

Textbook Treatment of Structured Programming Standards and Guidelines

ABSTRACT: COBOL is still the language of choice in most business computer information systems (BCIS). As a result, COBOL remains the language of choice in structured programming courses required in most undergraduate and graduate computer information system (CIS) and business administration curricula in the United States and Canada. The cost of maintaining COBOL systems, however, can be very expensive for business. A collection of structured programming standards and guidelines (SPSGs) for CIS designed with COBOL has evolved to reduce maintenance costs. This review examines SPSG coverage in textbooks currently used in structured programming courses. These textbooks are evaluated and ranked on the basis of SPSG content. In turn, each SPSG is prioritized according to the degree of inter-textbook coverage. Three textbooks and four SPSGs stand out in the research findings along with suggestions for further empirical research.

Professor Douglas J. Leif
Business Administration Department
Bemidji State University
Bemidji, MN 56601-2699

KEYWORDS: *CIS Education, Structured COBOL Programming, Textbooks*

INTRODUCTION

COBOL In Industry

COBOL is the most widely used business programming language for both current and new business computer information systems (BCIS) (1, 2, 3, 4, 5, 6, 7). Because it is adaptable to client-server computing, personal computer work stations, window environments, computer aided software engineering and re-engineering, graphical user interfaces (GUI), and object oriented programming (OOP), COBOL will likely maintain its popularity (4, 6, 7, 8, 9, 10, 11). The attributes and determinants of COBOL's evolution contribute to it remaining the language of choice in structured programming courses required in most undergraduate and graduate CIS and business administration curricula in the United States and Canada [12, 13, 14, 15, 16].

Structured Programming Standards and Guidelines In Curricula

The approaches to delivering COBOL in curricula however, greatly differ, due in part to the fact that the textbooks most utilized in COBOL courses vary pedagogically. Many textbooks emphasize application development in their COBOL coverage, with individual and unrelated report programs in each chapter. Some textbooks emphasize system development, using

related and coordinated data processing programs, not just report programs, throughout many chapters. Several textbooks emphasize program maintainability via structured programming by introducing structured programming techniques in separate textbook units within some chapters. Others emphasize the general design of program logic (flowcharts followed by pseudocode followed by hierarchy charts) prior to coding programs. Even textbooks that have similar emphases in their approach will sequence concepts differently (for example, flowcharting before and after introduction to programming, and control break processing prior to and following sorting).

Industry Needs Of Structured Programming Standards and Guidelines

Based on industry needs, maintainability merits the highest priority concerning BCIS because 70% to 90% of CIS professional attention is given to the maintenance of COBOL detail design [10, 17, 18, 19]. Stated in terms of cost, a \$40,000 salaried CIS professional earns a \$32,000 portion for maintaining existing BCIS and only \$8,000 for developing new BCIS proposals. By reducing BCIS maintenance 10% a company with \$6 million budgeted for information system (IS) salaries can invest an additional \$600,000 annually in BCIS development.

This additional investment (without additional expenditures) towards BCIS development would be very important for companies seeking a technological edge over competitors.

Meeting Industry Needs of Structured Programming Standards and Guidelines

Research efforts have attempted to assist industry in its constant quest for increasing CIS professional productivity [20]. In a landmark study, structured programming techniques improved maintainability [21]. A collection of structured programming standards and guidelines (SPSGs) have evolved aimed at improving detail design techniques in an effort to reduce BCIS maintenance. SPSGs were intended to generate a more conforming coding style in BCIS, leading to enhanced maintainability and increased programmer productivity. Fifteen SPSGs (Table 1) have been collected, defined, detailed and justified individually elsewhere [22]. The assessment of SPSGs in textbooks must occur to improve instruction of highly structured programming. It should not be ignored because businesses cannot afford to continue to throw 70% to 90% of IS programmer /analyst salary budgets towards the maintenance of legacy BCIS.

SCOPE

Because most CIS professionals learn programming concepts in COBOL courses, the assessment of SPSGs in COBOL textbooks can provide insight into the depth of structured programming as a component of teaching priorities. Thirteen COBOL textbooks (Table 2) were examined for inclusion of each of the 15 SPSGs and ranked accordingly. The SPSGs were then ranked as instructional priorities in the authors' opinions, based on inclusion in the textbooks.

METHODOLOGY

Each textbook was reviewed for any type of coverage of an individual SPSG. The first textbook mention, discussion, example, or illustration of an SPSG was assigned a value of one. The breadth or depth of each SPSG reference, discussion, example, or illustration was not weighted. Therefore, a maximum point total could reach 15 for a textbook that covered every SPSG. A zero value was assigned to each unmentioned SPSG in every textbook. All values were accumulated for each textbook and they were ranked. All values were also totalled for each SPSG and they were ranked accordingly.

RESULTS

Textbook Findings

The Grauer and Villar textbook ranked highest addressing more SPSGs (14) than the other textbooks (Table 3). This textbook delivers more attention to programming structure than the others. Grauer and Villar emphasize the importance of coding accurate applications that are physically attractive, comprehensible, and therefore more easily maintained. The textbook by Horn and Gleason followed by addressing 11 SPSGs. The Spence and Windsor textbook ranked third by addressing 10 SPSGs. Wolff and Feingold covered nine SPSGs, and four textbooks tied for fifth in their treatment of eight SPSGs.

Structured Programming Standards and Guidelines Findings

Four SPSGs receive heavy inter-textbook coverage (Table 3). Indentation, SPSG-4, though not discussed in any narration, was shown in example programs consistently in all 13 texts. SPSG-7, the technique of numbering module prefixes in assigning paragraph names, proved to be the second most popular SPSG incorporated in 12 textbooks, followed by coding one read per input file, SPSG-6, illustrated in 11 books.

Table 1. STRUCTURED PROGRAMMING STANDARDS AND GUIDELINES (SPSGs) (22)

SPSG NUMBER	SPSG TITLE	SPSG DESCRIPTION
1	Consistent Prefixes	Programmer supplied names should contain consistent and meaningful prefixes for files, records within files, and all subordinate field items, group and elementary, within records
2	Documentation	The entire program should be described after the identification division. Comments should appear before each procedure division module.
3	Strategic Spacing	Blank lines should be inserted between natural program components, specifically divisions, sections, and modules. Also, align PIC clauses and VALUE clauses vertically.
4	Indentation	Code should be indented four spaces when sentences continue on subsequent lines.
5	Top Down Performance	A module should perform other modules located physically below the performing module. Avoid unconditional branching (GO TO).
6	One Read Per Input File	Only one read sentence for each input file should be coded and located in a separate read module.
7	Number Module Prefixes	A module numbering system in hierarchical fashion should be developed. The module number then becomes a prefix to the module name.
8	Driver Module Performs	Only PERFORMs should be coded in the main driver module, with the exception of the STOP RUN at the end of the driver.
9	Avoid Literals	Literals, numeric and nonnumeric, should be avoided in the procedure division, except for end of file switches of 'Y' and 'N'.
10	Cohesive Modules	Each module should serve one processing purpose.
11	Loosely Coupled Modules	Confusion results when program control hinges on data being passed from module to module. Minimize code execution in one module based on the contents of fields modified in another module.
12	Avoid Commas	Use commas only when editing print fields.
13	Perform Until Control Break Processing	The PERFORM UNTIL condition is a cleaner coding technique for control break processing.
14	Use Cross Reference List	A CRL is generated by the compiler after the source program is listed. A CRL identifies every programmer supplied name including sections, modules, files, records, and fields.
15	Use Exit Modules	Exit modules contain one word: EXIT. There is one exit module for every procedure division module except for the main driver module.

Strategic spacing, SPSG-3, like indentation, was not addressed in narration but shown in examples in 10 books. SPSG-2 (documentation), SPSG-5 (top-down performance), SPSG-8 (driver module performs), SPSG-10 (cohesive modules), SPSG-11 (loosely coupled modules), and

SPSG-12 (avoid commas) each scored seven, eight, or nine. The eight highest ranking textbooks were the only discussants of SPSG-11 (loosely coupled modules). The quality of textbook SPSG references varied. Both SPSG reference extremes were represented with some textbooks briefly

Table 2. COBOL PROGRAMMING TEXTBOOKS (in alphabetical order by author)

Author(s)	Textbook Title	Publisher
Bradley, J.	Comprehensive Structured COBOL [23]	Mitchell McGraw-Hill, New York, NY. 1990
Grauer, R.T. & Villar, C.V.	COBOL: From Micro To Mainframe [24]	Prentice-Hall, Inc., Englewood Cliffs, NJ. 1994
Horn, L.W. & Gleason, G.M.	Comprehensive Structured COBOL, 2nd ed. [25]	Boyd and Fraser, Danvers, MA. 1992
Nickerson, R.C.	Fundamentals of Structured COBOL, 3rd ed. [26]	HarperCollins, New York, NY. 1991
Paquette, G.A.	Advanced Structured COBOL [27]	Wm. C. Brown, Dubuque, IA. 1991
Paquette, G.A.	Structured COBOL Revised Edition [28]	Wm. C. Brown, Dubuque, IA. 1991
Philippakis, A.S. & Kazmier, L.J.	Comprehensive COBOL [29]	Mitchell McGraw-Hill, Watsonville, CA. 1991
Popkin, G.S.	Comprehensive Structured COBOL [30]	PWS-Kent, Boston, MA. 1993
Spence, J.W. & Windsor, J.C.	COBOL for Today, 3rd ed. [31]	West Publishing Company, St. Paul, MN. 1989
Stern, N. & Stern, R.	Structured COBOL Programming [32]	John Wiley and Sons, 1991
Welburn, T. & Price, W.	Structured COBOL 74/85 Fundamentals and Style, 3rd ed. [33]	Mitchell McGraw-Hill, Watsonville, CA. 1990
Wolff, L. & Feingold, C.	Fundamentals of Structured COBOL Programming, 6th ed. [34]	Wm. C. Brown, Dubuque, IA. 1991
Yarmish, R. & Wohl, G.	Structured COBOL A Direct Approach [35]	Prentice-Hall, Inc., Englewood Cliffs, NJ. 1993

Table 3. TEXTBOOK RANK ACCORDING TO SPSG COVERAGE AND SPSG RANK ACCORDING TO TEXTBOOK COVERAGE

Author(s)	SPSGs Covered By Textbook	SPSGs														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Grauer, R.T. & Villar, C.V.	14	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
Horn, L.W. & Gleason, G.M.	11	0	1	1	1	1	1	1	1	0	1	1	0	1	0	1
Spence, J.W. & Windsor, J.C.	10	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
Wolff, L. & Feingold, C.	9	0	0	1	1	0	1	1	1	1	1	1	0	0	0	1
Philippakis, A.S. & Kazmier, L.J.	8	0	1	0	1	1	0	1	1	0	1	1	1	0	0	0
Yarmish, R. & Wohl, G.	8	0	1	1	1	0	1	1	0	0	1	1	1	0	0	0
Stern, N. & Stern, R.	8	1	0	1	1	1	1	1	1	0	0	1	0	0	0	0
Welburn, T. and Price, W.	8	0	1	0	1	1	1	1	0	0	1	1	1	0	0	0
Nickerson, R.C.	7	0	1	1	1	0	1	1	0	0	0	0	1	1	0	0
Paquette, G.A. (Adv.)	6	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0
Paquette, G.A. (Rev.)	6	0	1	1	1	0	1	1	1	0	0	0	0	0	0	0
Bradley, J.	5	0	1	0	1	1	0	1	0	0	0	0	1	0	0	0
Popkin, G.S.	4	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
Number of References Per SPSG		2	9	10	13	8	11	12	7	3	7	8	7	3	1	3


mentioning an SPSG and others dedicating several pages to an SPSG. Also, various SPSGs were exemplified programmatically in textbooks without specific narration. For example strategic spacing (SPSG-3) and indentation (SPSG-4) were not explained or discussed in the books incorporating them, yet example programs consistently employed the standards. A few books targeted SPSGs in their discussion but illustrated conflicting programming examples. For example, top-down processing (SPSG-5) was recommended but example programs illustrated one module performing another module located physically above the calling module [24, 29, 30, 31, 32].

The technique of naming files, their records, and their fields with the same prefix (SPSG-1) in a COBOL program provides a great advantage for quickly identifying data definition location (source) and purpose. Determining the source and purpose of files, records, and fields may be crucial in improving comprehension (analysis) leading to quicker and more accurate maintenance. Yet employing consistent prefixes (SPSG-1) scored only two. Other techniques that received low priority were SPSG-9 (avoid literals), SPSG-13 (perform until control break processing), and SPSG-15 (exit modules), scoring three each. Only SPSG-14 (use cross reference list) scored one, which isn't surprising because cross reference lists vary due to the fact that they are compiler options dependent upon software vendors.

CONCLUSION

Overall, current textbooks incorporate SPSGs in greatly varying degrees. The textbook by Grauer and Villar was rated the highest by citing 14 of 15 SPSGs. Horn and Gleason was second with 11 SPSGs cited. Spence and Windsor was third 10 points. Sixty-nine percent of the reviewed textbooks covered at least seven SPSGs. Four SPSGs were thoroughly covered throughout the textbooks: Indentation (SPSG-4); number module prefixes (SPSG-7); one read per input file (SPSG-6); and strategic spacing (SPSG-3). Also, authors seem to consistently express or imply their belief that employment of SPSGs positively correlates to reduced maintenance. Because research suggests COBOL's popularity will continue to spread, and because most CIS professionals receive their initial basic COBOL education in academic programming courses, SPSGs should be incor-

porated at this level. Industry's input on their perception of SPSG priorities would be useful to determine if academia is educating future CIS professionals adequately in SPSG techniques.

A marked difference exists in SPSG treatment among textbooks, and presumably CIS application courses. Further research is necessary to align the delivery of SPSGs within COBOL education with industry's use of SPSGs. Also, SPSGs must undergo laboratory research similar to the examination flowcharts have undergone [36]. This will enable us to empirically validate and prioritize individual SPSGs as maintenance improvement tools. Once a refined body of SPSGs is thoroughly tested and accepted, academia can better educate future CIS professionals for industry. 

REFERENCES

1. Arnett, K.P. and M.C. Jones. "Programming Languages: Today and Tomorrow," *Journal of Computer Information Systems*, 33:4, Summer 1993, pp. 77-81.
2. Bauman, B.M., J.K. Pierson and K.A. Forcht. "Business Programming Language Preferences in the 1990s," *Journal of Computer Information Systems*, 32:1, Fall 1991, pp. 13-16.
3. Beheshti H.M. and M.R. Mattson. "Information Systems Needs of Large Corporations," *Journal of Computer Information Systems*, 33:3, Spring 1993, pp. 35-37.
4. McMullen, J. "Why PC COBOL is gaining ground," *Datamation*, 8:19, May 15, 1991, pp. 70-72.
5. Metcalfe, R.M. "Are There Any COBOL Users of the Fifth Kind?" *InfoWorld*, 14:47, November 23, 1992, pp. 40.
6. Ray, G. "Interest In Object COBOL Grows: Users, Vendors Predict Language Will Be Used To Develop New Applications," *Computerworld*, 26:20, May 18, 1992, pp. 95.
7. Snell, N. "Are You Ready for Cutting Edge COBOL?" *Datamation*, 38:21, October 15, 1992, pp. 77-78.
8. Anthes, G.H. "Uncle Sam Enlisting Aid of Software Re-engineering," *Computerworld*, 26:11, March 2, 1992, pp. 51-56.
9. Cunningham, C.A. "COBOL-Based System Aids Firm's Downsizing," *PC Week*, 9:24, June 15, 1992, pp. 75-80.
10. Dewire, D.T. *Client/Server Computing*, James Martin McGraw-Hill Productivity Series, McGraw-Hill, New York, NY, 1993.
11. Leach, N. "Microsoft Widens Windows Features of COBOL Compiler," *PC Week*, 10:11, March 22, 1993, pp. 61-62.
12. Albin, M. and R.W. Otto. "The CIS Curriculum: What Employers Want From CIS and General Business Majors," *Journal of Computer Information Systems*, 28, Summer 1987, pp. 15-19.
13. Athey, S. "A comparison of Undergraduate Information Systems Programs and the DPMA Model," *Interface*, 1989, pp. 68-73.
14. Dawley, D.L. "A Management Information Systems MBA-Major," *Journal of Information Systems Education*, 3:1, Spring 1991, pp. 17-21.
15. Longenecker, H.E. and D.L. Feinstein. "A Comprehensive Survey of USA and Canadian Undergraduate Programs in Information Systems," *Journal of Information Systems Education*, 3:1, Spring 1991, pp. 8-13.
16. Stolen, J. "The Undergraduate MIS Curriculum: A Sampling Of AACSB Schools," *Journal of Computer Information Systems*, 33:2, Winter 1992-1993, pp. 54-57.
17. Martin, M.P. *Analysis And Design Of Business Information Systems*, Macmillan Publishing Company, New York, NY, 1991.
18. Corbi, T.A. "Program understanding: Challenge for the 1990s," *IBM Systems Journal*, 28:2, 1989, pp. 294-305.
19. Zeltmann, S.M. and M.L. Vineyard. "Maintaining Computer-Based Systems: A Case and Prescription for Computer Information Systems Curricula," *Journal of Computer Information Systems*, 33:1, Fall 1992, pp. 41-45.
20. Headrick, R.W. "COBOL-85's Impact on Teaching Programming: Opportunities For Improvement," *Proceedings of the Tenth Annual Information Systems Education Conference*, DPMA Education Foundation, November, 1993, pp.84-88.
21. Dijkstra, E. "Go To Statement Considered Harmful," *Communications of the ACM*, 11:2, February 1968, pp.147-48.
22. Leif, D.J. "Evaluating computer Information System Detail Design Conventions," 1993 *Journal/Proceedings: Information Systems and Quantitative Methods*, Midwest Business Administration Association, March, 1993, pp. 107-113.
23. Bradley, J. *Comprehensive Structured COBOL*, Mitchell McGraw-Hill, New York, NY, 1990.
24. Grauer, R.T. and C.V. Villar. *COBOL: From Micro To Mainframe*, 2nd edition, Prentice-Hall Incorporated, Englewood Cliffs, NJ 1994.
25. Horn, W.L. and G.M. Gleason. *Comprehensive Structured COBOL*, 2nd edition, Boyd and Fraser Publishing Company, Danvers, MA, 1992.
26. Nickerson, R.C. *Fundamentals of Structured COBOL*, Harper-Collins, New York, NY, 1991.
27. Paquette, G.A. *Advanced Structured Cobol*, W.C. Brown Publishers, Dubuque, IA, 1991.
28. Paquette, G.A. *Structured Cobol Revised Edition*, W.C. Brown Publishers, Dubuque, IA, 1991.
29. Philippakis, A.S. and L.J. Kazmier. *Comprehensive Cobol*, Mitchell McGraw-Hill, Watsonville, CA, 1991.
30. Popkin, G.S. *Comprehensive Structured COBOL*, 4th edition, PWS-KENT Publishing Company, Boston, MA, 1993.
31. Spence, J.W. and J.C. Windsor. *COBOL For Today*, 3rd edition, West Publishing Company, St. Paul, MN, 1989.
32. Stern, N. and R.A. Stern. *Structured Cobol Programming*, 6th edition, John Wiley and Sons, New York, NY, 1991.

33. Welburn, T. and W. Price. *Structured COBOL 74/85 Fundamentals and Style*, 3rd edition, Mitchell McGraw-Hill, New York, NY, 1990.

34. Wolff, L. and C. Feingold. *Fundamentals of Structured COBOL Programming*, 6th edition, W.C. Brown Publishers, Dubuque, IA, 1991.

35. Yarmish, R. and G. Wohl. *Structured COBOL: A Direct Approach*, Prentice-Hall Incorporated, Englewood Cliffs, NJ, 1993.

36. Hwang, M.I. "Flowcharts As Programming Aids: A Human Information Processing Perspective," *Journal of Computer Information Systems*, 32:3, Spring 1992, pp. 29-34.

AUTHOR'S BIOGRAPHY

Douglas Leif is an Assistant Professor of Business Administration at Bemidji State University where he teaches business and computer information system courses. He received his Masters degree specializing in management information systems from Iowa State University. His research interests include detail design in the analysis and design of information systems.

Subscribe!

Subscribe to the Journal of Information Systems Education and you'll get far more than this valuable journal! Because the Journal of Information Systems Education is published by EDSIG, the DPMA's special interest group for education. And as a subscriber, you'll get:

- Four quarterly issues
- Special discounts on EDSIG sponsored activities
- Networking opportunities with others in your profession
- Access to members via Bitnet/Internet
- A forum for discussion of IS teaching and training topics
- The ability to share information on student chapters

Yes! Start my subscription today. I've enclosed a check for \$35, payable to the Journal of Information Systems Education, with this completed coupon. (Mail to: John Corrigan, COMPUTERWORLD, 375 Cochituate Road, Framingham, MA 01701.)

Name: _____

Company or Institution: _____

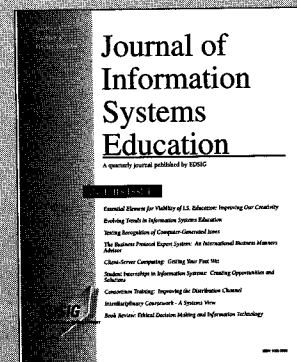
Address: _____

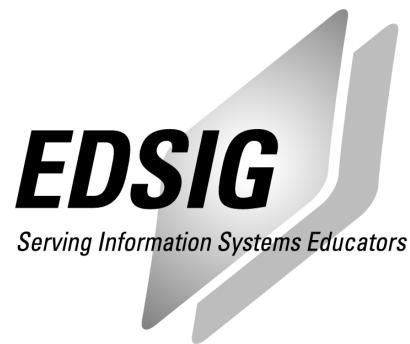
City/State/Zip: _____

Phone: _____

Fax: _____

E-mail: _____





STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

Copyright ©1994 by the Information Systems & Computing Academic Professionals, Inc. (ISCAP). Permission to make digital or hard copies of all or part of this journal for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial use. All copies must bear this notice and full citation. Permission from the Editor is required to post to servers, redistribute to lists, or utilize in a for-profit or commercial use. Permission requests should be sent to the Editor-in-Chief, Journal of Information Systems Education, editor@jise.org.

ISSN 1055-3096