UIC John Marshall Journal of Information Technology & Privacy Law

Volume 4
Issue 1 *Computer/Law Journal - Summer 1983*

Article 5

Summer 1983

Project Controls in Computer Contracting, 4 Computer L.J. 133 (1983)

Duncan M. Davidson

PROJECT CONTROLS IN COMPUTER CONTRACTING†

by DUNCAN M. DAVIDSON*

TABLE OF CONTENTS

* Sc.B., Brown University; J.D., University of Michigan. The author is Chairman of the Committee on Proprietary Rights in Software, Division on Law Relating to Computers, ABA Section on Science and Technology, and is associated with the law firm of Irell and Manella in Los Angeles, California.

The concept of "controls" is crucial to auditing and proper management. It should also be crucial to computer contracting. Many of the horror stories of computer procurements could have been avoided by proper project control. Project controls can be implemented in the computer contract itself. Such controls greatly increase the computer user's chances for a timely and successful procurement.

I. APPROACHES TO COMPUTER CONTRACTING

In the beginning, when computers were unfamiliar, computer vendors defined the manner of contracting for computers. Computer users, for the most part, signed the vendors' form contracts without change. In this manner, computer users entrusted critical parts of their businesses to vendors. Computer vendors had developed contractual terms which allocated practically all of the risk of a failed computer procurement to the computer user. The techniques for such allocation are relatively straight-forward.

While many of these procurements worked, some failed dramatically—with the user assuming the risk. For example, Triangle Underwriters had been in the insurance business for forty years, but within three years of switching from an IBM system to a more pervasive Honeywell system, which never worked, it was bankrupt.[1] Catamore Enterprises had been in the jewelry business for over thirty years, but three years after first hiring IBM to automate its production control systems, its hopes for an aggressive expansion of business were dashed by poor inventory control and a still inoperative computer system.[2] The Glovatorium, Inc., was a first-time computer user which needed a route accounting system. NCR agreed to provide a system still in the testing stage which had never been

1. Triangle Underwriters, Inc. v. Honeywell, Inc., 457 F. Supp. 765 (E.D.N.Y. 1978), *aff'd in part*, 604 F.2d 737 (2d Cir. 1979).

2. IBM Corp. v. Catamore Enter., 548 F.2d 1065 (1st Cir. 1976), *cert. denied*, 431 U.S. 960 (1977).

used or programmed for route accounting. Even though parts of the software never fully worked, components of the hardware never overcame repeated problems, and support services were inadequate, NCR maintained that because of the allocation of risk provisions under its form contracts, most of the responsibility was Glovatorium's. During arguments over the amount of a jury's verdict in favor of the user, the judge noted that "if there ever is a reason for a holding that these [allocation of risk] provisions in these contracts should not be enforced because of unconscionability, this is the A-number-one case."[3]

These situations represent the tip of the iceberg; many other failed procurements have been litigated and reported, and many more have been settled out of the public eye. In many of these cases, the vendors may not have been at fault, and in others, the users' mistakes may have compounded problems created by the vendors. In most of these cases, however, regardless of where fault lies, the users have suffered for failed procurements. By following the vendor's contracting procedures, these users have been left unprotected.

Some computer users are changing their computer contracting procedures to protect themselves. The glamour of computers has worn off. Many users are procuring their second or third computer system and are now treating computer procurement with the same hard-nosed attitude that they use for other business deals.

The techniques for protecting the user, however, are not obvious. Several generations of thinking have passed. The General Services Agency of the federal government has taken one approach, which has been emulated by many state and local governments, but which is not altogether suitable for private companies. Other users have applied the same type of procedures which they apply to other procurements, but with mixed success. A more promising approach is a systems approach in which the user defines in the contract the performance it wants. Such a contract includes functional specifications for the system and a comprehensive acceptance test to determine when the system is ready to perform the specified functions.

An even more successful approach is to incorporate project control concepts in the contract. By controlling the procurement, the user can maximize its chances of having a successful procurement, one that is on time and within budget. Controls that can be placed in a contract include a detailed project timetable, the segregation of performance into separately usable components, and inclusion of

3. Reporter's Transcript of Proceedings at 7, The Glovatorium, Inc. v. NCR Corp., No. C-79-3393 WWS (N.D. Cal., May 1, 1981).

contractual remedies for failed performance other than termination for breach and litigation.

This Article will discuss the different perspectives of vendors and users and will present a way to meet most of the concerns vendors have regarding allocation of risk and most of the concerns users have about satisfactory performance.

II. APPLICABLE LAW

Commercial law defines the manner in which a vendor in a contract will allocate risks and a user will implement project control. For most commercial transactions in the United States, a uniform body of law will apply, the Uniform Commercial Code (U.C.C.).

The U.C.C. applies to "transactions in goods." A computer itself is a good, and a computer with operating system software as a package is also a good. It is less clear that software is a "good," or that a license of software is a "transaction in goods." Nevertheless, most software licenses are drafted as if the U.C.C. would apply. The U.C.C. is often applied by analogy even if the transaction to which it is being applied is not technically a "transaction in goods."

Although the U.C.C. has been adopted almost uniformly throughout the country, there are local variations, and the law applying it by analogy is sketchy. Thus, attorneys drafting software contracts must consider the general law of the jurisdiction in which the transaction occurs in addition to applying general U.C.C. provisions.

III. VENDOR CONTRACTS

The U.C.C. has a number of provisions which allow a seller to allocate risks of a failed computer procurement to a buyer. These provisions were drafted with some care and are not considered to be unfair to a buyer. Instead, the underlying assumption of these provisions is that the price being charged by the seller has some relation to the risks being undertaken by the seller in providing the product. Thus, to the extent the seller can allocate risk to the buyer, the seller can lower its price.

The U.C.C. does not allow a seller to allocate risk to such an extent that the buyer fundamentally has no remedies. Moreover, often sellers will not follow the U.C.C. allocation of risk provisions to the extreme, particularly in areas such as warranties, but will undertake certain risks in order to remain competitive in the marketplace. It is essential that the allocation of risk provisions be understood and that the attorney for a vendor drafting a form contract first considers

drafting a contract which fully allocates risks to the buyer as permitted by the U.C.C.

A. Consequential Damages

The U.C.C. allows a seller to allocate all liability for consequential damages to a buyer. Consequential damages are damages that occur as a consequence of a breach because of collateral transactions, such as lost profits resulting from lost sales of the buyer. Traditionally, a buyer could not sue a seller for consequential damages in the absence of special circumstances, regardless of the contract. Consequential damages are often the source of unnecessary negotiation in a contract.

Under the traditional rule, the seller is liable for consequential damages only if the seller had special knowledge that the buyer would suffer these damages, and if the seller "tacitly" or openly assumed the risk of their occurrence. The U.C.C. purports to modify this standard by removing the "tacit" or open assumption of risk requirement. The courts, however, tended to apply the law prior to the U.C.C. to interpret language of the U.C.C. As a consequence, some vendors use the following manner of phrasing the exclusion of consequential damages:

> In no event will seller be liable for indirect, special or consequential damages even if seller has been advised of the possibility of such damages.

Consequential damages are distinct from incidental damages, such as storage costs or the costs of shipping the defective product back to the seller. Consequential damages are also different from proximate damages, which are the direct damages the buyer normally suffers from a breach. Proximate damages can consist of the difference between the purchase price of the goods and the price it could cost the buyer to purchase the goods from someone else. If the buyer retains the goods despite defects in them, the proximate damages consist of the difference between the value of the goods and the cost of the goods.

Clauses excluding liability for consequential damages often also exclude liability for "special" and "indirect" damages. Special damages, which must be specially pleaded in order to be awarded at trial, consist of damages which result from special circumstances, in contrast to general damages which ordinarily result from the breach in question. Indirect damages are damages which occur because of some intervening cause, such as a fortuity of nature or the independent acts of a third party. Despite these different theories, courts tend not to be overly technical or overly concerned as to

whether a certain damage was the result of an intervening cause or special circumstances. Instead, the courts tend to focus on whether the damages are those which normally follow from a breach and are foreseeable in that sense or whether the damages are unforeseeable and unusual.

B. Limited Damage Remedy

The U.C.C. allows a vendor to limit its liability for damages to an upper dollar limit, so long as that dollar limit is not so low that the buyer is left without remedies. A traditional limit is the purchase price or, in a time-payment situation, the amount paid over the last six months or year. Thus, at worst, the seller will have to refund the money paid in return for the nonfunctioning product. This limitation is often written in the following fashion:

> Seller's liability for damages to buyer for any cause whatsoever, regardless of the form of action and whether in contract or in tort, shall be limited to the purchase price in effect when the cause of action arose for the Product that caused the damages or that was the subject matter of or directly related to the cause of action.

C. Limited Warranty Remedy

The seller is allowed to provide an exclusive remedy for any problem with its products. This remedy can be as limited as a promise to repair or replace the product and is often written as a warranty. In order to be correctly drafted under the U.C.C., this warranty must state that it is the exclusive remedy; otherwise, it will be deemed to be an additional remedy and not the only one. This warranty can have exceptions for such matters as misuse of the product or use of the product with other products that cause problems. The warranty can also be limited in duration.

The warranty cannot be limited, however, so that circumstances cause it, in the language of the U.C.C., to "fail of its essential purpose." This phrase is not enlightening. It probably means that the vendor should at least be able to replace or repair the product. Thus, a product with a design defect (for which replacement is ineffective) or which a vendor fails to fix is one for which the limited warranty fails of its essential purpose.

The consequences of a limited warranty failing of its essential purpose are unclear. At a minimum, the remedies of the buyer are no longer exclusive, and the buyer would be able to recover monetary damages for breach of warranty. Since the vendor will be limiting its liability in other respects, particularly as to consequential damages, one argument which has met with some success is that if

the limited warranty fails, all of the limitations on liability are stricken from the contract. The buyer would then have all remedies available at law, including the right to recover consequential damages in addition to damages for breach of warranty.

Vendors would rather lose their exclusive remedy than be subject to liability for consequential damages. To mitigate this problem, certain vendors have drafted alternative exclusive remedies. Both must fail before the consequential damage exclusion is questioned. The most common alternative is a remedy granting monetary damages up to the amount of the purchase price. Thus, the limited warranty section would read in effect that the exclusive remedy of the buyer is repair or replacement, but if for any reason that remedy should fail of its essential purpose, the exclusive remedy of the buyer shall then be actual damages up to amounts paid.

Some vendors go further and purport to define when the limited warranty will "fail of its essential purpose." In the IBM standard form contract, for example, IBM states that the exclusive remedies for performance or nonperformance of its products will be as provided in its warranty section, and if "after repeated efforts" IBM is unable to make the product operate as warranted, then the exclusive remedy shall be actual damages up to amounts paid.

D. Disclaimer of Implied Warranties

The U.C.C. provides several implied warranties, including an implied warranty that the product will be "merchantable" and, in certain circumstances, an additional implied warranty that the products will be fit for the particular purposes for which they were designed. The U.C.C. also allows a vendor to disclaim these and other implied warranties.

A disclaimer is easy to write, but must be drafted with some care. The warranty must be conspicuous; this usually requires either boldface typing, capital letters, or a different color, such as red. A disclaimer should also mention that word "merchantability." The magic language often used is that "THERE ARE NO OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

Even if a vendor decides to be less strict in allocating risks to a buyer, the vendor should reconsider any decision not to include an appropriate disclaimer of implied warranties. The vendor can provide specific express warranties that may go far beyond the implied warranties of merchantability and fitness for a particular purpose. In that case, however, the seller would have some certainty as to

how its warranties will be interpreted and how it will have to perform in order to meet them. Such certainty is not present with implied warranties.

E. Integration

An important question in interpreting and construing any contract is whether the written document embodies the complete understandings between the parties. In nearly all computer procurement situations, representations are made in oral statements by the vendor's sales representatives and in letters or other written proposals transmitted by the seller to the buyer. In these oral statements and in these written documents, a court could find express warranties binding upon the seller. Therefore, it is almost always in the seller's interest to include an "integration clause" in the contract which states that the written contract is the entire contract between the two parties and that there are no other prior or contemporaneous oral or written statements, memoranda, understandings, obligations, and so forth between the parties. An integration clause is like a twin sister to a disclaimer of warranties. The disclaimer negates implied warranties; the integration clause negates express warranties.

Integration is a question of fact, and even with an integration clause, a contract can be construed not to have been the complete understanding between the parties. This is particularly so when certain aspects of the procurement contract, such as software programming, software licenses, training, and support obligations, are not even mentioned in the contract. Sometimes vendor attorneys are so concerned about not placing obligations on their clients that they fail to mention certain important aspects of a procurement in the contract. It is safer to mention normal support and software obligations in a computer procurement or to refer to other documents which are carefully drafted and protect the vendor in such arrangements.

F. Limited Time to Sue

The U.C.C. provides that all actions for breach must be commenced within four years of the date the cause of action for the breach first arose. The U.C.C. also allows the parties to contract to shorten this limitations period to as little as one year. Normally, this makes sense for a vendor because the only performance it desires from a customer is payment. Also, customers tend to try to make the procured product work rather than terminate the procurement and litigate, and they may attempt to make the products work

well beyond the one-year limitation period. They would then be foreclosed from suing for contract breach.

G. Limited Acceptance Testing

Under the U.C.C. the parties can agree to a standard of acceptance. Vendors often provide that acceptance occurs upon installation to the vendor's satisfaction (and when the vendor commences its maintenance service). Installation of hardware and commencing maintenance does not mean that the hardware works satisfactorily and, as a matter of timing, may have little relation to the completion of software and the adequate performance of the computer system. This standard of acceptance-upon-installation gives vendors control over payment (which usually is requested within thirty days of acceptance), and limits the remedies of the buyer, for most of a buyer's remedies at law relate to nonacceptance of the goods bought.

IV. USER CONTRACTS

A. The Traditional Approach

The trend to negotiate vendor form contracts appears to have begun with several well-managed or large companies which normally negotiate all procurements from a user's perspective. For some reason, unlike the rest of the country, these companies did not treat computer contracting as different from the normal procurements of a company. Their aggressive approach resulted in negotiation of previously intractable contract provisions. Unfortunately, this presumption of similarity creates its own problems. Many of these users approach computer contracting in the same manner as they do any other procurement with no recognition of the peculiarities of data processing. Computers are different, and the differences require more diligence than usual by the user, not more reliance upon the vendor's manner of contracting.

Instead of concentrating on problems endemic to the computer industry, these users concentrated on the allocation of risk provisions in the vendor form contract. Some success is possible in this area, particularly with respect to express warranties and acceptance testing. It is much more difficult to move a vendor away from disclaimers of implied warranties or exclusions of liability for consequential damages. These disclaimers and exclusions are reflected in the price being quoted, and an astute vendor would be justified in increasing the price to reflect greater risk or offering price concessions in lieu of the desired user amendments.

The problem with this traditional approach is that it only brings

the user marginally closer to achieving a satisfactory procurement. By slightly increasing the vendor's risk, the user may have increased its bargaining pressure in the event a dispute arises. This increase is negligible, however. Although the user lessens its monetary commitment by achieving price concessions from the vendor, it may also have created a situation where the vendor will seek to cut its costs in other respects and will provide less than satisfactory performance.

It is important not to overlook this stage in reviewing vendor contracts. Liability for consequential damages may be excluded in an inappropriate circumstance, such as when the user is sued by a third party because the vendor violated the third party's proprietary rights. Liability for other damages may be limited to a dollar amount which is extremely low. In addition, there are many contract issues outside of allocation of risk which may be important, such as ownership of software, assignment, installation and training services, term, ability to upgrade the system, and selection of vendor programming or training personnel.

B. The Systems Approach

A substantial improvement in user contracting occurs if the procurement is defined in terms of functional performance. A vendor form contract, objectively construed, promises very little other than the delivery of a collection of machinery and electronic components which will operate in an electrically satisfactory fashion according to limited "published specifications." Negotiating to change the allocation of risk provisions in such a form contract does not change this fundamental characteristic of the contract. Approaching the contract as a document which should explain the functional results the user wants, however, will dramatically change the nature of the final agreement.

This approach consists of applying systems' analysis to all aspects of the procurement and including the conclusions from the analysis in the contract. The contract will specify such matters as the performance of the system on a functional level, delivery and installation obligations, acceptance testing and what constitutes satisfactory performance, and warranties to cover aspects of the functional specifications which may not be fully realized, such as that the system design is a good one and that the system will meet the needs of the user.

1. Functional Specifications

The most fundamental element of the systems approach is to

create functional specifications for the computer system or software desired. Functional specifications clarify the responsibilities of the developer and provide the basis for acceptance testing. The specifications should be comprehensive and cover the transactions the software is intended to implement, the performance criteria of the particular units of equipment and the network characteristics which are created by the interface of the software with the hardware and separate processing units with each other.

The functional specifications in a system procurement transformation must be more than software specifications. Software which in theory may be able to perform under a variety of loads and volumes without limitation may in practice be subject to physical limits, such as disc storage volume, cache and central memory capacity, and data transfer speeds (between the processing unit and external storage and between the processing unit and other work stations in a network). One of the most common problems arising in computer system procurement is that the original system is designed for current volumes and loads, and it later proves wholly inadequate for increased volumes and loads during the full useful life of the system. It is fundamental to the concept of the systems approach that the system design and its incorporation into functional specifications be adequate at the expected capacity of the system during its complete useful life. The specifications should describe the functioning of the system in enough detail to enable uninitiated developers to create the software from the specifications alone. The specifications can then be used to solicit proposals and bids from computer vendors and software development companies.

Specifications are usually developed in three stages: in the initial internal review by the user of its needs, in the development and publication of a "request for proposals" which incorporates these internal standards and requests bids and proposals from vendors on how to implement the user's needs, and in discussions between the user's technical staff and the vendor regarding the capabilities of the vendor's system and how it can be designed to meet the user's specific needs. These specifications also can be created by a separate contract with an outside consulting firm.

Creating comprehensive specifications can have many beneficial effects to a user that are not directly related to the specific terms of any procurement contract. It forces the user to integrate this particular procurement with its other data processing and business activities. It educates the user as to what it can realistically expect from the computer system. It also develops a comfortable, although structured, relationship between the parties very early in the procurement process.

While functional specifications are essential to the buyer, they are also useful to the vendor. Project specifications clarify the vendor's duties. The contract may also require that any subsequent changes to the specifications be undertaken only under certain conditions including, for example, an adjustment in time and payment terms.

2. Acceptance Testing

The second essential element in the systems approach is to create an acceptance test or tests for the system and its components. If the acceptance test is properly drafted, once the system passes it the user has received the performance it set out to procure. Great care must be taken to draft a comprehensive test which covers all aspects of performance, including delivery dates, documentation, and training.

An acceptance test of a computer system can be difficult to create. A computer system can seldom be simply plugged in and turned on. While many performance problems with a system are normal and can be cured in the normal course of events by installation and maintenance service, other problems, particularly with custom-designed software or new models of equipment, can be intractable.

A test requiring the "satisfaction" of the user is theoretically the most powerful, for the user can terminate the agreement for any reason as long as it exercises good faith. Termination, however, as will be explained, is not usually a useful remedy to a user, for the user has already invested substantial time and effort in a particular vendor's system to the exclusion of alternate systems and may have no time to switch to the alternatives. The user's true alternative is to cure any defects in the system, but ironically, it may have lost leverage by the vagueness of the "satisfaction" provision. The vendor may take the strong position that it has provided all that it was obligated to provide and that the user is asking for new transactions or new features to be added to the system.

Instead, the user should design an acceptance test which incorporates the functional specifications previously created. The user should negotiate for the strongest standard of compliance to which the vendor will agree, but usually substantial compliance, excluding problems which can be expected to be corrected by normal installation, maintenance, or software support services, is sufficient. The use should also understand that acceptance testing is a dynamic process and that a system which works correctly one day may fail the next. It is therefore usually necessary to test the complete sys-

tem for a period at least as long as the normal cycle time for processing the transactions contemplated to be executed on the system. It is also necessary to have sufficient time in which to allow for a "burn-in" period of the equipment.

Typically, acceptance testing is broken into two parts: testing software modules with simulated data and testing the overall system with "live" data. Simulated data should be created with some care and with some independence from the developer. It is surprisingly easy to rig software tests, even unconsciously, because software developers can adopt a narrow frame of mind as to potential problems and the type of data for which the program is designed. A live test is essential since the load of work on a computer system can affect its performance, and software which works quickly and easily in clean, unloaded environments may bog down and work extremely poorly when operating in a live environment.

The user should be careful when relying upon preinstallation demonstrations of the software. The software may be demonstrated to correctly execute all transactions, but in combination with a computer system which is functioning with heavy processing loads, the software may not be as efficient as expected by the user. Such efficiency can usually only be determined by operation of the complete system.

Software vendors will often urge that acceptance of software is a one-shot matter. Thus they argue: "If it works now, it will always work." The argument is correct only if all possible loads on the system and combinations of processing are capable of being tested in one-shot situation. In reality, software in a system procurement is best tested "live" under the loads and with the processing combinations used by the system customer on any ongoing basis.

It is easier to test and be confident with the performance of hardware than it is to test software or a computer system. Hardware tends to have three types of problems during acceptance testing: (1) "dead on arrival," meaning that the hardware never works at all, usually because its electronic components have not been fully tested prior to shipment and some components are defective; (2) "infant mortality," meaning that during the first few days of use a high incidence of failure of individual components will occur to the point where it is better to replace the component than to attempt to repair it (a situation often caused because the individual components were not "burned in" for a sufficiently long period to determine their reliability); and (3) the normal incidents of failure which are to be expected during the lifetime of the equipment.[4] Accord-

4. Electronic components once "burned in" are highly reliable, but because of

ingly, a hardware acceptance test should anticipate immediate failures, a high incidence of failures during the first few days or weeks of testing, and a not-more-than-normal failure rate thereafter. Quality of hardware should not be of great concern once infant mortality is taken care of. If proper maintenance controls[5] are included in the procurement agreement, the user can control and will be protected from on-going hardware failure.

3. Payment

The user should condition any payments upon satisfactory completion of the test. In the vendor form contract, payment for a vendor's system may be due independent of any final acceptance procedures. Payment may be specified as due within a time period of completion of the software, or more typically, within a time period of completion of the installation of the hardware. Instead, payment should relate to the user's acceptance test.

Often the best approach for payments from the point of view of the user is to allow for a partial payment upon satisfactory demonstration of the completion of the software, a substantial payment upon satisfactory completion of the complete system acceptance test, and a final payment of perhaps ten percent of the total price upon completion of all remaining obligations (such as training, providing documentation and correcting all the minor problems which may have arisen during the acceptance test). Surprisingly, the partial payment upon satisfactory completion of the software, even if it is made well before the final acceptance test, is often to the advantage of the user as well as the vendor, because it not only aids the relationship between the vendor and the user and makes the concept of contracting for performance easier to sell to the vendor, but it also tends to encourage quicker completion of the software by the vendor. Early completion lowers the risk that the vendor will be unable to have the complete system prepared by the commencement of the final acceptance test. Usually the earlier the user can have completed both the functional specifications and the software incorporating the specifications, the less likely it is that the vendor will be unable to provide an operating system, because any difficulties or

---

the high number of components in any individual computer, breakdown is commonplace. Breakdown can also occur because of poor design, poor selection of components (which either do not match well in terms of electrical characteristics or are of low quality) or poor electrical and environmental conditions. The problems of low quality components or bad design can often be ascertained by careful investigation prior to the procurement.

5. *See infra* Section V.E.

overselling by the vendor will be determined early in the procurement process.

C. Project Control

In some situations, negotiating a "tight" contract, a contract which covers all aspects of performance without loose ends may not be sufficient to protect a user. Once the user signs the contract and commits itself to a particular vendor, the leverage that the vendor holds over the user increases. This increase in leverage is due to the commitment of the user to the system of the vendor and the gradual loss of options as the user's internal start date approaches. This is particularly a problem in the procurement of computer systems involving custom-designed software elements or in software developments contracts. Software creation is often thought of as a process similar to the undisciplined creative efforts of artists or scientists. This attitude, prevalent among software developers, leads to poor management control of software development. Software has many more of the characteristics of an engineering project than of art or science and should be managed and controlled as such. The best-designed acceptance tests and other contract elements for defining and measuring performance will be ineffective if the software work is mismanaged, and, as the time for performance approaches, it becomes clear that the vendor will not provide the results desired. The only alternatives facing the user are accepting further delay, less tha adequate performance or sometimes an increase in cost.

The user's remedy is termination and litigation, but this option is hardly a bargain. Modern litigation, like nuclear war, is a frightening prospect. Its potential victims should seek more peaceful alternatives. Whether due to liberal discovery, an adversarial and litigous culture, clogged courts, the modern law firm with its extensive resources and eager associates, or simply the complexity of modern transactions, litigation can be expensive, slow, petty, time-consuming, unfathomable, capricious, and just plain exasperating. Litigation has its role, but in many circumstances, other alternatives are more welcome.

One alternative to termination and litigation is *project control*. A data processing contract should be viewed first and foremost as a mechanism for project control: a mechanism to ensure that the client receives precisely what it wishes to acquire in a timely manner and to ensure that the supplier understands the precise scope and timing of performance. The contract can contain clauses which control the data processing project being undertaken.

Project control can be an extremely effective device to avoid disputes. Many of the horror stories in the data processing industry can be traced to inadequate controls. For example, in one recent software development contract, the customer paid over $2,000,000 and anticipated paying of an additional $1,500,000 with no light visible at the end of the tunnel. An independent software consulting firm investigated the situation and found that it could have offered a fixed-price bid for as little as $350,000, completed the project many months before, and still have made a normal profit. While the investigation did not impune the integrity of the consulting firm actually hired, it noted that controls on product development in the contract were so weak that the project simply ran away with itself. In data processing contracts, particularly in software development and in computer system procurement and maintenance situations, this bottomless money-pit phenomenon can be avoided by correct project control.

V. PROJECT CONTROL CONCEPTS

Project control is achieved by several fundamental concepts. In many respects these concepts parallel the types of internal controls in any quality development project. The requirements to be performed should be clearly specified by functional specifications. First, the time and the manner of performance should be broken down into discrete components through a project timetable and other elements relating to it, in order to give early warning of slippage, to provide discrete components so that a failed project is not a complete waste, and to limit the possible excuses or grounds for excuses which could delay or raise the price of a project. Second, the contract should provide complete and adequate acceptance testing of each component and of the complete system. Third, it should provide on-going standards of performance for maintenance and support, and finally, it should provide contract remedies short of termination and litigation.

A. Performance Standards

A project control contract builds on the concepts already discussed. Allocation of risk provision should be reviewed for overviewing. Special requirements, such as assignment rights, should be specified. Most importantly, a project control contract is based upon the same concepts underlying a contract created by the systems approach. The project control contract should include detailed functional specifications and comprehensive acceptance test-

ing. Project control clauses are ways of implementing such a contract.

B. Project Timetable

The fundamental project control concept is to create a project timetable which describes major tasks to be completed by the vendor and the dates by which they are to be completed. The timetable should be part of the agreement, and the vendor's obligation to meet the various dates, or at least the most critical dates, should be made definite. The agreement should provide that if any task is not completed on time, the software developer would be in material breach of the agreement. The timetable should be airtight, creating fixed obligations for each major task. Wherever possible, if the vendor is to create the functional specifications, the user should insist upon an early completion date for development of the functional specifications. Then, if the specifications are unsatisfactory, the user should have the option to terminate the contract and return to the marketplace to discuss system procurement with other vendors. In an industry where firm deliver dates are uncommon, the user may have to commence the procurement process will in advance of its expected date of live operation in order to be able to find a vendor who could be capable of making a commitment to a firm delivery date. Invariable the vendor places most resistance upon the manner in which the various critical dates are made "firm." Most simply this can be done by making time "of the essence." This can also be accomplished by specifying very clear contract remedies[6] for failing to meet critical firm dates.

There are many techniques for creating a "tight" timetable. The type of contract determines which techniques are appropriate. For example, the key technique in a software development contract is to create a fixed time and price obligation for each major program or module. This may require a multitude of firm completion dates and a complicated timetable. On the other hand, in many computer system procurements, particularly when most of the software has already been developed, tested, and successfully marketed, the project timetable need not have as many individual major development tasks as in a software development contract. A suitable timetable may need as few as four principal tasks: (1) customization of forms and other "off-line" input messages and output documents; (2) completion of network or other "on-line" remote access protocols and features; (3) delivery and installation of equipment, and (4) testing of the complete system. Negotiation of a system procure-

6. See *infra* Section V.C.

ment agreement is easier if the number of individual deliverables and development task dates are minimized. The desire to achieve a satisfactory agreement, however, should not lead a customer to unduly truncate the project timetable.

1. Fixed Time and Price Contract

Software consultants generally prefer to bid on an open time and price basis. From a project control standpoint, this approach makes sense only if the customer is able to assume substantial responsibility for actual control and supervision of the project and if the developer is only assigned specific, isolated tasks. In such a circumstance, the customer can place its own controls on the software developer. Typically, however, the software developer will be working on its own. This type of wide open uncontrolled approach creates the bottomless pit of many software development projects. It is in the supplier's economic interest to low-ball its quotes since it will not be bound to them. Despite the customer's confidence in the supplier's low-ball estimate, the customer is signing a blank check.

The major argument of developers against fixed time and price arrangements is that it is very difficult to estimate what the project should entail. The solution to this problem is first to develop detailed functional specifications, preferably through an agreement, perhaps with a different development company or consultant, which contains many of the project control aspects of the software development contract, including a fixed time and price.

2. Rolling Estoppel

A second major argument against a fixed time and price contract is that unexpected problems may arise which justify delays or price increases. A useful negotiation solution to this problem is to provide a mechanism for changes in timing and payment with insulation from "surprises." This insulation consists of requiring that notice be given when problems arise and applying "rolling estoppel" to cut off stale claims.

Rolling estoppel is effected by requiring periodic reporting, usually bi-weekly or monthly, of the status of the project and requiring the developer to list in the report any problems which could cause a delay or an increase in price. Project reports should also include proposed remedies for any problems that may cause a price increase or delay in order to prevent the reporting process from being simply a mechanism for the developer to notice hypothetical problems and thereby to preserve its rights at an unspecified time in the future. A rolling estoppel is created if the developer is pre-

cluded or estopped by the agreement from raising problems that occurred during a previous reporting period but which were not discussed in the appropriate project report.

Rolling estoppel promotes diligence on the part of the developer. As a consequence, as more of the project is completed, there is successively less possibility of a surprise delay or price increase. Indeed, rolling estoppel can even turn a time and materials contract, over time, into a fixed time and price contract.

3. Rolling Payment

It is advisable to state a separate price quote for each task, and in each task the developer should be required to stay within the bounds of its quote. A fixed price contract does not mean the maximum price must be paid, but that the developer will not be paid more than the upper limit, absent justification. The developer should insist that it be paid upon completion of each task, but the customer should require a partial holdback of each payment until final acceptance. New problems may arise in the testing of a complete system even though the tests of search individual component revealed no problems. The developer's invoices should break down the total cost into the various development tasks, with an itemized statement of time, materials, and other pertinent information.

4. Deliverables

If possible, each task in the project timetable should be designed to result in an independent "deliverable." A deliverable may be a program or program module, documentation, training, service, or other discrete component of the finished project. By breaking the project into separate deliverables, the user gains a degree of independence from the developer, since as each deliverable is completed, less remains to be done by a new developer if the original transactions turns out. Termination rights are illusory if the customer becomes "locked in" to a particular developer, because the completion of the project would require the delay and expense of duplicating work done by the original developer. With modular deliverables, if the original developer is terminated, it should not be as onerous for another developer to step in and continue with the project.

A customer should be skeptical of the supplier's view of the correct order of deliverables. Leaving the most critical and difficult problems until the end can lead to an uncomfortable trap—the procurer is too committed to go to other alternate sources of supply,

and due to problems which arise at the last minute, will not achieve a timely and satisfactory installation.

5. Documentation

It is important to couple program or program module deliverables with an obligation for the developer to create documentation along with the programming code. The documentation will aid a new developer in understanding and completing already created programming.

6. Rolling Acceptance

Each important task should be subject to acceptance. No important development task should be deemed complete until all the deliverables involved have been delivered and accepted by the customer.

C. Contract Remedies

The second key concept in a project control contract is that of contract remedies—remedies other than termination or litigation. Termination is rarely the best alternative in a situation where a user has committed substantial resources to the particular computer system and no longer has the time, energy, or desire to return to the marketplace to procure an alternate system. Indeed, many vendors will give termination rights at various points in the project timetable fairly easily, as they realize that it is most likely that the user will simply waive the firm date rather than attempt to procure the system elsewhere. A more subtle vendor will allow firm dates in return for a short period in which to exercise the right to terminate (typically thirty days), knowing full well that not only will the user be very unlikely to make a decision to terminate within that thirty-day period, but also that the user may be placed in the uncomfortable position of having waived its right to terminate while still facing the obligation to make payments without having a functioning system. A truly subtle vendor will, in addition, upon exercise of the termination rights, ask for waiver of any claims for breach.

The user should consider mechanisms short of termination and litigation which help ensure proper compliance with the firm dates and the project timetable. These may include a speedy arbitration procedure, liquidated damages, credits or other rights to offset against payment, or other contractual remedies. Contract remedies will be useful whenever the vendor must perform and are applicable during the life of the procurement (in a maintenance contract, for example) as well as during the development and delivery stage.

D. Arbitration

Contract remedies are never a final solution if the parties do not agree on an appropriate settlement of the problems they are confronting. It is helpful to develop a procedure by which the project can continue despite disputes over certain of its aspects. As a practical matter, it is often difficult to be assured of timely and adequate performance when a developing series of disputes doom the project as a whole. The antagonistic nature of litigation renders it almost totally ineffective in controlling an ongoing project. A skillfully created arbitration or other quick dispute resolution process can often be extremely helpful in controlling a project. Disputes over acceptance, meeting documentation standards, or whether a project problem should give rise to an extension of time or an increase in price, all can be resolved by properly designed arbitration.

E. Maintenance Terms

An important area in the procurement process which should not be overlooked is the proper structuring of a maintenance arrangement. Indeed, in the long run, adequate maintenance may be more important than meeting firm delivery dates or having a "state of the art" system. If properly designed and installed, a computer system will be used for a much longer time than was required to develop, install and test it. Thus, proper attention to project controls during the period of use and maintenance of the system is essential.

The key element in any user-oriented maintenance agreement is a commitment by the supplier of maintenance services to standards of maintenance timeliness. The typical standards of such performance are response times, combinations of response and repair times, or guarantees of overall up-time for the system. Defining the standards, however, may not be sufficient for a user's purposes. The standard vendor form agreements typically include *force majeure* or "acts of God" clauses which excuse the vendor's failure to perform and which can be a mechanism for avoiding the standards of performance. The user or its attorney should be careful to consider the language of such clauses and amend them if necessary. In addition, if the user's remedies are limited to termination and litigation, the standards would have no practical effect despite any technical breach because the cost of litigation will almost always be much higher than the value being lost by nonconformity with the standards. Accordingly, the user should consider contract remedy mechanisms short of litigation to help ensure proper compliance with the standards. These may include a speedy arbitration procedure, liquidated damages, credits or other rights of offset against the mainte-

nance fees or lease payments, or other contractual remedies. A user should also consider self-help. With the present trend toward decreasing hardware costs and increasing labor costs, it may be less expensive for a user to purchase or have the right to purchase spare units of equipment and to train its personnel to replace such units in the event of any equipment malfunction rather than pay the steadily increasing maintenance charges for maintenance service.

1. Up-Time

The strongest control over the life of a system is to have a maintenance organization (which may not be the vendor) agree to a specified up-time for the system. A reasonable up-time percentage can be determined based on input from several sources, including promises made in the vendor's literature or during negotiation and the user's analysis of its own requirements. The user may not need more than a 95% up-time or the user may need a 99% or 99.5% up-time. From a negotiating standpoint, it is best first to determine the basis upon which the up-time percentage will be determined. If the manufacturer's literature promises an adequate percentage, it is relatively easy to negotiate for that percentage. It is often wiser, however, to follow the user's own analysis. If the vendor cannot promise to supply the necessary up-time assurances, the customer should look elsewhere.

Overall up-time for a system of many components is a very different calculation than up-time of individual components. In a negotiating environment, the user should first determine and have the vendor agree to an objective standard for determining the proper selection of up-time percentages and components to be considered. Often it is sufficient to have up-time percentages on critical components and an overall up-time on the complete system. Minor down-time of unnecessary components that do not affect actual processing can be excluded. Up-time controls can also be achieved by down-time limits. The parties may prefer to agree that the system will not suffer more than a specified maximum down-time percentage or number of down-time hours over some period of time.

2. Response Time

One of the most common project controls on maintenance contracts is a specified response time (the time following notification of a maintenance problem until arrival of maintenance personnel). A response time specification is not as attractive as an up-time percentage in most circumstances. An up-time percentage implicitly covers both response time and repair time, since down-time in-

cludes the time between notification of a problem and arrival of service personnel as well as the time for the service personnel to correct the problem at hand. In many circumstances, however, a response time clause makes sense, particularly if the user is in a critical environment in which length of an incident of down-time is as important as overall down-time. For example, in a retail environment, such as a bank, down-time for teller station terminals can create enormous customer dissatisfaction. Overall up-time for the system, including nighttime when the bank retail operation is closed, may not be as great a concern since the bank can catch up on processing overnight despite occasional down-time.

3. Credits

Use of effective project control requires that up-time and response time not be promised in isolation. Litigation over failing to meet an up-time percentage or the promised response time is rarely justified on a cost/benefit basis. Instead, a specific up-time percentage or response time should be coupled with contractual remedies such as credits or liquidated damages.

A user should be careful when structuring credit remedies. For example, credits are often applied against maintenance payments. Whether this makes sense in a particular circumstance depends in part upon the amount paid for maintenance. Maintenance has become an increasingly higher percentage of procurement cost as hardware costs have decreased and labor costs have increased. Some vendors however, rather than providing on-call maintenance personnel, offer non-labor-intensive—and less expensive—maintenance plans, such as factor diagnostics through remote communication and "depot" maintenance whereby the user itself exchanges components in the system and sends them to the factor for refurbishment or replacement.

If the system is procured on lease, the use of credits can be a relatively easy matter as the credits can be applied directly against lease payments. From a negotiating posture, the user has an excellent argument that a credit should be applied against all of a lease payment and not merely against the maintenance part of the lease payment. The user was promised a system with particular up-time performance (98%), and any degradation from that level of performance constitutes a breach of warranty with respect to the system as a whole. The credit does not compensate for bad maintenance alone, but also for poor equipment and breach of warranty.

Often a vendor will strongly argue that credits should be based upon averaged performance and will even suggest the performance

be averaged over a month or a quarterly period, or over a large set of installations. Averaging can completely defeat the usefulness of credits. Poor performance at some sites or at certain critical times will be averaged against normal or good performance elsewhere—and the user has little recourse. The user should be careful to evaluate such suggestions about the structuring of up-time and response time credits in the context of what makes good business sense and what will enable the user to be confident of achieving the proper project control.

4. Bonuses

Vendors faced with a demand for credits may well ask for corresponding bonus credits. It can make sense for the user to agree to some form of a bonus system, both to help negotiate the proper credits for down-time and to provide further incentive for the vendor to provide quick maintenance. Credits/bonuses often are structured in a linear and parallel manner because the supplier wants "mutuality" and "simplicity." Neither rationale may make business sense. The disruption to a user's data processing center will often increase on a nonlinear basis for decreased up-time, or for overly delayed response time. The user is paying a certain amount for a 98% system, and while the damages and disruption measured by a less-than-98% system are covered by the credits, the user does no gain a corresponding benefit by having a 99% or 100% system. For example, a 99% up-time for several weeks or months may not enable the user to catch-up for the enormous damage and disruption which occurred when at a year-end time period the up-time was only 90%.

VI. NEGOTIATION

An underlying problem in the systems approach or in contracting for project control is negotiation. While negotiation is not a game, it is competitive and must be played. The vendor wants the user to sign its form contract. The vendor has substantial negotiating experience. The user's data processing personnel may be friends of vendor's personnel and will no always be helpful in a negotiation. Instead, they may provide information to the vendor and may at times attempt to undercut the negotiators either by contracting levels of management over the negotiators or by taking other bureaucratical action in order to ensure that their relationship with their "friends" is not ruptured.

It is important to understand the nature of negotiation and to prepare for it. It is essential to be wary of the data processing people and the actual users in the organization. Also, the negotiators

should not rely upon support from upper level management which may not have any interest in negotiation because of its perception of date processing as wholly arcane and separate from the rest of the business.

Project control negotiations often are easier than one might expect. The user is not adjusting the risk allocation provisions of the vendor form contract, except indirectly. More importantly, throughout the process, the vendor will often only be asked to put in writing what it will orally agree to provide. The elements of project control can often be characterized as aiding the vendor as much as the user, because they clarify the performance of both parties. Clarifying the vendor's performance is often in its own interest. If the vendor accomplishes what the clarification states, it will have performed and will not be subject to justified dissatisfaction by the user.

The most difficult clauses to negotiate in the project control situation are the alternatives to the termination remedy, such as liquidated damages and other contract remedies. Negotiating contract remedies can be difficult if not handled properly. Negotiating a liquidated damages provision, for example, involves an inherent dilemma. If the dollar amount of liquidated damages is too large, the specified payment may be considered a penalty and be unenforceable at law. If the specified payment is too small, it may create little incentive for the developer to perform on time, and if the contract makes it the exclusive remedy, the customer may be left without an adequate remedy in the vent of litigation. As an element of project control, the liquidated damages clause should be drafted so as to be both a carrot and a stick. The amount of damages should not be unreasonably high, but it should be high enough to hurt.

In many contexts a creative negotiating strategy will require an attorney to present contract remedies other than a straightforward liquidated damage clause. Usually, however, the issue of credits will have to be met. It is helpful to use the following approach.

First, convince the vendor that it will have to answer your concerns. This may require control over your company's data processing personnel who often convey to the supplier the attitude that "the decision will be made down here" or "the negotiating team is not serious" or "I'll talk to their boss; don't worry." This gives the vendor false hopes and can lead to deadlocks. Second, reach early agreement that all requests will be judged by a "business sense" standard and that one overall concept which makes "business sense" is project control. Often in the process of convincing the vendor of your seriousness, requesting only items which make "business sense" and repeatedly explaining that rationale, will raise the vendor's consciousness to accept the "business sense" standard.

Third, negotiate concepts before clauses. It is easier to agree that an up-time percentage is an element in project control and makes "business sense" than to agree to a ten percent credit per percent of down-time. Finally, be flexible and creative. Relate your positions to what makes sense to control the project, and listen to the other side. Remain firm on project control concepts, but work with the needs and peculiar problems of the vendor to achieve them.