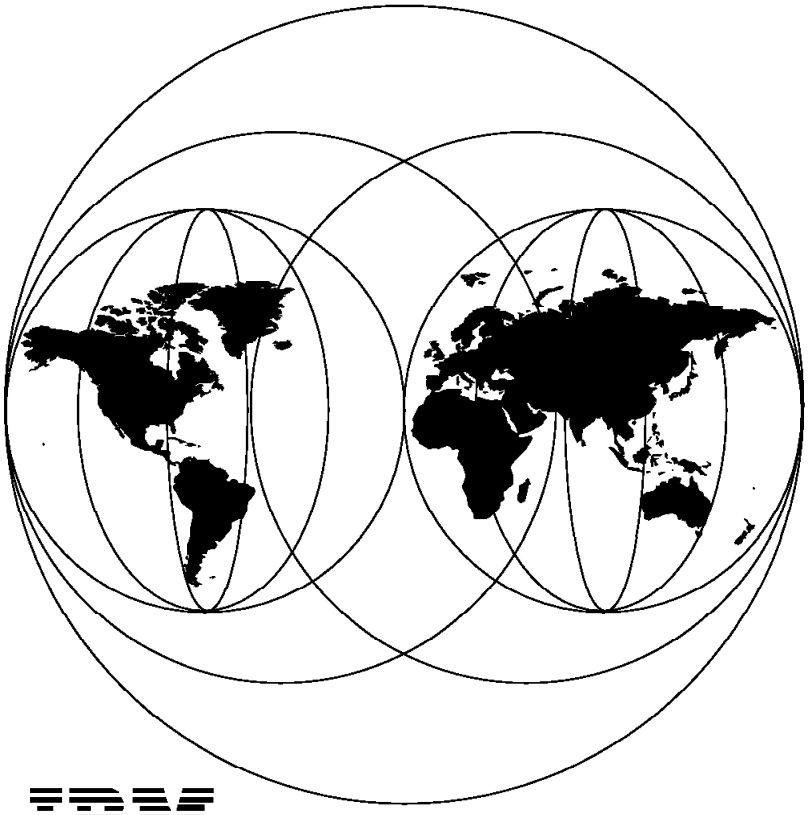
SG24-4277-00

IBM COBOL and Language Environment for VSE/ESA How to Upgrade Now

July 1997



IBM
International Technical Support Organization
Boeblingen Center

International Technical Support Organization

SG24-4277-00

IBM COBOL and Language Environment for VSE/ESA How to Upgrade Now

July 1997

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special Notices” on page 141.

First Edition (July 1997)

This edition applies to Version 1 Release 4 of the IBM Language Environment for VSE/ESA (LE/VSE), program number 5686-094, and IBM COBOL for VSE/ESA Version 1 Release 1, program number 5686-068, for use with the VSE/ESA Operating System Version 2, program number 5690-VSE.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. 3222 Building 71032-02
Postfach 1380
71032 Böblingen, Germany

Contents

Figures

Tables

Preface

The strategic VSE COBOL compiler is COBOL for VSE/ESA with Full-Function Feature that includes the Debug Tool for VSE/ESA. The run-time library is Language Environment for VSE/ESA Version 1 Release 4. With these products customers can move their COBOL applications into and beyond Year 2000.

The benefits of moving to the new environment are described in this document. The detailed description of the differences between the various language levels and of available migration aids will help you perform the upgrade.

This redbook has been written for all professionals with the responsibility for planning and executing the migration of DOS/VS COBOL or VS COBOL II applications to COBOL/VSE and the run-time environment of LE/VSE.

Good knowledge of VSE/ESA concepts and HLL programming is required.

The Team that Wrote this Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Böblingen Center.

Annegret Ackel from the International Technical Support Organization Böblingen Center, was the project leader.

Hiroshi Arai from IBM Japan.

Carey Fu from IBM China.

Omar Qureshi from IBM Pakistan.

Special thanks to **Mike Moriarty**, ISSC Australia, for his advice and guidance provided in the production of this document.

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 173 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web Sites:

  For Internet users `http://www.redbooks.ibm.com`
  For IBM Intranet users `http://w3.itso.ibm.com`

- Send us a note at the following address:

  `redbook@vnet.ibm.com`

Chapter 1. Introduction

Migration to IBM Language Environment for VSE/ESA (LE/VSE) and COBOL for VSE/ESA (COBOL/VSE) is a requirement for all customers, since there are no future enhancements planned for DOS/VS COBOL and VS COBOL II. All future enhancements will be to the strategic product LE/VSE with the LE/VSE-conforming languages. In addition the migration from DOS/VS COBOL and VS COBOL II to a COBOL VSE environment is essential for exploiting the advantages of LE/VSE.

DOS/VS COBOL and VS COBOL II have already been withdrawn from marketing and it has been announced that service for VS COBOL II 1.4 will end 04/28/1998.

Therefore, it is in the best interests of the customers to migrate their applications and take advantage of the increased functions and ease of maintenance that LE/VSE and the new compiler contain, as soon as possible.

The purpose of this book is to give the reader a basic idea of the benefits of migration and also to give exposure to migration tools such as CCCA/VSE (COBOL & CICS Command Level Conversion Aid for VSE), COBOL/SF (COBOL Structuring Facility), and DT/VSE (Debug Tool for VSE/ESA). The language differences will not be described in detail, only specific incompatibilities will be explained so that the reader can also use this book during migration.

1.1 The Year 2000 Aspect

The basic Year 2000 problem is the 2-digit year data representation. At the turn of the century the current year will be less than the previous year because the date field only contains the last two digits of the year. This requires that the applications are migrated to a programming language that supports a full 4-digit year format.

The redbook *Preparing your VSE System for the Year 2000* describes how to overcome the Year 2000 problem with short-term and long-term solutions (together with LE/VSE) pertaining to programs written in high level languages. For COBOL users, the conclusions are as follows:

1. When programs use a 2-digit year for calculation, comparison will encounter logical mistakes after the year 2000.
2. The accurate calculation of a year requires current date information containing a 4-digit year.
3. There are theoretically three solutions for the Year 2000 problem.
   - Change all years to four digits .. the long-term solution
   - Use two digits with conversion for calculation .. century window
   - Use two digits containing coded four digits

   The long-term solution is the best. But not everyone will be able to complete this project prior to year 2000, since this includes eventually rebuilding the databases and data files with 4-digit years. Therefore, the practical way for modification of existing applications is the solution using the century window.
4. There is no 4-digit year date support with DOS/VS COBOL or VS COBOL II. Only COBOL/VSE fully supports 4-digit year dates with the COBOL/VSE

intrinsic functions, as does its prerequisite LE/VSE with its date/time callable service routines for date conversion (sliding century window).

VS COBOL II programs may make calls to LE/VSE date/time service routines, although there will be no support internally for the year 2000. DOS/VS COBOL programs cannot use the services of LE/VSE date/time callable routines.

5. It may be possible for other COBOL (DOS/VS COBOL, VS COBOL II) users to get a 4-digit year date using an Assembler subroutine and adopt a century window approach with their own logic. However, the workload of modification and test of application source is the same as using COBOL/VSE. In addition, development and maintenance of user-written routines may require great effort.

1.2 The Benefits of LE/VSE and COBOL/VSE

The following section gives you an overview of the new functions that you can use when you migrate to LE/VSE and the new compilers. Details are described in Chapter 2, “Why Migrate?” on page 11.

1.2.1 What is LE/VSE

LE/VSE is a set of common services and language-specific routines that provide a single run-time environment for applications written in LE/VSE-conforming versions of the COBOL, PL/I and C high level languages (HLLs), and for many applications written in previous versions of COBOL. An LE/VSE-conforming language is any HLL that adheres to the LE/VSE common interface.

Table 1 lists the LE/VSE-conforming language compiler products you can use to generate applications that run with LE/VSE.

Table 1. LE/VSE-Conforming Languages

Language	LE/VSE-Conforming Language	Minimum Release
COBOL/VSE	IBM COBOL for VSE/ESA	Release 1
PL/I VSE	IBM PL/I for VSE/ESA	Release 1
C/VSE *	IBM C for VSE/ESA	Release 1

Note:

* Applications written in C/VSE can only run with LE/VSE Release 4.

Any HLL not listed in Table 1 is known as a non-LE/VSE-conforming or, alternatively, a pre-LE/VSE-conforming language. Some examples of non-LE/VSE-conforming languages are C/370, DOS/VS COBOL, VS COBOL II, and DOS PL/I.

Only the following products can generate applications that run with LE/VSE:

- LE/VSE-conforming languages
- High Level Assembler (HLASM)

  LE/VSE also supports applications written in assembler language using LE/VSE-provided macros and assembled using HLASM.
- DOS/VS COBOL and VS COBOL II, with some restrictions

Although DOS/VS COBOL and VS COBOL II are non-LE/VSE-conforming languages, many applications generated with these compilers can run with LE/VSE without recompiling or relink-editing. For details see Chapter 3, “Migration Considerations” on page 19.

LE/VSE is the prerequisite run-time environment for applications generated with COBOL/VSE, PL/I VSE or C/VSE. LE/VSE does not include compilers, whereas COBOL/VSE, PL/I VSE and C/VSE are compilers only, they do not include a run-time environment, as the languages DOS/VS COBOL and VS COBOL II did before.

Release 1 and Release 4 of the LE/VSE product are based on Release 2 and Release 4 of the LE/370 product respectively (now called IBM Language Environment for MVS and VM).
Therefore, the basic architecture is the same across the platforms and it is possible (in many cases) for programs written in LE-conforming languages to be easily adapted to run on an alternate platform.

1.2.2 What You Can Do with LE/VSE

LE/VSE is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services, common math functions, application utilities, and system services. LE/VSE enables existing applications to function as before with few, if any, changes required, thus helping preserve company investment in those applications.

In a single product LE/VSE combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With LE/VSE, application programmers can use one run-time environment for their applications, regardless of the programming language or system resources.

1.2.3 COBOL/VSE

COBOL/VSE is the newest VSE COBOL compiler and is a direct descendent from VS COBOL II and DOS/VS COBOL. The major differences between the compilers are:

- DOS/VS COBOL supports ANSI 68 and ANSI 74 Standard
- VS COBOL II supports ANSI 85 Standard
- COBOL/VSE supports ANSI 85 Standard with the ANSI 85 Addendum (Intrinsic Functions)

The fundamental change from DOS VS COBOL to VS COBOL II and COBOL/VSE is that source modules may be changed from unstructured source code to structured source code. This significant change means you will have no more ′spaghetti′ code which only one person can maintain because they are the only one that understands it.

COBOL/VSE has new language extensions and provides support for LE/VSE features on top of the VS COBOL II language. New with COBOL/VSE is a set of intrinsic functions that enable you to perform mathematical, statistical, financial, character string or date and time calculations with simple invocations from COBOL statements.

With LE/VSE, COBOL users can:

- Solve the Year 2000 problem by using
  - COBOL/VSE intrinsic functions
  - LE/VSE callable services
- Write applications that utilize 31-bit addressing
- Have increased control over compiler output, such as Associated Data.

1.3 Debug Tool for VSE/ESA

LE/VSE Release 4 supports the Debug Tool for VSE/ESA (DT/VSE) for debugging applications written in any LE/VSE-conforming language.

Prior to LE/VSE, each programming language provided its own separate run-time environment. LE/VSE combines essential and commonly-used run-time services - such as message handling, condition handling, storage management, date and time services, and math functions - and makes them available through a set of interfaces that are consistent across programming languages. With LE/VSE, you can use one run-time environment for your applications, regardless of the application′s programming language or system resource needs, because most system dependencies have been removed.

1.4 How to Get Started

Planning is very important in upgrading your COBOL technology and moving towards Year 2000, and planning is the key to the success of your migration. It is important to communicate with management and the users so they understand what you are planning.

The following plan items have to be developed:

- Develop a cost/benefit analysis

  This helps everyone understand what is being done and why.

- Take inventory of your COBOL development tools

  This is a good time to improve programmer productivity with enhanced tools.

- Assess requirements of new software

  Make sure you check all the prerequisite product levels, the compatibility with vendor products and IBM products.

- Take inventory of existing applications

  By taking an inventory you will get a detailed picture of the work that is required. You will also find all unused code and executables that exist in your libraries.

- Assess migration effort for each application

  Assign complexity ratings and prioritize your applications. Tables that will help you in doing this can be found in the manual *COBOL/VSE Migration Guide*.

- Schedule training for programmers

  This is very important, since there is a new compiler, new run-time library, new COBOL ANSI standard, new tools, and more.

The following charts give you an overview of the whole migration project.

1.4.1 The Overall Picture

In Figure 1 on page 6 you will find the overall picture of the migration project also with regard to project management.

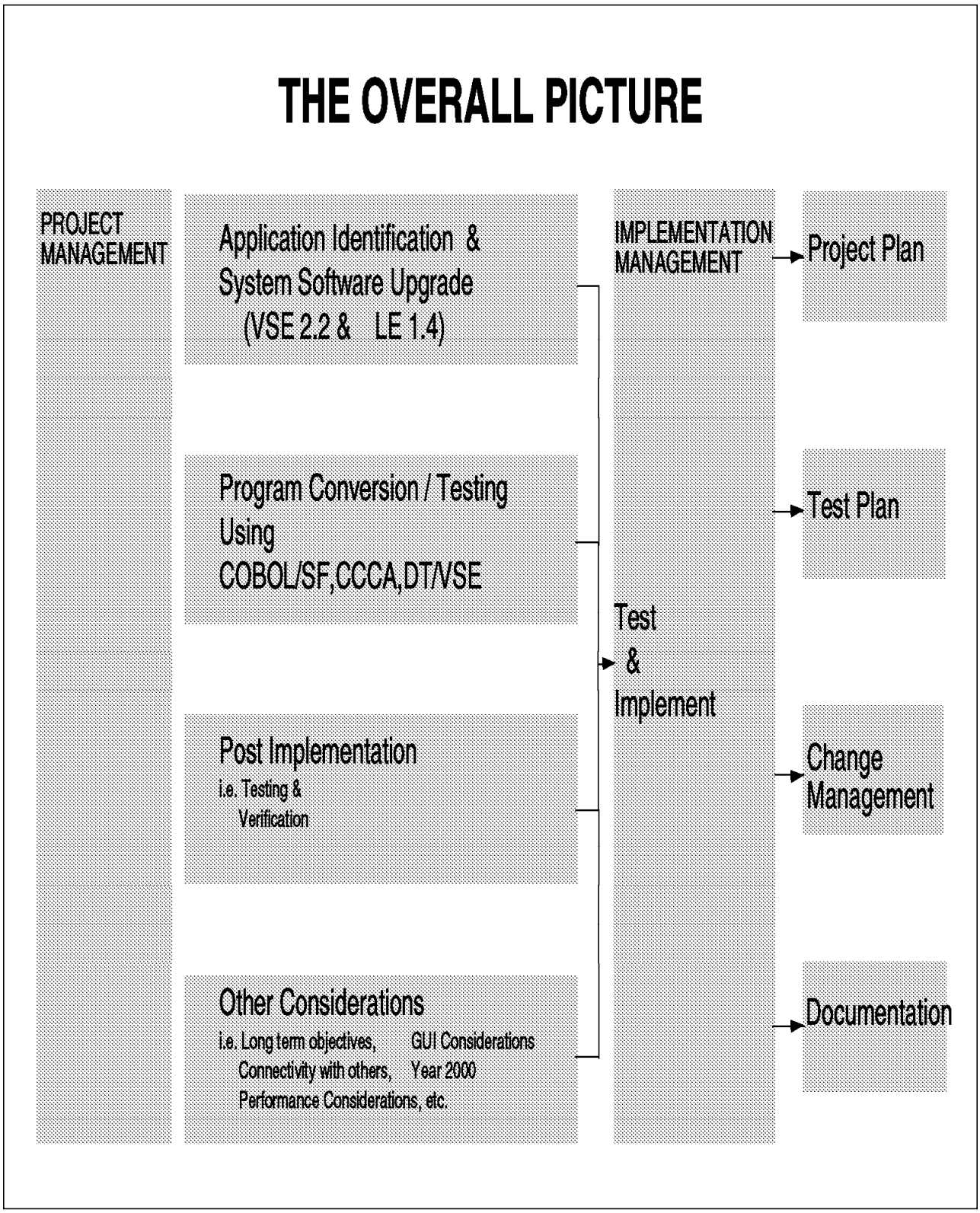


Figure 1. The Overall Picture

First of all the project management needs:

1. to give a project plan to the implementation team
2. to give a test plan to the implementation team
3. identify the change management procedure
4. define the method of documenting the changes made to the application source.

The overall picture shows that the implementation team has to identify all applications as well as the system software level.

The conversion/testing has to be done using migration aids such as COBOL/SF, CCCA and DT/VSE. In the post implementation stage, testing and verification has to be done to check whether the migration was successful.

The step ′other considerations′ covers what the customer considers are the important jobs to be done at the same time as the migration. These are usually long term objectives the company might have and include adding new modules of software, Year 2000 support, expanding hardware, considering GUI (Graphical Users Interface), and improving performance.

1.4.2 The Migration Process

The migration process as shown in Figure 2 gives you a picture of what has to be done and the sequence of performing the different jobs.

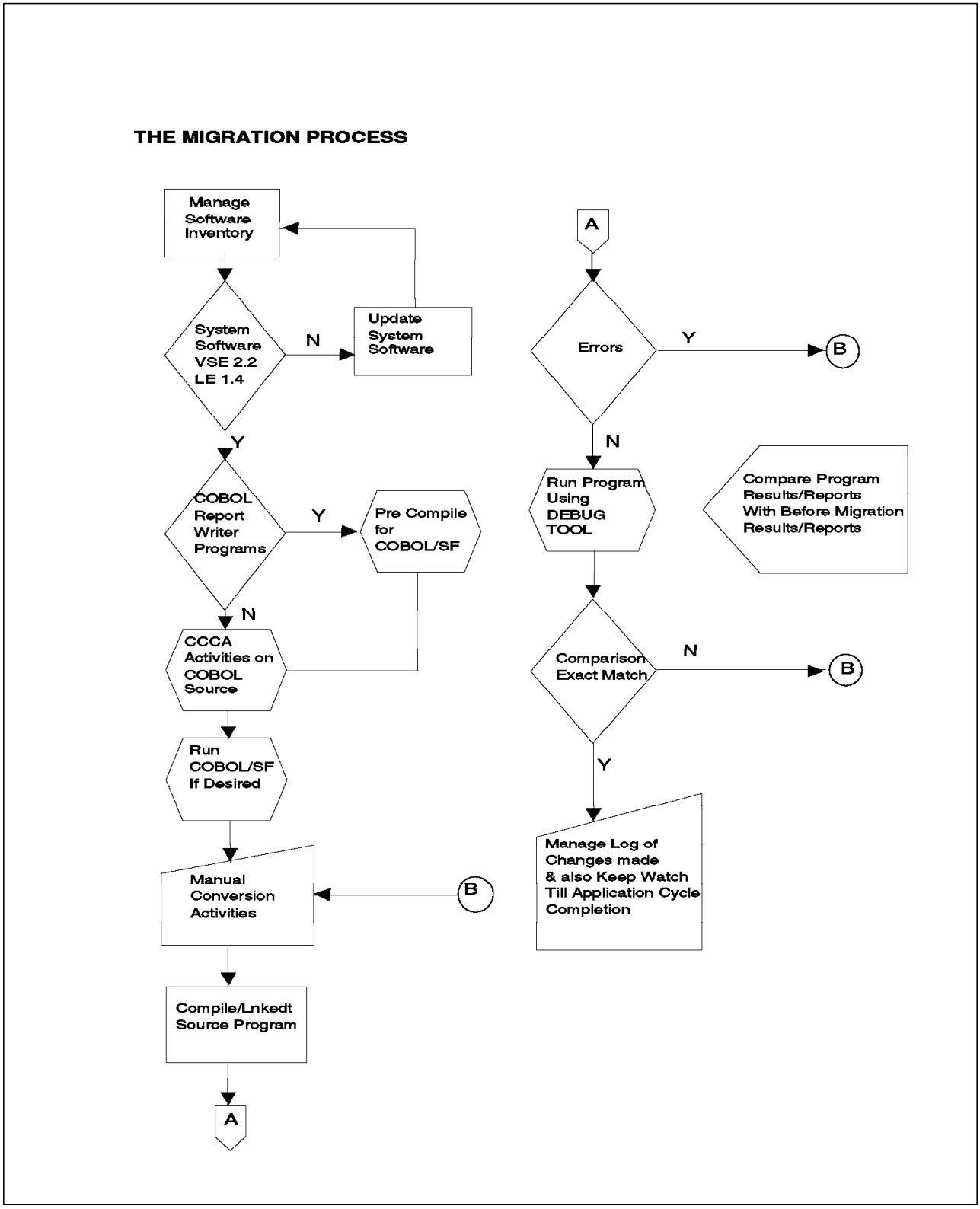


Figure 2. The Migration Process

First the application identification has to be done, to find out which are the latest error-free versions. Based on this, an inventory of the applications is done to get an idea of the size of the project.

As system software you need at least VSE/ESA 1.4.3 or VSE/ESA 2.2 and LE/VSE R4. If the release levels are less than this, they should be updated.

If there are Report Writer programs in your inventory, then they need to be converted through the COBOL Report Writer precompiler, so that COBOL/VSE can recognize the commands. Also CCCA requires that the Report Writer commands are changed to normal COBOL statements.

On the changed source code CCCA (COBOL & CICS Command Level Conversion Aid) can perform its activities to change/flag/eliminate old COBOL features that have been dropped.

You can run COBOL/SF (Structuring Facility) to structure the source code if you want to increase programmer productivity and save time in changing/modifying source code.

After all these activities you have to manually check the flagged statements for manual update. After compile and link-edit, if there are no errors, run the DT/VSE (Debug Tool) and check the before Results/Reports with the after conversion Results/Reports.

If the comparison is exact then ensure that the changes made are logged and observe the results carefully till the application cycle is completed.

Chapter 2. Why Migrate?

Migration to LE/VSE and COBOL/VSE is strongly recommended, since there are no future enhancements planned for DOS/VS COBOL and VS COBOL II. With applications written in these old languages your system will not be ready for the Year 2000.

2.1 DOS/VS COBOL to COBOL/VSE

The source migration effort from DOS/VS COBOL to COBOL/ESA is the same as the source migration effort to VS COBOL II. The COBOL/VSE compiler supports the same syntax as VS COBOL II and provides enhanced features.

The fundamental change from DOS/VS COBOL is that COBOL/VSE (as VS COBOL II) provides structured programming constructs. Using these constructs in the development of applications will ease the maintenance of source code and make it less costly.

LE/VSE provides the same run-time support for existing DOS/VS COBOL programs as does VS COBOL II. There may be some cases in which a few changes are required where the mixed language load modules will have to be re-linked.

For details see the manual *COBOL/VSE Migration Guide*.

2.2 VS COBOL II to COBOL/VSE

Migration from VS COBOL II (NOCMPR2 compiler option) to COBOL/VSE is a recompile effort, no source changes are required. For a few new reserved words some minor changes will be required.

LE/VSE provides the same run-time support for existing VS COBOL II programs. There are a few cases where changes may be needed, particularly for mixed language load modules which will require re-linking of the affected modules.

For details see the manual *COBOL/VSE Migration Guide*.

2.3 What Can be Achieved with LE/VSE

LE/VSE helps you create mixed-language applications and gives you a consistent method of accessing common, frequently used services. LE/VSE promotes efficient application development by providing enhanced capabilities to:

- Properly handle 2-digit years in the Year 2000 and beyond.
- Mix legacy code with new code.
- Debug applications interactively (with LE/VSE Release 4 only).
- Manage storage dynamically.
- Perform date and time calculations.
- Access an extensive set of math services.
- Share common run-time services.

- Handle conditions consistently.
- Perform interlanguage communication more efficiently.
- Customize routines for international requirements (LE/VSE Rel 4).

2.3.1 Properly Handle 2-digit Years in the Year 2000 and Beyond

With LE/VSE′s date and time services you can use existing date formats for the Year 2000 and beyond. LE/VSE employs a sliding scheme called a century window. When LE/VSE performs a date function, it determines the century of the 2-digit year by using this window and, optionally returns a 4-digit year. This means that you can write your applications with the Year 2000 in mind, without having to change the way dates are stored in your existing databases, or you can modify the dates stored in your existing databases by using the 4-digit years provided by LE/VSE date and time services.

2.3.2 Mix Legacy Code with New Code

LE/VSE protects your investment in your applications by minimizing disruption to your system and programming resources as you migrate your current applications, with certain exceptions such as:

- COBOL 85 Standard interpretation changes
- Reserved Words changes

These are described in more detail in 5.4, “Differences from VS COBOL II Using NOCMPR2” on page 38. For details to correct these problems, please refer to Modifying your VS COBOL II Source Programs in the *COBOL/VSE Migration Guide*.

This upward compatibility means that you can produce new applications largely from older object modules and phases. It also means that you do not have to recompile and relink-edit the whole application and only that part needs to be compiled which has been changed.

If your applications contain DOS PL/I programs, you must recompile them using the PL/I VSE compiler. There are certain exceptions such as COUNT, NOCOUNT, FLOW and NOFLOW which are not applicable and other reserved words which have had their names changed such as ISASIZE to STACK, REPORT to RPTSTG(ON), NOREPORT to RPTSTG(OFF), STAE to TRAP(ON) and NOSTAE to TRAP(OFF). DOS PL/I programs are source-level compatible. DOS PL/I programs are source-level compatible with PL/I VSE and can be compiled with the PL/I VSE compiler without change.

If your applications contain C/370 programs, you must recompile them using the C/VSE compiler. C/370 programs are source level compatible with C/VSE and can be compiled with the C/VSE compiler without change.

Routines that depend on dump formats, error message formats, error handling routines, Assembler routines and the like, may have to be changed.

2.3.3 Debug Applications Interactively

LE/VSE provides a common debugging interface for all LE/VSE-conforming languages. With the Debug Tool for VSE/ESA you can interactively:

- View a program listing while debugging.
- Step through execution code.
- Set dynamic break points.
- Track variables.
- Modify program and variable storage during debug.
- Debug mixed-language applications.
- Develop testing scripts for regression testing.

2.3.4 Manage Storage Dynamically

Common storage management services are provided for all LE/VSE conforming languages. LE/VSE controls the storage your routines use at run-time, removing the need for each language to maintain a unique storage manager and avoiding the incompatibilities between different storage mechanisms. Storage management support provides:

- a common heap for all conforming languages
- run-time options for storage tuning
- multiple heap support
- a storage reporting facility
- callable services to dynamically create, allocate, free, and discard heap storage

LE/VSE storage services supplement and cooperate with the existing C malloc() and free() functions, and with the PL/I ALLOCATE and FREE statements. You can also use the storage services to allocate and free storage dynamically for your COBOL routines, eliminating the need for Assembler.

2.3.5 Perform Date and Time Calculations

With LE/VSE′s date and time services you can:

- format date and time values according to country or custom′s formats
- parse date and time values
- convert values between Gregorian, Julian, Asian, Lilian calendar formats
- calculate days between dates
- calculate elapsed time

All LE/VSE date and time services conform to national language support guidelines, including full DBCS support.

2.3.6 Access an Extensive Set of Mathematical Services

LE/VSE′s math services are fast, accurate, and accommodate a wide range of data types. With LE/VSE, you can access a rich set of math services from COBOL, PL/I, C, and Assembler routines through a common interface, and thus avoid potentially conflicting results from language specific math services.

2.3.7 Share Common Run-time Services

Before LE/VSE, the language-specific run-time libraries provided many similar services, but they were managed independently and were not always consistent across languages.

LE/VSE consolidates essential run-time services for initialization, termination, message handling, condition handling, storage management, national language support, math calculations, and other common programming tasks into a single run-time environment. These services are available, through a common calling interface, to applications produced with LE/VSE-conforming language compilers.

2.3.8 Handle Conditions Consistently

One of LE/VSE′s most significant benefits is the way it handles conditions. LE/VSE establishes a consistent condition handling method for high-level languages and Assembler language routines that adhere to LE/VSE protocols. LE/VSE also gives you the flexibility to create your own condition handling routines, so you can deal with conditions as you wish.

LE/VSE′s condition handling honors both single and mixed-language applications, and is integrated with run-time message handling services to provide you with specific information about each condition.

When an application abends, LE/VSE provides a common dump environment where you can find, in one place, traceback information, the contents of program variables, control block information, error condition information, and program status data for all languages used in the application. This means that you need less language-specific knowledge to understand a run-time dump.

For COBOL applications, the Language Environment condition handling is a natural extension of the language that permits existing applications to function as before, or take advantage of the additional features.

2.3.9 Perform Interlanguage Communication More Efficiently

With LE/VSE, routines call one another within one common run-time environment. This makes ILC in mixed-language applications easier and more efficient.

LE/VSE eliminates incompatibilities among language-specific run-time environments, removing the need for initialization and termination of a language-specific run-time environment. This single run-time environment makes interlanguage communication in mixed-language applications easier, more efficient and more consistent.

This ILC capability also means that you can share and reuse code easily. For instance, you can write a service routine in the language of your choice COBOL, PL/I, C or Assembler - and allow that routine to be called from COBOL, PL/I, C or Assembler applications. Reusing already developed code instead of

repeating the function in more than one language saves application development time, testing, and maintenance costs, and improves the quality of the code.

2.3.10 Customize Routines for International Requirements

LE/VSE provides a set of data files, called locales, which define coded character sets to reflect the different specific requirements of users of various countries. From your COBOL, PL/I or C routines you can access these pre-defined locales at run time through a set of locale callable services. With locale callable services, application developers can build programs that can be marketed globally, and still meet the end user′s need to work with specific languages, cultures, and convention. You can also create your own locales, or modify an IBM supplied locale using the locale definition utility supplied with LE/VSE.

2.4 The Benefits of COBOL/VSE in LE/VSE

COBOL/VSE is the newest VSE COBOL compiler and is the recommended target compiler for your COBOL II and DOS/VS COBOL applications.

COBOL/VSE has new language extensions and provides support for LE/VSE features on top of the VS COBOL II language. New with COBOL/VSE is a set of intrinsic functions that enable you to perform mathematical, statistical, financial, character string or date and time calculations with simple invocations from COBOL statements. With LE/VSE, COBOL users can:

- Utilize COBOL/VSE intrinsic functions
- Write applications that utilize 31-bit addressing
- Use structured programming language constructs
- Allow connectivity to other products
- Use a wide variety of data types and character sets
- Have increased control over compiler output, such as Associated Data

Some of the important COBOL features are described below.

2.4.1 COBOL/VSE Intrinsic Functions

COBOL/VSE offers a set of functions, that you can invoke to provide values at execution time. These values require complex calculations, such as the calculation of present value or the average of a large number of values. Intrinsic functions and the ALL subscript are a powerful combination, making it possible to reduce the code needed for applications that require mathematical, statistical, financial and time calculations.

2.4.2 Structured Programming

The COBOL 85 Standard supported by COBOL/VSE, provides efficient language constructs. These constructs include nested programs, in-line PERFORM statements, nested statements using explicit scope terminators, and the EVALUATE statement. Using these constructs will aid in the development of applications that conform to top-down design, modular program development, and structured programming concepts.

2.4.3 Using Other Products from COBOL/VSE Programs

The COBOL/VSE language makes it easy to use the services of other products. Among the products that can be used from COBOL/VSE are:

- Customer Information Control System (CICS)
- DOS/VS DL/I
- DFSORT/VSE
- Sort/Merge II
- Structured Query Language/Data System (SQL/DS)
- IBM Language Environment for VSE/ESA (LE/VSE)

2.4.4 Support for Reentrancy

LE/VSE supports programs that are reentrant. If a reentrant program (or a reentrant subroutine) is placed in a shared area of virtual storage, a single copy of the program will satisfy all requests for the program, even simultaneous requests. With the improved ILC of LE/VSE, reentrant applications can now include both PL/I and COBOL subprograms.

2.4.5 COBOL/VSE Programs and LE/VSE

LE/VSE, the common run-time environment for COBOL/VSE programs, provides over 80 services that you can call directly from COBOL/VSE programs, using the CALL statement.

For LE/VSE callable services that require the address of a procedure to be passed as an argument, COBOL/VSE adds new extensions for the USAGE clause that a data item is being used as a procedure pointer; that is, it contains the address of a procedure entry point to the data item.

2.4.6 Double-Byte Character Set

COBOL/VSE accepts characters in the Double-Byte Character Set (DBCS) and the standard COBOL set of characters. DBCS support makes it easier to develop COBOL applications that require a DBCS character set - such as applications using Kanji data.

2.4.7 Advanced Compiler Features

COBOL/VSE is a full-function compiler that can:

- Assist in migration, compatibility, and conformance to standards
- Produce easy-to-use listing
- Generate code that is set up for debugging
- Optimize generated code
- Provide flexible numeric sign processing
- Help you manage storage
- Offer performance improvements in sorting

2.5 Debug Tool for VSE/ESA

Debug Tool for VSE/ESA (DT/VSE) is a source-level debugger for programs that were compiled under High Level Language compilers that support LE/VSE. DT/VSE is a program-testing and analysis aid that allows users to examine, monitor, and control the execution of their programs. Programs can be dynamically patched to overcome execution failures.

Debugging sessions may be performed in either interactive or batch mode.

The debug capability can be particularly valuable in the context of the Year 2000 challenge. You are likely to find yourself making substantial changes to a significant portion of the application portfolio. These debugging facilities can help you make the testing of those changes productive and comprehensive. Thus DT/VSE can help reduce the total cost and risk of the project.

DT/VSE is supplied as an optional feature with COBOL/VSE, PL/I VSE and C/VSE. If you want to obtain DT/VSE you will only need to order the feature via one of the compilers. DT/VSE will work with all supported LE-enabled programs irrespective of the compiler with which it was ordered.

Chapter 3. Migration Considerations

3.1 Planning for Migration

Depending on your site's situation you will most likely have to plan for the following conversion tasks:

- Upgrade your base operating system

  To take advantage of COBOL/VSE and LE/VSE Release 4 your operating system has to be upgraded to VSE/ESA 2.2 or VSE/ESA 1.4.3.

- Move your run-time environment to LE/VSE

  A run-time migration consists of link-editing the application program object module with the LE/VSE library modules to produce a program phase. The run-time environment is then changed to access the LE/VSE library modules.

- Upgrade your source to COBOL/VSE

  A source migration consists of updating the application program source if necessary, and compiling the program with an LE/VSE-conforming compiler.

If you already have LE/VSE Release 1 installed, then a migration to LE/VSE Release 4 should also be planned. For details see 3.4, "LE/VSE Migration from Release 1 to Release 4" on page 25.

3.2 Migration Scenarios

The COBOL environment is complex due to the number of supported products and the sharing of module and phase names.

The supported products are:

- DOS/VS COBOL 1.3.1
- VS COBOL II
- COBOL/VSE
- LE/VSE

The following table shows the valid combinations of COBOL products and the potential conflicts of module and phase names when multiple products are concurrently installed.

Table 2 (Page 1 of 2). Valid Scenarios for COBOL Compiler, Link-Edit, and Run Time

If you compile with this compiler...	And you link-edit with this run-time library...	Then you can run with this run-time library...
DOS/VS COBOL FCOB	DOS/VS COBOL ILB	DOS/VS COBOL *
		VS COBOL II *
		LE/VSE *
	VS COBOL II ILB	VS COBOL II *
		LE/VSE *
	LE/VSE ILB	LE/VSE *

Table 2 (Page 2 of 2). Valid Scenarios for COBOL Compiler, Link-Edit, and Run Time		
If you compile with this compiler...	**And you link-edit with this run-time library...**	**Then you can run with this run-time library...**
VS COBOL II IGY using RES compile-time option	VS COBOL II IGZ	VS COBOL II IGZ
		LE/VSE IGZ
	LE/VSE IGZ	LE/VSE IGZ
VS COBOL II IGY using NORES compile-time option	VS COBOL II IGZ	Not required **
	LE/VSE IGZ	LE/VSE IGZ
COBOL/VSE IGY	LE/VSE IGZ	LE/VSE IGZ

Note:

* For programs compiled by DOS/VS COBOL, there are a few library routines that can be used at run time, but this is application program dependent.

** Prior to APAR PN09126, the VS COBOL II run-time library was required for DTF build routines.

FCOB Prefix for DOS/VS COBOL compiler modules.

ILB Prefix for DOS/VS COBOL run-time modules.

IGY Prefix for VS COBOL II and COBOL/VSE compiler modules.

IGZ Prefix for VS COBOL II run-time modules and COBOL language component run-time modules of LE/VSE.

Table 2 on page 19 shows that for COBOL:

- There is a possibility of module name conflict when old and new products are installed concurrently.
- It is possible to use new LE/VSE run-time programs with older COBOL-compiled programs before implementing the COBOL/VSE compiler.

If you are a COBOL user, there are choices available so you must decide on a migration strategy. LE/VSE provides object compatibility for programs compiled with either DOS/VS COBOL or VS COBOL II. Therefore, in most cases it is possible to migrate the run-time component first and then gradually introduce the new LE-conforming compiler.

The migration scenarios for COBOL are:

- Source
  - DOS/VS COBOL 1.3.1 to COBOL/VSE
  - VS COBOL II (CMPR2) to COBOL/VSE
  - VS COBOL II (NOCMPR2) to COBOL/VSE
- Run-time
  - DOS/VS COBOL 1.3.1 to LE/VSE
  - VS COBOL II to LE/VSE
  - LE/VSE Release 1 to LE/VSE Release 4

Recompile and/or relink?

- When must I recompile?
  * For COBOL, it depends on whether source code has been amended to conform with COBOL/VSE requirements or if LE/VSE callable services are being introduced into the program.
- When must I relink?
  * Always if recompile has been done.
  * If no recompile is done for COBOL, it depends....., but highly recommended.
  * Few customers retain object modules, so usually recompile is done.

3.2.1 Ability to Combine Old and New Compile Units

When upgrading your applications to the new COBOL technology, you should think of **Selective Conversion**. If you have an application that consists of multiple programs in DOS/VS COBOL and/or VS COBOL II, you can upgrade a program to take advantage of some new LE/VSE features or even the COBOL/VSE intrinsic functions. Then you have to compile the program with COBOL/VSE and relink it with your old DOS/VS COBOL and/or VS COBOL II programs. Now your application can run with LE/VSE and the execution of your DOS/VS COBOL and VS COBOL II programs is supported. You can then upgrade the other programs when you open them for maintenance.

Recommendation

Where possible do a run-time migration to LE/VSE first, then carefully plan and implement a source migration. New development can be done with the COBOL/VSE compiler.

3.2.2 Run-time Migration

Run-time migration is required to:

- Position to prepare for source migration in ″compatibility mode″

  If you are able to do a run-time migration first then you can prepare for a source migration by implementing the new LE/VSE run-time components in compatibility mode.

  Compatibility mode

  ″Compatibility mode″ means running old programs compiled with pre-LE/VSE-conforming languages and executed using the LE/VSE run-time library. LE/VSE provides compatibility mode for both DOS/VS COBOL and VS COBOL II programs.

- Implement source migration to LE/VSE-conforming compiler

  To run any program compiled with an LE/VSE-conforming compiler, you must use LE/VSE run-time libraries.

Recommendation

SVA usage - although there will be recommendations to place LE/VSE phases in the SVA, consider these very carefully during the migration period. This is because there may be name conflicts with existing products which could result in accessing an incorrect copy of a phase.
During the migration period try to avoid placing phases in the SVA for products that have conflicting phase names.

The potential complexity of the COBOL run-time environment is shown in Table 2 on page 19.
By implication, you must ensure that the correct version of a module is accessed by:

- Having the correct LIBDEF order
- Avoiding use of the SVA if possible during migration
- Being aware of COBPACK contents

3.2.2.1 COBOL Run-time Migration Hints and Tips

- Access the correct modules

  LE/VSE COBOL has modules with names common to both VS COBOL II and DOS/VS COBOL. Also, some LE/VSE COBOL batch modules have the same names as LE/VSE COBOL CICS modules.
  It is important to ensure that the correct module is accessed.
  Unless overridden, by default LE/VSE modules reside in VSE library PRD2.SCEEBASE, except LE/VSE COBOL CICS modules which reside in PRD2.SCEECICS.

  – Migrating from DOS/VS COBOL

    Batch PRD2.SCEEBASE before PRD2.SCEECICS before PRD2.PROD

    CICS PRD2.SCEECICS before PRD2.SCEEBASE before PRD2.PROD

  – Migrating from VS COBOL II

    Batch PRD2.SCEEBASE before PRD2.SCEECICS before PRD2.DBASE and PRD2.CICSR

    CICS PRD2.SCEECICS before PRD2.SCEEBASE before PRD2.CICSR and PRD2.DBASE

- Tape or Disk Manager - see your OEM Vendor for required fixes

  If your installation has a Third-Party Tape or Disk Manager such as CA-DYNAM or CA-EPIC, then contact the Vendor for appropriate fixes for LE/VSE.

- ALL31(ON) default in CICS must be OFF to run AMODE 24 applications

- Assess suitability of both batch and CICS run-time options

  There are different LE/VSE supplied default options for batch and CICS. These should be examined and adjusted for suitability in your environment.

3.2.3 Source Migration

Source migration consists of amending the source if necessary and recompiling the program. If you have programs that run using CICS, translate the programs, using the CICS/VSE-supplied translator.

A source migration is required to:

- Make changes because of language syntax differences
- Add code to take advantage of new language functionality
- Make changes to provide for Year 2000 support
- Correct poor programming techniques
- Implement LE/VSE-conforming compiler

Your target environment should be source that can be compiled with COBOL/VSE and the NOCMPR2 compile option. If you have previously migrated to VS COBOL II and NOCMPR2 then your source will not need to be changed.

NOCMPR2

The NOCMPR2 compiler option provides the full ANSI 85 implementation of COBOL/VSE and VS COBOL II.
The CMPR2 compiler option provides compatibility with VS COBOL II Release 2 and is provided as an aid to migration (along with the FLAGMIG option). New COBOL/VSE language elements, such as intrinsic functions, are not supported under CMPR2.

Migrating from all other environments will require varying degrees of source changes. Consult the *COBOL for VSE/ESA Migration Guide* for specific details.

3.2.3.1 COBOL Source Migration Hints and Tips

- Change compiler name from FCOBOL to IGYCRCTL

  The COBOL/VSE compiler is invoked by executing phase IGYCRCTL. If you are migrating from DOS/VS COBOL then ensure that your JCL is changed to execute IGYCRCTL instead of FCOBOL.
  If you are converting from VS COBOL II, then no change is required.

- Modify SELECT and ASSIGN clauses

  The format of the SELECT and ASSIGN clauses has changed to a much simpler form. COBOL/VSE will in some cases allow the DOS/VS COBOL coding format but may produce unexpected results at run time due to a different interpretation of the ASSIGN clause format.

  For example, the statement in DOS/VS COBOL (for a standard label tape file):

  ```
  SELECT FILEA ASSIGN TO SYS010-UT-3420-S
  ```

  must be amended to:

  ```
  SELECT FILEA ASSIGN TO filename
  ```

  where *filename* must be a valid VSE/ESA filename that matches the name specified on the // TLBL JCL statement.
  For example: // TLBL *filename*,'Tape file label'

- Remove unsupported language elements

These are comprehensively documented in the *IBM COBOL for VSE/ESA Migration Guide*. Examples include the removal of Report Writer and ISAM support.

- RETURN-CODE special register

  COBOL/VSE (and VS COBOL II) provides a special register called RETURN-CODE that is used to pass a completion code back to the operating system. This register will be set after every COBOL/VSE CALL statement to contain the value returned by the subroutine (non-COBOL/VSE and non-VS COBOL II) in General Register 15. During program termination this value will be passed back to VSE/ESA as a return code. If the value is non-zero this can cause an unexpected completion code or at worse a VSE dump.

  To avoid this it is recommended that the statement:

  ```
  MOVE 0 TO RETURN-CODE
  ```

  is inserted before STOP RUN or GOBACK statements.

  Note: A new LE/VSE Run-Time Option (RETZERO) is being provided to assist customers migrating from DOS/VS COBOL. This will allow the RETURN-CODE to be always set to zero without modifying source code. See APARs PQ04876 and PQ04879 for PTF details.

- Include Year 2000 changes

  Although this is potentially a major exercise, you have a unique opportunity with the implementation of COBOL/VSE and LE/VSE to amend program source to be Year 2000 compliant. This may include incorporating new LE/VSE Date/Time callable services or using COBOL/VSE intrinsic functions to provide 4-digit years.

- Avoid file attribute mismatches

  DOS/VS COBOL file open processing does not comprehensively check that the file definition in your program exactly matches the file definition (as defined in the VTOC for a SAM file or a LISTCAT for a VSAM file).
  Because of ANSI 85 requirements, LE/VSE COBOL open processing carries out many detailed checks for consistency between program and actual file definition before opening the file. This can result in file open failures in LE/VSE COBOL even though no changes have been made. The best approach is to add a file status check following each OPEN statement. Then, if these subsequently indicate problems amend your program appropriately.

- Ensure ANSI85 option used for CICS translation

  Either ANSI85 or its equivalent COBOL2 must be specified in the XOPTS CICS/VSE translator options for CICS/VSE Command Level Programs.
  (The keyword COBOL2 is valid for both VS COBOL II and COBOL/VSE programs.)
  This has two main effects on the translated code. Firstly, the translated program will contain the RENT compiler option which is required by programs running with LE/VSE and CICS/VSE. Secondly, the resulting COBOL/VSE code generated by translating certain EXEC CICS statements is necessarily different for the ANSI85 option.

3.3 Source Migration Aids and Tools

If you use the available conversion aids, you will find that upgrading to COBOL/VSE can be less difficult than expected. The following conversion tools can assist in upgrading your source program to COBOL/VSE:

- DOS/VS COBOL MIGR compiler option (for DOS/VS COBOL programs)
- DOS/VS COBOL migration PTFs
- COBOL/VSE CMPR2, FLAGMIG and NOCOMPILE compiler options
- COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)
- COBOL Report Writer Precompiler
- COBOL Structuring Facility (COBOL/SF)
- Debug Tool for VSE/ESA
- VisualAge for COBOL, Professional for OS/2

For a detailed description of these tools refer to Chapter 4, “Conversion Aids” on page 29.

3.4 LE/VSE Migration from Release 1 to Release 4

3.4.1 New in LE/VSE Release 4

LE/VSE Release 4 is a major functional enhancement to LE/VSE Release 1. LE/VSE Release 4 is based on Language Environment for MVS and VM (LE 370) Release 4 (but without multitasking support). New features introduced include:

- C Support

  C language run-time support has been added for C applications compiled with the new LE/VSE-conforming C language compiler.

- Support for Debug Tool for VSE/ESA

  Support has been added for interactive and batch-mode debugging of applications using a debug tool such as Debug Tool for VSE/ESA. The TEST run-time option specifies the conditions under which DT/VSE assumes control when the user application is being initialized. For meaningful results using DT/VSE, your program must have been compiled with the compiler TEST option set.

- New and changed run-time options

  For details about new and changed run-time options see the manuals *LE/VSE Run-Time Migration Guide* and *LE/VSE Programming Reference*.

- New callable services and other functions

  For details about new callable services see the manuals *LE/VSE Run-Time Migration Guide* and *LE/VSE Programming Reference*.

To take advantage of the many new features and facilities provided by this new release you should plan for a migration to LE/VSE Release 4. In addition to this, LE/VSE Release 1 was already withdrawn from marketing in 12/96, and will be withdrawn from service in 12/97.

LE/VSE Release 4 provides general object and phase compatibility for applications that run with a previous release of LE/VSE.

Phases will run compatibly with any level of LE/VSE that is equivalent to or higher than the level used to link-edit them.

Object modules can be link-edited with any level of LE/VSE that is equivalent to or higher than the level required by the compiler that generated them.

3.4.2 Abnormal Termination Considerations

With LE/VSE Release 1, the IBM-supplied assembler user exit for CICS always requested an abend when an enclave ended with the following types of unhandled LE/VSE condition of severity 2 or greater, regardless of the setting of the ABTERMENC run-time option:

- A software-raised condition, such as the condition caused by LE/VSE if you try to run an AMODE 24 program without specifying the ALL31(OFF) and STACK(,,BELOW) run-time options.
- A user-raised condition (caused by a call to the CEESGL callable service).

When the assembler user exit requests an abend at enclave termination, LE/VSE uses an abend code provided by the exit or, if the exit does not provide an abend code, one based upon the severity of the condition that caused termination. The IBM-supplied assembler user exit for CICS does not provide an abend code, so LE/VSE uses an abend code based upon the condition severity. A severity 2 condition produces an abend code of 2000, a severity 3 condition produces an abend code of 3000 and so on.

With LE/VSE Release 4, the assembler user exit for CICS does not specifically request an abend when an enclave terminates with an unhandled condition of severity 2 or greater. Instead, the ABTERMENC run-time option in effect at the time is honored. If ABTERMENC(ABEND) is in effect during abnormal termination, the enclave is terminated with abend code 4038.

In addition, with LE/VSE Release 4, the IBM-supplied abnormal termination exit for CICS, which is driven whenever an enclave terminated abnormally, requests a CICS transaction dump with a dump code of 4039. The CICS transaction dump is produced in addition to the abnormal termination information produced by LE/VSE under the control of the TERMTHDACT run-time option. Unlike LE/VSE abnormal termination information, which is written to the CESE transient data queue, the CICS dump is written to the CICS dump data set. There was no corresponding abnormal termination exit for CICS supplied with LE/VSE Release 1.

For more information about customizing, see the *LE/VSE Installation and Customization Guide*.

3.4.3 Packaging Changes Since Release 1

The packaging of LE/VSE has changed between Release 1 and Release 4. For details about these changes and planning considerations when upgrading to Release 4 see the redbook *Taking Advantage of Language Environment for VSE/ESA*.

3.5 Considerations about Prerequisite Products for Upgrading to COBOL/VSE

The list below gives the prerequisite product levels that you need to upgrade to COBOL/VSE and migrate to LE/VSE.

Table 3. Required Product Level

Enterprise System Architecture(ESA)	VSE/ESA 1.4.3 or VSE/ESA 2.2 is required to run applications with LE/VSE 1.4.
CICS/VSE Version 2 Release 3	LE/VSE Release 1 and later require CICS/VSE Version 2 Release 3.
Vendor Product Support	Any Vendor products used by your site should be LE/VSE-enabled for the LE/VSE TRAP(ON) run-time option.

If you find that any of the above prevents you from upgrading to COBOL/VSE and LE/VSE, you can still take incremental steps to prepare for conversion when these obstacles no longer apply. For example:

- Evaluate the effort to move to LE/VSE.
- Code applications based on COBOL/VSE and LE/VSE requirements to ease a future conversion. For example, specify the RES compiler option instead of NORES. (COBOL/VSE does not support the RES/NORES compiler option, and all programs are RES-like under LE/VSE.)
- Convert all source code to COBOL 85 Standard.

3.6 Major Changes with COBOL/VSE and LE/VSE

With LE/VSE, you will find that existing applications are affected in two areas: abends after severe errors and the RES/NORES environment. A brief description of the differences and actions required to ensure compatibility follows.

3.6.1 Change Default LE/VSE Run-time Options

The DOS/VS COBOL and VS COBOL II run-time options do not control whether or not your programs with severe errors end with an abend. With DOS/VS COBOL and VS COBOL II, all severe errors result in abends.

LE/VSE′s default run-time option settings allow you to intercept and handle severe errors instead of ending the application with an abend. To ensure your application ends with an abend when there is a severe error, specify the LE/VSE ABTERMENC(ABEND) run-time option as an installation default.

3.6.2 Understand the RES Environment

COBOL/VSE does not provide the RES/NORES compiler option. All programs are RES under COBOL/VSE. For existing applications compiled with NORES, two factors apply:

- If you recompile existing programs with your current compiler, specify the RES compiler option instead of the NORES compiler option. This allows you to use some LE/VSE services after link-editing with LE/VSE and also eliminates some link-edit requirements for a future move to LE/VSE.
- Since programs compiled with RES access the library when invoked, you now need to determine whether to place LE/VSE in the permanent LIBDEF chain or temporary LIBDEF chain, depending on your site′s current configuration.

Note: The RES/NORES compiler option only applies to VS COBOL II. This option does not exist in DOS/VS COBOL, which is usually referred to as a ′NORES′ environment, because all library routines are included at link-edit time.

Chapter 4. Conversion Aids

There are several migrations tools available, which you can use for a smooth migration/conversion to COBOL/VSE. We shall discuss each in order to get an idea of their capabilities. A detailed discussion of the differences between the various language levels can be found in Chapter 5, “Significant Differences between Old COBOL and COBOL/VSE” on page 35.

Table 4. Migration Aids

Migration Tool	Description
DOS/VS COBOL MIGR Compiler option	Flags most statements that are changed in, or not supported by COBOL/VSE with NOCMPR2. (Does not flag undocumented extensions to DOS/VS COBOL)
DOS/VS COBOL Migration PTFs	Support for customers for a migration to a VSE/ESA 2.2 or VSE/ESA 1.4.3 system.
COBOL/VSE FLAGMIG, CMPR2, NOCOMPILE Compiler option	Identifies DOS/VS COBOL and VS COBOL II language elements that are incompatible with COBOL/VSE NOCMPR2.
COBOL and CICS Command Level Conversion Aid for VSE (CCCA) Program Offering 5785-CCC	Converts CICS and non-CICS source code into COBOL/VSE source code.
COBOL Report Writer Precompiler Program Offering 5798-DYR	Converts Report Writer statements into COBOL/VSE source code or runs as a precompiler for COBOL/VSE.
IBM COBOL Structuring Facility/MVS and VM Version 3 (COBOL/SF) Program Product 5696-737	Running under MVS or VM, COBOL/SF transforms unstructured DOS/VS COBOL or VS COBOL II to structured COBOL/VSE programs. Please note that COBOL/SF is not available under VSE.
IBM Debug Tool for VSE/ESA Release 1 Feature of the Full Function order of COBOL/VSE	Debug Tool for COBOL/VSE.
IBM VisualAge for COBOL Professional for OS/2 V2.0 Program Number 5639-B92	Workstation based COBOL compiler.
IBM COBOL Workstation Function Optional Feature of IBM COBOL/VSE V1 Program Number 5686-068	The same as VisualAge for COBOL Professional for OS/2 V 2.0.

4.1 DOS/VS COBOL MIGR Compiler Option

You can use the DOS/VS COBOL MIGR compiler option whenever you are planning to convert a DOS/VS COBOL program to COBOL/VSE with NOCMPR2. This option allows you to analyze the conversion effort, and helps identify required changes. By compiling your programs using MIGR, you can determine ahead of time what language elements must be converted.

There are incompatibilities in the following areas:

- New reserved words were introduced for added COBOL functions
- Language function is supported in a different manner
- Language function is no longer supported

You can set the MIGR compiler option either as an installation default at install time, or when compiling a DOS/VS COBOL program. When you set MIGR on, the compiler flags most statements that are changed in, or not supported by, COBOL/VSE with NOCMPR2.

4.2 DOS/VS COBOL Migration PTFs

Customers running DOS/VS COBOL 1.3.0 and DOS/VS COBOL 1.3.1 are no longer able to order these products since they were withdrawn from marketing on March 24, 1997. These products are no longer available on any optional product distribution media, or individual product order.

The components are:

5746CB100 E44 (RE44) DOS/VS COBOL 1.3.0 Compiler

5746LM400 E45 (RE45) DOS/VS COBOL 1.3.0 Library

5746CB100 E46 (RE46) DOS/VS COBOL 1.3.1 Compiler

5746LM400 E47 (RE47) DOS/VS COBOL 1.3.1 Library

Migration assistance for these customers is available by PTFs. These PTFs contain sample JCL that will assist in the migration of both libraries and MSHP service information to a new operating system environment. The target environment will be VSE/ESA 2.2 and above or VSE/ESA 1.4.3 and above. The following PTFs are available:

- PTF UQ02699 for DOS/VS COBOL 1.3.0 Library
- PTF UQ02700 for DOS/VS COBOL 1.3.1 Library
- PTF UQ02697 for DOS/VS COBOL 1.3.0 Compiler
- PTF UQ02698 for DOS/VS COBOL 1.3.1 Compiler

The PTFs provide Z. members in PROD2.PROD that create an MSHP installation tape for the product currently on your system. This tape is suitable for product installation on another system. The PTFs do this by defining a temporary sublibrary, copying all the members of the compiler/library from PRD2.PROD to that sublibrary, then backing up the product to tape from that sublibrary.

This is handy for people who have not kept their stacked product tape from VSE/ESA 2.1 or earlier. MSHP history information and appropriate service also are included on this tape to keep your service level up to date.

4.3 COBOL/VSE FLAGMIG, CMPR2, and NOCOMPILE Compiler Options

You can identify DOS/VS COBOL and VS COBOL II Release 2 statements that are either not supported or changed in COBOL/VSE NOCMPR2 by compiling existing old programs with the COBOL/VSE FLAGMIG compiler option together with the COBOL/VSE CMPR2 compiler option. The NOCOMPILE option can also be used with this for saving system resources over a full compile with the COMPILE compiler option.

4.4 COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)

CCCA/VSE is an effective tool for converting old COBOL source code and copy modules to the new COBOL standard. CCCA converts DOS/VS COBOL, and COBOL 74 Standard VS COBOL II (VS COBOL II Release 3 and 4(CMPR2)) source code to COBOL 85 standard VS COBOL II Release 3 or 4(NOCMPR2) or to COBOL/VSE.

CCCA is designed to automate identifying incompatible source code and converting it to COBOL/VSE source code. Using CCCA will help to reduce the effort required to convert programs, and to minimize conversion errors.

In cases where a statement is no longer supported and has no equivalent statement in the target COBOL, CCCA flags the statement. Table 5 on page 43 shows how CCCA handles incompatible source code.

CCCA is designed so that you can customize the conversion process to meet unique conversion requirements. Installation and usage are easy, fast and straight forward.

For a detailed description on how to use CCCA/VSE see Chapter 6, “CCCA/VSE” on page 47.

Note: If you need to both convert your programs and structure your code and you have COBOL/SF available, you can select an option within COBOL/SF that will activate CCCA. CCCA will convert your program; then COBOL/SF will structure it.

4.5 COBOL Report Writer Precompiler Release 4

The COBOL Report Writer statements are not supported in COBOL/VSE (and VS COBOL II), instead the Report Writer feature is supported through use of the Report Writer Precompiler Program Offering. The precompiler enables you to use COBOL/VSE to compile your source programs written for DOS/VS COBOL that incorporate Report Writer without needing to convert or rewrite the Report Writer code. The precompiler also enables you to use Report Writer in new programs, with the additional benefit of a greatly enhanced set of functions. All the ANSI-85 features affecting Report Writer that first appeared with VS COBOL II R3 are supported. The additional ANSI-85 Report Writer features are also supported, provided the NOCMPR2 option is in effect.

You can use the COBOL Report Writer Precompiler to perform two functions:

1. To precompile applications containing Report Writer statements so the code will be acceptable to the COBOL/VSE compiler.

   For this function the precompiler runs under the control of the compiler by using the compiler′s **EXIT** option. By this method the compiler appears to handle Report Writer itself as a built-in part of COBOL. When used in its built-in form, COBOL Report Writer Precompiler R4 uses either COBOL/VSE and LE/VSE or VS COBOL II Compiler and Library.

2. To permanently convert Report Writer statements to valid COBOL statements that can be compiled in COBOL/VSE.

   In this case you run the precompiler as a separate step. It scans the source code for any Report Writer elements, and converts them to valid COBOL, leaving the rest of the source program unchanged. The resultant

intermediate source program is written to SYSPCH. This may then be compiled in a second JCL step. When used in this stand-alone form, it uses LE/VSE or the run-time library of VS COBOL II, and is used together with COBOL/VSE or the VS COBOL II compiler.

The Report Writer Precompiler offers the following features:

- Extended Report Writer language capabilities.
- Automatic invocation of the target COBOL compiler - as though Report Writer statements in the source program are being processed by the COBOL compiler itself.
- Single consolidated source listing merge information from the Precompiler and COBOL compiler listings.
- COPY library members can contain Report Writer statements.
- Supports the COBOL/VSE nested COPY feature.
- Performs a diagnostic check of the Report Writer source statements.
- Can be run in stand-alone mode to convert Report Writer statements in your COBOL programs into non-Report Writer COBOL source statements acceptable to the COBOL/VSE compiler.
- VSE/ESA, MVS/ESA and VM/ESA support.

Further information can be found in the manuals *COBOL Report Writer Precompiler Installation and Operation for VSE/ESA* and *COBOL Report Writer Precompiler Programmer's Manual*.

Note: A service update to COBOL Report Writer Release 4 is available for Year 2000 readiness. To receive this service update please contact the following vendor:

SPC Systems - Mr. John Piggott

69 Merton Hall Road, Wimbledon

London SW19 3PX, United Kingdom

Telephone: (44) 181 540 8409

Fax: (44) 181 540 6152

4.6 COBOL Structuring Facility/MVS and VM

IBM COBOL Structuring Facility (COBOL/SF) offers you a way to re-engineer your unstructured COBOL programs into structured COBOL code. By automating this process, you can reduce the cost of program maintenance and increase programmer productivity. This helps you protect your investments in existing COBOL programs.

Note: Although COBOL/SF is not available for execution under VSE, you can still benefit from the features it offers if your site operates either a VM or MVS system. When installed under VM or MVS, COBOL/SF can be used to restructure COBOL programs.

The input to COBOL/SF should be valid DOS/VS COBOL, VS COBOL II, or COBOL/VSE source code.

COBOL/SF can also help you establish and enforce structured programming standards for new application program development. You can process all newly developed code through COBOL/SF to ensure that your structured code standards are being maintained.

COBOL/SF can sometimes uncover previously unknown information about the input program - information that might provide an entirely different view of how the program works. The COBOL/SF report documents potential anomalies in the code and in each case describes the action taken by COBOL/SF. The report also details complex parts of the unstructured program. There are other reports available that do the following:

- Sequence and list the input program
- List the output program
- Display cross-reference information from the input program to the output program.

Note: If you need to both convert your programs and structure your code and you have CCCA available, you can select an option within COBOL/SF that will activate CCCA. CCCA will convert your program; then COBOL/SF will structure it.

Details can be found in the manuals *COBOL/SF Host User's Guide* and *COBOL/SF Reference Guide*.

4.7 Debug Tool for VSE/ESA Release 1

Debug Tool for VSE/ESA (DT/VSE) is a source-level debugger for programs that were compiled under High Level Language compilers that support LE/VSE. DT/VSE is a program-testing and analysis aid that allows users to examine, monitor, and control the execution of their programs.

The Debug Tool can prove invaluable during conversion and testing. Details can be found in Chapter 7, “Debug Tool for VSE/ESA” on page 107.

4.8 VisualAge for COBOL, Professional for OS/2 V 2.0

VisualAge for COBOL for OS/2 provides a workstation environment for developing and maintaining COBOL programs. With a context sensitive editor, and enhancements to the Redeveloper it provides an environment for doing much of the Year 2000 changes for COBOL applications. The Redeveloper Program Understanding tool can be used to analyze COBOL/VSE programs. Program Understanding uses the Associated Data (ADATA) files created by the COBOL compiler to perform its analysis. To use Program Understanding to analyze your programs you can either:

- Download your COBOL/VSE program source and copybooks and compile using VisualAge for COBOL on the workstation to create the ADATA files.
- Compile your COBOL/VSE programs on your VSE/ESA system and download the resultant ADATA files. PTF UQ04730 is available for COBOL/VSE to enable ADATA support needed for Program Understanding.

Since these enhancements were not yet available, when this redbook was written, we could not gain any experience with them.

4.9 COBOL Workstation Feature

An alternate method for you to acquire the functions delivered by VisualAge for COBOL, Professional for OS/2 V2.0 is to order the COBOL Workstation Feature. This offering is orderable as a feature of COBOL/VSE and provides the following additional benefits over VisualAge for COBOL:

- Monthly license charging (orderable in increments of 20 user licenses).
- Program services, including phone support.
- Future releases at no additional charge.

Chapter 5. Significant Differences between Old COBOL and COBOL/VSE

There are three categories in the ″difference of languages″:

- New functions
- Obsolete functions
- Changed functions

You can find the new functions of COBOL/VSE and LE in Chapter 2, “Why Migrate?” on page 11.

For existing programs, the problems are caused by the latter two cases (obsolete functions, changed functions).
In this chapter, we will consider these functions.

The manual *COBOL/VSE Migration Guide* describes how to modify your old COBOL source programs to COBOL/VSE level.

The old source programs can be separated into the following four groups depending on the compiler and options for compilation of them.

- DOS/VS COBOL using LANGLVL(1) (COBOL 68 standard)
- DOS/VS COBOL using LANGLVL(2) (COBOL 74 standard)
- VS COBOL II using CMPR2 (COBOL 74 standard)
- VS COBOL II using NOCMPR2 (COBOL 85 standard)

A detailed list of all incompatibilities can be found in the manual *COBOL/VSE Migration Guide* (and *VS COBOL II Migration Guide* for the difference between CMPR2 and NOCMPR2).

This chapter concentrates on some aspects based on experience in actual migration projects:

- Which incompatibilities frequently happen?
- Which incompatibilities cause a lot of impact?
- Which incompatibilities are easy/difficult to find?
- Which incompatibilities are easy/difficult to modify?

In addition, you will find typical examples and the result/experience of converting them using CCCA. See Appendix A, “Sample Case of Language Conversion” on page 135. This shows how CCCA can help your conversion and what kind of work remains to be done after CCCA conversion.

5.1 Differences from DOS/VS COBOL Using LANGLVL(1)

1. Difference from COBOL 68 Standard

   - Report Writer function was removed

     CCCA does not support this conversion. In this case, you need to use the program product COBOL Report Writer Precompiler. For more detail, see Chapter 4, “Conversion Aids” on page 29.

- ISAM function was removed

  ISAM handling statements must be converted to VSAM KSDS/ESDS. CCCA supports this conversion for source code. The file must be converted to KSDS/ESDS.

- DAM function was removed

  DAM handling statements must be converted to VSAM RRDS. CCCA supports this conversion for source code. The file must be converted to RRDS.

- Some statements were removed

  The EXAMINE and TRANSFORM statements to handle character strings were removed and integrated into a new statement INSPECT.
  The EXHIBIT statement to display data value was removed. It must be converted to a DISPLAY statement which has a different syntax rule.

- Combined Abbreviated Relational condition was changed

  Combined relation using NOT and abbreviated condition causes different result.

- Reserved word list change

  COBOL reserved words cannot be used for identifiers. Therefore, some variable names must be changed.

- Others not listed here

2. Other Differences

   After the above modifications, your source code is now LANGLVL(2). There are more differences between LANGLVL(2) and COBOL/VSE. Please see the next section, ″Difference from DOS/VS COBOL Using LANGLVL(2)″.

   This does not mean the conversion from LANGLVL(1) takes two steps. You can migrate a program with LANGLVL(1) or (2) in one step using CCCA.

5.2 Differences from DOS/VS COBOL Using LANGLVL(2)

1. Difference from COBOL 74 Standard - general enhancement

   - ALPHABETIC class was changed to contain lower-case
   - FILE STATUS code was changed

     FILE STATUS code returns more detail codes. In addition, we can use another status key for a VSAM file.

   - Reserved word change

     COBOL reserved words cannot be used for identifiers. Therefore, some variable names must be changed.

   - Others not listed here

2. Difference from COBOL 74 standard -- avoiding unexpected result

   In some cases, dynamic change of control variables during execution causes unexpected results decreasing the stability of structured code. Therefore, in COBOL 85, dynamic evaluation of control variables was minimized.

   - PERFORM ..VARYING/AFTER control
   - UNSTRING subscript evaluation

- STOP RUN and GOBACK

  In a COBOL/VSE (and VS COBOL II) subprogram you can end a subprogram with STOP RUN to terminate the run unit. If you use the run-time option RTEREUS you have to change the STOP RUN statements to GOBACK statements. In DOS/VS COBOL, when a GOBACK or EXIT PROGRAM statement is executed within a PERFORM structure, the PERFORM remains in its uncompleted state. In COBOL/VSE (and VS/COBOL II), however, the end of the PERFORM range is considered to have been reached.

3. Difference from COBOL 74 standard -- performance improvement

   The SEARCH statement requires good performance and also consumes many general registers, so the coding syntax became stricter.

   - SEARCH statement change

     Table value must appear on the left side in a WHEN comparison

   - SEARCH or SEARCH ALL .. WHEN with no imperative

4. Device independence

   This change is not a COBOL standard change but an improvement in the device independence of the COBOL language.

   - ASSIGN clause change

     COBOL/VSE supports only the restricted formats of the Assignment name.

5. SORT enhancement

   This change is not a COBOL standard change but an improvement in SORT.

   - SORT special register was changed
   - SORT-OPTION IS clause was replaced by SORT-CORE-SIZE and SORT-MESSAGE registers

6. Syntax checking has become stricter to improve accuracy

   This change is not a COBOL standard change but an improvement in checking by the compiler. These changes are to improve the quality of the program and avoid unexpected results. However, from the customer viewpoint, the compiler might show errors on some programs. These problems can be solved by recompilation after small modifications.

   - Paragraph with no period is checked as error
   - Occurs depending on .. clause syntax
   - Unstring statement syntax
   - Others not listed here

5.3 Differences from VS COBOL II Using CMPR2

If your VS COBOL II source programs were compiled with the CMPR2 compiler option, the source programs are COBOL 74 standard level. The difference is described in the manual *VS COBOL II Migration Guide*, Appendix C, CMPR2 to NOCMPR2 language difference.

1. Difference from COBOL 74 Standard

In general, the differences between CMPR2 and NOCMPR2 are contained in the differences from DOS/VS COBOL LANGLVL(2). New items are as follows:

- BLOCK CONTAINS clause for SAM ESDS
- CALL.. ON OVERFLOW
- COPY..REPLACING using non-COBOL characters

  Replacing of non-COBOL characters raises an E-level message. Lower case characters and colon(:) became COBOL characters.
- comparison between scaled integers and non-numerics

  Integer value with ″P″ attribute (for example, PIC 999PP) is now treated as a character string without P digits in comparison with characters.
- handling of unavailable SAM files
- IMPLICIT EXIT PROGRAM
- PERFORM RETURN mechanism

  This is also to avoid unexpected results after a subroutine terminated in pending status.
- PICTURE clause with ′A′s and ′B′s

  ′A′ ′B′ data was ALPHABETIC class in CMPR2, now it is ALPHANUMERIC-EDITED. So, the String statement cannot be used for it.
- reserved word list change

  COBOL reserved words cannot be used for identifiers. Therefore, some variable names must be changed.
- SET ... TO TRUE

5.4 Differences from VS COBOL II Using NOCMPR2

1. If your VS COBOL II source programs were compiled with the NOCMPR2 compiler option (COBOL 85 Standard), in general they will be able to be compiled by COBOL/VSE without change. The only differences are the following four points:

   - REPLACE and Comment lines

     Blank or comment lines will not appear in the output of the REPLACE statements.
   - Precedence of USE procedure

     In VS COBOL II, a file-specific USE procedure always takes precedence over a mode-specific USE procedure. In COBOL/VSE, USE procedure precedence is based on a program-by-program level.
   - Reference Modification of a Variable-Length Group

     In VS COBOL II R3.2 or older version, if the length of an ODO (Occurs Depending On) object is not specified, the actual length is used. in COBOL/VSE and VS COBOL II R4, the maximum length is used.
   - Reserved Words List change

FUNCTION, PROCEDURE-POINTER were added to the reserved word list and now cannot be used for identifiers.

5.5 Incompatibilities that Frequently Happen

When we estimate total migration workload, the analysis of frequency of each problem is very important. The frequency depends on each customer′s application environment. Therefore, analysis of actual frequency of problems is essential.

According to some experiences, actual frequency of the problems is very small, although there are many incompatible points in the quick reference list.

In general, we can say as follows:

1. COBOL 68 level commands are less frequently used.
2. Old commands are less frequently used.

   Some old commands are not described even in the manual for DOS/VS COBOL R3.1. Such commands seem to be kept for compatibility for very old COBOL languages so we can assume they are less frequent.

   - NOTES statement
   - ON statement
3. Old commands which depend on old environment are less frequent.

   The operating environment changes more rapidly than the languages. The language elements which have a strong relationship with the old environment cannot be kept so long.

   - UPSI switch function
4. Some incompatibilities were caused by rarely used data attributes

   These problems are only in programs which use these data attributes.

   - Scaled integer (PIC 999PP) causes incompatibility when it is used for comparison, MOVE to alphabetic characters.
   - Picture ″B″ attribute causes many incompatibilities when it is used for CALL/CANCEL identifier and initialized.
5. Some incompatibilities happen only in very complicated usage of not so frequently used statements/data.

   These cases happen less frequently:

   - UNSTRING statement subscript evaluation change
   - PERFORM statement changes in VARYING/AFTER
   - OCCURS DEPENDING ON value for receiving items

5.6 Incompatibilities that Cause More Impact

This study is to estimate the impact of each incompatibility from the following aspects.

- If no compile error occurs, can we accept the different result?
- If a compile error occurs, can we omit the statement? or can we say the modification effort is very small?

1. Less frequently used statements will not cause a large impact.
2. The removal of restrictions does not cause any impact.
   - APPLY WRITE ONLY... Usage restriction was removed
   - SERVICE RELOAD .... became not necessary, only treated as a comment.
3. Debug purpose statements will not cause a large impact.

   Debugging statements should be removed in the production environment or just used for reference.
   - TRACE or NOTRACE
   - WHEN-COMPILED special register format change
4. The changes to make the result more accurate will not cause a large impact.
   - arithmetic statement changes

     This change causes the improvement of precision for floating point calculation.
   - subscripts out of range .. flagged at compile time
   - ON SIZE ERROR ..... changes in intermediate result

     This error condition was caused when intermediate results overflowed. In COBOL/VSE, this error is checked only for the final result.
     - DIVIDE .. ON SIZE ERROR
     - MULTIPLY .. ON SIZE ERROR
   - implicit EXIT PROGRAM
   - File Status change

     Most of the new status keys indicate the status which caused an ABEND in the old COBOL (COBOL 74 and before). So we can say this change is a new chance for us to understand the system status in more detail.
5. Some changes only to the compiler's internal logic can be ignored.
   - Unstring statement in Program Collating Sequence
   - String statement in Program Collating Sequence
   - Inspect statement in Program Collating Sequence
6. Some changes were made only to check compile errors more strictly. This kind of problem can be solved by small modifications and recompilation.
   - non-unique Program-ID names
   - paragraph period missing
   - UNSTRING syntax check

5.7 Incompatibilities that are Easy/Difficult to Find

How can we find the incompatible points in application libraries?

1. Some points are easily checked by keyword scanning.

   If you want to estimate the migration effort without using a lot of CPU resources, keyword scanning is a practical way.

- Reserved keywords change
- Obsolete commands

2. Some points are easily checked by compile error.

   The compiler can find most of the problems. If you use the FLAGMIG compiler option of DOS/VS COBOL or the MIGR compile option of COBOL/VSE, you can get the flag message for migration.

3. Some points cannot be found unless statement analysis is done.
   - PERFORM statement change in the VARYING/AFTER
   - UNSTRING ...subscript evaluation change

4. Some points cannot be found unless data and statement analysis are done.
   - CALL identifier statement B symbol in PICTURE clause

5.8 Incompatibilities that are Easy/Difficult to Modify

This section describes how to find the most difficult points in your application environment.

If you use CCCA (COBOL and CICS Command Level Conversion Aid), most syntax changes are supported. Some incompatible points are not supported by CCCA. Some incompatible points are supported by CCCA only for FLAG (checking). So, you must check and modify these flagged points manually. (This information is based on the manual *COBOL and CICS Command Level Conversion Aid for VSE*, Appendix B. Converted COBOL Language Elements.)

1. Changes which are automatically converted by CCCA

   The following items are already covered by the CCCA conversion aid. If you have to convert them without CCCA it will be very difficult work.
   - EXAMINE statement
   - TRANSFORM statement
   - EXHIBIT statement
   - CURRENT-DATE register
   - TIME-OF-DAY register
   - WHEN-COMPILED register
   - reserved word list change
   - and others

2. Changes which are only indicated by CCCA

   The following items can be found by CCCA. However, the solution can not be decided automatically. So, you must check them and modify them manually.
   - CURRENCY SIGN .. if not valid, you must specify another character.
   - CALL ON OVERFLOW
   - CALL USING
   - COPY REPLACING
   - DIVIDE ON SIZE ERROR
   - MULTIPLY ON SIZE ERROR

- NSTD-REELS special register
- FILE STATUS value checking
- PERFORM VARYING AFTER
- PICTURE clause scaled integers(P)
- REDEFINE clause for table
- REPLACE statement
- REPORTWRITER related statements

  The aid Report Writer Precompiler can convert these statements.
- SEARCH ALL
- STRING statement
- UNSTRING statement
- OCCURS syntax error
- INDEX NAME (qualified)
- COM-REG use

3. Changes which are not supported by CCCA
   - APPLY WRITE ONLY clause restriction removed
   - arithmetic statement changes
   - COPY statement changes 68 standard
   - MOVE statement binary value vs display value
   - NUMERIC class tests for group items
   - OCCURS DEPENDING ON ASCENDING/DESCENDING option
   - OPEN statement failing for VSAM files (file status 39)
   - OPEN statement failing for SAM files (file status 39)
   - READ statement redefined record keys in the KEY phrase
   - READ and RETURN statement changes INTO phrase
   - RECORD CONTAINS n CHARACTERS INTO phrase
   - READ and RETURN statement changes INTO phrase
   - RENAMES clause nonunique, nonqualified data names
   - RERUN clause changes
   - SEARCH or SEARCH ALL ..WHEN with no imperative
   - Segmentation changes..PERFORM statement in independent segments
   - SERVICE RELOAD
   - SORT special registers
   - SORT-OPTION clause
   - subscript out of range check

5.9 List of Incompatibilities from Old COBOLs to COBOL/VSE

The following table is a summary of incompatibilities between old COBOLs and COBOL/VSE.

1. Category column(*)
   - 68 means only DOS/VS COBOL LANGLVL(1)
   - 74 means only DOS/VS COBOL LANGLVL(2) or VS COBOL II CMPR2
   - DOS means only DOS/VS COBOL
   - OLD means only DOS/VS COBOL and VS COBOL II CMPR2
   - NOC means only VS COBOL II NOCMPR2
   - ALL means it is applicable for all language levels
2. Frequency column means the frequency level of the difference in the general environment.

   L/M/H stand for **L**ow, **M**edium and **H**igh frequency.
   LL stands for very Low frequency.
3. IMPACT column means the impact level when the differences remain unchanged.

   L/M/H stand for **L**ow, **M**edium and **H**igh impact.
   LL stands for very Low impact.
4. CCCA support means the support level of the CCCA migration aid.(*)

   C=converted R=removed F=flagged W=warning N=no support
5. Difficulty of manual check means the difficulty of manual check without CCCA.

   E/L/M/H stand for **E**rror when compile **L**ow, **M**edium and **H**igh difficulty
6. Difficulty of manual modify means the difficulty of manual modification after the use of CCCA.

   -/L/M/H stand for **-** No, **L**ow, **M**edium and **H**igh difficulty.

Notes (*): The Category and CCCA support columns are based on the CCCA manual.

Table 5 (Page 1 of 4). List of Incompatibilities

Incompatible point name	CATEGORY	Frequency	IMPACT	CCCA support level	difficulty of manual check	difficulty of manual modify
ALPHABETIC class changes	OLD	L	L	C	L	-
ALPHABET clause change	DOS	L	E	C	E	-
APPLY WRITE ONLY restriction removed	68	LL	LL	N	L	-
arithmetic statement change	68	LL	LL	N	M	-
ASSIGN TO integer system-name	DOS	L	L	C	E	-
ASSIGN .. FOR MULTIPLE REEL/UNIT	DOS	L	LL	C	E	-
ASSIGN clause changes - assign name forms	DOS	M	L	C	E	-

Table 5 (Page 2 of 4). List of Incompatibilities

Incompatible point name	CATEGORY	Frequency	IMPACT	CCCA support level	difficulty of manual check	difficulty of manual modify
B symbol in PICTURE clause -changes in evaluation	OLD	LL	L	F	E	M
BLANK WHEN ZERO clause and asterisk replace	DOS	LL	L	C	E	M
BLOCK CONTAINS clause for SAM ESDS	OLD	L	LL	C	L	-
CALL identifier statement B symbol in PICTURE clause	OLD	LL	L	F	E	M
CALL statement changes -- procedure names and file names in USING	DOS	LL	M	F	E	M
CALL .. ON OVERFLOW	74	L	LL	F	L	-
CANCEL statement B symbol in PICTURE clause	74	LL	L	F	E	M
combined abbreviated relation condition changes	68	LL	M	CF	M	M
COM-REG special register	68	LL	L	F	E	M
comparison between scaled integers and nonnumerics	74	LL	L	F	M	M
COPY statement changes from 68 standard	68	LL	L	C	E	-
COPY ..REPLACING	74	LL	L	F	L	M
CURRENCY SIGN clause change	68	LL	L	F	E	L
CURRENT-DATE special register	68	LL	M	C	E	-
DAM file handling	DOS	L	L	C	E	M
DIVIDE ..ON SIZE ERROR change	OLD	L	LL	F	L	-
EXAMINE statement	68	LL	M	C	E	-
EXHIBIT statement	68	LL	M	C	E	-
EXIT PROGRAM/GOBACK	OLD	L	L	C	L	M
FILE-LIMIT clause of the FILE-CONTROL	DOS	LL	L	C	E	-
FILE STATUS code change	74	M	L	F	M	L
GIVING option of USE AFTER STANDARD ERROR	DOS	L	L	C	E	M
handling of unavailable SAM files	74	L	L	N	L	M
IF..OTHERWISE statement	DOS	L	M	C	E	-
implicit EXIT PROGRAM	OLD	L	LL	N	L	-
index names	DOS	L	L	C	E	L
INITIALIZE..replacing changes in evaluation	DOS	LL	L	F	L	M
ISAM file handling	DOS	L	L	C	E	M
INSPECT statement in PROGRAM collating sequence?	OLD	M	LL	F	L	-
JUSTIFIED clause change	68	LL	L	C	L	-
LABEL RECORD is data-name in non-sequential files	OLD	L	L	C	L	-
MOVE statement -binary value vs display value	DOS	L	L	N	L	M
MOVE statements and comparisons- scaled integer	68	LL	L	F	L	M

Table 5 (Page 3 of 4). List of Incompatibilities

Incompatible point name	CATEGORY	Frequency	IMPACT	CCCA support level	difficulty of manual check	difficulty of manual modify
MOVE CORRESPONDING statement	68	LL	L	C	E	-
MULTIPLY ON SIZE ERROR change in intermediate results	OLD	L	LL	F	L	M
non-unique program-ID names	DOS	LL	M	C	E	-
NOTE statement	DOS	LL	M	C	E	-
NSTD-REELS special register	DOS	LL	L	F	E	M
NUMERIC class tests for group items	DOS	L	M	N	L	M
OCCURS clause	DOS	LL	L	F	E	L
OCCURS DEPENDING ON ASCENDING/DESCENDING KEY option	DOS	LL	L	N	E	M
OCCURS DEPENDING ON value for receiving items changed	OLD	LL	L	C	M	M
OCCURS DEPENDING ON reference modification	NOC	LL	L	C	M	-
ON statement	DOS	LL	L	C	E	-
OPEN statement failing for VSAM files	DOS	L	L	N	L	L
OPEN statement failing for SAM files	DOS	L	L	N	L	L
PERFORM RETURN mechanism	74	LL	LL	F	M	M
PERFORM statement changes in the VARYING/AFTER	74	LL	L	F	M	M
periods on paragraph missing	DOS	LL	M	C	E	-
READ statement redefined record keys in the KEY phrase	DOS	LL	L	N	E	M
READ and RETURN statement changes .. INTO options	DOS	L	LL	N	M	M
READY TRACE and RESET TRACE statement	DOS	L	LL	C	E	M
RECORD CONTAINS n CHARACTERS	OLD	L	L	N	L	L
REDEFINES clause in SD or FD entries	DOS	L	LL	C	E	-
REDEFINE clause with tables	DOS	L	L	N	E	M
relation conditions	68	LL	L	C	E	-
REMARKS paragraph	68	LL	LL	C	E	-
RENAMES clause nonunique, nonqualified data names	DOS	L	L	N	E	L
REPLACE and comment lines	NOC	L	LL	F	M	-
Report Writer	68	LL	M	F	E	H *1
RERUN clause change	DOS	L	L	N	L	L
RESERVE clause change	DOS	L	L	C	L	-
reserved word list change	ALL	L	L	C	E	-
SEARCH statement changes	DOS	L	L	C	E	-

Table 5 (Page 4 of 4). List of Incompatibilities

Incompatible point name	CATEGORY	Frequency	IMPACT	CCCA support level	difficulty of manual check	difficulty of manual modify
SEARCH or SEARCH ALL..WHEN with no imperative	DOS	L	L	N	E	M
segmentation changes ..PERFORM statement in independent segments	DOS	L	L	N	L	M
SELECT OPTIONAL clause changes	68	LL	L	C	L	L
SERVICE RELOAD	DOS	L	LL	C	L	-
SORT special registers	DOS	L	L	N	L	L
SORT-OPTION IS clause	DOS	L	L	C	E	M
START..USING KEY statement	DOS	L	L	C	E	-
STRING statement in PROGRAM COLLATING SEQUENCE clause	74	L	LL	C	L	-
subscripts out of range -- flagged at compile time	DOS	LL	LL	N	E	-
THEN as a statement connector	DOS	L	L	C	E	-
TIME-OF-DAY special register	68	LL	L	C	E	-
TRANSFORM statement	68	LL	L	C	E	-
UNSTRING statement in PROGRAM-COLLATING SEQUENCE	74	L	LL	F	L	-
UNSTRING statement coding with OR/IS	74	LL	L	C	E	-
UNSTRING statements subscript evaluation changes	74	LL	L	F	L	M
UPSI switches	OLD	LL	L	CF	E	M
USE procedures	NOC	L	L	F	L	L
USE BEFORE STANDARD LABEL statement	DOS	L	L	C	E	-
VALUE clause signed value in relation to the PICTURE clause	DOS	LL	LL	C	E	-
VALUE clause condition names	DOS	L	LL	C	E	-
WHEN-COMPILED special register	DOS	L	LL	C	L	-
WRITE AFTER POSITIONING	68	LL	L	C	E	-

NOTE. *1 If the Report Writer Precompiler was used, this conversion is automatically done.

Chapter 6. CCCA/VSE

6.1 Introduction to CCCA/VSE

This section summarizes:

- What CCCA/VSE does
- How CCCA/VSE works
- LCP (Language Conversion Program)

6.1.1 What CCCA/VSE Does

CCCA/VSE converts old COBOL source and copy modules to the new COBOL standard. This tool shortens the COBOL migration and eases the upgrade to new COBOL technology.

CCCA/VSE helps you convert COBOL source from:

Source Language	Release	Program Number
DOS/VS COBOL	3	5746-CB1
OS/VS COBOL	2	5740-CB1
VS COBOL II	1,1.1,2,3,3.1,3.2	5668-958

to:

ANSI 85 Target Language	Release	Program Number
VS COBOL II	3 or later	5668-958
COBOL/VSE	1 or later	5686-068

The following are the key CCCA facilities:

- Conversion of most syntax differences between DOS/VS COBOL and VS COBOL II programs and COBOL/VSE programs.
- Elimination of conflicts between DOS/VS COBOL and VS COBOL II user-defined names and COBOL/VSE reserved words.
- Flagging of language elements that cannot be directly converted.
- Statement-by-statement diagnostic listing.
- Conversion management information, including where-used reports for COPY books and files.
- Conversion of EXEC CICS commands.
- Removal and/or conversion of the Base Locator for Linkage Section (BLL) mechanism and references.

6.1.2 How CCCA/VSE Works

CCCA/VSE is a combination of CICS application and batch application. You use CCCA/VSE online CICS panels to:

- Define the type of conversion you want
- Submit a batch job to convert your programs

The conversion consists of three phases:

Phase 1: Analyze input source

- Translates the original source program into a set of character strings known as tokenized source
- Extracts copy members from the appropriate copy libraries
- For each item in the tokenized source, identifies whether conversion is required, and if so, which LCP (Language Conversion Program) to use

Phase 2: Create change requests

For each item that needs converting, phase 2:

- Loads an LCP
- Runs the LCP
- Generates change requests

Phase 3: Apply changes and generate output

- Applies the change requests from phase 2, creating new source programs and, if required, new copy members
- Generates the diagnostic listing

6.1.3 LCP (Language Conversion Program)

The LCPs are written in a COBOL-like language used to describe the process of converting the differences between the old COBOL language, and the new COBOL language. A set of LCP modules describing how each old COBOL language element is to be converted into the new COBOL language is provided with this tool. The set also provides CICS command level-related statements conversion. The basic set enables users to convert most differences, and can be very easily customized for specific conversion requirements.

By adding new LCPs, the user can:

- Change COBOL source syntax.
- Add, replace or remove a word, clause or statement.
- Indicate where the new generated COBOL source needs to be reviewed for possible manual action.

For details of how to customize CCCA/VSE and LCP, please refer to Chapter 6, ′Customizing CCCA/VSE′ and Chapter 7, ′Developing Language Conversion Programs′ in the *COBOL and CICS Command Level Conversion Aid for VSE Installation and User′s Guide*.

6.2 Installation of CCCA/VSE

This section describes:

- Software requirements for CCCA/VSE
- Installing CCCA/VSE
- What to do if there are problems

6.2.1 Software Requirements for CCCA/VSE

The software requirements for using CCCA/VSE are:

- VSE/ESA 1.2 and subsequent releases
- VS COBOL II Library 1.3 or later, or LE/VSE V1
- A SORT program capable of being used with the COBOL SORT verb
- VS COBOL II Compiler 1.3 or later, or COBOL/VSE V1
- DL/I or SQL/DS for preprocessing if you have DL/I or SQL/DS programs

6.2.2 Installing CCCA/VSE

For details on installation of CCCA/VSE, please refer to ′Program Directory for COBOL and CICS Command Level Conversion Aid for VSE′.

6.2.3 What to Do If There are Problems

- **1QC3I - MEMBER ABJPPARM.P NOT FOUND ...**

  Please make sure you have run sample job ABJSLI with return code zero. Make sure you have ABJPPARM.P in your CCCA/VSE library. If the problem still exists, please check whether your CCCA/VSE library is in the search chain of the POWER partition.

- **DFH0951 - OPEN OF FILE ABJxxxx FAILED. JCL NOT AVAILABLE.**

  Please make sure you have DLBL definitions in your CICS startup job and the file definition in your FCT. If the problem still exists, please move those DLBL statements close to the EXEC DFHSIP statement in your CICS startup job.

6.3 CCCA/VSE Conversion Examples

This section gives you three CCCA/VSE conversion examples:

- ABJIVP01 - batch DOS/VS COBOL
- ABJIVP02 - batch DOS/VS COBOL with copy members
- ABJIVP03 - DOS/VS COBOL, CICS, with copy members and BLL cells in the Linkage Section

The source programs are in your CCCA/VSE library (the default is ABJ.PR$18W) with member type .C.

The converted programs are produced with the same name as the source program but in a different VSE library, which is specified by the user when doing CCCA/VSE conversion. The source programs remain unchanged.

The CCCA/VSE Diagnostic Reports are produced in the LST queue. As well as the report, CCCA/VSE creates conversion management reports which:

- produce cross-references between converted programs and files defined in the program (CCCA/VSE Converter Menu, Option 3 and 4)
- produce cross-references between converted programs and COPY modules (CCCA/VSE Converter Menu, Option 5 and 6)
- produce cross-references between converted programs and CALLed programs (CCCA/VSE Converter Menu, Option 7 and 8)
- produce status of conversion for all converted program runs (CCCA/VSE Converter Menu, Option L)

The conversion step received a Return Code of 8 requiring a manual update of the program, and consequently no compilation or linkage edit action as taken. Refer to the diagnostic report for an explanation of the actions taken by the conversion program CCCA/VSE.

6.3.1 ABJIVP01 - Batch COBOL Program

6.3.1.1 Input Source Program

```
000100 IDENTIFICATION DIVISION.                                         00010000
000200 PROGRAM-ID. ABJIVP01.                                            00020000
000300 REMARKS.        1                                                00030000
000400     THIS PROGRAM IS BEING WRITTEN TO TEST THE PROPER CONVERSION  00040000
000500     FROM OS/VS COBOL SOURCE LANGUAGE TO IBM SOURCE LANGUAGE.     00050000
000600 AUTHOR. XXXXXX.         2                                        00060000
000700 DATE-WRITTEN. JANUARY 24, 1983.        3                         00070000
000800                                                                  00080000
000900     NOTE - THE FOLLOWING AREAS ARE ADDRESSED        4            00090000
001000         1  REMARKS                                              00100000
001300         2  THEN                                                 00110000
001400         3  OTHERWISE                                            00120000
001500         4  CURRENT-DATE                                         00130000
001600         5  TIME-OF-DAY                                          00140000
001700         6  NOTE                                                 00150000
001800         7  EXAMINE...TALLYING...REPLACING                       00160000
001900         8  JUSTIFIED.                                           00170000
002000                                                                  00180000
002100 DATE-COMPILED. TODAYS DATE.          5                            00190000
002200 EJECT                                                            00200000
002300 ENVIRONMENT DIVISION.                                            00210000
002400 INPUT-OUTPUT SECTION.                                            00220000
002500 FILE-CONTROL.                                                    00230000
002600     SELECT PRINT-FILE                                            00240000
002700     ASSIGN TO UT-3330-S-DDPRINT.                                 00250000
002800 DATA DIVISION.                                                   00260000
002900 FILE SECTION.                                                    00270000
003000 FD  PRINT-FILE                                                   00280000
003100     RECORDING MODE IS F            6                             00290000
003200     LABEL RECORDS ARE STANDARD                                   00300000
003300     DATA RECORD IS OUT-LINE.                                     00310000
003400 01  OUT-LINE           PIC X(80).                               00320000
003500 WORKING-STORAGE SECTION.           7  8  9                       00330000
003600 77  MY-COUNTER         PIC 9(5)  VALUE 0.                       00340000
003700 77  TRIPSWCH           PIC 9     VALUE 0.                       00350000
003800 77  PASSWCH            PIC 9     VALUE 0.                       00360000
003900 77  FAILSWCH           PIC 9     VALUE 1.                       00370000
004100 77  CURRFLAG           PIC 9     VALUE 0.                       00380000
004200 77  TOFDFLAG           PIC 9     VALUE 0.                       00390000
004300 77  I                   PIC 9     VALUE 0.                       00400000
004400 77  DATE1              PIC X(8)  VALUE SPACES.                  00410000
004500 77  DATE2              PIC X(8)  VALUE SPACES.                  00420000
004600 77  DATE3              PIC X(8)  VALUE SPACES.                  00430000
004700 77  TIME1              PIC X(6)  VALUE SPACES.                  00440000
004800 77  TIME2              PIC X(6)  VALUE SPACES.                  00450000
004900 77  TIME3              PIC X(6)  VALUE SPACES.                  00460000
005400                                                                  00470000
005500 01  ORIGINAL-NUMBER.                                            00480000
005600     05  FILLER     PIC 9(18) VALUE 0.                           00490000
005700     05  FILLER     PIC 9(18) VALUE 0.                           00500000
```

Figure 3 (Part 1 of 4). ABJIVP01 - Source Program

```
005800     05  FILLER     PIC 9(18) VALUE 000000009099843576.          00510000
005900     05  FILLER     PIC 9(18) VALUE 121212121212121290.          00520000
006000                                                                  00530000
006100 01  THIS-DEF REDEFINES ORIGINAL-NUMBER.                         00540000
006200     03  A-NUMBER OCCURS 2 TIMES.                                00550000
006300         05  LINE1      PIC 9(18).                               00560000
006400         05  LINE2      PIC 9(18).                               00570000
006500                                                                  00580000
006600 01  A-POEM.                                                     00590000
006700     03  LINE1.                                                  00600000
006800         05  FILLER     PIC X(20) VALUE "ROSES ARE RED VIOLET".  00610000
006900         05  FILLER     PIC X(20) VALUE "S ARE BLUE,         ".  00620000
007000     03  LINE2.                                                  00630000
007100         05  FILLER     PIC X(20) VALUE "SUGAR IS SWEET AND S".  00640000
007200         05  FILLER     PIC X(20) VALUE "O ARE YOU.          ".  00650000
007300                                                                  00660000
007900                                                                  00670000
008000 01  FAIL1CON2.                                                  00680000
008100     03  FILLER         PIC XX    VALUE SPACES.                  00690000
008200     03  CPLACE         PIC X(20) VALUE SPACES.                  00700000
008300                                                                  00710000
008400 01  FAIL2CON.                                                   00720000
008500     03  FILLER         PIC X(20) VALUE "ALL THREE READINGS O".  00730000
008600     03  FILLER         PIC X(20) VALUE "F 'CURRENT-DATE' SHO".  00740000
008700     03  FILLER         PIC X(20) VALUE "ULD BE THE SAME, BUT".  00750000
008800     03  FILLER         PIC X(20) VALUE " THEY ARE:          ".  00760000
008900                                                                  00770000
009000 01  FAIL2CON2.                                                  00780000
009100     03  FILLER         PIC XX    VALUE SPACES.                  00790000
009200     03  DPLACE         PIC X(8)  VALUE SPACES.                  00800000
009300                                                                  00810000
009400 01  FAIL3CON.                                                   00820000
009500     03  FILLER         PIC X(20) VALUE "THE THREE READINGS O".  00830000
009600     03  FILLER         PIC X(20) VALUE "F 'TIME-OF-DAY' SHOU".  00840000
009700     03  FILLER         PIC X(20) VALUE "LD BE EQUAL OR IN AS".  00850000
009800     03  FILLER         PIC X(20) VALUE "CENDING ORDER,      ".  00860000
009900                                                                  00870000
010000 01  FAIL3CON1.                                                  00880000
010100     03  FILLER         PIC X(20) VALUE "BUT THEY ARE:       ".  00890000
010200                                                                  00900000
010300 01  FAIL3CON2.                                                  00910000
010400     03  FILLER         PIC XX    VALUE SPACES.                  00920000
010500     03  TPLACE         PIC X(6)  VALUE SPACES.                  00930000
010600                                                                  00940000
010700 01  FAILCON.                                                    00950000
010800     03  FILLER         PIC X(20) VALUE "TEST CASE SAMPLE   F".  00960000
010900     03  FILLER         PIC X(20) VALUE "AILED.              ".  00970000
011000                                                                  00980000
011100 01  SUCCESS.                                                    00990000
011200     03  FILLER         PIC X(20) VALUE "TEST CASE SAMPLE   I".  01000000
011300     03  FILLER         PIC X(20) VALUE "S SUCCESSFUL.       ".  01010000
```

Figure 3 (Part 2 of 4). ABJIVP01 - Source Program

```
011400 EJECT                                                            01020000
011500 PROCEDURE DIVISION.                                              01030000
011600 THIS-IS-A SECTION.                                               01040000
011700 START-HERE.                                                      01050000
011800     MOVE TIME-OF-DAY TO TIME1          10                       01060000
011900     OPEN OUTPUT PRINT-FILE                                       01070000
012000     MOVE CURRENT-DATE TO DATE1         11                       01080000
012500     MOVE CURRENT-DATE TO DATE2         12                       01090000
012700     MOVE CURRENT-DATE TO DATE3.        13                       01100000
012800                                                                 01110000
014000     MOVE TIME-OF-DAY TO TIME2.         14                       01120000
014100     IF DATE1 EQUAL DATE2 AND EQUAL DATE3 THEN                    01130000
014200         NEXT SENTENCE                                           01140000
014300     OTHERWISE          15                                       01150000
014400         MOVE FAILSWCH TO TRIPSWCH                               01160000
014500         MOVE DATE1 TO DPLACE                                    01170000
014600         WRITE OUT-LINE FROM FAIL2CON                            01180000
014700         WRITE OUT-LINE FROM FAIL2CON2                           01190000
014800         MOVE DATE2 TO DPLACE                                    01200000
014900         WRITE OUT-LINE FROM FAIL2CON2                           01210000
015000         MOVE DATE3 TO DPLACE                                    01220000
015100         WRITE OUT-LINE FROM FAIL2CON2.                          01230000
015200     MOVE TIME-OF-DAY TO TIME3.         16                       01240000
015300     IF (TIME1 LESS THAN TIME2 OR EQUAL TIME2) AND                01250000
015400        (TIME2 LESS THAN TIME3 OR EQUAL TIME3) THEN               01260000
015500         NEXT SENTENCE                                           01270000
015600     OTHERWISE          17                                       01280000
015700         MOVE FAILSWCH TO TRIPSWCH                               01290000
015800         MOVE TIME1 TO TPLACE                                    01300000
015900         WRITE OUT-LINE FROM FAIL3CON                            01310000
016000         WRITE OUT-LINE FROM FAIL3CON1                           01320000
016100         WRITE OUT-LINE FROM FAIL3CON2                           01330000
016200         MOVE TIME2 TO TPLACE                                    01340000
016300         WRITE OUT-LINE FROM FAIL3CON2                           01350000
016400         MOVE TIME3 TO TPLACE                                    01360000
016500         WRITE OUT-LINE FROM FAIL3CON2.                          01370000
016600 AFTER-THOUGHT.                                                   01380000
016700     EXAMINE A-POEM TALLYING ALL SPACES REPLACING BY "*"    18    01390000
016800     MOVE TALLY TO MY-COUNTER                                    01400000
016900     MOVE LINE1 OF A-POEM TO OUT-LINE WRITE OUT-LINE             01410000
017000     MOVE LINE2 OF A-POEM TO OUT-LINE WRITE OUT-LINE             01420000
017100     EXAMINE A-POEM TALLYING ALL "*".          19                01430000
017200     IF TALLY = MY-COUNTER                                       01440000
017300         MOVE "OK" TO OUT-LINE WRITE OUT-LINE                    01450000
017400     OTHERWISE          20                                       01460000
017500         MOVE "BAH" TO OUT-LINE WRITE OUT-LINE.                  01470000
017600     EXAMINE A-POEM TALLYING ALL "E"           21                01480000
017700     PERFORM THREE-LINES                                         01490000
017800     EXAMINE A-POEM TALLYING UNTIL FIRST "."          22         01500000
017900     PERFORM THREE-LINES                                         01510000
018000     EXAMINE A-POEM TALLYING LEADING "R"         23              01520000
```

Figure 3 (Part 3 of 4). ABJIVP01 - Source Program

```
018100     PERFORM THREE-LINES                                          01530000
018200     MOVE 2 TO I                                                  01540000
018300     EXAMINE A-NUMBER(I) TALLYING ALL 1          24               01550000
018400     PERFORM THREE-LINES                                          01560000
018500     EXAMINE A-NUMBER(I) TALLYING LEADING 0 REPLACING BY 2.    25 01570000
018600 THREE-LINES.                                                     01580000
018700     ADD TALLY TO MY-COUNTER.                                     01590000
018800     MOVE TALLY TO OUT-LINE WRITE OUT-LINE                        01600000
018900     MOVE MY-COUNTER TO OUT-LINE WRITE OUT-LINE.                  01610000
019000 THE-END.                                                         01620000
019100     IF TRIPSWCH EQUAL FAILSWCH OR MY-COUNTER NOT EQUAL 125       01630000
019200         WRITE OUT-LINE FROM FAILCON                              01640000
019300     OTHERWISE           26                                       01650000
019400         WRITE OUT-LINE FROM SUCCESS.                             01660000
019500     CLOSE PRINT-FILE.                                            01670000
019600     STOP RUN.           27                                       01680000
```

Figure 3 (Part 4 of 4). ABJIVP01 - Source Program

6.3.1.2 Output Program from CCCA/VSE Conversion

```
000010 IDENTIFICATION DIVISION.                                          00010000
000020 PROGRAM-ID. ABJIVP01.                                             00020000
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:12:10.
000060*REMARKS.        1                                                 00030000
000070*    THIS PROGRAM IS BEING WRITTEN TO TEST THE PROPER CONVERSION   00040000
000080*    FROM OS/VS COBOL SOURCE LANGUAGE TO IBM SOURCE LANGUAGE.      00050000
000090*AUTHOR. XXXXXX.          2                                       00060000
000100*DATE-WRITTEN. JANUARY 24, 1983.          3                       00070000
000110                                                                   00080000
000120*    NOTE - THE FOLLOWING AREAS ARE ADDRESSED          4          00090000
000130*        1  REMARKS                                                00100000
000140*        2  THEN                                                  00110000
000150*        3  OTHERWISE                                             00120000
000160*        4  CURRENT-DATE                                          00130000
000170*        5  TIME-OF-DAY                                           00140000
000180*        6  NOTE                                                  00150000
000190*        7  EXAMINE.  TALLYING.  REPLACING                        00160000
000200*        8  JUSTIFIED.                                            00170000
000210                                                                   00180000
000220*DATE-COMPILED. TODAYS DATE.           5                          00190000
000230 EJECT                                                             00200000
000240 ENVIRONMENT DIVISION.                                             00210000
000250 INPUT-OUTPUT SECTION.                                             00220000
000260 FILE-CONTROL.                                                     00230000
000270     SELECT PRINT-FILE                                            00240000
000280     ASSIGN TO UT-3330-S-DDPRINT.                                 00250000
000290 DATA DIVISION.                                                   00260000
000300 FILE SECTION.                                                    00270000
000310 FD  PRINT-FILE                                                   00280000
000320     .                 6                                         00300000
000330 01  OUT-LINE           PIC X(80).                               00320000
000340 WORKING-STORAGE SECTION.                                         00330000
000350 01  LCP-TIME-OF-DAY-68           PIC 9(6).          7
000360 01  LCP-TIME-OF-DAY-74.                             8
000370     05  LCP-TIME-74              PIC 9(6).
000380     05  FILLER                   PIC 9(2).
000390 01  LCP-CURRENT-DATE-68.                            9
000400     05  LCP-MONTH                PIC X(2).
000410     05  FILLER                   PIC X VALUE "/".
000420     05  LCP-DAY1                 PIC X(2).
000430     05  FILLER                   PIC X VALUE "/".
000440     05  LCP-YEAR                 PIC X(2).
000450 01  LCP-DATE-NEW-74.
000460     05  LCP-YEAR                 PIC X(2).
000470     05  LCP-MONTH                PIC X(2).
000480     05  LCP-DAY1                 PIC X(2).
000490 77  MY-COUNTER         PIC 9(5)  VALUE 0.                       00340000
```

Figure 4 (Part 1 of 4). ABJIVP01 - Converted Program

```
000500 77  TRIPSWCH           PIC 9     VALUE 0.                      00350000
000510 77  PASSWCH            PIC 9     VALUE 0.                      00360000
000520 77  FAILSWCH           PIC 9     VALUE 1.                      00370000
000530 77  CURRFLAG           PIC 9     VALUE 0.                      00380000
000540 77  TOFDFLAG           PIC 9     VALUE 0.                      00390000
000550 77  I                  PIC 9     VALUE 0.                      00400000
000560 77  DATE1              PIC X(8)  VALUE SPACES.                 00410000
000570 77  DATE2              PIC X(8)  VALUE SPACES.                 00420000
000580 77  DATE3              PIC X(8)  VALUE SPACES.                 00430000
000590 77  TIME1              PIC X(6)  VALUE SPACES.                 00440000
000600 77  TIME2              PIC X(6)  VALUE SPACES.                 00450000
000610 77  TIME3              PIC X(6)  VALUE SPACES.                 00460000
000620                                                                00470000
000630 **** No changes - same as source *****                         00480000
                  .
                  .
                  .
001180 PROCEDURE DIVISION.                                            01030000
001190 THIS-IS-A SECTION.                                             01040000
001200 START-HERE.                                                    01050000
001210     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME          10            01060000
001220     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001230     MOVE LCP-TIME-OF-DAY-68 TO TIME1
001240     OPEN OUTPUT PRINT-FILE                                     01070000
001250     ACCEPT  LCP-DATE-NEW-74 FROM DATE            11            01080000
001260     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001270     MOVE LCP-CURRENT-DATE-68 TO DATE1
001280     ACCEPT  LCP-DATE-NEW-74 FROM DATE            12            01090000
001290     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001300     MOVE LCP-CURRENT-DATE-68 TO DATE2
001310     ACCEPT  LCP-DATE-NEW-74 FROM DATE            13            01100000
001320     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001330     MOVE LCP-CURRENT-DATE-68 TO DATE3.
001340                                                                01110000
001350     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME          14            01120000
001360     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001370     MOVE LCP-TIME-OF-DAY-68 TO TIME2.
001380     IF DATE1 EQUAL DATE2 AND EQUAL DATE3 THEN                  01130000
001390         NEXT SENTENCE                                          01140000
001400     ELSE         15                                            01150000
001410         MOVE FAILSWCH TO TRIPSWCH                              01160000
001420         MOVE DATE1 TO DPLACE                                   01170000
001430         WRITE OUT-LINE FROM FAIL2CON                           01180000
001440         WRITE OUT-LINE FROM FAIL2CON2                          01190000
001450         MOVE DATE2 TO DPLACE                                   01200000
001460         WRITE OUT-LINE FROM FAIL2CON2                          01210000
001470         MOVE DATE3 TO DPLACE                                   01220000
001480         WRITE OUT-LINE FROM FAIL2CON2.                         01230000
001490     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME          16            01240000
001500     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001510     MOVE LCP-TIME-OF-DAY-68 TO TIME3.
```

Figure 4 (Part 2 of 4). ABJIVP01 - Converted Program

```
001520     IF (TIME1 LESS THAN TIME2 OR EQUAL TIME2) AND                01250000
001530        (TIME2 LESS THAN TIME3 OR EQUAL TIME3) THEN                01260000
001540         NEXT SENTENCE                                             01270000
001550     ELSE          17                                             01280000
001560         MOVE FAILSWCH TO TRIPSWCH                                 01290000
001570         MOVE TIME1 TO TPLACE                                      01300000
001580         WRITE OUT-LINE FROM FAIL3CON                              01310000
001590         WRITE OUT-LINE FROM FAIL3CON1                             01320000
001600         WRITE OUT-LINE FROM FAIL3CON2                             01330000
001610         MOVE TIME2 TO TPLACE                                      01340000
001620         WRITE OUT-LINE FROM FAIL3CON2                             01350000
001630         MOVE TIME3 TO TPLACE                                      01360000
001640         WRITE OUT-LINE FROM FAIL3CON2.                            01370000
001650 AFTER-THOUGHT.                                                    01380000
001660     MOVE ZERO TO TALLY         18                                 01390000
001670     INSPECT A-POEM TALLYING TALLY FOR ALL SPACES REPLACING ALL
001680                                       SPACES    BY "*"
001690     MOVE TALLY TO MY-COUNTER                                      01400000
001700     MOVE LINE1 OF A-POEM TO OUT-LINE WRITE OUT-LINE               01410000
001710     MOVE LINE2 OF A-POEM TO OUT-LINE WRITE OUT-LINE               01420000
001720     MOVE ZERO TO TALLY         19                                 01430000
001730     INSPECT A-POEM TALLYING TALLY FOR ALL "*".
001740     IF TALLY = MY-COUNTER                                         01440000
001750         MOVE "OK" TO OUT-LINE WRITE OUT-LINE                      01450000
001760     ELSE           20                                             01460000
001770         MOVE "BAH" TO OUT-LINE WRITE OUT-LINE.                    01470000
001780     MOVE ZERO TO TALLY         21                                 01480000
001790     INSPECT A-POEM TALLYING TALLY FOR ALL "E"
001800     PERFORM THREE-LINES                                           01490000
001810     MOVE ZERO TO TALLY         22                                 01500000
001820     INSPECT A-POEM TALLYING TALLY FOR CHARACTERS BEFORE "."
001830     PERFORM THREE-LINES                                           01510000
001840     MOVE ZERO TO TALLY         23                                 01520000
001850     INSPECT A-POEM TALLYING TALLY FOR LEADING "R"
001860     PERFORM THREE-LINES                                           01530000
001870     MOVE 2 TO I                                                   01540000
001880     MOVE ZERO TO TALLY         24                                 01550000
001890     INSPECT A-NUMBER(I) TALLYING TALLY FOR ALL "1"
001900     PERFORM THREE-LINES                                           01560000
001910     MOVE ZERO TO TALLY         25                                 01570000
001920     INSPECT A-NUMBER(I) TALLYING TALLY FOR LEADING "0" REPLACING
001930                                        LEADING "0" BY "2".
001940 THREE-LINES.                                                      01580000
001950     ADD TALLY TO MY-COUNTER.                                      01590000
001960     MOVE TALLY TO OUT-LINE WRITE OUT-LINE                         01600000
001970     MOVE MY-COUNTER TO OUT-LINE WRITE OUT-LINE.                   01610000
001980 THE-END.                                                          01620000
001990     IF TRIPSWCH EQUAL FAILSWCH OR MY-COUNTER NOT EQUAL 125        01630000
002000         WRITE OUT-LINE FROM FAILCON                               01640000
002010     ELSE         26                                               01650000
002020         WRITE OUT-LINE FROM SUCCESS.                              01660000
```

Figure 4 (Part 3 of 4). ABJIVP01 - Converted Program

```
002030      CLOSE PRINT-FILE.                                          01670000
002040      STOP RUN.                                                  01680000
002050 END PROGRAM ABJIVP01.          27
```

Figure 4 (Part 4 of 4). ABJIVP01 - Converted Program

6.3.1.3 CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01         05 FEB 1997  10:05:09  PAGE  1
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000010 IDENTIFICATION DIVISION.                                        00010000 000100
000020 PROGRAM-ID. ABJIVP01.                                           00020000
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:05:03.                        000200
*OLD** REMARKS.       1                                                00030000 000300   ABJ6011 00 REMARKS CHANGED TO COMMENT
000060*REMARKS.                                                         00030000 000300
*OLD**     THIS PROGRAM IS BEING WRITTEN TO TEST THE PROPER CONVERSION  00040000 000400
000070*    THIS PROGRAM IS BEING WRITTEN TO TEST THE PROPER CONVERSION  00040000 000400
*OLD**     FROM OS/VS COBOL SOURCE LANGUAGE TO IBM SOURCE LANGUAGE.     00050000 000500
000080*    FROM OS/VS COBOL SOURCE LANGUAGE TO IBM SOURCE LANGUAGE.     00050000 000500
*OLD** AUTHOR. XXXXXX.         2                                        00060000 000600   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000090*AUTHOR. XXXXXX.                                                  00060000 000600
*OLD** DATE-WRITTEN. JANUARY 24, 1983.       3                        00070000 000700   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000100*DATE-WRITTEN. JANUARY 24, 1983.                                  00070000 000700
000110                                                                 00080000 000800
*OLD**     NOTE - THE FOLLOWING AREAS ARE ADDRESSED       4            00090000 000900   ABJ6022 00 NOTE CHANGED TO COMMENT
000120*    NOTE - THE FOLLOWING AREAS ARE ADDRESSED                     00090000 000900
*OLD**         1  REMARKS                                              00100000 001000
000130*        1  REMARKS                                               00100000 001000
*OLD**         2  THEN                                                 00110000 001300
000140*        2  THEN                                                  00110000 001300
*OLD**         3  OTHERWISE                                            00120000 001400
000150*        3  OTHERWISE                                             00120000 001400
*OLD**         4  CURRENT-DATE                                         00130000 001500
000160*        4  CURRENT-DATE                                          00130000 001500
*OLD**         5  TIME-OF-DAY                                          00140000 001600
000170*        5  TIME-OF-DAY                                           00140000 001600
*OLD**         6  NOTE                                                 00150000 001700
000180*        6  NOTE                                                  00150000 001700
*OLD**         7  EXAMINE...TALLYING...REPLACING                       00160000 001800
000190*        7  EXAMINE.  TALLYING.  REPLACING                        00160000 001800
*OLD**         8  JUSTIFIED.                                           00170000 001900
000200*        8  JUSTIFIED.                                            00170000 001900
000210                                                                 00180000 002000
*OLD** DATE-COMPILED. TODAYS DATE.          5                        00190000 002100   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000220*DATE-COMPILED. TODAYS DATE.                                      00190000 002100
000230 EJECT                                                           00200000 002200
000240 ENVIRONMENT DIVISION.                                           00210000 002300
000250 INPUT-OUTPUT SECTION.                                           00220000 002400
000260 FILE-CONTROL.                                                   00230000 002500
000270     SELECT PRINT-FILE                                          00240000 002600
000280     ASSIGN TO UT-3330-S-DDPRINT.                               00250000 002700
000290 DATA DIVISION.                                                  00260000 002800
000300 FILE SECTION.                                                   00270000 002900
000310 FD  PRINT-FILE                                                  00280000 003000
*OLD**     RECORDING MODE IS F          6                            00290000 003100   ABJ6119 00 RECORDING MODE CLAUSE REMOVED
*OLD**     LABEL RECORDS ARE STANDARD                                00300000 003200   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
*OLD**     DATA RECORD IS OUT-LINE.                                  00310000 003300   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000320     .                                                          00300000 003300
000330 01  OUT-LINE           PIC X(80).                             00320000 003400
000340 WORKING-STORAGE SECTION.                                        00330000          ABJ6004 00 LCP-TIME-OF-DAY-68 GENERATED
000350 01  LCP-TIME-OF-DAY-68           PIC 9(6).        7                                    IN WORKING-STORAGE
```

Figure 5 (Part 1 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01         05 FEB 1997  10:05:09  PAGE  2
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000360 01  LCP-TIME-OF-DAY-74.                          8                          ABJ6002 00 LCP-CURRENT-DATE-68 GENERATED
000370     05  LCP-TIME-74              PIC 9(6).                                             IN WORKING-STORAGE
000380     05  FILLER                   PIC 9(2).
000390 01  LCP-CURRENT-DATE-68.                         9
000400     05  LCP-MONTH                PIC X(2).
000410     05  FILLER                   PIC X VALUE "/".
000420     05  LCP-DAY1                 PIC X(2).
000430     05  FILLER                   PIC X VALUE "/".
000440     05  LCP-YEAR                 PIC X(2).
000450 01  LCP-DATE-NEW-74.
000460     05  LCP-YEAR                 PIC X(2).
000470     05  LCP-MONTH                PIC X(2).
000480     05  LCP-DAY1                 PIC X(2).                                  003500
000490 77  MY-COUNTER         PIC 9(5)  VALUE 0.                        00340000 003600
000500 77  TRIPSWCH           PIC 9     VALUE 0.                        00350000 003700
000510 77  PASSWCH            PIC 9     VALUE 0.                        00360000 003800
000520 77  FAILSWCH           PIC 9     VALUE 1.                        00370000 003900
000530 77  CURRFLAG           PIC 9     VALUE 0.                        00380000 004100
000540 77  TOFDFLAG           PIC 9     VALUE 0.                        00390000 004200
000550 77  I                   PIC 9     VALUE 0.                        00400000 004300
000560 77  DATE1              PIC X(8)  VALUE SPACES.                   00410000 004400
000570 77  DATE2              PIC X(8)  VALUE SPACES.                   00420000 004500
000580 77  DATE3              PIC X(8)  VALUE SPACES.                   00430000 004600
000590 77  TIME1              PIC X(6)  VALUE SPACES.                   00440000 004700
000600 77  TIME2              PIC X(6)  VALUE SPACES.                   00450000 004800
000610 77  TIME3              PIC X(6)  VALUE SPACES.                   00460000 004900
000620                                                                  00470000 005400
000630 01  ORIGINAL-NUMBER.                                             00480000 005500
000640     05  FILLER     PIC 9(18) VALUE 0.                            00490000 005600
000650     05  FILLER     PIC 9(18) VALUE 0.                            00500000 005700
000660     05  FILLER     PIC 9(18) VALUE 000000009099843576.           00510000 005800
000670     05  FILLER     PIC 9(18) VALUE 121212121212121290.           00520000 005900
000680                                                                  00530000 006000
000690 01  THIS-DEF REDEFINES ORIGINAL-NUMBER.                          00540000 006100
000700     03  A-NUMBER OCCURS 2 TIMES.                                 00550000 006200
000710         05  LINE1      PIC 9(18).                                00560000 006300
000720         05  LINE2      PIC 9(18).                                00570000 006400
000730                                                                  00580000 006500
000740 01  A-POEM.                                                      00590000 006600
000750     03  LINE1.                                                   00600000 006700
000760         05  FILLER     PIC X(20) VALUE "ROSES ARE RED VIOLET".   00610000 006800
000770         05  FILLER     PIC X(20) VALUE "S ARE BLUE,         ".   00620000 006900
000780     03  LINE2.                                                   00630000 007000
000790         05  FILLER     PIC X(20) VALUE "SUGAR IS SWEET AND S".   00640000 007100
000800         05  FILLER     PIC X(20) VALUE "O ARE YOU.          ".   00650000 007200
000810                                                                  00660000 007300
000820                                                                  00670000 007900
000830 01  FAIL1CON2.                                                   00680000 008000
000840     03  FILLER         PIC XX    VALUE SPACES.                   00690000 008100
000850     03  CPLACE         PIC X(20) VALUE SPACES.                   00700000 008200
000860                                                                  00710000 008300
000870 01  FAIL2CON.                                                    00720000 008400
000880     03  FILLER         PIC X(20) VALUE "ALL THREE READINGS 0".   00730000 008500
```

Figure 5 (Part 2 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01         05 FEB 1997  10:05:09  PAGE  3
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000890     03  FILLER         PIC X(20) VALUE "F 'CURRENT-DATE' SHO".   00740000 008600
000900     03  FILLER         PIC X(20) VALUE "ULD BE THE SAME, BUT".   00750000 008700
000910     03  FILLER         PIC X(20) VALUE " THEY ARE:          ".   00760000 008800
000920                                                                  00770000 008900
000930 01  FAIL2CON2.                                                   00780000 009000
000940     03  FILLER         PIC XX    VALUE SPACES.                   00790000 009100
000950     03  DPLACE         PIC X(8)  VALUE SPACES.                   00800000 009200
000960                                                                  00810000 009300
000970 01  FAIL3CON.                                                    00820000 009400
000980     03  FILLER         PIC X(20) VALUE "THE THREE READINGS O".   00830000 009500
000990     03  FILLER         PIC X(20) VALUE "F 'TIME-OF-DAY' SHOU".   00840000 009600
001000     03  FILLER         PIC X(20) VALUE "LD BE EQUAL OR IN AS".   00850000 009700
001010     03  FILLER         PIC X(20) VALUE "CENDING ORDER,      ".   00860000 009800
001020                                                                  00870000 009900
001030 01  FAIL3CON1.                                                   00880000 010000
001040     03  FILLER         PIC X(20) VALUE "BUT THEY ARE:       ".   00890000 010100
001050                                                                  00900000 010200
001060 01  FAIL3CON2.                                                   00910000 010300
001070     03  FILLER         PIC XX    VALUE SPACES.                   00920000 010400
001080     03  TPLACE         PIC X(6)  VALUE SPACES.                   00930000 010500
001090                                                                  00940000 010600
001100 01  FAILCON.                                                     00950000 010700
001110     03  FILLER         PIC X(20) VALUE "TEST CASE SAMPLE   F".   00960000 010800
001120     03  FILLER         PIC X(20) VALUE "AILED.              ".   00970000 010900
001130                                                                  00980000 011000
001140 01  SUCCESS.                                                     00990000 011100
001150     03  FILLER         PIC X(20) VALUE "TEST CASE SAMPLE   I".   01000000 011200
001160     03  FILLER         PIC X(20) VALUE "S SUCCESSFUL.       ".   01010000 011300
001170 EJECT                                                           01020000 011400
001180 PROCEDURE DIVISION.                                             01030000 011500
001190 THIS-IS-A SECTION.                                              01040000 011600
001200 START-HERE.                                                     01050000 011700
*OLD**     MOVE TIME-OF-DAY TO TIME1        10                        01060000 011800
001210     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME                         01060000
001220     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001230     MOVE LCP-TIME-OF-DAY-68 TO TIME1                                    011800   ABJ6005 00 NEW CODE GENERATED FOR
                                                                                         TIME-OF-DAY
001240     OPEN OUTPUT PRINT-FILE                                      01070000 011900
*OLD**     MOVE CURRENT-DATE TO DATE1       11                        01080000 012000
001250     ACCEPT  LCP-DATE-NEW-74 FROM DATE                           01080000
001260     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001270     MOVE LCP-CURRENT-DATE-68 TO DATE1                                   012000   ABJ6003 00 NEW CODE GENERATED FOR
                                                                                         CURRENT-DATE
*OLD**     MOVE CURRENT-DATE TO DATE2       12                        01090000 012500
001280     ACCEPT  LCP-DATE-NEW-74 FROM DATE                           01090000
001290     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001300     MOVE LCP-CURRENT-DATE-68 TO DATE2                                   012500   ABJ6003 00 NEW CODE GENERATED FOR
                                                                                         CURRENT-DATE
*OLD**     MOVE CURRENT-DATE TO DATE3.      13                        01100000 012700
001310     ACCEPT  LCP-DATE-NEW-74 FROM DATE                           01100000
001320     MOVE CORRESPONDING LCP-DATE-NEW-74 TO LCP-CURRENT-DATE-68
001330     MOVE LCP-CURRENT-DATE-68 TO DATE3.                                  012700   ABJ6003 00 NEW CODE GENERATED FOR
                                                                                         CURRENT-DATE
```

Figure 5 (Part 3 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01         05 FEB 1997  10:05:09  PAGE  4
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

001340                                                                  01110000 012800
*OLD**     MOVE TIME-OF-DAY TO TIME2.       14                         01120000 014000
001350     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME                         01120000
001360     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001370     MOVE LCP-TIME-OF-DAY-68 TO TIME2.                                    014000   ABJ6005 00 NEW CODE GENERATED FOR
                                                                                             TIME-OF-DAY
001380     IF DATE1 EQUAL DATE2 AND EQUAL DATE3 THEN                   01130000 014100
001390         NEXT SENTENCE                                           01140000 014200
*OLD**     OTHERWISE                        15                         01150000 014300   ABJ6021 00 OTHERWISE REPLACED BY ELSE
001400     ELSE                                                        01150000 014300
001410         MOVE FAILSWCH TO TRIPSWCH                               01160000 014400
001420         MOVE DATE1 TO DPLACE                                    01170000 014500
001430         WRITE OUT-LINE FROM FAIL2CON                            01180000 014600
001440         WRITE OUT-LINE FROM FAIL2CON2                           01190000 014700
001450         MOVE DATE2 TO DPLACE                                    01200000 014800
001460         WRITE OUT-LINE FROM FAIL2CON2                           01210000 014900
001470         MOVE DATE3 TO DPLACE                                    01220000 015000
001480         WRITE OUT-LINE FROM FAIL2CON2.                          01230000 015100
*OLD**     MOVE TIME-OF-DAY TO TIME3.       16                         01240000 015200
001490     ACCEPT LCP-TIME-OF-DAY-74 FROM TIME                         01240000
001500     MOVE LCP-TIME-74 TO LCP-TIME-OF-DAY-68
001510     MOVE LCP-TIME-OF-DAY-68 TO TIME3.                                    015200   ABJ6005 00 NEW CODE GENERATED FOR
                                                                                             TIME-OF-DAY
001520     IF (TIME1 LESS THAN TIME2 OR EQUAL TIME2) AND               01250000 015300
001530        (TIME2 LESS THAN TIME3 OR EQUAL TIME3) THEN              01260000 015400
001540         NEXT SENTENCE                                           01270000 015500
*OLD**     OTHERWISE                        17                         01280000 015600   ABJ6021 00 OTHERWISE REPLACED BY ELSE
001550     ELSE                                                        01280000 015600
001560         MOVE FAILSWCH TO TRIPSWCH                               01290000 015700
001570         MOVE TIME1 TO TPLACE                                    01300000 015800
001580         WRITE OUT-LINE FROM FAIL3CON                            01310000 015900
001590         WRITE OUT-LINE FROM FAIL3CON1                           01320000 016000
001600         WRITE OUT-LINE FROM FAIL3CON2                           01330000 016100
001610         MOVE TIME2 TO TPLACE                                    01340000 016200
001620         WRITE OUT-LINE FROM FAIL3CON2                           01350000 016300
001630         MOVE TIME3 TO TPLACE                                    01360000 016400
001640         WRITE OUT-LINE FROM FAIL3CON2.                          01370000 016500
001650 AFTER-THOUGHT.                                                   01380000 016600
*OLD**     EXAMINE A-POEM TALLYING ALL SPACES REPLACING BY "*"    18           016700   ABJ6018 00 TALLY IS INITIALIZED
001660     MOVE ZERO TO TALLY                                          01390000         ABJ6019 00 EXAMINE REPLACED BY INSPECT
001670     INSPECT A-POEM TALLYING TALLY FOR ALL SPACES REPLACING ALL          016700
001680                                       SPACES    BY "*"                  016700
001690     MOVE TALLY TO MY-COUNTER                                    01400000 016800
001700     MOVE LINE1 OF A-POEM TO OUT-LINE WRITE OUT-LINE             01410000 016900
001710     MOVE LINE2 OF A-POEM TO OUT-LINE WRITE OUT-LINE             01420000 017000
*OLD**     EXAMINE A-POEM TALLYING ALL "*".         19                         017100   ABJ6018 00 TALLY IS INITIALIZED
001720     MOVE ZERO TO TALLY                                          01430000         ABJ6019 00 EXAMINE REPLACED BY INSPECT
001730     INSPECT A-POEM TALLYING TALLY FOR ALL "*".                          017100
001740     IF TALLY = MY-COUNTER                                       01440000 017200
001750         MOVE "OK" TO OUT-LINE WRITE OUT-LINE                    01450000 017300
*OLD**     OTHERWISE          20                                               017400   ABJ6021 00 OTHERWISE REPLACED BY ELSE
001760     ELSE                                                        01460000 017400
001770         MOVE "BAH" TO OUT-LINE WRITE OUT-LINE.                  01470000 017500
```

Figure 5 (Part 4 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01         05 FEB 1997  10:05:09  PAGE  5
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

*OLD**     EXAMINE A-POEM TALLYING ALL "E"         21                 01480000 017600   ABJ6018 00 TALLY IS INITIALIZED
001780     MOVE ZERO TO TALLY                                         01480000          ABJ6019 00 EXAMINE REPLACED BY INSPECT
001790     INSPECT A-POEM TALLYING TALLY FOR ALL "E"                          017600
001800     PERFORM THREE-LINES                                        01490000 017700
*OLD**     EXAMINE A-POEM TALLYING UNTIL FIRST "."          22        01500000 017800   ABJ6018 00 TALLY IS INITIALIZED
001810     MOVE ZERO TO TALLY                                         01500000          ABJ6019 00 EXAMINE REPLACED BY INSPECT
001820     INSPECT A-POEM TALLYING TALLY FOR CHARACTERS BEFORE "."            017800
001830     PERFORM THREE-LINES                                        01510000 017900
*OLD**     EXAMINE A-POEM TALLYING LEADING "R"        23              01520000 018000   ABJ6018 00 TALLY IS INITIALIZED
001840     MOVE ZERO TO TALLY                                         01520000          ABJ6019 00 EXAMINE REPLACED BY INSPECT
001850     INSPECT A-POEM TALLYING TALLY FOR LEADING "R"                      018000
001860     PERFORM THREE-LINES                                        01530000 018100
001870     MOVE 2 TO I                                                01540000 018200
*OLD**     EXAMINE A-NUMBER(I) TALLYING ALL 1         24              01550000 018300   ABJ6018 00 TALLY IS INITIALIZED
001880     MOVE ZERO TO TALLY                                         01550000          ABJ6019 00 EXAMINE REPLACED BY INSPECT
001890     INSPECT A-NUMBER(I) TALLYING TALLY FOR ALL "1"                     018300
001900     PERFORM THREE-LINES                                        01560000 018400
*OLD**     EXAMINE A-NUMBER(I) TALLYING LEADING 0 REPLACING BY 2.   25        018500   ABJ6018 00 TALLY IS INITIALIZED
001910     MOVE ZERO TO TALLY                                         01570000          ABJ6019 00 EXAMINE REPLACED BY INSPECT
001920     INSPECT A-NUMBER(I) TALLYING TALLY FOR LEADING "0" REPLACING       018500
001930                                         LEADING "0" BY "2".            018500
001940 THREE-LINES.                                                 01580000 018600
001950     ADD TALLY TO MY-COUNTER.                                01590000 018700
001960     MOVE TALLY TO OUT-LINE WRITE OUT-LINE                   01600000 018800
001970     MOVE MY-COUNTER TO OUT-LINE WRITE OUT-LINE.             01610000 018900
001980 THE-END.                                                     01620000 019000
001990     IF TRIPSWCH EQUAL FAILSWCH OR MY-COUNTER NOT EQUAL 125  01630000 019100
002000         WRITE OUT-LINE FROM FAILCON                         01640000 019200
*OLD**     OTHERWISE         26                                    01650000 019300   ABJ6021 00 OTHERWISE REPLACED BY ELSE
002010     ELSE                                                    01650000 019300
002020         WRITE OUT-LINE FROM SUCCESS.                        01660000 019400
002030     CLOSE PRINT-FILE.                                       01670000 019500
002040     STOP RUN.                                               01680000          ABJ6126 99 *----------------------------*
002050 END PROGRAM ABJIVP01.          27                                   019600              *  END OF COBOL CONVERSION   *
                                                                                               * 5785-CCC COBOL CONVERSION  *
```

Figure 5 (Part 5 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

```
                                                                                *--------------------------*
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01        05 FEB 1997  10:05:09  PAGE  6
CONVERSION FROM DOS/VS COBOL      TO  COBOL/VSE
OPTIONS IN EFFECT :           28
  PROCEDURE NAME CHECKING ........ YES   LANGLEVEL ..................... DOS/VS COBOL
  FLAG REPORT WRITER STMTS ....... YES   CICS ........................... NO CICS ST.
  REMOVE OBSOLETE ELEMENTS ....... YES   LINE COUNT ..................... 60
  GENERATE CALL ILBOABNO STMT..... YES   DATE FORMAT ................... MM/DD/YY
  GENERATE END PROGRAM STMT ...... YES   RESEQUENCING ................... YES
  POST-CONVERSION COMPILE ........ NO    INCREMENT ..................... 0010
  MANUAL CHANGE FLAGGING ......... YES   RESERVED WORD SUFFIX ........... 74
  HANDLE EXEC SQL INCLUDE AS COPY. YES   GENERATE NEW PROGRAM ........... YES
  REMOVE NON 88 VALUE CLAUSE IN FS YES   GENERATE NEW COPY ............. YES
  FLAG IF FILE-STATUS (NOT) = "00" YES   REPLACE LIKE-NAMED COPY MBR .... YES
  FLAG 31-BIT ADDRESS ARITHMETIC.. YES   PRINT REFERENCE SOURCE LINE .... YES
  INCL.W-S IN CICS COMPILE OF L-S. YES   PRINT COPY MODULE ............. YES
  OPTION-13 ...................... NO    LEVEL DIAGNOSTIC ............... 00
  OPTION-14 ...................... NO    DEBUG MODE ..................... 0
  OPTION-15 ...................... NO    SQL ............................ N
HIGHEST SEVERITY MESSAGE FOR THIS CONVERSION:   00        29
       0034 MESSAGES ISSUED
       0034 MESSAGES PRINTED
SEQNBR  CPYNBR    MSGID   RC  MESSAGE TEXT

000060           ABJ6011  00  REMARKS CHANGED TO COMMENT
000090           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000100           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000120           ABJ6022  00  NOTE CHANGED TO COMMENT
000220           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000320           ABJ6119  00  RECORDING MODE CLAUSE REMOVED
000320           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000320           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000340           ABJ6004  00  LCP-TIME-OF-DAY-68 GENERATED IN WORKING-STORAGE
000340           ABJ6002  00  LCP-CURRENT-DATE-68 GENERATED IN WORKING-STORAGE
001230           ABJ6005  00  NEW CODE GENERATED FOR TIME-OF-DAY
001270           ABJ6003  00  NEW CODE GENERATED FOR CURRENT-DATE
001300           ABJ6003  00  NEW CODE GENERATED FOR CURRENT-DATE
001330           ABJ6003  00  NEW CODE GENERATED FOR CURRENT-DATE
001370           ABJ6005  00  NEW CODE GENERATED FOR TIME-OF-DAY
001400           ABJ6021  00  OTHERWISE REPLACED BY ELSE
001510           ABJ6005  00  NEW CODE GENERATED FOR TIME-OF-DAY
001550           ABJ6021  00  OTHERWISE REPLACED BY ELSE
001660           ABJ6018  00  TALLY IS INITIALIZED
001660           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001720           ABJ6018  00  TALLY IS INITIALIZED
001720           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001760           ABJ6021  00  OTHERWISE REPLACED BY ELSE
001780           ABJ6018  00  TALLY IS INITIALIZED
001780           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001810           ABJ6018  00  TALLY IS INITIALIZED
001810           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001840           ABJ6018  00  TALLY IS INITIALIZED
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP01        05 FEB 1997  10:05:09  PAGE  7


001840           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001880           ABJ6018  00  TALLY IS INITIALIZED
001880           ABJ6019  00  EXAMINE REPLACED BY INSPECT
001910           ABJ6018  00  TALLY IS INITIALIZED
001910           ABJ6019  00  EXAMINE REPLACED BY INSPECT
002010           ABJ6021  00  OTHERWISE REPLACED BY ELSE
```

Figure 5 (Part 6 of 6). ABJIVP01 - CCCA/VSE Diagnostics Report

Notes:

1 **2** **3** **4** **5** The input source program contains remarks, author, note and date compiled information within the IDENTIFICATION DIVISION of the program. If you specify Y for the Remove obsolete elements option on conversion Options panel 2, these paragraphs are commented out.

6 The target language compilers ignore this clause, if it is specified for a VSAM file. If the clause is in a file description entry for a VSAM file or a file that is to be converted to VSAM, it is removed.

7 **8** **10** **14** **16** The TIME-OF-DAY register is not supported by the target languages. Wherever TIME-OF-DAY is referenced in the program, it is replaced by code that obtains the time from the system and puts it in the format of the TIME-OF-DAY register. The fields required for the reformatting are generated in the WORKING STORAGE section. For non-CICS programs, the ACCEPT...FROM TIME statement is used to obtain the time.

9 **11** **12** **13** The CURRENT-DATE register is not supported by the target languages. Wherever CURRENT-DATE is referenced in the program, it is replaced by code that obtains the date from the system and puts it in the format of the CURRENT-DATE register. The fields required for the reformatting are generated in the WORKING- STORAGE section. For non-CICS programs, the ACCEPT... FROM DATE statement is used to obtain the date.

15 **17** **20** **26** OTHERWISE is replaced by ELSE.

18 **19** **21** **22** **23** **24** **25** The EXAMINE statement is changed to an INSPECT statement and the statement MOVE ZERO TO TALLY is put in front of it.

27 Under the ANSI 85 standard, control can not flow beyond the last line of a called subprogram. The conversion generates an implicit END PROGRAM at the end of each program.

28 This section of the report shows the conversion options that were in effect during processing. These options were obtained from the CCCA/VSE panels completed prior to job submission.

29 This section of the report shows the highest severity for this conversion. In this case the highest level was 0, which means the conversion was completed successfully without taking any manual actions.

6.3.2 ABJIVP02 - Batch COBOL Program with Copy Members

6.3.2.1 Input Source Program

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID. ABJIVP02.
000300 REMARKS.
000600 ---
000700     THIS PROGRAM COMPUTES THE GROSS SALARY, TAX AND NET SALARY
000800     OF A GROUP OF EMPLOYEES.
001100 AUTHOR. YOUR NAME FOLLOWED BY A PERIOD.
001200 INSTALLATION. IBM-370.
001300 DATE-WRITTEN. FEB 27,1981.
001400*
001500     NOTE - THE FOLLOWING AREAS ARE ADDRESSED
001600         1  REMARKS
001700         2  NOTE
001800*        3  COPY FOR LANGLVL(1).
001900*
002000 DATE-COMPILED. TODAYS DATE.
002100 EJECT
002200 ENVIRONMENT DIVISION.
002300 CONFIGURATION SECTION.
002400*SOURCE-COMPUTER. IBM-370.
002500*OBJECT-COMPUTER. IBM-370.
002600 INPUT-OUTPUT SECTION.
002700 FILE-CONTROL.
002800     SELECT PRINT-OUT ASSIGN TO UR-2540R-S-PRINT.
002900 DATA DIVISION.
002910*
003000 FILE SECTION.
003100 FD  PRINT-OUT
003200     LABEL RECORDS ARE OMITTED          1
003300     DATA RECORDS ARE OUTPUT-RECORD ENTRY-DET.
003500 01  OUTPUT-RECORD COPY ABJL901.        2
004400 01  COPY ABJL902 REPLACING STEMPL BY PREMPL STHOURS BY PRHOURS          3
004500         STSALARY BY PRSALARY STTAX BY PRTAX STNET BY PRNET.
004600*
005400 WORKING-STORAGE SECTION.
005500*
005600 77  NUM-OF-ITEMS COPY ABJL903.         4
005700*
006500 77  COPY ABJL903A.                     5
006600*
007300 77  WORK-TAX        PIC 9(3)V9(4).
007400 77  WORK-NET        PIC 9(3)V9(4).
007500 77  SUB1            PIC 99     VALUE 1.
007600 77  ERROR-FLAG      PIC 9      VALUE 0.
007700 01  INPUT-AREA.
007800     03  ENTRYA.
007900         06  FILLER  PIC X      VALUE "A".
008000         06  FILLER  PIC 99     VALUE 40.
008100     03  ENTRYB.
008200         06  FILLER  PIC X      VALUE "B".
```

Figure 6 (Part 1 of 3). ABJIVP02 - Source Program

```
008300         06  FILLER  PIC 99     VALUE 41.
008400     03  ENTRYC.
008500         06  FILLER  PIC X      VALUE "C".
008600         06  FILLER  PIC 99     VALUE 39.
008700     03  ENTRYD.
008800         06  FILLER  PIC X      VALUE "D".
008900         06  FILLER  PIC 99     VALUE 16.
009000     03  ENTRYE.
009100         06  FILLER  PIC X      VALUE "E".
009200         06  FILLER  PIC 99     VALUE 21.
009300     03  ENTRYF.
009400         06  FILLER  PIC X      VALUE "F".
009500         06  FILLER  PIC 99     VALUE 44.
009600     03  ENTRYG.
009700         06  FILLER  PIC X      VALUE "G".
009800         06  FILLER  PIC 99     VALUE 55.
009900     03  ENTRYH.
010000         06  FILLER  PIC X      VALUE "H".
010100         06  FILLER  PIC 99     VALUE 60.
010200     03  ENTRYI.
010300         06  FILLER  PIC X      VALUE "I".
010400         06  FILLER  PIC 99     VALUE 41.
010500     03  ENTRYJ.
010600         06  FILLER  PIC X      VALUE "J".
010700         06  FILLER  PIC 99     VALUE 42.
010800     03  ENTRYK.
010900         06  FILLER  PIC X      VALUE "K".
011000         06  FILLER  PIC 99     VALUE 39.
011100     03  ENTRYL.
011200         06  FILLER  PIC X      VALUE "L".
011300         06  FILLER  PIC 99     VALUE 32.
011400*
011500 01  COPY ABJL9O4 REPLACING A BY REDEF-AREA B BY INPUT-AREA.          6
011600*
012400 01  HDG-1.
012500     03  FILLER      PIC X(8)   VALUE SPACES.
012600     03  FILLER      PIC X(21)  VALUE "_____________________".
012700     03  FILLER      PIC X(21)  VALUE "_____________________".
012800     03  FILLER      PIC X(82)  VALUE SPACES.
012900 01  HDG-2.
013000     03  FILLER      PIC X(8)   VALUE SPACES.
013100     03  FILLER      PIC X(10)  VALUE SPACES.
013200     03  FILLER      PIC X(16)  VALUE "HOURS   GROSS   ".
013300     03  FILLER      PIC X(13)  VALUE "TAX       NET".
013400     03  FILLER      PIC X(85)  VALUE SPACES.
013500 01  HDG-3.
013600     03  FILLER      PIC X(8)   VALUE SPACES.
013700     03  FILLER      PIC X(10)  VALUE "EMPLOYEE  ".
013800     03  FILLER      PIC X(8)   VALUE "WORKED  ".
013900     03  FILLER      PIC X(8)   VALUE "SALARY  ".
014000     03  FILLER      PIC X(10)  VALUE "DEDUCTED  ".
```

Figure 6 (Part 2 of 3). ABJIVP02 - Source Program

```
014100     03  FILLER      PIC X(6)   VALUE "SALARY".
014200     03  FILLER      PIC X(82)  VALUE SPACES.
014300 01  HDG-4.
014400     03  FILLER      PIC X(8)   VALUE SPACES.
014500     03  FILLER      PIC X(10)  VALUE "________ ".
014600     03  FILLER      PIC X(8)   VALUE "_______ ".
014700     03  FILLER      PIC X(8)   VALUE "_______ ".
014800     03  FILLER      PIC X(10)  VALUE "________ ".
014900     03  FILLER      PIC X(6)   VALUE "______".
015000     03  FILLER      PIC X(82)  VALUE SPACES.
015100 PROCEDURE DIVISION.
015200     OPEN OUTPUT PRINT-OUT.
015300     WRITE OUTPUT-RECORD FROM HDG-1.
015400     WRITE OUTPUT-RECORD FROM HDG-2.
015500     WRITE OUTPUT-RECORD FROM HDG-3.
015600     WRITE OUTPUT-RECORD FROM HDG-4.
015700     PERFORM PROCESS THRU PROCESS2 VARYING SUB1 FROM 1 BY 1
015800         UNTIL SUB1 GREATER THAN NUM-OF-ITEMS.
015900     WRITE OUTPUT-RECORD FROM HDG-4.
016000     GO TO EOJ-ROUTINE.
016100 PROCESS.
016200     MOVE SPACES TO ENTRY-DET.
016300     MOVE EMPLOYEE(SUB1) TO PREMPL.
016400     MOVE HOURS-WORK(SUB1) TO PRHOURS.
016500     COMPUTE WORK-GROSS ROUNDED = HOURS-WORK(SUB1) * 4.00.
016600     MOVE WORK-GROSS TO PRSALARY.
016700     IF WORK-GROSS GREATER THAN 150.00
016800         COMPUTE WORK-TAX ROUNDED = (WORK-GROSS - 150) * .2 + 5
016900         GO TO PROCESS2.
017000     IF WORK-GROSS NOT LESS THAN 100.00
017100         COMPUTE WORK-TAX = (WORK-GROSS - 100) * .1
017200         GO TO PROCESS2.
017300     MOVE ZEROS TO WORK-TAX.
017400 PROCESS2.
017500     MOVE WORK-TAX TO PRTAX
017600         COMPUTE WORK-NET = WORK-GROSS - WORK-TAX
017700         MOVE WORK-NET TO PRNET
017800         WRITE ENTRY-DET.
017900 EOJ-ROUTINE.
018000     IF ERROR-FLAG = ZERO
018100         MOVE "TEST CASE LCPTST09 IS SUCCESSFUL." TO OUTPUT-RECORD
018200         WRITE OUTPUT-RECORD
018300     OTHERWISE
018400         MOVE "TEST CASE LCPTST09 FAILED." TO OUTPUT-RECORD
018500         WRITE OUTPUT-RECORD.
018600     CLOSE PRINT-OUT.
018700     STOP RUN.
```

Figure 6 (Part 3 of 3). ABJIVP02 - Source Program

6.3.2.2 Output Program from CCCA/VSE Conversion

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. ABJIVP02.
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:07:10.
000060*REMARKS.
000070*---
000080*    THIS PROGRAM COMPUTES THE GROSS SALARY, TAX AND NET SALARY
000090*    OF A GROUP OF EMPLOYEES.
000100*AUTHOR. YOUR NAME FOLLOWED BY A PERIOD.
000110*INSTALLATION. IBM-370.
000120*DATE-WRITTEN. FEB 27,1981.
000130*
000140*    NOTE - THE FOLLOWING AREAS ARE ADDRESSED
000150*        1  REMARKS
000160*        2  NOTE
000170*        3  COPY FOR LANGLVL(1).
000180*
000190*DATE-COMPILED. TODAYS DATE.
000200 EJECT
000210 ENVIRONMENT DIVISION.
000220 CONFIGURATION SECTION.
000230*SOURCE-COMPUTER. IBM-370.
000240*OBJECT-COMPUTER. IBM-370.
000250 INPUT-OUTPUT SECTION.
000260 FILE-CONTROL.
000270     SELECT PRINT-OUT ASSIGN TO UR-2540R-S-PRINT.
000280 DATA DIVISION.
000290*
000300 FILE SECTION.
000310 FD  PRINT-OUT          1
000320     .
000330 01  OUTPUT-RECORD COPY ABJL901 REPLACING ==01  STD-LINE== BY       2
000340                                          ==    ==.
000350 COPY ABJL902 REPLACING STEMPL BY PREMPL STHOURS BY PRHOURS         3
000360         STSALARY BY PRSALARY STTAX BY PRTAX STNET BY PRNET.
000370*
000380 WORKING-STORAGE SECTION.
000390*
000400 77  NUM-OF-ITEMS COPY ABJL903 REPLACING ==77  A== BY ==    ==.        4
000410*
000420 COPY ABJL903A.           5
000430*
000440 77  WORK-TAX        PIC 9(3)V9(4).
000450 **** No Change - same as source ********************
               .
               .
               .
000820     03  ENTRYL.
000830         06  FILLER  PIC X      VALUE "L".
000840         06  FILLER  PIC 99     VALUE 32.
```

Figure 7 (Part 1 of 2). ABJIVP02 - Converted Program

```
000850*
000860 COPY ABJL904 REPLACING A BY REDEF-AREA B BY INPUT-AREA.          6
000870*
000880 01  HDG-1.
000890 **** No Change - same as source ********************
                .
                .
                .
001510     STOP RUN.
```

Figure 7 (Part 2 of 2). ABJIVP02 - Converted Program

6.3.2.3 CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP02         05 FEB 1997  10:07:16  PAGE  1
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000010 IDENTIFICATION DIVISION.                                                 000100
000020 PROGRAM-ID. ABJIVP02.
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:07:10.                        000200
*OLD** REMARKS.                                                               000300   ABJ6011 00 REMARKS CHANGED TO COMMENT
000060*REMARKS.                                                                000300
*OLD** ---                                                                    000600
000070*---                                                                     000600
*OLD**     THIS PROGRAM COMPUTES THE GROSS SALARY, TAX AND NET SALARY         000700
000080*    THIS PROGRAM COMPUTES THE GROSS SALARY, TAX AND NET SALARY          000700
*OLD**     OF A GROUP OF EMPLOYEES.                                           000800
000090*    OF A GROUP OF EMPLOYEES.                                            000800
*OLD** AUTHOR. YOUR NAME FOLLOWED BY A PERIOD.                                001100   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000100*AUTHOR. YOUR NAME FOLLOWED BY A PERIOD.                                 001100
*OLD** INSTALLATION. IBM-370.                                                 001200   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000110*INSTALLATION. IBM-370.                                                  001200
*OLD** DATE-WRITTEN. FEB 27,1981.                                             001300   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000120*DATE-WRITTEN. FEB 27,1981.                                              001300
000130*                                                                        001400
*OLD**     NOTE - THE FOLLOWING AREAS ARE ADDRESSED                           001500   ABJ6022 00 NOTE CHANGED TO COMMENT
000140*    NOTE - THE FOLLOWING AREAS ARE ADDRESSED                            001500
*OLD**         1  REMARKS                                                     001600
000150*        1  REMARKS                                                      001600
*OLD**         2  NOTE                                                        001700
000160*        2  NOTE                                                         001700
000170*        3  COPY FOR LANGLVL(1).                                        001800
000180*                                                                        001900
*OLD** DATE-COMPILED. TODAYS DATE.                                            002000   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000190*DATE-COMPILED. TODAYS DATE.                                             002000
000200 EJECT                                                                    002100
000210 ENVIRONMENT DIVISION.                                                    002200
000220 CONFIGURATION SECTION.                                                   002300
000230*SOURCE-COMPUTER. IBM-370.                                               002400
000240*OBJECT-COMPUTER. IBM-370.                                               002500
000250 INPUT-OUTPUT SECTION.                                                    002600
000260 FILE-CONTROL.                                                            002700
000270     SELECT PRINT-OUT ASSIGN TO UR-2540R-S-PRINT.                        002800
000280 DATA DIVISION.                                                           002900
000290*                                                                        002910
000300 FILE SECTION.                                                            003000
000310 FD  PRINT-OUT                                                            003100
*OLD**     LABEL RECORDS ARE OMITTED       1                                  003200   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
*OLD**     DATA RECORDS ARE OUTPUT-RECORD ENTRY-DET.                          003300   ABJ6181 00 OBSOLETE ELEMENT IS REMOVED
000320     .                                                                   003300
*OLD** 01  OUTPUT-RECORD COPY ABJL901.     2                                 003500   ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
000330 01  OUTPUT-RECORD COPY ABJL901 REPLACING ==01  STD-LINE== BY           003500
000340                                           ==    ==.                    003500
000010 01  STD-LINE   PIC X(132).                                             002400 +
*OLD** 01  COPY ABJL902 REPLACING STEMPL BY PREMPL STHOURS BY PRHOURS       3 004400   ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
000350 COPY ABJL902 REPLACING STEMPL BY PREMPL STHOURS BY PRHOURS            004400
000360         STSALARY BY PRSALARY STTAX BY PRTAX STNET BY PRNET.           004500
000010 01  ENTRY-DET.                                                         000100 +
```

Figure 8 (Part 1 of 5). ABJIVP02 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP02         05 FEB 1997  10:07:16  PAGE  2
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000020     03  FILLER      PIC X(8).                                          000200 +
000030     03  FILLER      PIC X(3).                                          000300 +
000040     03  STEMPL      PIC X.                                             000400 +
000050     03  FILLER      PIC X(8).                                          000500 +
000060     03  STHOURS     PIC 99.                                            000600 +
000070     03  FILLER      PIC X(4).                                          000700 +
000080     03  STSALARY    PIC ZZZ.99.                                        000800 +
000090     03  FILLER      PIC X(2).                                          000900 +
000100     03  STTAX       PIC ZZZ.99.                                        001000 +
000110     03  FILLER      PIC X(4).                                          001100 +
000120     03  STNET       PIC ZZZ.99.                                        001200 +
000130     03  FILLER      PIC X(82).                                         001300 +
000370*                                                                       004600
000380 WORKING-STORAGE SECTION.                                                005400
000390*                                                                       005500
*OLD** 77  NUM-OF-ITEMS COPY ABJL903.       4                               005600   ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
000400 77  NUM-OF-ITEMS COPY ABJL903 REPLACING ==77  A== BY ==    ==.          005600
000010 77  A    PIC 99     VALUE 12.                                          002700 +
000410*                                                                       005700
*OLD** 77  COPY ABJL903A.           5                                       006500   ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
000420 COPY ABJL903A.                                                         006500
000010 77  WORK-GROSS      PIC 9(3)V9(4).                                     002800 +
000430*                                                                       006600
000440 77  WORK-TAX        PIC 9(3)V9(4).                                     007300
000450 77  WORK-NET        PIC 9(3)V9(4).                                     007400
000460 77  SUB1            PIC 99     VALUE 1.                                007500
000470 77  ERROR-FLAG      PIC 9      VALUE 0.                                007600
000480 01  INPUT-AREA.                                                        007700
000490     03  ENTRYA.                                                        007800
000500         06  FILLER  PIC X      VALUE "A".                             007900
000510         06  FILLER  PIC 99     VALUE 40.                              008000
000520     03  ENTRYB.                                                        008100
000530         06  FILLER  PIC X      VALUE "B".                             008200
000540         06  FILLER  PIC 99     VALUE 41.                              008300
000550     03  ENTRYC.                                                        008400
000560         06  FILLER  PIC X      VALUE "C".                             008500
000570         06  FILLER  PIC 99     VALUE 39.                              008600
000580     03  ENTRYD.                                                        008700
000590         06  FILLER  PIC X      VALUE "D".                             008800
000600         06  FILLER  PIC 99     VALUE 16.                              008900
000610     03  ENTRYE.                                                        009000
000620         06  FILLER  PIC X      VALUE "E".                             009100
000630         06  FILLER  PIC 99     VALUE 21.                              009200
000640     03  ENTRYF.                                                        009300
000650         06  FILLER  PIC X      VALUE "F".                             009400
000660         06  FILLER  PIC 99     VALUE 44.                              009500
000670     03  ENTRYG.                                                        009600
000680         06  FILLER  PIC X      VALUE "G".                             009700
000690         06  FILLER  PIC 99     VALUE 55.                              009800
000700     03  ENTRYH.                                                        009900
000710         06  FILLER  PIC X      VALUE "H".                             010000
000720         06  FILLER  PIC 99     VALUE 60.                              010100
000730     03  ENTRYI.                                                        010200
```

Figure 8 (Part 2 of 5). ABJIVP02 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP02         05 FEB 1997  10:07:16  PAGE  3
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000740         06  FILLER  PIC X      VALUE "I".                                 010300
000750         06  FILLER  PIC 99     VALUE 41.                                  010400
000760     03  ENTRYJ.                                                           010500
000770         06  FILLER  PIC X      VALUE "J".                                 010600
000780         06  FILLER  PIC 99     VALUE 42.                                  010700
000790     03  ENTRYK.                                                           010800
000800         06  FILLER  PIC X      VALUE "K".                                 010900
000810         06  FILLER  PIC 99     VALUE 39.                                  011000
000820     03  ENTRYL.                                                           011100
000830         06  FILLER  PIC X      VALUE "L".                                 011200
000840         06  FILLER  PIC 99     VALUE 32.                                  011300
000850*                                                                           011400
*OLD** 01  COPY ABJL904 REPLACING A BY REDEF-AREA B BY INPUT-AREA.          6    011500   ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
000860 COPY ABJL904 REPLACING A BY REDEF-AREA B BY INPUT-AREA.                     011500
000010 01  A REDEFINES B.                                                        007000 +
000020     03  ENTRY-ITEM OCCURS 12 TIMES.                                       007100 +
000030         06 EMPLOYEE    PIC X.                                            007200 +
000040         06 HOURS-WORK  PIC 99.                                            007300 +
000870*                                                                           011600
000880 01  HDG-1.                                                                012400
000890     03  FILLER      PIC X(8)   VALUE SPACES.                              012500
000900     03  FILLER      PIC X(21)  VALUE "_____________________".             012600
000910     03  FILLER      PIC X(21)  VALUE "_____________________".             012700
000920     03  FILLER      PIC X(82)  VALUE SPACES.                              012800
000930 01  HDG-2.                                                                012900
000940     03  FILLER      PIC X(8)   VALUE SPACES.                              013000
000950     03  FILLER      PIC X(10)  VALUE SPACES.                              013100
000960     03  FILLER      PIC X(16)  VALUE "HOURS   GROSS   ".                  013200
000970     03  FILLER      PIC X(13)  VALUE "TAX       NET".                     013300
000980     03  FILLER      PIC X(85)  VALUE SPACES.                              013400
000990 01  HDG-3.                                                                013500
001000     03  FILLER      PIC X(8)   VALUE SPACES.                              013600
001010     03  FILLER      PIC X(10)  VALUE "EMPLOYEE  ".                        013700
001020     03  FILLER      PIC X(8)   VALUE "WORKED  ".                          013800
001030     03  FILLER      PIC X(8)   VALUE "SALARY  ".                          013900
001040     03  FILLER      PIC X(10)  VALUE "DEDUCTED  ".                        014000
001050     03  FILLER      PIC X(6)   VALUE "SALARY".                            014100
001060     03  FILLER      PIC X(82)  VALUE SPACES.                              014200
001070 01  HDG-4.                                                                014300
001080     03  FILLER      PIC X(8)   VALUE SPACES.                              014400
001090     03  FILLER      PIC X(10)  VALUE "________  ".                        014500
001100     03  FILLER      PIC X(8)   VALUE "______  ".                          014600
001110     03  FILLER      PIC X(8)   VALUE "______  ".                          014700
001120     03  FILLER      PIC X(10)  VALUE "________  ".                        014800
001130     03  FILLER      PIC X(6)   VALUE "______".                            014900
001140     03  FILLER      PIC X(82)  VALUE SPACES.                              015000
001150 PROCEDURE DIVISION.                                                      015100
001160     OPEN OUTPUT PRINT-OUT.                                             015200
001170     WRITE OUTPUT-RECORD FROM HDG-1.                                    015300
001180     WRITE OUTPUT-RECORD FROM HDG-2.                                    015400
001190     WRITE OUTPUT-RECORD FROM HDG-3.                                    015500
001200     WRITE OUTPUT-RECORD FROM HDG-4.                                    015600
001210     PERFORM PROCESS THRU PROCESS2 VARYING SUB1 FROM 1 BY 1             015700
```

Figure 8 (Part 3 of 5). ABJIVP02 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                   ABJIVP02         05 FEB 1997  10:07:16  PAGE  4
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

001220         UNTIL SUB1 GREATER THAN NUM-OF-ITEMS.                          015800
001230     WRITE OUTPUT-RECORD FROM HDG-4.                                    015900
001240     GO TO EOJ-ROUTINE.                                                 016000
001250 PROCESS.                                                                016100
001260     MOVE SPACES TO ENTRY-DET.                                          016200
001270     MOVE EMPLOYEE(SUB1) TO PREMPL.                                     016300
001280     MOVE HOURS-WORK(SUB1) TO PRHOURS.                                  016400
001290     COMPUTE WORK-GROSS ROUNDED = HOURS-WORK(SUB1) * 4.00.              016500
001300     MOVE WORK-GROSS TO PRSALARY.                                       016600
001310     IF WORK-GROSS GREATER THAN 150.00                                  016700
001320         COMPUTE WORK-TAX ROUNDED = (WORK-GROSS - 150) * .2 + 5         016800
001330         GO TO PROCESS2.                                                016900
001340     IF WORK-GROSS NOT LESS THAN 100.00                                 017000
001350         COMPUTE WORK-TAX = (WORK-GROSS - 100) * .1                     017100
001360         GO TO PROCESS2.                                                017200
001370     MOVE ZEROS TO WORK-TAX.                                            017300
001380 PROCESS2.                                                               017400
001390     MOVE WORK-TAX TO PRTAX                                             017500
001400         COMPUTE WORK-NET = WORK-GROSS - WORK-TAX                       017600
001410         MOVE WORK-NET TO PRNET                                         017700
001420         WRITE ENTRY-DET.                                               017800
001430 EOJ-ROUTINE.                                                            017900
001440     IF ERROR-FLAG = ZERO                                               018000
001450         MOVE "TEST CASE LCPTST09 IS SUCCESSFUL." TO OUTPUT-RECORD      018100
001460         WRITE OUTPUT-RECORD                                            018200
*OLD**     OTHERWISE                                                          018300   ABJ6021 00 OTHERWISE REPLACED BY ELSE
001470     ELSE                                                               018300
001480         MOVE "TEST CASE LCPTST09 FAILED." TO OUTPUT-RECORD             018400
001490         WRITE OUTPUT-RECORD.                                           018500
001500     CLOSE PRINT-OUT.                                                   018600
001510     STOP RUN.                                                                ABJ6126 99 *----------------------------*
001520 END PROGRAM ABJIVP02.                                                   018700             *  END OF COBOL CONVERSION   *
                                                                                                 * 5785-CCC COBOL CONVERSION  *
                                                                                                 *----------------------------*
```

Figure 8 (Part 4 of 5). ABJIVP02 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP02        05 FEB 1997  10:07:16  PAGE  5
CONVERSION FROM DOS/VS COBOL      TO  COBOL/VSE
OPTIONS IN EFFECT :
  PROCEDURE NAME CHECKING ........ YES   LANGLEVEL ..................... DOS/VS COBOL
  FLAG REPORT WRITER STMTS ....... YES   CICS ........................... NO CICS ST.
  REMOVE OBSOLETE ELEMENTS ....... YES   LINE COUNT ..................... 60
  GENERATE CALL ILBOABNO STMT..... YES   DATE FORMAT ................... MM/DD/YY
  GENERATE END PROGRAM STMT ...... YES   RESEQUENCING .................. YES
  POST-CONVERSION COMPILE ........ NO    INCREMENT ..................... 0010
  MANUAL CHANGE FLAGGING ......... YES   RESERVED WORD SUFFIX ........... 74
  HANDLE EXEC SQL INCLUDE AS COPY. YES   GENERATE NEW PROGRAM ........... YES
  REMOVE NON 88 VALUE CLAUSE IN FS YES   GENERATE NEW COPY .............. YES
  FLAG IF FILE-STATUS (NOT) = "00" YES   REPLACE LIKE-NAMED COPY MBR .... YES
  FLAG 31-BIT ADDRESS ARITHMETIC.. YES   PRINT REFERENCE SOURCE LINE .... YES
  INCL.W-S IN CICS COMPILE OF L-S. YES   PRINT COPY MODULE .............. YES
  OPTION-13 ...................... NO    LEVEL DIAGNOSTIC ............... 00
  OPTION-14 ...................... NO    DEBUG MODE ..................... 0
  OPTION-15 ...................... NO    SQL ............................ N
HIGHEST SEVERITY MESSAGE FOR THIS CONVERSION:   00
       0014 MESSAGES ISSUED
       0014 MESSAGES PRINTED
SEQNBR  CPYNBR    MSGID   RC  MESSAGE TEXT

000060           ABJ6011  00  REMARKS CHANGED TO COMMENT
000100           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000110           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000120           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000140           ABJ6022  00  NOTE CHANGED TO COMMENT
000190           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000320           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000320           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000330           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
000350           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
000400           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
000420           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
000860           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
001470           ABJ6021  00  OTHERWISE REPLACED BY ELSE
```

Figure 8 (Part 5 of 5). ABJIVP02 - CCCA/VSE Diagnostics Report

Notes:

1 If you specify Y for the Remove obsolete elements option on the Optional Processing Panel, this clause is removed.

2 **4** **5** COPY statements with associated names are not supported by the target languages. If the name starts with $, the COPY $name is flagged.

3 **6** If the operands of the REPLACING phrase contain an ANSI 85 standard non-COBOL character that is not in a non-numeric literal, the statement is flagged. Under the ANSI 68 and ANSI 74 standard non-COBOL characters in the REPLACING phrase are diagnosed. You should remove all non-COBOL characters from the REPLACING phrase and from the copybook.

6.3.3 ABJIVP03 - COBOL Program with Copy Members and CICS Statements

CCCA/VSE converts CICS Command Level statements from the syntax of the source language level to the target language level.

The Base Locator for Linkage sections (BLLs) are classified as either primary or secondary. Primary BLLs are associated with the portion of the record that is equal to or less than 4096 bytes, and secondary BLLs correspond to record portions greater than 4096 bytes.

6.3.3.1 Input Source Program

```
CBL QUOTE                                                              00001000
      ID DIVISION.                                                     00002000
      PROGRAM-ID. ABJIVP03.                                            00003000
      ENVIRONMENT DIVISION.                                            00004000
      DATA DIVISION.                                                   00005000
      WORKING-STORAGE SECTION.          1                              00006000
      77  PCB     PIC X(4) VALUE "PCB ".                              00007000
      77  GN      PIC X(4) VALUE "GN  ".                              00008000
      77  GU      PIC X(4) VALUE "GU  ".                              00009000
      77  GNP     PIC X(4) VALUE "GNP ".                              00010000
      77  TERM    PIC X(4) VALUE "TERM".                              00011000
      77  SAVE-TCAFCRC PIC X VALUE SPACE.                             00012000
      77  SAVE-TCADLTR PIC X VALUE SPACE.                             00013000
      77  SAVE-STATUS-CODE PIC XX VALUE SPACES.                       00014000
      01  SAVE-TCACCCA PIC X(32) VALUE SPACES.                        00015000
      01  PAGE-OVERFLOW-CTR PIC S9(4) COMP.                           00016000
      01      DFHBMSCA.                                              00017000
        02    DFHBMPEM  PICTURE X   VALUE  IS  "".                  00017020
        02    DFHBMPNL  PICTURE X   VALUE  IS  "".                  00017040
        02    DFHBMASK  PICTURE X   VALUE  IS  "0".                  00017060
        02    DFHBMUNP  PICTURE X   VALUE  IS  " ".                  00017080
        02    DFHBMUNN  PICTURE X   VALUE  IS  "&".                  00017100
        02    DFHBMPRO  PICTURE X   VALUE  IS  "-".                  00017120
        02    DFHBMBRY  PICTURE X   VALUE  IS  "H".                  00017140
        02    DFHBMDAR  PICTURE X   VALUE  IS  "<".                  00017160
        02    DFHBMFSE  PICTURE X   VALUE  IS  "A".                  00017180
        02    DFHBMPRF  PICTURE X   VALUE  IS  "/".                  00017200
        02    DFHBMASF  PICTURE X   VALUE  IS  "1".                  00017220
        02    DFHBMASB  PICTURE X   VALUE  IS  "8".                  00017240
        02    DFHBMEOF  PICTURE X   VALUE  IS  "Ø".                  00017260
        02    DFHBMDET  PICTURE X   VALUE  IS  "".                  00017280
        02    DFHSA     PICTURE X   VALUE  IS  "".                  00017300
        02    DFHCOLOR  PICTURE X   VALUE  IS  "â".                  00017320
        02    DFHPS     PICTURE X   VALUE  IS  "ä".                  00017340
        02    DFHHLT    PICTURE X   VALUE  IS  "".                  00017360
        02    DFH3270   PICTURE X   VALUE  IS  "{".                  00017380
        02    DFHVAL    PICTURE X   VALUE  IS  "A".                  00017400
        02    DFHALL    PICTURE X   VALUE  IS  "".                  00017420
        02    DFHERROR  PICTURE X   VALUE  IS  "".                  00017440
        02    DFHDFT    PICTURE X   VALUE  IS  "".                  00017460
        02    DFHDFCOL  PICTURE X   VALUE  IS  "".                  00017480
        02    DFHBLUE   PICTURE X   VALUE  IS  "1".                  00017500
        02    DFHRED    PICTURE X   VALUE  IS  "2".                  00017520
        02    DFHPINK   PICTURE X   VALUE  IS  "3".                  00017540
        02    DFHGREEN  PICTURE X   VALUE  IS  "4".                  00017560
        02    DFHTURQ   PICTURE X   VALUE  IS  "5".                  00017580
        02    DFHYELLO  PICTURE X   VALUE  IS  "6".                  00017600
        02    DFHNEUTR  PICTURE X   VALUE  IS  "7".                  00017620
        02    DFHBASE   PICTURE X   VALUE  IS  "".                  00017640
        02    DFHDFHI   PICTURE X   VALUE  IS  "".                  00017660
```

Figure 9 (Part 1 of 13). ABJIVP03 - Source Program

```
  02    DFHBLINK  PICTURE X   VALUE  IS  "1".                    00017680
  02    DFHREVRS  PICTURE X   VALUE  IS  "2".                    00017700
  02    DFHUNDLN  PICTURE X   VALUE  IS  "4".                    00017720
  02    DFHMFIL   PICTURE X   VALUE  IS  "".                    00017740
  02    DFHMENT   PICTURE X   VALUE  IS  "".                    00017760
  02    DFHMFE    PICTURE X   VALUE  IS  " ".                    00017780
  02    DFHUNNOD  PICTURE X   VALUE  IS  "(".                    00017800
  02    DFHUNIMD  PICTURE X   VALUE  IS  "I".                    00017820
  02    DFHUNNUM  PICTURE X   VALUE  IS  "J".                    00017840
  02    DFHUNINT  PICTURE X   VALUE  IS  "R".                    00017860
  02    DFHUNNON  PICTURE X   VALUE  IS  ")".                    00017880
  02    DFHPROTI  PICTURE X   VALUE  IS  "Y".                    00017900
  02    DFHPROTN  PICTURE X   VALUE  IS  "%".                    00017920
  02    DFHMT     PICTURE X   VALUE  IS  "".                    00017940
  02    DFHMFT    PICTURE X   VALUE  IS  "    ".                    00017960
  02    DFHMET    PICTURE X   VALUE  IS  "".                    00017980
  02    DFHMFET   PICTURE X   VALUE  IS  "'".                    00018000
                                                                 00018020
01    DFHAID.                                                    00018040
  02  DFHNULL   PIC  X  VALUE IS "".                            00018060
  02  DFHENTER  PIC  X  VALUE IS QUOTE.                         00018080
  02  DFHCLEAR  PIC  X  VALUE IS "_".                           00018100
  02  DFHCLRP   PIC  X  VALUE IS "]".                           00018120
  02  DFHPEN    PIC  X  VALUE IS "=".                           00018140
  02  DFHOPID   PIC  X  VALUE IS "W".                           00018160
  02  DFHMSRE   PIC  X  VALUE IS "X".                           00018180
  02  DFHSTRF   PIC  X  VALUE IS "h".                           00018200
  02  DFHTRIG   PIC  X  VALUE IS """".                          00018220
  02  DFHPA1    PIC  X  VALUE IS "%".                           00018240
  02  DFHPA2    PIC  X  VALUE IS ">".                           00018260
  02  DFHPA3    PIC  X  VALUE IS ",".                           00018280
  02  DFHPF1    PIC  X  VALUE IS "1".                           00018300
  02  DFHPF2    PIC  X  VALUE IS "2".                           00018320
  02  DFHPF3    PIC  X  VALUE IS "3".                           00018340
  02  DFHPF4    PIC  X  VALUE IS "4".                           00018360
  02  DFHPF5    PIC  X  VALUE IS "5".                           00018380
  02  DFHPF6    PIC  X  VALUE IS "6".                           00018400
  02  DFHPF7    PIC  X  VALUE IS "7".                           00018420
  02  DFHPF8    PIC  X  VALUE IS "8".                           00018440
  02  DFHPF9    PIC  X  VALUE IS "9".                           00018460
  02  DFHPF10   PIC  X  VALUE IS ":".                           00018490
  02  DFHPF11   PIC  X  VALUE IS "#".                           00018520
  02  DFHPF12   PIC  X  VALUE IS "@".                           00018550
  02  DFHPF13   PIC  X  VALUE IS "A".                           00018580
  02  DFHPF14   PIC  X  VALUE IS "B".                           00018610
  02  DFHPF15   PIC  X  VALUE IS "C".                           00018640
  02  DFHPF16   PIC  X  VALUE IS "D".                           00018670
  02  DFHPF17   PIC  X  VALUE IS "E".                           00018700
  02  DFHPF18   PIC  X  VALUE IS "F".                           00018730
  02  DFHPF19   PIC  X  VALUE IS "G".                           00018760
  02  DFHPF20   PIC  X  VALUE IS "H".                           00018790
```

Figure 9 (Part 2 of 13). ABJIVP03 - Source Program

```
  02  DFHPF21   PIC  X  VALUE IS "I".                          00018820
  02  DFHPF22   PIC  X  VALUE IS "¢".                          00018850
  02  DFHPF23   PIC  X  VALUE IS ".".                          00018880
  02  DFHPF24   PIC  X  VALUE IS "<".                          00018910
                                                               00018940
                                                               00018970
01  PSBNAME PIC X(8).                                         00019000
01  DLIO PIC X(70).                                           00020000
01  SSA1.                                                     00021000
    02  FILLER PIC X(19) VALUE "ID      (NUM      =".         00022000
    02  SSA1KEY  PIC X(5).                                    00023000
    02  FILLER  PIC X VALUE ")".                              00024000
01  SSA2.                                                     00025000
    02  FILLER PIC X(19) VALUE "CHEQUE  (COMPTE   =".         00026000
    02  SSA2KEY PIC X(5).                                     00027000
    02  FILLER PIC X VALUE ")".                               00028000
01  SSA3.                                                     00029000
    02  FILLER PIC X(19) VALUE "PRET    (PRENUM   =".         00030000
    02  SSA3KEY PIC X(6).                                     00031000
    02  FILLER PIC X VALUE ")".                               00032000
01  MAP1I COPY ABJCQIN.                                       00033000
LINKAGE SECTION.                                             00034000
01  DFHBLLDS SYNCHRONIZED.          2                         00035000
  02  BLLCBAR PICTURE XXXX.                                   00035200
  02  CSACBAR PICTURE XXXX.                                   00035400
  02  CSAOPBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.          00035600
  02  TCACBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.           00035800
    02  PCB-LIST-PTR  PIC S9(8) COMP.                         00036000
    02  PCB1-PTR  PIC S9(8) COMP.                             00037000
    02  CINQOUT-PTR PIC S9(8) COMP.                           00038000
    02  ERRORMP-PTR PIC S9(8) COMP.                           00039000
    02  CIDLOUT-PTR PIC S9(8) COMP.                           00040000
01  DFHCSADS SYNCHRONIZED.                                    00041000
  02  CSAFILLER PICTURE X(512).                               00041005
  02  FILLER1 REDEFINES CSAFILLER.                            00041010
    03  FILLER.                                               00041015
      04  FILLER PICTURE X(76).                               00041020
      04  CSACDTA PICTURE S9(8) USAGE IS COMPUTATIONAL.       00041025
      04  CSATODP PICTURE S9(7) USAGE IS COMPUTATIONAL-3.     00041030
      04  FILLER PICTURE X(12).                               00041035
      04  CSACTODB PICTURE S9(8) USAGE IS COMPUTATIONAL.      00041040
      04  FILLER PICTURE X(24).                               00041045
      04  CSAJYDP PICTURE 9(7) USAGE IS COMPUTATIONAL-3.      00041050
      04  FILLER PICTURE X(64).                               00041055
    03  FILLER.                                               00041060
      04  FILLER PICTURE X(8).                                00041065
      04  CSAOPFLA PICTURE S9(8) USAGE IS COMPUTATIONAL.      00041070
      04  FILLER PICTURE X(20).                               00041075
      04  FILLER.                                             00041080
        05  CSAKCNAC PICTURE XXXX.                            00041085
        05  CSASCNAC PICTURE XXXX.                            00041090
```

Figure 9 (Part 3 of 13). ABJIVP03 - Source Program

```
           05  CSAPCNAC PICTURE XXXX.                              00041095
           05  CSAICNAC PICTURE XXXX.                              00041100
           05  CSADCNAC PICTURE XXXX.                              00041105
           05  CSATCNAC PICTURE XXXX.                              00041110
           05  CSAFCNAC PICTURE XXXX.                              00041115
           05  CSATDNAC PICTURE XXXX.                              00041120
           05  CSATSNAC PICTURE XXXX.                              00041125
           05  CSASANAC PICTURE XXXX.                              00041130
           05  CSATRNAC PICTURE XXXX.                              00041135
           05  CSAPINAC PICTURE XXXX.                              00041140
           05  FILLER PICTURE X(4).                                00041145
           05  CSASPNAC PICTURE XXXX.                              00041150
           05  CSATCRWE PICTURE XXXX.                              00041155
         03  FILLER PICTURE X(215).                                00041160
         03  CSAUTA1 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.       00041165
         03  CSAUTA2 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.       00041170
         03  CSAUTA3 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.       00041175
         03  CSAUTA4 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.       00041180
         03  FILLER  PICTURE X(1).                                 00041185
      *    ABOVE FILLER ADDED BY APAR PN26174                      00041190
                                                                   00041195
                                                                   00041200
       01  DFHTCADS PICTURE X(64) SYNCHRONIZED.                   00041205
       01  CSAOPFL REDEFINES DFHTCADS SYNCHRONIZED.               00041210
         02  CSAATP PICTURE XXXX.                                 00041215
         02  CSAATTCH PICTURE XXXX.                               00041220
         02  CSADLI PICTURE XXXX.                                 00041225
         02  CSABFNAC PICTURE XXXX.                               00041230
         02  CSABMS PICTURE XXXX.                                 00041235
         02  CSATMSVT PICTURE XXXX.                               00041240
         02  CSAJCNA1 PICTURE XXXX.                               00041245
         02  CSAJCNA2 PICTURE XXXX.                               00041250
         02  CSASRNAC PICTURE XXXX.                               00041255
         02  CSASRTBA PICTURE XXXX.                               00041260
         02  CSAKPNAC PICTURE XXXX.                               00041265
         02  CSAATMSP PICTURE XXXX.                               00041270
         02  CSAXLTBA PICTURE XXXX.                               00041275
         02  CSAJCTBA PICTURE XXXX.                               00041280
       01  DFHTCA SYNCHRONIZED.                                   00041285
         02  FILLER.                                              00041290
           03  FILLER PICTURE X(8).                               00041295
           03  TCAFCAAA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00041300
           03  FILLER REDEFINES TCAFCAAA.                         00041305
             04  TCAFCAA1 PICTURE X.                              00041310
             04  FILLER PICTURE X(3).                             00041315
           03 FILLER.                                             00041320
             04  FILLER PICTURE X(8).                             00041325
             04  TCATCEA PICTURE S9(8) USAGE IS COMPUTATIONAL.    00041330
             04  FILLER REDEFINES TCATCEA.                        00041335
               05  TCATCQA PICTURE S9(8) USAGE IS COMPUTATIONAL.  00041340
             04  TCATCTR1 PICTURE  9(4) USAGE IS COMPUTATIONAL.   00041345
```

Figure 9 (Part 4 of 13). ABJIVP03 - Source Program

```
    04  FILLER REDEFINES TCATCTR1.                              00041350
      05  TCATCEI PICTURE X.                                    00041355
      05  TCATCTR PICTURE X.                                    00041360
    04  FILLER REDEFINES TCATCTR1.                              00041365
      05  TCATCDC PICTURE X.                                    00041370
      05  FILLER PICTURE X.                                     00041375
    04  TCATCDP PICTURE X.                                      00041380
    04  FILLER PICTURE X(5).                                    00041385
    04  TCATCRS PICTURE X(60).                                  00041390
    04  FILLER REDEFINES TCATCRS.                               00041395
      05  TCATCDP1 PICTURE  9(4) USAGE IS COMPUTATIONAL.        00041400
      05  FILLER PICTURE X(58).                                 00041405
  03  FILLER.                                                   00041410
    04  TCASCCA.                                                00041415
      05  TCASCSA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041420
    04  FILLER REDEFINES TCASCCA.                               00041425
      05  TCAFCTL PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041430
    04  FILLER REDEFINES TCASCCA.                               00041435
      05  TCASCTR PICTURE X.                                    00041440
      05  TCASCIB PICTURE X.                                    00041445
      05  TCASCNB PICTURE  9(4) USAGE IS COMPUTATIONAL.         00041450
    04  FILLER REDEFINES TCASCCA.                               00041455
      05  TCASCRI PICTURE  9(4) USAGE IS COMPUTATIONAL.         00041460
      05  FILLER PICTURE X(2).                                  00041465
    04  TCAFCTL1 PICTURE S9(8) USAGE IS COMPUTATIONAL.          00041470
    04  FILLER PICTURE X(28).                                   00041475
  03  FILLER.                                                   00041480
    04  TCACCCA.                                                00041485
      05  TCACCCA1 PICTURE X(32).                               00041490
      05  TCACCRS1 PICTURE X(56).                               00041495
      05  TCACCSV1 PICTURE S9(4) USAGE IS COMPUTATIONAL.        00041500
      05  TCACCRSV PICTURE XX.                                  00041505
      05  TCACCSV2 PICTURE XXXX.                                00041510
    04  FILLER REDEFINES TCACCCA.                               00041515
      05  TCATPAPR PICTURE X.                                   00041520
        88  TCATPVAL   VALUE  "6".                              00041525
        88  TCATPNVL   VALUE  "7".                              00041530
        88  TCATPLNR   VALUE  "".                              00041535
      05  FILLER PICTURE X.                                     00041540
      05  TCATPOS PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041545
      05  TCATPCS PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041550
      05  TCATPOC PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041555
      05  TCATPLDM PICTURE XX.                                  00041560
      05  TCATPCON PIC S9(4) USAGE IS COMPUTATIONAL.            00041565
      05  TCATPPNM PICTURE X(8).                                00041570
      05  FILLER PICTURE X(76).                                 00041575
    04  FILLER REDEFINES TCACCCA.                               00041580
      05  FILLER PICTURE X(24).                                 00041585
      05  TCAKCTI PICTURE X(4).                                 00041590
      05  TCAKCFA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041595
      05  FILLER PICTURE X(64).                                 00041600
```

Figure 9 (Part 5 of 13). ABJIVP03 - Source Program

```
04  FILLER REDEFINES TCACCCA.                                    00041605
  05  TCAICDA PICTURE S9(8) USAGE IS COMPUTATIONAL.              00041610
  05  FILLER REDEFINES TCAICDA.                                 00041615
    06  TCAICTR PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041620
    06  FILLER PICTURE X(2).                                    00041625
  05  FILLER REDEFINES TCAICDA.                                 00041630
    06  TCAICRC PICTURE X.                                      00041635
    06  FILLER PICTURE X(3).                                    00041640
  05  TCAICQID.                                                 00041645
      07  TCAICQPX PICTURE XX.                                  00041650
      07  FILLER   PICTURE X(6).                                00041655
    05  TCAICRT PICTURE S9(7) USAGE IS COMPUTATIONAL-3.         00041660
  05  TCAICTI PICTURE X(4).                                     00041665
  05  TCAICTID PICTURE X(4).                                    00041670
  05  FILLER PICTURE X(4).                                      00041675
  05  TCAFCTR1 PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041680
  05  FILLER PICTURE X(66).                                     00041685
04  FILLER REDEFINES TCACCCA.                                  00041690
  05  TCAPCLA PICTURE S9(8) USAGE IS COMPUTATIONAL.            00041695
  05  FILLER REDEFINES TCAPCLA.                                 00041700
    06  TCAPCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041705
    06  FILLER PICTURE X(2).                                    00041710
  05  FILLER REDEFINES TCAPCLA.                                 00041715
    06  TCAPCRC PICTURE X.                                      00041720
      88  PCPGMIDER   VALUE "".                                 00041725
      88  PCNORESP    VALUE "".                                 00041730
      88  ICNORESP    VALUE "".                                 00041735
      88  ICENDDATA   VALUE "".                                 00041740
      88  ICIOERROR   VALUE "".                                 00041745
      88  ICTRNIDER   VALUE "│".                                00041750
      88  ICTRMIDER   VALUE "└".                                00041755
      88  ICTSINVLD   VALUE "".                                 00041760
      88  ICEXPIRD    VALUE "".                                 00041765
      88  ICNOTFND    VALUE "£".                                00041770
      88  ICINVREQ    VALUE "".                                 00041775
      88  TSNORESP    VALUE "".                                 00041780
      88  TSENERROR   VALUE "".                                 00041785
      88  TSIDERROR   VALUE "".                                 00041790
      88  TSIOERROR   VALUE "".                                 00041795
      88  TSINVREQ    VALUE "".                                 00041800
      88  TDNORESP    VALUE "".                                 00041805
      88  TDQUEZERO   VALUE "".                                 00041810
      88  TDIDERROR   VALUE "".                                 00041815
      88  TDIOERROR   VALUE "".                                 00041820
      88  TDNOTOPEN   VALUE ";".                                00041825
      88  TDNOSPACE   VALUE "┌".                                00041830
      88  FCNORESP    VALUE "".                                 00041835
      88  FCDSIDER    VALUE "".                                 00041840
      88  FCSEGIDER   VALUE "".                                 00041845
      88  FCINVREQ    VALUE ";".                                00041850
      88  FCDUPDS     VALUE "".                                 00041855
```

Figure 9 (Part 6 of 13). ABJIVP03 - Source Program

```
           88  FCNOTOPEN   VALUE "".                              00041860
           88  FCENDFILE   VALUE "".                              00041865
           88  FCIOERROR   VALUE "0".                              00041870
           88  FCNOTFND    VALUE "a".                              00041875
           88  FCDUPREC    VALUE "b".                              00041880
           88  FCNOSPACE   VALUE "c".                              00041885
           88  FCDUPKEY    VALUE "d".                              00041890
           88  FCILLOGIC   VALUE "".                              00041896
         06  TCAPCFLA PICTURE X.                                   00041902
         06  TCAPCARO PICTURE X.                                   00041908
         06  FILLER PICTURE X.                                     00041914
       05  TCAPCPI PICTURE X(8).                                   00041920
       05  FILLER REDEFINES TCAPCPI.                               00041926
         06  TCAPCERA PICTURE S9(8) USAGE IS COMPUTATIONAL.        00041932
         06  FILLER PICTURE X(4).                                  00041938
       05  TCAPCAC PICTURE XXXX.                                   00041944
       05  TCAPCPSW PICTURE X(8).                                  00041950
       05  TCAPCINT PICTURE X(8).                                  00041956
       05  FILLER PICTURE X(64).                                   00041962
     04  FILLER REDEFINES TCACCCA.                                 00041968
       05  TCADCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041974
       05  TCADCNB PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041980
       05  TCADCSA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00041986
       05  FILLER PICTURE XXXX.                                    00041992
       05  TCADCDC PICTURE XXXX.                                   00041998
       05  FILLER PICTURE X(80).                                   00042004
     04  FILLER REDEFINES TCACCCA.                                 00042010
       05  TCAFCAA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042016
       05  FILLER REDEFINES TCAFCAA.                               00042022
         06  TCAFCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042028
         06  FILLER PICTURE X(2).                                  00042034
       05  FILLER REDEFINES TCAFCAA.                               00042040
         06  TCAFCRC PICTURE X.                                    00042046
         06  FILLER PICTURE X(3).                                  00042052
       05  TCAFCDI PICTURE X(8).                                   00042058
       05  TCAFCURL PICTURE  9(4) USAGE IS COMPUTATIONAL.          00042064
       05  FILLER REDEFINES TCAFCURL.                              00042070
         06  TCAFCNRD PICTURE  9(4) USAGE IS COMPUTATIONAL.       00042076
       05  FILLER  PICTURE X(6).                                   00042082
       05  FILLER  PICTURE X(8).                                   00042088
       05  TCAFCRI PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042094
       05  FILLER PICTURE X(64).                                   00042100
     04  FILLER REDEFINES TCACCCA.                                 00042106
       05  TCATDAA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042112
       05  FILLER REDEFINES TCATDAA.                               00042118
         06  TCATDRC PICTURE X.                                    00042124
         06  FILLER PICTURE X(3).                                  00042130
       05  FILLER REDEFINES TCATDAA.                               00042136
         06  TCATDTR PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042142
         06  FILLER PICTURE X(2).                                  00042148
       05  TCATDDI PICTURE XXXX.                                   00042154
```

Figure 9 (Part 7 of 13). ABJIVP03 - Source Program

```
  05  FILLER PICTURE X(88).                                    00042160
04  FILLER REDEFINES TCACCCA.                                 00042166
  05  TCATSDA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042172
  05  FILLER REDEFINES TCATSDA.                               00042178
    06  TCATSRC PICTURE X.                                    00042184
    06  FILLER PICTURE X(3).                                  00042190
  05  FILLER REDEFINES TCATSDA.                               00042196
    06  TCATSTR PICTURE  9(4) USAGE IS COMPUTATIONAL.         00042202
    06  FILLER PICTURE X(2).                                  00042208
  05  TCATSDI PICTURE X(8).                                   00042214
  05  TCATSRN PICTURE  9(4) USAGE IS COMPUTATIONAL.           00042220
  05  FILLER PICTURE  X(2).                                   00042226
  05  TCATSTR2 PICTURE  9(4) USAGE IS COMPUTATIONAL.          00042232
  05  FILLER PICTURE X(78).                                   00042238
04  FILLER REDEFINES TCACCCA.                                 00042244
  05  TCAMSTR1 PICTURE X(8).                                  00042250
  05  FILLER REDEFINES TCAMSTR1.                              00042256
    06  FILLER PICTURE X.                                     00042262
    06  TCAMSTR2 PICTURE X.                                   00042268
    06  TCAMSTR3 PICTURE X.                                   00042274
    06  TCAMSTR4 PICTURE X.                                   00042280
    06  TCAMSTR5 PICTURE X.                                   00042286
    06  TCAMSTR6 PICTURE X.                                   00042292
    06  TCAMSTR7 PICTURE X.                                   00042298
    06  TCAMSTR8 PICTURE X.                                   00042304
  05  FILLER REDEFINES TCAMSTR1.                              00042310
    06  TCAMSRC1 PICTURE X.                                   00042316
    06  TCAMSRC2 PICTURE X.                                   00042322
    06  TCAMSRC3 PICTURE X.                                   00042328
    06  TCAMSRI1 PICTURE X.                                   00042334
    06  TCAMSPGN PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042340
    06  TCAMSOCN PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042346
  05  FILLER REDEFINES TCAMSTR1.                              00042352
    06  FILLER PICTURE XX.                                    00042358
    06  TCAMSRC3H PICTURE 9(4) USAGE IS COMPUTATIONAL.        00042364
    06  FILLER PICTURE X(4).                                  00042370
  05  FILLER REDEFINES TCAMSTR1.                              00042376
    06  TCAMSRC PICTURE XXX.                                  00042382
    06  FILLER PICTURE X(5).                                  00042388
  05  TCAMSIOA PICTURE S9(8) USAGE IS COMPUTATIONAL.          00042394
  05  FILLER REDEFINES TCAMSIOA.                              00042400
    06  TCAMSTA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00042406
  05  TCAMSFSC PICTURE XXXX.                                  00042412
  05  FILLER REDEFINES TCAMSFSC.                              00042418
    06  TCABMSFB PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042424
    06  FILLER REDEFINES TCABMSFB.                            00042430
      07  TCABMSWC PICTURE X.                                 00042436
      07  FILLER PICTURE X.                                   00042442
    06  FILLER REDEFINES TCABMSFB.                            00042448
      07  TCAMSWCC PICTURE X.                                 00042454
      07  TCAMSJ PICTURE X.                                   00042460
```

Figure 9 (Part 8 of 13). ABJIVP03 - Source Program

```
    06  TCABMSCP PICTURE S9(4) USAGE IS COMPUTATIONAL.     00042466
  05  TCABMSMN PICTURE X(8).                              00042472
  05  FILLER REDEFINES TCABMSMN.                          00042478
    06  TCABMSMA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042484
    06  FILLER PICTURE X(4).                              00042490
  05  FILLER REDEFINES TCABMSMN.                          00042496
    06  TCAMSHDR PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042502
    06  FILLER PICTURE X(4).                              00042508
  05  FILLER REDEFINES TCABMSMN.                          00042514
    06  TCAMSRLA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042520
    06  TCAMSRTI PICTURE S9(7) USAGE IS COMPUTATIONAL-3.   00042526
    06  FILLER REDEFINES TCAMSRTI.                        00042532
      07  TCAMSTRL PICTURE S9(8) USAGE IS COMPUTATIONAL.   00042538
  05  TCAMSMSN PICTURE X(8).                              00042544
  05  FILLER REDEFINES TCAMSMSN.                          00042550
    06  TCAMSMSA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042556
    06  FILLER PICTURE X(4).                              00042562
  05  FILLER REDEFINES TCAMSMSN.                          00042568
    06  TCAMSTI PICTURE X(4).                             00042574
    06  FILLER PICTURE X.                                 00042580
    06  TCAMSOC PICTURE XXX.                              00042586
  05  TCAMSLDM PICTURE XX.                                00042592
  05  TCAMSLDC PICTURE X.                                 00042598
  05  TCAMSRID PICTURE XX.                                00042604
  05  FILLER PICTURE XXX.                                 00042610
  05  TCAMSFMP PICTURE X(8).                              00042616
  05  FILLER PICTURE X(48).                               00042622
04  FILLER REDEFINES TCACCCA.                            00042628
  05  TCASPTR PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042634
  05  FILLER PICTURE X(94).                               00042640
04  FILLER REDEFINES TCACCCA.                            00042646
  05  TCADLIO PICTURE S9(8) USAGE IS COMPUTATIONAL.        00042652
  05  FILLER REDEFINES TCADLIO.                           00042658
    06  FILLER PICTURE X.                                 00042664
    06  TCADLTR PICTURE X.                                00042670
      88  FCDLINA   VALUE "".                            00042676
      88  FCPSBSCH  VALUE "".                            00042682
      88  FCPSBNF   VALUE "".                            00042688
      88  FCTASKNA  VALUE "".                            00042694
      88  FCPSBNA   VALUE " ".                            00042700
      88  FCLANGCON VALUE "".                            00042706
      88  FCPSBFAIL VALUE "  ".                             00042712
      88  FCFUNCNS  VALUE ";".                            00042718
      88  FCTERMNS  VALUE "'".                            00042724
    06  FILLER PICTURE X(2).                              00042730
  05  TCADLPCB PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042736
  05  TCADLPSB PICTURE X(8).                              00042742
  05  TCADLSSA PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042748
  05  TCADLPAR PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042754
  05  TCADLECB PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042760
  05  FILLER REDEFINES TCADLECB.                          00042766
```

Figure 9 (Part 9 of 13). ABJIVP03 - Source Program

```
           06  TCADLLAN PICTURE X(4).                              00042772
         05  TCADLFUN PICTURE X(4).                                00042778
         05  FILLER PICTURE X(64).                                 00042784
       04  FILLER.                                                 00042790
         05  TCATRF1 PICTURE S9(8) USAGE IS COMPUTATIONAL.         00042796
         05  FILLER REDEFINES TCATRF1.                             00042802
           06  TCATRF1H PICTURE  9(4) USAGE IS COMPUTATIONAL.      00042808
           06  FILLER PICTURE X(2).                                00042814
         05  FILLER REDEFINES TCATRF1.                             00042820
           06  TCATRF1F PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042826
         05  FILLER REDEFINES TCATRF1.                             00042832
           06  TCATRF1C PICTURE X(4).                              00042838
         05  FILLER REDEFINES TCATRF1.                             00042844
           06  TCATRF1P PICTURE 9(7) USAGE IS COMPUTATIONAL-3.     00042850
         05  FILLER REDEFINES TCATRF1.                             00042856
           06  TCATRF1A PICTURE X.                                 00042862
           06  FILLER PICTURE X(3).                                00042868
         05  TCATRF2 PICTURE S9(8) USAGE IS COMPUTATIONAL.         00042874
         05  FILLER REDEFINES TCATRF2.                             00042880
           06  TCATRF2H PICTURE  9(4) USAGE IS COMPUTATIONAL.      00042886
           06  FILLER PICTURE X(2).                                00042892
         05  FILLER REDEFINES TCATRF2.                             00042898
           06  TCATRF2F PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042904
         05  FILLER REDEFINES TCATRF2.                             00042910
           06  TCATRF2C PICTURE X(4).                              00042916
         05  FILLER REDEFINES TCATRF2.                             00042922
           06  TCATRF2P PICTURE 9(7) USAGE IS COMPUTATIONAL-3.     00042928
         05  FILLER REDEFINES TCATRF2.                             00042934
           06  TCATRF2A PICTURE X.                                 00042940
           06  FILLER PICTURE X(3).                                00042946
         05  TCATRRI PICTURE  9(4) USAGE IS COMPUTATIONAL.         00042952
         05  TCATRRI1 PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042958
         05  FILLER PICTURE X(4).                                  00042964
         05  TCAJCAAD PICTURE S9(8) USAGE IS COMPUTATIONAL.        00042970
         05  TCAATAC PICTURE S9(8) USAGE IS COMPUTATIONAL.         00042976
     03  FILLER.                                                   00042982
       04  TCACSPE PICTURE XXXX.                                   00042988
       04  TCANXTID PICTURE X(4).                                  00042994
01   PCB-ADDR.                                                     00043000
     02  PCB1-ADDR PIC S9(8) COMP.                                00044000
01   PCB1.                                                         00045000
     02  DBD-NAME  PIC X(8).                                      00046000
     02  SEG-LEVEL  PIC XX.                                       00047000
     02  STATUS-CODE  PIC XX.                                     00048000
     02  PROC-OPTIONS  PIC X(4).                                  00049000
     02  RESERVE-DLI  PIC S9(5) COMP.                             00050000
     02  SEG-NAME-FB  PIC X(8).                                   00051000
     02  LENGTH-FB-KEY PIC S9(5) COMP.                            00052000
     02  NUMB-SENS-SEGS PIC S9(5) COMP.                           00053000
     02  KEY-FB-AREA PIC X(30).                                   00054000
01   MAP11I COPY ABJCQOUT.                                        00055000
```

Figure 9 (Part 10 of 13). ABJIVP03 - Source Program

```
01  MAP12I COPY ABJERRMP.                                          00056000
01  MAP13I COPY ABJCIOUT.                                          00057000
PROCEDURE DIVISION.                                                00058000
    MOVE CSACDTA TO TCACBAR.          3                            00059000
    EXEC CICS HANDLE CONDITION ERROR(ERRORS) MAPFAIL(CIDL)         00060000
        OVERFLOW(PAGE-OVERFLOW) END-EXEC.                          00061000
    EXEC CICS RECEIVE MAP("MAP1") MAPSET("CINQIN") END-EXEC.       00062000
    IF CUSTNOI = SPACES OR CUSTNOL = +0000                         00063000
        MOVE "CUSTOMER" TO ERRNAMEO                                00064000
        MOVE SPACES TO ERRNOO GO TO ERR-MSG.                       00065000
    MOVE "PSBCLIG" TO PSBNAME.                                     00066000
    CALL "CBLTDLI" USING PCB PSBNAME.                              00067000
    IF TCAFCRC NOT EQUAL TO "" GO TO INTERFACE-ERROR.             00068000
    MOVE TCADLPCB TO PCB-LIST-PTR.          4                      00069000
    MOVE PCB1-ADDR TO PCB1-PTR.             5                      00070000
    MOVE CUSTNOI TO SSA1KEY.                                       00071000
    MOVE CHECKNOI TO SSA2KEY.                                      00072000
    MOVE LOANNOI TO SSA3KEY.                                       00073000
    IF SSA2KEY NOT = LOW-VALUE GO TO CHECK-PROC.                   00074000
    IF SSA3KEY NOT = LOW-VALUE GO TO LOAN-PROC.                    00075000
    CALL "CBLTDLI" USING GU PCB1 DLIO SSA1.                        00076000
    IF TCAFCRC NOT EQUAL TO "" GO TO INTERFACE-ERROR.             00077000
    IF STATUS-CODE = "  " GO TO GU-OK.                             00078000
    IF STATUS-CODE = "GE" MOVE "CUSTOMER" TO ERRNAMEO              00079000
        MOVE CUSTNOI TO ERRNOO                                     00080000
        GO TO ERR-MSG.                                             00081000
    GO TO ERROR1.                                                  00082000
CHECK-PROC.                                                        00083000
    MOVE CHECKNOI TO SSA2KEY.                                      00084000
    CALL "CBLTDLI" USING GU PCB1 DLIO SSA1 SSA2.                   00085000
    IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                    00086000
    IF STATUS-CODE = "  " GO TO GU-OK.                             00087000
    IF STATUS-CODE = "GE" MOVE "CHECK" TO ERRNAMEO                 00088000
        MOVE CHECKNOI TO ERRNOO                                    00089000
        GO TO ERR-MSG.                                             00090000
LOAN-PROC.                                                         00091000
    MOVE LOANNOI TO SSA3KEY.                                       00092000
    CALL "CBLTDLI" USING GU PCB1 DLIO SSA1 SSA3.                   00093000
    IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                    00094000
    IF STATUS-CODE = "  " GO TO GU-OK.                             00095000
    IF STATUS-CODE = "GE" MOVE "LOAN" TO ERRNAMEO                  00096000
        MOVE LOANNOI TO ERRNOO                                     00097000
        GO TO ERR-MSG.                                             00098000
GU-OK.                                                             00099000
    MOVE 1 TO PAGE-OVERFLOW-CTR.                                   00100000
    EXEC CICS GETMAIN SET(CINQOUT-PTR) LENGTH(96) END-EXEC.    6   00101000
    EXEC CICS SEND MAP("MAP11") MAPSET("CINQOUT") ACCUM            00102000
        ERASE PAGING FRSET FREEKB END-EXEC.                        00103000
PAGE-BUILD.                                                        00104000
    MOVE SEG-NAME-FB TO SEGNAMEO.                                  00105000
    MOVE DLIO TO SEGCONTO.                                         00106000
```

Figure 9 (Part 11 of 13). ABJIVP03 - Source Program

```
SEND-MAP2.                                                          00107000
    MOVE 2 TO PAGE-OVERFLOW-CTR.                                    00108000
    EXEC CICS SEND MAP("MAP21") MAPSET("CINQOUT") ACCUM             00109000
        PAGING FRSET FREEKB END-EXEC.                               00110000
GNP-LOOP.                                                           00111000
    CALL "CBLTDLI" USING GNP PCB1 DLIO.                             00112000
    IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                     00113000
    IF STATUS-CODE = "  " OR STATUS-CODE = "GA"                     00114000
        OR STATUS-CODE = "GK" GO TO PAGE-BUILD.                     00115000
    IF STATUS-CODE = "GE" OR STATUS-CODE = "GB"                     00116000
        GO TO END-GNP-LOOP.                                         00117000
    GO TO ERROR1.                                                   00118000
END-GNP-LOOP.                                                       00119000
    EXEC CICS SEND MAP("MAP31") MAPSET("CINQOUT") ACCUM             00120000
        PAGING FRSET FREEKB END-EXEC.                               00121000
    EXEC CICS SEND MAP("MAP41") MAPSET("CINQOUT") ACCUM             00122000
        PAGING FRSET FREEKB END-EXEC.                               00123000
PAGE-OUT.                                                           00124000
    EXEC CICS SEND PAGE NOAUTOPAGE END-EXEC.                        00125000
END-PROG.                                                           00126000
PROG-RETURN.                                                        00127000
    CALL "CBLTDLI" USING TERM.                                      00128000
    EXEC CICS RETURN TRANSID("CINQ") END-EXEC.                      00129000
ERRORS.                                                             00130000
    PERFORM SAVE-INFO.                                              00131000
    EXEC CICS DUMP DUMPCODE("ERRS") END-EXEC.                       00132000
    GO TO PROG-RETURN.                                              00133000
CIDL.                                                               00134000
    EXEC CICS GETMAIN SET(CIDLOUT-PTR) LENGTH(12) END-EXEC.  7      00135000
    MOVE LOW-VALUE TO MAP13O.                                       00136000
    EXEC CICS SEND MAP("MAP13") MAPSET("CIDLOUT") ERASE END-EXEC.00137000
    EXEC CICS RETURN END-EXEC.                                      00138000
PAGE-OVERFLOW.                                                      00139000
    EXEC CICS SEND MAP("MAP41") MAPSET("CINQOUT") ACCUM             00140000
        PAGING FREEKB END-EXEC.                                     00141000
    EXEC CICS SEND MAP("MAP11") MAPSET("CINQOUT") ACCUM             00142000
        ERASE PAGING FRSET FREEKB END-EXEC.                         00143000
    GO TO GU-OK SEND-MAP2 DEPENDING ON PAGE-OVERFLOW-CTR.           00144000
ERR-MSG.                                                            00145000
    EXEC CICS SEND MAP("MAP1") MAPSET("CINQIN") ACCUM               00146000
        PAGING FREEKB END-EXEC.                                     00147000
    EXEC CICS SEND MAP("MAP2") MAPSET("CINQIN") ACCUM               00148000
        PAGING FREEKB END-EXEC.                                     00149000
    EXEC CICS SEND PAGE END-EXEC.                                   00150000
    GO TO END-PROG.                                                 00151000
INTERFACE-ERROR.                                                    00152000
    MOVE TCAFCRC TO SAVE-TCAFCRC.                                   00153000
    MOVE TCADLTR TO SAVE-TCADLTR.                                   00154000
    PERFORM SAVE-INFO.                                              00155000
    EXEC CICS DUMP DUMPCODE("INTE") END-EXEC.                       00156000
    EXEC CICS GETMAIN SET(ERRORMP-PTR) LENGTH(85) END-EXEC.  8      00157000
```

Figure 9 (Part 12 of 13). ABJIVP03 - Source Program

```
    MOVE "*** INTERFACE ERROR. DUMP IN PROGRESS.***" TO ERRMSGO. 00158000
    EXEC CICS SEND MAP("MAP12") MAPSET("ERRORMP") ACCUM          00159000
        PAGING FREEKB END-EXEC.                                  00160000
    GO TO CIDL.                                                  00161000
ERROR1.                                                          00162000
    PERFORM SAVE-INFO.                                           00163000
    EXEC CICS DUMP DUMPCODE("ERRO") END-EXEC.                    00164000
    EXEC CICS GETMAIN SET(ERRORMP-PTR) LENGTH(85) END-EXEC.  9   00165000
    MOVE "*** DL/1 CALL ERROR. DUMP IN PROGRESS.***" TO ERRMSGO. 00166000
    EXEC CICS SEND MAP("MAP12") MAPSET("ERRORMP") ACCUM          00167000
        PAGING FREEKB END-EXEC.                                  00168000
    GO TO CIDL.                                                  00169000
SAVE-INFO.                                                       00170000
    MOVE STATUS-CODE TO SAVE-STATUS-CODE.                        00171000
    MOVE TCACCCA TO SAVE-TCACCCA.                                00172000
END-PGM.                                                         00173000
    STOP RUN.                                                    00174000
```

Figure 9 (Part 13 of 13). ABJIVP03 - Source Program

6.3.3.2 Output Program from CCCA/VSE Conversion

```
 CBL QUOTE                                                              00001000
000010 ID DIVISION.                                                     00002000
000020 PROGRAM-ID. ABJIVP03.                                            00003000
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:09:39.
000060 ENVIRONMENT DIVISION.                                            00004000
000070 DATA DIVISION.                                                   00005000
000080 WORKING-STORAGE SECTION.                                         00006000
000090 77  LCP-WS-ADDR-COMP             PIC S9(8) COMP.          1
000100 77  LCP-WS-ADDR-PNTR             REDEFINES LCP-WS-ADDR-COMP
000110                                   USAGE POINTER.
000120 77  PCB     PIC X(4) VALUE "PCB ".                                00007000
000130 77  GN      PIC X(4) VALUE "GN  ".                                00008000
000140 77  GU      PIC X(4) VALUE "GU  ".                                00009000
000150 77  GNP     PIC X(4) VALUE "GNP ".                                00010000
000160 77  TERM    PIC X(4) VALUE "TERM".                                00011000
000170 77  SAVE-TCAFCRC PIC X VALUE SPACE.                               00012000
000180 77  SAVE-TCADLTR PIC X VALUE SPACE.                               00013000
000190 77  SAVE-STATUS-CODE PIC XX VALUE SPACES.                         00014000
000200 01  SAVE-TCACCCA PIC X(32) VALUE SPACES.                          00015000
000210 **** No Change - same as source ********************
               .
               .
               .
001230 01  SSA3.                                                         00029000
001240     02  FILLER PIC X(19) VALUE "PRET    (PRENUM   =".           00030000
001250     02  SSA3KEY PIC X(6).                                         00031000
001260     02  FILLER PIC X VALUE ")".                                   00032000
001270 01  MAP1I. COPY ABJCQIN REPLACING ==01  MAP1I.== BY ====.         00033000
001280 LINKAGE SECTION.                                                 00034000
001290*01  DFHBLLDS SYNCHRONIZED.          2                             00035000
001300*  02  BLLCBAR PICTURE XXXX.                                       00035200
001310*  02  CSACBAR PICTURE XXXX.                                       00035400
001320*  02  CSAOPBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.              00035600
001330*  02  TCACBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.               00035800
001340*    02  PCB-LIST-PTR  PIC S9(8) COMP.                             00036000
001350*    02  PCB1-PTR  PIC S9(8) COMP.                                 00037000
001360*    02  CINQOUT-PTR PIC S9(8) COMP.                               00038000
001370*    02  ERRORMP-PTR PIC S9(8) COMP.                               00039000
001380*    02  CIDLOUT-PTR PIC S9(8) COMP.                               00040000
001390 01  DFHCSADS SYNCHRONIZED.                                        00041000
001400 **** No Change - same as source ********************
               .
               .
               .
005140 01  MAP11I. COPY ABJCQOUT REPLACING ==01  MAP11I.== BY ====.      00055000
005150 01  MAP12I. COPY ABJERRMP REPLACING ==01  MAP12I.== BY ====.      00056000
005160 01  MAP13I. COPY ABJCIOUT REPLACING ==01  MAP13I.== BY ====.      00057000
```

Figure 10 (Part 1 of 3). ABJIVP03 - Converted Program

```
005170 PROCEDURE DIVISION.                                              00058000
005180     MOVE CSACDTA TO LCP-WS-ADDR-COMP          3                 00059000
005190     SET ADDRESS OF DFHTCA TO LCP-WS-ADDR-PNTR.
005200     EXEC CICS HANDLE CONDITION ERROR(ERRORS) MAPFAIL(CIDL)       00060000
005210         OVERFLOW(PAGE-OVERFLOW) END-EXEC.                        00061000
005220     EXEC CICS RECEIVE MAP("MAP1") MAPSET("CINQIN") END-EXEC.     00062000
005230     IF CUSTNOI = SPACES OR CUSTNOL = +0000                       00063000
005240         MOVE "CUSTOMER" TO ERRNAMEO                              00064000
005250         MOVE SPACES TO ERRNOO GO TO ERR-MSG.                     00065000
005260     MOVE "PSBCLIG" TO PSBNAME.                                   00066000
005270     CALL "CBLTDLI" USING PCB PSBNAME.                            00067000
005280     IF TCAFCRC NOT EQUAL TO "" GO TO INTERFACE-ERROR.           00068000
005290     MOVE TCADLPCB TO LCP-WS-ADDR-COMP         4                  00069000
005300     SET ADDRESS OF PCB-ADDR TO LCP-WS-ADDR-PNTR.
005310     MOVE PCB1-ADDR TO LCP-WS-ADDR-COMP        5                  00070000
005320     SET ADDRESS OF PCB1 TO LCP-WS-ADDR-PNTR.
005330     MOVE CUSTNOI TO SSA1KEY.                                     00071000
005340     MOVE CHECKNOI TO SSA2KEY.                                    00072000
005350     MOVE LOANNOI TO SSA3KEY.                                     00073000
005360 **** No Change - same as source ********************
               .
               .
               .
005610 GU-OK.                                                           00099000
005620     MOVE 1 TO PAGE-OVERFLOW-CTR.                                 00100000
005630     EXEC CICS GETMAIN SET(ADDRESS OF MAP11I) LENGTH(96) END-EXEC.00101000  6
005640     EXEC CICS SEND MAP("MAP11") MAPSET("CINQOUT") ACCUM          00102000
005650         ERASE PAGING FRSET FREEKB END-EXEC.                      00103000
005660 **** No Change - same as source ********************
               .
               .
               .
005960 CIDL.                                                            00134000
005970     EXEC CICS GETMAIN SET(ADDRESS OF MAP13I) LENGTH(12) END-EXEC.00135000  7
005980     MOVE LOW-VALUE TO MAP13O.                                    00136000
005990     EXEC CICS SEND MAP("MAP13") MAPSET("CIDLOUT") ERASE END-EXEC.00137000
006000     EXEC CICS RETURN END-EXEC.                                   00138000
006010 **** No Change - same as source ********************
               .
               .
               .
006140 INTERFACE-ERROR.                                                 00152000
006150     MOVE TCAFCRC TO SAVE-TCAFCRC.                                00153000
006160     MOVE TCADLTR TO SAVE-TCADLTR.                                00154000
006170     PERFORM SAVE-INFO.                                           00155000
006180     EXEC CICS DUMP DUMPCODE("INTE") END-EXEC.                    00156000
006190     EXEC CICS GETMAIN SET(ADDRESS OF MAP12I) LENGTH(85) END-EXEC.00157000  8
006200     MOVE "*** INTERFACE ERROR. DUMP IN PROGRESS.***" TO ERRMSGO. 00158000
006210     EXEC CICS SEND MAP("MAP12") MAPSET("ERRORMP") ACCUM          00159000
006220         PAGING FREEKB END-EXEC.                                  00160000
006230     GO TO CIDL.                                                  00161000
```

Figure 10 (Part 2 of 3). ABJIVP03 - Converted Program

```
006240 ERROR1.                                                          00162000
006250     PERFORM SAVE-INFO.                                           00163000
006260     EXEC CICS DUMP DUMPCODE(”ERRO”) END-EXEC.                    00164000
006270     EXEC CICS GETMAIN SET(ADDRESS OF MAP12I) LENGTH(85) END-EXEC.00165000 9
006280     MOVE ”*** DL/1 CALL ERROR. DUMP IN PROGRESS.***” TO ERRMSGO. 00166000
006290     EXEC CICS SEND MAP(”MAP12”) MAPSET(”ERRORMP”) ACCUM          00167000
006300         PAGING FREEKB END-EXEC.                                  00168000
006310     GO TO CIDL.                                                  00169000
006320 SAVE-INFO.                                                       00170000
006330     MOVE STATUS-CODE TO SAVE-STATUS-CODE.                        00171000
006340     MOVE TCACCCA TO SAVE-TCACCCA.                                00172000
006350 END-PGM.                                                         00173000
006360     STOP RUN.                                                    00174000
006370 END PROGRAM ABJIVP03.
```

Figure 10 (Part 3 of 3). ABJIVP03 - Converted Program

6.3.3.3 CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  1
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

 CBL QUOTE                                                                  00001000  CBL Q
000010 ID DIVISION.                                                         00002000
000020 PROGRAM-ID. ABJIVP03.                                                00003000
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 02/05/97 10:09:39.
000060 ENVIRONMENT DIVISION.                                                00004000
000070 DATA DIVISION.                                                       00005000
000080 WORKING-STORAGE SECTION.                                             00006000          ABJ6212 OO WORKING POINTER FOR CICS
000090 77  LCP-WS-ADDR-COMP              PIC S9(8) COMP.        1                                ADDED TO WORKING STORAGE
000100 77  LCP-WS-ADDR-PNTR              REDEFINES LCP-WS-ADDR-COMP
000110                                    USAGE POINTER.
000120 77  PCB     PIC X(4) VALUE "PCB ".                                    00007000
000130 77  GN      PIC X(4) VALUE "GN  ".                                    00008000
000140 77  GU      PIC X(4) VALUE "GU  ".                                    00009000
000150 77  GNP     PIC X(4) VALUE "GNP ".                                    00010000
000160 77  TERM    PIC X(4) VALUE "TERM".                                    00011000
000170 77  SAVE-TCAFCRC PIC X VALUE SPACE.                                   00012000
000180 77  SAVE-TCADLTR PIC X VALUE SPACE.                                   00013000
000190 77  SAVE-STATUS-CODE PIC XX VALUE SPACES.                             00014000
000200 01  SAVE-TCACCCA PIC X(32) VALUE SPACES.                              00015000
000210 01  PAGE-OVERFLOW-CTR PIC S9(4) COMP.                                 00016000
000220 01      DFHBMSCA.                                                     00017000
000230   02    DFHBMPEM  PICTURE X   VALUE  IS  "".                        00017020
000240   02    DFHBMPNL  PICTURE X   VALUE  IS  "".                        00017040
000250   02    DFHBMASK  PICTURE X   VALUE  IS  "0".                       00017060
000260   02    DFHBMUNP  PICTURE X   VALUE  IS  " ".                       00017080
000270   02    DFHBMUNN  PICTURE X   VALUE  IS  "&".                       00017100
000280   02    DFHBMPRO  PICTURE X   VALUE  IS  "-".                       00017120
000290   02    DFHBMBRY  PICTURE X   VALUE  IS  "H".                       00017140
000300   02    DFHBMDAR  PICTURE X   VALUE  IS  "<".                       00017160
000310   02    DFHBMFSE  PICTURE X   VALUE  IS  "A".                       00017180
000320   02    DFHBMPRF  PICTURE X   VALUE  IS  "/".                       00017200
000330   02    DFHBMASF  PICTURE X   VALUE  IS  "1".                       00017220
000340   02    DFHBMASB  PICTURE X   VALUE  IS  "8".                       00017240
000350   02    DFHBMEOF  PICTURE X   VALUE  IS  "Ø".                       00017260
000360   02    DFHBMDET  PICTURE X   VALUE  IS  "".                        00017280
000370   02    DFHSA     PICTURE X   VALUE  IS  "".                        00017300
000380   02    DFHCOLOR  PICTURE X   VALUE  IS  "â".                       00017320
000390   02    DFHPS     PICTURE X   VALUE  IS  "ä".                       00017340
000400   02    DFHHLT    PICTURE X   VALUE  IS  "".                        00017360
000410   02    DFH3270   PICTURE X   VALUE  IS  "{".                       00017380
000420   02    DFHVAL    PICTURE X   VALUE  IS  "A".                       00017400
000430   02    DFHALL    PICTURE X   VALUE  IS  "".                        00017420
000440   02    DFHERROR  PICTURE X   VALUE  IS  "".                        00017440
000450   02    DFHDFT    PICTURE X   VALUE  IS  "".                        00017460
000460   02    DFHDFCOL  PICTURE X   VALUE  IS  "".                        00017480
000470   02    DFHBLUE   PICTURE X   VALUE  IS  "1".                       00017500
000480   02    DFHRED    PICTURE X   VALUE  IS  "2".                       00017520
000490   02    DFHPINK   PICTURE X   VALUE  IS  "3".                       00017540
000500   02    DFHGREEN  PICTURE X   VALUE  IS  "4".                       00017560
000510   02    DFHTURQ   PICTURE X   VALUE  IS  "5".                       00017580
000520   02    DFHYELLO  PICTURE X   VALUE  IS  "6".                       00017600
```

Figure 11 (Part 1 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                   ABJIVP03         05 FEB 1997  10:09:49  PAGE  2
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000530   02    DFHNEUTR  PICTURE X   VALUE  IS  "7".                    00017620
000540   02    DFHBASE   PICTURE X   VALUE  IS  "".                     00017640
000550   02    DFHDFHI   PICTURE X   VALUE  IS  "".                     00017660
000560   02    DFHBLINK  PICTURE X   VALUE  IS  "1".                    00017680
000570   02    DFHREVRS  PICTURE X   VALUE  IS  "2".                    00017700
000580   02    DFHUNDLN  PICTURE X   VALUE  IS  "4".                    00017720
000590   02    DFHMFIL   PICTURE X   VALUE  IS  "".                     00017740
000600   02    DFHMENT   PICTURE X   VALUE  IS  "".                     00017760
000610   02    DFHMFE    PICTURE X   VALUE  IS  " ".                    00017780
000620   02    DFHUNNOD  PICTURE X   VALUE  IS  "(".                    00017800
000630   02    DFHUNIMD  PICTURE X   VALUE  IS  "I".                    00017820
000640   02    DFHUNNUM  PICTURE X   VALUE  IS  "J".                    00017840
000650   02    DFHUNINT  PICTURE X   VALUE  IS  "R".                    00017860
000660   02    DFHUNNON  PICTURE X   VALUE  IS  ")".                    00017880
000670   02    DFHPROTI  PICTURE X   VALUE  IS  "Y".                    00017900
000680   02    DFHPROTN  PICTURE X   VALUE  IS  "%".                    00017920
000690   02    DFHMT     PICTURE X   VALUE  IS  "".                     00017940
000700   02    DFHMFT    PICTURE X   VALUE  IS  "       ".                    00017960
000710   02    DFHMET    PICTURE X   VALUE  IS  "".                     00017980
000720   02    DFHMFET   PICTURE X   VALUE  IS  "'".                    00018000
000730                                                                  00018020
000740 01    DFHAID.                                                    00018040
000750   02  DFHNULL   PIC  X  VALUE IS "".                            00018060
000760   02  DFHENTER  PIC  X  VALUE IS QUOTE.                         00018080
000770   02  DFHCLEAR  PIC  X  VALUE IS "_".                           00018100
000780   02  DFHCLRP   PIC  X  VALUE IS "¬".                           00018120
000790   02  DFHPEN    PIC  X  VALUE IS "=".                           00018140
000800   02  DFHOPID   PIC  X  VALUE IS "W".                           00018160
000810   02  DFHMSRE   PIC  X  VALUE IS "X".                           00018180
000820   02  DFHSTRF   PIC  X  VALUE IS "h".                           00018200
000830   02  DFHTRIG   PIC  X  VALUE IS """".                          00018220
000840   02  DFHPA1    PIC  X  VALUE IS "%".                           00018240
000850   02  DFHPA2    PIC  X  VALUE IS ">".                           00018260
000860   02  DFHPA3    PIC  X  VALUE IS ",".                           00018280
000870   02  DFHPF1    PIC  X  VALUE IS "1".                           00018300
000880   02  DFHPF2    PIC  X  VALUE IS "2".                           00018320
000890   02  DFHPF3    PIC  X  VALUE IS "3".                           00018340
000900   02  DFHPF4    PIC  X  VALUE IS "4".                           00018360
000910   02  DFHPF5    PIC  X  VALUE IS "5".                           00018380
000920   02  DFHPF6    PIC  X  VALUE IS "6".                           00018400
000930   02  DFHPF7    PIC  X  VALUE IS "7".                           00018420
000940   02  DFHPF8    PIC  X  VALUE IS "8".                           00018440
000950   02  DFHPF9    PIC  X  VALUE IS "9".                           00018460
000960   02  DFHPF10   PIC  X  VALUE IS ":".                           00018490
000970   02  DFHPF11   PIC  X  VALUE IS "#".                           00018520
000980   02  DFHPF12   PIC  X  VALUE IS "@".                           00018550
000990   02  DFHPF13   PIC  X  VALUE IS "A".                           00018580
001000   02  DFHPF14   PIC  X  VALUE IS "B".                           00018610
001010   02  DFHPF15   PIC  X  VALUE IS "C".                           00018640
001020   02  DFHPF16   PIC  X  VALUE IS "D".                           00018670
001030   02  DFHPF17   PIC  X  VALUE IS "E".                           00018700
001040   02  DFHPF18   PIC  X  VALUE IS "F".                           00018730
001050   02  DFHPF19   PIC  X  VALUE IS "G".                           00018760
```

Figure 11 (Part 2 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  3
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

001060   02  DFHPF20   PIC  X  VALUE IS "H".                              00018790
001070   02  DFHPF21   PIC  X  VALUE IS "I".                              00018820
001080   02  DFHPF22   PIC  X  VALUE IS "¢".                              00018850
001090   02  DFHPF23   PIC  X  VALUE IS ".".                              00018880
001100   02  DFHPF24   PIC  X  VALUE IS "<".                              00018910
001110                                                                    00018940
001120                                                                    00018970
001130 01  PSBNAME PIC X(8).                                             00019000
001140 01  DLIO PIC X(70).                                               00020000
001150 01  SSA1.                                                         00021000
001160     02  FILLER PIC X(19) VALUE "ID      (NUM      =".            00022000
001170     02  SSA1KEY  PIC X(5).                                        00023000
001180     02  FILLER  PIC X VALUE ")".                                  00024000
001190 01  SSA2.                                                         00025000
001200     02  FILLER PIC X(19) VALUE "CHEQUE  (COMPTE   =".            00026000
001210     02  SSA2KEY PIC X(5).                                         00027000
001220     02  FILLER PIC X VALUE ")".                                   00028000
001230 01  SSA3.                                                         00029000
001240     02  FILLER PIC X(19) VALUE "PRET    (PRENUM   =".            00030000
001250     02  SSA3KEY PIC X(6).                                         00031000
001260     02  FILLER PIC X VALUE ")".                                   00032000
*OLD** 01  MAP1I COPY ABJCQIN.                                          00033000          ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
001270 01  MAP1I. COPY ABJCQIN REPLACING ==01  MAP1I.== BY ====.         00033000
000010 01  MAP1I.                                                                        +
000020     02  FILLER PIC X(12).                                                      +
000030     02  TITLEL    COMP  PIC  S9(4).                                            +
000040     02  TITLEF    PICTURE X.                                                   +
000050     02  FILLER REDEFINES TITLEF.                                               +
000060       03 TITLEA    PICTURE X.                                                 +
000070     02  TITLEI  PIC X(35).                                                     +
000080     02  CUSTNOL    COMP  PIC  S9(4).                                           +
000090     02  CUSTNOF    PICTURE X.                                                  +
000100     02  FILLER REDEFINES CUSTNOF.                                              +
000110       03 CUSTNOA    PICTURE X.                                                 +
000120     02  CUSTNOI  PIC X(5).                                                     +
000130     02  CHECKNOL    COMP  PIC  S9(4).                                          +
000140     02  CHECKNOF    PICTURE X.                                                 +
000150     02  FILLER REDEFINES CHECKNOF.                                             +
000160       03 CHECKNOA    PICTURE X.                                               +
000170     02  CHECKNOI  PIC X(5).                                                    +
000180     02  LOANNOL    COMP  PIC  S9(4).                                           +
000190     02  LOANNOF    PICTURE X.                                                  +
000200     02  FILLER REDEFINES LOANNOF.                                              +
000210       03 LOANNOA    PICTURE X.                                                 +
000220     02  LOANNOI  PIC X(6).                                                     +
000230 01  MAP1O REDEFINES MAP1I.                                                     +
000240     02  FILLER PIC X(12).                                                      +
000250     02  FILLER PICTURE X(3).                                                   +
000260     02  TITLEO  PIC X(35).                                                     +
000270     02  FILLER PICTURE X(3).                                                   +
000280     02  CUSTNOO  PIC X(5).                                                     +
000290     02  FILLER PICTURE X(3).                                                   +
000300     02  CHECKNOO  PIC X(5).                                                    +
```

Figure 11 (Part 3 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  4
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000310     02  FILLER PICTURE X(3).                                                  +
000320     02  LOANNOO  PIC X(6).                                                    +
000330 01  MAP2I.                                                                   +
000340     02  FILLER PIC X(12).                                                     +
000350     02  ERRNAMEL    COMP  PIC  S9(4).                                         +
000360     02  ERRNAMEF    PICTURE X.                                                +
000370     02  FILLER REDEFINES ERRNAMEF.                                            +
000380       03 ERRNAMEA    PICTURE X.                                              +
000390     02  ERRNAMEI  PIC X(8).                                                   +
000400     02  ERRNOL    COMP  PIC  S9(4).                                           +
000410     02  ERRNOF    PICTURE X.                                                  +
000420     02  FILLER REDEFINES ERRNOF.                                              +
000430       03 ERRNOA    PICTURE X.                                                +
000440     02  ERRNOI  PIC X(6).                                                     +
000450 01  MAP2O REDEFINES MAP2I.                                                   +
000460     02  FILLER PIC X(12).                                                     +
000470     02  FILLER PICTURE X(3).                                                  +
000480     02  ERRNAMEO  PIC X(8).                                                   +
000490     02  FILLER PICTURE X(3).                                                  +
000500     02  ERRNOO  PIC X(6).                                                     +
001280 LINKAGE SECTION.                                                  00034000
*OLD** 01  DFHBLLDS SYNCHRONIZED.          2                           00035000          ABJ6203 00 BLL'S ARE REMOVED
001290*01  DFHBLLDS SYNCHRONIZED.                                       00035000
*OLD**   02  BLLCBAR PICTURE XXXX.                                     00035200
001300*  02  BLLCBAR PICTURE XXXX.                                      00035200
*OLD**   02  CSACBAR PICTURE XXXX.                                     00035400
001310*  02  CSACBAR PICTURE XXXX.                                      00035400
*OLD**   02  CSAOPBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.           00035600
001320*  02  CSAOPBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.             00035600
*OLD**   02  TCACBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.            00035800
001330*  02  TCACBAR PICTURE S9(8) USAGE IS COMPUTATIONAL.              00035800
*OLD**     02  PCB-LIST-PTR  PIC S9(8) COMP.                          00036000
001340*    02  PCB-LIST-PTR  PIC S9(8) COMP.                            00036000
*OLD**     02  PCB1-PTR  PIC S9(8) COMP.                              00037000
001350*    02  PCB1-PTR  PIC S9(8) COMP.                                00037000
*OLD**     02  CINQOUT-PTR PIC S9(8) COMP.                            00038000
001360*    02  CINQOUT-PTR PIC S9(8) COMP.                              00038000
*OLD**     02  ERRORMP-PTR PIC S9(8) COMP.                            00039000
001370*    02  ERRORMP-PTR PIC S9(8) COMP.                              00039000
*OLD**     02  CIDLOUT-PTR PIC S9(8) COMP.                            00040000
001380*    02  CIDLOUT-PTR PIC S9(8) COMP.                              00040000
001390 01  DFHCSADS SYNCHRONIZED.                                       00041000
001400   02  CSAFILLER PICTURE X(512).                                  00041005
001410   02  FILLER1 REDEFINES CSAFILLER.                               00041010
001420     03  FILLER.                                                  00041015
001430       04  FILLER PICTURE X(76).                                  00041020
001440       04  CSACDTA PICTURE S9(8) USAGE IS COMPUTATIONAL.          00041025
001450       04  CSATODP PICTURE S9(7) USAGE IS COMPUTATIONAL-3.        00041030
001460       04  FILLER PICTURE X(12).                                  00041035
001470       04  CSACTODB PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041040
001480       04  FILLER PICTURE X(24).                                  00041045
001490       04  CSAJYDP PICTURE 9(7) USAGE IS COMPUTATIONAL-3.         00041050
001500       04  FILLER PICTURE X(64).                                  00041055
```

Figure 11 (Part 4 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  5
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

001510     03  FILLER.                                                  00041060
001520       04  FILLER PICTURE X(8).                                   00041065
001530       04  CSAOPFLA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041070
001540       04  FILLER PICTURE X(20).                                  00041075
001550       04  FILLER.                                                00041080
001560         05  CSAKCNAC PICTURE XXXX.                               00041085
001570         05  CSASCNAC PICTURE XXXX.                               00041090
001580         05  CSAPCNAC PICTURE XXXX.                               00041095
001590         05  CSAICNAC PICTURE XXXX.                               00041100
001600         05  CSADCNAC PICTURE XXXX.                               00041105
001610         05  CSATCNAC PICTURE XXXX.                               00041110
001620         05  CSAFCNAC PICTURE XXXX.                               00041115
001630         05  CSATDNAC PICTURE XXXX.                               00041120
001640         05  CSATSNAC PICTURE XXXX.                               00041125
001650         05  CSASANAC PICTURE XXXX.                               00041130
001660         05  CSATRNAC PICTURE XXXX.                               00041135
001670         05  CSAPINAC PICTURE XXXX.                               00041140
001680         05  FILLER PICTURE X(4).                                 00041145
001690         05  CSASPNAC PICTURE XXXX.                               00041150
001700         05  CSATCRWE PICTURE XXXX.                               00041155
001710     03  FILLER PICTURE X(215).                                   00041160
001720     03  CSAUTA1 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.          00041165
001730     03  CSAUTA2 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.          00041170
001740     03  CSAUTA3 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.          00041175
001750     03  CSAUTA4 PICTURE S9(5) USAGE IS COMPUTATIONAL-3.          00041180
001760     03  FILLER  PICTURE X(1).                                    00041185
001770*    ABOVE FILLER ADDED BY APAR PN26174                           00041190
001780                                                                  00041195
001790                                                                  00041200
001800 01  DFHTCADS PICTURE X(64) SYNCHRONIZED.                         00041205
001810 01  CSAOPFL REDEFINES DFHTCADS SYNCHRONIZED.                     00041210
001820   02  CSAATP PICTURE XXXX.                                       00041215
001830   02  CSAATTCH PICTURE XXXX.                                     00041220
001840   02  CSADLI PICTURE XXXX.                                       00041225
001850   02  CSABFNAC PICTURE XXXX.                                     00041230
001860   02  CSABMS PICTURE XXXX.                                       00041235
001870   02  CSATMSVT PICTURE XXXX.                                     00041240
001880   02  CSAJCNA1 PICTURE XXXX.                                     00041245
001890   02  CSAJCNA2 PICTURE XXXX.                                     00041250
001900   02  CSASRNAC PICTURE XXXX.                                     00041255
001910   02  CSASRTBA PICTURE XXXX.                                     00041260
001920   02  CSAKPNAC PICTURE XXXX.                                     00041265
001930   02  CSAATMSP PICTURE XXXX.                                     00041270
001940   02  CSAXLTBA PICTURE XXXX.                                     00041275
001950   02  CSAJCTBA PICTURE XXXX.                                     00041280
001960 01  DFHTCA SYNCHRONIZED.                                         00041285
001970   02  FILLER.                                                    00041290
001980     03  FILLER PICTURE X(8).                                     00041295
001990     03  TCAFCAAA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00041300
002000     03  FILLER REDEFINES TCAFCAAA.                               00041305
002010       04  TCAFCAA1 PICTURE X.                                    00041310
002020       04  FILLER PICTURE X(3).                                   00041315
002030     03 FILLER.                                                   00041320
```

Figure 11 (Part 5 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  6
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

002040       04  FILLER PICTURE X(8).                                    00041325
002050       04  TCATCEA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00041330
002060       04  FILLER REDEFINES TCATCEA.                               00041335
002070         05  TCATCQA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041340
002080       04  TCATCTR1 PICTURE  9(4) USAGE IS COMPUTATIONAL.          00041345
002090       04  FILLER REDEFINES TCATCTR1.                              00041350
002100         05  TCATCEI PICTURE X.                                    00041355
002110         05  TCATCTR PICTURE X.                                    00041360
002120       04  FILLER REDEFINES TCATCTR1.                              00041365
002130         05  TCATCDC PICTURE X.                                    00041370
002140         05  FILLER PICTURE X.                                     00041375
002150       04  TCATCDP PICTURE X.                                      00041380
002160       04  FILLER PICTURE X(5).                                    00041385
002170       04  TCATCRS PICTURE X(60).                                  00041390
002180       04  FILLER REDEFINES TCATCRS.                               00041395
002190         05  TCATCDP1 PICTURE  9(4) USAGE IS COMPUTATIONAL.        00041400
002200         05  FILLER PICTURE X(58).                                 00041405
002210     03  FILLER.                                                   00041410
002220       04  TCASCCA.                                                00041415
002230         05  TCASCSA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041420
002240       04  FILLER REDEFINES TCASCCA.                               00041425
002250         05  TCAFCTL PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041430
002260       04  FILLER REDEFINES TCASCCA.                               00041435
002270         05  TCASCTR PICTURE X.                                    00041440
002280         05  TCASCIB PICTURE X.                                    00041445
002290         05  TCASCNB PICTURE  9(4) USAGE IS COMPUTATIONAL.         00041450
002300       04  FILLER REDEFINES TCASCCA.                               00041455
002310         05  TCASCRI PICTURE  9(4) USAGE IS COMPUTATIONAL.         00041460
002320         05  FILLER PICTURE X(2).                                  00041465
002330       04  TCAFCTL1 PICTURE S9(8) USAGE IS COMPUTATIONAL.          00041470
002340       04  FILLER PICTURE X(28).                                   00041475
002350     03  FILLER.                                                   00041480
002360       04  TCACCCA.                                                00041485
002370         05  TCACCCA1 PICTURE X(32).                               00041490
002380         05  TCACCRS1 PICTURE X(56).                               00041495
002390         05  TCACCSV1 PICTURE S9(4) USAGE IS COMPUTATIONAL.        00041500
002400         05  TCACCRSV PICTURE XX.                                  00041505
002410         05  TCACCSV2 PICTURE XXXX.                                00041510
002420       04  FILLER REDEFINES TCACCCA.                               00041515
002430         05  TCATPAPR PICTURE X.                                   00041520
002440           88  TCATPVAL   VALUE  "6".                              00041525
002450           88  TCATPNVL   VALUE  "7".                              00041530
002460           88  TCATPLNR   VALUE  "".                              00041535
002470         05  FILLER PICTURE X.                                     00041540
002480         05  TCATPOS PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041545
002490         05  TCATPCS PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041550
002500         05  TCATPOC PICTURE S9(4) USAGE IS COMPUTATIONAL.         00041555
002510         05  TCATPLDM PICTURE XX.                                  00041560
002520         05  TCATPCON PIC S9(4) USAGE IS COMPUTATIONAL.            00041565
002530         05  TCATPPNM PICTURE X(8).                                00041570
002540         05  FILLER PICTURE X(76).                                 00041575
002550       04  FILLER REDEFINES TCACCCA.                               00041580
002560         05  FILLER PICTURE X(24).                                 00041585
```

Figure 11 (Part 6 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  7
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

002570         05  TCAKCTI PICTURE X(4).                                 00041590
002580         05  TCAKCFA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041595
002590         05  FILLER PICTURE X(64).                                 00041600
002600       04  FILLER REDEFINES TCACCCA.                               00041605
002610         05  TCAICDA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041610
002620         05  FILLER REDEFINES TCAICDA.                             00041615
002630           06  TCAICTR PICTURE  9(4) USAGE IS COMPUTATIONAL.       00041620
002640           06  FILLER PICTURE X(2).                                00041625
002650         05  FILLER REDEFINES TCAICDA.                             00041630
002660           06  TCAICRC PICTURE X.                                  00041635
002670           06  FILLER PICTURE X(3).                                00041640
002680         05  TCAICQID.                                             00041645
002690             07  TCAICQPX PICTURE XX.                              00041650
002700             07  FILLER   PICTURE X(6).                            00041655
002710           05  TCAICRT PICTURE S9(7) USAGE IS COMPUTATIONAL-3.     00041660
002720         05  TCAICTI PICTURE X(4).                                 00041665
002730         05  TCAICTID PICTURE X(4).                                00041670
002740         05  FILLER PICTURE X(4).                                  00041675
002750         05  TCAFCTR1 PICTURE  9(4) USAGE IS COMPUTATIONAL.        00041680
002760         05  FILLER PICTURE X(66).                                 00041685
002770       04  FILLER REDEFINES TCACCCA.                               00041690
002780         05  TCAPCLA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00041695
002790         05  FILLER REDEFINES TCAPCLA.                             00041700
002800           06  TCAPCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.       00041705
002810           06  FILLER PICTURE X(2).                                00041710
002820         05  FILLER REDEFINES TCAPCLA.                             00041715
002830           06  TCAPCRC PICTURE X.                                  00041720
002840             88  PCPGMIDER   VALUE "".                            00041725
002850             88  PCNORESP    VALUE "".                            00041730
002860             88  ICNORESP    VALUE "".                            00041735
002870             88  ICENDDATA   VALUE "".                            00041740
002880             88  ICIOERROR   VALUE "".                            00041745
002890             88  ICTRNIDER   VALUE "│".                            00041750
002900             88  ICTRMIDER   VALUE "└".                            00041755
002910             88  ICTSINVLD   VALUE "".                            00041760
002920             88  ICEXPIRD    VALUE "".                            00041765
002930             88  ICNOTFND    VALUE "£".                           00041770
002940             88  ICINVREQ    VALUE "".                            00041775
002950             88  TSNORESP    VALUE "".                            00041780
002960             88  TSENERROR   VALUE "".                            00041785
002970             88  TSIDERROR   VALUE "".                            00041790
002980             88  TSIOERROR   VALUE "".                            00041795
002990             88  TSINVREQ    VALUE "".                            00041800
003000             88  TDNORESP    VALUE "".                            00041805
003010             88  TDQUEZERO   VALUE "".                            00041810
003020             88  TDIDERROR   VALUE "".                            00041815
003030             88  TDIOERROR   VALUE "".                            00041820
003040             88  TDNOTOPEN   VALUE ";".                           00041825
003050             88  TDNOSPACE   VALUE "┌".                           00041830
003060             88  FCNORESP    VALUE "".                            00041835
003070             88  FCDSIDER    VALUE "".                            00041840
003080             88  FCSEGIDER   VALUE "".                            00041845
003090             88  FCINVREQ    VALUE ";".                           00041850
```

Figure 11 (Part 7 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  8
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

003100             88  FCDUPDS     VALUE ″″.                              00041855
003110             88  FCNOTOPEN   VALUE ″″.                              00041860
003120             88  FCENDFILE   VALUE ″″.                              00041865
003130             88  FCIOERROR   VALUE ″Ø″.                              00041870
003140             88  FCNOTFND    VALUE ″a″.                              00041875
003150             88  FCDUPREC    VALUE ″b″.                              00041880
003160             88  FCNOSPACE   VALUE ″c″.                              00041885
003170             88  FCDUPKEY    VALUE ″d″.                              00041890
003180             88  FCILLOGIC   VALUE ″″.                              00041896
003190           06  TCAPCFLA PICTURE X.                                   00041902
003200           06  TCAPCARO PICTURE X.                                   00041908
003210           06  FILLER PICTURE X.                                     00041914
003220         05  TCAPCPI PICTURE X(8).                                   00041920
003230         05  FILLER REDEFINES TCAPCPI.                               00041926
003240           06  TCAPCERA PICTURE S9(8) USAGE IS COMPUTATIONAL.        00041932
003250           06  FILLER PICTURE X(4).                                  00041938
003260         05  TCAPCAC PICTURE XXXX.                                   00041944
003270         05  TCAPCPSW PICTURE X(8).                                  00041950
003280         05  TCAPCINT PICTURE X(8).                                  00041956
003290         05  FILLER PICTURE X(64).                                   00041962
003300       04  FILLER REDEFINES TCACCCA.                                 00041968
003310         05  TCADCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041974
003320         05  TCADCNB PICTURE  9(4) USAGE IS COMPUTATIONAL.           00041980
003330         05  TCADCSA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00041986
003340         05  FILLER PICTURE XXXX.                                    00041992
003350         05  TCADCDC PICTURE XXXX.                                   00041998
003360         05  FILLER PICTURE X(80).                                   00042004
003370       04  FILLER REDEFINES TCACCCA.                                 00042010
003380         05  TCAFCAA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042016
003390         05  FILLER REDEFINES TCAFCAA.                               00042022
003400           06  TCAFCTR PICTURE  9(4) USAGE IS COMPUTATIONAL.         00042028
003410           06  FILLER PICTURE X(2).                                  00042034
003420         05  FILLER REDEFINES TCAFCAA.                               00042040
003430           06  TCAFCRC PICTURE X.                                    00042046
003440           06  FILLER PICTURE X(3).                                  00042052
003450         05  TCAFCDI PICTURE X(8).                                   00042058
003460         05  TCAFCURL PICTURE  9(4) USAGE IS COMPUTATIONAL.          00042064
003470         05  FILLER REDEFINES TCAFCURL.                              00042070
003480           06  TCAFCNRD PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042076
003490         05  FILLER  PICTURE X(6).                                   00042082
003500         05  FILLER  PICTURE X(8).                                   00042088
003510         05  TCAFCRI PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042094
003520         05  FILLER PICTURE X(64).                                   00042100
003530       04  FILLER REDEFINES TCACCCA.                                 00042106
003540         05  TCATDAA PICTURE S9(8) USAGE IS COMPUTATIONAL.           00042112
003550         05  FILLER REDEFINES TCATDAA.                               00042118
003560           06  TCATDRC PICTURE X.                                    00042124
003570           06  FILLER PICTURE X(3).                                  00042130
003580         05  FILLER REDEFINES TCATDAA.                               00042136
003590           06  TCATDTR PICTURE  9(4) USAGE IS COMPUTATIONAL.         00042142
003600           06  FILLER PICTURE X(2).                                  00042148
003610         05  TCATDDI PICTURE XXXX.                                   00042154
003620         05  FILLER PICTURE X(88).                                   00042160
```

Figure 11 (Part 8 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE  9
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

003630       04  FILLER REDEFINES TCACCCA.                                00042166
003640         05  TCATSDA PICTURE S9(8) USAGE IS COMPUTATIONAL.          00042172
003650         05  FILLER REDEFINES TCATSDA.                              00042178
003660           06  TCATSRC PICTURE X.                                   00042184
003670           06  FILLER PICTURE X(3).                                 00042190
003680         05  FILLER REDEFINES TCATSDA.                              00042196
003690           06  TCATSTR PICTURE  9(4) USAGE IS COMPUTATIONAL.        00042202
003700           06  FILLER PICTURE X(2).                                 00042208
003710         05  TCATSDI PICTURE X(8).                                  00042214
003720         05  TCATSRN PICTURE  9(4) USAGE IS COMPUTATIONAL.          00042220
003730         05  FILLER PICTURE  X(2).                                  00042226
003740         05  TCATSTR2 PICTURE  9(4) USAGE IS COMPUTATIONAL.         00042232
003750         05  FILLER PICTURE X(78).                                  00042238
003760       04  FILLER REDEFINES TCACCCA.                                00042244
003770         05  TCAMSTR1 PICTURE X(8).                                 00042250
003780         05  FILLER REDEFINES TCAMSTR1.                             00042256
003790           06  FILLER PICTURE X.                                    00042262
003800           06  TCAMSTR2 PICTURE X.                                  00042268
003810           06  TCAMSTR3 PICTURE X.                                  00042274
003820           06  TCAMSTR4 PICTURE X.                                  00042280
003830           06  TCAMSTR5 PICTURE X.                                  00042286
003840           06  TCAMSTR6 PICTURE X.                                  00042292
003850           06  TCAMSTR7 PICTURE X.                                  00042298
003860           06  TCAMSTR8 PICTURE X.                                  00042304
003870         05  FILLER REDEFINES TCAMSTR1.                             00042310
003880           06  TCAMSRC1 PICTURE X.                                  00042316
003890           06  TCAMSRC2 PICTURE X.                                  00042322
003900           06  TCAMSRC3 PICTURE X.                                  00042328
003910           06  TCAMSRI1 PICTURE X.                                  00042334
003920           06  TCAMSPGN PICTURE  9(4) USAGE IS COMPUTATIONAL.       00042340
003930           06  TCAMSOCN PICTURE  9(4) USAGE IS COMPUTATIONAL.       00042346
003940         05  FILLER REDEFINES TCAMSTR1.                             00042352
003950           06  FILLER PICTURE XX.                                   00042358
003960           06  TCAMSRC3H PICTURE 9(4) USAGE IS COMPUTATIONAL.       00042364
003970           06  FILLER PICTURE X(4).                                 00042370
003980         05  FILLER REDEFINES TCAMSTR1.                             00042376
003990           06  TCAMSRC PICTURE XXX.                                 00042382
004000           06  FILLER PICTURE X(5).                                 00042388
004010         05  TCAMSIOA PICTURE S9(8) USAGE IS COMPUTATIONAL.         00042394
004020         05  FILLER REDEFINES TCAMSIOA.                             00042400
004030           06  TCAMSTA PICTURE S9(8) USAGE IS COMPUTATIONAL.        00042406
004040         05  TCAMSFSC PICTURE XXXX.                                 00042412
004050         05  FILLER REDEFINES TCAMSFSC.                             00042418
004060           06  TCABMSFB PICTURE  9(4) USAGE IS COMPUTATIONAL.       00042424
004070           06  FILLER REDEFINES TCABMSFB.                           00042430
004080             07  TCABMSWC PICTURE X.                                00042436
004090             07  FILLER PICTURE X.                                  00042442
004100           06  FILLER REDEFINES TCABMSFB.                           00042448
004110             07  TCAMSWCC PICTURE X.                                00042454
004120             07  TCAMSJ PICTURE X.                                  00042460
004130           06  TCABMSCP PICTURE S9(4) USAGE IS COMPUTATIONAL.       00042466
004140         05  TCABMSMN PICTURE X(8).                                 00042472
004150         05  FILLER REDEFINES TCABMSMN.                             00042478
```

Figure 11 (Part 9 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE 10
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

004160           06  TCABMSMA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042484
004170           06  FILLER PICTURE X(4).                              00042490
004180         05  FILLER REDEFINES TCABMSMN.                          00042496
004190           06  TCAMSHDR PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042502
004200           06  FILLER PICTURE X(4).                              00042508
004210         05  FILLER REDEFINES TCABMSMN.                          00042514
004220           06  TCAMSRLA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042520
004230           06  TCAMSRTI PICTURE S9(7) USAGE IS COMPUTATIONAL-3.   00042526
004240           06  FILLER REDEFINES TCAMSRTI.                        00042532
004250             07  TCAMSTRL PICTURE S9(8) USAGE IS COMPUTATIONAL.   00042538
004260         05  TCAMSMSN PICTURE X(8).                              00042544
004270         05  FILLER REDEFINES TCAMSMSN.                          00042550
004280           06  TCAMSMSA PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042556
004290           06  FILLER PICTURE X(4).                              00042562
004300         05  FILLER REDEFINES TCAMSMSN.                          00042568
004310           06  TCAMSTI PICTURE X(4).                             00042574
004320           06  FILLER PICTURE X.                                 00042580
004330           06  TCAMSOC PICTURE XXX.                              00042586
004340         05  TCAMSLDM PICTURE XX.                                00042592
004350         05  TCAMSLDC PICTURE X.                                 00042598
004360         05  TCAMSRID PICTURE XX.                                00042604
004370         05  FILLER PICTURE XXX.                                 00042610
004380         05  TCAMSFMP PICTURE X(8).                              00042616
004390         05  FILLER PICTURE X(48).                               00042622
004400       04  FILLER REDEFINES TCACCCA.                             00042628
004410         05  TCASPTR PICTURE  9(4) USAGE IS COMPUTATIONAL.       00042634
004420         05  FILLER PICTURE X(94).                               00042640
004430       04  FILLER REDEFINES TCACCCA.                             00042646
004440         05  TCADLIO PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042652
004450         05  FILLER REDEFINES TCADLIO.                           00042658
004460           06  FILLER PICTURE X.                                 00042664
004470           06  TCADLTR PICTURE X.                                00042670
004480             88  FCDLINA   VALUE "".                            00042676
004490             88  FCPSBSCH  VALUE "".                            00042682
004500             88  FCPSBNF   VALUE "".                            00042688
004510             88  FCTASKNA  VALUE "".                            00042694
004520             88  FCPSBNA   VALUE " ".                            00042700
004530             88  FCLANGCON VALUE "".                            00042706
004540             88  FCPSBFAIL VALUE "  ".                           00042712
004550             88  FCFUNCNS  VALUE ";".                           00042718
004560             88  FCTERMNS  VALUE "'".                           00042724
004570           06  FILLER PICTURE X(2).                              00042730
004580         05  TCADLPCB PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042736
004590         05  TCADLPSB PICTURE X(8).                              00042742
004600         05  TCADLSSA PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042748
004610         05  TCADLPAR PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042754
004620         05  TCADLECB PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042760
004630         05  FILLER REDEFINES TCADLECB.                          00042766
004640           06  TCADLLAN PICTURE X(4).                            00042772
004650         05  TCADLFUN PICTURE X(4).                              00042778
004660         05  FILLER PICTURE X(64).                               00042784
004670       04  FILLER.                                               00042790
004680         05  TCATRF1 PICTURE S9(8) USAGE IS COMPUTATIONAL.       00042796
```

Figure 11 (Part 10 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE 11
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

004690         05  FILLER REDEFINES TCATRF1.                          00042802
004700           06  TCATRF1H PICTURE  9(4) USAGE IS COMPUTATIONAL.    00042808
004710           06  FILLER PICTURE X(2).                             00042814
004720         05  FILLER REDEFINES TCATRF1.                          00042820
004730           06  TCATRF1F PICTURE S9(8) USAGE IS COMPUTATIONAL.   00042826
004740         05  FILLER REDEFINES TCATRF1.                          00042832
004750           06  TCATRF1C PICTURE X(4).                           00042838
004760         05  FILLER REDEFINES TCATRF1.                          00042844
004770           06  TCATRF1P PICTURE 9(7) USAGE IS COMPUTATIONAL-3.  00042850
004780         05  FILLER REDEFINES TCATRF1.                          00042856
004790           06  TCATRF1A PICTURE X.                              00042862
004800           06  FILLER PICTURE X(3).                             00042868
004810         05  TCATRF2 PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042874
004820         05  FILLER REDEFINES TCATRF2.                          00042880
004830           06  TCATRF2H PICTURE  9(4) USAGE IS COMPUTATIONAL.    00042886
004840           06  FILLER PICTURE X(2).                             00042892
004850         05  FILLER REDEFINES TCATRF2.                          00042898
004860           06  TCATRF2F PICTURE S9(8) USAGE IS COMPUTATIONAL.   00042904
004870         05  FILLER REDEFINES TCATRF2.                          00042910
004880           06  TCATRF2C PICTURE X(4).                           00042916
004890         05  FILLER REDEFINES TCATRF2.                          00042922
004900           06  TCATRF2P PICTURE 9(7) USAGE IS COMPUTATIONAL-3.  00042928
004910         05  FILLER REDEFINES TCATRF2.                          00042934
004920           06  TCATRF2A PICTURE X.                              00042940
004930           06  FILLER PICTURE X(3).                             00042946
004940         05  TCATRRI PICTURE  9(4) USAGE IS COMPUTATIONAL.      00042952
004950         05  TCATRRI1 PICTURE  9(4) USAGE IS COMPUTATIONAL.     00042958
004960         05  FILLER PICTURE X(4).                               00042964
004970         05  TCAJCAAD PICTURE S9(8) USAGE IS COMPUTATIONAL.     00042970
004980         05  TCAATAC PICTURE S9(8) USAGE IS COMPUTATIONAL.      00042976
004990     03  FILLER.                                                00042982
005000       04  TCACSPE PICTURE XXXX.                                00042988
005010       04  TCANXTID PICTURE X(4).                               00042994
005020 01  PCB-ADDR.                                                  00043000
005030     02  PCB1-ADDR PIC S9(8) COMP.                              00044000
005040 01  PCB1.                                                      00045000
005050     02  DBD-NAME  PIC X(8).                                    00046000
005060     02  SEG-LEVEL  PIC XX.                                     00047000
005070     02  STATUS-CODE  PIC XX.                                   00048000
005080     02  PROC-OPTIONS  PIC X(4).                                00049000
005090     02  RESERVE-DLI  PIC S9(5) COMP.                           00050000
005100     02  SEG-NAME-FB  PIC X(8).                                 00051000
005110     02  LENGTH-FB-KEY PIC S9(5) COMP.                          00052000
005120     02  NUMB-SENS-SEGS PIC S9(5) COMP.                         00053000
005130     02  KEY-FB-AREA PIC X(30).                                 00054000
*OLD** 01  MAP11I COPY ABJCQOUT.                                     00055000          ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
005140 01  MAP11I. COPY ABJCQOUT REPLACING ==01  MAP11I.== BY ====.   00055000
000010 01  MAP11I.                                                                    +
000020     02  FILLER PIC X(96).                                                     +
000030 01  MAP11O REDEFINES MAP11I.                                                  +
000040     02  FILLER PIC X(96).                                                     +
000050 01  MAP21I REDEFINES MAP11I.                                                  +
000060     02  FILLER PIC X(12).                                                     +
```

Figure 11 (Part 11 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                   ABJIVP03         05 FEB 1997  10:09:49  PAGE 12
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

000070     02  SEGNAMEL    COMP  PIC  S9(4).                                          +
000080     02  SEGNAMEF    PICTURE X.                                                +
000090     02  FILLER REDEFINES SEGNAMEF.                                            +
000100       03 SEGNAMEA    PICTURE X.                                               +
000110     02  SEGNAMEI  PIC X(8).                                                   +
000120     02  SEGCONTL    COMP  PIC  S9(4).                                         +
000130     02  SEGCONTF    PICTURE X.                                                +
000140     02  FILLER REDEFINES SEGCONTF.                                            +
000150       03 SEGCONTA    PICTURE X.                                               +
000160     02  SEGCONTI  PIC X(70).                                                  +
000170 01  MAP21O REDEFINES MAP21I.                                                  +
000180     02  FILLER PIC X(12).                                                     +
000190     02  FILLER PICTURE X(3).                                                  +
000200     02  SEGNAMEO  PIC X(8).                                                   +
000210     02  FILLER PICTURE X(3).                                                  +
000220     02  SEGCONTO  PIC X(70).                                                  +
000230 01  MAP31I REDEFINES MAP11I.                                                  +
000240     02  FILLER PIC X(12).                                                     +
000250 01  MAP31O REDEFINES MAP31I.                                                  +
000260     02  FILLER PIC X(12).                                                     +
000270 01  MAP41I REDEFINES MAP11I.                                                  +
000280     02  FILLER PIC X(12).                                                     +
000290 01  MAP41O REDEFINES MAP41I.                                                  +
000300     02  FILLER PIC X(12).                                                     +
000310 01  MAP51I REDEFINES MAP11I.                                                  +
000320     02  FILLER PIC X(12).                                                     +
000330 01  MAP51O REDEFINES MAP51I.                                                  +
000340     02  FILLER PIC X(12).                                                     +
*OLD** 01  MAP12I COPY ABJERRMP.                                  00056000          ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
005150 01  MAP12I. COPY ABJERRMP REPLACING ==01  MAP12I.== BY ====.  00056000
000010 01  MAP12I.                                                                   +
000020     02  FILLER PIC X(12).                                                     +
000030     02  ERRMSGL    COMP  PIC  S9(4).                                          +
000040     02  ERRMSGF    PICTURE X.                                                 +
000050     02  FILLER REDEFINES ERRMSGF.                                             +
000060       03 ERRMSGA    PICTURE X.                                               +
000070     02  ERRMSGI  PIC X(70).                                                   +
000080 01  MAP12O REDEFINES MAP12I.                                                  +
000090     02  FILLER PIC X(12).                                                     +
000100     02  FILLER PICTURE X(3).                                                  +
000110     02  ERRMSGO  PIC X(70).                                                   +
*OLD** 01  MAP13I COPY ABJCIOUT.                                  00057000          ABJ6088 00 LANGLEVEL 1 COPY IS CHANGED
005160 01  MAP13I. COPY ABJCIOUT REPLACING ==01  MAP13I.== BY ====.  00057000
000010 01  MAP13I.                                                                   +
000020     02  FILLER PIC X(12).                                                     +
000030 01  MAP13O REDEFINES MAP13I.                                                  +
000040     02  FILLER PIC X(12).                                                     +
005170 PROCEDURE DIVISION.                                        00058000
*OLD**     MOVE CSACDTA TO TCACBAR.          3                    00059000          ABJ6207 00 BLL CONVERTED TO SET POINTER
005180     MOVE CSACDTA TO LCP-WS-ADDR-COMP                       00059000                     SET ADDRESS OF ...
005190     SET ADDRESS OF DFHTCA TO LCP-WS-ADDR-PNTR.                                ABJ6301 04 31 BIT ESA ADDRESSES WILL BE
                                                                                               TREATED AS NEGATIVE NUMBERS:
                                                                                               RESULTS MAY BE UNPREDICTABLE
```

Figure 11 (Part 12 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE 13
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

                                                                                                          *** MANUAL UPDATE RECOMMENDED
005200     EXEC CICS HANDLE CONDITION ERROR(ERRORS) MAPFAIL(CIDL)       00060000
005210         OVERFLOW(PAGE-OVERFLOW) END-EXEC.                        00061000
005220     EXEC CICS RECEIVE MAP("MAP1") MAPSET("CINQIN") END-EXEC.     00062000
005230     IF CUSTNOI = SPACES OR CUSTNOL = +0000                       00063000
005240         MOVE "CUSTOMER" TO ERRNAMEO                              00064000
005250         MOVE SPACES TO ERRNOO GO TO ERR-MSG.                     00065000
005260     MOVE "PSBCLIG" TO PSBNAME.                                   00066000
005270     CALL "CBLTDLI" USING PCB PSBNAME.                            00067000
005280     IF TCAFCRC NOT EQUAL TO "" GO TO INTERFACE-ERROR.           00068000
*OLD**     MOVE TCADLPCB TO PCB-LIST-PTR.          4                  00069000          ABJ6207 00 BLL CONVERTED TO SET POINTER
005290     MOVE TCADLPCB TO LCP-WS-ADDR-COMP                            00069000                     SET ADDRESS OF ...
005300     SET ADDRESS OF PCB-ADDR TO LCP-WS-ADDR-PNTR.                                      ABJ6301 04 31 BIT ESA ADDRESSES WILL BE
                                                                                                       TREATED AS NEGATIVE NUMBERS:
                                                                                                       RESULTS MAY BE UNPREDICTABLE
                                                                                                       *** MANUAL UPDATE RECOMMENDED
*OLD**     MOVE PCB1-ADDR TO PCB1-PTR.             5                  00070000          ABJ6207 00 BLL CONVERTED TO SET POINTER
005310     MOVE PCB1-ADDR TO LCP-WS-ADDR-COMP                           00070000                     SET ADDRESS OF ...
005320     SET ADDRESS OF PCB1 TO LCP-WS-ADDR-PNTR.                                          ABJ6301 04 31 BIT ESA ADDRESSES WILL BE
                                                                                                       TREATED AS NEGATIVE NUMBERS:
                                                                                                       RESULTS MAY BE UNPREDICTABLE
                                                                                                       *** MANUAL UPDATE RECOMMENDED
005330     MOVE CUSTNOI TO SSA1KEY.                                     00071000
005340     MOVE CHECKNOI TO SSA2KEY.                                    00072000
005350     MOVE LOANNOI TO SSA3KEY.                                     00073000
005360     IF SSA2KEY NOT = LOW-VALUE GO TO CHECK-PROC.                 00074000
005370     IF SSA3KEY NOT = LOW-VALUE GO TO LOAN-PROC.                  00075000
005380     CALL "CBLTDLI" USING GU PCB1 DLIO SSA1.                      00076000
005390     IF TCAFCRC NOT EQUAL TO "" GO TO INTERFACE-ERROR.           00077000
005400     IF STATUS-CODE = "  " GO TO GU-OK.                           00078000
005410     IF STATUS-CODE = "GE" MOVE "CUSTOMER" TO ERRNAMEO            00079000
005420         MOVE CUSTNOI TO ERRNOO                                   00080000
005430         GO TO ERR-MSG.                                           00081000
005440     GO TO ERROR1.                                                00082000
005450 CHECK-PROC.                                                      00083000
005460     MOVE CHECKNOI TO SSA2KEY.                                    00084000
005470     CALL "CBLTDLI" USING GU PCB1 DLIO SSA1 SSA2.                 00085000
005480     IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                  00086000
005490     IF STATUS-CODE = "  " GO TO GU-OK.                           00087000
005500     IF STATUS-CODE = "GE" MOVE "CHECK" TO ERRNAMEO               00088000
005510         MOVE CHECKNOI TO ERRNOO                                  00089000
005520         GO TO ERR-MSG.                                           00090000
005530 LOAN-PROC.                                                       00091000
005540     MOVE LOANNOI TO SSA3KEY.                                     00092000
005550     CALL "CBLTDLI" USING GU PCB1 DLIO SSA1 SSA3.                 00093000
005560     IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                  00094000
005570     IF STATUS-CODE = "  " GO TO GU-OK.                           00095000
005580     IF STATUS-CODE = "GE" MOVE "LOAN" TO ERRNAMEO                00096000
005590         MOVE LOANNOI TO ERRNOO                                   00097000
005600         GO TO ERR-MSG.                                           00098000
005610 GU-OK.                                                           00099000
005620     MOVE 1 TO PAGE-OVERFLOW-CTR.                                 00100000
*OLD**     EXEC CICS GETMAIN SET(CINQOUT-PTR) LENGTH(96) END-EXEC.  6   00101000          ABJ6201 00 POINTER OPTION IN EXEC CICS
```

Figure 11 (Part 13 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE 14
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

005630     EXEC CICS GETMAIN SET(ADDRESS OF MAP11I) LENGTH(96) END-EXEC.00101000                    CHANGED TO ADDRESS OF  ...
005640     EXEC CICS SEND MAP("MAP11") MAPSET("CINQOUT") ACCUM          00102000
005650         ERASE PAGING FRSET FREEKB END-EXEC.                      00103000
005660 PAGE-BUILD.                                                       00104000
005670     MOVE SEG-NAME-FB TO SEGNAMEO.                                00105000
005680     MOVE DLIO TO SEGCONTO.                                       00106000
005690 SEND-MAP2.                                                        00107000
005700     MOVE 2 TO PAGE-OVERFLOW-CTR.                                 00108000
005710     EXEC CICS SEND MAP("MAP21") MAPSET("CINQOUT") ACCUM          00109000
005720         PAGING FRSET FREEKB END-EXEC.                            00110000
005730 GNP-LOOP.                                                         00111000
005740     CALL "CBLTDLI" USING GNP PCB1 DLIO.                          00112000
005750     IF TCAFCRC NOT = "" GO TO INTERFACE-ERROR.                  00113000
005760     IF STATUS-CODE = "  " OR STATUS-CODE = "GA"                  00114000
005770         OR STATUS-CODE = "GK" GO TO PAGE-BUILD.                  00115000
005780     IF STATUS-CODE = "GE" OR STATUS-CODE = "GB"                  00116000
005790         GO TO END-GNP-LOOP.                                      00117000
005800     GO TO ERROR1.                                                00118000
005810 END-GNP-LOOP.                                                     00119000
005820     EXEC CICS SEND MAP("MAP31") MAPSET("CINQOUT") ACCUM          00120000
005830         PAGING FRSET FREEKB END-EXEC.                            00121000
005840     EXEC CICS SEND MAP("MAP41") MAPSET("CINQOUT") ACCUM          00122000
005850         PAGING FRSET FREEKB END-EXEC.                            00123000
005860 PAGE-OUT.                                                         00124000
005870     EXEC CICS SEND PAGE NOAUTOPAGE END-EXEC.                     00125000
005880 END-PROG.                                                         00126000
005890 PROG-RETURN.                                                      00127000
005900     CALL "CBLTDLI" USING TERM.                                   00128000
005910     EXEC CICS RETURN TRANSID("CINQ") END-EXEC.                   00129000
005920 ERRORS.                                                           00130000
005930     PERFORM SAVE-INFO.                                           00131000
005940     EXEC CICS DUMP DUMPCODE("ERRS") END-EXEC.                    00132000
005950     GO TO PROG-RETURN.                                           00133000
005960 CIDL.                                                             00134000
*OLD**     EXEC CICS GETMAIN SET(CIDLOUT-PTR) LENGTH(12) END-EXEC.  7  00135000         ABJ6201 00 POINTER OPTION IN EXEC CICS
005970     EXEC CICS GETMAIN SET(ADDRESS OF MAP13I) LENGTH(12) END-EXEC.00135000                    CHANGED TO ADDRESS OF  ...
005980     MOVE LOW-VALUE TO MAP13O.                                    00136000
005990     EXEC CICS SEND MAP("MAP13") MAPSET("CIDLOUT") ERASE END-EXEC.00137000
006000     EXEC CICS RETURN END-EXEC.                                   00138000
006010 PAGE-OVERFLOW.                                                    00139000
006020     EXEC CICS SEND MAP("MAP41") MAPSET("CINQOUT") ACCUM          00140000
006030         PAGING FREEKB END-EXEC.                                  00141000
006040     EXEC CICS SEND MAP("MAP11") MAPSET("CINQOUT") ACCUM          00142000
006050         ERASE PAGING FRSET FREEKB END-EXEC.                      00143000
006060     GO TO GU-OK SEND-MAP2 DEPENDING ON PAGE-OVERFLOW-CTR.        00144000
006070 ERR-MSG.                                                          00145000
006080     EXEC CICS SEND MAP("MAP1") MAPSET("CINQIN") ACCUM            00146000
006090         PAGING FREEKB END-EXEC.                                  00147000
006100     EXEC CICS SEND MAP("MAP2") MAPSET("CINQIN") ACCUM            00148000
006110         PAGING FREEKB END-EXEC.                                  00149000
006120     EXEC CICS SEND PAGE END-EXEC.                                00150000
006130     GO TO END-PROG.                                              00151000
006140 INTERFACE-ERROR.                                                  00152000
```

Figure 11 (Part 14 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03         05 FEB 1997  10:09:49  PAGE 15
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S  MSGID SEV --- D I A G N O S T I C S ---

006150     MOVE TCAFCRC TO SAVE-TCAFCRC.                                 00153000
006160     MOVE TCADLTR TO SAVE-TCADLTR.                                 00154000
006170     PERFORM SAVE-INFO.                                            00155000
006180     EXEC CICS DUMP DUMPCODE(”INTE”) END-EXEC.                     00156000
*OLD**     EXEC CICS GETMAIN SET(ERRORMP-PTR) LENGTH(85) END-EXEC.  8    00157000         ABJ6201 00 POINTER OPTION IN EXEC CICS
006190     EXEC CICS GETMAIN SET(ADDRESS OF MAP12I) LENGTH(85) END-EXEC.00157000                    CHANGED TO ADDRESS OF  ...
006200     MOVE ”*** INTERFACE ERROR. DUMP IN PROGRESS.***” TO ERRMSGO. 00158000
006210     EXEC CICS SEND MAP(”MAP12”) MAPSET(”ERRORMP”) ACCUM           00159000
006220         PAGING FREEKB END-EXEC.                                   00160000
006230     GO TO CIDL.                                                   00161000
006240 ERROR1.                                                           00162000
006250     PERFORM SAVE-INFO.                                            00163000
006260     EXEC CICS DUMP DUMPCODE(”ERRO”) END-EXEC.                     00164000
*OLD**     EXEC CICS GETMAIN SET(ERRORMP-PTR) LENGTH(85) END-EXEC.  9    00165000         ABJ6201 00 POINTER OPTION IN EXEC CICS
006270     EXEC CICS GETMAIN SET(ADDRESS OF MAP12I) LENGTH(85) END-EXEC.00165000                    CHANGED TO ADDRESS OF  ...
006280     MOVE ”*** DL/1 CALL ERROR. DUMP IN PROGRESS.***” TO ERRMSGO. 00166000
006290     EXEC CICS SEND MAP(”MAP12”) MAPSET(”ERRORMP”) ACCUM           00167000
006300         PAGING FREEKB END-EXEC.                                   00168000
006310     GO TO CIDL.                                                   00169000
006320 SAVE-INFO.                                                        00170000
006330     MOVE STATUS-CODE TO SAVE-STATUS-CODE.                         00171000
006340     MOVE TCACCCA TO SAVE-TCACCCA.                                 00172000
006350 END-PGM.                                                          00173000
006360     STOP RUN.                                                     00174000         ABJ6126 99 *----------------------------*
006370 END PROGRAM ABJIVP03.                                                                          *  END OF COBOL CONVERSION   *
                                                                                                      * 5785-CCC COBOL CONVERSION  *
                                                                                                      *----------------------------*
```

Figure 11 (Part 15 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                  ABJIVP03        05 FEB 1997  10:09:49  PAGE 16
CONVERSION FROM DOS/VS COBOL      TO  COBOL/VSE
OPTIONS IN EFFECT :
  PROCEDURE NAME CHECKING ........ YES   LANGLEVEL ..................... DOS/VS COBOL
  FLAG REPORT WRITER STMTS ....... YES   CICS ........................... CICS ST. WITH BLL
  REMOVE OBSOLETE ELEMENTS ....... YES   LINE COUNT ..................... 60
  GENERATE CALL ILBOABNO STMT..... YES   DATE FORMAT ................... MM/DD/YY
  GENERATE END PROGRAM STMT ...... YES   RESEQUENCING .................. YES
  POST-CONVERSION COMPILE ........ NO    INCREMENT ..................... 0010
  MANUAL CHANGE FLAGGING ......... YES   RESERVED WORD SUFFIX ........... 74
  HANDLE EXEC SQL INCLUDE AS COPY. YES   GENERATE NEW PROGRAM ........... YES
  REMOVE NON 88 VALUE CLAUSE IN FS YES   GENERATE NEW COPY .............. YES
  FLAG IF FILE-STATUS (NOT) = "00" YES   REPLACE LIKE-NAMED COPY MBR .... YES
  FLAG 31-BIT ADDRESS ARITHMETIC.. YES   PRINT REFERENCE SOURCE LINE .... YES
  INCL.W-S IN CICS COMPILE OF L-S. YES   PRINT COPY MODULE .............. YES
  OPTION-13 ...................... NO    LEVEL DIAGNOSTIC ............... 00
  OPTION-14 ...................... NO    DEBUG MODE ..................... 0
  OPTION-15 ...................... NO    SQL ............................ N
HIGHEST SEVERITY MESSAGE FOR THIS CONVERSION:   04          10
       0016 MESSAGES ISSUED
       0016 MESSAGES PRINTED
SEQNBR  CPYNBR    MSGID   RC  MESSAGE TEXT

000080           ABJ6212  00  WORKING POINTER FOR CICS ADDED TO WORKING STORAGE
001270           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
001290           ABJ6203  00  BLL'S ARE REMOVED
005140           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
005150           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
005160           ABJ6088  00  LANGLEVEL 1 COPY IS CHANGED
005180           ABJ6207  00  BLL CONVERTED TO SET POINTER SET ADDRESS OF ...
005180           ABJ6301  04  31 BIT ESA ADDRESSES WILL BE TREATED AS NEGATIVE NUMBERS: RESULTS MAY BE UNPREDICTABLE      11
                              *** MANUAL UPDATE RECOMMENDED
005290           ABJ6207  00  BLL CONVERTED TO SET POINTER SET ADDRESS OF ...
005290           ABJ6301  04  31 BIT ESA ADDRESSES WILL BE TREATED AS NEGATIVE NUMBERS: RESULTS MAY BE UNPREDICTABLE      12
                              *** MANUAL UPDATE RECOMMENDED
005310           ABJ6207  00  BLL CONVERTED TO SET POINTER SET ADDRESS OF ...
005310           ABJ6301  04  31 BIT ESA ADDRESSES WILL BE TREATED AS NEGATIVE NUMBERS: RESULTS MAY BE UNPREDICTABLE      13
                              *** MANUAL UPDATE RECOMMENDED
005630           ABJ6201  00  POINTER OPTION IN EXEC CICS CHANGED TO ADDRESS OF  ...
005970           ABJ6201  00  POINTER OPTION IN EXEC CICS CHANGED TO ADDRESS OF  ...
006190           ABJ6201  00  POINTER OPTION IN EXEC CICS CHANGED TO ADDRESS OF  ...
006270           ABJ6201  00  POINTER OPTION IN EXEC CICS CHANGED TO ADDRESS OF  ...
```

Figure 11 (Part 16 of 16). ABJIVP03 - CCCA/VSE Diagnostics Report

Notes:

1 If needed by the conversion of statements involving primary BLLs, these codes are generated in the Working-Storage Section for use with the POINTER facility.

2 If the CICS option on the Conversion panel is set to Y, the BLL definitions are removed. If the entire BLL structure is redefined, the redefined structure is removed. If the BLLs are not defined with a length of four bytes, the CICS conversion cannot be preformed. If the level 01 of the BLL structure is FILLER, the BLL definitions are not removed from the Linkage Section, but all of the references to BLLs in the Procedure Division are processed.

3 **4** **5** The primary BLL references are changed to ADDRESS OF special register and POINTER facility. For example:

- MOVE BLL1 TO BLL2 is changed to SET ADDRESS OF REC2 to ADDRESS OF REC1
- MOVE ID to BLL is changed to MOVE ID TO LCP-WS-ADDR-COMP SET ADDRESS OF REC TO LCP-WS-ADDR-PNTR
- MOVE BLL to ID is changed to SET LCP-WS-ADDR-PNTR TO ADDRESS OF REC MOVE LCP-WS-ADDR-COMP TO ID

6 **7** **8** **9** The primary BLLs are replaced by corresponding ADDRESS OF special register. For example: EXEC CICS READ ... SET(BLL1) ... is replaced by EXEC CICS READ ... SET(ADDRESS OF REC1)... The statements affected are GETMAIN, READ, READNEXT, READPREV, READQ, RECEIVE, RETRIEVE, SEND CONTROL, SEND PAGE, SEND TEXT, LOAD, CONVERSE, ISSUE RECEIVE, and POST.

10 The highest severity of 4 means that manual update is recommended. The reason for the severity 4 message is described in **11** **12** **13**.

11 **12** **13** You should review your source code where arithmetic is being done against any address field. In theory you could add a number large enough to make a negative number positive, that is transform from indicating a 31-bit address to a 24-bit address! This is where an error could occur. So you may want to alter your code so you either don't have to do arithmetic on address fields or handle the situation where the sign of your address field changes.

For details on the converted COBOL statements, see Appendix A. 'Converted COBOL Language Elements' in the *COBOL and CICS Command Level Conversion Aid for VSE Installation and User's Guide*.

6.4 Limitations of CCCA/VSE

- CCCA does not support total conversion and there are some incompatibilities. (See Chapter 5, "Significant Differences between Old COBOL and COBOL/VSE" on page 35 for details.)
- Exact knowledge of the input source level is necessary for exact conversion, otherwise the output source member will still contain old and rejected statements.
- Each member has to be individually identified, and diagnosed. The person running the tool has to be very meticulous in his activity, if the number of programs is very high. This approach of individual processing has its drawbacks.
- CCCA does not give an overall impact analysis before the start of a project. This kind of impact analysis report is very useful in proposal submission. This report can contain the following fields:
  - Name of Library
  - Number of Programs
  - Number of Lines of Code
  - Possible conversion required in Lines of Code
  - Percentage of Whole
- A Summary report after the whole job is done is useful in accessing the automated and manual work done on the overall project.
- The actual conversion process runs in batch mode and not on-line where user intervention can be called to change the output source code and also to change the statements if required.

Chapter 7. Debug Tool for VSE/ESA

This chapter summarizes:

- What is Debug Tool
- What do you need to run Debug Tool
- How to invoke Debug Tool
- How to debug your program in full-screen mode
- Limitations of Debug Tool

This chapter does not intend to give you all the information about Debug Tool. You can find detailed information in:

- *Debug Tool for VSE/ESA User's Guide and Reference*.
- *Debug Tool for VSE/ESA Installation and Customization Guide*.

This chapter is more a quick start guide which will help you start work with Debug Tool quickly and easily.

7.1 What is Debug Tool

Debug Tool for VSE/ESA is a powerful interactive source-level debug tool for application programs written in the following high-level languages:

- IBM C for VSE/ESA (C/VSE)
- IBM COBOL for VSE/ESA (COBOL/VSE)
- IBM PL/I for VSE/ESA (PL/I VSE)

You receive Debug Tool when you order the full function offering of any of the above IBM compiler products.

Debug Tool allows programmers to address difficult problems at the source level where they are comfortable. Debugging sessions may be performed in either interactive or batch mode. Debug Tool provides an interactive full-screen interface to a 3270 device. The full-screen interface is made up of session panel windows containing information about the debugging session. In batch mode, command files provide the mechanism to predefine a series of Debug Tool commands to be performed on an executing batch application. Neither terminal input nor user interaction is required in batch mode.

While a program is running, you can control and examine its execution with functions such as:

- Viewing the source listing and stepping through the source one statement at a time.
- Setting dynamic break points, which can be simple or conditional based on other values in the program.
- Monitoring the value of program variables.
- Modifying program and variable storage.
- Debugging mixed-language applications from a single debug session.

The debug session can be recorded in a log file, so you can replay the session. You can use Debug Tool to help capture test cases for future program validation or to further isolate a problem within an application.

7.2 What Do You Need to Run Debug Tool

The following sections describe system requirements for running Debug Tool.

7.2.1 Licensed Programs

Debug Tool runs under VSE/ESA with the required licensed programs listed in Table 6 and optional licensed programs listed in Table 7. You should install all licensed programs with the minimum release listed or with any subsequent release.

The licensed programs listed in Table 6 are required to install and customize Debug Tool, or to run Debug Tool.

Table 6. Required Licensed Programs for Debug Tool

Required Licensed Program	Minimum Release	Program Number
One of:		
VSE/ESA	Version 1 Release 4.3	5750-ACD
VSE/ESA	Version 2 Release 2	5690-VSE
One of:		
IBM LE/VSE including the C-specific base component	Version 1 Release 4	5686-094
VSE C Language Run-Time Support feature of VSE/ESA Version 2 Release 2		5690-VSE
One of:		
IBM C for VSE/ESA	Release 1	5686-A01
IBM COBOL for VSE/ESA •	Release 1	5686-068
IBM PL/I for VSE/ESA •	Release 1	5686-069

Note:

1. To run Debug Tool with output from this compiler the LE/VSE run-time library support for the compiler language is required.

The licensed programs listed in Table 7 can optionally be used with Debug Tool.

Table 7 (Page 1 of 2). Optional Licensed Programs for Debug Tool

Optional Licensed Program	Minimum Release	Program Number
CSP/AD	Version 3 Release 3	5668-813
CSP/AE	Version 3 Release 3	5668-814
DFSORT/VSE	Version 3 Release 1	5746-SM3
BookManager/Read	Release 2 (to view softcopy documentation)	73F6-023 (Read/2)
IBM DL/I DOS/VS	Release 10	5746-XX1
IBM DOS/VS Sort/Merge	Version 2 Release 5	5746-SM2

Table 7 (Page 2 of 2). Optional Licensed Programs for Debug Tool

Optional Licensed Program	Minimum Release	Program Number
QMF/VSE	Version 3 Release 2	5648-061
SQL/DS	Version 3 Release 4	5688-103

7.2.2 DASD Storage

When installing Debug Tool, you must provide DASD storage for the VSE Librarian library.

Table 8 lists the estimated minimum DASD storage required for VSE Librarian library storage. The storage requirements given in Table 8 are in addition to the storage requirements of any other licensed programs installed in the library. You should consider allowing 10% to 15% extra library storage for any future enhancements and service updates.

Table 8. Approximate Library Storage Requirements for Debug Tool

Component	LIBR BLKS •	3380 TRK	3390 TRK	9345 TRK	FBA BLKS
Debug Tool Base	4600	150	120	150	9200
Debug Tool Japanese NLF	220	8	7	8	440
Total	4820	158	127	158	9640

Note: One library block equals 1024 bytes.

7.2.3 VTAM Considerations

If you plan to use Debug Tool to debug your batch applications interactively you must add the Debug Tool VTAM definitions to your system by following the instructions on Page 18 of the *Debug Tool for VSE/ESA Installation and Customization Guide*.

7.2.4 CICS Considerations

If you plan to use Debug Tool to debug your CICS applications you must update your CICS system by following the instructions on Page 19 of the *Debug Tool for VSE/ESA Installation and Customization Guide*.

7.2.5 Debug Tool Run-time Environment

Debug Tool requires the LE/VSE run-time library and LE/VSE C-specific run-time library. To run Debug Tool you must install the base and C components of LE/VSE or the VSE C language Run-time Support feature of VSE/ESA Version 2 Release 2.

If you are writing your programs in COBOL/VSE or PL/I VSE then the LE/VSE run-time library components that match the language you are using must be installed.

7.2.6 VSE Partition Requirements

The minimum recommended partition size for batch programs running with Debug Tool is 6MB. Batch programs can run with Debug Tool in either a static or dynamic partition. The partition in which an interactive Debug Tool session starts is unavailable to other users for the duration of the Debug Tool session.

7.3 How to Invoke Debug Tool

You can invoke Debug Tool in any of the following ways:

1. Interactive debug with CICS
2. Batch debug using a command file
3. Batch execution with interactive debug session
4. Batch debug using CEETEST

In the following sessions, we use a sample COBOL program SAMPD5 to show you the various ways to invoke Debug Tool. The program SAMPD5 is as the following in Figure 12:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPD5.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  AA PIC 99.
77  BB PIC 99.
77  CC PIC 99.
77  DD PIC 99.
PROCEDURE DIVISION.
    MOVE 4 TO AA BB.
    MOVE 5 TO CC.
    MOVE 3 TO DD.
    IF AA = BB AND NOT LESS THAN CC OR DD DISPLAY '1'.
    IF (AA = BB) AND ( AA NOT < CC ) OR ( AA NOT < DD )
       DISPLAY '2'.
    STOP RUN.
```

Figure 12. SAMPD5 - COBOL Sample Program for Debug Tool

7.3.1 Interactive Debug with CICS

To use Debug Tool under CICS, you need to ensure that you have completed all of the required installation and configuration steps for CICS, LE/VSE, and Debug Tool. Please see the related LE/VSE and Debug Tool manuals for details.

In this section, you compile the program SAMPD5 with compile option TEST and link-edit it. Then you define the program SAMPD5 and a transaction SAM5 to CICS. After entering transaction code SAM5, you start the execution of the program SAMPD5. An interactive debug session starts immediately on the same terminal as the SAM5 transaction entered.

Before you compile SAMPD5, you need to assemble the LE/VSE default options module CEEUOPT. The sample job EQAWIVC3 already assembles and catalogs an LE/VSE options module CEEUOPT for the CICS environment. Therefore, you

can also use this options module for link-editing your SAMPD5 phase. The TEST option in the CEEXOPT macro sets the Debug Tool options for your CICS program. Modify the sample job control to meet your requirements before submitting the job. Figure 13 shows the job EQAWIVC3; tailoring requirements for each statement are discussed in the notes. You can find this job in your Debug Tool library.

```
* $$ JOB JNM=EQAWIVC3,CLASS=4          1
* $$ PUN DISP=I,PRI=6,CLASS=4
// JOB EQAWIVC3
// LIBDEF *,SEARCH=(PRD2.SCEEBASE)     2
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
// JOB EQAWIVC3
*
*       Step 2: Catalog module CEEUOPT.OBJ
*
// EXEC LIBR,PARM='MSHP;ACCESS SUBLIB=PRD2.TEMP'       3
* $$ END
*
*       Step 1: Assemble the LE/VSE default options
*
// OPTION DECK
// EXEC ASMA90,SIZE=ASMA90
   PUNCH ' CATALOG CEEUOPT.OBJ,REPLACE=YES'
   PRINT ON,NOGEN
CEEUOPT  CSECT
CEEUOPT  AMODE ANY
CEEUOPT  RMODE ANY
       CEEXOPT TEST=(ALL,*,PROMPT,*)        4
       END
/*
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
/*
#&
$ $$ EOJ
* $$ END
/&
* $$ EOJ
```

Figure 13. Job Control for EQAWIVC3

Notes:

1 Modify the VSE/POWER job control statements for your site.

2 If necessary, change the sublibrary to match the sublibrary where LE/VSE is installed.

3 Change 'SUBLIB=PRD2.TEMP' as appropriate for your site. This is where the options module CEEUOPT will be cataloged.

4 The TEST option in the CEEXOPT macro.

EQAWIVC3 creates two jobs and returns two output listings. The first job is the assembly of the default options module CEEUOPT; if successful, this job

completes with return code zero. The second job catalogs the object module CEEUOPT; if successful, this job completes with return code zero.

In order to invoke Debug Tool interactively under CICS, you need the following job control COM5N to compile and link-edit your program. You may need to modify the job control to meet your requirements before submitting the job. Tailoring requirements for each statement are discussed in the notes.

```
* $$ JOB JNM=COM5N,DISP=D,CLASS=4,NTFY=YES       1
* $$ LST DISP=D,CLASS=Q,PRI=3          +------------------- 2
// JOB COM5N COMPILE PROGRAM SAMPD5     |        +---------- 3
// LIBDEF *,SEARCH=(PRD2.TEMP,PRD2.SCEEBASE,PRD2.PROD)       4
// SETPARM CATALOG=1       |_____________________________________|
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=PRD2.TEMP         4
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE SAMPD5,*
   INCLUDE CEEUOPT       5
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL,PARM='EXIT(PRTEXIT(EQALIST))'   6
 CBL LIB,APOST,NOADV,RENT,BUF(4096),TEST      7
* $$ SLI ICCF=(SAMPD5),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT
/. NOLNK
/&
* $$ EOJ
```

Figure 14. COM5N - Interactive Debug with CICS

Notes:

1 Make sure the partition size is big enough to run Debug Tool interactively.

2 If necessary, change the sublibrary to match the sublibrary where LE/VSE is installed.

3 If necessary, change the sublibrary to match the sublibrary where Debug Tool and COBOL/VSE are installed (Debug Tool and COBOL/VSE might not be installed in the same sublibrary).

4 Debug Tool writes a profile settings file (member-name. DTSAFE) to the first sublibrary in the LIBDEF SOURCE search chain. The DTSAFE member is used to save the Debug Tool session settings when your debug session terminates; these settings are used as your defaults for future debugging sessions. The phase SAMPD5 created by the link-edit is written to this sublibrary also. You need to add this sublibrary to the LIBDEF SOURCE and LIBDEF PHASE job control statements in your CICS startup job control. If you do not want to add this sublibrary to your CICS system, you can:

- Copy the listing member SAMPD5.LIST to a sublibrary that exists in your CICS LIBDEF SOURCE search chain.
- Copy the phase SAMPD5.PHASE to a sublibrary that exists in your CICS LIBDEF PHASE search chain.

5 You include a user run-time options module, CEEUOPT, to define TEST run-time options. Debug Tool runs in the mode defined in the run-time TEST option you supplied. Don't forget to remove the CEEUOPT containing your run-time TEST option when you finish debugging your program. For instructions on how to create the CEEUOPT run-time options module, follow the steps described in page 116 of the *Debug Tool for VSE/ESA User's Guide and Reference*. In this sample, we used:

```
CEEXOPT TEST=(ALL,*,PROMPT,*)
```

This can be considered as an overwrite modifying the system-wide LE/CICS run-time options in order to activate Debug Tool.

6 In order for you to view the program you are debugging, Debug Tool must have access to a file containing the program source statements. For COBOL, Debug Tool must have access to the listing generated by the compiler. Debug Tool supplies the compiler print exit program EQALIST to assist you in storing the compiler listing for your COBOL programs to disk files. EQALIST is invoked by using the COBOL compile-time option.

7 When you compile with the TEST option, the compiler creates the dictionary tables that Debug Tool uses to obtain information about program variables, and inserts program hooks at selected points in your program. Your source is not modified. These points can be at the entrances and exits of blocks, at statement boundaries, and at points in the program where program flow might change between statement boundaries (called path points), such as before and after a CALL statement. Using these hooks, you can set breakpoints to instruct Debug Tool to gain control of your program at selected points during its execution.

The compile-time TEST sub-options control the production of such debugging aids as dictionary tables and program hooks that Debug Tool needs to debug your program. Specifying TEST with no sub-option is equivalent to TEST(ALL,SYM).

Submit this job and make sure it runs without error. (Return code 2 indicates unresolved weak external references during the link-edit step. This return code is normal and does not indicate an error.) Now you need to define your program and transaction to CICS in order to invoke Debug Tool interactively:

- CEDA DEF PROG(sampd5) LANG(COBOL) GROUP(test)
- CEDA DEF TRAN(sam5) PROG(sampd5) GROUP(test)
- CEDA INSTALL GROUP(test)
- CEMT SET PROG(sampd5) NEW

Note: Whenever you modify your source program, you need to submit COM5N to recompile the program and issue *CEMT SET PROG(sampd5) NEW* to get an updated copy of the program.

7.3.2 Batch Debug Using a Command File

You can invoke Debug Tool from a batch job, including the Debug Tool commands in the batch job. You need to compile your program with compile option TEST. Then submit the job to execute your program. You get the debugging result in SYSLST.

The job control COM5B in Figure 15 on page 115 compiles, link-edits, and executes program SAMPD5 that invokes Debug Tool.

```
* $$ JOB JNM=COM5B,DISP=D,CLASS=4,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COM5B BATCH DEBUG USING A COMMANDS FILE
// LIBDEF *,SEARCH=(PRD2.TEMP,PRD2.SCEEBASE,PRD2.PROD)
// SETPARM CATALOG=1
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=PRD2.TEMP
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE SAMPD5B,*
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL       1
 CBL LIB,APOST,NOADV,RENT,BUF(4096),TEST
* $$ SLI ICCF=(SAMPD5),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
*
* TO EXEC YOUR PROGRAM WITH DEBUG TOOL
*
// EXEC SAMPD5B,SIZE=AUTO,PARM='/TEST(,SYSIPT,,SYSIPT)'    2
      *
      * Preference file Input
      *
        SET MSGID ON;              3
/*
      *
      * Command file Input
      *
        LIST AA;                   4
        LIST BB;
        LIST CC;
        LIST DD;
        GO;
        LIST AA;
        LIST BB;
        LIST CC;
        LIST DD;
        SET MSGID OFF;
        QUIT;
/*
/. NOLNK
/&
* $$ EOJ
```

Figure 15. COM5B - Batch Debug Using a Command File

Note:

1 The 'batch compile' process in COBOL only produces one open/close call for the listing file, no matter how many END PROGRAM units there are. Thus, all compiler listings for the batch compile will be written to a file named after the first compile unit name. The use of batch compiles

with the EQALIST exit is not recommended, as a single output file for each compile unit will not be produced.

2 You can use the run-time TEST option to invoke Debug Tool and begin testing your program. The option is passed as an execution time parameter when you invoke your application program, as shown in the following example:

```
// EXEC MYPROG,PARM='prog arg list / TEST(suboptions)'
```

The simplest form of the TEST option is TEST with no sub-option. However, sub-options provide you with more flexibility. There are four sub-options available:

- test_level
- commands_file
- prompt_level
- session_par and preferences_file

In this sample, it tells Debug Tool to read the preference file and the command file from SYSIPT.

3 The preferences file is specified as a sub-option of the TEST run-time option. It is processed by Debug Tool before any primary commands file and is useful for setting up the Debug Tool environment. This file will usually contain Debug Tool commands that you would use during every Debug Tool session. For example, you can use it to set up your preferred layout of the Debug Tool windows and the preferred screen colors for a full-screen session.

If no preferences file is specified in the TEST run-time option, Debug Tool looks for a default preferences file in the sublibrary member userid.DTPREF. Debug Tool searches for this member in the sublibraries specified in the SOURCE search chain.

4 The commands file can be specified to Debug Tool as a sub-option of the TEST run-time option or by the Debug Tool USE command.

- If a commands file is specified as part of the commands_file sub-option of the TEST run-time option, it is termed a primary commands file. The commands in a file specified in this way are executed to end of file.
- If a commands file is specified by the USE command, it is termed a USE file. Commands in a USE file (unless the USE command is included as part of a primary commands file) are executed until a ′non-returning′ command, such as GO, is encountered; any further commands in the USE file are ignored.

The debugging results in SYSLST are as following:

```
      * DEBUG TOOL FOR VSE/ESA VERSION 1 RELEASE 1 MOD 0
      * 02/12/97 10:33:42 AM
      * (C) COPYRIGHT IBM CORP. 1992, 1996
        SET MSGID ON ;
        LIST AA ;
      * EQA1576E THE ENVIRONMENT IS NOT YET FULLY INITIALIZED.
        LIST BB ;
      * EQA1576E THE ENVIRONMENT IS NOT YET FULLY INITIALIZED.
        LIST CC ;
      * EQA1576E THE ENVIRONMENT IS NOT YET FULLY INITIALIZED.
        LIST DD ;
      * EQA1576E THE ENVIRONMENT IS NOT YET FULLY INITIALIZED.
        GO ;
1
2
      * EQA1306I YOU WERE PROMPTED BECAUSE THE CEE067  CONDITION WAS RAISED IN YOUR PROGRAM.
      * EQA1309I CEE0G7 IS A SEVERITY OR CLASS 1 CONDITION.
      * EQA1334I THE OPERATING SYSTEM HAS GENERATED THE FOLLOWING MESSAGE:
      * EQA1335I    CEE0199W THE TERMINATION OF A THREAD WAS SIGNALED DUE TO A STOP STATEMENT.
      * EQA1238I THE CURRENT LOCATION IS SAMPD5 :> 17.1.
        LIST AA ;
      * EQA1141I AA = 04
        LIST BB ;
      * EQA1141I BB = 04
        LIST CC ;
      * EQA1141I CC = 05
        LIST DD ;
      * EQA1141I DD = 03
        SET MSGID OFF ;
      *  QUIT ;
```

Figure 16. COM5B - Debugging Result in SYSLST

7.3.3 Batch Execution with Interactive Debug

You can get a full-screen debugging session by running a batch job, without CICS. Firstly, you still need to compile your program with compile option TEST. Then execute your program with run-time option /TEST. You will get a full-screen debugging session on the terminal which you specified in the batch job.

The following job control in Figure 17 on page 118 compiles, link-edits and executes program SAMPD5 that invokes Debug Tool interactively.

```
* $$ JOB JNM=COM5BI,DISP=D,CLASS=4,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COM5BI BATCH EXECUTION WITH INTERACTIVE DEBUG
// LIBDEF *,SEARCH=(PRD2.TEMP,PRD2.SCEEBASE,PRD2.PROD)
// SETPARM CATALOG=1
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=PRD2.TEMP
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE SAMPD5BI,*
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL,PARM='EXIT(PRTEXIT(EQALIST))'
 CBL LIB,APOST,NOADV,RENT,BUF(4096),TEST
* $$ SLI ICCF=(SAMPD5),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
*
* RUN SAMPD5BI INVOKING DEBUG TOOL ON TERMINAL D08101
*
// EXEC SAMPD5BI,SIZE=AUTO,PARM='/TEST(,SYSIPT,,MFI%D08101:*)' 1
      *
      * Command file Input              2
      *
        SET LOG ON FILE prd2.temp(sampd5.LOG);      3
        MONITOR LIST DD;
        LIST AA;
/*
/. NOLNK
/&
* $$ EOJ
```

Figure 17. COM5BI - Batch Execution with Interactive Debug

Notes:

1 The interactive debug session runs on the terminal with the VTAM LU name 'D08101'. This terminal must not be in session with another application when Debug Tool attempts to acquire it.

2 You can include your command file in your job, while you get an interactive debug session. The command file in your batch job will be executed first. Then you can get a full-screen debugging session and you can enter Debug Tool commands also.

3 Debug Tool can record commands and their generated output in a session log file. By default, Debug Tool writes the session log to SYSLST. The default under CICS is no log file. The SET LOG command can be used to change the log file from the default. Issuing the command SET LOG OFF can be used to suppress output to the log file. The session log can be used as a commands file for a later Debug Tool session without having to edit out the results from a previous run.

In Figure 17, you indicate to Debug Tool that a file is a sublibrary (prd2.temp) member by enclosing the member name and type (sampd5.log) in parentheses and connecting the name and type with a period (.) character.

7.3.4 Batch Debug using CEETEST

Debug Tool can also be invoked directly from within your program using the LE/VSE callable service CEETEST and get the debugging result in SYSLST.

Using CEETEST, you can invoke Debug Tool from within your program and send it a string of commands. If no command string is specified, or the command string is insufficient, Debug Tool prompts you for commands from your terminal or reads them from the commands file. In addition, you have the option of receiving a feedback code that tells you whether the invocation procedure was successful.

The syntax for CEETEST is:

```
CALL "CEETEST" USING string_of_commands,fc
```

You can define 'string_of_commands' and 'fc' (feedback code) in the WORKING-STORAGE SECTION of your COBOL program.

The following is a sample program invoking Debug Tool with CEETEST. A command string is passed to Debug Tool at its invocation and the feedback code is returned.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPD55.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  FC        PIC X(12) VALUE ZEROES.          1
77  DEBUGTOOL PIC X(7)  VALUE 'CEETEST'.
.
.  (your source code)
.
01  PARMS.
    02 EE        PIC 99    VALUE 14.
    02 FF        PIC X(11) VALUE 'LIST CALLS;'.      2
PROCEDURE DIVISION.
    CALL DEBUGTOOL USING PARMS FC.       3
    .
    .  (your source code)
    .
    STOP RUN.
```

Figure 18. SAMPD55 - Sample Program Invoking Debug Tool with CEETEST

Notes:

1 FC is a 12-byte feedback code that indicates the result of this service. LE/VSE provides a callable service called CEEDCOD to help you decode the fields in the feedback code. Requesting the return of the feedback code is recommended. See *LE/VSE Programming Reference* for details.

2 A command string is passed to Debug Tool at its invocation and the feedback code is returned.

3 The statement that invokes Debug Tool.

The job control in Figure 19 compiles, link-edits and runs program SAMPD55 in Figure 18 on page 119 with Debug Tool.

```
* $$ JOB JNM=COM55,DISP=D,CLASS=4,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COM55 COMPILE PROGRAM SAMPD55
// LIBDEF *,SEARCH=(PRD2.TEMP,PRD2.SCEEBASE,PRD2.PROD)
// SETPARM CATALOG=1
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS,SXREF,SYM,LIST,NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE,CATALOG=PRD2.TEMP
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE SAMPD55,*
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL
 CBL LIB,APOST,NOADV,RENT,BUF(4096),TEST          4
* $$ SLI ICCF=(SAMPD55),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKEDT,SIZE=256K
// EXEC SAMPD55,SIZE=AUTO          5
/. NOLNK
/&
* $$ EOJ
```

Figure 19. COM55 - Sample Job Invoking Debug Tool Using CEETEST

Notes:

4 If you don't want to compile your program with hooks, you can use CEETEST calls to invoke Debug Tool at strategic points in your program. If you decide to use this method, you still need to compile your program with the TEST option so that symbolic information is created for Debug Tool.

5 When SAMPD55 is executed, Debug Tool is invoked. You will get the debugging result in SYSLST.

7.4 How to Debug Your Program in Full-Screen Mode

This section describes the most common feature of Debug Tool. It helps you start using this tool to debug your programs quickly.

Before using Debug Tool you must compile at least one part of your program with the compile-time TEST option. This inserts hooks, which are assembly instructions that you can see in an assembly listing. The execution of these

hooks enables Debug Tool to gain control during program execution. Please see previous samples for the details of the compile-time TEST option.

7.4.1 Using a COBOL Program to Demonstrate a Debug Tool Session

This section uses a simple COBOL program SAMPSRT to demonstrate how to use Debug Tool to debug your program when there is a problem during the execution. You can find the complete description of Debug Tool commands in the *Debug Tool for VSE/ESA User's Guide and Reference*.

When a program does not work as you expect, you may determine and solve the problem with the following steps:

1. If an abend message occurred, find the statement that is issuing the message.
2. If output data is not as expected, find the first 'bad' record of the output and the corresponding input record. Then find the statements which process the input record.
3. Review the source code lines which are suspected.
4. Insert some DISPLAY statements to monitor the status of variables.
5. Recompile the program.
6. Execute and test the program.
7. Analyze the result.
8. Retry Step 4 to Step 7 until the cause of the problem is found.
9. Modify the source program.
10. Test the program again till you get the expected result.
11. Delete DISPLAY statements for debugging from the program.
12. Source program update complete.

Using Debug Tool, you can interactively perform Step 4 to Step 11 without modifying the source program.

The following program SAMPSRT is supposed to sort characters alphabetically, and by intention contains a flaw.

```
01     IDENTIFICATION DIVISION.
02     PROGRAM-ID. SAMPSRT.
03     ENVIRONMENT DIVISION.
04     DATA DIVISION.
05     WORKING-STORAGE SECTION.
06     01  STRX.
07         02  STR1   PIC X(40).
08         02  STRD   REDEFINES STR1 OCCURS 40 PIC X.
09     77  STRLEN PIC 99 VALUE ZEROS.
10     77  COL1   PIC 99 VALUE ZEROS.
11     77  SORTMORE PIC 9 VALUE ZERO.
12     77  WW  PIC X.
13     PROCEDURE DIVISION.
14         MOVE 'RED BOOK' TO STR1.
15         MOVE 8 TO STRLEN.
16         DISPLAY 'SORT START  =' STR1.
17         MOVE 1 TO SORTMORE.
18         PERFORM UNTIL SORTMORE = 0
19            MOVE 0 TO SORTMORE
20            PERFORM VARYING COL1 FROM    1 BY 1 UNTIL COL1 > STRLEN
21               IF STRD ( COL1 ) > STRD ( COL1 + 1 )
22                  MOVE STRD ( COL1 + 1 ) TO WW
23                  MOVE STRD ( COL1 ) TO STRD ( COL1 + 1 )
24                  MOVE WW TO STRD ( COL1 )
25                  MOVE 1 TO SORTMORE
26               END-IF
27            END-PERFORM
28         END-PERFORM.
29         DISPLAY 'SORT RESULT =' STR1.
30         STOP RUN.
```

Figure 20. SAMPSRT - COBOL Source Program to Demonstrate Debug Tool

When you execute this program you will get the follow result:

```
SORT START  =RED BOOK
SORT RESULT =  BDEKOOR
```

Note: There are two blanks at the beginning of the result. There should be only one blank. That is the problem.

Before you start to use the interactive debug session to debug this sample program, you need to compile the program with the TEST option and define the program and transaction to CICS. For details on how to invoke Debug Tool interactively with CICS, refer to 7.3.1, “Interactive Debug with CICS” on page 110.

7.4.1.1 Simple Execution to the End of the Program

You can execute this program as normal.

- In CICS mode, enter transaction SSRT to start the debug session.
- When you see the full debugging screen, press PF9(GO). PF9(GO) runs your program until a breakpoint is reached, the program ends, or a condition is raised.
- The program is executed and stops at the *STOP RUN* statement.
- Enter *LIST STR1* in the command line and you will get the following result in the Log window:

```
STR1 = '  BDEKOOR                                    '
```

If you use the *LIST* command to list the contents of an uninitialized variable, or a variable that contains invalid data, Debug Tool will display *INVALID DATA*.

- Enter *DESCRIBE ATTRIBUTES STR1* and you will see:

```
ATTRIBUTES FOR STR1
  ITS LENGTH IS 40
  ITS ADDRESS IS 01F88408
  02 SAMPSRT:>STR1 X(40) DISP
```

You can use this action as a simple browser for group items and data hierarchies.

- Enter *LIST STORAGE (STR1,16)* and you will get 16 bytes of storage for STR1:

```
01F88408 4040C2C4 C5D2D6D6D9404040 40404040 *  BDEKOOR        *
```

- Press PF3 to quit and enter *Y* to confirm.

7.4.1.2 Step Through the Program and Monitor a Variable's Value

To continuously display or monitor a variables value, you can issue most LIST commands preceded by the word MONITOR. For example, enter *MONITOR LIST STR1*. The output for this command will be continuously displayed in the Monitor window. The MONITOR command makes it easy to watch values while stepping through your program.

Pressing PF2(STEP) runs your program, halting on the next hook encountered. If you compiled your program with *TEST(ALL,SYM)*, PF2(STEP) performs one statement.

Now you can keep pressing PF2(STEP) to execute your program step by step. At the same time, you can monitor the value of STR1 from the Monitor window.

Press PF3 to quit and enter *Y* to confirm.

7.4.1.3 Setting Breakpoints to Halt the Execution

The AT command defines a breakpoint or a set of breakpoints. By defining breakpoints, you can temporarily suspend program execution and use Debug Tool to perform other tasks. By specifying an AT-condition, you instruct Debug Tool when to gain control. You can also specify in the AT command what action Debug Tool should take when the AT-condition occurs.

A breakpoint for the specified AT-condition remains established until either another AT command establishes a new action for the same AT-condition or a CLEAR command removes the established breakpoint.

In this example, there are two PERFORM statements. You want to check at every end of the outer loop. You can do the following:

- In CICS mode enter transaction SSRT to start the debugging session.
- For example, you want to halt at line 27 (END-PERFORM). However, *AT 27 LIST STR1* command is rejected because END-PERFORM statement is not suitable for a breakpoint.
- Enter *AT 19 LIST STR1* and press PF9(GO). You will get the results every time at the beginning of the outer loop. Press PF9(GO) three times, you will see:

```
GO;
STR1 = ' RED BOOK                                    '
GO;
STR1 = ' ED BOOK R                                   '
GO;
STR1 = ' D BEOK OR                                   '
```

According to the result, the second display of STR1 has two blanks which is not the expected result. To confirm that, you need to quit and enter the session again.

- Press PF3 and enter *Y* to confirm.
- In CICS mode, enter transaction SSRT.
- Enter *AT 19 LIST STR1*.
- Enter *AT 24 IF WW=' ' LIST TITLED(COL1,WW);END-IF;*
- Press PF9(GO) nine times and you will see the following:

```
GO;
STR1 = ' RED BOOK                                    '
GO;
GO;
GO;
COL1 = 03
WW = " "
GO;
GO;
GO;
GO;
GO;
COL1 = 08
WW = " "
```

There is a problem in the first loop, when COL1 is 8. At this point, it is clear that this sort logic uses an out of range character. The condition of *UNTIL COL1 > STRLEN* is not correct.

7.4.1.4 Modifying the Value of a Variable

Now that you know where the problem is, you can modify the value of STRLEN interactively to fix the problem.

- Press PF3 and enter *Y* to confirm it.
- In CICS mode, enter transaction SSRT.
- Enter *AT 17 COMPUTE STRLEN = STRLEN - 1*.

  (You may also enter *AT 17 MOVE 7 TO STRLEN*.)
- Press PF9(GO) twice.
- Program execution finished. Enter *LIST (STR1,STRLEN)* and you will see:

```
STR1 = ' BDEKOOR          '
STRLEN = 07
```

This is the expected result. Now you can start to modify your source program using one of the following:

- Add *COMPUTE STRLEN = STRLEN - 1.* at line 17, or
- Change the UNTIL statement to *COL1 > STRLEN - 1*, or

- Change the UNTIL statement to *COL1 >= STRLEN*.

7.4.1.5 Restrictions on COBOL-like Commands

To make testing COBOL programs easier, Debug Tool allows you to write debugging commands in a manner resembling COBOL statements. It does this by providing an interpretive subset of COBOL language statements. This interpretive subset is a list of commands recognized by Debug Tool that either closely resembles or duplicates the syntax and action of the appropriate COBOL statements. This not only allows you to work with familiar commands, but also simplifies the insertion into your source code of program patches developed while in your Debug Tool session.

However, some restrictions apply to the use of the COBOL commands COMPUTE, MOVE, and SET; the conditional execution command, IF; the multiway switch, EVALUATE; the iterative looping command, PERFORM; and the subroutine call, CALL.

For details on the restrictions, please see Chapter 10 ′Using Debug Tool with COBOL Programs′ in the *Debug Tool for VSE/ESA User ′s Guide and Reference*.

For a complete list of Debug Tool commands, see Chapter 13 ′Using Debug Tool Commands′ in the *Debug Tool for VSE/ESA User ′s Guide and Reference*.

7.5 Limitations of Debug Tool for VSE/ESA

Debug Tool for VSE fully supports the process of debugging multilanguage applications. There are no limitations when all the programs in the application were compiled with an LE-conforming compiler.

However, if the application contains programs compiled with a non-LE-conforming compiler, Debug Tool will not support source-level debugging of those programs.

The application will execute unhindered while control is within a program written in an unsupported language. Debug Tool will regain control and allow debug commands to be issued once execution returns to a program unit written in a supported language.

Debug Tool for VSE does not support the cooperative mode of operation where the program being debugged is executing on a VSE/ESA host system and Debug Tool is executing on a programmable workstation (PWS).

Debug Tool for VSE cannot execute without an active LE/VSE Release 4 environment.

Chapter 8. Performance Considerations

Although the LE/VSE and language publications provide information about performance, here are some very specific recommendations to help you overcome potential problems.

Please realize that the performance you may experience depends upon the release of COBOL to which you are comparing, the amount of tuning you have done, and the particular characteristics of the programs themselves.

In this chapter we will discuss the performance considerations and the specific compiler options that affect the performance.

8.1 Transferring Control to Another Program

In the Procedure Division, a program can call another program, and the called program may itself call yet another program. When the called program processing is completed, the program can either transfer control back to the calling program or end the run unit. This kind of technique helps to limit the use of GETVIS area, and thus improves the performance.

A called program must not directly or indirectly execute its caller (such as program X calling program Y; program Y calling program Z; and program Z then calling program X). This is called a recursive call. If you attempt to execute a recursive call to a COBOL program, the run unit will end abnormally (abend).

Very often you will want your COBOL/VSE programs to communicate with other COBOL and non-COBOL programs. So to transfer control from one COBOL/VSE program to another COBOL/VSE program, you can use one of these methods:

- Calls to nested program
- Static calls
- Dynamic calls

8.1.1 Nested Programs

Nested programs, first introduced in VS COBOL II, means that a COBOL source program can contain another COBOL source program, and a contained COBOL source program can in turn contain other COBOL source programs.

Using nested programs instead of separately compiled programs will improve the performance. Nested programs are resolved during compilation, thereby removing the call altogether which substantially improves performance. You can use nested copy statements to code nested programs.

8.1.2 Static and Dynamic Calls

In addition to making calls between COBOL/VSE programs, you can also make static and dynamic calls between COBOL/VSE and VS COBOL II programs and, in a non-CICS environment, between COBOL/VSE and DOS/VS COBOL programs. In a CICS environment you must use EXEC CICS LINK to transfer control between COBOL/VSE and DOS/VS COBOL programs.

Dynamic calls from VS COBOL II and COBOL/VSE to DOS/VS COBOL programs are supported with the following restrictions:

- The DOS/VS COBOL program must be relinked with the LE/VSE COBOL Run-Time compatibility library.
- DOS/VS COBOL programs that are segmented cannot be loaded into GETVIS storage. These are programs that are compiled with LANGLVL(1) and specify the SEGMENT-LIMIT clause.
- The debug options STATE, FLOW, COUNT and SYMDMP are disabled.

A static call is used to invoke a separately compiled program that is link-edited into the same phase as the calling program. A dynamic call is used to invoke a separately compiled program that has been link-edited into a separate phase from the calling program. In this case, the subprogram is loaded into storage the first time it is called.

Note: Although they may use more storage than dynamic calls, static calls are executed more quickly.

8.1.3 Dynamic CALL instead of EXEC CICS LINK

Within a CICS environment use a dynamic CALL rather than an EXEC CICS LINK. DOS/VS COBOL had a restriction that any called subroutine could not issue EXEC CICS commands. This resulted in widespread use of EXEC CICS LINK to subroutines. However, this restriction does not exist with COBOL/VSE and LE/VSE and dynamic calls are recommended.

Why? EXEC CICS LINK causes LE/VSE to create a nested enclave which will have its own resources such as storage. The termination of this enclave with an EXEC CICS RETURN will free these resources, but along the way there will be more LE/VSE code executed and a corresponding degradation in performance. CICS/VSE 2.3 uses VSE/ESA GETVIS/FREEVIS services for CICS storage obtained or released above 16MB, so this will incur further overhead.

Note: There is the APAR PQ03907 for CICS/VSE that addresses this overhead. The PTFs UQ06448 and UQ06449 solve this problem.

8.2 COBOL/VSE Compiler Options that Affect Performance

The default options that were set up when your compiler was installed will be in effect for your program unless you replace them with other options, and they can affect performance at run time.

The tuning methods and performance information discussed here are intended to help you select from various COBOL/VSE options for compiling your program. But before changing defaults of the options make sure that the installed options are not required.

A brief description of each item is followed by performance advantages and disadvantages.

A detailed description of these options can be found in the manual *COBOL/VSE Programming Guide*.

8.2.1 DYNAM

The DYNAM compiler option dynamically loads subprograms invoked through the CALL statement at run time.

Performance advantages

Using DYNAM means easier subprogram maintenance since the application will not have to be relink-edited if the subprogram is changed.

When using the DYNAM option, you can free virtual storage that is no longer needed by issuing the CANCEL statement.

Performance disadvantages

You pay a slight performance penalty using DYNAM since the call must go through an LE/VSE routine.

8.2.2 FASTSRT

The FASTSRT compiler option specifies that the DFSORT/VSE product (or equivalent) will handle the I/O.

Performance advantages

FASTSRT eliminates the overhead of returning to COBOL/VSE after each record is processed.

Performance disadvantages

No performance disadvantages.

8.2.3 NUMPROC(PFD),(NOPFD),(MIG)

Use this compiler option for sign processing when coding numeric comparisons.

Performance advantages

NUMPROC(PFD) generates significantly more efficient code for numeric comparisons.

Performance disadvantages

For most references to COMP-3 and DISPLAY numeric data items, using NUMPROC(MIG) and NUMPROC(NOPFD) causes extra code to be generated because of sign ″fix up″ processing. This extra code may also inhibit some other types of optimizations.

8.2.4 OPTIMIZE

Use the OPTIMIZE compiler option to ensure your code is optimized for better performance.

Performance advantages

Generally results in more efficient run-time code.

Performance disadvantages

OPTIMIZE requires more processing time for compiles than NOOPTIMIZE.

8.2.5 RENT

Use the RENT compiler option to generate a reentrant program.

Performance advantages

Using RENT enables the program to be placed in the Shared Virtual Area (SVA) for running above the 16-megabyte line on an extended addressing system.

Performance disadvantages

Using RENT generates additional code to ensure that the program is reentrant.

8.2.6 SSRANGE

The SSRANGE compiler option verifies that all subscripts, indexes, and reference modification expressions are within proper bounds.

Performance advantages

No performance advantages.

Performance disadvantages

SSRANGE generates additional code for verifying subscripts.

8.2.7 TEST

The TEST compiler option with any hook-location sub-option other than NONE (that is, ALL,STMT,PATH,BLOCK) produces object code that can take full advantage of and be run under Debug Tool/VSE.

Performance advantages

No performance advantages.

Performance disadvantages

Since the TEST compiler option with any hook-location sub-option other than NONE generates additional code, it can cause significant performance degradation when used in a production environment.

8.2.8 TRUNC(STD),(OPT),(BIN)

The compiler option creates code that will shorten the receiving fields of arithmetic operations.

Performance advantages

TRUNC(OPT) does not generate extra code and generally improves performance.

Performance disadvantages

Both TRUNC(BIN) and TRUNC(STD) generate extra code whenever a BINARY data item is changed. TRUNC(BIN) is usually the slowest of these options.

8.3 CICS Compiler Options Considerations

If you compile CICS code, the following compiler options are **required**:

RES

RENT

NODYNAM

LIB (if the program has a COPY on BASIS statement in it)

The CICS translator always inserts a line into COBOL/VSE programs that specifies:

CBL RES,RENT,NODYNAM,LIB

You cannot replace these compiler options with options passed when you invoke the compiler (for example, with PARM=...).

The input file for the compiler is the file you received as a result of the translation, which is SYSPCH by default.

Although the CICS translator automatically issues the above CBL statement, the COBOL/VSE compiler will always issue a warning message, because the RES option is not valid (COBOL/VSE always runs RESIDENT), whereas VS COBOL II has the RES/NORES compiler option. The warning message can be ignored.

Recommended Options:

TRUNC(BIN) is recommended only for those applications that use binary data items that may contain more than nine digits in a fullword or more than four digits in a halfword.

WORD(CICS) is recommended if you want those COBOL language elements not supported under CICS to be flagged at compile time. You use WORD(xxxx) to specify that an alternative reserved word table is to be used during compilation. COBOL/VSE provides an alternate reserved word table (IGYCCICS) specifically for CICS applications. It is set up so that COBOL words not supported under CICS are flagged with an error message.

In addition to the default COBOL restricted word table, the CICS reserved word table restricts the following COBOL words:

Table 9. CICS Reserved Word Table

ACCEPT	FILE-CONTROL	RERUN
CLOSE	INPUT-OUTPUT	REWRITE
DELETE	I-O-CONTROL	SD
DISPLAY	MERGE	SORT
FD	OPEN	START
FILE	READ	WRITE

8.4 Additional Performance Considerations

8.4.1 ALL31(ON) - LE/VSE Run-time Option

Running all applications with run-time option ALL31(ON) provides a performance improvement over ALL31(OFF). This is because there is code within LE/VSE to test and switch addressing modes if required. This has further implications. Running with ALL31(OFF) means that HEAP and STACK storage must be allocated below-the-line.
Under CICS with ALL31(OFF) the Thread control blocks are not pre-allocated, so the Run-Unit Initialization routine must issue a CICS GETMAIN for below-the-line storage for these control blocks. This therefore requires the Run-Unit Termination routine to also issue a CICS FREEMAIN to release the storage for these control blocks.

The supplied batch default is ALL31(OFF), the supplied CICS default is ALL31(ON).

8.4.2 Link-Edit Parameters

To gain maximum benefit from storage above the 16MB line, always attempt to link-edit your program AMODE 31 and RMODE ANY, either by default, or if appropriate by using AMODE and RMODE link-edit parameters.
This is especially true for programs link-edited AMODE ANY because VSE/ESA will always invoke these programs in AMODE 24. In this case you must explicitly override the link-edit AMODE by means of link-edit parameters.

Note: For COBOL/VSE programs the RMODE and RENT compiler options can affect the resulting AMODE and RMODE of the link-edited program.

8.4.3 Considerations for VSAM Performance

APAR PN84947/UN92489 VSAM SHOWCB - (LE/VSE COBOL only)

Improve your VSAM performance, particularly when reading a large number of records, by ensuring that this fix is applied, if you are still running LE/VSE 1.1. This PTF is already incorporated in the LE/VSE 1.4 base code.
Although there will be an improvement on VSE/ESA 1.4 there will be even better gains on a VSE/ESA 2.1 system with Turbo Dispatcher.

8.5 DFSORT/VSE STXIT/NOSTXIT

Were you aware that DFSORT/VSE is provided as the only IBM sort product with VSE/ESA 2.1 and above?
Maybe not! This is because SORT control statements are mostly compatible amongst the different products, and you may be using DFSORT/VSE without being aware of it. Further, you may not be aware that DFSORT/VSE has introduced new error handling features which will have a direct effect on SORT performance. The new STXIT option introduced by DFSORT/VSE (and supplied as the default), will invoke additional SVC overhead for every DFSORT/VSE user exit.

Why may this be a problem?
All language products use SORT user exits to handle input and output processing (E15 and E35 exits). These exits are usually called for each record passed to SORT(E15) and passed back from SORT(E35). With the STXIT option, DFSORT/VSE issues the STXIT SVC on return from *every* SORT user exit. So for languages, there is an additional overhead for each record passed to sort and each record returned. In addition, LE/VSE also provides error condition handling via the TRAP(ON) run-time option, so it also must issue the STXIT SVC to re-establish its error handling on return from DFSORT/VSE. But this will only occur if the DFSORT/VSE STXIT option is set. So use the DFSORT/VSE NOSTXIT option. Either change the DFSORT/VSE supplied defaults (STXIT=YES|NO) or add NOSTXIT to a DFSORT/VSE OPTION statement and pass it to the application program.

Notes:

1. A detailed description of these performance issues can be found in the Information APAR II09745.
2. For COBOL/VSE, the FASTSRT compiler option is available to bypass DFSORT/VSE exit processing in some circumstances. Read the appropriate section in the *IBM COBOL for VSE/ESA Programming Guide* for details.

3. If you use a non-IBM SORT product, the same considerations may apply. Change the equivalent SORT STXIT option if necessary.
4. The same considerations will apply to DOS/VS COBOL using DFSORT/VSE.

8.6 Service Issues with Vendor Products

Most Vendor tape and disk managers such as CA-DYNAM, CA-DYNAM/T and CA-EPIC, interface with LE/VSE OPEN processing. Typically, these products intercept the OPEN to provide the required functionality.
Because of the increased number of pre-OPEN checks introduced by VS COBOL II (for ANSI 85 conformance), changes to VS COBOL II OPEN processing were required to enable the vendor code to function correctly. For VS COBOL II the vendor interface was provided by a number of VS COBOL II APARs/PTFs.
The solution provided was considered to be a temporary measure because of the technique used (communicating via the Label Parameter List (LPL) during a call to the LABEL macro processor ($IJBSLA)).
With LE/VSE, the VSE/ESA PRODEXIT facility was introduced, providing a standard interface without the need to change VSE/ESA control blocks. Ask your Vendor, if the product they supply, requires the PRODEXIT interface.

What kinds of problems can be expected, if the vendor interface has not been enabled?
Typically problems will be seen during OPEN processing of SAM or VSAM managed SAM files.

How do I obtain diagnostic information for file OPEN problems?
An SDAID Branch trace is usually the best way to obtain information to help debug a failing file OPEN.
You will be directed by your IBM support representative on the modules that must be traced. This will vary according to the file type (for example VSAM or SAM) of the file being opened.
The following are examples of SDAID control statements:

- COBOL SAM file

  ```
  TRACE BR AR=ALL PHASE=IGZEQOC OUTP=GREG
  ```

- COBOL VSAM file

  ```
  TRACE BR AR=ALL PHASE=IGZEVOC OUTP=GREG
  TRACE BR AR=ALL PHASE=IGZEVOP OUTP=GREG
  ```

Appendix A. Sample Case of Language Conversion

This chapter shows an example to give you practical information about COBOL migration to COBOL/VSE.
(REMARKS: This example is not a ″good program″. Please keep in mind the average program cannot contain so many incompatibilities.)

A.1 Sample Conversion of DOS/VS COBOL LANGLVL(2)

There are six problems in total:

- **1** ALPHABET clause changes -SPECIAL NAMES paragraph
- **2** reserved keyword list change
- **3** MOVE statements and comparison change
- **4** UNSTRING statements -subscript evaluation changes
- **5** ALPHABETIC class change
- **6** PERFORM statement -changes in the VARYING/AFTER options

Since the programs which can be compiled by VS COBOL II with CMPR2 are very similar to DOS/VS COBOL LANGLVL(2), these problems are common also in VS COBOL II CMPR2.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. SAMPD2.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES.
     ALPHA-NAME IS STANDARD-1. 1
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 77  AA PIC X(40).
 77  BB PIC X(40).
 77  N1 PIC 9(3).
 77  N2 PIC 9(5).
 01  FLAG-AREA.
     03 REPLACE PIC 9. 2
     03 NO-DATA PIC 9.
 01  P1 PIC 999PP.  3
 01  CC-TAB.
     03 CCX  PIC X(120) VALUE SPACE.
     03 CC REDEFINES CCX OCCURS 3 PIC X(40).
 01  I  PIC 999.
*
 PROCEDURE DIVISION.
*
** UNSTRING
     MOVE 'XX YYY ZZZZ WW'     TO AA.
     MOVE 1 TO I.
     UNSTRING AA DELIMITED BY ALL SPACES 4
        INTO  BB COUNT IN I
              CC( I ) .
     DISPLAY 'I=' 1 ' VALUE=' CC ( 1 )
     DISPLAY 'I=' 2 ' VALUE=' CC ( 2 )
     DISPLAY 'I=' 3 ' VALUE=' CC ( 3 )
*
     MOVE 12300 TO P1.
     IF P1 = '12300' DISPLAY 'P1=12300'.  3
     IF P1 = '123'   DISPLAY 'P1=123'.
     IF P1 = '123  ' DISPLAY 'P1=123__'.
*
     MOVE 'Abcde' TO AA.
     IF AA IS ALPHABETIC DISPLAY 'Ax IS ALPHA'.  5
     MOVE 'ABCDE' TO AA.
     IF AA IS ALPHABETIC DISPLAY 'AX IS ALPHA'.  5
*
     PERFORM DISP1 VARYING N1 FROM 1 BY 1 UNTIL N1 > 3 6
          AFTER N2 FROM N1 BY 1 UNTIL N2 > 5.
     STOP RUN.
*
*
 DISP1.
     DISPLAY 'N1,N2= ' N1 ' ' N2.
```

Figure 21. SAMPD2 - Sample Source of DOS/VS COBOL LANGLVL(2)

This program can be compiled and executed in a DOS/VS COBOL LANGLVL(2) environment. The result is:

```
I=1 VALUE=          4
I=2 VALUE=YYY
I=3 VALUE=
P1=12300         3
AX IS ALPHA      5
N1,N2= 001 00001  6
N1,N2= 001 00002
N1,N2= 001 00003
N1,N2= 001 00004
N1,N2= 001 00005
N1,N2= 002 00001
N1,N2= 002 00002
N1,N2= 002 00003
N1,N2= 002 00004
N1,N2= 002 00005
N1,N2= 003 00002
N1,N2= 003 00003
N1,N2= 003 00004
N1,N2= 003 00005
```

Figure 22. Result of SAMPD2 in Old Environment

When this source is compiled in a COBOL/VSE environment, there are some compile errors. After the correction of compile errors, some incompatibilities remain.

- **1** ALPHABET clause changes -SPECIAL NAMES paragraph

  IGYDS1325-E The ALPHABET clause did not start with the ″ALPHABET″ keyword. The clause was accepted. (ALPHABET keyword must be added.)

- **2** reserved keyword change

  SYNTAX ERROR (REPLACE is a new keyword. You must rename this.)

A.1.1 OUTPUT of CCCA for the Sample Source

If you use CCCA, most of the syntax conversion was made automatically. In this case CCCA converted the source as follows:

```
 MSGID  RC MESSAGE TEXT

ABJ6170 00 ALPHABET WORD IS ADDED 1
ABJ6268 00 DATA NAME IS AN ANSI 85 RESERVED WORD 2
ABJ6171 00 ALPHABETIC CHANGED TO ALPHABETIC  5
ABJ6026 04 SCALED VARIABLE FOUND 68 STANDARD INTERPRETATION
           *MANUAL UPDATE MAY BE REQUIRED    3
ABJ6025 08 UNSTRING..DEL. BY ALL FOUND 68 STANDARD INTERPRETATION
           *MANUAL UPDATE REQUIRED    4
ABJ6253 08 RULES FOR AUGMENTING VARIABLES HAVE CHANGED. IF DEPENDENCIES
           BETWEEN VARIABLES EXIST, THEN
           *MANUAL UPDATE MAY BE REQUIRED    6
```

Figure 23. CCCA Messages for Conversion of SAMPD2

- **1** **2** **5** were converted.
- **3** **4** **6** were flagged.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. SAMPD2.
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 01/31/97 12:27:35.
000060 ENVIRONMENT DIVISION.
000070 CONFIGURATION SECTION.
000080 SPECIAL-NAMES.
000090     ALPHABET ALPHA-NAME IS STANDARD-1. 1
000100 DATA DIVISION.
000110 WORKING-STORAGE SECTION.
000120 01  LCP-ABND-CODE                PIC S999 COMP VALUE +519.
000130 77  AA PIC X(40).
000140 77  BB PIC X(40).
000150 77  N1 PIC 9(3).
000160 77  N2 PIC 9(5).
000170 01  FLAG-AREA.
000180     03 REPLACE-74 PIC 9. 2
000190     03 NO-DATA PIC 9.
000200 01  P1 PIC 999PP. 3
000210 01  CC-TAB.
000220     03 CCX  PIC X(120) VALUE SPACE.
000230     03 CC REDEFINES CCX OCCURS 3 PIC X(40).
000240 01  I  PIC 999.
000250*
000260 PROCEDURE DIVISION.
000270*
000280** UNSTRING
000290     MOVE 'XX YYY ZZZZ WW'     TO AA.
000300     MOVE 1 TO I.
000310     UNSTRING AA DELIMITED BY ALL SPACES 4
000320        INTO  BB COUNT IN I
000330              CC( I ) .
000340     DISPLAY 'I=' 1 ' VALUE=' CC ( 1 )
000350     DISPLAY 'I=' 2 ' VALUE=' CC ( 2 )
000360     DISPLAY 'I=' 3 ' VALUE=' CC ( 3 )
000370*
000380     MOVE 12300 TO P1.
000390     IF P1 = '12300' DISPLAY 'P1=12300'. 3
000400     IF P1 = '123'   DISPLAY 'P1=123'.
000410     IF P1 = '123  ' DISPLAY 'P1=123__'.
000420*
000430     MOVE 'Abcde' TO AA.
000440     IF AA IS ALPHABETIC-UPPER DISPLAY 'Ax IS ALPHA'. 5
000450     MOVE 'ABCDE' TO AA.
000460     IF AA IS ALPHABETIC-UPPER DISPLAY 'AX IS ALPHA'. 5
000470*
000480     PERFORM DISP1 VARYING N1 FROM 1 BY 1 UNTIL N1 > 3 6
000490          AFTER N2 FROM N1 BY 1 UNTIL N2 > 5.
000500     STOP RUN.
000510*
000520*
000530 DISP1.
000540     DISPLAY 'N1,N2= ' N1 ' ' N2.
000550 LCP-LCP-LCP-EXIT-PROGRAM SECTION.
000560     CALL 'ILBOABNO' USING LCP-ABND-CODE.
000570 END PROGRAM SAMPD2.
```

Figure 24. CCCA Output of SAMPD2

The compile errors were solved automatically in this case. However, the result of execution is different as follows: (In addition, if 5 was not converted by CCCA, there will be two lines for the result: Ax is ALPHA AX is ALPHA.)
To get the exact same result, we need some manual modifications.

```
I=1 VALUE=YYY   4
I=2 VALUE=
I=3 VALUE=
P1=123       3
P1=123__
AX IS ALPHA   5
N1,N2= 001 00001  6
N1,N2= 001 00002
N1,N2= 001 00003
N1,N2= 001 00004
N1,N2= 001 00005
N1,N2= 002 00002
N1,N2= 002 00003
N1,N2= 002 00004
N1,N2= 002 00005
N1,N2= 003 00003
N1,N2= 003 00004
N1,N2= 003 00005
```

Figure 25. Result of SAMPD2 in New Environment

The difference of the result is:

- 4 YYY is assigned to cc(1) instead of cc(2).

  In the UNSTRING statement, the evaluation of variable I is not made dynamically. So the value of I is 1 during execution of UNSTRING.

- 3 P1 not equal to ″12300″.

- 6 The values of N2 are different.

  In the PERFORM..VARYING/AFTER statement, the increment rule of second /third.. loop counters was changed to the ″natural″ way.
  According to the new rule, this loop is the same as follows:

```
PERFORM VARYING N1 FROM 1 BY 1 UNTIL N1 > 3
   PERFORM  VARYING N2 FROM N1 BY 1 UNTIL N2 > 5
      PERFORM DISP1
   END-PERFORM
END-PERFORM.
```

A.1.2 Manual Conversion of the Sample Source

The following manual conversions were required:

- 3 A test area (TESTAREA) was reserved for comparison.
- 4 The UNSTRING statement was divided into two statements.
- 6 A dummy subroutine EX-1 was defined for emulating intermediate control of COBOL 74 standard.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. SAMPD2.
000030*              PROGRAM CONVERTED BY
000040*              COBOL CONVERSION AID PO 5785-ABJ
000050*              CONVERSION DATE 01/31/97 12:27:35.
000060 ENVIRONMENT DIVISION.
000070 CONFIGURATION SECTION.
000080 SPECIAL-NAMES.
000090     ALPHABET ALPHA-NAME IS STANDARD-1. 1
000100 DATA DIVISION.
000110 WORKING-STORAGE SECTION.
000120 01  LCP-ABND-CODE                PIC S999 COMP VALUE +519.
000130 77  AA PIC X(40).
000140 77  BB PIC X(40).
000150 77  N1 PIC 9(3).
000160 77  N2 PIC 9(5).
000160 77  N3 PIC 9(5).
000170 01  FLAG-AREA.
000180     03 REPLACE-74 PIC 9. 2
000190     03 NO-DATA PIC 9.
000200 01  TEST-AREA.
000200     03 P1 PIC 999PP. 3
000200     03 FILLER PIC XX VALUE '00'.
000210 01  CC-TAB.
000220     03 CCX  PIC X(120) VALUE SPACE.
000230     03 CC REDEFINES CCX OCCURS 3 PIC X(40).
000240 01  I  PIC 999.
000250 01  USTR-PTR PIC 999.
000260*
000270 PROCEDURE DIVISION.
000280*
000290** UNSTRING
000300     MOVE 'XX YYY ZZZZ WW'     TO AA.
000310     MOVE 1 TO I.
000320     MOVE 1 TO USTR-PTR.
000330     UNSTRING AA DELIMITED BY ALL SPACES 4
000340        INTO  BB COUNT IN I POINTER USTR-PTR.
000330     UNSTRING AA DELIMITED BY ALL SPACES 4
000350        INTO  CC( I ) POINTER USTR-PTR.
000360     DISPLAY 'I=' 1 ' VALUE=' CC ( 1 )
000370     DISPLAY 'I=' 2 ' VALUE=' CC ( 2 )
000380     DISPLAY 'I=' 3 ' VALUE=' CC ( 3 )
000390*
000400     MOVE 12300 TO P1.
000410     IF TEST-AREA = '12300' DISPLAY 'P1=12300'. 3
000420     IF TEST-AREA = '123'   DISPLAY 'P1=123'.
000430     IF TEST-AREA = '123  ' DISPLAY 'P1=123__'.
000440*
000450     MOVE 'Abcde' TO AA.
000460     IF AA IS ALPHABETIC-UPPER DISPLAY 'Ax IS ALPHA'. 5
000470     MOVE 'ABCDE' TO AA.
000480     IF AA IS ALPHABETIC-UPPER DISPLAY 'AX IS ALPHA'.
000490*
000491     MOVE 1 TO N3.
000500     PERFORM EX-1 VARYING N1 FROM 1 BY 1 UNTIL N1 > 3. 6
000520     STOP RUN.
000530*
000530 EX-1.
000510     PERFORM DISP1 VARYING  N2 FROM N3 BY 1 UNTIL N2 > 5.
000511     MOVE N1 TO N3.
000540*
000550 DISP1.
000560     DISPLAY 'N1,N2= ' N1 ' ' N2.
000570 LCP-LCP-LCP-EXIT-PROGRAM SECTION.
000580     CALL 'ILBOABN0' USING LCP-ABND-CODE.
000590 END PROGRAM SAMPD2.
```

Figure 26. SAMPD2 After Manual Update

*

Appendix B. Special Notices

This publication is intended to assist customer and IBM technical personnel in the conversion of DOS/VS COBOL and VS COBOL II applications to COBOL/VSE and LE/VSE.

The information in this publication is not intended as a specification of any programming interfaces that are provided by VSE/ESA or COBOL/VSE and LE/VSE. See the PUBLICATIONS section of the IBM Programming Announcement for these products for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

ACF/VTAM	AFP
APPN	C/370
CICS	CICS/VSE
Current	DATABASE 2
DB2	DFSORT
GDDM	IBM
IIN	ILE
ISSC	Language Environment
MERVA	NetView
PR/SM	Print Services Facility
Processor Resource/Systems Manager	PS/2
PSF	QMF
SKI	SQL/DS
System/390	VisualGen
VM/ESA	VSE/ESA
VTAM	XT
3090	

The following terms are trademarks of other companies:

C-bus	Corollary, Inc.
DOS	Microsoft Corporation
HP	Hewlett-Packard Company
PC Direct	Ziff Communications Company (used by IBM Corporation under license)
SX	Intel Corporation
UNIX	X/Open Company Ltd. (registered trademark in the United States and other countries)
Windows, Windows 95 logo	Microsoft Corporation
386	Intel Corporation

Appendix C. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 145.

- *Migration to VSE/ESA 2.1 - Why and How*, SG24-4773
- *Taking Advantage of IBM Language Environment for VSE/ESA*, SG24-4798
- *Preparing Your VSE System for the Year 2000*, SG24-4932

C.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

C.3 Other Publications

These publications are also relevant as further information sources.

Language Environment for VSE/ESA Release 4

- *Installation and Customization Guide*, SC33-6682
- *Programming Reference*, SC33-6685
- *Run-Time Migration Guide*, SC33-6687

COBOL for VSE/ESA

- *Migration Guide*, GC26-8070
- *Installation and Customization Guide*, SC26-8071
- *Programming Guide*, SC26-8072
- *Language Reference*, SC26-8073

VS COBOL II

- *Migration Guide for VSE*, GC26-3150

Debug Tool for VSE/ESA

- *User's Guide and Reference*, SC26-8797

- *Installation and Customization Guide*, SC26-8798

COBOL and CICS Command Level Conversion Aid for VSE

- *CCCA/VSE Installation and User's Guide*, SC26-8269

COBOL Report Writer Precompiler

- *Installation and Operation for VSE/ESA*, SC26-4864
- *Programmer's Manual*, SC26-4301

COBOL Structuring Facility/MVS and VM Version 3

- *COBOL/SF Host User's Guide*, SC26-3278
- *COBOL/SF Reference Guide*, SC26-3411

VSE/ESA Publications

- *VSE/ESA General Information - What's New V2.1 and V2.2*, GC33-6627
- *VSE/ESA Enhancements V2.2*, SC33-6629
- *VSE/ESA Planning V2.2*, SC33-6603
- *VSE/ESA Installation V2.1*, SC33-6604
- *VSE/ESA System Upgrade and Service V2.1*, SC33-6602

Year 2000 related publications

- *VSE/ESA Software Newsletter Special Issue - VSE and Year 2000*, G225-4508
- *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251, on the World Wide Web at *http://www.software.ibm.com/year2000/index.html*
- *VSE/ESA Home Page*, on the World Wide Web at *http://www.s390.ibm.com/vse/*.

Softcopy Publications

The following collection kit contains LE/VSE and LE/VSE-conforming language product publications:

- *VSE Collection*, SK2T-0060

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO: type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Web Site on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (http://www.redbooks.ibm.com/redpieces.htm). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Glossary

abend. Abnormal end of task.

abend code. A system code that identifies the system message number and type of error condition causing the abend.

abnormal termination. *(1)* The cessation of processing prior to planned termination.*(2)* A system failure or operator action that causes a job to end unsuccessfully.

access. *(1)* To obtain the use of a computer resource. *(2)* The use of an access method. *(3)* The manner in which files or data sets are referred to by the computer. *(4)* To obtain data from or to put data in storage. *(5)* In FORTRAN, the means by which a scoping unit accesses entities in a module subprogram or, in the case of an internal procedure, in its host. Such entities may be explicitly or implicitly accessible. Access is provided by the USE statement.

accuracy. *(1)* A quality of that which is free of error. *(2)* A qualitative assessment of freedom from error, with a high assessment corresponding to a small error. *(3)* Contrast with precision.

action. In a conceptual schema language, one or more elementary actions that, as a unit, change a collection of sentences into another one or make known a collection of sentences present in the information base or conceptual schema.

activate. *(1)* To put a device into an operational state. *(2)* To pass control to a program, procedure, or routine. *(3)* To make a resource ready to perform its function.

active. *(1)* Operational. *(2)* Pertaining to a file, page, or program that is in main storage or memory, as opposed to a file, page, or program that must be retrieved from auxiliary storage; for example, an active page in an IBM personal computer. *(3)* The state of a resource when it has been activated and is operational. Contrast with inactive, inoperative.

address. *(1)* A character or group of characters that identifies a register, a particular part of storage, or some other data source or destination. *(2)* To refer to a device or an item of data by its address. *(3)* A name, label, or number identifying a location in storage, a device in a system or network, or any other data source.

addressing. *(1)* The assignment of addresses to the instructions of a program. *(2)* A means of identifying storage locations. *(3)* Specifying an address or location within a file.

allocate. To assign a resource, such as a disk or a diskette file, to perform a task.

analysis. The methodical investigation of a problem, and the separation of the problem into smaller related units for further detailed study.

American National Standards Institute (ANSI). An organization consisting of producers, consumers, and general interest groups, that establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States.

authorized program analysis report (APAR). A report of a problem caused by a suspected defect in a current unaltered release of a program.

application. *(1)* The use to which an information processing system is put; for example, a payroll application, an airline reservation application, a network application. *(2)* A collection of software components used to perform specific types of user-oriented work on a computer.

application program. *(1)* A program that is specific to the solution of an application problem. Synonymous with application software. *(2)* A program written for or by a user that applies to the user's work, such as a program that does inventory control or payroll.

application software. *(1)* Software that is specific to the solution of an application problem. *(2)* Software coded by or for an end user that performs a service or relates to the user's work. See also system software.

argument. *(1)* An independent variable. *(2)* Any value of an independent variable; for example, a search key; a number identifying the location of an item in a table. *(3)* A parameter passed between a calling program and a called program.

arithmetic statement. In COBOL, a statement that causes an arithmetic operation to be executed.

> **Note:** The arithmetic statements are the ADD, COMPUTE, DIVIDE, MULTIPLY, and SUBTRACT statements.

array element. A data item in an array.

assemble. To translate an assembly language program into an object program.

assembler. *(1)* A translator that can assemble. *(2)* A computer program that converts assembly language instructions into object code.

assembly language. Deprecated term for assembler language.

assembly listing. The output of an assembler.

attribute. *(1)* A named property of an entity. *(2)* In FORTRAN, a property of a data object that may be specified in a type declaration statement, namely data type, type parameters, rank, shape, whether variable or constant, initial value, accessibility (PUBLIC or PRIVATE), intent (IN, OUT, or INOUT), whether allocatable, whether alias, whether optional, whether to be saved, and whether ranged.

automate. To convert a process or equipment to automatic operation.

batch. *(1)* An accumulation of data to be processed. *(2)* A group of records or data processing jobs brought together for processing or transmission. *(3)* Pertaining to activity involving little or no user action. Contrast with interactive.

batch application. In VSE, a set of programs that normally processes data without user interaction; for example, an application to print a company payroll. Such an application uses a device, a data file, or the processor intensively for a longer time than online applications.

batch job. A job submitted as a predefined series of actions to be performed with little or no interaction between user and system.

binary. *(1)* Pertaining to a selection, choice, or condition that has two possible different values or states. *(2)* Pertaining to a system of numbers to the base two; the binary digits are 0 and 1.

binary-coded decimal notation (BCD). A binary-coded notation in which each of the decimal digits is represented by a binary numeral; for example, in binary-coded decimal notation that uses the weights 8, 4, 2, 1, the number ″twenty-three″ is represented by 0010 0011 (compare its representation 10111 in the pure binary numeration system).

blank. A part of a data medium in which no characters are recorded.

block. *(1)* In programming languages, a compound statement that coincides with the scope of at least one of the declarations contained within it. A block may also specify storage allocation or segment programs for other purposes. *(2)* A string of data elements recorded or transmitted as a unit. The elements may be characters, words or physical records. *(3)* A collection of contiguous records recorded as a unit. Blocks are separated by interblock gaps and each block may contain one or more records. *(4)* A subdivision of a track on a diskette. *(5)* In PL/I, a sequence of statements, processed as a unit, that specify the scope of names and the allocation of storage for names declared within it. Contrast with DO group. *(6)* In FORTRAN, a sequence of executable constructs embedded in another executable construct, bounded by statements that are particular to the construct, and treated as an integral unit. *(7)* In COBOL, a physical unit of data that is normally composed of one or more logical records. Synonymous with physical record.

breakpoint. *(1)* A point in a computer program where execution may be halted. A breakpoint is usually at the beginning of an instruction where halts, caused by external intervention, are convenient for resuming execution. *(2)* An instruction in a program for halting execution. Breakpoints are usually established at positions in a program where halts, caused by external intervention, are convenient for restarting. *(3)* A place in a program, specified by a command or a condition, where the system halts execution and gives control to the workstation user or to a specified program.

browse. *(1)* To look at records in a file. *(2)* To look at information without changing it. See also scan and search. *(3)* To rapidly scan information on a display screen by scrolling or paging.

built-in. In programming languages, pertaining to a language object that is declared by the definition of the programming language; for example, the built-in function SIN in PL/I, the predefined data type INTEGER in FORTRAN. Synonymous with predefined.

call. *(1)* The action of bringing a computer program, a routine, or a subroutine into effect, usually by specifying the entry conditions and jumping to an entry point. *(2)* To transfer control to a procedure, program, routine, or subroutine.

caller. The requester of a service.

calling. The process of transmitting selection signals in order to establish a connection between data stations.

calling conventions. Specified ways for routines and subroutines to exchange data with each other.

calling program. A program that requests execution of another program (a called program).

cancel. To end a task before it is completed.

capture. In optical character recognition, to gather picture data from a field on an input document, using a special scan.

CASE. Computer assisted software engineering. A set of tools or programs to help develop complex applications.

CEMT. The CICS-supplied transaction that allows checking of the status of terminals, connections and

other CICS entities from a console or from CICS terminal sessions.

character. *(1)* A member of a set of elements that is used for the representation, organization, or control of data. *(2)* A letter, digit, or other symbol that is used as part of the organization, control, or representation of data. A character is often in the form of a spatial arrangement of adjacent or connected strokes. *(3)* In COBOL, the basic indivisible unit of the COBOL language.

character set. *(1)* A finite set of different characters that is complete for a given purpose; for example, the character set in ISO Standard 646. *(2)* An ordered set of unique representations called characters; for example, the 26 letters of the English alphabet, Boolean 0 and 1, the set of symbols in the Morse code, and the 128 ASCII characters. *(3)* A defined collection of characters. *(4)* All the valid characters for a programming language or for a computer system.

character string. *(1)* A string consisting solely of characters. *(2)* A sequence of consecutive characters that are used as a value. *(3)* In COBOL, a sequence of contiguous character that form a COBOL word, a literal, a PICTURE character string, or a comment-entry.

check. *(1)* A process for determining accuracy. *(2)* An error condition. *(3)* To look for a condition.

choice. An item that a user can select.

class. *(1)* In object-oriented design or programming, a group of objects that share a common definition and that therefore share common properties, operations, and behavior. Members of the group are called instances of the class. *(2)* Any category to which things are assigned or defined. *(3)* In VSE/POWER, a means of grouping jobs that require the same set of resources.

clause. *(1)* In COBOL, an ordered set of consecutive COBOL character-strings whose purpose is to specify an attribute of an entry. *(2)* In SQL, a distinct part of a statement in the language structure, such as a SELECT clause or a WHERE clause.

clear. *(1)* To put one or more storage locations or registers into a prescribed state, usually that denoting zero. *(2)* To cause one or more storage locations to be in a prescribed state, usually that corresponding to zero or that corresponding to the blank character.

close. *(1)* The function that ends the connection between a file and a program, and ends the processing. Contrast with open. *(2)* A data manipulation function that ends the connection between a file and a program. Contrast with open. *(3)* To end the processing of a file.

COBOL (common business-oriented language). A high-level programming language, based on English, that is used primarily for business applications.

code. *(1)* A set of items, such as abbreviations, representing the members of another set. *(2)* Loosely, one or more computer programs, or part of a computer program. *(3)* Instructions written for a computer. *(4)* A representation of a condition, such as an error code. *(5)* To write a routine. *(7)* To write instructions for a computer. Synonymous with program.

coded. See binary-coded decimal notation.

collate. To alter the arrangement of a set of items from two or more ordered subsets to one or more other subsets each containing a number of items, commonly one, from each of the original subsets in a specified order that is not necessarily the order of any of the original subsets.

column. *(1)* One of two or more vertical arrangements of lines, positioned side by side on a page or screen. *(2)* A vertical arrangement of characters or other expressions. *(3)* A character position within a print line or on a display. The positions are numbered from 1, by 1, starting at the leftmost character position and extending to the rightmost position. *(4)* In COBOL, a character position within a print line. Columns are numbered consecutively from 1, starting at the leftmost character position of the print line and extending to the rightmost position of the print line.

combination. A given number of different elements selected from a set without regard to the order in which the selected elements are arranged. Contrast with permutation.

Command. The typed name and parameters associated with an action that can be performed by an application. A command is one form of action request.

command line. On a display screen, a display line usually at the bottom of the screen, in which only commands can be entered.

compact. Synonym for compress.

comparison. The process of examining two or more items for identity, similarity, equality, relative magnitude, or for order in a sequence.

compatibility. The capability of a hardware or software component to conform with the interface requirements of a given data processing system without adversely affecting its functions.

compatible. *(1)* Pertaining to computers on which the same programs can be run without appreciable alteration. See also upward compatibility. *(2)*

Pertaining to the capability of hardware or software to meet the requirements of a specified interface.

compilation. Translation of a source program into an executable program (an object program).

compile. *(1)* To translate all or part of a program expressed in a high-level language into a computer program expressed in an intermediate language, an assembly language, or a machine language. *(2)* To prepare a machine language program from a computer program written in another programming language by making use of the overall logic structure of the program, or generating more than one computer instruction for each symbolic statement, or both, as well as performing the function of an assembler. *(3)* To translate a source program into an executable program (an object program). *(4)* To translate a program written in a high-level programming language into a machine language program.

compiled program. The set of machine language instructions that is the output of the compilation of a source program.

compiler. *(1)* A translator that can compile. *(2)* A program that translates a source program into an executable program (an object program). *(3)* A program that decodes instructions written as pseudo codes and produces a machine language program to be executed at a later time. *(4)* A program that translates instructions written in a high-level programming language into machine language.

compiler listing. A printout produced by compiling a program or creating a file and that optionally includes, for example, a line-by-line source listing, cross-reference list, diagnostic information, and for programs, a description of externally described files.

compiler options. Keywords that can be specified to control certain aspects of compilation. Compiler options can control the nature of the load module generated by the compiler, the types of printed output to be produced, the efficient use of the compiler, and the destination of error messages.

completion code. A return code indicating that an operation has ended.

component. *(1)* Hardware or software that is part of a functional unit. *(2)* A functional part of an operating system; for example, the scheduler or supervisor. *(3)* A set of modules that performs a major function within a system; for example, a compiler or a master scheduler. *(4)* In systems with VSAM, a named, cataloged collection of stored records, such as the data component or index component of a key-sequenced file or alternate index. *(5)* A part of a structured type or value, such as an array.

conceptual system design. A system design activity concerned with specifying the logical aspects of the system organization, its processes, and the flow of information through the system.

condition. *(1)* One of a set of specified values that a data item can assume. *(2)* An expression in a program or procedure that can be evaluated as either true or false when the program or procedure is running. *(3)* In COBOL, the status of a program at object time for which a truth value can be determined. *(4)* In FORTRAN, a named circumstance in which it is inappropriate to continue the normal execution sequence.

connectivity. *(1)* The capability of a system or device to be attached to other systems or devices without modification. *(2)* The capability to attach a variety of functional units without modifying them.

context. A stated or implied sense in which a thing has meaning, or a category or scope to which it applies.

control. The determination of the time and order in which the parts of a data processing system and the devices that contain those parts perform the input, processing, storage, and output functions.

control block. A storage area used by a computer program to hold control information. Synonymous with control area.

conversion. *(1)* In programming languages, the transformation between values that represent the same data item but belong to different data types. Information may be lost due to conversion since accuracy of data representation varies among different data types. *(2)* The process of changing from one method of data processing to another or from one data processing system to another. *(3)* The process of changing from one form of representation to another; for example, to change from decimal representation to binary representation.

convert. To change the representation of data from one form to another, without changing the information they convey; for example, radix conversion, code conversion, analog to digital conversion, media conversion.

CPU. Central processing unit. See also processor.

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining databases.

cycle. *(1)* An interval of space or time in which one set of events or phenomena is completed. *(2)* Any set of operations repeated regularly in the same

sequence. The operations may be subject to variations on each repetition.

data. *(1)* A re-interpretable representation of information in a formalized manner suitable for communication, interpretation, or processing. Operations can be performed upon data by humans or by automatic means. *(2)* A representation of facts or instructions in a form suitable for communication, interpretation, or processing by human or automatic means. Data include constants, variables, arrays, and character strings.

DATABASE 2 (DB2). An IBM relational database management system.

data item. *(1)* The smallest unit of named data that has meaning in the schema or subschema. Synonymous with data element. *(2)* A unit of data, either a constant or a variable, to be processed. *(3)* In COBOL, a unit of data, excluding literals, defined by a COBOL program. *(4)* In FORTRAN, a constant, variable, or array element.

data set. *(1)* The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. *(2)* See also file.

data value. In a database, an item of data viewed as a member of a data type.

debug. *(1)* To detect, to locate, and to eliminate errors in computer programs. *(2)* To detect, diagnose, and eliminate errors in programs. *(3)* Synonymous with checkout, troubleshoot.

debugger. A program or programs used to detect, trace, and eliminate errors in computer programs or other software.

debugging. Acting to detect and correct errors in software or system configuration.

decode. *(1)* To convert data by reversing the effect of some previous encoding. *(2)* To interpret a code.

default. Pertaining to an attribute, condition, value, or option that is assumed when none is explicitly specified.

default option. The implicit option assumed when no option is explicitly stated.

delete. *(1)* A function that enables a user to remove all or part of a previously entered text. *(2)* To remove an object or a unit of data, such as a character, field, or record.

design. See conceptual system design, functional design, logic design, system design.

development time. The part of operating time used for debugging new routines or hardware.

device. A mechanical, electrical, or electronic contrivance with a specific purpose.

device independence. The capability to write and execute programs or perform functions without regard for the physical characteristics of devices. Contrast with device dependence.

diagnostic. Pertaining to the detection and isolation of errors in programs and faults in equipment.

dictionary. A database of specifications of data and information processing resources.

disabled. Pertaining to a state of a processing unit that prevents the occurrence of certain types of interruptions. Synonymous with masked.

disk. *(1)* A round, flat, data medium that is rotated in order to read or write data. *(2)* Loosely, a magnetic disk unit. *(3)* Synonym for magnetic disk.

display. *(1)* A visual presentation of data. *(2)* To present data visually. *(3)* Deprecated term for panel.

download. To transfer programs or data from a computer to a connected device, typically a personal computer. Contrast with upload.

dummy. Pertaining to the characteristic of having the appearance of a specified thing but not having the capacity to function as such; for example, a dummy character, dummy plug, or dummy statement.

dump. *(1)* To record, at a particular instant, the contents of all or part of one storage device in another storage device. Dumping is usually for the purpose of debugging. *(2)* To copy data in a readable format from main or auxiliary storage onto an external medium such as tape, diskette, or printer. *(4)* To copy the contents of all or part of virtual storage for the purpose of collecting error information.

dynamic. *(1)* In programming languages, pertaining to properties that can only be established during the execution of a program; for example, the length of a variable-length data object is dynamic. *(2)* Pertaining to an operation that occurs at the time it is needed rather than at a predetermined or fixed time. *(3)* Pertaining to events occurring at run time, or during processing. *(4)* Contrast with static.

edit. *(1)* To add, change, delete, or rearrange data and to perform operations such as code conversion and zero suppression. *(2)* To enter, modify, or delete data. *(3)* To modify a numeric field for output by suppressing zeros and inserting commas, periods, currency symbols, the sign status, or other constant information.

editor program. A computer program designed to perform such functions as rearrangement, modification, and deletion of data in accordance with prescribed rules.

element. *(1)* In a set, an object, entity, or concept having the properties that define a set. *(2)* A parameter value in a list of parameter values. *(3)* The smallest unit of data in a table or array. *(4)* In FORTRAN, see array element.

enable. *(1)* To make functional. *(2)* In interactive communications, to load and start a subsystem. Contrast with disable.

end of file (EOF). A coded character recorded on a data medium to indicate the end of the medium. See also end-of-file label.

end-of-file label. *(1)* An internal label indicating the end of a file and possibly containing data for file control. *(2)* Synonymous with trailer label.

enter. *(1)* To place on the line a message to be transmitted from a terminal to the computer. *(2)* To press the Enter/Rec Adv key (on a workstation keyboard) or the Enter key (on the system console) or a command function key to transfer keyed-in information to the system for processing. *(3)* To type in information on a keyboard and press the Enter key to send the information to the computer.

entry. *(1)* In programming languages, a language construct within a procedure, designating the start of the execution sequences of the procedure. A procedure may have more than one entry; each entry usually includes an identifier, called the entry name, and may include formal parameters. *(2)* An element of information in a table, list, queue, or other organized structure of data or control information. *(3)* A single input operation on a terminal. *(4)* In FORTRAN, a language construct within a procedure, designating the start of the execution sequences of the procedure. *(6)* In COBOL, any descriptive set of consecutive clauses terminated by a separator period and written in the Identification Division, Environment Division, or Data Division of a COBOL program.

entry point (EP). *(1)* The address or label of the first instruction executed on entering a computer program, routine, or subroutine. A computer program, routine, or subroutine may have a number of different entry points, each perhaps corresponding to a different function or purpose. Synonymous with entrance, entry. *(2)* In a routine, any place to which control can be passed.

error. *(1)* A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. *(2)* Contrast with failure, fault, malfunction, mistake.

error condition. The state that results from an attempt to execute instructions in a computer program that are invalid or that operate on invalid data.

error message. An indication that an error has been detected.

EXEC. In a VM operating system, a user-written command file that contains CMS commands, other user-written commands, and execution control statements, such as branches.

execute. *(1)* To perform the actions specified by a program or a portion of a program. *(2)* To carry out an instruction.

executing phase. Synonym for execute phase.

execution element (EE). An element in a central processor that performs all floating-point, fixed-point multiply, fixed-point divide, and convert operations.

execution. The process of carrying out an instruction or instructions of a computer program by a computer.

execution time. *(1)* Any instant at which the execution of a particular computer program takes place. *(2)* The amount of time needed for the execution of a particular computer program. *(3)* The time during which an instruction in an instruction register is decoded and performed.

exit. To execute an instruction within a portion of a computer program in order to terminate the execution of that portion. Such portions of computer programs include loops, subroutines, modules, and so on.

exit program. Synonym for exit routine.

extended addressing. A direct-addressing mode that can access any area in storage.

external. In programming languages, pertaining to a language object that has a scope that extends beyond one module; for example, the entry names of a module.

facility. *(1)* An operational capability, or the means for providing such a capability. *(2)* A service provided by an operating system for a particular purpose; for example, the checkpoint/restart facility.

feature. A part of an IBM product that may be ordered separately by the customer. A feature is designated as either special or specify and may be designated also as diskette-only.

feedback. The return of part of the output of a machine, process, or system as input to the computer, especially for self-correcting or control purposes.

FF. The form feed character.

field. On a data medium or a storage, a specified area used for a particular class of data; for example, a group of character positions used to enter or display wage rates on a screen.

file. *(1)* A named set of records stored or processed as a unit. *(2)* A collection of information treated as a unit. *(3)* A collection of related data that is stored and retrieved by an assigned name. *(4)* In COBOL, a collection of logical records.

file attribute. Any of the attributes that describe the characteristics of a file.

file definition. *(1)* Information that describes the contents and characteristics of a file. *(2)* In VM, equating a CMS file identifier (file name, file type, file mode) with an OS data set name via the FILEDEF command, or equating a VSE file-id with a CMS file identifier via the DLBL command. *(3)* In VM, identifying the input or output files to be used during execution of a program (via either the FILEDEF or DLBL commands).

file description. A part of a file where file and field attributes are described.

file description (FD) entry. In COBOL, an entry in the File Section of the Data Division that is composed of the level indicator FD, followed by a file-name, and then followed by a set of file clauses as required.

file layout. The arrangement and structure of data or words in a file, including the order and size of the components of the file.

FILE-CONTROL. In COBOL, the name of an Environment Division paragraph in which the data files for a given source program are declared.

find. Synonym for search.

fix. A correction of an error in a program, usually a temporary correction or bypass of defective code.

flag. *(1)* A variable indicating that a certain condition holds. *(2)* Any of various types of indicators used for identification *(3)* A character that signals the occurrence of some condition, such as the end of a word. *(4)* Deprecated term for mark.

flaw. In computer security, an error of commission, omission, or oversight that allows protection mechanisms to be bypassed or disabled.

format. *(1)* In programming languages, a language construct that specifies the representation, in character form, of data objects in a file. *(2)* A specified arrangement of such things as characters, fields, and lines, usually used for displays, printouts, or files. *(3)* To arrange such things as characters, fields, and lines. *(4)* To arrange information on a page, in a file, or on a display screen.

forward. To send information, such as received mail, to someone else.

frame. *(1)* A data structure that consists of fields, predetermined by a protocol, for the transmission of user data and control data. The composition of a frame, especially the number and types of fields, may vary according to the type of protocol. Synonymous with transmission frame. *(2)* A housing for machine elements. *(3)* The hardware support structure, covers, and all electrical parts mounted therein that are packaged as one entity for shipping.

free storage. Storage that is not allocated.

FS. The file separator character.

full-screen mode. *(1)* A form of screen presentation in which the contents of an entire terminal screen can be displayed at once. Full-screen mode is often used for fill-in-the-blanks prompting. *(2)* In VM/ESA, a mechanism to allow a virtual machine to have control over a 3270 display screen. Contrast with line mode.

fullword. Synonym for computer word.

function. *(1)* A mathematical entity whose value, that is, the value of the dependent variable, depends in a specified manner on the values of one or more independent variables, not more than one value of the dependent variable corresponding to each permissible combination of values from the respective ranges of the independent variables. *(2)* A subroutine that returns the value of a single variable and that usually has a single exit; for example, subroutines that compute mathematical functions, such as sine, cosine, logarithm, or that compute the maximum of a set of numbers. *(3)* In the C and FORTRAN languages, a named group of statements that can be called and evaluated, and can return a value to the calling statement. *(4)* In PL/I, a procedure that has a RETURNS option in the PROCEDURE statement. A function ends by running a RETURNS statement and returning a scalar value to the point of call.

functional design. The specification of the functions of the components of a system and of the working relationships among them.

generate. *(1)* To produce a computer program by selecting subsets from skeletal code under the control of parameters. *(2)* To produce assembler language statements from the model statements of a macrodefinition when the definition is called by a macroinstruction.

get. To obtain a record from an input file.

group. *(1)* A set of related records that have the same value for a particular field in all records. *(2)* A series of records logically joined together. *(3)* A series of lines repeated consecutively as a set on a

full-screen form or full-screen panel. *(4)* A list of names that are known together by a single name.

halfword. A contiguous sequence of bits or characters that constitutes half a computer word and can be addressed as a unit.

hardware. *(1)* All or part of the physical components of an information processing system, such as computers or peripheral devices. *(2)* The equipment, as opposed to the programming, of a system. *(3)* Contrast with software.

host system. *(1)* A data processing system used to prepare programs and operating environments for use on another computer or controller. *(2)* The data processing system to which a network is connected and with which the system can communicate. *(3)* The controlling or highest level system in a data communication configuration; for example, a System/390 is the host system for the workstations connected to it.

I-O-CONTROL. In COBOL, the name of an Environment Division paragraph in which object program requirements for rerun points, sharing of the same areas by several data files, and multiple file storage on a single input/output device are specified.

ID. *(1)* Identifier. *(2)* Identification.

identification. In computer security, the process that allows a system to recognize an entity by means of personal, equipment, or organizational characteristics or codes.

identifier. *(1)* One or more characters used to identify or name a data element and possibly to indicate certain properties of that data element. *(2)* In programming languages, a token that names a data object such as a variable, an array, a record, a subprogram, or a function. *(3)* In COBOL, a syntactically correct combination of a data name, with its qualifiers, subscripts, and reference modifiers, as required for uniqueness of reference, that names a data item. The rules for an identifier associated with the general formats may, however, specifically prohibit qualification, subscripting, or reference modification. *(4)* In FORTRAN, a lexical unit that names a language object; for example, the names of variables, arrays, and program units. The name of a declared unit. *(5)* In PL/I, a single alphabetic character or a string of alphabetic characters, digits, and break characters that starts with an alphabetic character. identifier, ordinary identifier.

implementation. The system development phase at the end of which the hardware, software and procedures of the system considered become operational.

increment. A value used to alter a counter or register.

index. *(1)* A list of the contents of a file or of a document, together with keys or references for locating the contents. *(2)* In programming, an integer that identifies the position of a data item in a sequence of data items. *(3)* A symbol or a numeral used to identify a particular quantity in an array of similar quantities; for example, the terms of an array represented by X1, X2,..., X100 have the indexes 1, 2,..., 100, respectively. *(4)* A table used to locate records in an indexed sequential data set or an indexed file. *(5)* In VSAM, an ordered collection of pairs, each consisting of a key and a pointer, used by VSAM to sequence and locate the records of a key-sequenced data set or file; it is organized in levels of index records. *(6)* In COBOL, a computer storage area or register, the contents of which represents the identification of a particular element in a table.

information. In information processing, knowledge concerning such things as facts, concepts, objects, events, ideas, and processes, that within a certain context has a particular meaning.

initial program load (IPL). *(1)* The initialization procedure that causes an operating system to commence operation. *(2)* The process of loading system programs and preparing a system to run jobs. *(3)* Synonymous with system restart, system startup.

initialization. *(1)* The operations required for setting a device to a starting state, before the use of a data medium, or before implementation of a process. *(2)* Preparation of a system, device, or program for operation. *(3)* See also initial machine load, IML, initial program load.

input. *(1)* Pertaining to a device, process, or channel involved in an input process, or to the associated data or states. The word ″input″ may be used in place of ″input data,″ ″input signal″, ″input process″, when such a usage is clear in a given context. *(2)* Pertaining to a functional unit or channel involved in an input process or to the data involved in such a process. *(3)* Loosely, input data, input process. *(4)* Information or data to be processed.

input file. *(1)* A file that has been opened in order to allow records to be read. Contrast with output file. *(2)* In COBOL, a file that is opened in the input mode.

input program. A utility program that organizes the input process of a computer.

insert. *(1)* A function or mode that enables the introduction of additional characters within previously entered text. *(2)* To introduce data between previously stored items of data.

insertion. *(1)* The introduction of data or text within previously stored data or text. *(2)* In a conceptual

schema language, the addition of a sentence to the information base or to the conceptual schema.

install. *(1)* To add a program, program option, or software to a system in such a manner that it is runnable and interacts properly with all affected programs in the system. *(2)* To connect hardware to a system.

installation. *(1)* In system development, preparing and placing a functional unit in position for use. *(2)* A particular computing system, including the work it does and the people who manage it, operate it, apply it to problems, service it, and use the results it produces.

instruction. *(1)* A language construct that specifies an operation and identifies its operands, if any. *(2)* A statement that specifies an operation to be performed by a system and that identifies data involved in the operation. *(3)* In COBOL, one or more clauses, the first of which starts with a keyword that identifies the instruction. Instructions affect the flow of control, provide services to the programmer, or both.

integer. *(1)* One of the numbers zero, +1, -1, +2, -2... Synonymous with integral number. *(2)* A positive or negative whole number, that is, an optional sign followed by a number that does not contain a decimal place or zero. *(3)* In COBOL, a numeric literal or a numeric data item that does not include any digit position to the right of the assumed decimal point. When the term ″integer″ appears in general formats, the integer must not be a numeric data item, and must not be signed, nor zero unless explicitly allowed by the rules of that format.

integrated. Pertaining to a feature that is part of a device. Synonymous with built-in.

integrated software. *(1)* Application software such as spreadsheets, word processing programs, and database management programs that can be used interchangeably to exchange and operate on the same data. *(2)* Personal computer application software that allows the use of two or more applications concurrently.

interaction. *(1)* A basic unit used to record system activity, consisting of the acceptance of a line of terminal input, processing of the line, and a response, if any. *(2)* The exchange of information between a user and a computer.

interactive. Pertaining to a program or system that alternately accepts input and then responds. An interactive system is conversational, that is, a continuous dialog exists between user and system. Contrast with batch.

interface. *(1)* A shared boundary between two functional units, defined by functional characteristics, signal characteristics, or other characteristics, as appropriate. The concept includes the specification of the connection of two devices having different functions. *(2)* Hardware, software, or both, that links systems, programs, or devices.

intrinsic. In FORTRAN, pertaining to types, operators, procedures, and conditions, defined in the ANSI FORTRAN Standard, that may be used in any scoping unit without further definition or specification.

invocation. *(1)* The activation of a program or procedure. *(2)* An execution of a program.

invoke. To start a command, procedure, or program.

item. An element of a set of data; for example, a file may consist of a number of items such as records which in turn may consist of other items.

job. *(1)* A unit of work defined by a user that is to be accomplished by a computer. Loosely, the term job is sometimes used to refer to a representation of a job. This representation may include a set of computer programs, files, and control statements to the operating system. *(2)* A collection of related programs, identified by appropriate job control statements.

job control. In VSE, a program called into storage to prepare each job or job step to be run. Some of its functions are to assign I/O devices to symbolic names, set switches for program use, log (or print) job control statements, and fetch the first phase of each job step.

key. *(1)* An identifier within a set of data elements. *(2)* One or more characters used to identify the record and establish the order of the record within an indexed file. *(3)* In VSAM, one or more consecutive characters taken from a data record, used to identify the record and establish its order with respect to other records. *(4)* The value used to identify a record in a keyed sequence file. *(5)* In COBOL, a data item that identifies the location of a record, or a set of data

keyword. *(1)* In programming languages, a lexical unit that, in certain contexts, characterizes some language construct; for example, in some contexts, IF characterizes an if-statement. A keyword normally has the form of an identifier. *(2)* One of the predefined words of an artificial language. *(3)* A name or symbol that identifies a parameter. *(4)* Part of a command operand that consists of a specific character string, such as DISP=. *(5)* In PL/I, an identifier used with related material that takes on a specific meaning, such as an action to be taken or the characteristics of data. *(6)* In COBOL, a reserved word whose presence is required when the format in which the word appears is used in a source program. *(7)* In FORTRAN, a statement keyword or an argument keyword. *(8)* Deprecated term for reserved word.

label. *(1)* In programming languages, a language construction naming a statement and including an identifier. *(2)* An identifier within or attached to a set of data elements. *(3)* A record that identifies a volume on tape, disk, or diskette or that identifies a file on the volume. *(4)* An identifier of a command generally used for branching. *(5)* In PL/I, an identifier that names a statement so that it can be referred to at some other point in the program. *(5)* Synonymous with tag.

language. A set of characters, conventions, and rules that is used for conveying information.

layout. See file layout, record layout.

level. *(1)* The degree of subordination of an item in a hierarchic arrangement. *(2)* In a hierarchical database, the successive vertical dependencies of records or segments. *(3)* The version of a program.

library. *(1)* A file or a set of related files; for example, in stock control, a set of inventory control files. *(2)* A repository for demountable recorded media, such as magnetic disk packs and magnetic tapes. *(3)* A collection of functions, calls, subroutines, or other data. *(4)* A data file that contains files and control information that allows them to be accessed individually. *(5)* In VSE, a collection of data stored in sublibraries on disk. A library consists of at least one sublibrary in which data is stored as members of various types such as phase, object module, or source book.

library block. In VSE, a block of data stored in a sublibrary.

library member. *(1)* A named collection of records or statements in a library. *(2)* In VSE, the smallest unit of data that can be stored into and retrieved from a sublibrary.

link-edit. To create a loadable computer program by means of a linkage editor.

Linkage Section. In COBOL, the section in the Data Division of the called program that describes the data items available from the calling program. These data items may be referred to by both the calling and the called program.

list. *(1)* An ordered set of data. *(2)* A data object consisting of a collection of related records.

listing. A printout that lists the source language statements and the output resulting from execution of a program. See also compiler listing.

literal. *(1)* In programming languages, a lexical unit that directly represents a value; for example, 14 represents the integer fourteen, ″APRIL″ represents the string of characters APRIL, 3.0005E2 represents the number 300.05. *(2)* A symbol or a quantity in a source program that is itself data, rather than a reference to data. *(3)* A character string whose value is given by the characters themselves; for example, the numeric literal 7 has the value 7, and the character literal ″CHARACTERS″ has the value CHARACTERS. *(4)* In COBOL, a character-string whose value is implied by the ordered set of characters comprising the string.

load. *(1)* To feed data into a database. *(2)* To bring all or part of a computer program into memory from auxiliary storage so that the computer can run the program.

logic design. A functional design that uses formal methods of description, such as symbolic logic.

logical. *(1)* Pertaining to content or meaning as opposed to location or actual implementation. *(2)* Pertaining to a view or description of data that does not depend on the characteristics of the computer system or of the physical storage. *(3)* Contrast with physical.

long. A signed 4-byte number.

loop. *(1)* A sequence of instructions that is to be executed iteratively. *(2)* A closed unidirectional signal path connecting input/output devices to a system.

looping. Repetitive execution of the same statement or statements, usually controlled by a DO statement.

macro. Synonym for macroinstruction.

maintenance. *(1)* Any activity intended to retain a functional unit in, or to restore it to, a state in which it can perform its required function. Maintenance includes keeping a functional unit in a specified state by performing activities such as tests, measurements, replacements, adjustments, and repairs. *(2)* The activities intended to keep a machine in, or restore a machine to, good working order.

match. A comparison to determine identity of items.

member. *(1)* A partition of a partitioned data set. *(2)* In VSE, the smallest unit of data that can be stored in and retrieved from a sublibrary. See also library member. *(3)* A data object in a structure, a union, or a library. *(4)* Synonym for element.

member name. In SDF/CICS, a minor identification of an object in the map specification library. Member names can be map names, profile names, and page names.

merge. *(1)* To combine the items of two or more sets that are each in the same given order into one set in that order. *(2)* See also collate.

message. *(1)* In information theory, an ordered series of characters intended to convey information. *(2)* An assembly of characters and sometimes control

codes that is transferred as an entity from an originator to one or more recipients. *(3)* A communication sent from a person or program to another person or program. *(4)* A unit of data sent over a telecommunication line. *(5)* In COBOL, data associated with an end of message indicator or an end of group indicator.

method. Methods are used to implement behavior specified by an operation.

migrate. *(1)* To move data from one hierarchy of storage to another. *(2)* To move to a changed operating environment, usually to a new release or version of a system.

migration. *(1)* The process of moving data from one computer system to another without converting the data. *(2)* Installation of a new version or release of a program to replace an earlier version or release.

minimize. To remove from the screen all windows associated with an application and replace them with an icon that represents the application.

mode. A method of operation.

modification. *(1)* An addition or change to stored data or a deletion of stored data. *(2)* The change or customization of a system, subsystem, or application to work more effectively at a given installation.

module. *(1)* In programming languages, a language construct that consists of procedures or data declarations and that can interact with other such constructs. *(2)* A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading; for example, the input to or output from an assembler, compiler, linkage editor, or executive routine. *(3)* A packaged functional hardware unit designed for use with other components. *(4)* A part of a program that usually performs a particular function or related functions. *(5)* In FORTRAN, an external program unit that contains or accesses definitions to be accessed by other program units.

monitor. *(1)* A device that observes and records selected activities within a data processing system for analysis. Possible uses are to indicate significant departure from the norm, or to determine levels of utilization of particular functional units. *(2)* Software or hardware that observes, supervises, controls, or verifies operations of a system. *(3)* Synonym for video display terminal.

multitasking. A mode of operation that provides for concurrent performance, or interleaved execution of two or more tasks.

name. An alphanumeric term that identifies a data set, statement, program, or cataloged procedure. The first character of the name must be alphabetic. See also label.

number. *(1)* A mathematical entity that may indicate quantity or amount of units. *(2)* Loosely, a numeral.

numeric. *(1)* Pertaining to data that consist of numerals. *(2)* Pertaining to data or to physical quantities that consist of numerals. Synonymous with numerical. *(3)* Pertaining to any of the digits 0 through 9.

numeric data. *(1)* Data represented by numerals. *(2)* Data in the form of numerals and some special characters; for example, a date represented as 81/01/01. Synonymous with numerical data.

object. *(1)* A passive entity that contains or receives data; for example, bytes, blocks, clocks, fields, files, directories, displays, keyboards, network nodes, pages, printers, processors, programs, records, segments, words. Access to an object implies access to the information it contains. *(2)* In programming languages, a data object.

object code. Output from a compiler or assembler which is itself executable machine code or is suitable for processing to produce executable machine code.

online. *(1)* Pertaining to the operation of a functional unit when under the direct control of the computer. *(2)* Pertaining to a user′s ability to interact with a computer. *(3)* Pertaining to a user′s access to a computer via a terminal. *(4)* Controlled by, or communicating with, a computer. *(5)* Contrast with offline.

open. *(1)* The function that connects a file to a program for processing. *(2)* To prepare a file for processing. *(3)* Contrast with close.

operating environment. See operational environment

operating system (OS). Software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input/output control, and data management. Although operating systems are predominantly software, partial hardware implementations are possible.

operation. *(1)* A well-defined action that, when applied to any permissible combination of known entities, produces a new entity; for example, the process of addition in arithmetic; in adding five and three and obtaining eight, the numbers five and three are the operands, the number eight is the result, and the plus sign is the operator indicating that the operation performed is addition. *(2)* A program step undertaken or executed by a computer; for example, addition, multiplication, extraction, comparison, shift, transfer. The operation is usually specified by the operator part of an instruction.

operational environment. *(1)* The physical environment; for example, temperature, humidity, and

layout. *(2)* All of the supplied basic functions and the user programs that can be executed by a store controller to enable the devices in the system to perform specific operations. *(3)* The collection of supplied store controller data, user programs, lists, tables, control blocks, and files that reside in a subsystem store controller and control its operation.

operational sign. An algebraic sign associated with a numeric data item or a numeric literal that indicates whether the item is positive or negative.

option. *(1)* A specification in a statement that may be used to influence the execution of the statement. *(2)* See default option.

option set. A set of functions that may be supported by products that implement a particular architecture. A product may support any number of option sets or none. For each option set supported, all functions in that set are supported.

order. *(1)* A specified arrangement used in ordering. *(2)* To place items in an arrangement in accordance with specified rules. *(3)* Deprecated term for instruction, sequence.

> **Note:** In contrast to a sequence, an order need not be linear; for example, the ordering of a hierarchy of items.

output. *(1)* Pertaining to a device, process, or channel involved in an output process, or to the associated data or states. The word ″output″ may be used in place of ″output data,″ ″output signal″, ″output process″, when such a usage is clear in a given context. *(2)* Data that has been processed. *(3)* Data transferred from storage to an output device.

output data. *(1)* Data that a data processing system or any of its parts transfers outside of that system or part. *(2)* Data being produced or to be produced by a device or a computer program. *(3)* Data delivered or to be delivered from a functional unit or from any part of a functional unit. *(4)* Synonymous with output

output file. *(1)* A file that contains the results of processing. *(2)* A file that has been opened in order to allow records to be written. *(3)* In COBOL, a file that is opened in either the output mode or the extend mode. *(4)* Contrast with input file.

override. *(1)* A parameter or value that replaces a previous parameter or value. *(2)* The attributes specified at run time that change the attributes specified in the file description or in the program. *(3)* To replace a parameter or value.

overwrite. To write into an area of storage, thereby destroying the data previously stored in the same area.

page. *(1)* In a virtual storage system, a fixed-length block that has a virtual address and is transferred as a unit between real storage and auxiliary storage. *(2)* The information displayed at the same time on the screen of a display device. *(3)* A defined unit of space on a storage medium. *(4)* In VSE, a fixed-length block of instructions, data, or both that can be located in processor storage or in the page data set on disk. *(5)* In COBOL, a vertical division of a report representing a physical separation of report data, the separation being based on internal reporting requirements, external characteristics of the reporting medium, or both.

panel. *(1)* In VSE/ESA, the complete set of information shown in a single display on a display station screen. Each panel is like a manual page; scrolling back and forth through panels is like turning manual pages. *(2)* A set of logically related information displayed on the screen for the purpose of communicating information to or from a computer user. *(3)* A formatted display of information that appears on a display screen.

paragraph. *(1)* In the COBOL Procedure Division, a paragraph-name followed by a separator period and by zero, one, or more than one sentence. In the Identification and Environment Divisions, a paragraph header followed by zero, one, or more than one entries. See FILE-CONTROL, I-O-CONTROL.

parameter. *(1)* A variable that is given a constant value for a specified application and that may denote the application. *(2)* An item in a menu for which the user specifies a value or for which the system provides a value when the menu is interpreted. *(3)* Data passed between programs or procedures.

parse. *(1)* In systems with time sharing, to analyze the operands entered with a command and create a parameter list for the command processor from the information. *(2)* In REXX, to split a string into parts, using function calls or by using a parsing template on the ARG, PARSE, or PULL instructions.

partition. *(1)* A fixed-size division of storage. *(2)* In VSE, a division of the virtual address area that is available for program execution.

pass. One cycle of processing a body of data.

path. *(1)* The route used to locate files; the storage location of a file. A fully qualified path lists the drive identifier, directory name, subdirectory name (if any), and file name with the associated extension. *(2)* In VSAM, a named, logical entity providing access to the records of a base cluster either directly or through an alternate index.

performance. One of the two major factors, together with facility, on which the total productivity of a system depends. Performance is largely determined by a combination of throughput, response time, and availability.

phase. *(1)* A distinct part of a process in which related operations are performed. *(2)* A part of a sort/merge program; for example, sort phase, merge phase. *(3)* In VSE, the smallest complete unit of executable code that can be loaded into virtual storage.

phrase. In COBOL, an ordered set of one or more consecutive COBOL character-strings that form a portion of a COBOL procedural statement or of a COBOL clause.

picture. *(1)* In a programming language, a language construct that describes a data type by means of a model character string literal. *(2)* In a program, a string of characters used in editing to modify the individual characters in a field. There is a one-to-one relationship between the characters in a picture and the characters in its field.

PL/I. Programming language one. A programming language designed for numeric scientific computations, business data processing, systems programming, and other applications. PL/I is capable of handling a large variety of data structures and easily allows variation of precision in numeric computation.

platform. *(1)* The operating system environment in which a program runs. *(2)* In computer technology, the principles on which an operating system is based.

post. *(1)* To note the occurrence of an event. *(2)* To add information to a record in order to keep the record current.

precision. *(1)* A measure of the ability to distinguish between nearly equal values; for example, four-place numerals are less precise than six-place numerals; nevertheless, a properly computed four-place numeral may be more accurate than an improperly computed six-place numeral. *(2)* The degree of discrimination with which a quantity is stated; for example, a three-digit numeral discriminates among 1000 possibilities.

precompile. Enables COBOL/VSE to compile source programs written for DOS/VS COBOL that incorporate Report Writer without needing to convert or rewrite the Report Writer code.

previous release. The last required release of a system, such as Release 1.0, prior to the current release, such as Release 2.0, including any modification levels, such as Release 1.0 Modification Level 1 or Modification Level 2, that were not required.

procedure. *(1)* In a programming language, a block, with or without formal parameters, whose execution is invoked by means of a procedure call. *(2)* The description of the course of action taken for the solution of a problem. *(3)* A set of related control statements that cause one or more programs to be performed. *(4)* In COBOL, a paragraph or group of logically successive paragraphs, or a section or group of logically successive sections, within the Procedure Division. *(5)* In PL/I, a block of programming statements that can be started from various points within a program by use of a CALL statement and can process data passed to it from the block in which it was called. *(6)* In FORTRAN, a computation that may be invoked during program execution. It may be a function or a subroutine. A procedure subprogram may define more than one procedure if it contains ENTRY statements.

Procedure Division. One of the four main parts of a COBOL program. The Procedure Division contains instructions for solving a problem. The Procedure Division may contain imperative statements, conditional statements, paragraphs, procedures and sections.

process. *(1)* The course of events that occurs during the execution of all or part of a program. *(2)* Any operation or combination of operations on data. *(3)* A function being performed or waiting to be performed. *(4)* To perform operations on data in a process.

processing. *(1)* The performance of logical operations and calculations on data, including temporary retention of data in processor storage while the data is being operated on. *(2)* The action of performing operations on input data.

processor. A functional unit that interprets and executes instructions. A processor consists of at least an instruction control unit and an arithmetic and logic unit.

profile. *(1)* Data that describes the significant characteristics of a user, a group of users, or one or more computer resources. *(2)* In computer security, a description of the characteristics of an entity to which access is controlled. *(3)* A description of the control available to a particular network operator.

program. *(1)* A sequence of instructions suitable for processing by a computer. Processing may include the use of an assembler, a compiler, an interpreter, or a translator to prepare the program for execution, as well as to execute it. *(2)* In programming languages, a logical assembly of one or more interrelated modules. *(3)* A syntactic unit that conforms to the rules of a particular programming language composed of declarations and statements or instructions needed to solve a certain function, task, or problem. Synonymous with computer program.

program product. Deprecated term for licensed program.

program unit. In FORTRAN, the fundamental component of a FORTRAN program; a sequence of

statements and comment lines. It may be a main program or a subprogram.

programmable. Pertaining to a device that can accept instructions that alter its basic functions.

programmable workstation. A workstation that has some degree of processing capability and that allows a user to change its functions.

programmer. A person who designs, writes, and tests computer programs.

programming. The design, writing, modifying, and testing of programs.

programming language. An artificial language for expressing computer programs.

project. An undertaking with prescribed objectives, magnitude, and duration.

project management. The activities concerned with project planning and project control.

put. To place a single data record into an output file.

queue. *(1)* A line or list of items waiting to be processed; for example, work to be performed or messages to be displayed. *(2)* In COBOL, a logical collection of messages awaiting transmission or processing.

quick start. Synonym for system restart

quit. A key, command, or action that tells a system to return to a previous state or stop a process.

range. *(1)* The set of values that a quantity or function may take. *(2)* Synonym for span

range check. A limit check in which both high and low values are stipulated.

read. To acquire or interpret data from a storage device, from a data medium, or from another source.

reader. *(1)* A device that converts information in one form of storage to information in another form of storage. *(2)* A part of an operating system scheduler that reads an input stream into the system. *(3)* A program that reads jobs from an input device or database file and places them on a job queue.

reading. Acquisition or interpretation of data from a storage device, from a data medium, or another source.

receive. *(1)* To obtain and store data. *(2)* In systems with VTAM, to obtain a message transmitted from a terminal to the computer over a line. Contrast with send.

record. *(1)* In programming languages, an aggregate that consists of data objects, possibly with different attributes, that usually have identifiers attached to them. In some programming languages, records are called structures. *(2)* A set of one or more related data items grouped for processing. *(3)* In COBOL, synonym for logical record.

record layout. The arrangement and structure of data or words in a record including the order and size of the components of the record.

recursive. *(1)* Pertaining to a process in which each step makes use of the results of earlier steps. *(2)* Pertaining to a program or routine that calls itself after each run until it is interrupted or until a specified condition is met.

reentrant. The attribute of a program or routine that allows the same copy of a program or routine to be used concurrently by two or more tasks.

reentrant program. *(1)* A computer program that may be entered at any time before any prior execution of the program has been completed. *(2)* Synonymous with reenterable program.

reference. In programming languages, a language construct designating a declared language object.

register. A part of internal storage having a specified storage capacity and usually intended for a specific purpose.

relation. *(1)* The comparison of two expressions to see if the value of one is equal to, less than, or greater than the value of the other. *(2)* In COBOL, synonym for relational operator.

relation condition. In COBOL, the proposition, for which a truth value can be determined, that the value of an arithmetic expression, data item, nonnumeric literal, or index name has a specific relationship to the value of another arithmetic expression, data item, nonnumeric literal, or index-name.

release. A distribution of a new product or new function and APAR fixes for an existing product. Normally, programming support for the prior release is discontinued after some specified period of time following availability of a new release. The first version of a product is announced as Release 1, Modification Level 0.

replace. A function or mode that enables the user to substitute text for a specified part of previously entered text.

request. A directive, by means of a basic transmission unit, from an access method that causes the network control program to perform a data-transfer operation or auxiliary operation.

requirement. An essential condition that a system has to satisfy.

reserved word. *(1)* In programming languages, a keyword that may not be used as an identifier. *(2)* A word used in a source program to describe an action to be taken by the program or the compiler. It must not appear in the program as a user-defined name or a system name. *(3)* In COBOL, a COBOL word specified in the list of words that may be used in a COBOL source program, but that must not appear in the program as user-defined words or system-names.

resource. *(1)* Any facility of a computing system or operating system required by a job or task, and including main storage, input/output devices, processing unit, data sets, and control or processing programs. *(2)* In COBOL, a facility or service, controlled by the operating system, that can be used by an executing program.

result. An entity produced by the performance of an operation.

return. *(1)* Within a subroutine, to effect a link to the computer program that called the subroutine. *(2)* In programming languages, a language construct within a procedure designating an end of an execution sequence in the procedure.

return code. *(1)* A code used to influence the execution of succeeding instructions. *(2)* A value returned to a program to indicate the results of an operation requested by that program.

risk. The probability that a particular threat will exploit a particular vulnerability of the system.

routine. A program, or part of a program, that may have some general or frequent use.

row. A horizontal arrangement of characters or other expressions. *(2)* Contrast with column.

run. *(1)* A performance of one or more jobs. *(2)* A performance of one or more programs. *(3)* To cause a program, utility, or other machine function to be performed.

run time. Synonym for execution time.

run-time environment. In programming languages, a logical grouping of one or more program objects that must be connected at application run time to do some task.

save. *(1)* A function that enables a user to write into a file of a previously entered or modified text. *(2)* To retain data by copying it from main storage to another storage device.

scan. *(1)* To examine sequentially, part by part. *(2)* To briefly examine or read. See also browse, search. *(3)* To search records for a specified character string or syntax error.

scanning. The systematic examination of data.

scope. *(1)* The portion of an expression to which the operator is applied. *(2)* The portion of a computer program within which the definition of the variable remains unchanged.

screen. Deprecated term for display panel.

search. *(1)* A function or mode that enables the user to locate occurrences of such things as particular character strings, embedded commands, or boldface letters in a document. Synonymous with find. *(2)* The process of looking for a particular item. See also browse, scan. *(3)* To scan one or more data elements of a set in order to find elements that have a certain property.

search chain. In VSE, the order in which chained sublibraries are searched for the retrieval of a certain library member of a specified type.

section. *(1)* In a VSAM index record, a group of consecutive index entries. The index entries in an index record are divided into approximately as many sections as the square root of the number of entries in order to speed up a search for an entry. *(2)* In COBOL, a set of zero, one, or more than one paragraphs or entries, called a section body, the first of which is preceded by a section header. Each section consists of the section header and the related section body.

semantic error. A compile-time error caused by incorrect definition of constants and identifiers.

separately compiled program. In COBOL, a program that, together with its contained programs, is compiled separately from all other programs.

sequence. *(1)* A series of items that have been sequenced. *(2)* An arrangement of items according to a specified set of rules; for example, items arranged alphabetically, numerically, or chronologically. *(3)* To place items in an arrangement in accordance with the order of the natural numbers. *(4)* Synonym for collating sequence.

service. A customer- related or product-related business function such as design/manufacturing error correction, installation planning, maintenance, customer education, or programming assistance.

service routine. Synonym for utility routine.

session. The period of time during which a user of a terminal can communicate with an interactive system, usually, elapsed time between logon and logoff.

set. *(1)* A finite or infinite number of objects of any kind, of entities, or of concepts that have a given property or properties in common. *(2)* To cause a counter to take the state corresponding to a specified number. *(3)* To put all or part of a data processing device into a specified state.

share. To make a resource available to remote users or other processes.

shared. Pertaining to the availability of a resource for more than one use at the same time.

sign. See operational sign.

simultaneous. *(1)* Pertaining to the occurrence of two or more events at the same instant of time. *(2)* In a process, pertaining to two or more events that occur within the same interval of time, each one handled by a separate functional unit; for example, in the execution of one or more programs, several input/output operations handled by I/O equipments may be simultaneous with one another and with other operations handled directly by the processing unit.

softcopy. *(1)* A nonpermanent copy of the contents of storage in the form of a display image. *(2)* One or more files that can be electronically distributed, manipulated, and printed by a user. Contrast with hardcopy.

software. *(1)* All or part of the programs, procedures, rules, and associated documentation of a data processing system. Software is an intellectual creation that is independent of the medium on which it is recorded. *(2)* Contrast with hardware. *(3)* See application software, integrated software, system software.

sort. *(1)* The operation of sorting. *(2)* To arrange a set of items according to keys used as a basis for determining the sequence of the items; for example, to arrange the records of a personnel file in alphabetical sequence by using the employee names as sort keys. *(4)* Synonym for order.

source. *(1)* In advanced program-to-program communications, the system or program that starts jobs on another system. *(2)* A system, a program within a system, or a device that makes a request to a target. Contrast with target. *(3)* In COBOL, the symbolic identification of the originator of a transmission to a queue.

source code. The input to a compiler or assembler, written in a source language. Contrast with object code.

source program. *(1)* A program that a particular translator can accept. *(2)* A set of instructions written in a programming language that must be translated to machine language before the program can be run. *(3)* In COBOL, a syntactically correct set of COBOL statements. A COBOL source program commences with the Identification Division, a COPY statement, or a REPLACE statement. It is terminated by the end program header, if specified, or by the absence of additional source program lines. *(4)* Contrast with object program.

special register. In COBOL, a compiler-generated storage area whose primary use is to store information produced in conjunction with the use of specific COBOL feature.

SPECIAL-NAMES. In COBOL, the name of an Environment Division paragraph in which implementor-names are related to user-specified mnemonic-names.

specification. In system development, a description of how the design of a system, device, or program is to be implemented.

STAE (specify task abnormal exit). A macroinstruction that specifies a routine to receive control in the event of abnormal termination of the issuing task.

stand-alone. Pertaining to operation that is independent of any other device, program, or system.

standard label. A fixed-format record that identifies a volume of data such as a tape reel or a file that is part of a volume of data.

startup. See system startup.

statement. *(1)* In programming languages, a language construct that represents a step in a sequence of actions or a set of declarations. *(2)* In computer programming, a symbol string or other arrangement of symbols. *(3)* An instruction in a program or procedure. *(4)* In COBOL, a syntactically valid combination of words, literals, and separators, beginning with a verb, written in a COBOL source program. *(5)* A language syntactic unit consisting of an operator, or other statement identifier, followed by one or more operands.

static. *(1)* In programming languages, pertaining to properties that can be established before execution of a program; for example, the length of a fixed length variable is static. *(2)* Pertaining to an operation that occurs at a predetermined or fixed time. *(3)* Contrast with dynamic.

status. The condition or state of hardware or software, usually represented by a status code.

step. *(1)* One operation in a computer routine. *(2)* To cause a computer to execute operation.

storage. *(1)* A functional unit into which data can be placed, in which they can be retained and from which

they can be retrieved. *(2)* The action of placing data into a storage device. *(3)* A storage device.

storing. *(1)* The action of placing data into a storage device. *(2)* To place data into a storage device. *(3)* To retain data in a storage device.

string. *(1)* A sequence of elements of the same nature, such as characters considered as a whole. *(2)* In programming languages, the form of data used for storing and manipulating text. *(3)* In PL/I, a sequence of characters or bits that is treated as a single data item.

structure. *(1)* A variable that contains an ordered group of data objects. Unlike an array, the data objects within a structure can have varied data types. *(2)* In PL/I, a collection of data items that need not have identical attributes. Contrast with array. *(3)* In FORTRAN, an object of derived type.

structured programming. *(1)* A method for constructing programs using only hierarchically nested constructs each having a single entry and a single exit point. Three types of control flow are used in structured programming: sequential, conditional, and iterative. *(2)* A technique for organizing and coding programs that makes them easier to debug, modify, and replace.

> **Note:** Typically, a structured program is a hierarchy of modules that all have a single entry point and a single exit point. Control is passed downward through the structure without unconditional branches to higher levels of the structure.

sublibrary. In VSE, a subdivision of a library. See also library.

submit. In VSE, a function that passes a job to the system for processing.

subprogram. *(1)* A program invoked by another program. Contrast with main program. *(2)* In FORTRAN, a function subprogram, a subroutine subprogram, a module subprogram, or a block data subprogram. *(3)* In COBOL, synonym for called program.

subroutine. *(1)* A sequence of instructions whose execution is invoked by a call. *(2)* A sequenced set of instructions or statements that may be used in one or more computer programs and at one or more points in a computer program. *(3)* A group of instructions that can be part of another routine or can be called by another program or routine. *(4)* In PL/I, a procedure that has no RETURNS option in the PROCEDURE statement. Contrast with function. *(6)* In FORTRAN, a procedure that is invoked by a CALL statement or an assignment statement.

subroutine call. *(1)* The subroutine in object coding that performs the call function. *(2)* In a source program, a language construct that invokes a subroutine. *(3)* In PL/I, an entry reference that must represent a subroutine, followed by an optional and possibly empty argument list that appears in a CALL statement. Contrast with function reference.

subscript. *(1)* A symbol associated with the name of a set to identify a particular subset or element. *(2)* An integer or variable whose value selects a particular element in a table or array. *(3)* In COBOL, an occurrence number represented by either an integer, a data-name optionally followed by an integer with the operator + or -, or an index-name optionally followed by an integer with the operator + or -, which identifies a particular element in a table. *(4)* In FORTRAN, an item of a list of subscripts that selects an element of a named array or an array-valued structure component.

subset. *(1)* A set each element of which is an element of a specified other set. *(2)* A variant form of a programming language with fewer features or more restrictions than the original language. *(3)* A set contained within a set.

support. In system development, to provide the necessary resources for the correct operation of a functional unit.

switch. A device or programming technique for making a selection; for example, a toggle, a conditional jump.

symbol. *(1)* A representation of something by reason of relationship, association, or convention. *(2)* A name in a source document that can be replaced with something else, for example, a character string. *(3)* In S/390 operating systems, any group of eight or less alphanumeric and national characters that begins with an alphabetic character or the characters (@, #, $).

syntax. *(1)* The relationship among characters or groups of characters, independent of their meanings or the manner of their interpretation and use. *(2)* The rules governing the structure of a language. *(3)* The rules for the construction of a statement.

syntax error. A compile-time error caused by incorrect syntax. See also semantic error.

system. In data processing, a collection of people, machines, and methods organized to accomplish a set of specific functions.

system resources. Those resources controlled by the system, such as programs, devices, and storage areas that are assigned for use in jobs.

system software. *(1)* Application-independent software that supports the running of application software. *(2)* Software that is part of or made available with a computer system and that determines how application programs are run; for

example, an operating system. Contrast with application software.

system startup. Synonym for initial program load.

system termination. The state in which all processing on a system is stopped.

table. *(1)* A two-dimensional array in which each item and its position with respect to other items is identified. *(2)* An orderly arrangement of data in rows and columns that can contain numbers, text, or a combination of both. *(3)* In COBOL, a set of logically consecutive items of data that are defined in the Data Division of a COBOL program by means of the OCCURS clause.

tape (magnetic tape). A tape with a magnetizable layer on which data can be stored.

target. *(1)* Pertaining to a storage device to which information is written. *(2)* The program or system to which a request is sent. *(3)* The location to which the information is destined. *(4)* A system, program, or device that interprets, rejects or satisfies, and replies to requests received from a source.

target language. The output language of a translator.

term. *(1)* A construct in a conceptual schema language that refers to an entity. *(2)* The smallest part of an expression that can be assigned a value.

terminal. *(1)* A point in a system or communication network at which data can either enter or leave. *(2)* A device, usually equipped with a keyboard and display device, capable of sending and receiving information. *(3)* In COBOL, the originator of a transmission to a queue, or the receiver of a transmission from a queue.

terminate. *(1)* To stop the operation of a system or device. *(2)* To stop execution of a program.

termination. *(1)* The act of putting a system or an element of a system in a state in which it no longer performs its normal function. See also system termination. *(2)* Cessation of the execution of a task.

test plan. A plan that establishes detailed requirements, criteria, general methodology, responsibilities, and general planning for test and evaluation of a system.

testing. The running of a system or a program against a predetermined series of data to arrive at a predictable result for the purpose of establishing the acceptability of the system or program.

tool. Software that permits the development of an application program without using a traditional programming language.

trace. *(1)* A record of the execution of a computer program. It exhibits the sequences in which the instructions were executed. *(2)* The process of recording the sequence in which the statements in a program are executed and, optionally, the values of the program variables used in the statements. *(3)* To record a series of events as they occur.

transaction. *(1)* In a batch or remote batch entry, a job or job step. *(2)* An exchange between a workstation and another device that accomplishes a particular action or result. *(3)* A specific set of input data that triggers execution of a specific process or job; a message destined for an application program. *(4)* In CICS/VSE, one or more application programs that can be used by a display station operator. A given transaction can be used concurrently from one or more display stations. The execution of a transaction for a certain operator is also referred to as a task: a task can relate to only one operator.

transfer. *(1)* To send data from one place and receive the data at another place. Synonymous with move. *(2)* To read data from auxiliary storage or from an input device into processor storage or from processor storage to auxiliary storage or to an output device.

> **Note:** A transfer usually does not erase data from the original location.

transient. Pertaining to a program or subroutine that does not reside in main storage or to a temporary storage area for such a program.

transient data queue. A sequential data set used by the Folder Application Facility in CICS/VSE to log system messages.

translate. In programming languages, to transform all or part of a program expressed in one programming language, into another programming language or into a machine language suitable for execution.

tuning. The process of adjusting an application or a system to operate in a more efficient manner in the work environment of a particular installation.

type. A class of objects. All objects of a specific type can be accessed through one or more of the same interfaces.

update. *(1)* To add, change, or delete items. *(2)* To modify a master file with current information according to a specified procedure.

upward compatibility. The capability of a computer to execute programs written for another computer without major alteration, but not vice versa.

user. *(1)* Any person or any thing that may issue or receive commands and messages to or from the

information processing system. *(2)* Anyone who requires the services of a computing system.

user exit. *(1)* A point in an IBM-supplied program at which a user exit routine may be given control. *(2)* A programming service provided by an IBM software product that may be requested during the execution of an application program for the service of transferring control back to the application program upon the later occurrence of a user-specified event.

user ID. User identification.

userid. A string of characters that uniquely identifies a user to a system.

utility. The capability of a system, program, or device to perform the functions for which it is designed.

validation. The checking of data for correctness or for compliance with applicable standards, rules, and conventions.

value. A quantity assigned to a constant, a variable, parameter or a symbol. See also argument.

variable. *(1)* In programming languages, a language object that may take different values, one at a time. The values of a variable are usually restricted to a certain data type. *(2)* A name used to represent data whose value can be changed while the program is running by referring to the name of the variable. *(3)* In FORTRAN, a named storage location whose value can be changed while the program is running by referring to the name of the variable. *(4)* In COBOL, a data item whose value may be changed by execution of the object program. A variable used in an arithmetic expression must be a numeric elementary item.

verb. In COBOL, a word that expresses an action to be taken by a COBOL compiler or object program.

verification. The act of determining whether an operation has been accomplished correctly.

version. A separate IBM-licensed program, based on an existing IBM-licensed program, that usually has significant new code or new function. Each version has its own license, terms, conditions, product type number, monthly charge, documentation, test allowance (if applicable), and programming support category.

> **Note:** Numbering of versions starts with version 2. The first release of an IBM-licensed program is referred to as Release 1 with no indication of version number.

viewpoint. In computer graphics, the origin from which angles and scales are used to map virtual space into display space.

virtual. Pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means.

virtual storage. The storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of auxiliary storage available, not by the actual number of main storage locations.

warning message. An indication that a possible error has been detected. Contrast with error message.

window. A portion of a display surface in which display images pertaining to a particular application can be presented. Different applications can be displayed simultaneously in different windows.

word. *(1)* A character string or a bit string considered as an entity. *(2)* In COBOL, a character-string of not more than 30 characters that forms a user-defined word, a system-name, or a reserved word.

workstation. *(1)* A functional unit at which a user works. A workstation often has some processing capability. *(2)* One or more programmable or nonprogrammable devices that allow a user to do work. *(3)* A terminal or PC, usually one that is connected to a mainframe or to a network, at which a user can perform applications.

write. To make a permanent or transient recording of data in a storage device or on a data medium.

writing. The action of making a permanent or transient recording of data in a storage device or on a data medium.

zero. In data processing, the number that, when added to or subtracted from any other number, does not alter the value of the other number. Zero may have different representations in computers, such as positively or negatively signed zero (which may result from subtracting a signed number from itself) and floating-point zero (in which the fixed point part is zero while the exponent in the floating-point representation may vary).

List of Abbreviations

ACB	Access control block
AFP	Advanced Function Printer
AMODE	Addressing MODE
ANSI	American National Standards Institute
APAR	Authorized Program Analysis Report
API	Application Program Interface
AR	Attention Routine
ASC	Address Space Control
ASI	Automated System Initialization
ATMS	Advanced Text Management System
BAM	Basic Access Method
BIT	Binary Digit
BLL	Base Locator for Linkage section
BSC	Binary Synchronous Communication
BTAM	Basic Telecommunications Access Method
CCCA	COBOL CICS Conversion Aid
CICS	customer information control system
CMS	Conversational Monitor System
COBOL	COmmon Business Oriented Language
COMREG	Communication region
CPU	Central Processing Unit
CSECT	Control SECTion
CSP	Cross-System Product
DASD	Direct Access Storage Device
DB2	DATABASE 2
DBCS	Double Byte Character Set
DCF	Document Composition Facility
DD	Data Definition
DL/I	Data Language 1
DLBL	Disk Label
DOS	Disk Operating System
DOS/VS	Disk Operating System/Virtual Storage
DSECT	Dummy control SECTion
DTF	Define The File
EC	Engineering Change
EE	Execution Element
EIB	Error Information Block
EOJ	End Of job
ESA	Enterprise Systems Architecture
ESDS	Entry Sequenced Data Set
EBA	Fixed/Block-Architecture
FD	File Definition
FSU	Field Serviceable Unit
GDDM	Graphical Data Display Manager
GUI	Graphical User Interface
HLL	High Level Language
I/O	Input/Output
IBM	International Business Machines
ICA	Integrated Communication Adapter
IDCAMS	Program name for Access Method Services
II	Interactive Interface
ILC	Inter Language Communication
IPL	Initial Program Load
ISAM	Indexed Sequential Access Method
ISQL	Interactive Structured Query Language
ITSO	International Technical Support Organization
JCL	job control language
KSDS	Key Sequenced Data Set
LCP	Language Conversion Program
LE	Language Environment
LE/370	Language Environment/370
LU	Logical Unit
MPS	Multiple Partition Support
MRI	Machine Readable Information

MSHP	Maintain System History Program
MVS	Multiple Virtual Storage
MVS/ESA	Multiple Virtual Storage/Enterprise Systems Architecture
NETVIEW	Network observation tool
NJE	Network Job Entry
NLDM	Network Logical Data Manager
NLF	NEWS Library Files
NLS	National Language Support
NPDA	Network Problem Determination Aid
NPSI	NCP Packet Switching Interface
OEM	Original Equipment Manufacturer
OS/2	Operating System/2
OS/VS	Operating System/Virtual Storage
PC	Personal computer
PCB	Pool control block.
PL/I	programming language 1
PNET	Power NETworking interface
PR/SM	Processor Resource/Systems Manager
PSB	Program Status Block
PSF/VSE	Print Services Facility/Virtual Storage Extended
PTF	Program Temporary Fix
RC	Return Code
RCF	Report Controller Feature
REXX	REstructured eXtended eXecutor language
RMODE	Residency MODE
RPG	Report Program Generator
RPG II	Report Program Generator II
RRDS	Relative Record Data Set II
SAM	Sequential Access Method
SDAID	System Debugging AIDs
SDF/CICS	Screen Definition Facility/Customer Information Control System
SDL	System Directory List
SLI	source Library Inclusion
SMF	Source Macro File
SNA	Systems Network Architecture
SQL	Structured Query Language
SQL/DS	Structured Query Language/Data System
SVA	Shared Virtual Area
SVC	SuperVisor Call Instruction
SYSLST	System List Device
SYSPCH	System Punch Device
TOD	Time Of Day
TP	Teleprocessing
VM	Virtual Machine
VM/ESA	Virtual Machine/Enterprise Systems Architecture
VSAM	Virtual Storage Access Method
VSE	Virtual Storage Extended
VSE/ESA	Virtual Storage Extended/Enterprise Systems Architecture
VSE/ICCF	Virtual Storage Extended/Interactive Computing and Control Facility
VSE/POWER	Virtual Storage Extended/Priority Output Writers, Execution processor, and input Readers
VSE/SP	Virtual Storage Extended/System Package
VSE/VSAM	Virtual Storage Extended/Virtual Storage Access Method
VTAM	Virtual Telecommunications Access Method
VTOC	Volume Table Of Contents

Index

Numerics

A

B

C

D

E

F

G

I

K

L

M

ITSO Redbook Evaluation

IBM COBOL and Language Environment for VSE/ESA How to Upgrade Now
SG24-4277-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@vnet.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

Printed in U.S.A.