wire EFT transfer between Chicago and Switzerland where the telephone lines connected but the message was never received.

## KEY LEGAL ISSUES OF THE MESSAGE

**Diagram VII**

Privacy and Data Protection

Ownership of Data

Liability for Incomplete, Outdated or Incorrect Data

Standard of Care of the Provider

Security

Access

Applicable Law

**Susan Nycum** is a lawyer with Baker & McKenzie, Palo Alto, CA

## THE HISTORY OF SOFTWARE PROTECTION

**Martin Greenstein**

The history of software protection dates back to the early days - seven or eight years ago. It's interesting in that it puts into perspective the "look and feel" expansion of the copyright doctrines; the expansion of protection of software under the patent laws we see today. If you go back to the video games of the late 1970's and early 1980's, you find software being protected essentially as audio video works. These were, in essence, cartoon characters. They were the games that were placed and were the original user-interactive interface. In playing the game you controlled the programming. These could be protected under traditional copyright principles, not as software, but as visual works.

The personality of the characters themselves add copyright features similar to cartoon characters and greeting card characters. One of the early cases concerned a take-off deal with the personality of the little animal that went around and ate the person it was chasing. The look of the eyes as it was about to seize upon the character was pointed out by the court as one of the infringing features. The code itself was irrelevant; it was the original "look and feel" doctrine.

The US courts moved from that into protection of the more traditional software. They first dealt with the issue of what code is and whether it is protected as a literary work. I should mention as a preliminary note, however, that the US Copyright Act, which was passed in 1976 (effective in 1978) after several decades in the making, did not specifically mention that computer programs are copyrightable. The regulations certainly provide the mechanism for recording registration for computer programs but the Act itself doesn't specifically say so. In fact, the '76 Act didn't even mention the words "computer program". It took the 1980 amendments to bring in an oblique reference to the fact that software was protectable by copyright. Even then they did not do it by saying so expressly; they did it by defining some of the things you cannot restrict with the copyright, namely, the right to make an archival backup copy of the software and a recognition that in executing a computer program the very first thing you do is load it from a disk or tape into memory and therefore make a copy of it - the original copyright infringement. That thing itself, a copy necessary to the functioning of the machine, was held not be an infringement of the copyright.

It was not a big jump to recognise protection for **source code** because it is a literary work. Granted that it is in a language with a very small syntax and very few vocabulary words, but it is a readable work. **Object code** was a different situation however, and the court struggled with that for a period of time, eventually deciding that either under the concept of a transalation or an adaptation or some transformation, object code would be protected. Copyright Office actually will accept object codes which it cannot read. It prefers to accept source code deposits for copyright registrations, but will accept object code deposits. They can't read them but they accept them under what is known as **rule of doubt.** If you tell them that it is copyright material, tell them it is a program, they will accept it because you told them so, but then the burden is a little higher if you have to go and sue on that registration later. All of that, of course, presumes you understand that in the United States it is possible to get certificates of copyright registration. It is one of the few countries that actually does accept deposits and issues official-looking certificates for copyright registration.

Having passed the object code barrier, the courts then had to deal with another form of object code, **ROM base code**, which was internal to the machine. ROM base code had a different set of problems. It is suspiciously similar to patent protection. After all, what you've got is a piece of hardware. It's internal to the machine. There were other types of computers - analogue computers to be specific. You don't program an analogue computer - you build it, you wire it. That principle applies to digital computers as well; you can do anything in a program if you have enough wire. Actually, you hotwire the machine. When you stick a chip in the machine and tell the Copyright Office that it's really a program, it's very difficult to distinguish that from a wiring pattern that has to be in a chip, that would be protected under the patent laws. Nonetheless, the courts got past that and Apple Computer, Inc. v Franklin Computer Corporation*[1] and some other cases have accepted that object code, even if it is on a chip in ROM, would be protectable.

The last jump was into **micro-code.** THAT decision is a bit on the rocks right now because of a peculiar situation in the States. But the decision is basically sound and it will be followed again by the next court. It's not an intellectually interesting decision, however. I do not find anything particularly notable in the concept that micro-code would in fact be protectable. Micro-code consists of the micro-instructions that are inside a micro-processor that the assembly language causes to execute the individual gates and registers, and move bits of information around the micro-processor chip. That pretty much exhausted the limits of the question of protection by copyright at the time. You must keep in mind, however, that whilst all this was developing the computer industry itself was changing.

Before 1975 most software was developed in-house. Mainframes were designed by the large companies that owned the mainframe. There was almost nothing you could buy off the shelf. Anything needed to run the computer came with the computer, (the so called "bundling" that caused IBM some problems in its early days). From 1975 till 1980 (or so) most software sold was mainframe software and it was sold to large companies. It was fairly expensive software, and was sold by individual licence agreements. They brought you an agreement anywhere from one to twenty pages, your lawyer looked at it, probably didn't understand it, said okay and then signed it. It gave you very good trade secret protection but no one worried about copyright protection. Buyers were companies of repute and were not likely to duplicate the software. In any case you couldn't sell it on the market because nobody could really use it and if a couple of hundred copies or five hundred copies were sold that was a lot. People knew who the customers were and if some customer showed up with a copy and they weren't the licensees it was very easy to find them.

The real change in software came in the 1980's with the advent and expansion of PCs, when software became a mass market item and less expensive. New techniques had to be developed to distribute software. It was no longer possible to have a separate licence agreement with every PC purchase. People came up with the concept of a "shrink wrap", or as it's called in the U.K. and in Europe, a "box top" licence that purported to create a licence by virtue of breaking the cello wrap and opening the package. That concept has not been tested in the States and is of doubtful validity in other Commonwealth countries. Friends in civil law countries tell me that there is no question that it would be invalid. The only thing left was copyright protection. So the courts began to address the issue of copyright protection more and more.

Having dealt with whether or not something should be protected the next question is: How far does that protection extend?

Again the early cases were easy. People copied the software, no question about it. The really interesting questions came about in conversions of software to run in different environments. The so called "look and feel" series of cases actually began with a case back in the late '70s that went the other way. Whelan Associates, Inc. v Jaslow Dental Laboratory, Inc.,*2 is the case best known throughout the world in establishing the "look and feel" principle. The case never mentioned the "look and feel"; it talks about the structure, sequence and organisation. There was a companion case, SAS Institute Inc., v S&H Computer Systems Inc.*3 That was a District Court case with very similar facts and a very similar outcome. Both of those cases involved computer programs to operate dental laboratories, which put to rest once and for all the often held notion that the copyright laws have no teeth!

Another case, Broderbund Software Inc. v Unison World Inc.*4 expanded the "look and feel" doctrine. From the standpoint of business software it was less interesting than Whelan. Whelan dealt with the conversion of a program to run under a different system. The Court went through it and determined that the structure, sequence and organisation of the file structures and the flow of information from module to module in the computer was identical. Broderbund had a similar analysis but it dealt with a graphics program to print greeting cards called "Printshop". There were graphics aspects to it and the Courts have found it much easier to deal with the "look and feel" of graphics than the "look and feel" of words on the screen.

In both Whelan and Broderbund there was a common thread. They were business deals gone bad. In each instance, the bad guy had some kind of a business contractor arrangement to do a conversion of the program, had access to the source code and tried to adapt it for a different system. The deal went sour, the parties had a fight and lost through default. The bad guy lost. The courts awarded relief to the person that should have lost but they then had to fashion that relief into something that fell under the copyright laws because that is where the main claim rested.

A later case, Digital Communications Associations Inc. v SoftKlone Distributing Corporation*5 again concerning a clone, didn't really have a bad guy. There was no business deal gone sour. It was a legitimate clone situation where someone decided to make a program that emulated 'Crosstalk', a very common and well respected communications package. In this case, there was no access, there was no attempt to convert; it was built as a clone and marketed. They called it "Mirror". The company was called

'SoftKlone' so you knew they were trying to position it from a competition standpoint as head-to-head clone competition. There was another aspect of DCA v SoftKlone that didn't exist in the others. The plaintiff had a copyright registration from the Copyright Office on the main screen of the program. The Court found that that main screen, the user interface, the Log-on screen was highly creative, even though it was a collection of words. It talked about the commercial value of that screen to the product. But what it was really finding was that you can build the same program, but if you absolutely copy the same screen you have violated that copyright certificate - that registration. They awarded relief but it was fairly mild relief. It ordered the rearrangement of some of the words on the main screen. Some of the words changed were fairly descriptive but the requirement was that they had to change the screen around. It turned out that the company had already decided to change the screeen and had ALREADY brought out the second version of the program which turned out to be a tremendous success.

Now let us look at the Lotus situation. You have doubtless read about the Lotus cases against Paperback Software and a product called TWIN from Mosaic software. Lotus, earlier criticised for the filing, came out saying, "We're not claiming copyright on spreadsheets, are not claiming copyright on the two line command interface. We're bringing this law suit because the defendants have taken ninety-nine percent of our program". That's in contrast to some of the clean clones that we'll talk about a little later that actually added enhancement. It's interesting in the Lotus case because as early as January 1987, a couple of weeks before the suit was filed, Lotus was saying that they would be bringing a motion for preliminary injunction to take the product off the market. It still has not been done and the case is sitting there in discovery - in the early stages of discovery and pleadings. U.S. litigation is very slow.

There are a lot of dangers in the broad "look and feel" approach, that have been read into Whelan and some of the other decisions. Part of the danger is not focussing on what exactly are the protectable elements. All Karate games look alike because it is Karate, so the second Karate game shouldn't be a "look and feel" infringement of the first just because it has someone who rolls, kicks, punches and does the things it is expected that Karate does. Yet one court found an infringement because of the overall effect; what they call the total concept and feel. (That's the real "buzz" word that the Courts picked up from some earlier cases in the graphic arts area.)

All spreadsheets look alike. You can begin to confuse the patent standard with the copyright standard if you don't focus on where the protectable elements are. Copyright protection then can get too broad.

Susan Nycum talks about Dolby and Postscript and dedication issues. It is every software vendor's dream that his program becomes the standard in the industry. Once it happens, as it happened with Dolby, they want to grab it back; they don't want it to be the standard anymore. Software vendors in the early days encouraged people to put out utility programs - they even encouraged what would later be considered trade mark infringements. The Ashton Tate Data Base, for instance - there was a whole line of utilities that came out with "little d this", "d you tell" or "d you print". Dolby claimed trademark infringement by Ashton Tate. Another client programmed Turbo Pascal in its earlier days. People cut out "Turbo this", "Turbo that" programs. That client would now like all other Turbo users to disappear.

Software is evolutionary, not revolutionary. A lot of people believe the "look and feel" cases are situations in which large companies with a lot of economic muscle and the ability to pay a lawyer can harrass smaller companies into making changes. But the development of software is an evolutionary process. The MacIntosh, which was Apple's machine, came from a laser which, in turn, came from early Xerox work. Lotus copied the whole concept of Icons and Visicalc and made an overall copy of its spreadsheet. Lotus later bought Visicalc and the remnants of Visicorp (which produced Visicalc) are now suing Lotus for "look and feel" infringement of those pieces that they claim Lotus forgot to buy from them. When DRI went head-to-head with Apple over the Gem system, there wasn't an actual law suit but there were negotiations that led to a change in the "look and feel" of the DRI Gem screen by something as momentous as changing the shape and configuration of a garbage can, which was one of the Icons that were used. Not particularly important cases, but evidence of companies at battle. DRI was a fairly large company and it's often said that when elephants fight it's only the grass that gets trampled. That's not particularly true when big companies fight little companies and the "look and feel" issue has the potential for allowing very broad, undefined claims that can ramble on for years. In this business if you inhibit someone's sales for a period of time, technology has passed them by. That's one of the claims that Mosaic made in its suit.

If you look ahead and try to predict where we are going with "look and feel", take a look at translations and adaptations of programs which I believe would almost certainly be considered an infringement. In the Whelan/Broderbund standard the structure, sequence and organisation are the same. When you translate to a different operating system of language, as those technically trained will know, you actually have to do a bit of redesigning of the program, but you still have to start with the basic structure, the same file instructions and the same organisation of a program.

On the other hand, if you just look at file structures and say "Hey! it's got the same file structure so it's an infringement!" you've got a real problem because you can't exchange data between programs, you can't import and export data at some point, matching up the file structure, taking it in and then changing it. So that's much too broad a standard to use.

If you re-code a program from functional specifications only I believe it will not be held to be a "look and feel" infringement. That's the "clean room" situation. It will be extended more and more into the software field, where people will look at the functional specifications, go back and re-code the program.

If you copy the screens identically you are certainly running the risk of infringement, particularly if the screen is a very complex screen. You have a real functionality issue because there are some things that you have to copy; there are some words that you have to use. Others you clearly don't have to use. If you copy the screens identically and they are complex screens the court will probably find an infringement on some kind of equity issue, particularly if you go out and market your product, positioning it as a 100 percent clone. If you take a close look at the way clones are being marketed now, none of them are really clones; they are all work-a-likes that have added something different ... it's the evolutionary process. Developers are trying to position their products as something better, built upon what went on in the past, similar but not identical. The enhancement is probably known as infringement.

## COMMAND STRUCTURE

Command structure is an interesting issue. If your commands are very complex and you copy the entire set you run the risk of the Courts finding copyright infringement. There is no copyright for individual words. On the other hand, there is clearly copyright for a <u>collection</u> of words - that is the essence of copyright. A collection of words, language, a book or an expression in a language is fine. A collection, however, should have to convey the expression, the essence of copyright. You are protecting the <u>expression</u>, not the idea. As a command language and expression as it is implemented it is probably protected but the language taken as a whole is probably not. It is more functional than it is an expression. The commands themselves - the individual commands - are clearly utilitarian and functional. Your utilitarian and functional aspects are generally not protected by the copyright laws of the United States and the rest of the world.

In the command structure, you come up against the "dedication of a standard" argument again. A lot of command structures are in essence languages. dBase and some of the largest Data Base programs are essentially languages. Languages are generally not copyrightable. A big controversy is brewing in the United States now over the Data Base language or d-Base. The users group wants to establish standards for the d-Base language. Ashton Tate, which in its early days tried to do whatever it could to encourage that to be become the standard (but never actually said so, never made the Dolby mistake), is now taking the position that it does not want a users group defining what the d-Base standard is. If they allowed that they would run into the anomaly that d-Base IV (when it comes out) might not be d-Base compatible. Someone else is defying the standard.

It is my view that the concept of "look and feel" is going and that, as an issue, it will fade in the future. It will be presented in a different environment at least.

The MacIntosh has become a very serious business machine. You don't really hear about "look and feel" issues in the MacIntosh environment. That's because it's the machine that provides the interface, not the actual user programs themselves. Now, as the IBM environment moves past MS-DOS into windows presentation and manager with Gem becomes more popular, you are again going to have a machine defined, or at least an operating system defined standard interface, so you lose the issue whether one program interface is the same as another program interface and again, "look and feel". We're really talking about the user inerface; the work-alike MacIntosh programs work alike because Apple defines the interface. In a sense "look and feel" is the price we paid for MS-DOS allowing IBM to buy MS-DOS for fifty thousand dollars or so in the early days.

Talking about copyright on screens, there is an interesting issue with the U.S. Copyright Office right now dealing with the layout and the form of text on a screen. If we're talking about graphics, there's no problem. The Copyright Office registers graphics screens. They are just artistic works, so there is no issue at all.

In defining structured sequence and organisation in Whelan's case, the Court commented on the screens, but commented on them quickly in passing as evidence of what must be in the computer program. It was further evidence of the infringement. It was not an infringement itself to have a similar screen. One of the early cases in Texas in 1978 clearly limited protection to the source and the object codes and not the input formats - the screens themselves. The Copyright Office for awhile did register textual screens as copyrights separately from the programs. They stopped doing that, however, after Whelan's case.

In Digital Communications Associates v SoftKlone*5 the Court held that the underlying software does not protect the screen. There again, the plaintiff had a separate screen registration issued before the Copyright Office stopped it. The Copyright Office was then faced with a dilemna. It had stopped issuing registration certificates and the Courts said, "Hey! you need these things, in fact that's why you win - you have a registration". People, like Lotus, were clamouring and saying, "We want our screens registered". The Copyright Office published a notice in 1987 and said that it would hold hearings in September and October and take comment. It is now chewing on the results of that hearing and the submissions received trying to make a decision as to whether or not it will allow the registration of screens. Apple said that screens should be registered and submitted several different programs that created the identical screen programs written in different languages to prove that the screen depiction is totally separate from the underlying code. Lotus took the position that it was not necessary to register screens because they were covered by the underlying program.

In its early days Apple did not worry much about registering the screens. They were much more interested in pushing the MacIntosh display graphics as a standard. There is some question as to whether or not Apple just dedicated that standard or whether there was an implicit licence to MacIntosh developers to use the interface. Lotus, on the other hand, which took the position that it shouldn't have to register screens, had tried to register its screens and had been turned down by the Copyright Office because of the view that the screens were included in the underlying programs. My own view is that the Copyright Office will try to find a compromise and it will allow you to submit screen layouts as part of your registration to make sure that they're protected. If the screen is really a detailed screen it may find enough creativity to allow registration.

These battles, as well as other developments in the States, have led to renewed interest in patents for software. The US patent system was really in the doldrums in the early 1980's. Some sixty to seventy percent of the patents were held invalid when they got to litigation. The Government restructured the Patent Appeals system and created the Court of Appeals for the Federal Circuit from which all patent appeals are supposed to flow, in order to gain some consistency. The result has been a renewed interest in some significant patent decisions, e.g. Polaroid.

In the U.S. the laws are quite clear (sort of!). Software is patentable. Inventions are covered by patent law. But you have to keep in mind the three N's. The invention has to be Non-obvious, Novel and Not-described-as-software!. You can register a patent for software if you describe it as a computer and it's a program controlled operaton. The programmed computer, not the program - is a machine which implements a process. That was determined in 1981 in a U.S. Supreme Court decision.

Pitfalls in patents, however, are fourfold. You cannot get a patent on a principle of nature, like a mathematical algorithm. You can't get a patent on a method of doing business. You can't get a patent on the method of selling hamburgers for fifteen or twenty cents by putting up golden arches and having them ready when people walk in etc. You can't get a patent on mental steps or thought processes and you can't get a patent on the arrangement of printed matter.

However, you can get a patent on a machine that implements a mathematical algorithm e.g. a machine to break code or predict weather. You can't get a patent on a method of doing business but patents have issued on the cash management account for Merryl Lynch that maintains accounts for people and sweeps money into a interest bearing account so that you always have zero balances at the end of the day! As long as it's described as a system, the buzz word for a machine that does this, you have a chance. An old mental step case in the Patent Office involved a fairly complex process of drilling a bunch of bore holes and analysing data and looking for anomalies in the data to determine if there was oil down below. The patent application was turned down as being a "mental step". A fairly recent patent application for a similar process implemented by a computer was successful. It was patently distinguishable! Patent recently issued to IBM for a user interactive screen that had the patentably novel idea of highlighting the spots where you had to enter data, doing a check and if the data checked, taking away the highlighting.

Patents are expensive, they are territorial and they don't apply anywhere other than where you apply for them. They are fairly easy to hold invalid and there is a tremendous paucity of ability to search the software field in the Patent Office. On the other hand, once issued you can take your patent and go around "beating on" people for fairly inexpensive licences ($25-50,000). Given the cost of U.S. litigation, it's easier to pay the licence than go to court, so there can be some real value in getting patents in the first place. Patents are also fairly easy to avoid in most situations. If the patent is too broad it's almost certain that it will be held invalid later. There is a "doctrine of equivalence" principle in patent law and there have been some recent decisions that have narrowed the application of the doctrine considerably.

## BUREAU OF CUSTOMS AND THE INTERNATIONAL TRADE COMMISSION

Section 602 of the Copyright Law allows the Bureau of Customs to seize infringing copies if your copyright is registered and you deposit copies of that registration with the Bureau of Customs. Customs can also act on registered trademarks seizing infringing goods. The Semi-Conductor Protection Act has that provision as well but the Customs Department has decided that it will not seize infringing mask works unless there is an Order of Court or the ITC ordering it to do so. That may seem like a very unfair decision but if you think about it, you can look at a trademark and determine that this is a Gucci wallet or alternatively an infringing Gucci wallet. You can actually look at some copyright material and make some kind of decision. It's a little harder to envisage Customs Agents peeling back the layers of a chip and trying to decide whether it really infringes mask work. So, as we don't have the resources to adjudicate this or test it, right now we'll just wait for Court to tell us.

For tax reasons, a lot of the French perfume manufacturers sold the U.S. trademarks and distribution rights for the French perfumes to U.S. companies. Then they turned around and shipped the perfumes back into the United States anyway so the Congress in its wisdom said, "Hey! We will pass some laws that by virtue of depositing the trademark with Customs will hold the stuff up at the border".

The International Trade Commission (ITC), is a quasi-judicial body pursuant to s337 of the Tarrif Act 1930. It has the power to take action to protect American industry from unfair trade practices and methods of competition and it deals with copyrights, trademarks and patents. It's of interest for a couple of reasons. U.S. Courts are very slow. The ITC has, by the terms of the Statute Act, to render a decision within 12 months, or 18 months if it's a complex case. More importantly though, its Statutory Mandate is not to dispense justice and be neutral as is the Court, but to protect U.S. business. It is also an executive rather than a judiciary branch body and it was and is extensively used in the patent field because the Courts have been so willing to declare patents invalid. There is an understandable reluctance for a sister executive branch agency to rule that the Patent Office - another executive branch agency - made a mistake and should not have issued patent. There was a law suit filed against the U.S. Patent Office a couple of years ago for infringement of a patent in a filing system. The U.S. Patent Office, in its wisdom, raised as one of its defences that the patent was invalidly issued!!!

An ITC proceeding is an action in rem. You don't need the defendant present. You can issue an order excluding seven hundred computers labelled such and such. (Most other litigation requires you to have the defendant in court or at least requires you to be able to get jurisdiction for the defendant.) It's a highly political mechanism because it involves or includes the provision for Presidential veto. The President does not have to give any reason for his veto other than that it is in the interest of the country.

In addition to finding an infringement or violation the ITC has to find the infringement or violation as likely to injure or destroy effectively a U.S. industry. It also has to find, as a pre-requisite, that the relevant U.S. industry is economically and efficiently operated.

One of the leading ITC cases in the computer industry was an Apple case, Personal Computers and Components thereof. It's interesting because in the early days of Apple compatibles if you did an absolute check on the code it was something like eighteen percent identical. If you took into account location shifting and moving modules around a little bit you came up with something like twenty-five percent identical. The ITC would dissect the program and of the seventy routines in the Auto Start

ROM and the Apple Soft (the leading software there) there were thirty-two that Apple described as the most useful. Of those thirty-two, twenty-three were copied virtually identically.

Customs and ITC differ also in their exclusion abilities. It is possible to bring in ROMless mother boards through Customs. If the infringing ROM isn't in there it doesn't matter that there is no other use for the thing. Customs can exclude only the action infringing product. The ITC, on the other hand, can take a look and, under principles of contributory infringement, determine that there's nothing else you can do with an Apple mother board other than put infringing chips in it when it gets here. But that's not true of an IBM board because you can buy other copies of MS-DOS. You can buy Phoenix ROM Bio-Chips and put those in there, so an IBM mother board does have non-infringing uses that can't be excluded, whilst an Apple mother board does not.

## COMPUTER CRIME

Computer crime can be made to fit into a couple of traditional categories of wrongful acts; criminal acts that just happen to use the computer as a tool or crimes that are uniquely computer based. There is some specific Federal Legislation. One piece unplugged a hole in the Telecommunications Act that made it illegal to eavesdrop on a telephone conversation but not illegal to eavesdrop on data communications. So they unplugged that hole and now it's illegal to eavesdrop and pick up that stuff as well. There is also Federal Computer Crime Legislation that makes it illegal to break into Government computers or deal with computers for Federal Banks. (Federal Banks are different from State Banks, and are regulated by Federal Legislation.) Basically the principle is that all power resides in the States unless it's been expressly granted to the Federal Government. Unless the crime is against a Federally chartered bank or affects interstate commerce, such as the telecommunications system, the Federal Government has no power to act. What you find in computer crime legislation in the United States is a series of state laws that are beginning to fall into conformity as the States learn from one another.

One of the traditional problems was that theft of goods was well-defined. Theft of services was also well-defined in a lot of States but when you steal computer time you are not stealing goods - the computer still has it all; you are stealing services. Even the theft of services laws don't work so well if all you are doing is going in there and browsing, because you haven't taken anything of value. If you are going to enchain something and gain by it, it can be fraud but if all you are doing is going in and having a look around there may not have been a crime at all. So it's very difficult to write appropriate legislation.

1 Apple Computer Inc. v Franklin Computer Corporation United States Court of Appeals for the Third Circuit 714 F.2d 1240; 70 ALR Fed. 153; 219 USPQ (BNA) 113

2 Whelan Associates Inc. v Jaslow Dental Laboratory Inc. United States Court of Appeals for the Third Circuit 609 F. Supp. 1367 (ED Pa 1985); 797 F.2d 1222; 230 USPQ (BNA) 481

3 S.A.S. Institute Inc v S. & H. Computer Systems Inc. 605 F. Supp. 816 (MD TENN. 1985)

4 Broderbund Software Inc. v Unison World Inc. United States District Court for the Northern District of California, 648 F. Supp. 1127; 231 USPQ (BNA) 700

5 Digital Communications Associates Inc. v SoftKlone Distributing Corporation United States District Court for the Northern District of Georgia, Atlanta Division, 659 F. Supp. 449; 1987 US Dist. LEXIS 3701

6 "Personal Computers and Components" Investigation No. 337-TA-140, VSITC Publication 1504, (VSITC, March 1984)

---

**Martin Greenstein** works for Baker & McKenzie in its Chicago office