise by or with the consent of the author').
[63] See also the arguments made in relation the US Copyright Act of 1976, which does contain a fixation requirement, that protecting programming languages by copyright would be to allow for the protection of 'unfixed expressions': Phillips (n 19), 78 – 81; Hamilton and Sabety (n 8), 269 – 270 . cf. Doerr (n 6), 139 – 141; Lowry (n 12), 1308 – 1309.

cassation in *Sté Senteur Mazal v SA Beauté Prestige International*, a case involving copyright in a perfume, where it focused on the question of whether the fragrance of a perfume constituted 'a form of expression that benefits from the copyright protection intended for works of the mind', rather than going directly to the question of whether it was original in the sense that it bore the imprint of its author's personality, as the Cour d'Appel had previously done.[64] This construction of the language used by the CJEU allows for the application of the reasoning described in the previous section, that programming languages are the means which permit expression to be given, but are not themselves expression and hence not 'works'.

Part IV Policy considerations

In addition to the legal and doctrinal considerations elaborated upon in Part III, what is perhaps more important in the context of the present debate are the policy considerations that militate against the protection by copyright of programming languages in themselves. Part IV explores these considerations from two differing perspectives. *First*, it will argue that there is no justification for protecting programming languages by copyright, as this would not significantly incentivise the development of new programming languages. *Second*, it will demonstrate that such protection, rather than incentivising technological innovation, would instead stifle it, as software users and competing developers would be barred from making use of tools that are essential for the creation of new software products.

Copyright protection would not incentivise the creation of new programming languages

One of the major justifications which have traditionally been invoked in favour of protecting certain intellectual products by copyright is that such protection incentivises the creation of a socially optimal number of intellectual products. Without copyright, runs the argument, there would be little inducement for individuals or firms to invest time, money and effort in the creation of new works; instead, it would be in their self-interest to allow others to develop new works, and then to devote their energies towards producing imitations of these new works, thus saving themselves the costs of prototyping and initial creation.[65]

It is doubtful, however, whether such an inducement is necessary in the case of programming languages, based on the history of their development. As stated in the introduction to this paper, from the earliest days of computing until quite recently, the software industry appears to have operated on the assumption that programming languages were freely available for the use of any person; yet the development of new programming languages has continued apace.[66] Even in the present day, there have been relatively few overt attempts to assert copyright over programming languages, when compared with the sheer number of such languages that are available; in this respect, developers such as SAS Institute are in a minority. Some developers have even

[64] *Sté Senteur Mazal v SA Beauté Prestige International* (2008) 39(1) IIC 113; (2010) 41(2) IIC 234.
[65] See generally EC Hettinger, 'Justifying Intellectual Property' (1989) *Philosophy & Public Affairs* 31, 47 – 48; WM Landes and RA Posner, 'An Economic Analysis of Copyright Law' (1989) 18 *Journal of Legal Studies* 325
[66] Lowry, 1343 – 1345 (pointing out that between 1976 and 1977 alone, there were over 150 freely available programming languages in existence).

expressly dedicated their programming languages to the public domain.[67] All this indications point towards the conclusion that copyright protection is not a major incentive for the creation of new programming languages. Instead, factors such as competition within the industry and the need for technical solutions that are not provided by existing languages may prove to be the main drivers for such innovation.[68] Indeed, as many new programming languages are in fact built upon predecessor languages, copyright protection might well prove to be an impediment to innovation in this respect, as subsequent developers would no longer be free to make use of various aspects of existing languages in constructing new languages.

In any case, the developers of new programming languages will generally be sufficiently rewarded through the temporal and technological advantages of being the first movers in their particular market.[69] Their greater expertise in the language concerned places them in the best position to begin creating and marketing new application programs based on that language, at a time when their competitors are still familiarising themselves with the language and attempting to determine the best uses for it. This, it is submitted, constitutes sufficient incentive for the development of new programming languages, particularly when a technological need arises.

Copyright protection for programming languages would potentially inhibit technological development

The implications of conferring copyright protection upon programming languages, taken to their logical conclusion, are extreme: potentially, every single use of a protected programming language would require a licence from the owner of the copyright in that language.[70] Under these circumstances, software developers would be left with two practical options: either to obtain a licence, possibly upon onerous terms, from the owner of the copyright in an existing language; or to develop their own programming language. This would, in turn, have a negative effect upon innovation within the software industry, due to two related factors: the risk of creating a monopoly, and the imposition of restrictions upon interoperability.

Risk of creating a monopoly

The conferment of copyright protection upon programming languages could potentially result in the grant of a *de facto* monopoly to developers of the most popular and most frequently used languages, allowing them to exert an unprecedented level of control over both programmers and programs which make use of those languages. This small number of developers would be free to charge exorbitant prices for programming

[67] The legal effect of dedicating one's copyright work to the public domain is doubtful. It has been argued, in the context of UK law, that such a dedication would amount at best to a bare licence: P Johnson, '"Dedicating" Copyright to the Public Domain' (2008) 71(4) *Modern Law Review* 587. In Germany, it has been held that copyright does not cease by reason of abandonment, a position which would appear to be applicable, by extension, to any attempts at dedication: *Berlin Wall Pictures* (1997) 28 IIC 282.

[68] Lowry, 1344 – 1345 (arguing that market competition provides better incentives to create new programming languages than copyright protection does).

[69] This reflects, in part, another traditional justification for the grant of copyright, namely that copyright is awarded to creators because they deserve to benefit from the products of their creativity. See generally Hettinger (n 65), 40 – 41; J Hughes, 'The Philosophy of Intellectual Property' (1988) 77 *Georgia Law Journal* 287, 305 – 310; LC Becker, 'Deserving to Own Intellectual Property' (1992) 68 *Chicago-Kent Law Review* 609.

[70] See also Hamilton and Sabety (n 8), 270 – 272.

language licences which other developers, lacking the expertise for creating their own languages, would have little alternative but to pay.[71] This is particularly the case where a developer or other user has invested heavily in a certain programming language; it would be 'locked in' to that language, as the users of the SAS System were prior to the advent of the WPS, and would be forced to continue licensing the use of that language even if a new, superior alternative were to become available, due to associated costs such as the expense of retraining employees in the new language and translating existing programs into the new language.[72]

Perhaps even more disturbingly, a developer who owns the copyright in a programming language would be in a position to impose restrictive terms upon its licensees. It could, for instance, limit the types of programs in which those languages to be used, thus shutting out any potential competitors. Developers that are not in a position to create their own programming languages would be in the unenviable position of having to select an existing language based not upon its technical merits, but solely upon the terms which the copyright owner is willing to offer.[73] At its most extreme, the copyright owner could terminate the licence of any venture which it perceives to be a threat, thereby reinforcing its monopoly not only in the market for programming languages, but in the market for other types of software products as well.[74] In this context, it should be noted that programming languages in themselves would not constitute computer programs or parts of programs under the hypothetical legal paradigm that would allow them to be protected as copyright works, particularly in the light of the CJEU's decision in *SAS v WPL*; consequently, the licensees of these languages would not be entitled even to the limited range of permitted uses available to 'lawful acquirers' and licensees of computer programs under the Software Directive.[75]

Restrictions upon interoperability

If programming languages were to be protected by copyright, this would severely impede the ability of software developers to create products that are interoperable with existing programs. Such a developer would be compelled to pay the high licence fees and to comply with the potential onerous licensing terms imposed by the owner of the copyright in the language used in the existing program; again, the copyright owner might take advantage of its position to terminate the licence of a potential competitor, or even to decline to grant one in the first place. This is a probable outcome of *SAS v WPL*, should the UK High Court hold that copyright subsists in the SAS Language.

The ability of software developers to engage in the independent creation of new products that are capable of interoperating with existing programs has been identified as being 'of key importance for competition, innovation and market entry' in the software market'.[76] It provided the impetus for the drafting and, ultimately, implementation of article 6 of the Software Directive, which allows for the decompilation of computer programs 'to obtain the information necessary to achieve the interoperability of an independently

[71] Lowry (n 12), 1341.
[72] Samuelson, Vinje and Cornish (n 8), 162 – 163; Lowry (n 12), 1341.
[73] Hamilton and Sabety (n 8), 272.
[74] Hamilton and Sabety (n 8), 271 – 272.
[75] Software Directive, arts. 5 and 6.
[76] European Commission, ' Commission Staff Working Paper on the Review of the EC Legal Framework in the Field of Copyright and Related Rights', SEC(2004) 995, para. 2.2.1.3

created computer program with other programs' under certain circumstances.[77] More recently, it was demonstrated in the case of *Microsoft Corp v Commission of the European Communities*[78] that consumers will be deprived of innovative software products if competitors are unable to create software that is fully interoperable with that produced by the market leader.[79] The conferment of copyright protection upon programming languages would prove to be detrimental to the interests of consumers, as it would effectively prevent software developers from creating competing products that are interoperable with existing programs, ultimately 'locking' consumers into software products that are created by a single developer or a small group of developers. In these circumstances, the lack of competition within the market would leave these developers with little incentive to continue improving their products, thus leading to the stagnation of technological innovation.

Conclusion

This paper has demonstrated that there are sufficient reasons, both on legal and policy grounds, to hold that programming languages ought not to be protected as copyright works. While copyright does indeed subsist in the specification of a programming language as well as its implementations, it does not entitle the owner to control the use of that programming language in the manner sought by claimants in the same position as SAS Institute. In addition, it is difficult to argue that a programming language, considered as a free-standing concept existing independently of any particular specification or implementation, constitutes a form of expression that can appropriately be characterised as a 'work' in the copyright sense; instead, it would be preferable to regard it as a means through which expression is given. Perhaps more importantly, it cannot be demonstrated with any great certainty that copyright protection would provide an incentive for the creation of new programming languages. Indeed, the reverse appears to be the case, as the conferment of such protection would have the potential to undermine competition and innovation within the software industry in Europe. It is to be hoped that the UK High Court will bear these considerations in mind in its determination of this aspect of *SAS v WPL*.

[77] Software Directive, art. 6(1).
[78] [2007] 5 CMLR 11 (Case T-201/04).
[79] Samuelson, Vinje and Cornish (n 8), 158.