The South African Institute of Computer Scientists
and
Information Technologists

Proceedings

of the

1996 National Research and Development Conference

Industry meets Academia

Interaction Conference Centre, University of Natal,
Durban.
26 & 27 September

Edited by
Vevek Ram

ISBN 0-620-20568-7

Cover printed by Natal Printers (Pty) Ltd, Pietermaritzburg
Copying by the Multicopy Centre, University of Natal, Pietermaritzburg
Binding by Library Technical Services, University of Natal, Pietermaritzburg

The views expressed in this book are those of the individual authors

FOREWORD

This book is a collection of papers presented at the National Research and Development Conference of the Institute of Computer Scientists and Information Technologists, held on 26 & 27 September, at the Interaction Conference Centre, University of Natal, Durban. The Conference was organised by the Department of Computer Science and Information Systems of The University of Natal, Pietermaritzburg.

The papers contained herein range from serious technical research to work-in-progress reports of current research to industry and commercial practice and experience. It has been a difficult task maintaining an adequate and representative spread of interests and a high standard of scholarship at the same time. Nevertheless, the conference boasts a wide range of high quality papers. The program committee decided not only to accept papers that are publishable in their present form, but also papers which reflect this potential in order to encourage young researchers and to involve practitioners from commerce and industry.

The organisers would like to thank IBM South Africa for their generous sponsorship and all the members of the organising and program committees, and the referees for making the conference a success. The organisers are indebted to the Computer Society of South Africa (Natal Chapter) for promoting the conference among its members and also to the staff and management of the Interaction Conference Centre for their contribution to the success of the conference.

On behalf of the Organising Committee
Vevek Ram
Editor and Program Chair
Pietermaritzburg, September 1996

Organising Committee

Conference General Chairs
Mr Rob Dempster and Prof Peter Warren (UNP)

Organising Chair
Dr Don Petkov (UNP)

Secretariat
Mrs Jenny Wilson

Program Chair
Prof Vevek Ram (UNP)

Program Committee

Prof Peter Wentworth, Rhodes
Dr Milan Hajek, UDW
Prof Derek Smith, UCT
Prof Anthony Krzesinski, Stellenbosch
Dr Don Petkov, UNP
Mr Rob Dempster, UNP
Prof Peter Warren, UNP

Table of Contents

Keynote Speaker

Plenary

Research

Intelligent Systems

Real-Time Systems

Database Systems

Education

Practice and Experience

Information Technology and Organizational Issues

Abstracts

List of Contributors

Vladimir B Bajic
Centre for Engineering Research,
Technikon Natal,
P O Box 953
Durban 4000

C N Blewett
Department of Accounting
University of Natal
King George V Avenue
Durban 4001

Justin Cansfield
Department of Accounting
University of Natal
King George V Avenue
Durban 4001

Tom Considine
Apron Services (Pty) Ltd
P O Johannesburg
International Airport
1600

Eric Cloete
School of Electrical Engineering
Cape Technikon
Box 652
Cape Town

I Cloete
Computer Science Department
University of Stellenbosch
Stellenbosch
7600

O A Daini
Department of Computer Science
University of Botswana
Gaborone
Botswana

Nirvani Devcharan
Umgeni Water
Box 9
Pietermaritzburg
3200

P J A de Villiers
Department of Computer Science
University of Stellenbosch
Stellenbosch
7700

Ruth de Villiers
Department of Computer Science and
Information Systems
UNISA
Box 392, Pretoria, 0001

G J Erwin
Business Information Systems
University of Durban-Westville
Private Bag X54001
Durban 4000

G Ganchev
Computer Science Department
University of Botswana
PBag 0022
Gaberone, Botswana

J Geldenhuys
Department of Computer Science
University of Stellenbosch
Stellenbosch
7700

Louise Gibson
BIS, Dept Accounting & Finance
University of Durban
Pvt Bag X10
Dalbridge 4014

Mike Hart
Department of Information Systems
University of Cape Town
Rondebosch
7700

M. Hajek
Department of Computer Science
University of Durban-Westville
Pvt Bag X54001
Durban 4000

A C Hansen
Dept of Agricultural Engineering
University of Natal
Private Bag X01
Scottsville 3209

J M Hattingh
Department of Computer Science
Potchefstroom University for CHE
Potchefstroom 2520

Boris Jankovic
Centre for Engineering Research
Technikon Natal
P O Box 953,
Durban 4000

Paula Kotze
Department of Computer Science and
Information Systems
UNISA
Box 392
Pretoria, 0001

J W Kruger
Vista University
Soweto Campus
Box 359
Westhoven 2124

A C Leonard
Dept of Informatics
University of Pretoria
Pretoria
2000

Ben Laauwen
Laauwen and Associates
P O Box 13773
Sinoville
0129

Mari Le Roux
Information technology, development: project
leader
Telkom IT 1015
Box 2753
Pretoria 0001

P W L Lyne
Dept of Agricultural Engineering
University of Natal
Private Bag X01
Scottsville 3209

Rose Mazhindu-Shumba
Computer Science Department
University of Zimbabwe
Box MP167
Harare, Zimbabwe

Meredith McLeod
Centre for Engineering Research,
Technikon Natal,
P O Box 953
Durban 4000

D Moodley
Computer Management Systems
Box 451
Umhlanga Rocks
4320

Andrew Morris
P O Box 34200
Rhodes Gift
7707

R C Nienaber
Technikon Pretoria
Dept of Information Technology
Private Bag X680
Pretoria 0001

E Parkinson
Department of Computer Science
University of Port Elizabeth
Box 1600
Port Elizabeth 6000

Don Petkov
Department of Computer Science and
Information Systems
University of Natal
PBag x01
Scottsville 3209

Olga Petkov
Technikon Natal
Box 11078
Dorpspruit 3206
Pietermaritzburg

N Pillay
Technikon Natal
Box 11078
Dorpspruit 3206
Pietermaritzburg

V Ram
Department of Computer Science and
Information Systems
University of Natal
PBag x01
Scottsville 3209

Melinda Redelinghuys
Department of Computer Science and
Information Systems
UNISA
Box 392
Pretoria, 0001

Karen Renaud
Computer Science and Information Systems
UNISA
Box 392
Pretoria, 0001

H N Roux
Department of Computer Science
University of Stellenbosch
Stellenbosch
7700

T G Scott
Department of Computer Science
Potchefstroom University for CHE
Potchefstroom
2520

T Seipone
Department of Computer Science
University of Botswana
Gaborone
Botswana

Derek Smith
Department of Information Systems
University of Cape Town
Rondebosch
7700

Anette L Steenkamp
Department of Computer Science and
Information Systems
UNISA
Box 392
Pretoria, 0001

T Steyn
Department of Computer Science
Potchefstroom University for CHE
Potchefstroom 2520

H. Suleman
Department of Computer Science
University of Durban-Westville
Pvt Bag X54001
Durban 4000

A Vahed
Department of Computer Science
University of Western Cape
Private Bag X17
Bellville 7530

A Van der Merwe
Computer science and Informations Systems
UNISA
P O Box 392
Pretoria,0001

Tjaart J Van Der Walt
Foundation for Research and Development
Box 2600
Pretoria, 0001

K Vavatzandis
Department of Information Systems
University of Cape Town
Rondebosch
7700

Y Velinov
Dept Computer Science
University of Natal
Private Bag X01
Scottsville 3209

H Venter
Department of Computer Science
University of Port Elizabeth
Box 1600
Port Elizabeth 6000

H L Viktor
Computer Science Department
University of Stellenbosch
Stellenbosch
7600

R Von Solms
Department of Information Technology
Port Elizabeth Technikon
Private Bag X6011
Port Elizabeth 6000

A J Walker
Software Engineering Applications
Laboratory
Electrical Engineering
University of Witwatersrand
Johannesburg

P Warren
Computer Science Department
University of Natal
P/Bag X01
Scottsville 3209

Max Watzenboeck
University of Botswana
Private Bag 0022
Gaberone
Botswana

COBIE: AN INTEGRATED COBOL ENVIRONMENT

N.Pillay
Department of Financial Studies
Technikon Natal - Pietermaritzburg

Abstract

Development software is one of the courses that first year students studying towards a Diploma in Financial Studies enrol for. This course contains an appreciable amount of RMCOBOL programming. Past experience is indicative of the fact that students seem to have difficulty programming in RMCOBOL. This paper looks at some of the problems experienced by these students and describes COBIE, a **Cobol Integrate Environment**, developed to assist students overcome these problems. Furthermore the paper provides a description of the students experience with COBIE. Attention is also given to the effect that a student's learning style may have on the difficulty experienced by a student in using COBIE. Finally ways in which COBIE can be extended to meet the needs of novice computer programmers is examined.

Introduction

As part of the course on software development first year Financial Studies Students are required to complete a course on COBOL programming. These students, majority of whom are novice computer users, experience much difficulty in programming in RMCOBOL. This paper firstly provides an account of the problems experienced by these students. It then goes on to illustrate how a phased development of a programming environment can possibly provide a means of overcoming the problems experienced by students.

On outline of each of the phases involved in the development of this programming environment is then provided. A detailed description of the first phase of the development of COBIE is given. This description firstly examines the facilities provided by COBIE and then gives an account of the students' experiences with COBIE.

Finally further extensions to the system in order to meet the needs of these novice programmers is reviewed.

Problems Encountered

The problems experienced by students are two-fold:

Problem 1 :

Students experienced difficulty in using the RMCOBOL system. In order to create, run and compile COBOL programs using RMCOBOL the following procedure had to be followed:

1. The user had to create the source and data files in an editor or wordprocessor of his or her choice.
2. The user had to then evoke the RMCOBOL Compiler at the command line.
3. Step 2 resulted in a compiler listing either being scrolled across the screen or being written to a list file if the user specified this option.
4. To view the list file the user would have to again use an editor or word processor of his or her choice.
5. Based on the listing described in steps 3 and 4 the user would then have to correct any errors listed by the compiler in the source code.
6. Finally the user would call the program RUNCOBOL at the DOS prompt to run the COBOL program.

As a result these students, being novice computer users experienced much difficulty in just creating, editing, compiling and running RMCOBOL programs. Students found the procedure that had to be followed to complete these tasks difficult to use and remember.

Problem 2:

Students experienced much difficulty an d frustration i n programming in COBOL. Barstow et a l [BARS84] define this as a "complexity barrier" which hampers the productivity of novice programmers. According to Barstow et al this barrier can best be broken by building large programming support systems called **programming environments**.

A possible solution to **Problem 1** is for students to use **RMCOSTAR** which is a RMCOBOL project manager and editor from which COBOL programs can be edited, compiled, run and debugged. Reasons for not using **RMCOSTAR** include the following:

- RMCOSTAR is generally not sufficiently user friendly for novice computer users. For example files are stored according to projects which can lead to these confusion.

- It would not be possible to extend the RMCOSTAR system in order to provide a solution to the difficulties specified in **Problem 2**.

Proposed Solution

According to Teitelman et al [BARS84] a programming environment that is "cooperative and helpful" will prevent a novice programmer from spending most of his or her "time and energy fighting a system that at times seems bent on frustrating his best efforts". Hence based on the literature surveyed it was decided that a possible solution to the **Problem 2** defined above would be to develop a programming environment which these novice programmers would be "comfortable" using. Examples of such programming environments already in existence include Cedar which was developed from the Mesa environment, Pecan, Pict, and the Cornell Program Synthesiser just to name a few.

The foundation of the programming environment to be developed would be an integrated environment which would help users overcome the difficulties specified in **Problem 1**.

According to Teitelman et al [BARS84] the contribution made by users in terms of suggestions of improvements that should be made to Interlisp and facilities that should be provided by the Interlisp Programming Environment have proven invaluable. Hence throughout this study much emphasis will be placed on user feedback.

The development of COBIE will be undertaken in three phases:

Phase 1:

1. The development of an integrated environment that enables the user to edit, compile, and run COBOL programs in an attempt to provide a solution to **Problem 1**.

2. Evaluation of this environment.

Phase 2:

1. During this phase improvements will be made to COBIE based on the evaluation in Phase 1.

2. According to Barstow et al [BARS84] a programming environment must combine the powers of

a compiler, an editor, a debugging system, a documentation system and a problem solver. COBIE will be extended to provided a debugger and documentation. During the process of debugging users will be able to make necessary changes to the source code.

3. Evaluation of th e extended system.

Phase 3:

1. Improvements made to the system based on the evaluation in **Phase 2**.

2. Additional mechanisms need to be added to COBIE to assist students to program in COBOL. The following mechanisms will be examined in order to extend COBIE to a environment in which these novice programmers are "comfortable" programming in:

- In their description of an integrated Prolog Programming Environment Schreweis et al [SCHR93] emphasise that an "UNDO" option should be available to users. The importance of an "UNDO" option is still further emphasised in Teitelman et al's description of the Interlisp programming environment [BARS84].

- An automatic error correction facility:

  Teitelman et al in their paper " Automated Programmering: The Programmer's Assistant" [BARS84] describe DWIM a mechanism for automatically correcting trivial errors made by programmers such as spelling mistakes and punctuation errors. The importance of such a facility is further stressed by the developers of the Interlisp environment.

- Barstow et al [BARS84] emphasis that "In writing a program, the use should not need to be continually concerned with the exact form of the structures being used."

  This idea forms the basis of the Cornell Program Synthesizer (CPS). CPS is a programming environments that provides users with templates of the grammars of each simplified statement of a programming language. All the user has to do is provide the arguments or expressions at a cursor position in these templates. The use of such COBOL templates in COBIE will be looked at.

- Question and answering:

  Barstow et al [BARS94] describe a question and answering technique to help uses identify the effects of making certain changes in their programs prior to making these changes.

- An interactive browser to display various view of the current state of computation of a program.

- According to Schreweis [SCHR93] " The human mind is strongly visual oriented and acquires information at a significantly higher rate by discovery of graphical relationships in complex pictures than by reading text". The significance of the use of visual environments is further stressed by Shu [GLIN90] who states that pictures are more powerful than words as a means of communication.

  An example of a visual environment is PECAN. Shu describes PECAN as a system which supports multiple views of a user's program. These views include a syntax directed editor, a Nassi-Schneiderman flowchart, a module interconnection diagram of how the program is organised, and stack data view showing the current state of the data stack.

  Other examples of visual environments include Pict, PIGS and Xerox Star System. The effect of COBIE being extended to a visual environment based on the above examples will be examined.

Various monitoring mechanisms similar to those described by Barstow et al [BARS84] to keep track of statistics regarding the utilization of the facilities provided by COBIE at each stage in this phase will be built into the system.

Cobol Integrated Environment (COBIE)

The integrated environment developed to meet the objectives of Phase 1 described above was created using Turbo Vision and is similar to the Integrated Development Environment (IDE) provided by Turbo Pascal.

System Specifications

COBIE, like RMCOBOL, is a DOS based system which can be run on any IBM compatible microcomputer.

COBIE Interface

The COBIE interface consists of three components namely a pull down menu, a status bar and a desktop.

According to Kay [GLIN90] multiple windows that have the following characteristics:

- displays associated with several user tasks which could be viewed simultaneously;
- switching between tasks must be done with the press of a button;
- no information will be lost in the process of switching; and
- screen space would be used economically;

formed the basis of an "integrated environment. COBIE provides users with this multiple window facility in which to implement the necessary tasks required.

Facilities Offered by COBIE

The pull down menu which forms part of the COBIE interface is comprised of three submenus namely **File**, **Cobol**, and **View**.

The **File** submenu provides the user with the option of creating new source and data files or editing existing files, and saving these files.

The **Cobol** submenu provides the user with a **Compile** and **Run** option. Upon choosing the **Compile** option the user is prompted to confirm the name of the file to be compiled. The RMCOBOL compiler is then evoked. Thereafter the user's screen is divided into two windows:

- one containing the source program; this window contains a line number indicator to assist the user find any errors indicated in the compiler listing;
- one containing the listing produced by the compiler.

The user can switch between the two windows and hence correct errors specified in the compiler listing in the source code. The **Run** option results in the RMCOBOL program **RUNCOBOL** being implemented. Upon choosing the **Run** option the user is required to confirm the name of the COBOL file to be run.

The **View** submenu provides the user with the options of viewing a particular compiler listing or a report file that has been created as the result of running a specific COBOL program.

Evaluation of the COBIE System

"Hands-on" sessions with COBIE were held during which students were required to firstly make minor changes to two COBOL programs and then compile and run these programs. In order to evaluate the COBIE environment at the end of these sessions students were issued with questionnaire. Eighteen students of the twenty two enrolled for the course completed and returned the questionnaire. This questionnaire was designed based on that used by Glinert et al [GLIN90] in their student evaluation of the Pict system. A list of the questions contained in the questionnaire are listed in **Table 1**.

Table 1 : Evaluation Questionnaire

1.	Have you found the program easy to use? Explain
2.	Could you easily remember the steps that were required to carry the necessary tasks?
3.	Did you find the program fun to use?
4.	Would you like to use the program again?
5.	What improvements do you think should be made to the program?
6.	List any problems that you have experienced.
7.	Describe **your** experience of using the program.
8.	Any other comments

To obtain a measure of the effect of students' learning styles prior to the use of the system a survey was conducted using Kolb's Inventory to determine the learning styles of these students. Nineteen of the twenty two students enrolled for the course completed and returned the questionnaires.

In order to obtain a correlation between a student's learning style and his or her experience with COBIE only the feedback of those students that responded to both questionnaires was examined. Sixteen students responded to both questionnaires of which nine were female and seven male.

Responses to questionnaire

Table 2 lists the responses to the first four questions:

Table 2: Responses to questionnaire

QUESTION NUMBER	YES	NO	UNSURE	NO RESPONSE
1	9	7	-	-
2	12	2	2	-
3	10	5	-	1
4	15	-	-	1

From **Table 2** it is clear that a majority of the students found the COBIE system easy to use. Comments made by students included:

"The program eliminated time wasting and irritating procedures."

"Steps were not difficult."

"Easy to grasp, understand and remember."

However seven students did not find the program easy to use. Comments made by these students included "as time goes on it will be easier"; "Not easy - because it was my first time to use it".
This is consistent with responses obtained by Glinert et al [GLIN90] in the evaluation of the Pict system. Students attributed the lack of ease of use of the Pict system to the fact that it was the first time that they were using the system.

It is evident from **Table 2** that again a majority of students found that they could easily remember the steps that were required to edit, compile and run COBOL programs using COBIE.

All students with the exception of one student who did not respond to this question indicated that they definitely wanted to use the COBIE system again.

Two problems were listed by students:

1. Firstly due to COBIE being used in a network environment students were prompted to "Insert a disk in drive A:" four times during the compilation process. This made compiling of a program rather tedious and frustrating.

2. Secondly if a compiler listing of a source program could not be created a blank window was displayed on the screen by COBIE and this window could not be "closed".

Improvements mentioned by students were based on these two problems experienced.

Comments describing the students experience with COBIE included:

"It was most educational."

" I found it interesting and fun to use. The steps were easy to remember, however I experienced some difficulties when trying to compile my program."

"Challenging"

"At the moment the program is fine because I understand what we've done at this point in time."

Only five students provided any additional comments. This comments emphasised that students required more practical sessions with COBIE.

The Effect of Learning Styles on the ease of use of COBIE

Finnie [FINN90] defines four learning styles namely, Converger, Diverger, Assimilator and Accommodator as derived by Kolb. From the study conducted it was found that students who responded positively to the questions "Have you found the program easy to use?" and "Could you easily remember the steps that were required to carry out the necessary tasks?" predominately possessed a the Converger or Accommodator learning style. Those students that responded negatively to these questions predominately possessed either a Diverger or Assimilator learning style. Hence it is evident that students possessing a Converger or Accomodator learning style experience less difficulties in using the integrated COBOL environment compared to students possessing a Diverger or Assimilator learning style.

Conclusion

From the study conducted it is evident that the COBIE system has helped novice programmers overcome some of the difficulties experienced with using the RMCOBOL system. Certain improvements have been made to COBIE based on the feedback obtained from. students. Currently the COBIE system is being used by first year students to write COBOL programs.

COBIE will be extended further to provide a debugging and documentation system. Finally a number of mechanisms (as outlined above) such as the use of syntactical templates and visual programming environments will be examined in order to obtain a means by which these novice programmers can be assisted to program in COBOL.

References

[BARS84] - Interactive Programming Environments, David R. Barston, Howard E. Shrobe, Erik Sandewill, Mcgraw Hill, 1984.

[DONA85] - Integration Techniques in Cedar, James Donahue, Sigplan Notices, Vol 20, No 7, July 1985, Pages 265 - 286.

[FINN90] - On Learning Styles and Novice Computer Use, G.R. Finnie, Quaestiones Informaticae, Vol 5., No. 1, 1990, Pages 1- 10.

[GLIN90] - Visual programming Environments - Paradigms and Systems, E.P. Glinert, IEEE Computer Society Press, 1990.

[SCHR93] - An Integrated Prolog Programming Environment, U. Schreiweis, A Keune, H. Langendorfer, ACM Sigplan Notices, Vol. 28, Part 2, 1993.

[SWEE85] - The Mesa Programming Environment, R.E. Sweet, Sigplan Notices, Vol. 20, No.7, July 1985, Pages 216-229.