The John Marshall Journal of Information Technology & Privacy Law

Volume 16
Issue 2 *Journal of Computer & Information Law - Winter 1997*

Article 7

Winter 1998

Decloaking Development Contracts, 16 J. Marshall J. Computer & Info. L. 403 (1998)

Micalyn S. Harris

Follow this and additional works at: https://repository.law.uic.edu/jitpl

Part of the Computer Law Commons, Contracts Commons, Internet Law Commons, Privacy Law Commons, and the Science and Technology Law Commons

Recommended Citation
Micalyn S. Harris, Decloaking Development Contracts, 16 J. Marshall J. Computer & Info. L. 403 (1998)

https://repository.law.uic.edu/jitpl/vol16/iss2/7

This Symposium is brought to you for free and open access by UIC Law Open Access Repository. It has been accepted for inclusion in The John Marshall Journal of Information Technology & Privacy Law by an authorized administrator of UIC Law Open Access Repository. For more information, please contact repository@jmls.edu.

DECLOAKING DEVELOPMENT CONTRACTS

by MICALYN S. HARRIS†

I. INTRODUCTION

Development contracts are arrangements between a developer and a client pursuant to which the developer agrees to develop some form of computer software for the client. Developers and clients, or more frequently their respective legal counsel, often collide over issues relating to whether the contract is for "goods" or "services." If the contract is for goods, then the purchasing client would enjoy the protections of a "buyer" under Article 2 of the Uniform Commercial Code ("U.C.C."). If, on the other hand, the contract is for "services," Article 2 would not apply, and the parties' relationship would be governed by general contract and common law principles.

Proposed Article 2B seeks to resolve that issue by recognizing *sub silencio* that the development of custom software by a developer for a client is likely to be a mixture of "goods" and "services," and establishing default rules. These rules include rules for both the developer and the client concerning ownership of the resulting program when the record regarding the contractual relationship of the parties is either not authenticated or is incomplete in that it fails to provide material specifics.

The rules for conduct are largely covered under the warranty provisions discussed elsewhere in this Symposium. Section 2B-617 and Section 2B-501, to which Section 617 refers, deal with default rules when an authenticated record fails to provide specifics as to ownership of intellectual property rights in a program developed by a developer.

† Copyright 1997 Micalyn S. Harris, All Rights Reserved. The author wishes to thank Dr. Louis J. Cutrona, Jr., Ph.D., President of Winpro, Inc., who has generously provided technical information based on his 25 years of experience as a software consultant, developer and systems analyst, and has patiently reviewed this article for technical accuracy.

II. THE PROPOSED RULE[1]

Section 617(a) defines a "developer" as "a person hired or commissioned to create, modify, or develop a computer program," and a "client" as "a person [who] hires a developer."

Section 617(b)(1)(A) states that, in the absence of an authenticated record providing otherwise, a "developer retains ownership of the intellectual property rights [in a program] except to the extent that the program includes intellectual property of the client or the client would be deemed a co-owner under other law." Section 617(b)(1)(B) provides that, although the developer retains rights in the program, "the client receives a nonexclusive and irrevocable license to use the computer program information in any manner consistent with the agreement."

Section 617(b)(2) provides that if a client requests, the developer must notify the client of the use of independent contractors, and confirm, within specified time limits, that all applicable intellectual property rights have been obtained or state that it (the developer) makes "no representation about those rights beyond any stated in the agreement."

Section 617(b)(3) sets forth a default meaning of what the parties shall be deemed to understand when a developer grants a client "ownership of the intellectual property rights in the program." Subsection (A) provides that ownership in the program passes pursuant to the provisions of Section 2B-501. Section 501(a)(1) provides that if the information is in existence at the time of the agreement, ownership is transferred at the time the agreement becomes enforceable. Section 501(a)(2) provides that if the information is not in existence when the contract becomes enforceable, ownership is transferred when the information is "so far identified to the contract as to be distinguishable in fact from similar property even if it has not been fully completed and any required delivery has not occurred."[2]

Subsection (B) of 2B-617(b)(3) provides that the client receives the program free of restrictions on its use, and states that the client's rights may not be canceled by the developer after ownership vests in the cli-

[1] This description is based on the September 25, 1997 draft of Article 2B, the most recent available at the time this article was submitted. The National Conference of Commissioners on Uniform State Laws, *Uniform Commercial Code Article 2B Licenses* (Sept. 25, 1997) <http://www.law.upenn.edu/library/ulc/ulc.htm>.

[2] Section 501 refers to "the information," whereas Section 617 refers to "the program" or, in some sections, "the completed program." It appears that for purposes of the reference in Section 617 to the provisions of Section 501, "the completed program" of Section 617 would be seen as interchangeable with "the information" of Section 510. Possibly, a Reporter's Note will be added to clarify the point.

ent.[3] The Reporter's Notes state that the subsection provides "what amounts to an implied license for the client." Subsection (C) provides that the developer retains ownership of rights in pre-existing components or code and of components and code developed independently of the contract, but that there is an implied grant, to the client, of an irrevocable license "to use, consistent with the agreement, the components or code as part of the completed program delivered to the client."

Section 617(b)(4) provides "safe harbor" language which will result in application of the default rules for establishing ownership of the completed program. That language is as follows: "All rights, title and interest in the completed program will be owned by [named party], or words of similar import."

III. BACKGROUND AND DISCUSSION

Before discussing the specific provisions of Section 2B-617: Development Contract, it is useful to understand how software is developed. Conceptually, a developer develops software using methods similar to those of a lawyer when developing a contract.[4] A lawyer may go to a form book, or review contracts drafted by others and available as public documents, as a basis for a particular contractual arrangement. The work product of many lawyers is public, accessible, and available in electronic form. This work product can be captured and revised by interested lawyers via the mandate that public companies file annual reports with the Securities and Exchange Commission on Form 10K.[5] These reports must include copies of "material contracts."[6] An experienced lawyer may have collected her own form files, and refer to them as a starting point. Such provisions may be rewritten or revised to fit a particular client's needs and desires. In some instances, language from court decisions may be incorporated to provide a basis for clearly expressing the results the incorporating party wishes to secure. Sentences, and occasionally, whole paragraphs may be written "from scratch," however, lawyers rarely start with a blank page and write a complex agreement completely from scratch and without reference to other sources.

Copyright inheres from the moment of inception, and thus the "creative expression" aspects of a contract which a lawyer drafts are, at least

[3] When "vesting" occurs, and when the rights become non-cancelable, will depend upon the provisions of the governing contract, and to the extent reference to default provisions is required, application of the provisions of Section 2B-501.

[4] For a more detailed description, see Appendix I to this article.

[5] *See* Securities Exchange Act of 1934 §§ 13, 15(d), 15 U.S.C. §§ 78l, 78n (1994 & Supp. I 1995).

[6] A variety of examples of various agreements are publicly available in electronic form from the SEC's web site. SEC, *EDGAR Database* (last modified Jan. 15, 1998) <http://www.sec.gov/edgarhp.htm>.

arguably, copyrighted. Traditionally, however, lawyers have not relied upon assertion of copyrights to protect the commercial value of their contracts, even though these contracts probably fall within the ambit of a creative literary work as defined in the Copyright Act.[7]

Like a complex contract, a computer program is rarely, if ever, developed "from scratch." Usually, a program is developed based on code found in existing code in a developer's own library and examples found in manuals, books, professional journals and elsewhere. Programs may also incorporate commercial code libraries, which a developer has collected and developed, code developed using code generating tools, which again, may be commercially available or the developer's own creation.[8] In addition, code is generally written in a source code "language" that has to be compiled and linked by a compiler (generally a commercially available compiler) in order to become "executable" code. A compiler is, itself, a software program.[9]

Like a contract, computer code is complex, highly integrated, and interdependent. Inserting a single line or letter can significantly change how (or even whether) a program will run, just as inserting the word "not" in a key place in a sentence can radically change the meaning of a key contractual provision. A medium-sized program has as many lines as a jet airplane has parts, and while parts of a program, like parts of a jet plane or a contract, may have use in other contexts, a single "part" will not "operate" alone. The size of a program is not the only element of complexity, however. Even a relatively small program, especially if it is efficient, can be highly complex.

As a result of the nature of computer programs, software developers have declined to "sell" software. Rather, developers have looked to a combination of protection based on copyright and utilization of basic principles of contract law to protect and enjoy the commercial value of

[7] See 17 U.S.C. §§ 101, 102 (1994). The difference between software and other types of literary works is recognized in The Copyright Act of 1976. For example, the concept of a "work for hire" appears to apply to software in only limited circumstances, if at all. *See* 17 U.S.C. §§ 101, 102; RAYMOND T. NIMMER, THE LAW OF COMPUTER TECHNOLOGY ¶¶ 4.04, 4.07 (1997).

[8] Whether code generated by a commercially-available code generator is "owned" by the licensor of the code generator or the licensee may depend upon the provisions of the license for that development tool. Some licenses provide specifically that the generated code belongs to the licensee. Other licenses may provide varying rights and limitations on how the licensee may incorporate and use the generated code.

[9] To date, we are not aware of any challenges to a developer's ownership of the compiled code, but it would be theoretically possible, although it may be commercially unacceptable to license a compiler under a contract, i.e., a license, stating that the generated code belongs to the compiler's creator or owner, and restricting use as the licensor saw fit.

their creative labors by licensing, rather than selling, their software.[10]

Virtually all software developers are both licensors and licensees. Anyone who uses Windows is a licensee. Software developers are licensees of compilers, software development kits, commercially available code libraries (e.g. dynamic and static link libraries), and a host of "tools" and other licensed items.

Software developers need to be able to rely on the licenses they receive to use the programs of others, as well as on the licenses they grant to protect the commercial value of their programs. They are accustomed to reading licenses, and generally treat licensed items as they wish to have their own licensed items treated. When a license is insufficiently broad to permit a licensee's contemplated use, the would-be licensee either negotiates for what it needs, or finds an alternative route to meet its needs. Most developers look to the basic principles of contract law, combined with the pressures of the marketplace, to commercially exploit their software creations.

Developers also understand that unreasonable licenses have effectively "killed" products.[11] For example, several years ago, Borland introduced a new version of its flagship software development product. Developers who purchased the new product promptly discovered that Borland had changed its license to provide that a developer would be permitted to distribute up to 10,000 copies of any software developed using the new product, but that for more than 10,000 copies, a separate license would have to be negotiated. Developers realized that if they used the product (as intended) to develop applications that turned out to be very successful, they would be "over a barrel" when they tried to negotiate the sell of the 10,000th copy. Some "wrote" to Borland on-line, while others returned the product for a refund, or simply refused to purchase the product. Within six to twelve weeks, Borland posted a revised license. Despite the revision, a significant segment of the developer community found substitute products or other ways to meet its needs, and Borland lost a valuable market share that it never regained.

Thus, within the industry, it is generally accepted that, for commercial reasons, licensors must draft licenses thoughtfully. Licensors understand that if a license fails to grant essential rights, the opportunity to obtain significant revenue from the product may be lost. The software industry has found that the licensing model works well. Licensing offers maximum flexibility in an industry in which change is constant and rapid. To the extent that a uniform law confirms the enforceability of

[10] *See, e.g.*, "Shrink-wrap" licenses on commercial software products of Microsoft, Borland, IBM, etc., and "Read-me" licenses of on-line service providers.

[11] *See* Al Stevens, *Borland Nonsense: Ready, Aim, Shoot! Restrictive Licensing for Borland International's C++ 4.0 Programming Language*, DR. DOBBS J., April 1994, at 115.

contractual arrangements, a uniform law is likely to be welcomed. To the extent that a uniform law reduces flexibility and makes enforcement of contracts less predictable, the uniform law presents a potential for disruption and is less likely to be welcomed by the industry.

IV. SECTION BY SECTION ANALYSIS

The default rules of Section 2B-617(b) apply only if there is an agreement between a developer and a client for the development of a computer program, and only as between the developer and the client. The parties are, therefore, left to their own devices to allocate risk, if any, of third party challenges.

Section 2B-617(b)(1)(A) provides that "[i]f an agreement requires the development of a computer program, as between the developer and the client, in the absence of an authenticated record transferring title in intellectual property rights, the developer retains ownership rights except to the extent that the program includes materials of the client or the client would be considered a co-owner under other law." The provision reflects the law as established by the Copyright Act,[12] and also appears to reflect what the parties would intuitively expect.

The Reporter's Notes not only include a reminder that federal copyright law provides that unless there is an express transfer of copyrights in a writing, those rights remain with the developer, but also that the rule was included in the Copyright Act of 1976 after "substantial deliberation." The Reporter's Notes emphasize the importance to the developer of retaining primary rights in its intellectual work product unless specifically and clearly transferring those rights, as well as the public policy concern with protecting small developers. Thus, both federal law and public policy support the correctness of the provisions of Section 2B-617(b)(1)(A), and these provisions have generally been uncontroversial.

The default rule of Section 2B-617(b)(1)(B), providing that "the client receives a nonexclusive but irrevocable license to use the computer program information in any manner consistent with the agreement," is clearly intended to assure that the client receives the right to use a program in the manner anticipated. In general, a developer will be asked to provide a "completed program," and, in the absence of a specific provision, both developer and client anticipate that the client will receive a license to use the completed program "in any manner consistent with the agreement."[13]

[12] 17 U.S.C. § 204(a) (1994).

[13] As noted above, if the reference to "the computer program" rather than "the completed program" is intentional, adding a "Reporter's Note explaining the rationale for referencing "the computer program" here, rather than "the completed program" would be helpful.

The default rule's assumption in Section 2B-617(b)(1)(B) that there is some "agreement" with which "use" of the "completed program" can be "consistent" is reasonable. The fact that a client and a developer come to some agreement does not mean that, in a particular case, their agreement will encompass how the program is to be used. It follows, therefore, that whether a challenged use is "consistent with the agreement" may be problematical, and may raise questions of fact. Such issues, however, are no different than other issues of fact resulting from incomplete contractual provisions giving rise to questions of fact. Moreover, to the extent the default provisions become the standard, the proposed default provisions may encourage communication regarding the issue.

As a practical matter, a client typically advises a developer of the intended use of the program to be developed, and expects the developer to provide a completed program. In the absence of other explicit arrangements, client and developer also expect that the developer intends to grant a license to the client which will allow the client to use the completed program as the developer understands the client contemplates and expects.

Thus, Section 2B-617(b)(1)(B), particularly if the reference to "the computer program" is changed to "the completed program," has potential benefits to both the developer and the client. To the extent the provision encourages a client to communicate regarding the intended use of the program, the developer is placed in a better position to serve the client's needs. The revised provision would also assist the client by assuring that the developer is aware of the required scope of licenses relating to code incorporated pursuant to third party licenses.

The provision has potentially salutary public policy as well as business effects. From the client's perspective, these effects include imposing on the developer the responsibility for assuring that the scope of the license covering a licensed computer program developed for a client is adequate to encompass the use to which the client will put the computer program. From the developer's perspective, these effects include enabling the developer to make an adequate review of such licenses in light of its client's needs and desires.

The Reporter's Notes indicate that an alternative scope to the implied license would be to make the rights "unrestricted," as opposed to limiting the implied rights to "uses consistent with the developmental purposes." "Unrestricted" rights may, however, be misunderstood or misinterpreted to support permitting a client to use a program not only as an integrated whole, but also, to use selected pieces of the program elsewhere. Such separate use poses problems under many licenses, as for example, when third party code libraries are incorporated in a program. Thus, from a developer's point of view, the problem is not whether the use is, as the Reporter states, "consistent with the developmental purposes," but whether the use is separate and apart from the (completed) program. For example, most licenses of commercial code libraries permit use of the code library when incorporated into a program, but in the absence of an additional license, do not permit the party to whom the program which includes such a code library was licensed to remove the code library from the (completed) program and use it elsewhere. Thus, the issue is not whether the use is "consistent with the developmental purpose," but whether the use is "consistent with the agreement" to deliver a (completed) computer program.

Current law and practice give rise to the expectation of an implied license to use a (completed) program "consistent with the agreement." Clients understand that if they wish to obtain rights to use a completed program other than as the completed integrated program, they must make further inquiry. Implying an "unrestricted" license is contrary to current law and practice, and is likely to impose unexpected and unreasonable burdens on unwary developers, while giving windfalls to sophisticated clients.[14]

Section 2B-617(b)(2) provides that a developer will, if requested, provide what amounts to additional assurances regarding use of independent contractors by the developer. The U.C.C. Subcommittee on Software Contracting of the American Bar Association's Business Law Section ("Subcommittee") recommended deleting this section on the grounds that the issues are fully and appropriately covered by Section 410: Warranties. The Subcommittee noted that there is nothing about warranties relating to the transfer of intellectual property rights that makes them different from other warranties. Moreover, there is no legal or public policy reason to differentiate these obligations from other contractual obligations, and therefore, the provision was redundant. Even in the absence of the provision, a developer would be responsible for obtaining whatever rights it needs to obtain from subcontractors to make whatever representations it makes to the client.

In drafting meetings, however, client representatives expressed concern that developers may not be aware that their subcontractors retain intellectual property rights to work delivered to the developer for incorporation in the computer program to be delivered to the client. To meet this concern, Section 2B-617(b)(2) grants clients a new right to demand and obtain additional assurances that the developer has done what it needs to do to grant the rights promised to the client. In effect, the provision gives clients a right to demand and obtain reiteration of rights granted by the contract. Where the contract is silent, the provision gives clients the right to demand articulation of the point after the contract

[14] *See infra*, discussion of Reporter's Note regarding deleted subparagraph (D) (discussing several similar issues).

has been signed and possibly after work has gone forward on the basis of a contract that was silent as to the rights the client subsequently decides it wishes to seek. As the Reporter notes, this right to further assurances is not found in the current law; it provides new and additional protection to clients.

Section 2B-617(b)(3) sets forth default rules applicable under limited circumstances, implicitly recognizing that no default rules exist for circumstances not specified. The subsection postulates the existence of "an authenticated record" providing that ownership of intellectual property rights in the program pass to the client. This is in accord with applicable intellectual property law, which provides that ownership of intellectual property rights requires an express written transfer of rights.

More specifically, Section 2B-617(b)(3) sets forth what the parties shall be deemed to understand it means for a developer to grant a client "ownership of the intellectual property rights in the program"[15] when there is a contractual provision so stating, but the provision fails to deal with various other issues. (Again, there is an elegant variation between the reference here, to "ownership of the intellectual property rights in the program" and the references in subparagraph (A) and Section 2B-617(b)(4), which refer to "ownership [rights in] the *completed* program" (emphasis added). If the variation is intentional, it would be helpful, as indicated above, to include a Reporter's Note explaining the rationale for the variation. If the variation is inadvertent, changing all references to "the program" and "the computer program" to "the completed program" would clarify the provision.)

Subparagraph (A) of Section 2B-617(b)(3) deals with the situation in which ownership of the completed program passes, but makes no reference to the timing of *when* ownership is to pass. Section 501 makes the transfer of ownership rights subject to transferee's performance of its contractual obligations. Section 2B-617(b)(3)(A) states that if transfer of ownership of a completed program is to occur, then the transfer will occur pursuant to the provisions of Section 2B-501. As a practical matter, it appears that only 2B-501(a) and 2B-501(a)(1), will apply, as Section 2B-501(a)(2) applies only if "the information" (which must be read, for purposes of Section 2B-617(b)(3)(A) as meaning, "the completed program") is "not in existence." As the provision is referenced only in connection with the transfer of intellectual property rights in a completed program, transfer pursuant to Section 2B-501 must refer to information in existence, i.e., a completed program. By implication, if a completed program is to be delivered, and the program is not completed, ownership

[15] *See* U.C.C. §§ 2B-617(b)(3), 617(b)(3)(B) (Proposed Draft, Sept. 25, 1997), referring to "ownership of intellectual property rights in the program." *See also* U.C.C. §§ 2B-617(b)(3)(A), 617(b)(4) (referring to "ownership of the completed program").

of the intellectual property rights does not pass. This is in accordance with current law on point, which says that where a computer program is being developed for a client, title cannot pass until the program is fully completed and delivered.[16]

Subparagraph (B) of Section 2B-617(b)(3) provides that the client receives "the program" (presumably, and if the safe harbor language of Section 2B-617(b)(4) is used, necessarily, "the completed program") free of restrictions on its use, and that its ownership rights cannot be canceled by the developer once ownership vests in the client. If "free of restrictions on its use" means use of the completed program as an integrated whole, the provision appears to reflect the usual expectations of both parties. If, however, "free of restrictions on its use" is understood to mean that the client may take the program apart and reuse pieces elsewhere, such use is not what developers expect, and the provision may give rise to conflict and difficulties.

The problem may be ameliorated by making all of Section 2B-617(b)(3) relate to transfer of ownership rights in a "completed program." The current provision, which includes varying references to "the program" and "the completed program," may, in fact, reflect a drafting oversight. For Subparagraphs (A) and (C) both refer to a "completed program." The omission of the word "completed" from Subparagraph (B) would not be objectionable to developers in most cases if the unrestricted license was for use of the completed program (as an integrated whole). Moreover, the reference to a "completed program" would dove-tail with the "safe harbor" language provided in Section 2B-617(b)(4).

Subparagraph (C) of Section 2B-617(b)(3) provides that the developer retains ownership of pre-existing code and code developed independently of the contract but grants the client an irrevocable right to use the code "consistent with the agreement . . . as part of the completed program delivered to the client." The provision reflects current law as it relates to ownership, and practical reality in terms of assuming a grant of an irrevocable license to use code incorporated in a completed program.[17]

Thus, with regard to Section 2B-617(b)(3), the Commissioners may also wish to consider two clarifying changes: (i) consistent references to "the completed program" rather than varying references to "the computer program" and "the completed program" and (ii) adding a Reporter's Note indicating that for purposes of Section 2B-617, the references in Section 2B-501 to "the information" are to be read for purposes of Section 2B-617 as references to "the completed program." These suggested changes will clarify the scope of what the client receives, i.e., a nonexclu-

[16] *See generally In re* Amica, 135 B.R. 534 (Bankr. N.D. Ill. 1992).

[17] *See generally In re* Bedford Computer, 62 B.R. 555 (Bankr. D. N. H. 1986).

sive, irrevocable license to use the completed program in any manner consistent with the agreement.

The previous draft of Section 2B-617(b)(3) included a Subsection (D), which provided that the client receives ownership of "generally applicable components or code, including development tools or the like, developed in performance of the contract, but the developer has an irrevocable nonexclusive right to use in other contracts generally applicable components or code that do not include confidential or otherwise proprietary information of the client." The Reporter's Notes in the earlier draft stated that the provision was an effort to balance the developer's right to retain ownership in pre-existing materials and the client's desire to own "components or code, including development tools or the like, developed in performance of the contract." This allocation of rights was to be achieved by granting a developer's client ownership of such "components or code" with an irrevocable nonexclusive license back to the developer to use such components or code to the extent such use would not "include confidential or otherwise proprietary information of the client."

Both developers and licensees objected to the provisions of subparagraph (D). Developers are aware that clients often take the position that if code was developed "on their nickel," they want to own it. Such clients see software programs as being made up of easily identifiable "components" and "development tools." They are aware that language which transfers ownership of a completed program transfers such ownership in the completed program as a whole, does not automatically include ownership rights to routines separate and apart from their function in the program, and clearly does not include "development tools," which is an uncertain term that potentially includes routines that are not part of the completed program (e.g., a testing routine developed to test the functioning of the completed program, but not incorporated in the completed program).

Developers objected to subparagraph (D) on at least two grounds. First, it is contrary to current law and current custom in the industry. Under current law and practice, transfer of "all right, title and interest in the completed program" transfers all right, title and interest in the completed program, period. It does not transfer all right, title and interest in "generally applicable components or code" or "development tools," both of which are flexible terms, and neither of which have a generally agreed upon meaning in the industry. Current practice eliminates the problems created by the absence of a generally accepted definition of these terms by requiring the parties themselves to define what it is they expect beyond current law. The proposed expanded meaning of what has been considered in the industry to be clear and plain language would give to sophisticated clients rights they now must notify developers they want, that is, rights to something other than all right, title and interest to the completed program as an integrated whole. In addition, these are rights which clients often cannot obtain by negotiation, and rights which, if they are granted, must be considered in calculating a fair price. As described in more detail in Appendix I, in order to be able to determine "ownership" of each piece of code, a developer would have to monitor and record in detail every step of the development, a process likely to increase the cost of a development process, possibly beyond the benefit to either party. Thus, Subparagraph (D) would give sophisticated clients a windfall and be a trap for the unsophisticated and unwary developer.

Second, assuming that "component" could be better defined and that granting the additional rights would not affect the price to the client, whether or not a particular component or tool was developed "in the performance of the contract," may be a difficult determination, and will always give rise to an issue of fact.[18] Thus, the provision is likely to spawn, rather than reduce, litigation. One objective of Article 2B is to improve clarity and reduce litigation. The inclusion of subparagraph (D) seems likely to have the opposite effect.

Clients objected to the provision on the ground that as clients, they do not give developers a license back to use what was delivered to them, and do not wish to find themselves granting such a license back by default.

In short, both developers and clients have objected to granting by default what now must be granted explicitly. It may be that frequently there is no clear understanding as to what a grant of "ownership rights in the completed program" conveys. It appears however, that attempting to encompass under default rules more than the plain language of the safe harbor provisions of Section 2B-617(b)(4) suggests is intended, i.e., all right, title and interest in the completed program as a whole, is not a resolution likely to be supported by either developers or their clients.

In addition to agreeing that the proposed provision does not conform to current law or practice, the Subcommittee also noted that the proposed Subparagraph (D) does not comport with several of the overarching principles guiding the drafting of Article 2B. These principles include adherence to basic principles of contract law, codifying existing law and custom, elucidating the reasons for any changes in the law required by public policy, and improving clarity, thereby reducing litigation. Accordingly, the Subcommittee supported deletion of Subparagraph (D) in its entirety and suggested that ownership of rights

[18] *Id.* at 569 (citing 4A COLLIER ON BANKRUPTCY (Lawrence P. King ed., 14th ed. 1978) §§ 70.25[2], 70.39[3] regarding the requirement that intangible property be separately identified or segregated, in order to be claimed, noting that it was not enough "to show merely that the funds or property came into the bankrupt's business or . . . even that the funds or property are contained somewhere in the bankrupt's estate" in order to claim ownership).

in an application, without more elaboration, reflect current practice and expectation by granting ownership in the application as a whole, no less, and no more.

Section 2B-617(b)(4) states that the language: "All rights, title, and interest in the completed program will be owned by [named party]," will transfer ownership of intellectual property rights to a client (or developer) if it is in an authenticated record. Similar language may be substituted provided the words are "of similar import." Note that the language recites that the transfer of rights is in "the completed program." Thus, if portions of Section 2B-617(b)(3) are to apply to other than a completed program, a Reporter's Note clarifying the reasons for the broader scope of portions of that Section would be helpful.

An additional clarification indicating (if that is the case) that the reference to "the information" of Section 2B-501 may be read, for purposes of Section 2B-617 as a reference to "the completed program" would also be helpful.

V. SUMMARY AND CONCLUSIONS

Section 2B-617: Development Contracts of proposed Article 2B deals with an important and often contentious group of issues. These issues involve complex arrangements between developers and clients who often have widely differing levels of sophistication concerning the legal aspects and implications of their arrangements regarding ownership rights in the intellectual property created pursuant to a development contract.

The process by which software is created makes granting rights in a completed program relatively straightforward. Granting rights to "re-use" pieces in other contexts, as well as determining the origin and ownership of particular pieces of code, is difficult and expensive, possibly beyond the benefit to either party.

With the exception of the possible problems arising from inconsistent references to passage of ownership in a "program" in Section 2B-617(b)(3) and Section 2B-617(b)(3)(B) and a "completed program" in Section 2B-617(3)(A) and (C), and clarification that "the completed program" of Section 2B-617 is "the information" to which Section 2B-501 refers, the wording of Section 2B-617 in the September 25, 1997 draft appears to reflect current law and practice in the industry. As such, it appears likely to enable a developer to shoulder the responsibilities a client reasonably expects and enable a client to obtain what it reasonably expected in the absence of specific negotiation and bargaining. The result appears to be a viable formulation which both clients and developers can comfortably support.

APPENDIX I[19]

The following example may be useful in gaining an appreciation of how code is developed. The example involves a relatively small (about 1000 lines of source code) application (which may or may not be a "component"). The function of the application is to install another program. The application is part of the master disk which the developer provides to the client, and part of the application delivered by the client to its customers. The customers however do not retain a copy of the application; its final step is to delete itself, after installation, in order to avoid taking up unnecessary space on the customer's hard drive.

The application is generated as follows: As part of the Software Development Kit ("SDK") which Microsoft provides for Windows 3.1, there is a Set Up Toolkit. Developer used the Set Up Toolkit to make an installer for Client A. Client A received rights in the finished product as a whole, including source code, in compressed form, for the file that controls installation.[20]

The installer consists of 13 files which are redistributed from the Set Up Toolkit and not customized. Of these, 9 cannot be customized and 4 can be modified (but were not). The next three files are generated by another tool which Microsoft includes in the Toolkit. These three files control parts of the installation.

The developer then created a Dynamic Link Library ("DLL", in C, a source code "language") by customizing source code which Microsoft provided as an example. That code was then complied and linked. About 70% of this source code was unchanged from what Microsoft provided.

The final file, the installation script, was written in source code (in a dialect of Basic, another source code language). It also was based on an example provided by Microsoft. The initial rewrite of this file involved about 40% new code, some of which was simply changing the names of variables to make them more congenial to the developer's programming style.

The total, 13 files, plus three files, plus the DLL plus the installation script file, was incorporated into the application delivered to Client A.

[19] Copyright © 1997, Louis J. Cutrona, Jr. and Micalyn S. Harris. All rights reserved.

[20] "File" is used to indicate code and/or data grouped together for convenience, usually the convenience of the developer or the end user. What is placed in a particular file is flexible; usually the decision of the developer, who may or may not follow conventions established by others (e.g., if working in a Windows environment, conventions established by Microsoft). A "file" may have other "files" within it, just as one can place several paper file folders in a single (larger) file folder. The designation "file" is not a term of art with a fixed meaning, and new technologies such as "compound files," "storages," and "streams" may make the concept of a file obsolete or change its meaning from what is currently understood.

The next installer, for Client B's first project, started with Client A's installer. It had the same initial 13 files, the next three files differed from Client A's version by about 30%. The DLL also required a 20-30% rewrite, part of which was changing the name of the client, part of which was adding new features. The installation script required a 20-30% rewrite, again in part just changing the name of the client and the programs being "called."

For Client C's first project, developer started from Client B's installer. It used the first 13 files, unchanged. The next three files were changed about 20% from the previous version. The changes to the DLL involved deletion of 20% of the code and changing another 5%. The installation script required about 20% new code and another 5% of changes.

Next, for Client A, developer did another project, under a separate contract, which required an installer. For this installer, developer started from the original installer for Client A, but took the DLL and script from Client B's installer.

Next, for Client C, for a second project, developer did a new installer, starting with Client C's earlier installer. Developer made no changes to the first 13 files, made changes in 5% of the next three (generated) files, and no changes to the DLL (from the DLL used for Client C's earlier installer). In the installation script file, which consists of about 300 lines of code, one line was moved from one place to another and five new lines of code were inserted.

Microsoft clearly owns the first 13 files. Microsoft also owns the tool kit which was used to generate the next three files. The tool kit (at least arguably a "development tool") is not redistributable and is not incorporated in any of the installers. Microsoft also arguably owns the copyright on approximately 50% of the installation script and on approximately 50% of the source code for the DLL.

The installers also install files which Microsoft owns, including static link library files (incorporated in the files to be installed) and any redistributable DLLs required by the application (other than those required by the installer). In the case of Client C, the installer includes an additional DLL which belongs to American Online ("AOL") and is incorporated by developer under a contract between developer and AOL and redistributed to Client C's customers under a contract between Client C and AOL.

Under current law, "ownership" of the copyrights to the total applications for Clients A, B, and C includes ownership of the installer as part of the application. It does not, in the absence of a specific understanding granting such rights, include the right to remove the installer from the application and use it elsewhere.

Note that delivery of "copyrights" or "ownership" of the portions of the installer which were "new" for each client would not, without the other portions of the installer, produce a meaningful application for any of them. Moreover, use of "the installer" outside of the completed program delivered to each client would require additional code in order to integrate it into the new application.

Other problems arise if a client insists on "ownership" of "code . . . developed in performance of the contract." For example, who "owns" a line of code created for one client and reused, in a different position, for another client? The line is not "new" in the second application. If it is "owned" by the party for which it was originally created, but in a second installer its position is new, and that changes how the installer functions, is the line of code "owned" by both clients if the arrangement with each is "ownership" of everything "created for" that client?

Developers would also face an enormous administrative challenge and burden if every client of a developer decides (or Article 2B of the U.C.C. were to provide) that grant of ownership rights in an entire application also grants rights in each piece of code created for that client. Keeping track of which line or part of a line is owned by whom, and being able to determine who "owns" what, would be a significant and expensive undertaking, unlikely to be worth the cost to clients. For Article 2B to impose such a requirement seems likely to increase the cost of developing software with no discernible compensating benefit.

Lack of a generally agreed upon set of definitions also gives rise to difficulties. For example, the word "component" has different meanings to different people, and different meanings in different contexts. Under current law and practice, if a client wishes to impose a requirement that the client be able to use a "component" outside of the delivered application, the client is required to articulate what, exactly, is intended by the word "component" and both parties reach an understanding as to what is expected. Developer and client can then determine together whether the expectation is reasonable and whether the development process will have to be adjusted, use of certain pieces of code eliminated, permissions obtained, etc. Price would then be adjusted to reflect what is desired.

To summarize, a computer program functions as an integrated whole. "Pieces" do not function separately, and virtually identical code can function differently depending upon its position in a program. Accordingly, while grant of ownership rights in a completed program makes reasonable practical and business sense, granting by default rights in "components" or "development tools," both terms with flexible meanings and having no generally agreed upon definitions, is likely to create confusion where certainty now exists. Basic principles of freedom of contract, and codification reflecting use of plain language to mean what it appears to mean and currently means in commercial practice, support deleting Subparagraph (D) and confirming that a grant of rights in the completed program will continue to grant rights in the completed program, as a whole; no more and no less.