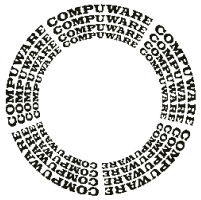
STROBE MVS

COOL:GEN Feature

Release 3.0



COMPUWARE®

Please direct questions about STROBE MVS
or comments on this document to:

STROBE MVS Technical Support
Compuware Corporation
124 Mount Auburn Street
Cambridge MA 02138-5758
1-800-585-2802 or
1-617-661-3020
1-617-498-4010 (fax)
strobe-sup@compuware.com

Outside the USA and Canada, please contact
your local Compuware office or agent.

Doc. CWSTUG3A
February 11, 2003.

Contents

Figures

Summary of Changes

This section discusses the functional changes to the STROBE COOL:Gen Feature, as well as the changes to this manual, from STROBE Version 2 Release 5.0.

Changes to the STROBE COOL:Gen Feature

There are no changes to the STROBE COOL:Gen Feature for STROBE MVS for Sysplex Release 3.0.

In iSTROBE online help, the guidelines and suggestions for improving performance of COOL:Gen applications have been updated and expanded.

Changes for this Manual

The list of supported releases of COOL:Gen has been updated.

The examples of performance analysis have been updated to use iSTROBE instead of APMpower.

Introduction

This manual describes measurement concepts applicable to and specific data made available by the STROBE COOL:Gen Feature of the STROBE MVS Application Performance Measurement System. The STROBE COOL:Gen Feature augments functions provided by the basic STROBE system.

The STROBE MVS Application Performance Measurement System, iSTROBE, and the STROBE COOL:Gen Feature are products designed for measurement and analysis of applications developed with one of the following:

- IEF (Release 5.2)
- Composer by IEF (Release 5.3)
- Composer 3 (Release 5.3.2) Composer 4, Composer 5
- COOL:Gen 4.1a, COOL:Gen 4.1.1, COOL:Gen 5.0, COOL:Gen 5.1, COOL:Gen 6.0

and executed in an MVS environment.

This manual uses the term COOL:Gen to refer to IEF, Composer, Composer, COOL:Gen, and Advantage GEN applications.

How This Manual Is Organized

Chapter 1, “Overview” presents an overview of the STROBE COOL:Gen Feature and the COOL:Gen application development environment.

Chapter 2, “Using the STROBE COOL:Gen Feature” describes how to use the STROBE COOL:Gen Feature to measure a COOL:Gen application, monitor the measurement request, and create a Performance Profile.

Chapter 3, “The iSTROBE Performance Profile for a COOL:Gen Application” explains how to interpret the Performance Profile reports using iSTROBE.

Chapter 4, “The STROBE Performance Profile for a COOL:Gen Application” explains how to interpret the Performance Profile reports created with STROBE/ISPF.

Appendix A, “The STROBE COOL:Gen Report Reference” presents a description of the STROBE Performance Profile reports that are specific to the STROBE COOL:Gen Feature.

How to Use This Manual

You should read Chapter 1, “Overview” and Chapter 2, “Using the STROBE COOL:Gen Feature” before you submit a measurement request. If you want to interpret a Performance Profile using iSTROBE, read Chapter 3, “The iSTROBE Performance Profile for a COOL:Gen Application”. If you want to interpret a STROBE Performance Profile created with STROBE/ISPF, read Chapter 4, “The STROBE Performance Profile for a COOL:Gen Application”. For a description of the STROBE Performance Profile reports specific to the STROBE COOL:Gen Feature, see Appendix A, “The STROBE COOL:Gen Report Reference”.

The STROBE Library

The STROBE base product manuals include:

- *STROBE MVS Concepts and Facilities*, document number CWSTGX3A

  STROBE MVS Concepts and Facilities explains how to decide which programs and online regions to measure, when to measure them, and how to interpret the reports in the STROBE Performance Profile.

- *STROBE MVS Messages,* document number CWSTXM3A

  STROBE MVS Messages lists all messages and abnormal termination (ABEND) codes, describes how to interpret them, and in many cases suggests a corrective action.

- *STROBE MVS System Programmer's Guide*, document number CWSTXI3A

  The *STROBE MVS System Programmer's Guide* explains how to install and maintain STROBE.

- *STROBE MVS User's Guide,* document number CWSTUX3A and the *STROBE MVS User's Guide with Advanced Session Management*, document number CWSTUA3A

  The *STROBE MVS User's Guide* explains how to use STROBE to measure application performance. The *STROBE MVS User's Guide with Advanced Session Management* explains how to use STROBE with the STROBE Advanced Session Management Feature to measure application performance. Users who have the STROBE Advanced Session Management Feature will use this manual rather than the *STROBE MVS User's Guide*.

- *STROBE MVS Application Performance Measurement System Quick Reference*

  The *STROBE MVS Application Performance Measurement System Quick Reference* is a convenient reference for how to use STROBE and for interpreting the STROBE Performance Profile.

STROBE Feature Manuals

These manuals describe the optional features of the STROBE MVS Application Performance Measurement System. Each manual describes measurement concepts applicable to and specific data made available by the feature.

- *STROBE MVS User's Guide with Advanced Session Management*, document number CWSTUA3A
- *STROBE ADABAS/NATURAL Feature*, document number CWSTUN3A
- *STROBE CA-IDMS Feature*, document number CWSTUR3A
- *STROBE CICS Feature*, document number CWSTUC3A
- *STROBE COOL:Gen Feature*, document number CWSTUG3A
- *STROBE CSP Feature*, document number CWSTUP3A
- *STROBE DB2 Feature,* document number CWSTUD3A
- *STROBE IMS Feature,* document number CWSTUI3A
- *STROBE Interface Feature*, document number CWSTUF3A
- *STROBE Java Feature,* document number CWSTUJ3A
- *STROBE MQSeries Feature,* document number CWSTUM3A
- *STROBE UNIX System Services Feature,* document number CWSTUU3A

Online Documentation

STROBE manuals are available in HTML, Adobe Acrobat PDF format, and IBM BookManager format, on CD-ROM and at Compuware's technical support Web site at http://frontline.compuware.com.

Online Help

STROBE products provide the following online information:

- STROBE/ISPF Online Tutorials, Option T from the STROBE/ISPF STROBE OPTIONS menu
- STROBE/ISPF Online Message Facility, Option M from the STROBE/ISPF STROBE OPTIONS menu

Other Compuware Application Performance Management Products

The following products and features work in conjunction with the STROBE MVS Application Performance Measurement System. These tools extend the benefits of application performance management (APM).

iSTROBE

iSTROBE enables you to view and analyze STROBE Performance Profile data on a workstation using a standard Web browser. Easy to install and easy to use, iSTROBE guides you through the performance analysis process and offers recommendations for improving performance. iSTROBE simplifies the performance analysis of applications that you measure with STROBE. For more information on iSTROBE, see the *iSTROBE Getting Started Guide*.

SQL Analysis Feature

The SQL Analysis Feature works in conjunction with STROBE and iSTROBE or APMpower to supply access path analyses and database and SQL coding recommendations for DB2 applications measured by STROBE. The SQL Analysis Feature pinpoints the most resource-consumptive static or dynamic SQL statements, explains why these statements might be inefficient, and provides recommendations to improve the performance of the DB2 application. For more information on the SQL Analysis Feature, see the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management*.

APMpower

The APMpower Application Performance Analysis System extends the benefits of STROBE to application developers who use workstations to develop, test, and maintain MVS applications. Developers employ the APMpower graphical user interface and advanced analytical aids to navigate the Performance Profile, analyze and improve application performance, and share performance knowledge across the IS organization. For more information about APMpower, see the APMpower documentation.

Compuware APM Technical Support

For North American customers, for technical support, please contact the Technical Support department by telephone at (800) 585-2802 or (617) 661-3020, by fax at (617) 498-4010, or by e-mail at strobe-sup@compuware.com.

To access online technical support, visit Compuware's FrontLine page on the World Wide Web at http://frontline.compuware.com and select the product "STROBE and APMpower."

For other international customers, please contact your local Compuware office or STROBE supplier.

Compuware APM Training

Compuware's Education Resources Group offers a range of training options for organizations that use STROBE, iSTROBE, and APMpower. To arrange Application Performance Management training, please contact Compuware at 1-800-835-3190 or visit Compuware's Education Resources Group at http://www.compuware.com/training

For other international customers, please contact your local Compuware office or STROBE supplier for a complete list of APM Training offerings.

Compuware APM Service Offerings

For North American customers, for information about current service offerings, please contact your local Compuware sales office or call Compuware Corporate Headquarters at 1-800-COMPUWARE (266-7892) or visit Compuware's APM Product page on the World Wide Web at http://www.compuware.com/products/strobe.

For other international customers, please contact your local Compuware office or STROBE supplier for a complete list of Services offerings.

APM Installation Assurance

The APM Installation Assurance service assists you in planning for, installing, customizing and using APM products. The service will help you maximize the value and benefits derived from the APM product family.

Consulting engineers work closely with your IT personnel to understand your operating environment and your organization's APM goals. The engineer will assist you in developing a customization and installation plan for STROBE, iSTROBE, and APMpower. The engineer will oversee the installation process and verify product readiness. The engineer will also help set up measurement request schedules, request groups, history records, AutoSTROBE measurement requests, and will verify the installation of the SQL Analysis Feature.

With APM Installation Assurance services, your organization can immediately maximize the value received from your investment in the APM product family. You will also benefit from a fully customized installation that will enhance the product functionality and increase the automation aspects of your APM initiatives.

Application Performance Management Consulting

The Application Performance Management (APM) Consulting services assist you in identifying and resolving specific performance problems in your OS/390 business-critical applications.

Using STROBE, iSTROBE, and APMpower, consulting engineers work closely with your IT personnel to measure an application's performance, identify performance improvement opportunities and make recommendations for implementing solutions.

With APM Consulting services, your organization cannot only resolve problems quickly and effectively, but also gain the skills necessary to prevent application performance degradation in the future.

Application Performance Assessment

The Application Performance Assessment (APA) service assists you in achieving a higher level of performance for your OS/390 business-critical applications.

Using STROBE, iSTROBE, and APMpower, consulting engineers work closely with your IT personnel to evaluate the efficiency of business-critical applications, identify opportunities for improving performance and document the potential savings that can result from implementing recommended solutions.

With APA services, you cannot only improve application performance quickly and effectively, but also gain the knowledge and skills necessary to implement and sustain a process-oriented application performance management (APM) program.

Chapter 1. Overview

The STROBE MVS Application Performance Measurement System is a product that determines where and how time is spent in online regions and batch processing programs. STROBE produces reports that help you determine how to revise applications to improve their performance. This collection of reports is called a Performance Profile. iSTROBE is a Web browser-based product that enables you to analyze your Performance Profile interactively from your workstation.

The STROBE COOL:Gen Feature extends the functions of STROBE and is designed to measure applications created with Computer Associates' COOL:Gen/Advantage GEN product family. For a list of supported releases, see the Introduction to this manual.

One of the strengths of creating applications with COOL:Gen, or with any product that generates application source code, is that application developers can remove themselves from much of the detailed programming associated with languages like COBOL. This high-level interface allows application developers without an extensive mainframe programming background to generate applications for environments such as MVS.

However, COOL:Gen can also create a problem that is usually not apparent until the application reaches the integration test phase, or worse yet, until the application is already in production. The potential problem? An application containing generated code that is inefficient or unresponsive. It is also difficult to locate the source of performance problems within the generated code.

The STROBE COOL:Gen Feature helps you measure, analyze, and improve the performance of COOL:Gen applications, enabling you to develop and maintain more efficient and responsive applications throughout the application lifecycle. The STROBE COOL:Gen Feature requires that you have the STROBE DB2 Feature and the STROBE COBOL Feature. If you are running in an IMS or CICS online environment, either the STROBE CICS Feature or the STROBE IMS Feature is also required.

The remainder of this chapter provides a brief overview of the COOL:Gen application development environment, describes the benefits of the STROBE COOL:Gen Feature, and offers some background information about using STROBE and iSTROBE.

Overview of the COOL:Gen Application Environment

COOL:Gen is an application development environment that constructs executable code based on business models. The COOL:Gen product consists of four sets of tools: the Planning, Analysis, Design, and Construction (PADC) Toolsets. All of these toolsets reside on the workstation and are used to create a business model. Of the four, only the Construction Toolset can reside in the MVS mainframe environment. Once the business model is created on the workstation, the Construction Toolset is used to generate the components of a system (for example, the programs, databases, screen or window designs, and transaction definitions).

The STROBE COOL:Gen Feature supports applications created with COOL:Gen, Composer, or IEF and executed in an MVS environment. The following sections briefly describe the COOL:Gen application development environment.

Developing COOL:Gen Applications

COOL:Gen enables you to define, test, and generate applications for execution in the following MVS environments: CICS, IMS, TSO, or batch.

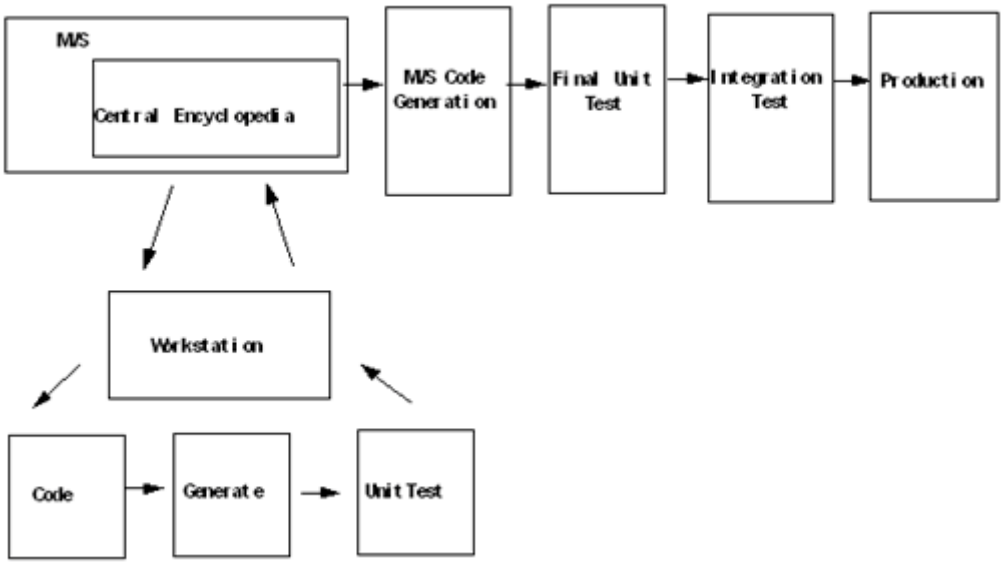
COOL:Gen applications are developed by using the PADC toolsets to create a business model (a graphical representation of an enterprise), the different business systems that comprise the enterprise, and the applications that comprise the business systems. At the heart of the business system is the *action diagram*. An action diagram is a set of related activities that define the logic of a procedure or action block. Many action diagrams can be combined in a dialog design to form a business system.

As shown in Figure 1-1, COOL:Gen stores these business models in a central repository called an encyclopedia. COOL:Gen uses two types of encyclopedias:

- Central Encyclopedia (on the host)
- Client/Server Encyclopedia

Once stored in an encyclopedia, a business model can be checked out of the encyclopedia so that the application developer can make necessary changes to the application. The model is then checked back to the encyclopedia and used to generate a COOL:Gen application for a specific environment.

Figure 1-1. Typical Development Cycle for COOL:Gen Applications



Executing COOL:Gen Applications in an MVS Environment

To create COOL:Gen applications for the MVS environment, the application programmer uses the Construction Toolset to generate COBOL source code. The generation process creates the procedures that precompile, compile, link, and bind the application. The generation process also creates the JCL or CLISTs used to execute the transactions.

COOL:Gen construction libraries are used to prepare and execute applications generated with the COOL:Gen Construction Toolset. COOL:Gen applications are prepared (compiled and linked) to execute in MVS environments. The compiled and linked modules are automatically stored in a standard load library as a part of the installation process. Once installed, the application then runs as a standard job under a specific target environment, such as CICS, IMS, TSO, or batch.

Benefits of the STROBE COOL:Gen Feature

The STROBE COOL:Gen Feature collects application performance information as the generated application executes. When execution completes, this information is organized into a *Performance Profile*, a series of reports that show, by action diagram and action diagram statement number, where and how time is spent during application execution, pinpointing possible problem areas. Measuring your application at successive stages of the development cycle can help you evaluate and improve your code before the applications are moved into production.

In fact, managing the performance of COOL:Gen-generated applications with the STROBE COOL:Gen Feature yields benefits in all phases of the application's lifecycle. The following sections describe these benefits.

During Production

Measuring your production applications with the STROBE COOL:Gen Feature enables you to identify areas of the application to evaluate for improvement. You can then evaluate these performance improvement opportunities, determine the changes to make, and then implement these changes.

Once these changes are implemented, and the application's performance improves, the possible benefits include:

- fewer occurrences of production-level performance crises
- improved efficiency of the application
- reduced batch processing time and costs
- acceptable online response time levels
- capacity is made available for processing other applications.
- access to the COOL:Gen help file for performance hints and tips stored in iSTROBE's online knowledge base.

The STROBE COOL:Gen Feature is integral to maintaining the performance of production applications. You gain greater benefit, however, if you identify and eliminate potential performance problems before they reach the production environment.

During Integration Test

The main benefit of using the STROBE COOL:Gen Feature to measure applications in the test environment is readily apparent: the ability to identify areas of the application to evaluate for performance improvement *before* the application goes into production.

Managing the performance of the application in this phase of development helps you to:

- deliver efficient applications that meet or exceed the expectations of your end users
- develop performance benchmarks for your COOL:Gen applications from the information you gather during the testing cycle
- avoid the costs associated with introducing inefficient applications into production.
- access to the COOL:Gen help file for performance hints and tips stored in the iSTROBE knowledge base.

During Earlier Test Cycles

Using the STROBE COOL:Gen Feature during earlier testing phases, such as unit test, yields subtle, yet significant, benefits. As soon as testing begins on the host, you can begin to measure the application and potentially realize many of the same benefits described in the previous section.

Over time the benefits of using the feature during the other phases of development flow into these early testing phases. These benefits can include:

- the availability of a repository of performance improvement information collected during other development phases
- an understanding of how certain COOL:Gen design and coding practices affect performance
- the ability to apply the COOL:Gen help file's performance hints and tips stored in iSTROBE's online help.

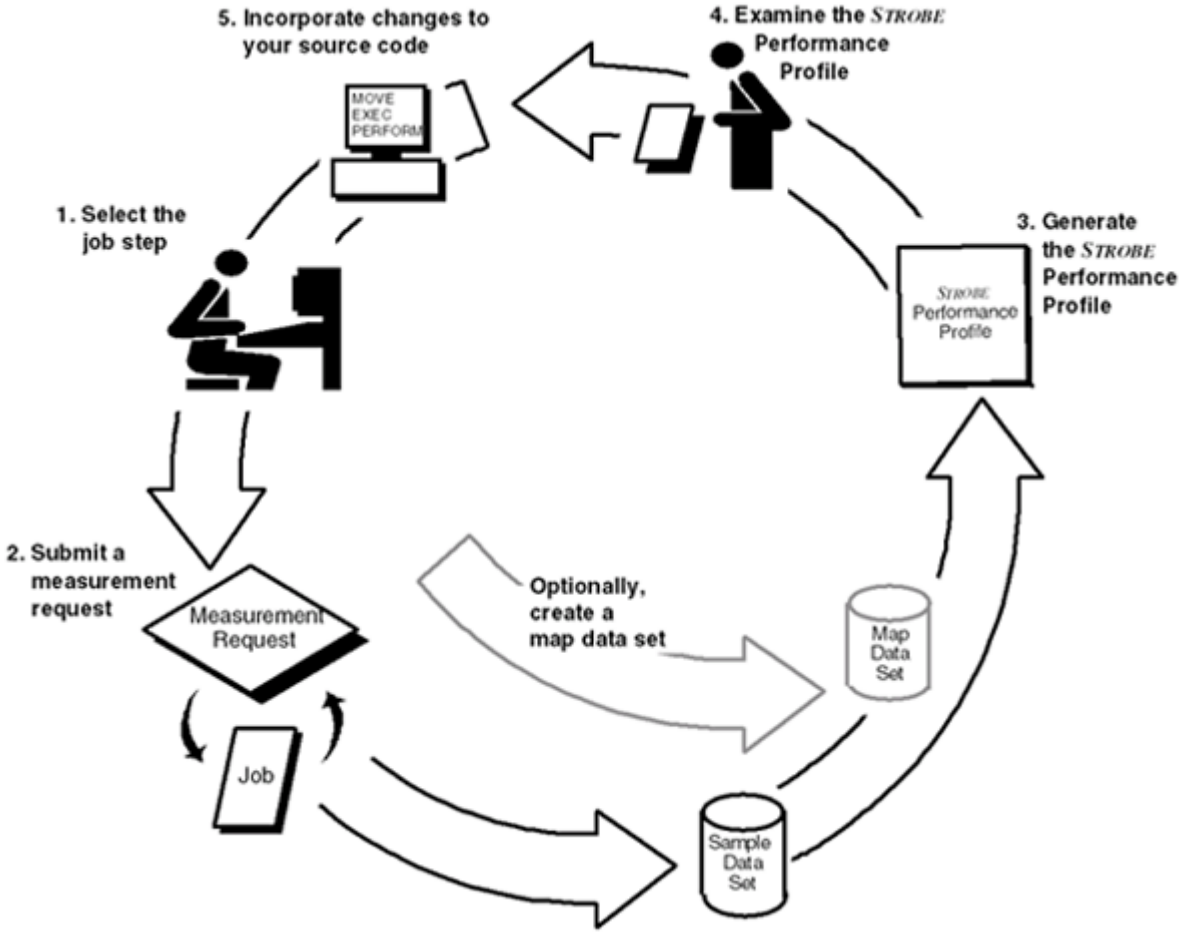
Over time, you can use the knowledge obtained from regular application performance management in test and production to systematically build efficiency into your COOL:Gen applications as you develop them.

The next section discusses the key concepts and terms that are central to the use of the STROBE COOL:Gen Feature.

Concepts and Terminology

Before using the STROBE COOL:Gen Feature, you need to become familiar with the basic concepts and terminology specific to STROBE and iSTROBE. The following sections introduce you to these concepts and terms.

Figure 1-2. Using the STROBE COOL:Gen Feature



What is a Measurement Request?

A *measurement request* specifies the parameters for measuring the performance of an application. As soon as the application is active, STROBE begins a *measurement session*, as shown in Figure 1-2, an interval during which STROBE collects performance data about the application while it executes.

Unlike tracing or monitoring products, STROBE employs a sampling technique that executes within the address space of the target application and periodically takes snapshots of the application's execution.

A measurement request can initiate one or more measurement sessions. During each session, STROBE collects and stores measurement data in a *sample data set*, a file that contains the information collected during a single measurement session. Each measurement session corresponds to one sample data set.

What is a Performance Profile?

After the sample data set is closed, you can use it to create the Performance Profile, a hierarchical series of reports that present the performance data collected during a measurement session. These reports show where and how time is spent during application execution.

For COOL:Gen applications, a Performance Profile contains easy-to-read reports that identify CPU use by specific action diagrams and by statement numbers within the action diagrams. The reports also identify wait time caused by specific files and devices.

By interpreting the report data, you can identify where the application's demand for resources is concentrated and determine where to make changes to improve the performance of the application. Among other performance improvement opportunities, the Performance Profile highlights:

- high concentrations of CPU activity
- excessive wait time
- how your application invokes system service routines

Based on the information provided in the Performance Profile, you may decide to change the way your application uses resources or rewrite the action diagram so that certain statements are executed only when necessary. Once you have decided on a course of action, implement your changes, test, and measure the application performance again.

What is a Map Data Set?

The process of relating activity and wait to source code statement text is called *indexing*. The output of the indexing process is a *map data set*. A map data set is the repository for the information STROBE uses to relate offsets and addresses in the generated COBOL code with COOL:Gen statements in the action diagram. If you supply a map data set when you create the Performance Profile, the resulting reports relate activity to your application's statement numbers *and* provide the actual statement text. Indexing is optional, but very helpful for pinpointing where to examine your source code. For information on how to create the COBOL and COOL:Gen map data sets, see "Creating the Compiler Listing Data Set" on page 2-3.

In addition, STROBE must obtain CSECT information for any load module you want to index. For detailed information about this process, see the section in the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management*, titled "Entering Module Mapping Specifications". For information particular to COOL:Gen applications, see "Measuring the Application" on page 2-1.

Note: To display the COOL:Gen statement number and the activity (CPU or Wait time) associated with the statement on the iSTROBE and STROBE reports, the Composer 4 Design dialog option "Execute Statement Number" must be set to "ON". This restriction applies only to Composer 4 and COOL:Gen users. The default for this option is "ON". If the Execute Statement Number option is set to "OFF", STROBE will not obtain the IEF statement number and all activity is assigned to statement number zero.

You may want to:

- Measure your application and create a Performance Profile
- Review the Profile to identify action diagrams which display significant resource use
- Create map data sets for each of those action diagrams
- Create the Performance Profile again, including the map data sets.
- Review the Performance Profile first to identify action diagrams that display a significant amount of CPU activity, and then create map data sets for each of those action diagrams.

Where to Find More Information

The following table lists the chapters, sections, or other publications that provide information for specific tasks.

For information on	Refer to
Creating and submitting measurement requests	*STROBE MVS User's Guide* or *STROBE MVS User's Guide with Advanced Session Management*
Creating and submitting measurement requests for COOL:Gen applications	"Measuring the Application" on page 2-1.
Creating Performance Profiles	*STROBE MVS User's Guide* or *STROBE MVS User's Guide with Advanced Session Management*
Creating map data sets	*STROBE MVS User's Guide* or *STROBE MVS User's Guide with Advanced Session Management*
Analyzing a Performance Profile with iSTROBE	Chapter 2, "Using the STROBE COOL:Gen Feature"
Analyzing a Performance Profile with STROBE/ISPF	Chapter 3, "The iSTROBE Performance Profile for a COOL:Gen Application"
Learning more about iSTROBE	The iSTROBE *Getting Started* guide and the iSTROBE online help
Learning more about STROBE Performance Profiles	*STROBE MVS Concepts and Facilities*
Improving the performance of COOL:Gen applications	The online COOL:Gen Hints, available through iSTROBE online help; see "Using the COOL:Gen Online Hints" on page 3-14.

Chapter 2.
Using the STROBE COOL:Gen Feature

After your COOL:Gen application has been installed in the MVS environment, you can measure its performance.

The *STROBE MVS User's Guide* and the *STROBE MVS User's Guide with Advanced Session Management* contain comprehensive instructions for

- submitting and managing measurement requests and creating Performance Profiles
- preparing Performance Profiles for iSTROBE and downloading them to the workstation for analysis.

This chapter describes only those requirements and considerations that are particular to creating Performance Profiles for COOL:Gen applications.

Before You Begin

Before you submit your measurement request, you should know:

- the name of the application and where it executes, according to the following table:

Environment	Determine
batch	The job name and program name, or step name
CICS	The name of the CICS region and the load module names of the applications
IMS	The name of the IMS region and the load module names of the applications
TSO	The TSO session name and the load module names of the application

- if an online region, the time the application will execute, and approximately how long the application runs
- the name of the host data set library where the application load modules are stored (required for the TSO environment)
- the name of the compiler listing data set for the COOL:Gen application (this information is vital to producing reports that contain the maximum amount of information)

If you are unsure of any of the above information, see your STROBE system programmer.

Measuring the Application

When you submit the STROBE measurement request to measure a COOL:Gen application, you need to provide STROBE with some information. From the panel at which you are adding the request, enter "Y" in the MODULE MAPPING DATA field. STROBE displays the STROBE - MODULE MAPPING panel, as shown in Figure 2-1.

Figure 2-1. STROBE - MODULE MAPPING Panel showing COOL:Gen Parameters

```
 ---------------------- STROBE - MODULE MAPPING --------------------------------
COMMAND ===>

ADDITIONAL MEASUREMENT INFORMATION FOR JOBNAME:  SAJVSTST


LIBRARIES TO SEARCH to get module mapping data:
     ===> wpa.ief53.ivp.tso.exeload
     ===>
     ===>

 DETAIL BASELINE   ===> 00           (Minimum percent of time spent executing
                                       a module for it to be mapped)

 BASELINE OVERRIDE ===>           ===>          (Names of load modules to be
                   ===>           ===>           mapped without regard to the
                   ===>           ===>           BASELINE restriction)

 SVC NUMBERS  ===> 19,20      ===>             (Specify SVCs--a number or range
              ===>            ===>              of numbers (e.g., 1-5)--whose
                                                modules are to be identified)
```

What you enter on the STROBE - MODULE MAPPING panel depends on the environment in which the application runs. Enter the appropriate information according to your environment to see detailed level information for action diagrams:

If environment is	Enter	In this field
batch	Name of MVS load library where application load modules reside	**LIBRARIES TO SEARCH**
	The load module names of the application	**BASELINE OVERRIDE**
CICS	Load module names of the applications	**BASELINE OVERRIDE**
IMS	Load module names of the applications	**BASELINE OVERRIDE**
TSO	Name of the MVS load library where the application load modules reside	**LIBRARIES TO SEARCH**
	Load module names of the application	**BASELINE OVERRIDE**

Note: You must specify load module names to see detailed level information for action diagrams. If you are unsure of the load module names you can submit the measurement request without specifying the names. Next, create the Performance Profile, and then review the Performance Profile, noting the load module names for the action diagrams that are consuming the most resources. For detailed information about this process, see the section in the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management* titled "Entering Module Mapping Specifications".

Creating a Profile That Relates CPU Activity with COOL:Gen Statement Text

By default, COOL:Gen reports associate the CPU execution time observed during the measurement session with specific COOL:Gen statement numbers.

You can also view the COOL:Gen statement text and associate the generated COBOL statements with the related COOL:Gen statement numbers in the action diagram. The process of creating this association is called *indexing*. This step is optional, but you may find it highly useful.

To create an indexed Performance Profile (one that includes COOL:Gen statement text and, optionally, the related COBOL statement text), you first need to create a map data set. A *map data set* is the repository for the information required to relate addresses in the generated code with COOL:Gen statements in the source program. STROBE must have obtained CSECT information for any load module you want to index. For detailed information about this process, see the section in the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management* titled "Entering Module Mapping Specifications".

Note: To see statement-specific information for an action diagram, you must provide a compiler listing data set for that action diagram and then produce a map data set from the listing. Since you may not need this level of detail for all action diagrams, you can first create a Performance Profile without providing any map data sets as input. You can then review the Performance Profile to determine the action diagrams for which you want to see statement-specific information. You need to create map data sets only for the action diagrams in which you are interested.

The following sections describe how to create a COBOL compiler listing data set with the appropriate compile options, and how to create the COBOL and COOL:Gen map data sets.

Creating the Compiler Listing Data Set

If you are interested in seeing only action diagram information and statement text in the Performance Profile, then you can use a compiler listing data set in any format as input to create the map data set. If you want to do this, skip this section and proceed to "Creating the Map Data Set" on page 2-5.

However, if you are also interested in viewing COBOL statement and line number information, you must provide a COBOL compiler listing data set with the appropriate compile options.

These steps describe how to prepare a COBOL compiler listing data set using Composer. The exact process may be vary for different releases of COOL:Gen.

1. Specify the member name for the compiler listing data set. Start from the Composer workstation and follow these steps:
   a. Click **Packaging** from the **Construction** pull-down menu.
   b. Click **Open** from the **Diagram** pull-down menu, and select the appropriate type of packaging for the application (online, batch, or window). Composer displays a list of action diagrams for your application.
   c. Click the dialog manager that contains the action diagram.
   d. Click **Expand all** from the **View** pull-down menu.
   e. Click the action diagram for which you want to save the listing data set.
   f. Click **Properties** from the **Detail** pull-down menu.

g. Specify the member name for the compiler listing data set in the **Source** field of the **Names** group box. If you do not specify a name, Composer generates a name for you.

2. Modify the Composer scripts so that the appropriate type of COBOL compiler listing data set is saved (specify the COBOL II compilation options of SOURCE, LIST, and NOOFFSET). For instructions on modifying the scripts, see "Modifying Local Compile Scripts (MVS Host Encyclopedia)" on page 2-4 or "Modifying Remote Compile Scripts (MVS Implementation Toolset)" on page 2-5.

   The location and type of script you need to modify depends on the way you installed the Composer application. If you installed the application using the MVS Host Encyclopedia, modify the local compile scripts. If you installed the application using the MVS Implementation (IP) Toolset, modify the remote compile scripts. If you do not have access to the COOL:Gen scripts, contact your COOL:Gen system programmer.

3. Reinstall the Composer application to produce the appropriate COBOL compiler listing data set. After you have reinstalled the application, note the name of the COBOL compiler listing data set. This data set is required to create the map data set, as described in the following section.

Modifying Local Compile Scripts (MVS Host Encyclopedia)

The local compile script for COOL:Gen applications is stored in the COOL:Gen installation CLIST library. The CLIST name for the compile script is TICCMPL.

Note: Modifying this script is a global change for all users. When you modify the COOL:Gen script, you are modifying it for everyone.

To modify the compile script, follow these steps:

1. Locate the CLIST TICCMPL.
2. Locate the variable SRCOPTS.
3. Edit the script options from **S,X,MAP,OFF**, to **S,X,MAP,NOOFF,LIST** as shown in the following example:

```
/*****************************************
/* SET UP COMPILE OUTLIST OPTIONS
/*****************************************
  SET SCROPTS = &STR(S,X,MAP,NOOFF,LIST)
  SET LISTFLAG = N
  ISPEXEC VGET TICGLIB1 ASIS
  SET SYSDVAL = &TICGLIB1
  READDVAL TICCMPL
/*
  IF &TICCMPL = THEN DO
        SET LISTFLAG = P
        SET SCROPTS = &STR(S,X,MAP,NOOFF,LIST)
        IF &TICCMPL = &TICGLIB1 THEN *
           SET TICCMPL = &TIUPREF..&TICCMPL&TIUSUFX
        END
```

Modifying Remote Compile Scripts (MVS Implementation Toolset)

The remote compile script for COOL:Gen applications is stored in the Implementation Toolset SLIB library. The REXX procedure name for the compile script is TIXMVSLM.

To modify the compile script, follow these steps:

1. Make a copy of the script, then modify the copy.

Note: Modifying this script is a global change for all users. When you modify the COOL:Gen script, you are modifying it for everyone.

2. Locate the copy of the REXX execute procedure TIXMVSLM.
3. Locate the variable TICCMPCO.
4. Edit the compile options from **S,X,MAP,OFF**, to **S,X,MAP,NOOFF,LIST** as shown in the following example:

```
/* TICCMPCO = COBOL compile options */
ticcmpco = 'S,X,MAP,NOOFF,LIST'
listflag = 'N'
/* TICCMPL = listing library */
if g.listing_lib = '' then do
   ticcmpl = remove_quotes(g.listing_lib)
   listflag = 'P'
   ticcmpco = 'S,X,MAP,NOOFF,LIST'
end
```

After you have modified the script, you have two scripts from which to choose when you install the application. If you plan to measure the application, install it with the script you modified for use with the STROBE COOL:Gen Feature.

Creating the Map Data Set

You can create a map data set in either of two ways: you can create it when you produce the Performance Profile or you can create it as a separate step. Since you may want to create two map data sets for your COOL:Gen application (one for COOL:Gen and one for COBOL), this section describes how to create the map data set as a separate step.

- For more information on language-specific rules that cover indexing, see *STROBE MVS Concepts and Facilities*.
- For more general information about creating a map data set, see the *STROBE MVS User's Guide*.

When you have a SYSPRINT data set, you can create the map data set. To do so, select **Option 5** (INDEX) from the STROBE OPTIONS menu. The STROBE - INDEX TO CREATE A MAP DATA SET panel (Figure 2-2) appears.

Figure 2-2. STROBE - INDEX TO CREATE A MAP DATA SET Panel

```
------------------- STROBE - INDEX TO CREATE A MAP DATA SET -------------------
OPTION ===> B
            B  - Background processing   F  - Foreground processing

PROGRAM LANGUAGE: (Specify Y -- select one language)
 ADABAS/NATURAL  ===>       ADS/O           ===>        ASSEMBLER       ===>
 C (IBM OR SAS)  ===>       CA OPTIMIZER    ===>        COBOL           ===>  Y
 CSP             ===>       FORTRAN G       ===>        FORTRAN VS OR H ===>
 IEF             ===>       PL/I            ===>

OUTPUT: MAP DATA SET
        ===> 'WPA.TST.MAINMENU.MAP'

  UNIT ===> WPAANY     VOLUME ===>

INPUT: COMPILER SYSPRINT DATA SETS
        ===> 'WPA.TST.MAINMENU.SYSPRINT'
        ===>
        ===>
        ===>
        ===>
        ===>
```

5. In the PROGRAM LANGUAGE section, type "Y" in the **IEF** or **COBOL** field, depending on which map data set you are creating. (You should always create an COOL:Gen map data set. The COBOL map data set is optional.)
6. In the DATA SET NAME field, enter the name for the map data set you are about to create.
7. For COOL:Gen, enter the name of the SYSPRINT listing data set in the INPUT: COMPILER SYSPRINT DATA SETS field. If you specified COBOL in the PROGRAM LANGUAGE field, enter the fully qualified name of the listing data set that was created after you modified the compile scripts.

   Note: Usually, this data set is a partitioned data set, so you will also have to specify the member name in this field. For example:

   ```
   COMPOSER.TESTAPP.COBOL(MAINMENU)
   ```

You now have all the information necessary to create an indexed Performance Profile.

Controlling the Performance Profile - COOL:Gen Options

When you have the information you need (the sample data set and, optionally, the map data sets) you can create a Performance Profile for your COOL:Gen application. You can find detailed information about creating a Performance Profile in the STROBE/ISPF Online Tutorials or the *STROBE MVS User's Guide*. This section describes how you:

- specify detail parameters for the reports; see "Controlling the Level of Detail in the Reports"
- request indexing; see "Indexing the Performance Profile" on page 2-8
- prepare the Profile for downloading to iSTROBE on the workstation; see "Preparing Performance Profiles for iSTROBE" on page 2-9.

Controlling the Level of Detail in the Reports

When you create the Performance Profile for your COOL:Gen application, you can supply a parameter to specify how you want COOL:Gen activity presented. This optional parameter, IEF, has three subparameters that control the level of detail in the reports:

Subparameter	Purpose
TIMEPCT=*nn.n*	Indicate the minimum percentage of CPU or run time for an COOL:Gen statement detail line to be printed on the report. For example, to print detail lines that display activity above 15%, enter: **IEF=TIMEPCT=15** *nn.n* is a value between 0.0 and 99.9 (default **0**)
TOPTEXT=*nn*	Specify the number of COOL:Gen statements for which you want to see full statement text. The report shows the statements with the most identified CPU time or run time percentage, if you supply an IEF map data set when you create the performance profile. A value of 99 shows full statement text for all COOL:Gen statements within an action diagram. For example, to examine the statement text for the five statements that consumed the most CPU or run time, specify: **IEF=TOPTEXT=5** *nn* is a value from 0 to 99 (default **3**)
OFFSET	Specify that activity is to be reported by offset within each COBOL statement within SQL statement within COOL:Gen statement number. (The default is not to report the offset.)

To change the level of detail, enter "Y" in the Detail Reports field of the STROBE - PRODUCE A PERFORMANCE PROFILE panel. STROBE displays the STROBE - DETAIL FOR A PERFORMANCE PROFILE panel. The example in Figure 2-3 specifies that any COOL:Gen statement that has more than 2% CPU use or run time associated with it will display as a detail line in the reports. If you included the COOL:Gen map data set when you created the Performance Profile, the expanded report will also show the statement text for the five COOL:Gen statements that consumed the most CPU or run time. Each detail line will also show activity by the COBOL offset.

Figure 2-3. STROBE - DETAIL FOR A PERFORMANCE PROFILE with COOL:Gen Parameters

```
------------------  STROBE - DETAIL FOR A PERFORMANCE PROFILE  ------------------
-
COMMAND ===>

SAMPLE DATA SET: 'WPANAC.WPAJPS.S001D001'

REPORT OPTIONS:
  TITLE      ===>

  COMPRESS   ===> AAA     =.AAAA      ===>         =.          ===>         =.
             ===>         =.          ===>         =.          ===>         =.

             ===>         =.          ===>         =.          ===>         =.

             ===>         =.          ===>         =.          ===>         =.


  DETAIL     ===>          ===>          ===>          ===>
             ===>          ===>          ===>          ===>
             ===>          ===>          ===>          ===>

  RESOLUTION ===> 32      SORT SIZE ===>           LINES/PAGE ===>

OTHER PARAMETERS ===> IEF=(TIMEPCT=2,TOPTEXT=5,OFFSET)
```

Indexing the Performance Profile

After you have created a map data set, you specify that STROBE use this indexing information when creating the Performance Profile by following these steps:

SYSPRINT DATA SETS field, enter the name or names of any map data set that you created for the measured COOL:Gen application, as described in “Creating the Compiler Listing Data Set” on page 2-3.

1. Type “**Y**” in the Indexing field of the STROBE - PRODUCE A PERFORMANCE PROFILE panel.
2. Press **Enter** to display the STROBE - INCLUDE MAPS OF INDEXED SOURCE MODULES AND/OR DDIO FILES panel (Figure 2-4).
3. Specify the names of map data sets:
   - Enter the name or names of any map data set that you created for the measured COOL:Gen application, as described in “Creating the Compiler Listing Data Set” on page 2-3.
   - Specify the names of existing map data sets in the MAP DATA SET NAMES field. (Clear the names of map data sets you do not want to use.)
   - If you choose to create a new map data set in this step, type “**Y**” in the NEW MAPS field and follow the directions in “Creating the Map Data Set” on page 2-5. When you have finished indexing, press **Enter** to return to the INCLUDE MAPS OF INDEXED SOURCE MODULES AND/OR DDIO FILES panel where your map data set has been added to the list of names.

Figure 2-4. STROBE - INCLUDE MAPS OF INDEXED SOURCE MODULES AND/OR DDIO FILES

```
--------- STROBE - INCLUDE MAPS OF INDEXED SOURCE MODULES AND/OR DDIO FILES --
COMMAND ===>


SAMPLE DATA SET: 'COA01.STQ80.ATTRTEST.S001D001'


NEW MAPS: Do you want to execute an Indexer
          to create a new map data set
          and add it to the list below? (Y or N) ===> Y


 Specify names of map and/or DDIO files for inclusion in the STROBE Profile
DDIO languages supported
 COBOL    PL/I

    ===>
    ===>
    ===>
    ===>
    ===>
    ===>
    ===>
```

Preparing Performance Profiles for iSTROBE

If you plan to use iSTROBE to analyze your COOL:Gen measurement data, you must specify Performance Profile parameters that prepare the data for STROBE. We recommend that you request the SQL Analysis Feature reports be created, as these reports provide additional opportunities for improving applications that access DB2. For a brief description, see “Using the SQL Statement Analysis Reports” on page 3-12.

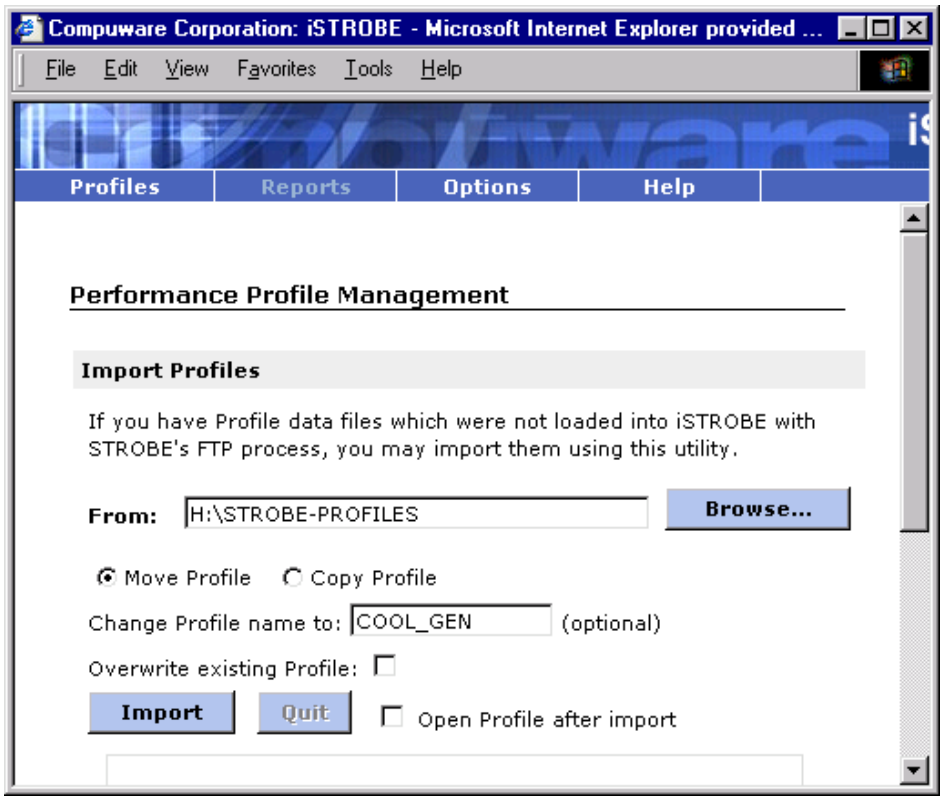
For more details, refer to the *STROBE MVS User’s Guide* or the *STROBE MVS User’s Guide with Advanced Session Management* and the section “Creating iSTROBE and APMpower Files”.

Updating the iSTROBE Profile List

When a Performance Profile is downloaded from the host to the workstation, you must update iSTROBE’s Profile list before you can analyze it. Start iSTROBE and:

- to view Performance Profiles downloaded using FTP directly from STROBE, select **Profiles**, **Update list**. When the Performance Profile Management window appears, click **Update Profile list**.
- to view Performance Profiles transferred to the workstation using another method, select **Profiles**, **Manage**, **Import**. When the Performance Profile Management window appears (Figure 2-5 on page 2-10), specify the location of the PROFILE.XML data file and click **Import**.

Figure 2-5. Importing Performance Profiles into iSTROBE



You are now ready to use the Performance Profile to analyze the efficiency of your COOL:Gen application.

- Chapter 3, “The iSTROBE Performance Profile for a COOL:Gen Application” describes how to interpret a Performance Profile using iSTROBE.
- Chapter 4, “The STROBE Performance Profile for a COOL:Gen Application” describes how to interpret a Performance Profile using with STROBE/ISPF.

Chapter 3.
The iSTROBE Performance Profile for a COOL:Gen Application

This chapter guides you through the process of analyzing a Performance Profile for a COOL:Gen application using iSTROBE. For a description of analyzing a Performance Profile with STROBE/ISPF, see Chapter 4, "The STROBE Performance Profile for a COOL:Gen Application". See the iSTROBE *Getting Started* guide and online help for additional information about using iSTROBE.

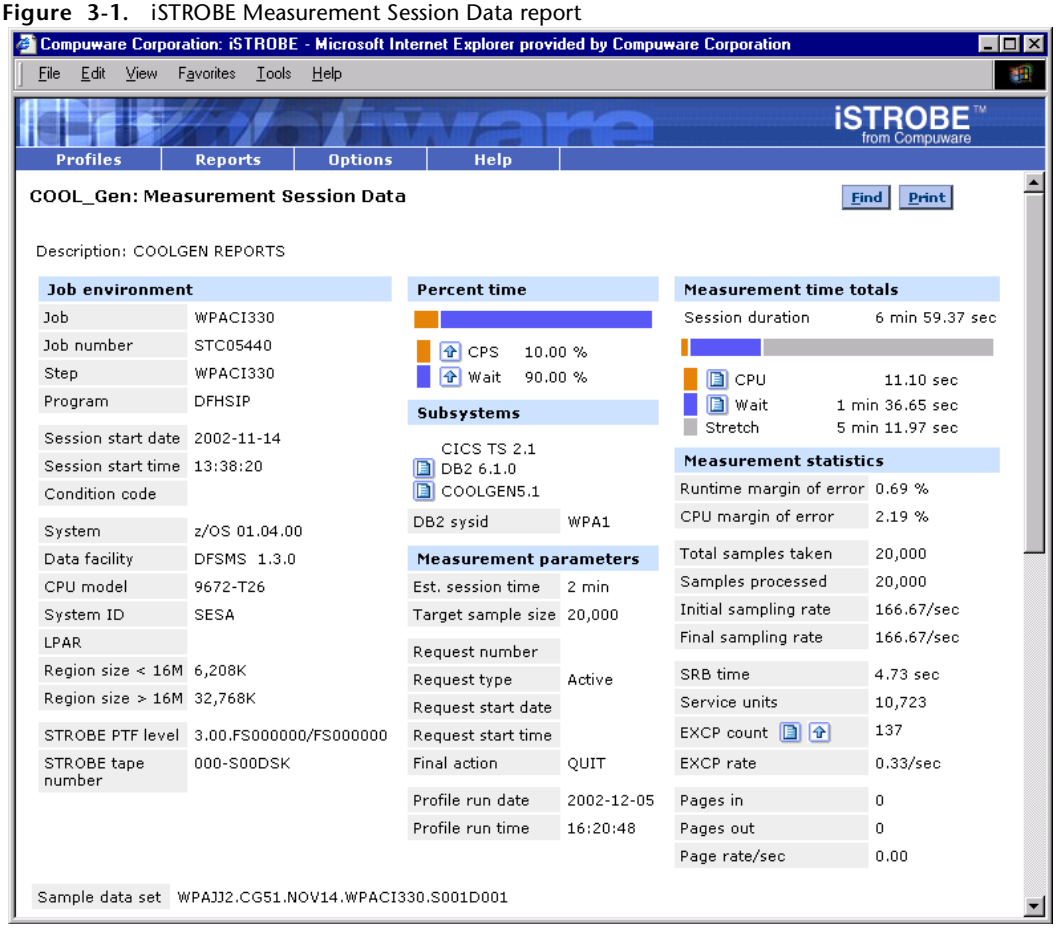
This chapter describes the following steps for interpreting the information in the Performance Profile:

1. opening the Performance Profile
2. verifying the measurement session information
3. choosing between the execution and wait reports
4. finding significant activity in a COOL:Gen application
5. identifying performance improvement opportunities

Verifying the Measurement Session Information

The first step is to ensure that the Performance Profile you are about to examine is valid. Start iSTROBE, open the list of available Performance Profiles, and click the name of the one associated with your application. iSTROBE displays the Measurement Session Data report for this Performance Profile (Figure 3-1).

Figure 3-1. iSTROBE Measurement Session Data report



The Measurement Session Data report is the starting point for analyzing any Performance Profile. This report contains information about the measurement session associated with the measurement request you submitted. There are a number of fields on this report that you should look at to determine that the measurement session is valid, and to determine which reports in the Performance Profile you should analyze.

To verify that the measurement session was valid, look at the following areas and fields on the report:

- In the top section of the report, ensure that the **Program**, **Session start date**, and **Session start time** are what you expect.
- Ensure that the appropriate version of COOL:Gen (or Composer or IEF) appears in the **Subsystems** field.

The format of the **Subsystems** field varies, depending on the number of different COOL:Gen releases running in the measured address space. The Subsystems field identifies the release level of Composer, IEF, or COOL:Gen that was used to generate the application running in the measured address space. The formats are as follows:

Value in Subsystems field	Release of COOL:Gen
IEF 5.2	IEF Release 5.2
IEF 5.3	Composer by IEF Release 5.3
IEF C.3	Composer 3
COOLGEN4.1A	COOL:Gen 4.1a
COOLGEN4.11	COOL:Gen 4.1.1
COOLGEN5.0	COOL:Gen 5.0
COOLGEN5.1	COOL:Gen 5.1
COOLGEN6.0	COOL:Gen 6.0

Note: The Subsystems ID ends with "+" when more than three releases are identified as running in the address space. For example, 5.2, 5.3, C.3+.

In Figure 3-1, the Subsystems field shows that there is one release of COOL:Gen running in the measured address space. The report additionally indicates that the DB2 Feature was invoked in conjunction with the STROBE COOL:Gen Feature by also showing the name and release level of DB2 in the Subsystems field.

- In the Measurement Statistics section on the left side of the report, examine the **Runtime Margin of Error** and the **CPU margin of error** fields. A run-time margin of error of less than 2% usually reflects a reliable measurement. If you find that the Performance Profile you are creating has a run-time margin of error higher than 2%, increase the target sample size and measure the application again. However, a high CPU margin of error does not mean that the Performance Profile is invalid. Rather, it indicates that you should focus your analysis on the reports that detail run time.

Having verified that the Performance Profile is valid, the next step is to determine which reports in the Performance Profile to analyze.

Choosing Between Execution and Wait Reports

You typically do not need to examine all of the reports in the Performance Profile. To determine whether to examine the CPU execution or wait reports in the Performance Profile, look at the values reported for the **CPS** field (the percentage of time that one or more CPUs were active) and the **Wait** field in the Percent Time section of the report. The values in these fields will determine whether you should focus on the execution reports or the wait reports.

For example, in the example in Figure 3-1, the **Wait** value of 90% indicates that trying to curtail wait time might yield the most significant performance improvements.

When the value for **Wait** is very high, you should examine the following wait-oriented reports to determine where the application is waiting:

- Resource Demand Distribution
- Wait Time by Module
- I/O Facility Usage
- Data Set Characteristics
- Attribution of Wait Time

When focusing on CPU, you should examine the following CPU reports:

- Program CPU Usage
- Transaction CPU Usage
- Attribution of CPU Usage

For more information about these reports, refer to the iSTROBE online help.

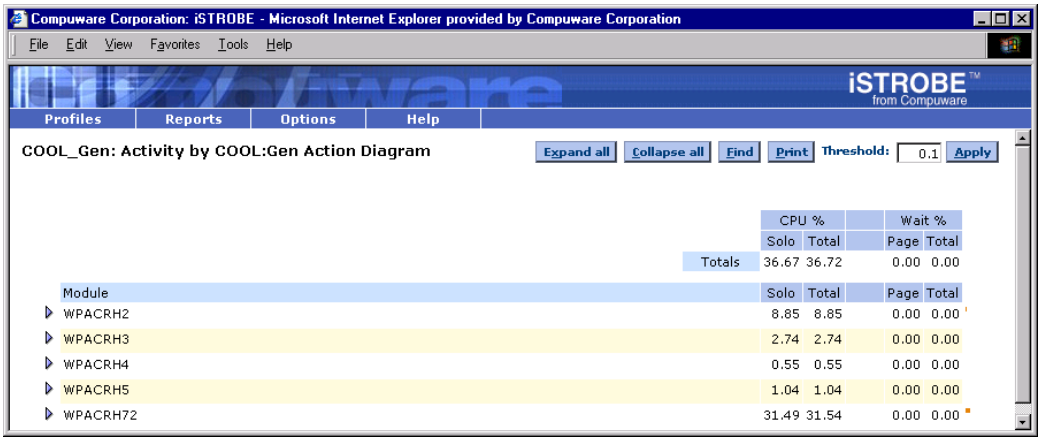
Having verified that the Profile is valid, and having determined whether you will focus on wait or CPU for performance improvement opportunities, you open the report at the summary level, choose specific modules to examine, and then look at the details for those modules.

Analyzing Activity in a COOL:Gen Application

The central resource for analyzing performance information about your COOL:Gen application is the Activity by COOL:Gen Action Diagram report. The Activity by COOL:Gen Action Diagram report identifies the CPU use and wait time for each module generated by COOL:Gen, and the action diagram names that correspond to each module or control section. As you examine the report, you can view additional detail about action diagrams, associated COOL:Gen and SQL statement text, and system services used by the application.

To view this report, on the Measurement Session Data report, find the Subsystems section near the middle of the report and click the report icon next to the COOL:Gen subsystem data. The Activity by COOL:Gen Action Diagram report appears (Figure 3-2). (Alternatively, you can go to the menu at the top of the report and click **Reports**, **COOL:Gen**, and then **Activity by COOL:Gen Action Diagram.**)

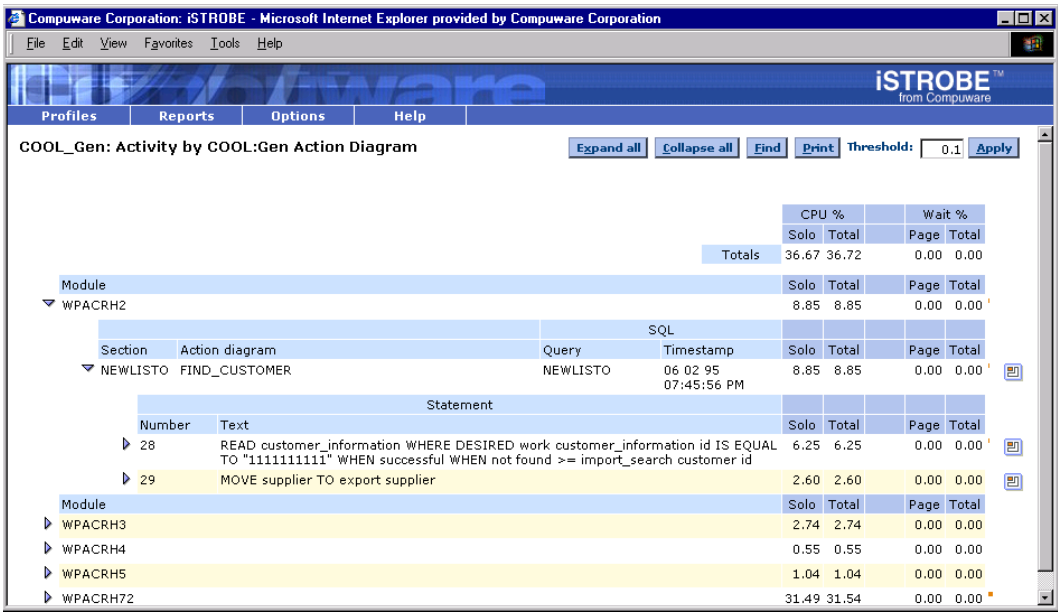
Figure 3-2. Activity by COOL:Gen Action Diagram report



The module names in the left column of the report are the COOL:Gen load module names that were specified during the **Packaging** of the application on the workstation.

The arrow icon next to a data item indicates that you can expand it to view additional detail, starting with a list of action diagrams associated with the load module and the percent of CPU and wait that each consumed. To view a list of COOL:Gen statements associated with an action diagram and their resource use, click the arrow icon next to the section name (Figure 3-3 on page 3-6).

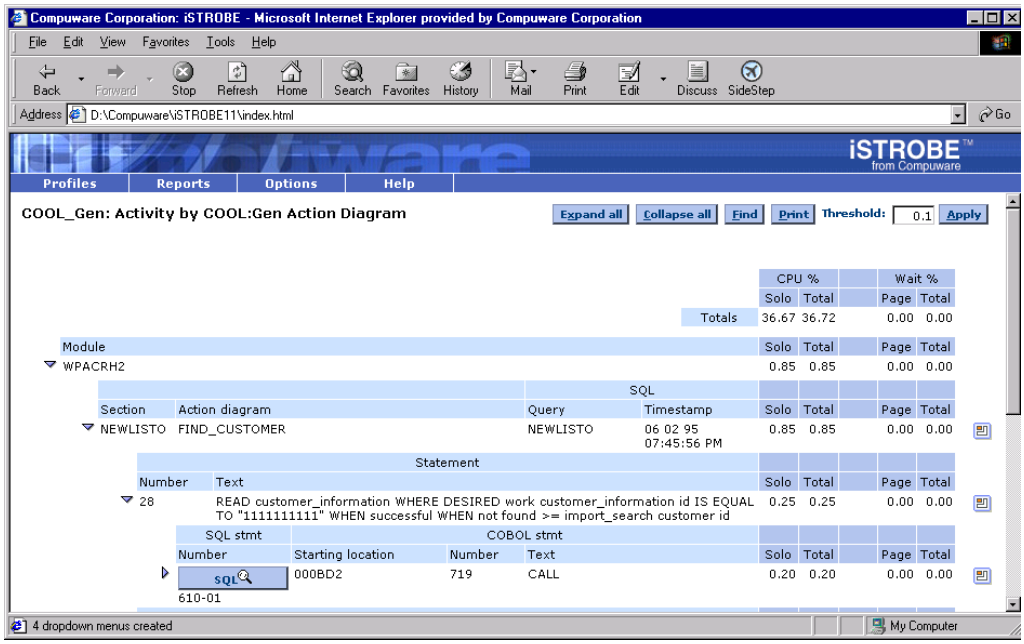
Figure 3-3. Expanding the Activity by COOL:Gen Action Diagram report



Note: To display the COOL:Gen statement number and the activity (CPU or wait time) associated with the statement, the Composer 4 Design dialog option "Execute Statement Number" must be set to "ON". This restriction applies only to Composer 4 and COOL:Gen users. The default for this option is "ON". If the Execute Statement Number option is set to "OFF", STROBE will not obtain the COOL:Gen statement number and all activity will be assigned to statement number zero.

To examine the activity for COOL:Gen statement number 28 in greater detail, click on the arrow to the left of statement 28. iSTROBE displays the detail level report for the COOL:Gen statement (Figure 3-4).

Figure 3-4. Viewing COOL:Gen Statement Detail



This report always shows the COOL:Gen statement number and the percentage of CPU use associated with the statement. The report can also show other information, such as SQL statement numbers and COBOL statement numbers and text associated with this action diagram statement, depending on how you created the Profile. To learn how to ensure that your Performance Profile includes this information, see “Controlling the Level of Detail in the Reports” on page 2-7. The Performance Profile example shown here was created by including both a COOL:Gen map data set and a COBOL compiler listing.

Tip: To view information about a single data item, without any other information in the report, click the context icon . A small pop-up report appears, showing only that data item and its parents. Doing so can be helpful if, for example, a module is associated with many action diagrams, and viewing the action diagram in which you are interested requires that you scroll so far down the list that the module is no longer visible in the browser window.

At this point in your analysis, you may want to:

- examine the action diagram code associated with a particular COOL:Gen statement, as described in “Identifying Performance Improvement Opportunities”, or
- examine the SQL text associated with a particular statement, if SQL statement numbers are available and SQL statements seem to a significant source of resource use. See “Examining How Action Diagrams Access DB2 Data” on page 3-11.

Identifying Performance Improvement Opportunities

To identify potential opportunities for performance improvement, you need to look at not only the COOL:Gen statement text, but the actual action diagram code as well. In Figure 3-4 on page 3-7, the text for statement 28 indicates that one or more customer records are being read. The next step is to view the action diagram code in Figure 3-5 and look at what is happening around statement number 28.

Figure 3-5. COOL:Gen Action Diagram Code from a COBOL Compiler Listing

```
Action Block Description:

      +- TIMEFUNC
      |    IMPORTS:
      |      Entity View import customer_time (optional,transient,import only)
      |        date (optional)
      |        time (optional)
      |        name (optional)
      |        id (optional)
   .
   .
   .
      |    EXPORTS:
      |    LOCALS:
      |      Entity View local customer_information
      |        time
      |        id
      |      Work View   ief_supplied
      |        select_char
      |        count
      |    ENTITY ACTIONS:
   .
   .
   .
   19 |  +- IF COMMAND IS EQUAL TO return
   20 |  |  EXIT STATE IS return_to_caller
   21<------ESCAPE
   19 |  +--
   22 |  SET ief_supplied count TO 1
   23 |
   24 |
   25 |  += WHILE ief_supplied count IS LESS THAN 1000
   26 |  |  SET ief_supplied count TO
   26 |  |               ief_supplied count + 1
   27 |  |
   28 |  |  +- READ customer_information
   28 |  |  |         WHERE DESIRED
   28 |  |  |               customer_information id
   28 |  |  |               IS EQUAL TO "1111111111"
   28 |  |  +- WHEN successful
   29 |  |  |  SET ief_supplied time_value TO
   29 |  |  |               timetimestamp(
   29 |  |  |               CURRENT_TIMESTAMP)
```

Looking at the code (Figure 3-5) around statement 28, notice that statement 25 is the beginning of a loop. This WHILE statement reads one or more customer records and eventually calls a COOL:Gen-supplied time function (CURRENT_TIMESTAMP) shown in statement 29, storing the result in the time_value field.

This action diagram populates a current time field on one of the data entry panels of the application. Each time a new customer record is created or updated, this time routine is invoked, consuming additional CPU time. This could become a performance problem when the use of this routine is high.

This represents a possible performance improvement opportunity: remove the call to the current time function from the loop.

Figure 3-6 shows a possible solution to this problem. You have moved the call to the current time routine outside of the WHILE loop. As shown in the example, the call is now made once (in statement 25) and the result is saved in the COOL:Gen-supplied local work view time_value, eliminating subsequent calls to the current time routine.

Figure 3-6. COOL:Gen Action Diagram Code (Revised)

```
Action Block Description:

     +- TIMENOFC
     |    IMPORTS:
     |      Entity View import customer_time (optional,transient,import only)
     |        id (optional)
  .
  .
  .
     |    EXPORTS:
     |    LOCALS:
     |      Entity View local customer_information
     |        time
     |        id
     |      Work View   work ief_supplied
     |        count
     |        time_value
     |    ENTITY ACTIONS:
  .
  .
  .
  19 |  +- IF COMMAND IS EQUAL TO return
  20 |  |  EXIT STATE IS return_to_caller
  21<------ESCAPE
  19 |  +--
  22 |  SET work ief_supplied time_value TO
  22 |              timetimestamp(CURRENT_TIMESTAMP)
  23 |  SET work ief_supplied count TO 1
  24 |
  25 |  += WHILE work ief_supplied count IS LESS THAN
  25 |  |           1000
  26 |  |  SET work ief_supplied count TO
  26 |  |              work ief_supplied count + 1
  27 |  |
  28 |  |  +- READ customer_information
  28 |  |  |        WHERE DESIRED
  28 |  |  |              customer_information id
  28 |  |  |              IS EQUAL TO "1111111111"
  28 |  |  +- WHEN successful
  29 |  |  |  |  SET number TO input customer_number
  30 |  |  |  |  SET time TO work ief_supplied
  30 |  |  |  |              time_value
```

After you complete the changes to the action diagram and reinstall the application on the host, measure the application again to see the effect of the changes. Comparing the Measurement Session Data reports for both measurements, you can see a significant reduction in CPU session time, as well as an overall decrease in elapsed time.

Additional COOL:Gen Performance Profile Data

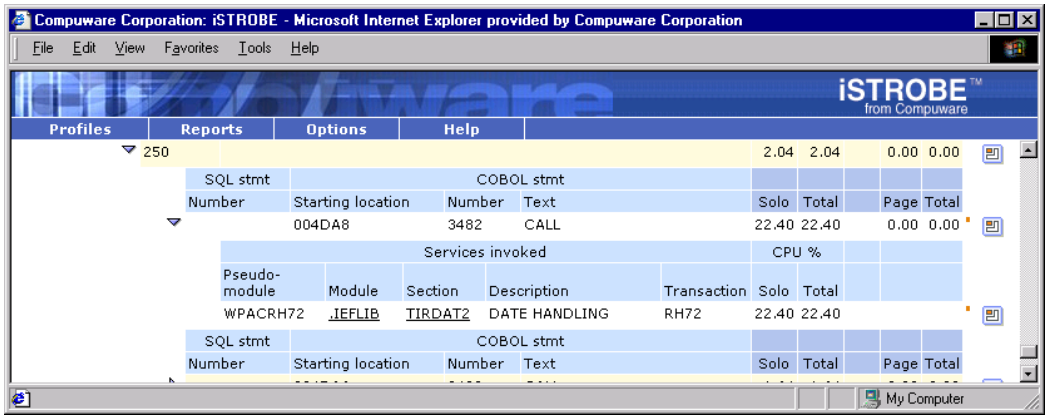
The example in the previous sections centered on the reports associated with COOL:Gen statements within action diagrams. However, you may also find significant performance improvement opportunities in the way that your application accesses and uses system service routines, and in the way that it accesses DB2 data. iSTROBE provides convenient ways to view this type of information, as described in the following sections.

Examining System Services Invoked by Action Diagrams

An arrow next to a COOL:Gen statement number indicates that this statement invoked a system service routine that used CPU time. Click the arrow to view details about the services invoked by the statement, in this case statement number 250. In the example in Figure 3-7, analyzing statement number 250 shows that one COOL:Gen service routine, TIRDAT2, is consuming a significant amount of CPU time.

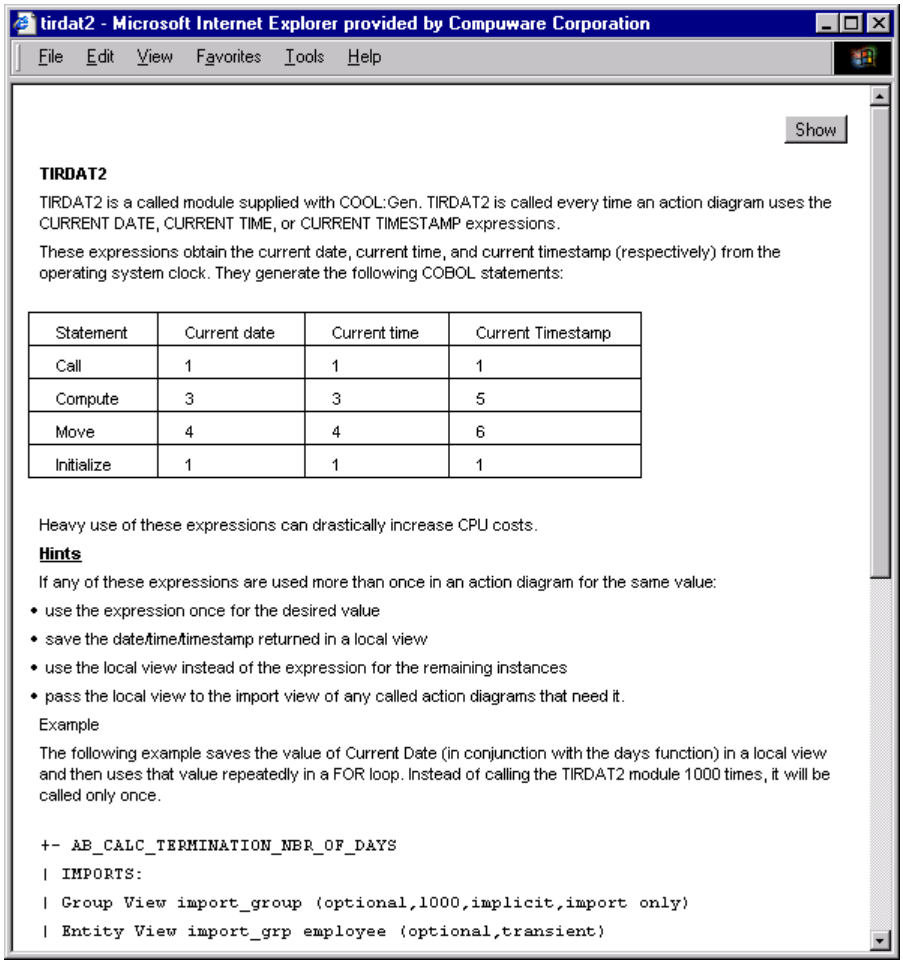
Note: The .IEFLIB pseudo-module indicates a service routine supplied with COOL:Gen.

Figure 3-7. Services Invoked by A COOL:Gen Statement



When iSTROBE underlines a report data item, for example TIRDAT2 in Figure 3-7, you can click it to view online help that provides additional information and performance tips, as shown in Figure 3-8.

Figure 3-8. Online Help for a COOL:Gen Module

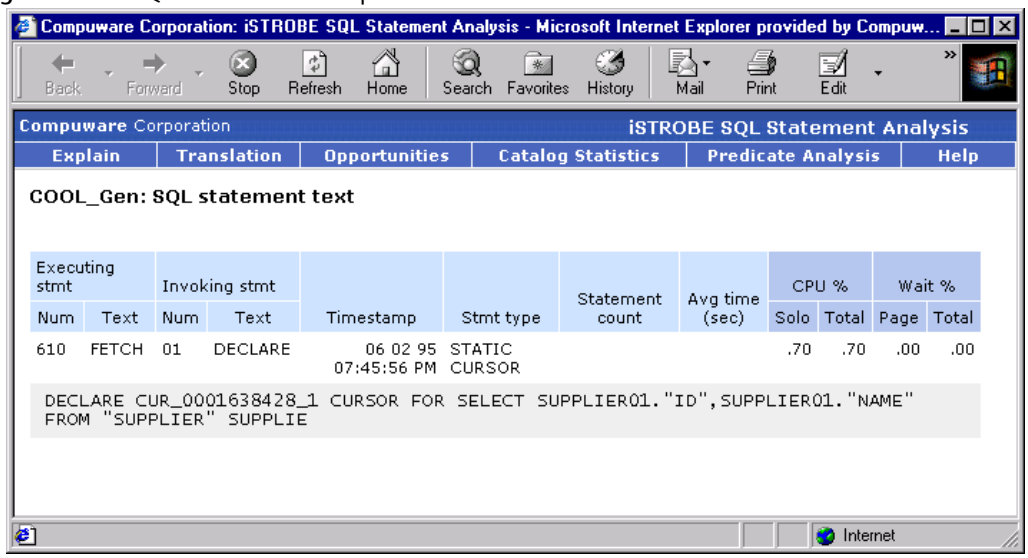


Examining How Action Diagrams Access DB2 Data

In addition to examining data specific to COOL:Gen action diagrams and COOL:Gen statements, it can be helpful to examine any SQL statement text generated on behalf of a COOL:Gen statement. If there are SQL statements associated with a COOL:Gen statement, iSTROBE displays the SQL statement number on the Activity by COOL:Gen Action Diagram report, as shown in Figure 3-4. iSTROBE also shows the line number of the statement called by the COBOL statement identified in the COBOL Statement Text column, as found in the DB2 precompiler listing.

To view the SQL statement text and the activity associated with the statement, click the SQL button SQL above the statement number. iSTROBE displays the SQL Statement Text report (Figure 3-9).

Figure 3-9. SQL Statement Text Report



Examining this additional data can reveal performance improvement opportunities other than the ones uncovered by examining only the COOL:Gen statement text. Because the SQL statement numbers and the CPU activity associated with them are identified on the report, it is relatively easy to identify which SQL statements within an action diagram are consuming the most resources. You are then able to focus your attention on a few SQL statements, out of possibly hundreds for a very large application.

For example, you may have identified a number of SQL statements that are consuming a relatively large amount of CPU time. By examining the text for these statements, you discover that they are all SELECT statements that have WHERE clauses requiring data from multiple tables, resulting in a JOIN operation. This operation can be quite expensive in terms of CPU time. With this information, you could then explore other coding alternatives that may offer a performance improvement. In this case, it may be possible to revise the views defined in the action diagram, eliminating the unnecessary JOINs.

Using the SQL Statement Analysis Reports

iSTROBE can provide additional reports that help you reduce the resource use of your SQL calls. These SQL statement analysis reports display in the lower part of the SQL statement analysis window shown in Figure 3-9. Choose the SQL statement analysis report from the menu at the top of the window.

- **Explain** identifies the DB2 access path for a measured SQL request
- **Translation** correlates and reports access path explanations with relevant DB2 catalog information about database objects
- **Opportunities** supplies specific recommendations for improving SQL statement efficiency using well documented rules
- Catalog Statistics displays the DB2 catalog statistics.

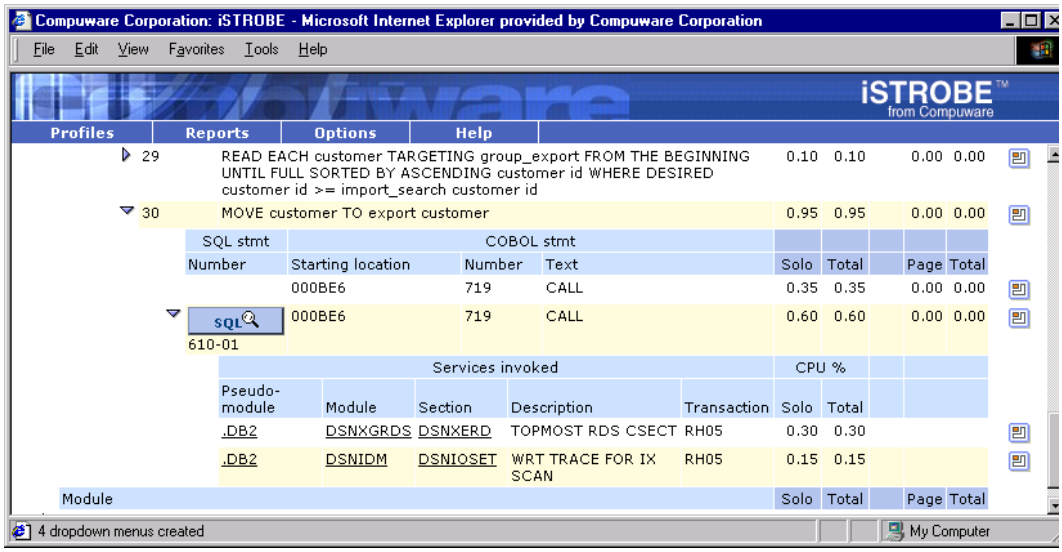
In addition, you can select **Predicate Analysis**. Predicate Analysis provides an easy way to see whether you can improve the efficiency of your SQL by changing the way DB2 will evaluate your predicates. Predicate Analysis is available for all SQL statements and opens a new browser window from which you can test changes to SQL code interactively.

To obtain the SQL statement analysis reports, you must request them when creating the Performance Profile. See the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management* for details.

Examining System Services Invoked by DB2

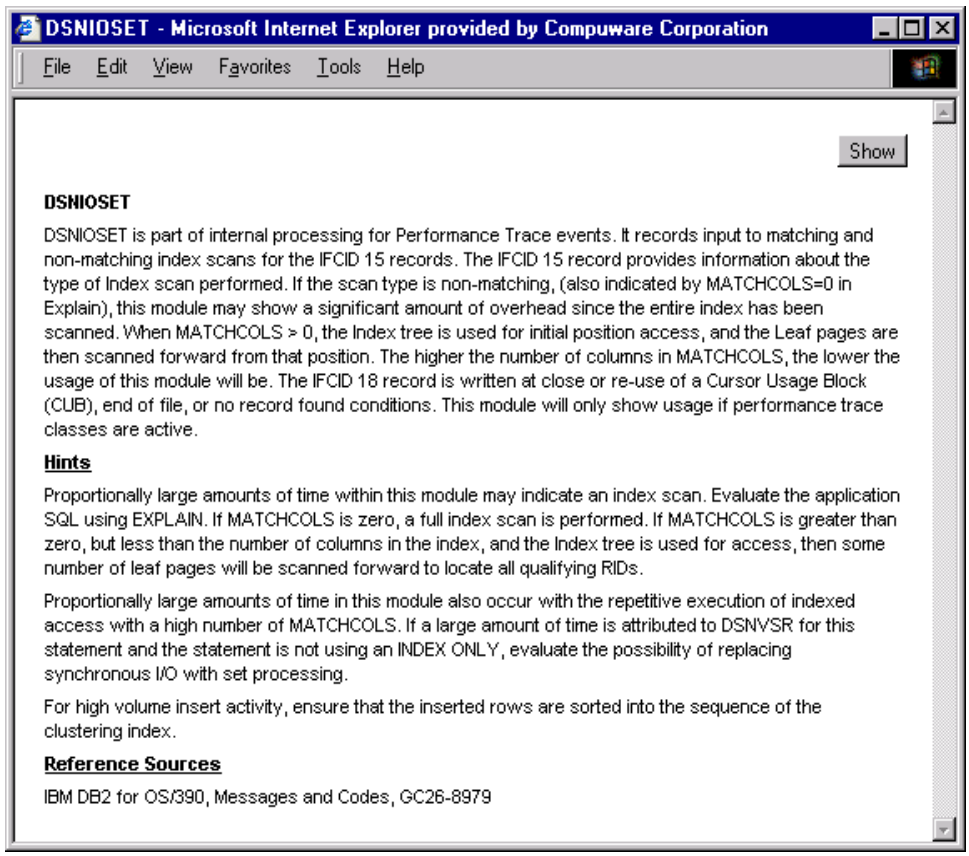
An arrow next to an SQL statement number indicates that this statement invoked a system service routine that used CPU time. Click the arrow to view details about the services invoked by the statement (Figure 3-10).

Figure 3-10. Services Invoked by an SQL Statement



For ideas about how to improve performance, review which services were invoked and be sure to look at the online help that iSTROBE provides for underlined modules. For example, the help for the DSNIOSET module suggests that you try to reduce CPU usage by increasing the number of matching columns used in an index scan (Figure 3-11).

Figure 3-11. iSTROBE Help for DB2 Module DSNIOSET



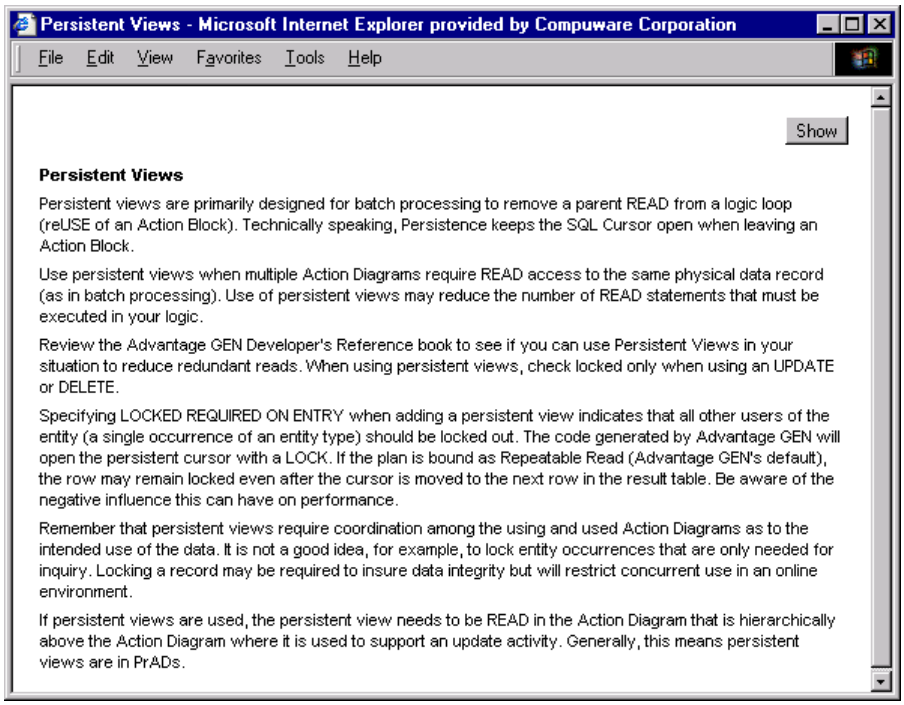
Using the COOL:Gen Online Hints

iSTROBE's online help includes an online book containing guidelines and suggestions for improving the performance of your COOL:Gen application. You may find this a useful reference to have while coding your programs. To access this information:

1. From the iSTROBE main window, select **Help**, and then select **iSTROBE help**.
2. From the Help table of contents, select **COOL:Gen Hints**. Click the plus sign (+) next to any book icon within COOL:Gen Hints to view an expanded table of contents.

Figure 3-12 shows an example of one of the suggestions provided in the COOL:Gen application performance hints.

Figure 3-12. Example of iSTROBE Online Hints for Composer



Chapter 4.
The STROBE Performance Profile for a COOL:Gen Application

This chapter describes how to read the STROBE Performance Profile as it relates to COOL:Gen applications. For a complete description of the STROBE Performance Profile, see Chapter 3 of *STROBE MVS Concepts and Facilities*. For a description of analyzing a Performance Profile using iSTROBE, see Chapter 3, "The iSTROBE Performance Profile for a COOL:Gen Application".

This chapter describes the following steps for interpreting the information in the Performance Profile:

- verifying the measurement session information
- choosing between the execution and wait reports
- finding significant activity in a COOL:Gen application
- identifying performance improvement opportunities

About the Application

The examples in this chapter refer to the Performance Profile produced after the measurement of a COBOL application generated using COOL:Gen and running under a TSO session accessing a DB2 database. The application stores and tracks information about potential customers, and is intended for use by the Sales and Marketing division of a small software company. The application is currently in the test phase of development. In the course of testing the application, the developers measured it to identify any opportunities for performance improvement.

Verifying the Measurement Session Information

The starting point for analyzing any Performance Profile is the first report in the Performance Profile, the Measurement Session Data report (Figure 4-1 on page 4-3). This report describes the environment during a measurement session and provides some overall measurement statistics.

There are a number of fields on this report that you should look at to determine that the Performance Profile is valid, and also to determine which reports in the Performance Profile you should analyze.

To verify that the Performance Profile is valid, look at the following fields:

- in the JOB ENVIRONMENT column, ensure that the PROGRAM MEASURED, DATE OF SESSION, and TIME OF SESSION are what you expect
- in the MEASUREMENT PARAMETERS column, ensure that the appropriate version of Composer, IEF or COOL:Gen appears in the SUBSYSTEM field

- the format of the SUBSYSTEM field varies, depending on the number of different Composer, IEF or COOL:Gen releases running in the measured address space. The SUBSYSTEM field identifies the release level of Composer, IEF or COOL:Gen, that was used to generate the application running in the measured address space. The formats nare as follows:

Value in SUBSYSTEM field	Release of COOL:Gen
IEF 5.2	IEF Release 5.2
IEF 5.3	Composer by IEF Release 5.3
IEF C.3	Composer 3
IEF C.4	Composer 4
COOLGEN4.1A	COOL:Gen 4.1a
COOLGEN4.11	COOL:Gen 4.1.1
COOLGEN5.0	COOL:Gen 5.0
COOLGEN5.1	COOL:Gen 5.1
COOLGEN6.0	COOL:Gen 6.0

  Note: The SUBSYSTEM id ends with "+" when more than three releases are identified as running in the address space. For example, 5.2, 5.3, C.3+.

  In Figure 4-1 on page 4-3, the SUBSYSTEM field shows that there is one release of COOL:Gen running in the measured address space. The report additionally indicates that the DB2 Feature was invoked in conjunction with the STROBE COOL:Gen Feature by also showing the name and release level of DB2 in the SUBSYSTEM field.

- in the MEASUREMENT STATISTICS column, examine the RUN MARGIN OF ERROR PCT and the CPU MARGIN OF ERROR PCT fields. A run-time margin of error of less than 2% usually reflects a reliable measurement. If you find that the Performance Profile you are creating has margin of error rates higher than 2%, increase the target sample size and measure the application again. A high CPU margin of error does not mean that the Performance Profile is invalid. Rather, it indicates that you should focus your analysis on the reports that detail run time.

Having verified that the Performance Profile is valid, your next step is to determine which reports in the Performance Profile to analyze.

Figure 4-1. Measurement Session Data Report

```
                                        ** MEASUREMENT SESSION DATA **
-------- JOB ENVIRONMENT --------      ------- MEASUREMENT PARAMETERS ------      ----- MEASUREMENT STATISTICS ----

PROGRAM MEASURED   -    IKJEFT01       ESTIMATED SESSION TIME  -       2 MIN      CPS TIME PERCENT -          56.12
JOB NAME           -      WPAKRM       TARGET SAMPLE SIZE      -      15,000      WAIT TIME PERCENT -          43.88
JOB NUMBER         -    TSU27381       REQUEST NUMBER   (A)    -         148      RUN MARGIN OF ERROR PCT  -    .80
STEP NAME   -  WPAIEF53.WPAIEF53       FINAL SESSION ACTION    -        QUIT      CPU MARGIN OF ERROR PCT  -   1.07

DATE OF SESSION    -    03/30/02       SYS REQ -  SCS01                          TOTAL SAMPLES TAKEN -      15,000
TIME OF SESSION    -    09.03.46                                                 TOTAL SAMPLES PROCESSED -  15,000
                                       BASELINE OVERRIDE -          CUSTEST1     INITIAL SAMPLING RATE- 125.00/SEC
                                       LIBRARY -   WPA.IEF53.IVP.TSO.EXELOAD     FINAL SAMPLING RATE  - 125.00/SEC
SYSTEM -             ESA SP4.3.0
DFSMS              -       1.1.0                                                 SESSION TIME -    5 MIN  0.86 SEC
SUBSYSTEM          -   DB2 5.1.0       --------- REPORT PARAMETERS ---------     CPU TIME -        1 MIN 44.91 SEC
                           IEF 5.3                                               WAIT TIME    -    1 MIN 21.82 SEC
CPU MODEL          -   3090-600S       REPORT RESOLUTION    -       64 BYTES     STRETCH TIME -    1 MIN 54.13 SEC
SMF/SYSTEM ID      -  SESA/SCS01       SORTSIZE             -        999,999
                                       LINES/PAGE           -             60     SRB TIME     -    0 MIN  4.76 SEC
REGION SIZE BELOW 16M  -  6,208K                                                SERVICE UNITS-             101343
REGION SIZE ABOVE  -     32,768K       DASD= 2.0%  DASDGAP= 5
                                       IEF=(NOSQL)                              PAGES IN-      0     OUT-       0
PTF LVL- 3.00.FS004701/FS000000                                                 PAGING RATE  -           0.00/SEC
                                                                                EXCPS -      4,067      13.52/SEC


SAMPLE DATA SET  - WPA.WPAKRM.S003D001
```

Choosing Between Execution and Wait Reports

You typically do not need to examine all of the reports in the Performance Profile. To determine whether to examine the CPU execution or wait reports in the Performance Profile, look under the MEASUREMENT STATISTICS column at the values reported for the CPS TIME PERCENT (the percentage of time that one or more CPUs were active) and WAIT TIME PERCENT fields.

In this example, you will examine the execution reports to determine opportunities for improvement in CPS time. CPS time is the run time during which the central processing subsystem (consisting of one or more CPUs) was in use by application tasks executing within the measured job step.

In Figure 4-1 on page 4-3, the CPS TIME PERCENT value of a little over 56% suggests that performance might be improved by reducing CPU usage. The WAIT TIME PERCENT value of almost 44% indicates that trying to curtail wait time might also improve performance.

If the value for WAIT TIME PERCENT is very high, then you should examine the following wait reports to determine where the application is waiting:

- Resource Distribution Demand
- Wait Time by Module
- I/O Facility Utilization Summary
- Data Set Characteristics
- Action Diagram Wait Summary
- Wait Time by IEF Statement
- Attribution of CPU Wait Time

Having verified that the Performance Profile is valid, and having determined that we will examine the execution reports for performance improvement opportunities, our next step is to look at the CPU summary reports to find out what modules are consuming the most CPU or run time. This approach would be the same if you were examining the wait reports. Essentially, you start with the summary reports, isolate specific modules to examine and then refer to the detail reports for those modules.

In our example, we notice that the most CPU activity appears in action diagrams. Therefore we will focus on the reports specific to action diagrams and the statements within the action diagrams.

Finding Significant Activity in a COOL:Gen Application

The first step in finding significant activity in a COOL:Gen application is to examine the summary report for the IEF action diagrams.

The IEF Action Diagram CPU Summary and the CPU Usage by IEF Statement reports show which action diagrams and, within an action diagram, which IEF statements are responsible for CPU activity. Turn to these reports when most of the CPU activity occurs in action diagrams.

The first report we will examine is the IEF Action Diagram CPU Summary report (Figure 4-2).

Figure 4-2. IEF Action Diagram CPU Summary Report

```
                                    ** IEF ACTION DIAGRAM CPU SUMMARY **
MODULE    SECTION   ACTION DIAGRAM              CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.07%
NAME      NAME      NAME                         SOLO     TOTAL        .00     23.00     46.00     69.00     92.00

CUSTEST1  CRCUSTOM  CREATE_CUSTOMER                .02       .02        .
CUSTEST1  DATEFUNC  TIMEFUNC                     90.39     90.41        .****************************************
CUSTEST1  DELCUSTO  DELETE_CUSTOMER                .01       .01        .
CUSTEST1  RDCUSTOM  READ_CUSTOMER                  .01       .01        .
CUSTEST1  REMOVECU  DELETE_A_CUSTOMER              .01       .01        .
                                               -------   -------
TOTAL CPU ACTIVITY                             90.45     90.47         -
```

Figure 4-2 shows the action diagram names for this load module and, at the far right of each action diagram name, the percent of CPU consumed. Because action diagram TIMEFUNC consumed the largest amount of CPU, focus your attention there.

To view a list of the COOL:Gen statements that consumed the most CPU for this action diagram, turn to the CPU Usage by IEF Statement report (Figure 4-3) for action diagram TIMEFUNC. This report breaks out the CPU use for the action diagram by IEF statement number. The statement numbers appear to the far left of the report.

This report always shows the IEF statement numbers and the percentage of CPU use associated with each statement. The report can also show other information, such as SQL statement numbers and COBOL statement numbers and text, depending on how you created the Performance Profile.

Note: To display the IEF statement number and the activity (CPU or Wait time) associated with the statement on the CPU Use by IEF Statement report, the Composer 4 Design dialog option “Execute Statement Number” must be set to “ON”. This restriction applies only to Composer 4 and COOL:Gen users. The default for this option is “ON”. If the Execute Statement Number option is set to “OFF”, STROBE will not obtain the IEF statement number and all activity will be assigned to statement number zero.

In addition to the subparameters described in “Controlling the Level of Detail in the Reports” on page 2-7, you can also suppress the display of SQL statement numbers associated with IEF statements. Specifying NOSQL on the IEF reporting parameter results in a more concise report. (The default is to display the SQL statement numbers.)

Note: If an action diagram has been indexed, the statement text is included in the CPU Usage by IEF Statement report. By default, this report shows the text for the three IEF statements that consumed the most CPU time. If you are interested in seeing the text for more statements, specify the IEF=TOPTEXT parameter when creating the Performance Profile. For more information, see "Controlling the Level of Detail in the Reports" on page 2-7.

For this report, we want to identify "spikes" of activity, or control sections generating the highest CPU activity. These spikes will be the areas of the action to focus on for possible performance improvement opportunities. As shown in Figure 4-3, statements 10 and 17 within action diagram TIMEFUNC are both relatively heavy users of CPU time.

Figure 4-3. CPU Usage by IEF Statement Report

```
                                        ** CPU USAGE BY IEF STATEMENT **
MODULE          - CUSTEST1
SECTION         - DATEFUNC
ACTION DIAGRAM - TIMEFUNC
DBRM            - DATEFUNC                              CREATED - 21 MAR 02 15:05:47

IEF STATEMENT
NUMBER  TEXT
    10   READ customer_information WHERE DESIRED customer_information id = "1111111111" WHEN successful  WHEN not found
    17   READ customer_time  WHERE DESIRED customer_time id = "1111111111" WHEN successful  WHEN not found
    19   UPDATE customer_time  WHEN successful  WHEN not unique  WHEN permitted value violation

IEF STMT    SQL   SECTION  STMT       STATEMENT       CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.07%
NUMBER     STMT   OFFSET  NUMBER        TEXT          SOLO     TOTAL        .00      6.00     12.00     18.00     24.00
     8                                                 .04       .04        .
    10                                               21.53     21.54        .***********************************
    11                                                7.23      7.23        .************
    12                                               14.86     14.86        .************************
    14                                                 .04       .04        .
    17                                               23.05     23.06        .*************************************
    18                                                6.99      6.99        .***********
    19                                               16.55     16.55        .***************************
    21                                                 .12       .12        .
                                                     -----     -----
DIAGRAM - TIMEFUNC                        TOTALS     90.39     90.41
```

At this point, you can examine statement text for a particular IEF statement or you can look at the SQL text associated with a particular statement, if SQL statement numbers are displayed and that seems to be where the activity is centered.

Note: This performance profile used the default settings for the IEF parameter, and included an IEF map data set.

If you were to examine the SQL statement text, you would refer to the CPU Use by SQL Statement report for the DBRM referenced at the top of the CPU Use by IEF Statement report. For more information, see "Additional Reports to Examine" on page 4-9.

Identifying Performance Improvement Opportunities

The CPU Usage by IEF Statement report shown in Figure 4-3 indicates that most of the CPU activity seems to be located at statement numbers 10 and 17. The text for the statements consuming the most CPU time is shown near the top of the report. For our example, we will examine the activity for statement 10.

To identify potential opportunities for performance improvement, we need to look at not only the statement text, but also the actual action diagram code. Because we are examining the activity for the IEF statements around statement 10, we will look at the text for both statements 10 and 11 and then examine the action diagram code around these statements.

As shown in Figure 4-4, the text for statement 10 indicates that one or more customer records are being read. Looking at the text for statement 11, we see a SET assignment action, indicating that the action diagram is assigning a value to some attribute view. SET assignment actions either establish an initial value for a view or change an existing value.

Figure 4-4. IEF Action Diagram Code

```
Action Block Description:

      +- TIMEFUNC
      |    IMPORTS:
      |      Entity View import customer_time (optional,transient,import only)
      |        date (optional)
      |        time (optional)
      |        name (optional)
      |        id (optional)
   .
   .
   .
      |    EXPORTS:
      |    LOCALS:
      |      Entity View local customer_information
      |        time
      |        id
      |       Work View   ief_supplied
      |        select_char
      |        count
      |    ENTITY ACTIONS:
   .
   .
   .
    1 |  +- IF COMMAND IS EQUAL TO return
    2 |  |  EXIT STATE IS return_to_caller
    3<------ESCAPE
    1 |  +--
    4 |  SET ief_supplied count TO 1
    5 |
    6 |
    7 |  += WHILE ief_supplied count IS LESS THAN 1000
    8 |  |  SET ief_supplied count TO
    8 |  |               ief_supplied count + 1
    9 |  |
   10 |  |  +- READ customer_information
   10 |  |  |         WHERE DESIRED
   10 |  |  |               customer_information id
   10 |  |  |               IS EQUAL TO "1111111111"
   10 |  |  +- WHEN successful
   11 |  |  |  SET ief_supplied time_value TO
   11 |  |  |               timetimestamp(
   11 |  |  |               CURRENT_TIMESTAMP)
```

Looking at the code (Figure 4-4) around statement 10, notice that statement 7 is the beginning of a loop. This WHILE statement reads one or more customer records and eventually calls an IEF-supplied time function, (CURRENT_TIMESTAMP) shown in statement 11, storing the result in the time_value field.

By turning to the Attribution of CPU Execution Time Report (Figure 4-5), and scanning through the reports, we can verify that the current time routine being called is in fact consuming a significant amount of CPU time.

Figure 4-5. Attribution of CPU Execution Time Report

```
                                        ** ATTRIBUTION OF CPU EXECUTION TIME **
CUSTEST1 .COBLIB  IGZEBST   BOOTSTRAP
-------------------------WAS INVOKED BY--------------------   -----------------VIA----------------------  CPU TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                SOLO  TOTAL

CUSTEST1 CUSTEST1 .IEFLIB  TIRDMGR    IEF DIALOG MANAGER                                                     .01   .01
CUSTEST1 CUSTEST1 DATEFUNC   002798   1796      CALL                                                        .01   .01
CUSTEST1 TSOAE    .IEFLIB  TIRTSOCA   IEF TSO APPL EXEC ENV                                                .02   .02
                                                                                                          ----- -----
                                                                                                           .05   .05
CUSTEST1 .IEFLIB  TIRDAT2   DATE HANDLING
-------------------------WAS INVOKED BY--------------------   -----------------VIA----------------------  CPU TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                SOLO  TOTAL

.NONDB2  CUSTEST1 RDACUSTS   0016E0                                                                        .01   .01
CUSTEST1 CUSTEST1 .IEFLIB  TIRDMGR    IEF DIALOG MANAGER                                                     .01   .01
CUSTEST1 CUSTEST1 DATEFUNC   0022A4   1654      CALL                                                       4.73  4.73
CUSTEST1 CUSTEST1 DATEFUNC   00231C   1662      CALL              CUSTEST1 TIRFTMTS TIMETIMESTAMP FUNCTION  2.38  2.38
CUSTEST1 CUSTEST1 DATEFUNC   002720   1788      CALL                                                       4.45  4.45
CUSTEST1 CUSTEST1 DATEFUNC   002720   1788      CALL              CUSTEST1 TIRDAT2  DATE HANDLING          .01   .01
CUSTEST1 CUSTEST1 DATEFUNC   002798   1796      CALL              CUSTEST1 TIRFTMTS TIMETIMESTAMP FUNCTION  2.35  2.35
CUSTEST1 CUSTEST1 DATEFUNS   00126C                                                                        .01   .01
                                                                                                          ----- -----
                                                                                                         13.96 13.96
```

This action diagram populates a current time field on one of the data entry panels of the application. Each time a new customer record is created or updated, this current time routine is invoked, consuming additional CPU time. This could become a performance problem when the use of this routine is high.

Examining the Attribution of CPU Execution Time Report shows that the current time routine TIRDAT2 is consuming almost 14% of the total CPU time for the application. This represents a possible performance improvement opportunity: remove the call to the time function from the loop.

Figure 4-6 shows a possible solution to this problem. You have moved the call to the current time routine outside of the WHILE loop. As shown in the example, the call is now made once (in statement 4) and the result is saved in the IEF-supplied local work view time_value, eliminating subsequent calls to the time routine.

Figure 4-6. Revised IEF Action Diagram

```
Action Block Description:

     +- TIMENOFC
     |    IMPORTS:
     |      Entity View import customer_time (optional,transient,import only)
     |        id (optional)
  .
  .
  .
     |    EXPORTS:
     |    LOCALS:
     |      Entity View local customer_information
     |        time
     |        id
     |      Work View   work ief_supplied
     |        count
     |        time_value
     |    ENTITY ACTIONS:
  .
  .
  .
   1 |  +- IF COMMAND IS EQUAL TO return
   2 |  |  EXIT STATE IS return_to_caller
   3<------ESCAPE
   1 |  +--
   4 |  SET work ief_supplied time_value TO
   4 |              timetimestamp(CURRENT_TIMESTAMP)
   5 |  SET work ief_supplied count TO 1
   6 |
   7 |  += WHILE work ief_supplied count IS LESS THAN
   7 |  |              1000
   8 |  |  SET work ief_supplied count TO
   8 |  |              work ief_supplied count + 1
   9 |  |
  10 |  |  +- READ customer_information
  10 |  |  |        WHERE DESIRED
  10 |  |  |              customer_information id
  10 |  |  |              IS EQUAL TO "1111111111"
  10 |  |  +- WHEN successful
  11 |  |  |  |  SET number TO input customer_number
  12 |  |  |  |  SET time TO work ief_supplied
  12 |  |  |  |              time_value
```

After completing the changes to the action diagram, and reinstalling the application on the host, measure the application again to see how effective the changes were. We executed the application again, performing the same tasks as in the earlier example.

Figure 4-7. Measurement Session Data Report (after)

```
                                        ** MEASUREMENT SESSION DATA **
-------- JOB ENVIRONMENT --------      ------- MEASUREMENT PARAMETERS ------      ----- MEASUREMENT STATISTICS ----

 PROGRAM MEASURED   -    IKJEFT01       ESTIMATED SESSION TIME  -       2 MIN       CPS TIME PERCENT -          34.93
 JOB NAME           -      WPAKRM       TARGET SAMPLE SIZE      -      15,000       WAIT TIME PERCENT -         65.07
 JOB NUMBER         -    TSU27411       REQUEST NUMBER   (A)    -         149       RUN MARGIN OF ERROR PCT  -    .80
 STEP NAME   -  WPAIEF53.WPAIEF53       FINAL SESSION ACTION    -        QUIT       CPU MARGIN OF ERROR PCT  -   1.35

 DATE OF SESSION    -    04/01/02       SYS REQ -  SCS01                           TOTAL SAMPLES TAKEN -      15,000
 TIME OF SESSION    -    09.18.23                                                  TOTAL SAMPLES PROCESSED -  15,000
                                                                                   INITIAL SAMPLING RATE- 125.00/SEC
                                                                                   FINAL SAMPLING RATE  - 125.00/SEC
 SYSTEM -             ESA SP4.3.0       BASELINE OVERRIDE -          CUSTEST1
 DFSMS              -       1.1.0       LIBRARY -   WPA.IEF53.IVP.TSO.EXELOAD     SESSION TIME -    4 MIN 21.05 SEC
 SUBSYSTEM   -          DB2 5.1.0                                                  CPU  TIME  -      1 MIN 10.07 SEC
                          IEF 5.3       --------- REPORT PARAMETERS ---------     WAIT TIME  -      2 MIN 10.33 SEC
 CPU MODEL          -   3090-600S                                                  STRETCH TIME -    1 MIN  0.65 SEC
 SMF/SYSTEM ID -       SESA/SCS01       REPORT RESOLUTION    -       64 BYTES     SRB TIME          0 MIN  4.31 SEC
 REGION SIZE BELOW 16M  -  6,208K       SORTSIZE             -        999,999     SERVICE UNITS-              67688
 REGION SIZE ABOVE  -     32,768K       LINES/PAGE           -             60     PAGES IN-      0          OUT-  0
 PTF LVL - 3.00.FS004701/FS000000       DASD= 2.0%  DASDGAP= 5  ATTR= 0.0%      PAGING RATE  -           0.00/SEC
                                        IEF=NOSQL                                EXCPS -       4,020       15.40/SEC

 SAMPLE DATA SET  - WPA.WPAKRM.S004D001
```

Comparing the Measurement Session Data reports for both measurements (Figure 4-1 on page 4-3 and Figure 4-7), you see a significant reduction in CPU time as well as an overall decrease in elapsed time.

By examining and comparing the CPU TIME fields, we see a total decrease in CPU time of about 39 seconds.

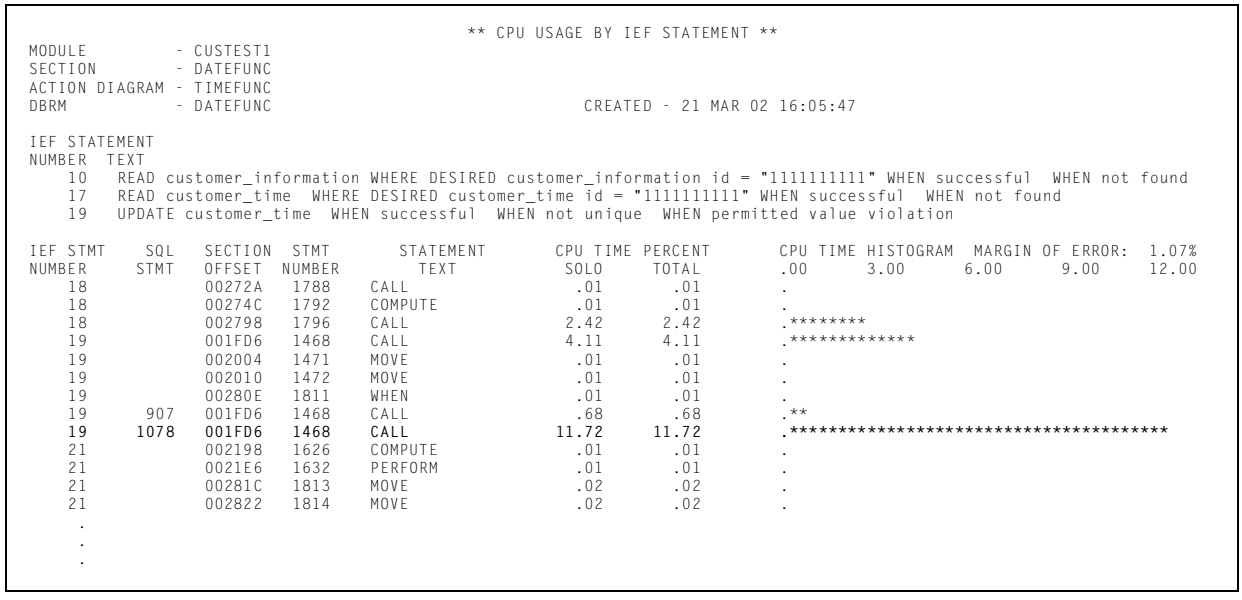
Additional Reports to Examine

Our example in the previous sections focused on the reports associated with IEF statements within an action diagram. However, you may also find that performance improvement opportunities exist in the way that your application accesses and uses system service routines, and in the way that it accesses DB2 data. STROBE provides additional reports that display this type of information. This section describes these reports.

In addition to examining the reports that are specific to action diagrams and statements, you may also want to see the SQL statement text generated on behalf of a particular IEF statement. If there are SQL statements associated with an IEF statement, the SQL statement numbers appear on the CPU Usage by IEF Statement, as shown in Figure 4-8.

As you can see in this report, SQL statement number 1078 is the highest consumer of CPU time.

Figure 4-8. CPU Usage by IEF Statement Report (with SQL statement numbers)



```
                                        ** CPU USAGE BY IEF STATEMENT **
MODULE          - CUSTEST1
SECTION         - DATEFUNC
ACTION DIAGRAM - TIMEFUNC
DBRM            - DATEFUNC                              CREATED - 21 MAR 02 16:05:47

IEF STATEMENT
NUMBER  TEXT
    10   READ customer_information WHERE DESIRED customer_information id = "1111111111" WHEN successful  WHEN not found
    17   READ customer_time  WHERE DESIRED customer_time id = "1111111111" WHEN successful  WHEN not found
    19   UPDATE customer_time  WHEN successful  WHEN not unique  WHEN permitted value violation

IEF STMT    SQL   SECTION  STMT       STATEMENT       CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.07%
NUMBER     STMT   OFFSET  NUMBER        TEXT          SOLO     TOTAL        .00      3.00      6.00      9.00     12.00
    18            00272A   1788    CALL                .01       .01        .
    18            00274C   1792    COMPUTE             .01       .01        .
    18            002798   1796    CALL               2.42      2.42        .********
    19            001FD6   1468    CALL               4.11      4.11        .*************
    19            002004   1471    MOVE                .01       .01        .
    19            002010   1472    MOVE                .01       .01        .
    19            00280E   1811    WHEN                .01       .01        .
    19     907    001FD6   1468    CALL                .68       .68        .**
    19    1078    001FD6   1468    CALL              11.72     11.72        .****************************************
    21            002198   1626    COMPUTE             .01       .01        .
    21            0021E6   1632    PERFORM             .01       .01        .
    21            00281C   1813    MOVE                .02       .02        .
    21            002822   1814    MOVE                .02       .02        .
     .
     .
     .
```

To view the SQL statement text, and the CPU activity associated with the statement, locate the DBRM name noted at the top of the CPU Usage by IEF Statement report. The DBRM name in our example is DATEFUNC, as shown in Figure 4-8. Then, refer to the CPU Use by SQL Statement report (Figure 4-9) for the DBRM.

Figure 4-9. CPU Usage by SQL Statement Report

```
                                        ** CPU USAGE BY SQL STATEMENT **

DBRM - DATEFUNC                                CREATED - 21 MAR 02 15:05:47

   STATIC, NON-CURSOR SQL
           841 UPDATE "CUSTOMER_INFORMATI" SET "TIMEO"=:H WHERE "ID"=:H
          1078 UPDATE "CUSTOMER_TIME" SET "TIMEO"=:H WHERE "ID"=:H

  STATIC, CURSOR SQL
             01 DECLARE CUR_0001638428_1 CURSOR FOR SELECT CUSTOMER_TIME01."DATEO",CUSTOMER_TIME01."TIMEO",CUSTOMER_
                   TIME01."NAME",CUSTOMER_TIME01."ID" FROM "CUSTOMER_TIME" CUSTOMER_TIME01 WHERE CUSTOMER_TIME01."ID"='
                   1111111111' FOR UPDATE OF "DATEO" OPTIMIZE FOR 1 ROW
             02 DECLARE CUR_0001376284_1 CURSOR FOR SELECTCUSTOMER_INFORMA01."TIMEO",CUSTOMER_INFORMA01."ID",CUSTOM
                   ER_INFORMA01."LAST_CONTACT",CUSTOMER_INFORMA01."PHONE",CUSTOMER_INFORMA01."STATE",CUSTOMER_
                   INFORMA01."CITY",CUSTOMER_INFORMA01."STREET",CUSTOMER_INFORMA01."CONTACT",CUSTOMER_INFORMA01
  STMT   STATEMENT                                    CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.07%
 NUMBER    TEXT                                        SOLO     TOTAL        .00      3.50      7.00     10.50     14.00
    841 UPDATE                                       11.33     11.33        .*********************************
   1078 UPDATE                                       12.66     12.66        .************************************
    712 OPEN                    02 DECLARE            1.00      1.00        .**
    904 OPEN                    01 DECLARE            1.32      1.32        .***
    715 FETCH                   02 DECLARE            8.07      8.07        .**********************
    907 FETCH                   01 DECLARE            8.46      8.46        .************************
    709 CLOSE                   02 DECLARE             .53       .53        .*
    901 CLOSE                   01 DECLARE             .43       .43        .*
                                                     -----     -----
 DBRM - DATEFUNC                             TOTALS  43.80     43.80
```

Examining these additional reports can reveal performance improvement opportunities other than the ones uncovered by examining the IEF statement text. Because the SQL statement numbers, and the CPU activity associated with them, are identified on the CPU Usage by IEF Statement report, it is relatively easy to identify which SQL statements within an action diagram are consuming the most resources. You are then able to focus your attention on a few SQL statements, possibly out of hundreds for a very large application.

For example, let's say you have identified a number of SQL statements on the CPU Usage by IEF Statement report that are consuming a relatively large amount of CPU time. By examining the text for these statements, you discover that they are all SELECT statements that have WHERE clauses requiring data from multiple tables, resulting in a JOIN operation. This operation can be expensive in terms of CPU time. Further investigation of the action diagram code reveals that several of these JOIN operations are unnecessary.

With this information, you could then explore other coding alternatives that may offer a performance improvement. In this case, it may be possible to revise the views defined in the action diagram, eliminating the unnecessary JOINs.

For more information on interpreting the SQL statement reports, see the *STROBE DB2 Feature*.

Appendix A.
The STROBE COOL:Gen Report Reference

This appendix describes the reports that are specific to the STROBE COOL:Gen Feature. The reports shown in this appendix were created using the STROBE/ISPF interface. For a complete description of the STROBE Performance Profile, see *STROBE MVS Concepts and Facilities*. For information about creating Performance Profiles using STROBE/ISPF, see the *STROBE MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management*.

This appendix describes the following reports:

- IEF Action Diagram CPU Summary
- CPU Usage by IEF Statement
- IEF Action Diagram Wait Summary
- Wait Time by IEF Statement
- Attribution Reports for COOL:Gen Applications

Note: The STROBE COOL:Gen Feature supports the measurement and analysis of applications developed with IEF (Release 5.2), Composer by IEF (Release 5.3), Composer 3 (Release 5.3.2), Composer 4, Composer 5, COOL:Gen 4.1a, COOL:Gen 4.1.1, and COOL:Gen 5.0 executed in an MVS environment. This document uses the term COOL:Gen to refer to Advantage GEN, COOL:Gen, Composer, and IEF applications. Although the reports described in this appendix reference IEF action diagrams and IEF statements, they also apply to Composer action diagrams and action diagram statements, as well as COOL:Gen action diagrams and action diagram statements

IEF Action Diagram CPU Summary Report

Figure A-1. IEF Action Diagram CPU Summary Report

```
                                    ** IEF ACTION DIAGRAM CPU SUMMARY **
MODULE    SECTION   ACTION DIAGRAM                  CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.07%
NAME      NAME      NAME                             SOLO     TOTAL        .00     23.00     46.00     69.00     92.00

CUSTEST1  CRCUSTOM  CREATE_CUSTOMER                   .02       .02        .
CUSTEST1  DATEFUNC  TIMEFUNC                        90.39     90.41        .****************************************
CUSTEST1  DELCUSTO  DELETE_CUSTOMER                   .01       .01        .
CUSTEST1  RDCUSTOM  READ_CUSTOMER                     .01       .01        .
CUSTEST1  REMOVECU  DELETE_A_CUSTOMER                 .01       .01        .
                                                    -----     -----
TOTAL CPU ACTIVITY                                  90.45     90.47
```

The IEF Action Diagram CPU Summary report (Figure A-1) shows the distribution of CPU activity among the action diagrams that STROBE identified during the measurement session. Each detail line in this report shows the executing action diagram's name and its solo and total CPU usage.

The following table describes the fields in the report.

Field	Description
MODULE NAME	The name of the load module for the COOL:Gen application.
SECTION NAME	The name of the control section within the module identified in MODULE NAME.
ACTION DIAGRAM NAME	The name of the action diagram to which STROBE associates the CPU usage it observes.
CPU TIME PERCENT SOLO	The SOLO column value indicates the portion of CPU activity for the action diagram that occurs while there is no concurrent I/O or CPU activity for the COOL:Gen application.
CPU TIME PERCENT TOTAL	The TOTAL column value indicates the portion of CPU activity for the action diagram that occurs with or without concurrent I/O or CPU activity for the COOL:Gen application.
CPU TIME HISTOGRAM	The proportion of the total CPU usage that an action diagram consumes. Solo CPU time is indicated by the symbol "*". The remaining CPU time is indicated by the symbol "+". The longest lines or "spikes" in the histogram indicate the action diagram generating the highest CPU activity.
MARGIN OF ERROR	The margin of error that can be applied to the CPU time percent values. For example, if the margin of error for the report is 2%, a reported value of 10% for CPU time total could actually be as high as 12% or as low as 8%. The margin of error applies only to the samples STROBE took when the CPU was active.

CPU Usage by IEF Statement Report

Figure A-2. CPU Usage by IEF Statement Report

```
                                          ** CPU USAGE BY IEF STATEMENT **
MODULE          - CUSTEST1
SECTION         - DATEFUNC
ACTION DIAGRAM - TIMEFUNC
DBRM            - DATEFUNC                              CREATED - 12 MAR 02 22:51:52

IEF STATEMENT
NUMBER  TEXT

    10  READ work customer_information WHERE DESIRED work customer_information id = "1111111111" WHEN succes
        sful WHEN not found
    11  UPDATE work customer_information WHEN successful MP) WHEN not unique WHEN permitted value violation

    16  READ work customer_time WHERE DESIRED work customer_time id = "1111111111" WHEN successful WHEN not
        found
    17  UPDATE work customer_timeWHEN successful AMP) WHEN not unique WHEN permitted value violation

IEF STMT    SQL   SECTION  STMT       STATEMENT       CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  1.88%
NUMBER     STMT   OFFSET  NUMBER        TEXT           SOLO     TOTAL        .00      3.50      7.00     10.50     14.00

    10            0017B4   1074    CALL                3.56      3.56        .**********
    10            0017F2   1079    CALL                3.16      3.19        .*********
    10            001840   1094    CALL                3.82      3.82        .**********
    10      708   0017F2   1079    CALL                1.28      1.28        .***
    10      711   001840   1094    CALL                8.70      8.70        .************************
    10     1074   0017B4   1074    CALL                 .92       .92        .**
    11            001B06   1217    CALL                3.34      3.38        .*********
    11            002286   1650    CALL                3.12      3.12        .********
    11            0022FE   1658    CALL                2.94      2.94        .********
    11      711   001B06   1217    CALL                 .95       .95        .**
    11      837   001B06   1217    CALL               13.18     13.18        .**************************************
    16            001C12   1274    CALL                3.05      3.05        .********
    16            001C50   1279    CALL                3.16      3.16        .*********
    16            001C9E   1289    CALL                4.11      4.15        .***********
    16      837   001C12   1274    CALL                1.10      1.10        .***
    16      900   001C50   1279    CALL                1.28      1.28        .***
    16      900   001C9E   1289    CALL                 .92       .92        .**
    16      903   001C9E   1289    CALL                9.10      9.10        .*************************
    17            001FB8   1464    CALL                2.79      2.79        .*******
    17            0026EA   1782    CALL                3.01      3.01        .********
    17            002762   1790    CALL                2.24      2.24        .******
    17      903   001FB8   1464    CALL                1.10      1.10        .***
    17     1074   001FB8   1464    CALL               12.19     12.22        .**********************************
                                                       -----     -----
DIAGRAM - TIMEFUNC                          TOTALS     89.02     89.17
```

The CPU Usage by IEF Statement report (Figure A-2) shows the distribution of CPU activity among IEF/COOL:Gen statements within a specific action diagram.

The following table describes the fields in the report.

Field	Description
MODULE	The name of the load module for the COOL:Gen application (procedure).
SECTION	The name of the control section within the module identified in MODULE (member).
ACTION DIAGRAM	The name of the action diagram to which STROBE associates the CPU usage it observes.
DBRM	If DB2 is executing in the same environment as the COOL:Gen application, the DBRM module being invoked by the COOL:Gen-generated SQL code.

IEF STATEMENT NUMBER\TEXT	If you included an IEF/COOL:Gen map data set when you created the Performance Profile, the statement number with which STROBE associates the CPU activity it observes and the statement text. The number of statements that appear depend upon the value you specified for IEF=TOPTEXT. The default is the three statements that consumed the most CPU or run time.
IEF STATEMENT NUMBER	The IEF/COOL:Gen statement number with which STROBE associates the CPU activity it observes.
SQL STMT	If DB2 is executing in the same environment as the COOL:Gen application, the SQL statement that was invoked by the code generated by COOL:Gen. SQL statement numbers can be suppressed on this report by specifying the IEF=NOSQL parameter.
SECTION OFFSET	The program section offset for the COOL:Gen-generated COBOL statement using CPU time. Values appear in this field only if you specified the IEF=OFFSET parameter when creating the Performance Profile.
STMT NUMBER	The STMT number of the COBOL statement that was generated on behalf of an IEF/COOL:Gen statement. This information is present only if the COBOL map data set was included when the Performance Profile was created.
STATEMENT TEXT	The COBOL statement text that was generated on behalf of an IEF/COOL:Gen statement. This information is present only if the COBOL map data set was included when the Performance Profile was created.
CPU TIME PERCENT SOLO	The SOLO column value indicates the portion of CPU activity for the action diagram statement that occurs while there is no concurrent I/O or CPU activity for the COOL:Gen application.
CPU TIME PERCENT TOTAL	The TOTAL column value indicates the portion of CPU activity for the action diagram statement that occurs with or without concurrent I/O or CPU activity for the COOL:Gen application.
CPU TIME HISTOGRAM	The proportion of the total CPU time that an action diagram statement consumes. Solo CPU time is indicated by the symbol "*". The remaining CPU time is indicated by the symbol "+". The longest lines or "spikes" in the histogram indicate the control sections generating the highest CPU activity.
MARGIN OF ERROR	The margin of error that can be applied to the CPU time percent values. For example, if the margin of error for the report is 2%, a value of 10% for CPU time total could be as high as 12% or as low as 8%. The margin of error applies only to the samples STROBE took when the CPU was active.

IEF Action Diagram Wait Summary Report

Figure A-3. IEF Action Diagram Wait Summary Report

```
                                   ** IEF ACTION DIAGRAM WAIT SUMMARY **
MODULE    SECTION   ACTION DIAGRAM                  RUN TIME PERCENT       RUN TIME HISTOGRAM  MARGIN OF ERROR:  .69%
NAME      NAME      NAME                             PAGE     TOTAL        .00       2.00      4.00      6.00     8.00

UPDTCUST  UPCUS     ABUPDATE                         3.00       3.01        .*********
UPDTCUST  READCUS   ABUPREAD                         1.09       1.11        .**************
                                                     -----     -----
TOTAL WAIT ACTIVITY                                  4.09       4.11
```

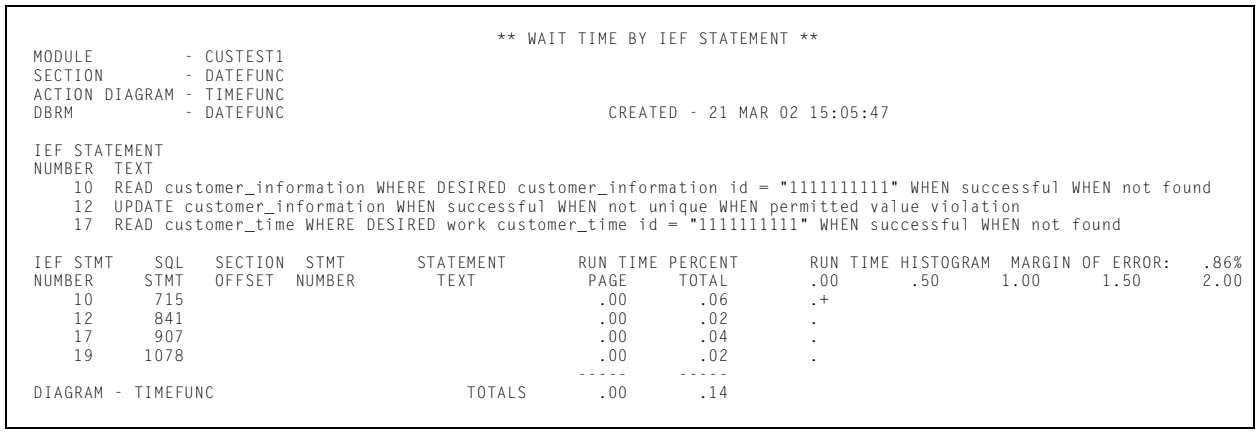
The IEF Action Diagram Wait Summary report (Figure A-3) identifies the total wait time by IEF/COOL:Gen action diagram.

The following table describes the fields in the report.

Field	Description
MODULE NAME	The name of the load module for the COOL:Gen application.
SECTION NAME	The name of the control section within the module identified in MODULE NAME.
ACTION DIAGRAM NAME	The name of the action diagram to which STROBE associates wait time.
RUN TIME PERCENT PAGE	The percentage of run or wait time for the action diagram that is caused by paging.
RUN TIME PERCENT TOTAL	The total percent of run-time that the action diagram waited because of paging and other system activities.
RUN TIME HISTOGRAM	The proportion of the total run-time that a control section consumes. Page run-time is indicated by the symbol "*". The remaining run-time is indicated by the symbol "+". The longest lines or "spikes" in the histogram indicate the control sections generating the highest amount of run-time.
MARGIN OF ERROR	The margin of error that can be applied to the run-time values. For example, if the margin of error for the report is 2%, a reported value of 10% for run-time total could actually be as high as 12% or as low as 8%. The margin of error applies only to the samples STROBE took when the application was waiting.

Wait Time by IEF Statement Report

Figure A-4. Wait Time by IEF Statement Report



```
                                        ** WAIT TIME BY IEF STATEMENT **
MODULE          - CUSTEST1
SECTION         - DATEFUNC
ACTION DIAGRAM - TIMEFUNC
DBRM            - DATEFUNC                              CREATED - 21 MAR 02 15:05:47

IEF STATEMENT
NUMBER  TEXT
    10  READ customer_information WHERE DESIRED customer_information id = "1111111111" WHEN successful WHEN not found
    12  UPDATE customer_information WHEN successful WHEN not unique WHEN permitted value violation
    17  READ customer_time WHERE DESIRED work customer_time id = "1111111111" WHEN successful WHEN not found

IEF STMT    SQL   SECTION  STMT       STATEMENT       RUN TIME PERCENT       RUN TIME HISTOGRAM  MARGIN OF ERROR:   .86%
NUMBER     STMT   OFFSET  NUMBER        TEXT           PAGE     TOTAL        .00       .50      1.00      1.50      2.00
    10      715                                         .00       .06        .+
    12      841                                         .00       .02        .
    17      907                                         .00       .04        .
    19     1078                                         .00       .02        .
                                                      -----     -----
DIAGRAM - TIMEFUNC                        TOTALS       .00       .14
```

The Wait Time by IEF Statement report (Figure A-4) provides detail information for each identified IEF statement within an action diagram.

The following table describes the fields in the report.

Field	Description
MODULE	The name of the load module for the COOL:Gen application.
SECTION	The name of the control section within the module identified in MODULE.
ACTION DIAGRAM	The name of the action diagram that contains the IEF/COOL:Gen statements for which STROBE has associated wait time.
DBRM	If DB2 is executing in the same environment as the COOL:Gen application, the DBRM module being invoked by the COOL:Gen-generated SQL code.
IEF STATEMENT NUMBER	The IEF/COOL:Gen statement number with which STROBE associates the wait time it observes.
SQL STMT	If DB2 is executing in the same environment as the COOL:Gen application, the SQL statement that was invoked by the COOL:Gen-generated code.
SECTION OFFSET	The program section offset for the COBOL statement that is waiting.
STMT NUMBER	The STMT number of the COBOL statement that was generated on behalf of an IEF/COOL:Gen statement. This information is present only if the COBOL map data set was included when the Performance Profile was created.

STATEMENT TEXT	The COBOL statement text that was generated on behalf of an IEF/COOL:Gen statement. This information is present only if the COBOL map data set was included when the Performance Profile was created.
RUN TIME PERCENT PAGE	The percentage of run or wait time for the action diagram that is caused by paging.
RUN TIME PERCENT TOTAL	The total percent of run-time that the action diagram waited because of paging and other system activities.
RUN TIME HISTOGRAM	The proportion of the total run-time that a control section consumes. Page run-time is indicated by the symbol "*". The remaining run-time is indicated by the symbol "+". The longest lines or "spikes" in the histogram indicate the control sections generating the highest amount of run-time.
MARGIN OF ERROR	The margin of error that can be applied to the run-time values. For example, if the margin of error for the report is 2%, a reported value of 10% for run-time total could actually be as high as 12% or as low as 8%. The margin of error applies only to the samples STROBE took when the application was waiting.

Attribution Reports for IEF Applications

Figure A-5. Attribution of CPU Execution Time Reports for COOL:Gen Applications

```
                                        ** ATTRIBUTION OF CPU EXECUTION TIME **
.SVC     SVC 009            PROGRAM MANAGER/DELETE
--------------------------WAS INVOKED BY---------------------   -----------------VIA-----------------------  CPU TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                 SOLO  TOTAL

CUSTEST1 CUSTEST1 .IEFLIB  TIRDMGR    IEF DIALOG MANAGER          CUSTEST1 TIRPROF  IEF PROFILE MANAGER        .04   .04
CUSTEST1 CUSTEST1 DATEFUNC   0017B4   1074      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .70   .70
CUSTEST1 CUSTEST1 DATEFUNC   0017F2   1079      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .37   .37
CUSTEST1 CUSTEST1 DATEFUNC   001840   1094      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY  1.25  1.25
CUSTEST1 CUSTEST1 DATEFUNC   001B06   1217      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .92   .92
CUSTEST1 CUSTEST1 DATEFUNC   001C12   1274      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .73   .73
CUSTEST1 CUSTEST1 DATEFUNC   001C50   1279      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .37   .37
CUSTEST1 CUSTEST1 DATEFUNC   001C9E   1289      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY  1.10  1.10
CUSTEST1 CUSTEST1 DATEFUNC   001FB8   1464      CALL              CUSTEST1 TIRHLI   DB2 CALL ATTACH FACILITY   .77   .77
                                                                                                             ----- -----
                                                                                                             6.24  6.24
CUSTEST1 .IEFLIB  TIRDAT2   DATE HANDLING
--------------------------WAS INVOKED BY---------------------   -----------------VIA-----------------------  CPU TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                 SOLO  TOTAL

CUSTEST1 CUSTEST1 DATEFUNC   002286   1650      CALL                                                        3.08  3.08
CUSTEST1 CUSTEST1 DATEFUNC   002286   1650      CALL              CUSTEST1 TIRDAT2  DATE HANDLING           .04   .04
CUSTEST1 CUSTEST1 DATEFUNC   0022FE   1658      CALL              CUSTEST1 TIRFTMTS TIMETIMESTAMP FUNCTION  2.75  2.75
CUSTEST1 CUSTEST1 DATEFUNC   0026EA   1782      CALL                                                        3.01  3.01
CUSTEST1 CUSTEST1 DATEFUNC   002762   1790      CALL              CUSTEST1 TIRFTMTS TIMETIMESTAMP FUNCTION  2.09  2.09
                                                                                                             ----- -----
                                                                                                            10.98 10.98
CUSTEST1 .IEFLIB  TIRHLI    DB2 CALL ATTACH FACILITY
--------------------------WAS INVOKED BY---------------------   -----------------VIA-----------------------  CPU TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                 SOLO  TOTAL

CUSTEST1 CUSTEST1 .IEFLIB  TIRDMGR    IEF DIALOG MANAGER          CUSTEST1 TIRPROF  IEF PROFILE MANAGER        .04   .04
CUSTEST1 CUSTEST1 DATEFUNC   0017B4   1074      CALL                                                        2.75  2.75
CUSTEST1 CUSTEST1 DATEFUNC   0017F2   1079      CALL                                                        2.72  2.72
CUSTEST1 CUSTEST1 DATEFUNC   001840   1094      CALL                                                        2.46  2.46
CUSTEST1 CUSTEST1 DATEFUNC   001B06   1217      CALL                                                        2.28  2.31
CUSTEST1 CUSTEST1 DATEFUNC   001C12   1274      CALL                                                        2.20  2.20
CUSTEST1 CUSTEST1 DATEFUNC   001C50   1279      CALL                                                        2.68  2.68
CUSTEST1 CUSTEST1 DATEFUNC   001C9E   1289      CALL                                                        2.94  2.97
CUSTEST1 CUSTEST1 DATEFUNC   001FB8   1464      CALL                                                        1.98  1.98
                                                                                                             ----- -----
                                                                                                            20.04 20.12
```

The Attribution of CPU Execution Time reports (Figure A-5) identify the sites of invocation of system routines. Use these reports to identify which modules/offsets caused activity in a system service routine.

Each section of the attribution report begins with a header line that identifies the invoked service routine. The header also shows the pseudo-module name, module name, control section name (when available), and a function descriptor for either the control section or the module. The .IEFLIB pseudo-module name indicates an IEF-supplied service routine.

The following table describes the fields in the report.

Field	Description
WAS INVOKED BY XACTION	If there is transaction activity in the environment where the COOL:Gen application is running, the module or pseudo-module, or the transaction or pseudo-transaction, invoking the transaction.
WAS INVOKED BY MODULE	The name of the load module for the COOL:Gen application.
WAS INVOKED BY SECTION	The name of the control section, or pseudo-section, within the module.
WAS INVOKED BY RETURN	The return address of the statement that invoked the service routine or the control section name for the pseudo-section shown in the WAS INVOKED BY SECTION column.
WAS INVOKED BY LINE	The line number for the statement that invoked the service routine or the function descriptor for the pseudo-section shown in the WAS INVOKED BY SECTION column. This field does not appear when no map data sets were specified when the Performance Profile was created.
WAS INVOKED BY PROCEDURE NAME	The procedure name associated with the statement using the CPU time.
VIA MODULE	The module that invoked the service routine if it was not directly invoked.
VIA SECTION	The section within the module that invoked the service routine if it was not directly invoked.
VIA FUNCTION	A brief description of the function of the control section or module that invoked the service routine if it was not directly invoked.
CPU TIME PERCENT SOLO	The SOLO column value indicates the portion of CPU activity for the offset that occurs while there is no concurrent CPU or I/O activity for the COOL:Gen application.
CPU TIME PERCENT TOTAL	The TOTAL column value indicates the portion of CPU activity for the offset that occurs with or without concurrent CPU or I/O activity for the COOL:Gen application.

Figure A-6. Attribution of CPU Wait Time Reports for COOL:Gen Applications

```
                                        ** ATTRIBUTION OF CPU WAIT TIME **
.SVC     SVC 008            PROGRAM MANAGER/LOAD
--------------------------WAS INVOKED BY--------------------   -----------------VIA---------------------- WAIT TIME %
XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                 PAGE  TOTAL

.NONDB2  WPATRH1  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .02
.NONDB2  .DB2     DSNALI             LANGUAGE INTERFACE                                                      .00   .01
.NONDB2  .DB2     DSNECP10           CNTL CSECT FOR DSNECP10                                                 .00   .01
.NONDB2  READCUST            00B84A                              IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .32
.NONDB2  READCUST            00B904                                                                          .00   .04
.NONDB2  TSOAE    .IEFLIB  TIRTSOCA   IEF TSO APPL EXEC ENV                                                  .00   .49
.NONDB2  TSOAE    .IEFLIB  TIRTSO2O   IEF LIBRARY SUBROUTINE                                                 .00   .01
.NONDB2  TSOAE    .SASCLIB L$CMAINO   SAS C LIBRARY SUBROUTINE                                               .00   .01
.NONDB2  TSOAE    .SASCLIB L$CSYLDO   SAS C LIBRARY SUBROUTINE                                               .00   .05
.NONDB2  UPDTCUST            00B232                              IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .13
.NONDB2  UPDTCUST            00B2EC                                                                          .00   .02
.NONDB2  WPATRH1  .COBLIB  IGZEBST    BOOTSTRAP                                                              .00   .05
.NONDB2  WPATRH1  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  AELOAD   ENVIRONMENT INITIALIZE     .00   .01
.NONDB2  WPATRH1  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .30
.NONDB2  WPATRH2             00B3E2                              IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .15
.NONDB2  WPATRH2             00B49C                                                                          .00   .02
.NONDB2  WPATRH3  .COBLIB  IGZEBST    BOOTSTRAP                                                              .00   .01
.NONDB2  WPATRH3  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .17
.NONDB2  WPATRH4  .COBLIB  IGZEBST    BOOTSTRAP                                                              .00   .04
.NONDB2  WPATRH4  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  AELOAD   ENVIRONMENT INITIALIZE     .00   .01
.NONDB2  WPATRH4  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .32
.NONDB2  WPATRH5  .COBLIB  IGZEBST    BOOTSTRAP                                                              .00   .02
.NONDB2  WPATRH5  .COBLIB  IGZEBST    BOOTSTRAP                  IGZEINI  IGZEINI  ENVIRONMENT INITIALIZE     .00   .15
                                                                                                           ----- -----
                                                                                                            .00  2.29
-
-
```

The Attribution of CPU Wait Time report (Figure A-6) identifies the sites of invocation of system routines. When wait time is attributed to a COOL:Gen service routine, the report header displays the invoked routine, showing:

- its pseudo-module, module, and control section name (when available)
- a function descriptor for either the control section or the module.

The following table describes the fields in the report.

Field	Description
WAS INVOKED BY XACTION	If there is transaction activity in the environment where the COOL:Gen application is running, the module or pseudo-module invoking the transaction.
WAS INVOKED BY MODULE	The name of the load module for the COOL:Gen application.
WAS INVOKED BY SECTION	The name of the control section, or pseudo-section, within the module.
WAS INVOKED BY RETURN	The return address of the statement that invoked the service routine or the control section name for the pseudo-section shown in the WAS INVOKED BY SECTION column.
WAS INVOKED BY LINE	The line number for the statement that invoked the service routine or the function descriptor for the pseudo-section shown in the WAS INVOKED BY SECTION column.
WAS INVOKED BY PROCEDURE NAME	The procedure name invoked by the module causing the wait time.
VIA MODULE	The module that invoked the service routine if it was not directly invoked.

VIA SECTION	The section within the module that invoked the service routine if it was not directly invoked.
VIA FUNCTION	A brief description of the function of the control section or module that invoked the service routine if it was not directly invoked.
WAIT TIME PERCENT PAGE	The percentage of wait time for the statement that was caused by paging.
WAIT TIME PERCENT TOTAL	The percentage of wait time for the statement that was caused either by paging or some other system activity.

Index

A

C

E

I

L

M

P

S

U

V

W