INTERFACING WITH THE SOFTWARE INDUSTRY - PART II -

SOME CONTRACTUAL AND LEGAL QUESTIONS AFFECTING CONTRACTED SOFTWARE

by

KARL REED

National Chairman,
Software Industry Committee,
Australian Computer Society

ABSTRACT

This paper discusses the role of standards and 'accepted good practice' in software contracts. Several classes of standards are discussed, those relating to performance and quality of work being given special attention. Comparison is made with the construction industry.

The contractual relationship is discussed with regard to the responsibility for modifications and disputes.

Client and contractor are urged to approach the situation fairly and with regard to each others problem.

Keywords and phrases: contract, software, arbitration, standards, system description methodology, design modification.

C.R. Categories: 2.2, 4.9.

Authors Address:
c/o Computer Centre, Monash University,
Wellington Rd., Clayton, Victoria 3168, Australia.

INTERFACING WITH THE SOFTWARE INDUSTRY - PART II

SOME CONTRACTUAL AND LEGAL QUESTIONS AFFECTING CONTRACTED SOFTWARE

1. INTRODUCTION

There are a number of problems relating to software contracts which receive scant attention in the literature. These relate particularly to arbitration as a means of resolving disputes and to the existence and legal significance of standards. The legal implications of claims of competence made by a supplier is also of significance. This paper deals with these questions, relating them to the engineering and construction industries.

2. ACCEPTED STANDARDS OF PERFORMANCE

2.1 Introduction

There are standards implied in most contractual situations involving the provision of goods or services. These enable both parties to avoid unnecessarily detailed specifications and contracts, and provide a basis for assessment in the advent of a dispute. Programming, system design and implementation do not have accepted standards and the existence or nature of any implied good practice is open to question. Clients of the industry will, however, make assumptions about the manner in which unspecified activity is carried out (see Symonds 1977). This paper identifies several major areas of standards and compares the software industry with its engineering and construction counterparts.

2.2 Professional versus Contractual Liability

Just exactly what should appear in a specification (and related contract) concerning actual details of implementation?

References RAIA (1970) and ACS (-) are interesting documents. The first is the standard specification for a brick-veneer house as used in Victoria. The second is the Code of Professional Conduct of the Australian Computer Society.

A comparison of these two documents shows that a builder's client (at least in Victoria) has a complete specification of what shall go into his house and exactly how it should be 'fixed'; while a software house's client has, apart from his own specification and contract, only the good intentions of the ACS Code of Good Conduct, ACS (-).

Rod Symonds (Symonds, 1977), when addressing a Software Industry Committee Seminar, said that a detailed specification did not exempt a supplier from 'professional responsibility' with respect to basic levels of back-up, vulnerability and performance. The client did not need to specify these, he argued.

The sad truth is that there is no generally accepted standards of performance covering our profession, so software industry users are less fortunate than those dealing with the construction industry.

The house-builder has the 'General Specifications' complete with its clear statement that *'Under no circumstances shall the general specifications be crossed out, added to, removed from, written on or altered in any way'* (RAIA, 1970, para. A.2, page 17). This document sets out detailed standards of workmanship. There is, in addition, the complete set of building regulations and relevant Australian standards to which the General Specification refers.

All of this should be well known to the reader, and its implications are clear. What is needed is some means of establishing similar 'standards of work' in our industry, a task beyond this paper. Professional liability and responsibility can then be related to such

standards. Failure to comply with these standards then involves failure to comply with the contract.

Three such areas are identified.

3. POTENTIAL STANDARDS DEVELOPMENT FOR SOFTWARE DEVELOPMENT

3.1 Standards of Workmanship

Standards of workmanship relate to things such as the clarity of system structure, 'visual' quality of written programs, documentation and testing.

How should a system actually be described so that its nature and function can be ascertained? There are many possible techniques, all of varying usefulness. There are not, however, any Australian Standards recommending which techniques are appropriate for given types of systems and levels of detail.

Flow charts and block diagrams are widely used as are decision tables. State transition flow charts can be useful in the description of real time systems, compilers and suites of programs. (Hemdahl, 1973). Tabular and non-tabular state transition diagrams are also useful as a means of checking that all eventualities are accounted for. (Heistand (1964), Oliver and Jones (1976), and Oliver (1977)).

The 'visual' quality of programs is a subject for continuing debate. There is little point in analysing the arguments in favour of 'pure' structured code, piece-wise refinement, modular programs, level structured programs, hiearchically structured programs, data driven programs and just programs. The literature contains ample material (Yohe (1974), Wirth (1974), Knuth (1974), Wirth (1971), Kernighan and Plauger (1974), Yourdon (1975), Parnas (1972), Peters and Tripp (1977), Maynard (1972) and Schneiderman (1976)). There is, however, little guidance as to what constitutes 'well written code', and little general agreement as was shown by a series of letters in the Communications of the ACM during the mid 1970's (Smolia (1974), Gries (1974), Reed (1975) and Metz (1975)).

Programming documentation is much better served. The remarks in the Code of Professional conduct (ACS (-) Appendix A1) make good sense; there is an Australian Standard on flow chart symbols (ASA (1969)) although some writers (Schneiderman et al (1976), Pollock and etc. (1977)) debate their value vigorously. Wayne (1973) contains an introduction to flow charting technique. Several modified forms of flow chart (cf. Lindsey (1977)) have also been described. There is clearly considerable value in standards which could be applied to these areas.

The importance of system and program testing is dealt with in Reed, (1978). However, it should be noted that it too is an appropriate subject for a standard.

3.2 Performance Related Implementation Detail

Many engineering standards specify what could be described as performance related implementation detail. RAIA (1970) contains large volumes of such material. The computing equivalent is the relationship between system size, data volume and system response, to the algorithms used in the implementation assuming fixed hardware and, therefore a given operating system.

The choice of algorithms must be influenced by the operating system and the hardware as the basic providers of performance. Disc handling software, schedulers, instruction speeds, memory mapping and terminal control sub-systems produce quantitively different limits on system size and performance depending upon the algorithms chosen by the implementors.

File handling is an obvious example. Some simple operating systems used a chained block allocation technique which will limit the performance of the 'direct' access application program. Operating system modifications may be needed if specified performance is to be achieved.

Who is responsible for specifying this aspect of the design, client or supplier?

It is possible, in some equivalent engineering situations to quote a standard and know that the result will be satisfactory. Practical experience in the software industry suggests that this point is not clearly understood or controlled and may be the cause of hardware upgrades during projects. The situations which are potentially most dangerous are those involving an intermediate software layer such as a data base management system. The precise performance of such a system, the storage and disc access requirements, may be difficult to ascertain and may, in the end, cause the design to 'fail' in some sense or another. (Having to upgrade the hardware indicates someone's failure!)

3.3 Basic Reliability, 'Security' and Recoverability

System reliability is a subject for ongoing research, and rightly so. There ought however to be a level at which it is basic to system design and implementation, basic enough to be covered by a standard, and hence to be included in professional responsibility.

Perhaps, a system could be said to be secure if, when encountering data which it cannot handle, it aborts the current phase providing enough data to permit some recovery to occur. The ability to correct the files and re-process is of course needed. This approach may be totally inappropriate for an on-line, multi-user system handling some critical application, but only minimal standards are being considered.

Obvious procedures, such as module interface data checking (cf. Smith 1969) error retransmission requests on data communications lines, and basic error processing on peripherals are also suitable subjects for standards.

3.4 Professional Competence

Professional competence cannot be specified by a standard. Learned and professional societies such as the Institution of Engineers, the Australian Computer Society and the RAIA, control minimum levels of competence only. There is marginal protection in using a Chartered Engineer, Accountant or Corporate Member of the ACS. All that is guaranteed is that the person concerned has obtained suitable qualifications and has obtained a certain amount of experience without being obviously incompetent.

Examination of work experience records helps in determining competence, but these are not totally reliable.

The supplier clearly has a responsibility to assign competent staff to a contract. Ensuring that this is done is difficult. The SIC has addressed the question and has proposed a Register of Software Companies which would record staff competence levels.

3.5 The Legal Definition of Professional Responsibility

The preceding comments attempt to specify what should be included in concept of professional responsibility; that body of performance to be regarded as intrinsic to a field of professional service. It has been argued that the practical path to this situation is to create standards which have the same power as engineering standards. This also involves the construction of a body of legal precedent, and the production of legislation controlling standards of work, as has occurred in the building and construction industries.

Again, it should be noted that the engineering profession is more tightly constrained than our own in this matter. Tomson & Coplan (1967 pp. 263-278) set out the American legal interpretation on this matter, and it is clear that the engineer undertaking a project has definite legal duties, and is liable for failing to carry them out. "An architect or engineer implies by the nature of his employment that he will perform the required services with reasonable skill, ability and judgement and without negligence" (p. 265).

This then remains a topic for future research and concern within the software and computer industries.

3.6 The Development of EDP Standards

There is a significant international effort on computer standards, even though one sees little by way of results. Dubnow (1977) out lines the work of the International Standards Organization TC97 'Computers and Information Processing'.

System description techniques are a subject for consideration (SC7, Sweden), but there are no standards relating to performance under consideration. The situation is similar in the American National Standards Committee, X3, where again some questions of documentation are being examined.

Dubnow (1977) is recommended to those interested in pursuing the subject.

4. CONTRACTUAL OBLIGATIONS AND ARBITRATION

4.1 Introduction

The bulk of the foregoing material deals with the technical and moral aspects of design, and of project organization. The emphasis is of course on those problems which effect the interface between supplier and customer. Consideration has not been given to the form and content of a software service contract, nor to the related issue of arbitration.

Proposing a standard contract seems to this writer to be beyond the task of this paper. In any case, ACS SIC sub-committee is examining this question. There are, however, some points which should be discussed. The reader should examine the 'General Conditions of Contract for Civil Engineering Works' AS CA 24.1-1964 (ASA 1964) as a possible outline for a contract.

4.2 Contractual Aspects of Specifications, Modification and Design Variation

Specification

The specification should not be part of the contract document which only regulates the agreement between the parties. It should be an appendix to the contract document, and there should be a clear statement that the specification defines the work to be performed by the supplier. Such other documents as specify the design in detail should be treated likewise.

Modifications and Design Variation

The problems arising due to modifications to the specification, and design variations have already been discussed. The procedures for handling these should be written into the contract with particular reference to defining who meets the cost.

The more fundamental question of responsibilities for changes needs to be spelt out in the contract.

Specification changes may originate from either party, and for two reasons.

1. The specification may not work. That is, there may be no realistic way of obtaining the result specified. (Marquet (1977) p. 48). The fault lies with the specification producer who ought to be liable for cost variations.
2. Modifications may be proprosed which improve or alter system capability in some way. Acceptance of and hence responsibility for these lies with the client, who can be expected to carry the cost.

Where the responsibility for design modifications should lie, and who should meet the cost is not so clear cut.

Design modifications originating within the project team are not the concern of the client unless he is auditing the work technically. Design modifications proposed by the client, even when technically auditing the project are a different matter, particularly if there are penalty clauses in the contract.

The supplier may quite fairly argue that such modifications will distort his design in such a way that it will not work, or that unnecessary implementation delays. Such modifications may preclude the use of existing proprietry packages, and the supplier may justifiably argue for a cost increase.

These remarks are intended to apply to a situation where a contract has been signed, a cost and penalty structure agreed to and work commenced. Such modifications should only be necessary if the client either believes the system will not work or that it's performance will be unsatisfactory.

He should, in the first case, not have let the contract to the particular supplier, or, he should seek assistance from a technical arbitrator instead of forcing the issue.

It could be argued that he is, in the second case being unfair. Any known performance related problem should appear in the tender document, or be specified prior to concluding the deal. Design modifications should go to arbitration if agreement cannot be reached between the parties.

The foregoing view does not allow for the existence of accepted standards, written or implied, governing these matters. The client needs to be able to identify any failure to comply to these standards, a task which might be beyond many.

It would seem reasonable that the contract should allow for variations in completion dates and performance or penalty clauses if the client is allowed to reject or modify the contractors designs produced during the project.

4.3 Penalties, Rewards and Completion Dates

Penalties

Penalties for late delivery of a system appear quite frequently in tender documents. There most common form is one in which the net contract price is amortized at a daily rate for the period of the delay. These rates can be quite punitive and therefore pose a serious problem for any supplier bound by a contract clause permitting the client to vary the design at whim!

The precise form of penalties needs some examination. A purely linear scale would seem to be inappropriate beyond a certain period of time. At some point the client should be able to claim damages for lost business, loss of expected savings and etc. Precedents for the acceptance of such liability have yet to be set in this country for software contracts, but awards have been made in the U.S. since the late 1960's. (See Duggen (1970)). Contractors will be in a loss situation in any case, if the project is delayed and this must be considered when formulating penalty clauses.

Rewards

Penalties for delays should not be demanded unless rewards for early delivery are offered. Financial incentives commensurate with the linear penalties should be offered so that the supplier has a real incentive to complete the work ahead of time. The client, naturally enough, will have some date beyond which earlier deliveries have no further value. This point ought to be clearly stated, as should the date after which the delivery is of no value.

Completion Dates

The completion date itself should not be a fixed thing in any case. It does not seem reasonable to tie the introduction of penalties to 5 pm on some day fixed at the time the contract is signed. A more realistic approach is to specify a completion period based upon the original target date, after consultation with the supplier.

4.4 Unexpected Obstructions

It should be noted that Bryson (1970, page 5) points out that unexpected occurrences in the form of 'Adverse Physical Conditions and Artificial Obstructions' may be grounds for an engineering contractor requesting a variation in contract price. The software equivalent is most likely the discovery that some item of third party's software or hardware does not function as specified, delaying the contract very significantly.

Instances of this are not unknown in our profession. Defining such 'un-expected obstructions' would require very careful thought since the contractor may have been selected upon the basis of his (presumed) knowledge of a particular environment, a knowledge going beyond mere specification detail.

However, a new generation of errors in a recent operating system release, or a pernicious hardware problem may qualify.

4.5 Methods of Payment

Methods of payment should, of course be specified in the contract. The precise procedure to be adopted is difficult to specify since there may be problems in assessing the progress made, particularly towards the end of the project, as pointed by Ingrassa (1978). Flexibility is therefore required, and the client ought to be sympathetic to requests for unscheduled progress payments from a small supplier making good time on a project with verifiable milestones.

Any progress payment scheme ought to leave the client with sufficient funds in-hand to have another party finish the job.

4.6 Disputes and Arbitration

It is almost certain that disputes will arise during the project, and that there will be disagreements which the two parties cannot settle by negotiation.

There is provision, in most states' law, for the appointment of an Arbitrator given that a suitable clause is written into the contract. (See the Victorian Arbitration Act 1958). The engineering profession has a long history of arbitrating contract disputes, and it seems to work fairly well. Arbitrators are professionals in the field in question who hear evidence on the dispute, examine witnesses, inspect the 'works' and make a ruling which both parties are bound to accept.

The Arbitrator's ruling is binding in law and must be accepted. It can only be overturned on points of law, not on technical assessment.

Proceedings are usually informal, although parties may elect to be represented by legal council. The Arbitrator may seek legal advice, and the costs of this may be borne by the warring parties - or by the Arbitrator. The SIC is taking steps to have these possibilities publicised within our profession, and to have the Institution of Arbitrators made aware of the fact that computer professionals are interested in this question.

One suspects, however, that disputes should be resolved amicably, taking into account previous comments.

The extent of client participation in the project will undoubtedly affect the detection and handling of problems which will lead to a dispute.

This has already been considered in Part I of this paper (Reed 1978).

4.7 Ownership Of The Finished Product

The question of ownership of the finished product needs to be specified in the contract, and should influence the price. Complete ownership by the client should mean that he pays the full price. The cost should be reduced accordingly if the supplier wishes to re-sell the finished product. For example, Data General (Data General, 1974 page 7) charge $1000 for their Fortran IV compiler, a fraction of it's production cost.

CINCOM, the marketers of the TOTAL data base system, will move the system to a completely new machine provided the purchaser pays the full cost. The new TOTAL naturally remains CINCOM's, but the original client gets a rebate based upon the sales until he has paid the same as those purchasing TOTAL for a similar machine.

It might be argued that the supplier should receive a higher payment if the purchaser intends to resell the finished product. Alternatively, he may agree to a lower price and a percentage of the sales.

There is plenty of room for manoeuver in this area, and both sides should use it to the full extent.

4.8 Limitations On Liability

The supplier will want to be protected against unlimited damages in the case of his failure, and the client to be protected against the supplier. However, the liability ought to be limited to losses arising directly out of the failure to deliver the system. Penalties for late delivery have already been discussed along with the possible scope of damages.

Such matters should be thoroughly covered in the contract.

5. SOME QUESTIONS OF ATTITUDE

The success of the contract must depend upon the attitudes of the protagonists. Each parties' reason for being involved needs to be understood, since only in this way will small disputes be eliminated without pain.

5.1 Customer Attitudes

The supplier is in the business to make money. He will therefore expect the client to pay for any work over and above what has been agreed, or for work resulting from modifications agreed to by the client. Customer recognition of this factor is important, and it should be considered in assessing attitudes to a genuine estimating error by the supplier. It does not seem reasonable to this author to insist on a fixed price if the project is going satisfactorily but is consuming more resources than originally projected. Fixed price contracts are designed to protect the customer from cost overruns due to supplier carelessness, not as a means by which a shrewd operator can 'screw' his supplier. (Refer Symonds 1977 page 41).

The advantages of supplier involvement in the design and specification stage of the contract have already been stressed. What needs to be added is that the client should listen carefully to improvements suggested by the supplier and be prepared to accept the increased costs upon agreeing to them. Let he who gets the benifits pay.

The need for general controls has been covered adequately, but the client's attitude should be one of maximum co-operation.

5.2 Supplier Attitudes

The supplier has also to regard the project as a mutually beneficial arrangement rather than an effort to rip the client off. A fixed price contract should only be agreed to when there is some very high degree of certainty that the project can be kept on budget. Such arrangements constitute something of a gamble, and the client should expect to pay a premium for having his liabilities limited.

Much has been written about professional responsibilities elsewhere in this paper, and it only remains to add that the supplier's reputation is at stake at least in every contract he undertakes.

The supplier's attitude to small changes originating from the client should be sympathetic and based upon the actual work they generate. Perhaps his flexibility should be proportional to the effort needed by the customer to get a fixed price on the deal. As Symonds says "Why, when the software developer himself is loathe to quote a

fixed price and still requires certain escape clauses, should he expect the document to be flowless." (Symonds 1977 page 40). The context was that of the professional liability and standards of work covered earlier, but the argument surely applies to minor alterations along the way.

It should be noted that several authors (Reed 1977a) and (Fenwick & Coppinger 1972) do address the view which software houses have of themselves.

Reed (1977b) contains, in addition, analysis of the origins of software houses.

These factors influence supplier attitudes, and it would seem that the potential customer would be well advised to acquire such knowledge of his supplier origins, as well as their track records.

6. CONCLUSION

This paper attempts to highlight some of the problems to be faced by the Software Industry and it's users, although the bias appears to be in favour of the user. It cannot be claimed that all facits of the problem have been covered. There is no list of bloodied combatants to titilate the gossip-mongers. This is a subject for another paper.

Nor are there detailed recommendations on standards and contract format - these are tasks for the appropriate committees.

It is hoped, however, that the reader will obtain some insight into the problem areas and, by following up the major references cited be able to resolve some of his difficulties.

References

ACS (-) 'The Australian Computer Society Code of Professional Conduct' Australian Computer Society Inc. 1974

Arbitration Act (1958) 'Arbitration Act 1958' The Victorian Law Statutes No. 6200 1958 Vol. xxx pp. 201-207

ASA (1964) 'General Conditions of Contract for Civil Engineering Works' ASCA 24.1-1964

ASA (1969) 'X2-1969 Flowchart Symbols for Information Processing.' The Standards Association of Australia

Bryson, J. P. (1970 'Standard Form of Contract, Part I' in 'The Engineer and The Law, Series II' University of Sydney, Committee for Graduate Studies in the Department of Law, 1970

Data General Corp. (1974) 'Program Availability Schedule' Rev. 9/74 01000107 Data General Corporation

Dubnow, A. E. (1977) 'Computer Standards' Data Management October 1977 pp. 42-69

Duggen, M. (1970) 'A Case Digest: Suit for Damages in U.S. Data Processing Case.' Law and Computer Technology Vol. 3. No. 4. pp. 95-96 April 1970

Evans, R. W. & Assoc. (1975) 'Developing EDP Projects Successfully' EDP In-Depth Reports R. W. Evans & Assoc. Ltd. (Canada) Vol. 4, No. 11 July 1975.

Fenwick, P. and Coppinger, F. (1972) 'Investigation of the Computer Software Industry' MBA Thesis, Melbourne University

Gries, D. (1975) 'On Structured Programming - A Reply to Smolia.' ACM Forum in Comm. ACM Vol. 17. No. 11. Nov. 1974 pp. 655-657

Heistand, R. E. (1964) 'An Executive System Implemented as a Finite-Sate Automation.' Comm. ACM Vol. 7. No. 11. Nov. 1964 pp. 669-677

Hemdahl, G. (1973) 'The Function Flow Chart, A Specification and Design Tool for SPC Exchanges' in 'IEE Software Engineering for Telecommunications Switching Systems' Proceedings of the conference 2-5 April, 1973.

Ingrassa, F. S. (1978) 'Combatting the 90% Complete Syndrome' Datamation Vol. 24. No. 1. Jan 978 pp. 171-176

Kernighan, B. W. and Plauger, P. J. (1974) 'The Elements of Programming Style' McGraw-Hill 1974

Knuth, D. E. (1974) 'Structured Programming with GO TO Statements' ACM Computing Surveys Vol. 6. No. 4. Dec 1974 pp. 261-

Lindsey, C. H. (1977) 'Structure Charts - A Structured Alternative to Flow Charts' ACM Sigplan Notices Vol. 12. No. 11. Nov. 1977 pp. 36-37

Marquet, J. (1977) 'Traps and Pitfalls in Software Dealing', in 'Soft-where, Soft-why, Soft-how' Proceedings of the ACS-SIC Seminar June 1st 1977 pp 43-50

Maynard, J. (1972) 'Modular Programming' Butterworth and Co. 1972

Metz, S. (1975) 'More on Structured Programming' ACM Forum in Comm. ACM Vol. 18 No. 10 Oct. 1975 pp. 600-601

Oliver, S. R. (1977) 'Notes on the Use of Decision Tables' ACM Forum in Comm. ACM Vol. 20 No. 11 Nov. 1977 pp. 890-896

Oliver, S. R. and Jones, N. D. (1976) 'Program Control via Transition Matrices - a novel application of Microprogramming.' SIGPLAN Notices (ACM) Vol. 11 No. 4 April 1976 pp. 70-77

Parnas, D. L. (1972) 'A Technique for Software Module Specification with Examples' Comm. ACM Vol. 15 No. 5 May 1972 pp. 330-335

Peters, L. J. and Tripp, L. L. (1977) 'Comparing Software Design Methodologies' Datamation Nov. 1977 pp. 89-94

Pollock, S. L. et al (1977) Letters in Reply to 'On the Utility of Detailed Flow Charts in Programming' by Schneiderman et al (1977) ACM Forum in Comm. ACM Vol. 20 No. 11 pp. 889- 89

RAIA, (1970) 'Brick Veneer Specifications' Issued by the Housing Service of the Royal Institute of Architects Second Edition Feb. 1970

Reed, K. (1975) 'More on SP' ACM Forum in Comm. ACM Vol. 18 No. 7 July 1975 p. 423

Reed, K. (1977a) 'The Role of the Software Houses in Australia' ACS Victorian Branch Seminar August 1977

Reed, K. (1977b) 'Soft-where, Soft-why, Soft-how' First Annual Conference Australian Computer Society Queensland Branch (Proceedings) September 23rd to 25th 1977

Reed, K. (1978) 'Interfacing With the Software Industry Part - 1 The Organization of Contracted Software' ACS 8

Schneiderman, B. (1976) 'A Review of Design Techniques for Programs and Data' Software - Practice and Experience Vol. 6 pp. 555-567 1976

Schneiderman, B. et al (1977) 'Experimental Investigations of the Utility of Detailed Flow Charts in Programming' Comm. ACM Vol. 20 No. 6 pp. 373-381

Smith, G. (1969) 'Some Guidelines for the Design of Primary Data-Vetting Programs' ACJ Vol. 1 No. 4 pp. 194-200 (1969)

Smolia, S. W. (1974) 'On Structured Programming' ACM Forum in Comm. ACM Vol. 17 No. 5 May 1974 p. 294

Symonds, R. (1977) 'Development of Large-Scale Commercial Systems Without In-house Programmers' in 'Soft-where, Soft-why, Soft-how' Proceedings of the ACS-SIC seminar June 1st 1977 pp. 35-42

Thomson, B. and Coplan, N. (1967) 'Architectural and Engineering Law' Reinhold, 1967

Wayne, M. N. (1972) 'Flowcharting Concepts and Data Processing Techniques' Cranfield Press 1973

Wirth, N. (1971) 'Program Development by Stepwise Refinement' Comm. ACM Vol. 14 No. 4 April 1971 pp. 221-227

Wirth, N. (1974) 'On the Composition of Well-Structured Programs.' ACM Computing Surveys Vol. 6 No. 4 Dec. 1974 Special Issue: Programming pp. 247-260

Yohe, J. M. (1974) 'An Overview of Programming Practices' ACM Computing Surveys Vol. 6 No. 4 Dec. 1974 Special Issue: Programming pp. 221-246

Yourdon, E. (1975) 'Techniques of Program Structure and Design' Prentice-Hall, 1975