

4 Intellectual property

4.1 The growing importance of intangible assets

Readers will appreciate that the concept of *property* is crucial for business. A firm needs to know what it owns (and can therefore use freely, and/or charge others who want to use it), and what belongs to others (so that if it needs to use those things it must do deals with their respective owners). Business looks to law to define property rights and enable them to be enforced.

Before the IT revolution, the things over which firms needed to assert ownership were usually tangible things – goods, land, and so forth. The law of “intellectual property”, under which for instance a company might own a patent on a newly-devised industrial process, was a fairly obscure legal backwater. Information technology has changed this, by hugely raising the profile of intangibles. Ever since the Industrial Revolution, the economies of nations like Britain and the USA had been dominated by manufacturing. But by the late 1980s, the share of GDP (gross domestic product) attributable to manufacturing fell below half in both nations, and it has continued to fall – outweighed partly by growth in services, but also by growth in trading of intangibles.



What do you want to do?

No matter what you want out of your future career, an employer with a broad range of operations in a load of countries will always be the ticket. Working within the Volvo Group means more than 100,000 friends and colleagues in more than 185 countries all over the world. We offer graduates great career opportunities – check out the Career section at our web site www.volvogroup.com. We look forward to getting to know you!

VOLVO
AB Volvo (publ)
www.volvogroup.com

VOLVO TRUCKS | RENAULT TRUCKS | MACK TRUCKS | VOLVO BUSES | VOLVO CONSTRUCTION EQUIPMENT | VOLVO PENTA | VOLVO AERO | VOLVO IT
VOLVO FINANCIAL SERVICES | VOLVO 3P | VOLVO POWERTRAIN | VOLVO PARTS | VOLVO TECHNOLOGY | VOLVO LOGISTICS | BUSINESS AREA ASIA



By now, intangibles form a large proportion of the assets of a typical firm, as measured by the prices which the market sets on them. Gordon Brown, then Chancellor of the Exchequer, said in 2006:

Twenty-five years ago the market value of our top companies was no more than the value of just their physical assets. Today the market value of Britain's top companies is five times their physical assets, demonstrating the economic power of knowledge, ideas and innovation.²⁰

What Brown was saying was that most property of the “top companies” is now intellectual property. It is largely IT which has brought about this change; and it naturally means that intellectual property law has become a very significant area of business law, which is having to develop in response to developments in the technology.

The topic which might perhaps come first to a student reader's mind is the way that sharing music over peer-to-peer networks has been undermining the copyrights owned by music companies, which have been struggling either to invoke the law to defend their position, or to develop novel business models that allow them to make money within the new technological environment. But for present purposes, this area is not actually very significant. The law of copyright as it applies to music is clear; the only change introduced by IT lies in making the law easy to break and hard to enforce. More interesting, for this textbook, are areas where the property itself (not just the means used to reproduce it or move it around) consists of things like computer software or electronic databanks. In those areas, it is often far from clear how the existing laws of intellectual property apply. Courts are adapting laws that were written long ago for other purposes in order to develop an intellectual-property régime for the IT industry, and so far this is not working too well.


The issues are not about enforcement – unlike with music filesharing, where many of the individuals involved do not care whether their activity is legal, provided they feel safe from detection! In civilized societies, most organizations by and large aim to keep within the law and respect one another's property rights – but they need to know what those rights are. It would be hard for a business to be profitable, if it made a habit of not insisting on rights which it did legally possess.

4.2 Copyright and patent

There are two longstanding legal devices for defining and protecting different sorts of intellectual property: copyright, and patent. Copyright was originally introduced to define ownership in “literary works”, such as novels, poems, or non-fiction books, but came to be extended by analogy to things like musical compositions, films, and so forth. Patents relate to newly-invented machines or industrial processes.

Neither copyright nor patent law was part of the Common Law; both devices were introduced by statute. (Indeed, the USA has had a general law of copyright only since the 1890s – it was a standing grievance for Victorian novelists that no sooner did the fruits of their labour emerge from the press than American publishers' agents would rush single copies across the Atlantic, where they would be reprinted and sold without reward to the author.) The original motivation of both copyright and patent law was the same: they were intended to stimulate advances (in literature, and in industry) which would benefit society, by creating concrete incentives for the innovators.

The kinds of protection offered by the two areas of law are different. Copyright is something that the author of a "literary work" acquires automatically in producing the work, and it forbids anyone else to make a copy of the work (for a set number of years into the future, and with various provisos that do not matter here) without the right-holder's permission. Thus an author's copyright is a piece of property which he can sell or license for money; in the case of books, typically a publishing company contracts with an author for permission to publish his book in exchange for royalties paid to him on copies sold. With newer media such as films, the business models are different, but the underlying law (which is what concerns us) is essentially the same.



The advertisement features a runner in a red shirt and black shorts running on a path. The background is a warm, orange-hued sunset or sunrise. The Gaieteye logo is in the top left, with the tagline "Challenge the way we run". Below the logo, the text "EXPERIENCE THE POWER OF FULL ENGAGEMENT..." is followed by a dotted line. Further down, the text "RUN FASTER. RUN LONGER.. RUN EASIER..." is displayed. To the right of this text, there are technical diagrams showing a runner's foot and leg with lines indicating angles and movement. A yellow button in the bottom right corner says "READ MORE & PRE-ORDER TODAY" and "WWW.GAITEYE.COM", with a hand cursor icon pointing at it.

gaiteye®
Challenge the way we run

**EXPERIENCE THE POWER OF
FULL ENGAGEMENT...**

**RUN FASTER.
RUN LONGER..
RUN EASIER...**

READ MORE & PRE-ORDER TODAY
WWW.GAITEYE.COM



A patent, on the other hand, is not acquired automatically by the inventor (or anyone else). Taking out a patent is a complicated and expensive undertaking, but if a patent is granted, it forbids anyone (again, for a set future period) from exploiting the process or mechanism without the patent-holder's permission; so again the patent is an economically-valuable piece of property, which can be sold or licensed to companies wanting to use the innovation in their business.

The legal contrast between copyright and patent was neatly summed up by Tim Press:

A document setting out a novel chemical process would attract copyright protection, but that protection would protect the document against copying, not the process from being carried out. A patent for the process would prevent it from being carried out but not from being written about or broadcast.²¹

Computer programs are “text” which defines and controls “processes”. So on the face of it there is a question about which kind of intellectual-property protection is more relevant to software. Over the years during which IT has been economically important, the answer has been shifting.

4.3 Do we need intellectual-property laws?

Before we look at how intellectual-property law is being adapted to the needs of our industry, it is worth taking a moment to recognise that quite a few people are sceptical about whether such laws are needed at all. Society has changed since these laws were introduced. The inventor of a useful industrial process will nowadays not typically be a lone genius who needs income from his patents to keep afloat: he will be a salaried researcher, working for a company which will be best placed to exploit his invention whether or not its competitors are legally forbidden to do so. Some commentators point to the numerous books which are written essentially for love of writing rather than for money, and to the success of the Open Source movement in producing software systems (such as Gnu/Linux) which are made freely available to all comers, and they argue that intellectual-property law as a whole has outlived its usefulness.

Others who do not go that far argue that legal protection is specially undesirable for computer software, because it interferes with the ways in which software advances. Tim Berners-Lee has expressed this by saying “Programming is always about reassembling existing stuff – novel ideas are rare”.²² To those who see things this way, legal protection for software creates progress-stifling monopolies rather than socially-desirable rewards for innovation.

A third group accept that there is a need for intellectual-property laws in our field, but they argue that trying to generate such a body of law by adapting copyright and/or patent law is not going to work – from poetry or Newcomen’s Atmospheric Engine to Java is just too great a stretch. They argue for *sui generis* laws, that is, new kinds of law which do not extend existing concepts of copyright or patent but introduce some third, separate type of protection. (*Sui generis* is Latin for “of its own kind”.) We shall see that in one area (databases) this argument has now prevailed.

On the whole, though, the consensus seems to be that the IT industry does need a régime of legal protection for intangible property, and that most of this protection will have to come via development of existing intellectual-property laws. People who suppose that the best way of dealing with a novel phenomenon must surely be through brand-new laws often fail to appreciate the massive amount of work and time needed to develop adequate legal frameworks from scratch. Some features of existing law may be inappropriate for the new area, but the body of case law and statutory revision which builds up round established legal concepts over the years will comprise a great deal of material which applies just as well to the new area as to older areas. By adapting existing law, society gets all that legal predictability for free.

4.4 Copyright for software

The initial assumption was that software should be protected by copyright rather than patent law. After all, what a programmer produces is lines of source code, usually on paper at first: this has at least a superficial resemblance to a “literary work”, but it is not at all like a physical machine. In English copyright law, the term “literary work” has no implication of aesthetic value – a user manual for a microwave oven counts as a “literary work” as much as a Shakespeare sonnet.

For a while there was debate about the status of a program after it was compiled into object code, when it was likely to exist only in electronic form rather than on paper – was object code still protected by copyright law? But Parliament settled this question with the *Copyright, Designs and Patents Act 1988*, which among other things laid down that for legal purposes computer programs in any physical form are literary works. Hence there is now no doubt that copyright law does apply to software. If firm A develops a valuable software application, firm B is not free just to copy and use the application, without negotiating a license fee with firm A.

However, this protection is less robust than it might seem. Remember that copyright law is only about *copying*. Imagine that I had never read the Harry Potter novels, but wrote a novel out of my own head which just happened to be word-for-word identical with one of those books. Then, in theory, I would be free to sell my book and compete for a share of J.K. Rowling’s income; I have copied nothing. Of course, in practice, no court would allow this; but that is because the chance of identical manuscripts being composed independently is so tiny that the law would assume I *must* have copied. With software, though, scenarios rather like this are more realistic than they are with novels.

Consider (1) a case where I take someone else's program and mechanically substitute new names for each variable – wherever, say, *myvar* occurs it is replaced by *varA*, and so on with the other variables. Variable names are arbitrary, so the new program will behave exactly as the old one does, and it is not an identical copy. Would copyright law allow this?

The literary analogy might be to publish a novel identical to one of J.K. Rowling's, except that "Harry Potter" is changed to "Jimmy Cotter" throughout, "muggles" are consistently replaced by "poggles", and so on. British copyright law is clear on this: it protects the plot of a novel, not just the words, so J.K. Rowling would win a breach of copyright case. Analogously, just changing the variable names in a program would not be a defence against an action for breach of software copyright.

But now consider cases where the copying is less direct:

(2) While working for firm A, I developed a program to carry out some task; having moved to firm B I write a new program from scratch for the same task, using the same techniques as I remember them, though without access to my old code. (Note that copyright in my old program will belong to firm A, not to me. Although I said above that copyright is automatically acquired by the author of a "literary work", that is not true when the writing is done as part of an employee's duties: in that case copyright belongs to employer rather than author.)



(3) Working for firm B, I examine the behaviour of a software system owned by firm A and write code to emulate its behaviour, but without access to the source code from which firm A's object code was compiled.

In these cases, the analogy with literature does not tell us whether there are breaches of copyright or not. (The literary analogue of (2) might be a case where I read a Harry Potter novel and then try some time later to reconstruct it from memory: the law would very likely not care about that, because the result would just be a laughably clumsy novel which would do nothing to damage J.K. Rowling's sales.) What is more, not only is it unclear what copyright law *does* say about these cases, but it is not obvious what we *want* the law to say. Society does not want to see producers of worthwhile software ripped off, but it does want to encourage fair competition.

4.5 Two software-copyright cases

To see how copyright law is being applied in practice, we must look at cases. An example like (2) above was *John Richardson Computers Ltd v. Flanders* (1993). Flanders was a programmer who worked for John Richardson's company as an employee and later as a consultant. He helped Richardson to write a program allowing chemists to print prescription labels and keep track of their stocks of medicines; the program was in assembly code for the BBC Micro (a popular home and small business computer of the 1980s). After leaving John Richardson Computers, Flanders wrote a program in QuickBASIC for the IBM PC to execute the same functions, and he set up a company to market this program.

Clearly, there will be no character-by-character similarity between a Basic program and one in assembly code. Any similarity would be at the level of the logic of the various routines – something that cannot be compared mechanically, but requires human understanding to detect. Richardson's side argued that the logical similarities in this instance did make it comparable to copying the plot of a novel, so that it amounted to breach of copyright. But on the whole that was not accepted by the court. The judgement was complex, but (to cut a long story short) it said that while “non-literal” copying of software might in principle be a breach of copyright, in this case there were only a few minor infringements.

A case like (3) was *Navitaire Inc. v. EasyJet Airline Co. & anor* (2004). Navitaire developed a reservation system for airlines, “OpenRes”, which EasyJet licensed to use in its business. Later, EasyJet wanted to own the software it relied on, so it commissioned another software house to develop a system “eRes” to emulate OpenRes. The two sides agreed that “EasyJet wanted a new system that was substantially indistinguishable from the OpenRes system...in respect of its ‘user interface’”. Again the court decided that eRes did involve some minor infringements of Navitaire's copyright, but the overall weight of the decision went in favour of EasyJet.

So the trend is clear: extended from “literature” to software, copyright law protects software producers against little more than direct, character-by-character copying. That level of protection is important in itself; it is copyright law which is invoked when people or organizations are convicted of using pirate copies of valuable proprietary software, or of uploading such software to P2P networks such as KaZaA. (Since 2007 the law has taken a tough line against software piracy, with new powers for Trading Standards officers to investigate suspected breaches and more serious penalties than before for convictions.) But copyright is not providing much defence against subtler ways of misappropriating programmers’ intellectual output.

Incidentally, discussions of this area in law textbooks often confuse two different kinds of similarity between programs. After Apple commercialized the first GUI (graphic user interface), it objected when competitors produced their own GUIs with a similar “look and feel”. For instance, having chosen to represent the “Trash” concept with a dustbin icon, Apple objected when others did the same (which is why some systems use a swirly “black hole” for the same concept). Without entering into the legal complexities of the look-and-feel arguments, this issue is rather separate from the question of copying program structure. Copyright in “look and feel” is rather like copyright in artistic images – the fact that in this case the graphic material is acting as gateway to a computer system has limited relevance. Copying the logical routines of a program, on the other hand, is something which relates exclusively to IT; and copyright law is not providing strong protection against it.

4.6 Databases

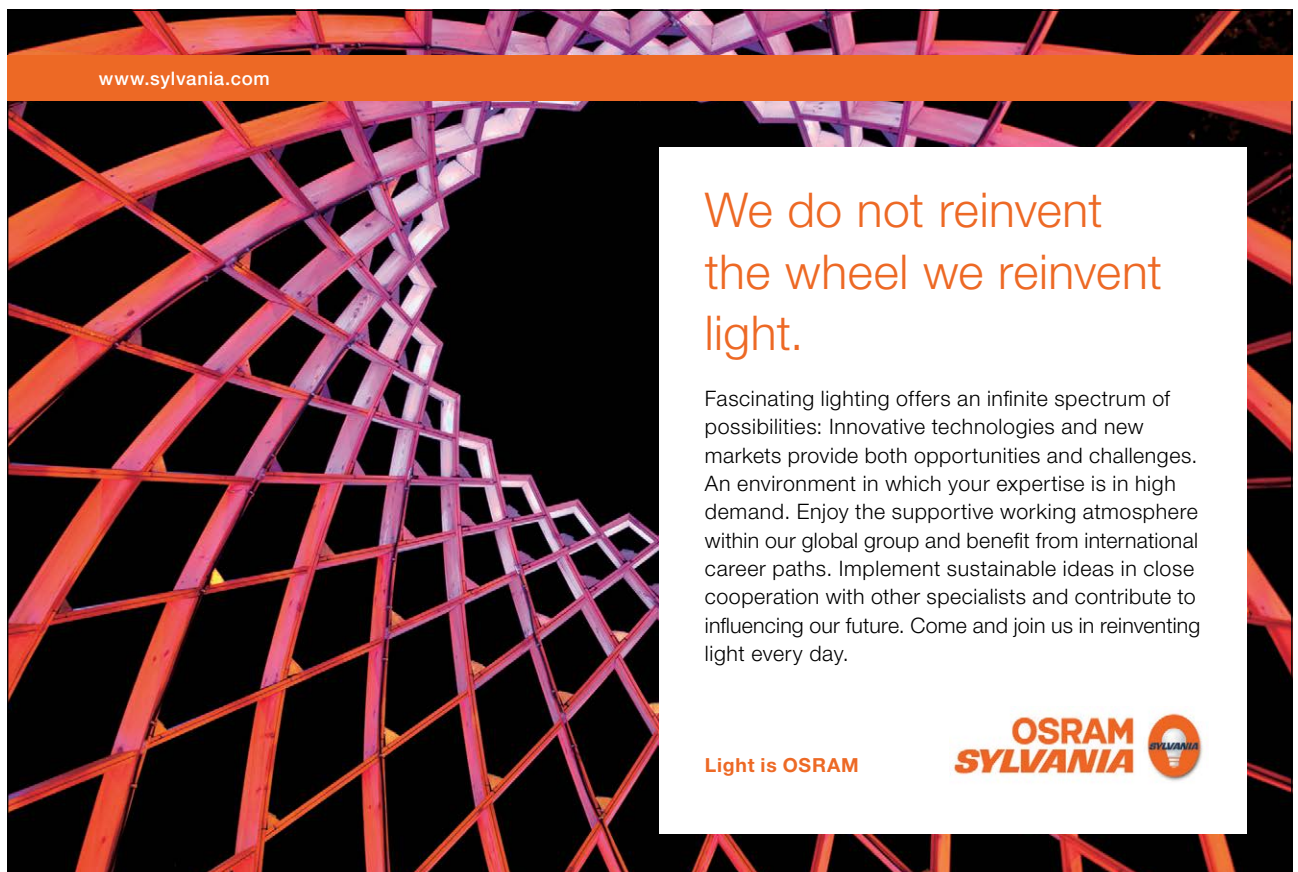
Commercial electronic assets comprise not only the software which processes information, but the databases of information to be processed. (The word “database” is ambiguous. It can refer to a DBMS – database management system – such as Oracle; a DBMS is itself a software application. But I am using “database” here to refer to the collection of pieces of data which a firm uses a DBMS to store and process, for instance a large collection of details of potential customers, or the geographical data assembled by the Ordnance Survey to generate its maps.) The IT revolution has turned databases into big business. A Department of Trade and Industry minister said in 1997:

Estimates of the size of the UK database market range up to £10 billion but even that may be an underestimate.... It is growing at more than 11 per cent. a year.²³

Although English copyright law has protected databases as “literary works”, they are as far as they could be from literature in the everyday sense. We have seen that our law did not care about that. But the corresponding laws in some other EU states did: German copyright law, for instance, applies only to documentation having at least some minimal aesthetic or scientific value. Consequently the EU introduced special *sui generis* intellectual-property protection for databases via a *Database Directive*, transposed into UK law in 1997. Under this, the copyright protection which had applied to databases in Britain was explicitly withdrawn in cases where the database is a purely mechanical listing of facts without intellectual content (e.g. a phone directory); those databases are now protected by new legal rules independent of both the copyright and the patent régimes.

Unfortunately, the new rules are not very clear. This was illustrated by the chief case so far brought under them in Britain: *British Horseracing Board & ors v. William Hill Organization Ltd* (2001).

The Horseracing Board keeps a database of horses and jockeys due to run in particular races. Maintaining it is a significant commitment, costing about £4 million to add or update about 800,000 entries annually. Naturally the information is important for betting firms like William Hill, and for many years they used it without objection. However, when the World Wide Web arrived and William Hill began displaying information taken from the Horseracing Board’s database on their website, the Board claimed unauthorized reuse of their data.




www.sylvania.com

We do not reinvent
the wheel we reinvent
light.

Fascinating lighting offers an infinite spectrum of possibilities: Innovative technologies and new markets provide both opportunities and challenges. An environment in which your expertise is in high demand. Enjoy the supportive working atmosphere within our global group and benefit from international career paths. Implement sustainable ideas in close cooperation with other specialists and contribute to influencing our future. Come and join us in reinventing light every day.

Light is OSRAM

OSRAM
SYLVANIA



When the initial decision was appealed, the Appeal Court found it necessary to ask the European Court of Justice for rulings on eleven questions about precisely what the Database Directive was intended to mean. (This is a standard procedure for European legislation; it contrasts with the English legal tradition, where a law means just what the words say and courts are supposed to do any necessary interpretation for themselves.) The upshot, based on the ECJ's rulings, was that what was crucial to the Board's property rights was the "stamp of authority" it could associate with its data by virtue of its role as governing body of the sport. A betting firm could never confer that stamp of authority on racing data, no matter how much it copied from the Board's database; so the verdict went in favour of William Hill – it had not and could not take over the crucial feature of the Horseracing Board's intellectual property.

Before 1997, the Board would have won the case under British copyright law. So, ironically, it seems that a Directive which was introduced in order to strengthen the protection of databases has actually reduced their protection (in some respects, at least) in Britain – and British databases are believed to account for about half of all databases in the EU.²⁴

4.7 The focus shifts from copyright to patent

Returning from databases to software: we saw that the profession initially looked to copyright rather than patent law to protect intellectual-property rights in software. More recently, though, patent law has begun to seem more relevant. This is for three reasons:

- copyright protection is proving inadequate
- the software industry is changing
- patent law is expanding its scope

Let us take these points in turn.

4.7.1 Copyright protection inadequate

We have seen that the trend in software cases has been to interpret copyright as covering little more than character-by-character copying – which is often not what is at issue in practice. Patent law, on the other hand, does not care whether anything has been *copied* or not. If A holds a patent on a mechanism or process X, then B is forbidden to use X (without A's permission) *even if B really did invent X independently*. What matters, for patent law, is which of A or B applied to the Patent Office first. If A is granted a patent on some programming technique – let's say, an efficient sorting algorithm – then anyone else who wants to use that technique must pay A for the right to do so, even if he has never heard of A or A's work.

So patent law offers the prospect of a more worthwhile level of protection for intellectual property in software than copyright law is providing.

4.7.2 The software industry is changing

In the early decades of industrial and commercial computing, a firm wanting to computerize some of its operations would typically buy the relevant hardware, and employ in-house programmers to develop software to automate its particular activities, or commission an outside software house to develop a bespoke system for its individual needs. Before the 1980s, the concept of standard software applications was scarcely known. But, as readers will be well aware, things have changed. A high proportion of all commercial software nowadays consists of standard application packages carrying out standard functions, with copies of the same package often being used by hundreds or thousands of different client organizations. Recent developments such as SaaS (software as a service)²⁵ are accelerating this trend.

That makes patent protection for software more economically attractive than before. It takes effort and expense to take out a patent, and for a one-off system this would often be pointless. It is not very likely that an outsider could study its details closely enough to adapt it for use elsewhere, and even if that were feasible, adapting the system to the different individual requirements of the new organization might be almost as expensive as producing a new system from scratch. But, once software applications are standardized and widely-used commodities, the balance changes. Spread over perhaps thousands of copies of a package, the cost of a patent becomes trivial; and the danger of a competitor emulating the package becomes much more realistic.

4.7.3 Patent law expanding its scope

From these points it may seem self-evident that someone wanting to protect his rights in novel software would be in a stronger position under patent than under copyright law; why would anyone bother with copyright law in the first place? But the attraction of patent law is irrelevant, if patent offices will not grant patents on software; and until recently that was the position. However, this has been changing. We need to look at the rules under which patent offices operate.

4.8 The nature of patent law

Countries have their own patent offices; but in the 1970s European countries agreed a European Patent Convention which aimed to harmonize patent rules across Europe, and established a European Patent Office (EPO) as a one-stop shop issuing patents valid in different European countries. (This is not an EU creation – the signatories to the Convention include non-EU countries such as Switzerland; and what the EPO issues are bundles of separate patents valid in separate countries – there is no such thing as a single Europe-wide or EU-wide patent, though the idea has been discussed.) Someone wanting a British patent can apply either to the EPO or to the Intellectual Property Office (as the UK patent office is known).

In discussing the legal systems of Western nations, much of the time we find Britain grouping with the USA and contrasting with the Continental European countries. Because of the Convention, patent is exceptional in this respect: British law resembles the laws of European nations and (as we shall see) contrasts in some important respects with American law. The UK *Patents Act 1977* aimed to implement the agreed principles of the European Patent Convention.

In order to patent an invention, one has to submit a *claim* showing that it meets a number of requirements. (Here I refer to British law, but these requirements are similar in any national patent law including that of the USA.)

- the invention must be genuinely new, so far as *public* knowledge is concerned;
- it must not be obvious – there must be an “inventive step”;
- it must be capable of industrial exploitation;
- it must not fall within a class of things which the law explicitly excludes from the scope of patent, which includes intellectual matters such as ideas or scientific discoveries, as opposed to industrial processes which exploit ideas or discoveries. Someone who invents a novel sorting algorithm would never be allowed to patent it – it is an idea rather than an industrial process; on the other hand, a machine which uses the algorithm to sort filecards could well be patentable. The EPO glosses the ideas v. processes distinction by saying that the invention must be “technical”, in the sense that it involves some tangible end product.



360°
thinking.

Deloitte.

Discover the truth at www.deloitte.ca/careers

© Deloitte & Touche LLP and affiliated entities.



When someone applies for a patent, an official called a *patent examiner* sets out to check whether the requirements are met. This is not straightforward: the test of novelty (lack of *prior art*, in patent-law lingo) implies attempting to prove a negative. Since patent examiners cannot be omniscient, they sometimes make mistakes and issue patents that ought not to be granted. The grantee's competitors can challenge a patent, for instance as not genuinely new, and if they make their case the patent will be revoked.

A patent on an industrially-significant process can be a valuable piece of property. It forces would-be competitors either to abandon attempts to compete, or to do things in some different way which may be less efficient, or less appealing to customers.

4.9 Is software patentable?

Where does software stand in all this? In Britain and elsewhere it was seen as more analogous to mathematical formulae or abstract algorithms than to physical machines or processes. The Patents Act explicitly lists, among the class of things that are not patentable:

a scheme, rule, or method for performing a mental act, playing a game or doing business, or
a program for a computer (my italics)

That is why people initially tried to use copyright law to protect their software; and one might think that it leaves no room for debate – patent law is just irrelevant to the software business.

However, the Act has a loophole. The article immediately following the one just excerpted goes on to say that the list of unpatentable things

shall exclude patentability...only to the extent to which a [patent claim] relates to such subject-matter or activities *as such*. (Again my italics.)

So the question arises: would a patent for software which executes process X be a patent for the “software as such”, or would it be a patent for process X? If the former, the patent would not be valid; but if the latter, it might be.

This is a good example of an issue which a scientist, a computer specialist, or another non-legal mind might well dismiss as a non-question. How could one possibly decide that an application relates to “software as such” rather than to the process which the software carries out? But the Patents Act is part of the law of the land, so lawyers are not allowed to treat the issue as meaningless and empty – even if it is. Cases are being fought out to give it a meaning. The trend of the decisions is towards increasing willingness to grant patents for software. Unfortunately, the trend is also turning this area of law into a very messy one indeed.

4.10 Some software-patent cases

To exemplify that last point, consider three patent claims from a period of a few years about the turn of the century.

4.10.1 PBS Partnership/controlling pension benefits system (1995)²⁶

In 1995, the Pension Benefit Systems Partnership asked the EPO for a patent on a software system which calculated pension benefits. The EPO refused the claim, not because it related to a program – the combination of computer hardware and software was deemed to be “a physical thing of a technical nature”, hence in principle patentable – but because of the nature of the “inventive step”: since pension benefits can be (and traditionally were) calculated manually as a purely clerical activity, the inventive step in this case was deemed non-technical, hence the claim failed. (The hardware was technical, of course – but the hardware was not novel.)

4.10.2 Fujitsu's Application (1996)

In 1996 the English courts upheld a refusal by the UK patent office to grant a patent on software which enabled chemists to display and manipulate crystal structures on screen. Part of the reasoning was that what was novel in this claim was the ability of the user to choose how to rotate a three-dimensional crystal structure one way or another, but this act of choice is a human rather than mechanical activity – one cannot patent “mental acts”, though one can patent “processes methods or apparatus based upon such acts”, quoting the judge who upheld the refusal in the Court of Appeal.

The judge went on to illustrate this distinction from a more concrete sphere of activity:

Rules as to the planting of potatoes in which the operator is instructed to measure and evaluate matters such as the type of soil, location, weather and availability of irrigation is a method for performing a mental act [and hence unpatentable]. Directions to plant one seed potato every metre is not. It is a precise process.

As Lloyd remarks (p. 332), this way of drawing the boundary round patentable processes seems paradoxical. One could easily imagine that a computer-controlled potato-planting machine might incorporate routines to take account of soil type, irrigation, and so forth. Apparently, this level of sophistication would prevent the machine being patented, while a simple machine that plants at regular intervals could be patented if novel; yet the sophisticated machine would surely be “more...deserving of protection”.

4.10.3 Microsoft Corp./Data transfer with expanded clipboard formats (2003)

A few years later, the EPO granted Microsoft a patent on a type of clipboard operation within Windows which allowed data in one format to be copied into an application that is based on some other format, for instance a graphic copied into a plain ASCII file. Microsoft's claim opened with the words "A method in a computer system...". Yet the EPO accepted that this was not a claim for a "computer program *as such*", which (as we have seen) would have made it unpatentable. They saw it as a novel technical process for making data available across applications, and granted the patent.

Perhaps the reader thinks he can see differences between these examples which might justify the different outcomes of the claims; but, if so, it would be easy to quote further examples to convince him that a consistent logic just is not there. Discussing another claim which was granted in 1994, Tim Press comments "The reasoning of the [EPO] Board in finding (as they did) technical content in the *Sohei* case is at times impenetrable".²⁷

By now, the situation is such a morass that in 2006 the English Court of Appeal announced that Britain should abandon the attempt to follow precedents set by the EPO and go its own way: it is impossible to follow EPO precedents, because the EPO is not following its own precedents consistently.

SIMPLY CLEVER

ŠKODA



We will turn your CV into
an opportunity of a lifetime

Do you like cars? Would you like to be a part of a successful brand?
We will appreciate and reward both your enthusiasm and talent.
Send us your CV. You will be surprised where it can take you.

Send us your CV on
www.employerforlife.com

Download free eBooks at bookboon.com



Click on the ad to read more

(Part of the problem here stems from the contrasting attitude to precedent in English versus Continental legal systems, discussed in chapter 2. Continental law treats precedent as persuasive only, rather than binding, so the charge of inconsistency might not seem so damning in the eyes of Continental lawyers as it does to English lawyers. But the fact remains that there is no clear basis at present for deciding whether some commercially-valuable new software might be patentable.)

The Court of Appeal's "declaration of independence" bore fruit in 2007, when the UK Intellectual Property Office refused to activate a patent that had already been granted by the EPO to Symbian for a software system which enables other software to run faster (*Mapping dynamic link libraries in a computing device*). The UK IPO saw this as clearly excluded from patentability by the law which both it and the EPO are supposed to be applying; and when the High Court allowed Symbian's appeal, the IPO counter-appealed – making it clear that its motive was simply to get some clarity about what rules it is meant to work by. (In 2008 the Court of Appeal gave its verdict in favour of Symbian, urging that the British and European patent offices should try to compromise with one another's ways of working where possible – while agreeing that the law on software patents is vague and inconsistent.)

4.11 The American position

Meanwhile, in the USA, software patents have become wholly normal. Before 1998, the American rules about unpatentability of computer programs were similar to ours, but in that year *State Street Bank v. Signature Financial Group* established a radically new precedent, allowing a patent on a software system for administering and keeping accounts for "mutual funds" (the US equivalent of unit trusts). Under English law, such a system would have been doubly unpatentable. Not only is it a program rather than a machine, but what it automates is "business methods" – the kind of processes carried out manually by clerical workers, rather than technical, industrial processes. Before 1998, business methods were unpatentable in the USA also, but since *State Street Bank* that rule has been abandoned; very large numbers of patents are being granted on business-process software.

This expansion of American patent law is being amplified by a separate development, independent of IT: excessive workload is leading the US patent office to grant many patents which it shouldn't, on "inventions" which are obvious, or not truly new. When patent examiners reject a claim, they have to justify the rejection with solid argument, but it is a straightforward matter to accept a claim. So, inevitably, when a patent office is overwhelmed by numbers of claims the outcome is that too many are granted. (A classic example is US Patent no. 5,965,809, granted in 1999 for a method of determining a woman's bra size by running a tape measure round her bust.) In the case of software, "prior art" is specially difficult to check, which exacerbates the problem. Consequently software patents, even for trivial-seeming techniques, are now very usual in the USA.

4.12 An unstable situation

The American patent situation creates pressure on Europe to move the same way: it is difficult, when the economies of the two regions are as closely bound together as they are nowadays, for their patent régimes to be widely different. And the fairly chaotic current nature of European patent law makes this pressure hard to resist (even supposing Europeans want to resist it). By now, the European Patent Office is in practice granting many software patents and refusing few claims in this area. Yet, on paper, it remains the law that one cannot patent “a program for a computer”.

One way to regularize the situation would be for the law in Europe to be brought into line with practice, by explicitly abandoning the rule which says that programs are not patentable. The European Commission proposed a *Directive on Software Patents* which would have done that. But this Directive proved highly controversial and, to the surprise of many observers, in 2005 the European Parliament overwhelmingly voted it down. They were swayed by arguments of the kind quoted from Tim Berners-Lee above. Many people see the likely effect of software patents as being to stifle rather than encourage valuable technological progress; they urge that software patents would merely confer “licences to print money” on Microsoft, Amazon, and the like.

These are weighty considerations. On the other hand, we saw in chapter 2 that predictability is valued by business. At present, whether or not a patent claim for a software system will succeed is far from predictable.

I joined MITAS because
I wanted **real responsibility**

The Graduate Programme
for Engineers and Geoscientists
www.discovermitas.com



Month 16

I was a construction supervisor in the North Sea advising and helping foremen solve problems

Real work
International opportunities
Three work placements





