Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1997 Proceedings

Americas Conference on Information Systems (AMCIS)

8-15-1997

A Database-project Approach to a Cobol Course

R.J Diagle PH.D
University of South Alabama, daigle@cis.usouthal.edu

A Database-project Approach to a Cobol Course

R. J. Daigle, Ph.D.
School of CIS
University of South Alabama
Mobile, AL 36688
(internet: daigle@cis.usouthal.edu)

Abstract

For many IS academic programs, COBOL is viewed as an obligatory tool necessary for its graduates. Yet, the immediate marketability of skills in Visual Basic and C++ creates a dilemma for curriculum architects: How to efficiently meet the COBOL obligation and in a manner that does not penalize the marketability of students? This paper describes an approach to a COBOL course that overcomes this dilemma.

Introduction

Most IS majors view COBOL as archaic and verbose: It is a burden to be endured. Yet the majority of legacy systems rely on COBOL code, and COBOL programmers continue to be in high demand. This fact requires that IS majors should have familiarity with COBOL. Confronted by demands for popular programming environments, faculty are challenged for ways to meet the COBOL obligation in a reasonable time frame and in a manner that is not regarded as a penalty by students. This paper describes an approach to a COBOL course that accomplishes several objectives: it reduces the COBOL time frame (in our situation, from two quarters to one quarter); it is not syntax-driven; and it provides a practical foundation to the Database Management and Systems Analysis and Design courses in the IS curriculum.

Course Methodology

COBOL textbooks(e.g. Stern and Stern, 1997; Horn and Gleason, 1995; Shelly and Cashman, 1977) typically take an incremental approach to learning: the approach is often syntax-driven. Students are exposed to the use of COBOL incrementally, building from sequential reporting to indexed sequential file maintenance and access. There are problems associated with the typical approach: significant time is required to cover even a part of the syntax, programming assignments seem contrived and artificial, and there is an overwhelming volume of detail that obscures seeing the "big picture". What is needed is a fresh, innovative approach that results in an interesting and meaningful COBOL course.

Children learn to play a sport when it is fun--when the focus is on the game rather than the benefits of physical activity or the skills used in the game. Kids become aware of the need for skill development because of play experiences. They become disinterested when there is a predominant interest in skills over play.

The principles we can learn from these observations are:

An interesting and meaningful game (problem goal) is more motivating than an excessive focus on detailed knowledge/skills.

Identifying knowledge/skills should be by the players as a consequence of playing the game.

"Coaching" should not be about details only; it should be about the game as well.

Our course objective reflects our understanding of these principles: *To implement, using COBOL, a maintenance and reporting system according to a normalized E-R diagram with 3-4 entity classes, one of the entity classes is on the "many" side of two one-to-many associations.*

Students entering the course are expected to have completed at least two quarters of introductory programming concepts: basic control structures, primitive data structures, and elements of program design. The course is laboratory-based and project-centered (rather than tool-centered). The traditional lecture approach is abandoned in favor of approaches that promote active and proactive learning. For example, *just-in-time-learning* is used to introduce concepts related to project development and management and *lecture-by-demand* is used to respond to implementation details. Project implementation is facilitated by two tools. Micro Focus Personal COBOL 2.0 for D0S is used for its Application Generator feature to build an initial maintenance program for entity classes. Each student is given a Menu Processing Tool, written in COBOL, that provides a common interface and a means for phased integration testing of modules tested as stand-alone programs. Additional student support is achieved by the access to an HTML laboratory manual that describes how to use the Micro Focus COBOL compiler according to course objectives.

Discussion

There were five phases to the course project: problem definition, model development, simple report development, complex report development, and project delivery and assessment. A discussion of the activities and concepts addressed in each phase follows.

During Phase I, the class is guided through a Systems Analysis and Design mini-exercise for a class developed scenario of limited complexity. The class agrees on a problem definition statement along with constraints. Solution requirements that center on simple maintenance of entity classes and relationships and on reporting requirements are also prepared by the class. The final derived logical design is expressed as an E-R data model.

There is an overlap of activities in Phase I and Phase II: While the problem analysis and design is taking place, students are given laboratory instruction in the use of the MicroFocus screen builder and application generator to obtain a maintenance program for an example, a one-to-many data model (DEPARTMENT--> EMPLOYEE). The first COBOL coding experience is the modification of the generated maintenance program to accept composite keys and duplicate alternate keys; next the students construct an external subprogram to implement referential integrity. The third coding exercises is the insertion of a CALL from generated maintenance programs to external subprograms. A student will have completed Phase II when the technical skills acquired during the laboratory experience have been applied to the E-R data model of Phase I. Phase II should be completed in time for midterm examination.

The emphasis in Phase III is preparing reports for one entity class. One method examined addresses the role of primary and alternate keys; another addresses the appropriate use of the SORT verb for orderings not accounted for by either a primary or alternate key. SQL statements are used as a specification language and a vehicle for communicating the relationship between file operations used in preparing the reports and the relational operators, Projection and Selection.

The emphasis in Phase IV is preparing multiple-entity reports. A single-level control break report is key-driven from the *one* side of a *one-to-many* relation via the foreign key (an alternate key) on the *many* side. A second report type examined (a relational join) is a report from the entity class on the *many* side to the entity class on the *one* side using the foreign key-to-key linkage. A third type combines the first two types to produce a report for two *one-to-many* relationships using a common *many* side entity class. Using SQL as a specification language is continued in this phase. The required file operations are explained in terms of the relational operators, Join, Projection, and Selection.

The last phase, Project Delivery and Assessment, is the culmination of the project. Several activities are on-going for the student from Phase II to Phase IV: standardized documentation defined by a word processing template, program testing as stand-alone applications, and system integration and incremental system testing with the Menu Processing Tool. A grade assessment form, distributed during Phase II, shows the relationship of project grade to successful phase completion: a C for Phase II only, a B for Phases II and III, and an A for Phases II, III, and IV. To emphasize individual responsibility, each student is required to

schedule a product demonstration. The product demonstration, student-keyed but instructor-directed, is treated as a black-box system test and is scored by the student before departure. Final delivery products are project documentation, a copy of the final assessment, a computation of space requirements according to reasonable assumptions of system activities for a one-year system deployment, and a project disk prepared for batch file installation.

Conclusions

The benefits of this approach are numerous:

COBOL is the means to achieve the end result, not the result itself.

Students are prepared for a Database course: they know and can use the terminology.

Students are prepared for a Systems Analysis and Design sequence: they have participated in and documented a mini-systems project from definition to implementation and delivery.

Students have experience with three different types of testing: module, system integration, and black box testing.

Using SQL statements as a specification language accomplishes two things: familiarity with SQL statements and the relationship of SQL statements to file and relational operations.

The approach demonstrates a practical approach to gaining mastery of a new tool: the database course could begin with an assignment to implement the project with Access.

Former students express self-confidence and freedom from tool-enslavement: when a wizard does not provide a desired outcome, they have the confidence to perform at a lower level of implementation.

There are two inherent weaknesses with the approach. Coverage is not complete: arrays and table processing, sequential file processing, and building indexed sequential file maintenance applications are not covered. The approach is facilitated by the Application Generator of MicroFocus COBOL: if there is a commitment to a specific compiler that does not have the Application Generator, this approach may require the development of additional course tools to leverage the presence of indexed files to reach a database oriented approach.

Acknowledgment: The author wishes to express appreciation to three seniors who assisted in the "Proof of Concept" for the approach: K. C. Sng, Anne Hartley, and Seyed Abbas.

References

Horn, L. Wayne and Gleason, Gary, *Comprehensive Structured COBOL*, Third Edition, Boyd & Fraser Publishing Company, 1995.

Micro Focus Personal COBOL 2.0 for DOS, Programmer's Guide, Micro Focus Group, 1993.

Stern, Nancy and Stern, Robert, *Structured COBOL Programming*, Eighth Edition, John Wiley & Sons, Inc., 1997.

Shelly, Gary and Cashman, Thomas, *Introduction to Computer Programming Structured COBOL*, Anaheim Publishing Company, Fullerton, CA, 1977.