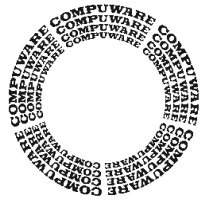
STROBE MVS

CSP 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. CWSTUP3A
January 21, 2003.

Contents

Figures

Summary of Changes

The following sections describe changes to the CSP Feature and to the Manual.

Changes to the CSP Feature

The STROBE CSP Feature has not changed from STROBE MVS for Sysplex Release 2.5.0 to Release 3.0.

Changes to this Manual

The technical content of this manual did not change for STROBE MVS for Sysplex Release 3.0.

Introduction

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

The STROBE MVS Application Performance Measurement System and the STROBE *CSP Feature* are products designed for IBM MVS/ESA systems. The STROBE CSP Feature is designed for use with IBM SAA Cross System Product/370 Application Development (CSP/370AD V4.1), program number 5688-218 and IBM SAA Cross System Product/370 Runtime Services (CSP/370RS V2.1 and above), program number 5688-206.

How to Use This Manual

Before you submit a STROBE measurement request, you should read Chapter 1, "Overview" and Chapter 2, "Using the STROBE CSP Feature". For information on analyzing STROBE CSP Performance Profiles, read Chapter 3, "Analyzing the Performance Profile for a CSP Application".

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 an application spends time in online regions and batch processing programs and how the application uses system resources. STROBE collects several types of data as it tracks activity within an MVS environment and produces a collection of reports that helps you determine where to revise applications to improve their performance.

The STROBE CSP Feature extends the functions of STROBE and measures applications created with the Cross System Product/370 (CSP/370). The measurement information that you collect with the STROBE CSP Feature will help you to analyze and improve the performance of CSP-generated COBOL applications.

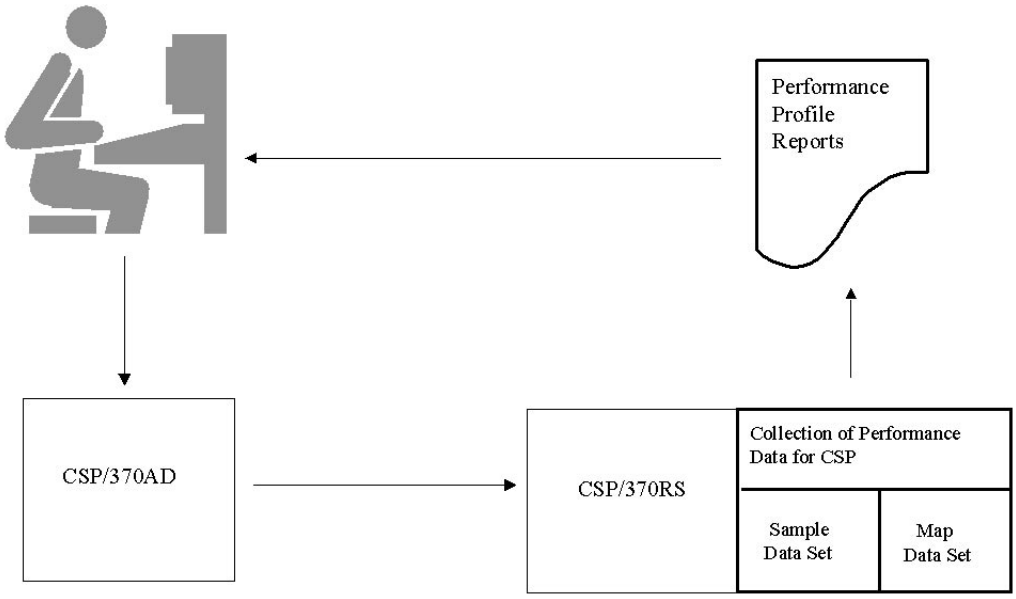
This chapter describes the benefits of the STROBE CSP Feature, defines the basic concepts and terms necessary for effectively using STROBE, and provides a brief overview of the CSP/370 application development environment.

Benefits of the STROBE CSP Feature

The STROBE CSP Feature collects application performance information as the CSP-generated application executes. When measurement completes, STROBE organizes this information into the STROBE *Performance Profile*, a series of reports that show where and how time is spent during application execution, pinpointing possible areas for performance improvement. Measuring your application at successive stages of the development cycle helps you to evaluate and improve your code before moving the applications into production.

Figure 1-1 shows the process for measuring your CSP application with the STROBE CSP Feature and realizing those benefits.

Figure 1-1. The STROBE CSP Feature with CSP/370AD and CSP/370RS



During Production

Measuring your production applications with the STROBE CSP 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, implement the changes, and measure again to verify the effects of your changes.

Once you implement these changes and the application's performance improves, the possible benefits include

- reduced occurrences of production-level performance crises
- improved efficiency of the application
- reduced batch processing time and costs
- acceptable online response time levels
- increased capacity for processing other applications

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

- Measuring your production applications 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, implement the changes, and measure again to verify the effects of your changes.
- Managing the performance of the application during the intergration test phase of development enables you to
  - deliver efficient applications that meet or exceed the expectations of your end users
  - develop performance benchmarks for your applications from the information you gather during the testing cycle

  - avoid the costs associated with introducing inefficient applications into production

- Using the STROBE CSP Feature during early testing phases such as unit tests also yields significant benefits. As soon as testing begins, you can measure the application and potentially realize many of the same benefits described in the previous section.

During Earlier Test Cycles

Using the STROBE CSP Feature during early testing phases such as unit tests yields significant benefits. As soon as testing begins, you can 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

- a repository of performance improvement information collected during other development phases
- an understanding of how certain design and coding practices affect performance

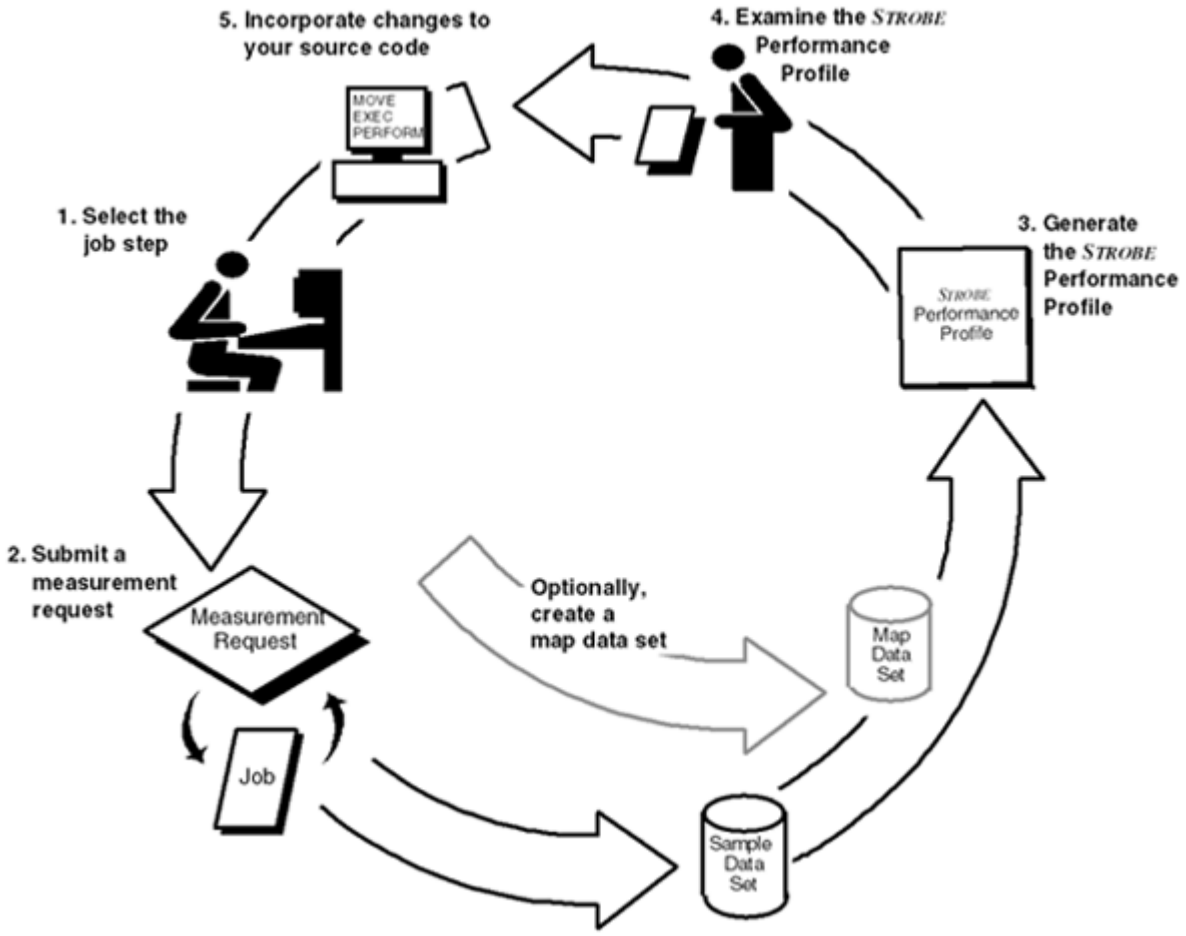
The knowledge obtained from regular application performance management in test and production enables you to systematically build efficiency into your CSP applications as you develop them.

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

Concepts and Terminology

Prior to using the STROBE CSP Feature, you need to become familiar with the basic concepts and terminology specific to STROBE. Figure 1-2 introduces you to these concepts and terms.

Figure 1-2. Overview of STROBE Tasks



A *measurement request* specifies the parameters for measuring the performance of an application. When the application is active, STROBE begins a *measurement session*, an interval where it collects performance data about the application while it is executing. STROBE 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.

After STROBE closes the sample data set, you can use it to create the *Performance Profile*, a hierarchical series of reports that presents the performance data collected during a measurement session. These reports show where and how the application spends its time during execution. Examine the Performance Profile to identify performance improvement opportunities, and incorporate the appropriate changes to your source code. You can then repeat the process to see the results of your code changes, and identify other potential performance improvement opportunities that may have been masked by the initial problem.

What is a Measurement Request?

A *measurement request* specifies the parameters for measuring the performance of an application. You can specify the parameters through STROBE/ISPF. As soon as the application is active, STROBE begins a *measurement session*, an interval during which STROBE collects performance data about the application while it is executing.

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 STROBE closes the sample data set, 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 the application spends the time during execution.

For CSP applications, the Performance Profile contains easy-to-read reports that identify CPU use by specific modules and statement numbers. The reports also identify wait time caused by specific files and devices. The Performance Profile highlights

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

By interpreting the report data, you can identify concentrations of the application's demand for resources and determine where to make changes to improve the performance of the application. Based on the information provided in the Performance Profile, you may decide to change the way your application uses resources or rewrite the source code so that the application executes certain statements 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?

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. A map data set is the repository for the information STROBE uses to relate offsets and addresses in the object code with statements in the source code. *Indexing* is the process of relating activity and wait to source code statement text.

Indexing is optional. Review the Performance Profile first to identify modules that display a significant amount of CPU activity, and then create map data sets for each of those modules.

What is Attribution?

Attribution identifies the sites of invocation of system service routines, relating activity or wait to the modules that called these routines. Examine two Performance Profile reports to get a complete picture of the overall performance of your application.

- Attribution of CPU Wait Time report
- Attribution of CPU Execution Time report

Examine these reports when system services are responsible for significant CPU use or wait time. The Attribution of CPU Wait Time report identifies the modules or service routines where the CPU activity or wait is concentrated. First examine this report, and then examine the Attribution of CPU Execution Time report to identify which modules invoked the service routines. Used in conjunction, the Attribution reports enable you to identify opportunities for performance improvement.

Overview of the CSP/370 Environment

The STROBE CSP Feature supports applications created with the CSP Application Development facility (CSP/370AD) and executed in a CSP Runtime Services (CSP/370RS) environment. The next section briefly describes these IBM Cross System Products.

Note: The STROBE CSP Feature supports an application developed with CSP/2AD as long as you generated the application with CSP/370AD.

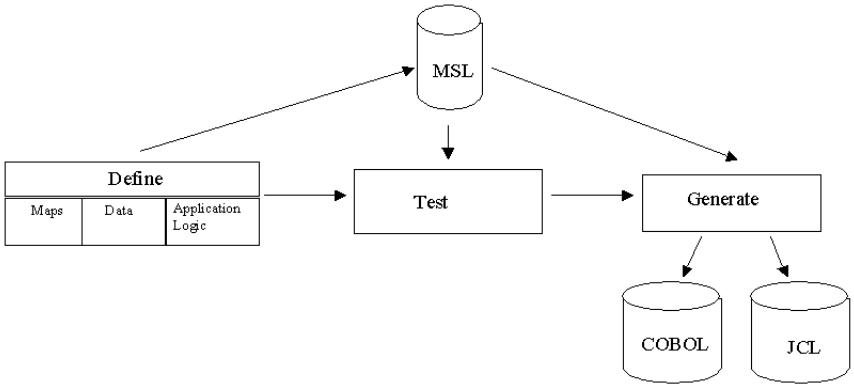
CSP/370 Application Development

CSP/370AD enables you to define, test, maintain, and generate COBOL applications for execution in the following MVS environments: CICS, IMS, TSO, and batch. To generate a COBOL application with CSP, you must first define data, maps, and application logic.

During data definition, you must define both data items and the records and tables they comprise. Record and table definitions affect the format of file or database data items and serve editing and reference functions. During map definition, you determine how the end user will view your application by defining screen and printer formats. During application logic definition, you determine how your application will process the data and map definitions.

As shown in Figure 1-3, CSP/370AD stores these definitions in a member specification library (MSL). CSP uses the definitions, or *members*, to test the application and to generate the COBOL source code and the job control language (JCL) required to compile, link, and run the application. After generating the application with CSP/370AD, run it on CSP/370RS.

Figure 1-3. Defining, Testing, and Generating CSP Applications

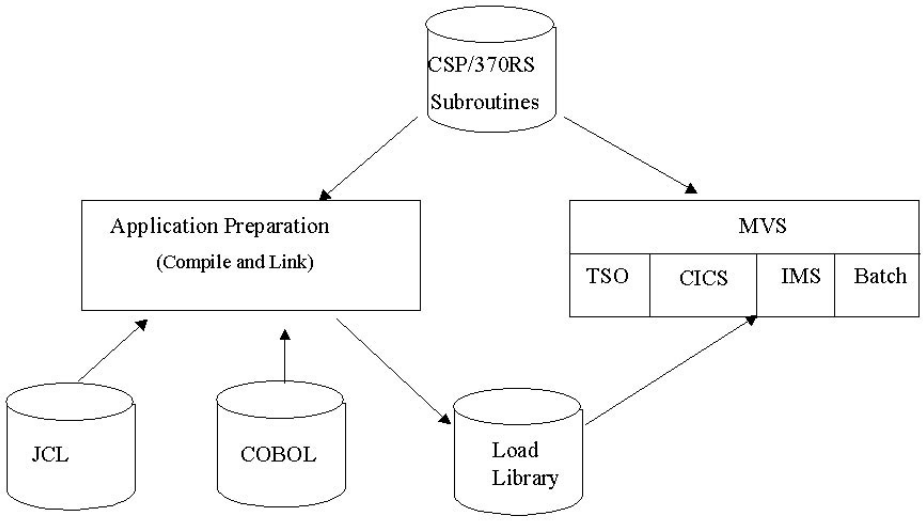


CSP/370 Runtime Services

CSP/370RS uses the COBOL source code, the JCL, and a library of subroutines to prepare the application to run in a production environment. The run-time library implements functions shared among generated applications.

As shown in Figure 1-4, CSP compiles and links the COBOL source code and JCL, and then stores them as modules within a standard Load Library. The modules, used in conjunction with the subroutines, then execute in MVS environments.

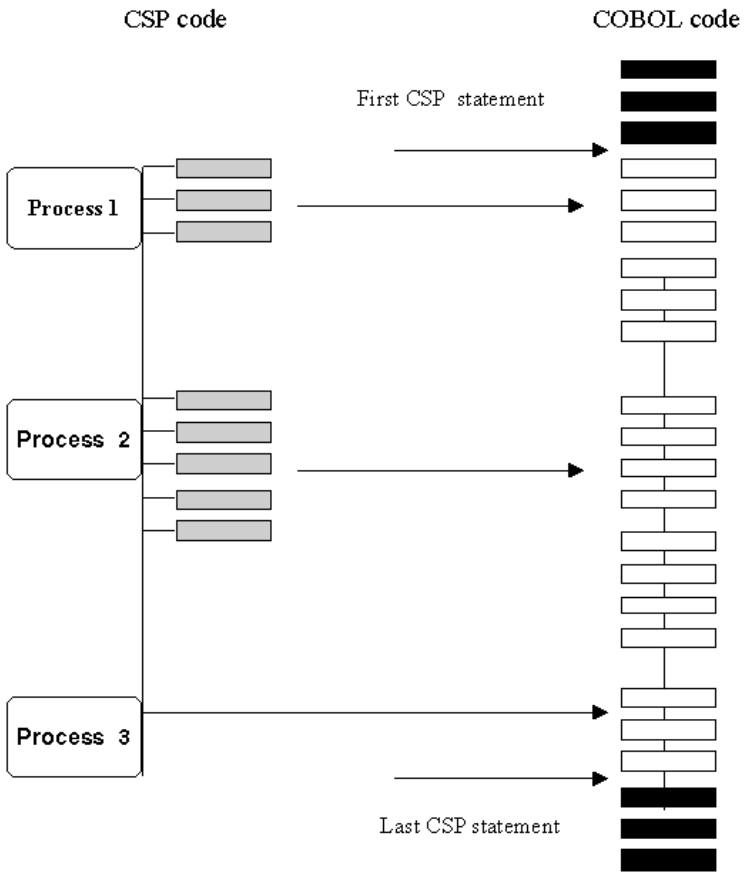
Figure 1-4. Preparing and Executing CSP Applications



Indexing with the CSP Feature

Because CSP generates COBOL code, STROBE measures COBOL modules. To analyze specific CSP statements and processes, you need to know which COBOL modules correspond to them. STROBE does associate most of these CSP statements and processes to COBOL statements. However, STROBE reports any COBOL statements it cannot identify without a CSP association. Figure 1-5 illustrates STROBE's indexing process. The CSP processes and statements shown correspond to one COBOL module.

Figure 1-5. STROBE CSP Feature Indexing



As Figure 1-5 shows, STROBE associates the COBOL code (white boxes) with specific CSP statements (gray boxes) or with general CSP processes, such as Process 3. The black boxes represent COBOL routines not associated with any CSP statements or processes. In addition, STROBE labels the first and last CSP statements for which it identified activity.

Note: The Performance Profile reports provide the generated COBOL statements to help analyze the effectiveness of the CSP code. Refer only to the CSP code and definitions to improve application performance.

Summary and Next Step

The conceptual information presented in this chapter should help you properly use the STROBE CSP Feature and maximize the effectiveness of the measurement session you submit. In addition, understanding the function of CSP/370AD and CSP/370RS will guide you through the Performance Profile reports.

The next step in improving application performance is to use the STROBE CSP Feature to measure your application. Chapter 2, “Using the STROBE CSP Feature” describes this process.

Where to Find More Information

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

For more information on...	**Refer to...**
Creating and submitting measurement requests	"Measuring the Application" on page 2-2
Creating a Performance Profile	"Creating the Performance Profile" on page 2-7
Creating map data sets	"Creating the Map Data Set" on page 2-5
Analyzing a Performance Profile	Chapter 3, "Analyzing the Performance Profile for a CSP Application"
Learning more about STROBE	*STROBE MVS Concepts and Facilities*

Chapter 2.
Using the STROBE CSP Feature

After installing your CSP application in the MVS environment, you can measure its performance. This chapter describes the procedures required to evaluate application performance, which include the following:

- determine when and how to measure your application
- measure the application
- view the status of your measurement request
- associate CPU activity or wait with COBOL statements
- create a compiler listing data set, a map data set, and a Performance Profile

Before You Begin

With STROBE you can measure applications that run as batch jobs, use TSO, or execute in online regions. This section describes conditions that indicate that you should measure a particular application with STROBE.

For a complete picture of a batch application's activity, measure it over the entire duration of the job step. Submit a measurement request for each job step of interest before the batch job executes. To ensure efficient sampling, specify an estimate of the run time for each job step.

Online applications place varying demands on a system over the course of a day. The first step in measuring an online application is to determine the period of time and the region that you are most interested in measuring. There are several approaches to measuring online regions.

- Submit one measurement request that spans the time frame in which you are most interested. This approach can help indicate trends in resource demand.
- Submit one measurement request, but specify that the request use multiple measurement sessions (more than one sample data set associated with the request). STROBE allows you to produce a Performance Profile from the closed sample data set while sampling continues in a different sample data set. This approach enables you to check program performance without interrupting the measurement process.

  For example, the transaction response time for a crucial CICS or IMS application slows down every day at the same time. In this situation you can measure the application when the response time is acceptable, and measure it again when performance degrades. By comparing the Performance Profile reports for the two measurement sessions, you can identify and examine potential areas of improvement in your online application.

After you have identified an application to measure, think about how to configure your measurement request. Decide on your session duration and target sample size.

Starting STROBE

You have three options when starting STROBE :

- STROBE/ISPF — a menu-driven interface to the STROBE system
- STROBE command language — a set of procedural operators and parameters that you can invoke through TSO commands or batch jobs

If necessary, consult your systems programmer for specific information on how to start STROBE in your particular environment.

When you submit a measurement request, STROBE automatically starts the CSP Feature. See "Disabling the STROBE CSP Feature" on page 2-3 for instructions on submitting a request *without* measuring your CSP application.

Measuring the Application

This section describes how to submit a measurement request and view the status of the request.

- Select Option 1 (ADD ACTIVE) if you want to measure a job that is currently executing. This option enables you to measure online applications, as well as batch programs that are running far longer than you expected. When you submit the request, STROBE immediately begins measuring the job step.
- Select Option 2 (ADD QUEUED) if you want to measure a job that is not yet executing. After you submit the request, STROBE will initiate a measurement task when the target job step becomes active. Because you are requesting a measurement session for a job that has not yet begun executing, you can target specific steps in a job to measure.

The following sections describe how to measure active and queued jobs using the procedures described above.

Measuring an Active Job

To measure an active job or online region, select Option 1 (ADD ACTIVE) from the STROBE OPTIONS menu. Complete the following information in the STROBE - ADD ACTIVE REQUEST panel:

1. Specify the target job name in the JOBNAME field or clear this field to select from a list of active jobs.
2. Specify the target system in the SYSTEM field or clear this field to select from a list of systems.
3. Configure the measurement session by specifying
   - how long you want to measure the job in the SESSION DURATION field
   - how many performance samples you want STROBE to take in the TARGET SAMPLE SIZE field
   - the TSO user ID for STROBE to notify when the request is complete in the TSO USER ID TO NOTIFY field
4. Specify the sample data set information in the SAMPLE DATA SET INFORMATION fields.
5. Specify any additional measurement options by entering "Y" in the corresponding field and then pressing **Enter**. You can
   - specify special data collectors

- specify additional module mapping facilities
- add session management parameters
- request a retention time frame

STROBE displays the selected panels. When you press **Enter** on the last panel, STROBE submits the measurement request. See the STROBE *MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management* for more information on these measurement options.

6. Press **Enter**.

Measuring a Job That Is Not Yet Executing

To add a measurement request for a job that is not yet executing, select Option 2 (ADD QUEUED) from the STROBE OPTIONS menu. Complete the following information in the STROBE - ADD QUEUED REQUEST panel:

1. Specify the target job step in the PROGRAM or STEP fields.
2. Specify the target job name in the JOBNAME field or clear this field to select from a list of jobs not yet executing.
3. Specify the target system in the SYSTEM field or clear this field to select from a list of systems.
4. Configure the measurement session by specifying
   - how long you want to measure the job in the SESSION DURATION field
   - how many performance samples you want STROBE to take in the TARGET SAMPLE SIZE field
   - the TSO user ID for STROBE to notify when the request is complete in the TSO USER ID TO NOTIFY field
5. Specify the sample data set information in the SAMPLE DATA SET INFORMATION fields.
6. Specify any additional measurement options by entering "Y" in the corresponding field and then pressing **Enter**. You can
   - specify special data collectors
   - specify additional module mapping facilities
   - add session management parameters
   - request a retention time frame

   STROBE displays the selected panels. When you press **Enter** on the last panel, STROBE submits the measurement request. See the STROBE *MVS User's Guide* or the *STROBE MVS User's Guide with Advanced Session Management* for more information on these measurement options.
7. Press **Enter**.

Disabling the STROBE CSP Feature

To submit a measurement request that will *not* measure your CSP application, complete the following steps in the STROBE - ADD ACTIVE REQUEST or STROBE - ADD QUEUED REQUEST panel:

1. Enter "Y" in the DATA COLLECTORS field. The STROBE - DATA COLLECTORS panel appears.
2. Enter "N" in the CSP field.
3. Press **Enter**.

Viewing the Status of Your Measurement Request

Once you have submitted a measurement request, you can view the status of the request, regardless of the system on which it is executing. You can also monitor the status of other measurement requests on other systems in the sysplex, as long as you are authorized to do so.

To view the status of a measurement request, complete the following steps:

1. Select Option 3 (STATUS) from the STROBE OPTIONS menu. STROBE/ISPF displays the STROBE - STATUS panel, which shows the status (either active, suspended, queued, or completed) of each measurement request on a given system.

   The STROBE - STATUS panel displays the status of each measurement request on a given system, as well as specific information about the measurement request. The following table explains each status type.

Status Type	Definition
Active	STROBE is currently placing samples in the sample data set.
Suspended	STROBE is not sampling. If you have suspended the measurement session, the sample data set is open. If you have ended the measurement session, the sample data set is closed.
Queued	The target program has not yet begun executing.
Completed	The request has been terminated and all sample data sets have been closed.

2. Enter the system name in the SYSTEM field to view the status of measurement requests on a particular system. Alternately, clear this field using the space bar, and press **Enter** to view a list of all active systems in the sysplex. STROBE displays the STROBE - SYSTEM SELECTION LIST panel, which lists the active systems in the sysplex. Select one or more of these systems by entering “S” to the left of the system name.
3. Press the **F3** key to exit from the STROBE - STATUS panel.

Creating a Performance Profile That Relates CPU Activity with Source Statement Text

By default, STROBE CSP reports associate the CPU execution time observed during the measurement session with specific statement numbers. You can also view the statement text and associate the generated statements with the related statement numbers in the source code. The process of creating this association from observed activity to source code statement text is called *indexing*.

To create an indexed Performance Profile (one that includes statement text and, optionally, the related 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 statements in the source program. STROBE must obtain CSECT information for any load module you want to index. For detailed information on this process, see the *STROBE MVS User’s Guide*.

The following sections describe how to create a compiler listing data set, a map data set, and a Performance Profile.

Creating the Compiler Listing Data Set

To create a map data set, you must have a compiler listing data set. STROBE reads the SYSPRINT listing produced by a compiler or by the assembler and links procedure statement names to the address of each statement generated by the language processor. For STROBE to accurately index your program, specify the appropriate options when you compile your program.

Using the CSP-generated source code, the COBOL compiler produces the compiler listing data set, which is normally written to the DDNAME SYSPRINT. To prepare a compiler listing data set, you must do the following:

- Generate the CSP application with the **COMMLVL=3** option to create CSP statements and comments.
- Specify the SOURCE, LIST, and NOOFFSET compilation options to create the correct COBOL compiler listing.
- Modify the CSP-generated JCL or the CSP templates to save the listing to a data set.

The following sections describe how to modify the JCL and templates. Use either procedure to save the COBOL compiler listing to a data set.

Modifying the CSP-Generated JCL

To modify the JCL, use your normal compile procedure. In the procedure, override the SYSPRINT data definition by allocating SYSPRINT to a data set. For example,

```
//COB.SYSPRINT DD DSN=MY.CSP.LISTING,UNIT=SYSDA,
//             SPACE=(TRK,(50,10),RLSE),
//             DISP=(NEW,CATLG,DELETE),
//             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3059)
```

Modifying the CSP Templates

To modify the templates, add the SYSPRINT statements to the CSP templates in the ELASAMP library. Contact your system programmer if you cannot access your CSP templates.

To avoid uncataloged duplicate data sets, add a DELETE step to the beginning of your compile procedure. For example,

```
//DELETE EXEC PGM=IEFBR14
//DD1      DD DSN=&CGHLQ..&ENV..&MBR..SYSPRINT,DISP=(MOD,DELETE)
//             UNIT=WPAANY,SPACE=(TRK,(0))
```

To replace the SYSPRINT DD statement, follow this example:

```
//SYSPRINT DD DSN=&CGHLQ..&ENV..&MBR..SYSPRINT,
//             SPACE=(TRK,(200,10),RLSE),
//             UNIT=WPAANY,DISP=(NEW,CATLG,DELETE),
//             DISP=(MOD,PASS),
//             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=30
```

Creating the Map Data Set

With STROBE/ISPF, you can create a map data set in two ways: you can create it when you produce the Performance Profile or you can create it as a separate step.

Creating a Map Data Set When You Produce a Performance Profile

To create a map data set when you produce the Performance Profile, complete the following steps:

1. Select Option 4 (PROFILE) from the STROBE OPTIONS menu. The STROBE - PRODUCE A PERFORMANCE PROFILE panel appears.
2. In the SPECIFY PROFILE REPORT PARAMETERS section, enter “Y” to the right of the Indexing field. The STROBE - INCLUDE MAPS OF INDEXED SOURCE MODULES panel appears.
3. Enter “Y” in the NEW MAPS field. STROBE/ISPF displays the STROBE - INDEX TO CREATE A MAP DATA SET panel. Complete this panel as described in the following section.

Creating a Map Data Set as a Separate Step

To create a map data set as a separate step, select Option 5 (INDEX) from the STROBE OPTIONS menu. The STROBE - INDEX TO CREATE A MAP DATA SET panel appears.

On the STROBE - INDEX TO CREATE A MAP DATA SET panel, complete these steps to create a map data set:

1. Specify the type of index processing in the OPTION field by entering “B” for background processing or “F” for foreground processing.
2. Specify the program language by selecting one language in the PROGRAM LANGUAGE field. The panel displays only those program languages supported by STROBE at your site. Enter “Y” to the right of the appropriate language field.
3. Specify the name of the map data set in the OUTPUT: MAP DATA SET field.
   - Enter the name of a sequential data set or the name of a previously allocated partitioned data set and member name. The data sets must have DCB characteristics of RECFM=FB, LRECL=36. If the sequential data set does not exist, it will be dynamically allocated.
   - To specify a unit name and volume serial numbers, enter them in the UNIT and VOLUME fields. To use the installation defaults, clear the fields.
4. Specify the names of compiler listing data sets in the INPUT: COMPILER SYSPRINT DATA SETS field. Enter the names of data sets containing SYSPRINT listings. You can specify either sequential data set names or partitioned data set member names.
5. Press **Enter**.

Figure 2-1 on page 2-7 shows the STROBE - INDEX TO CREATE A MAP DATA SET panel for a CSP application.

Figure 2-1. STROBE - INDEX TO CREATE A MAP DATA SET Panel

```
 ------------------- STROBE - INDEX TO CREATE A MAP DATA SET -------------------
OPTION ===>
            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           ===>
 CSP             ===>   Y   DB2 DBRM        ===>       FORTRAN G       ===>
 FORTRAN VS OR H ===>       PL/I            ===>

OUTPUT: MAP DATA SET
        ===> 'wpa.mapfile'
  UNIT ===> WPAANY     VOLUME ===>

INPUT: COMPILER SYSPRINT DATA SETS
        ===> 'CSP.TSO.EPOTA42.SYSPRINT'
        ===>
        ===>
        ===>
        ===>
        ===>
```

Creating the Performance Profile

Once you have the information you need (the sample data set and, optionally, the map data sets) you can create a Performance Profile for your CSP application.

You have three options for creating a Performance Profile with STROBE/ISPF

- Create a Performance Profile automatically when submitting a measurement request for an active or queued request.

  Enter "Y" in the AUTO PROFILE CREATION field on the STROBE - ADD ACTIVE REQUEST panel or the STROBE - ADD QUEUED REQUEST panel.

- Create a Performance Profile from the STROBE OPTIONS menu.

  Select Option 4 (PROFILE) from the STROBE OPTIONS menu. STROBE displays the STROBE - PRODUCE A PERFORMANCE PROFILE panel, where you can specify options for the Performance Profile.

- Create a Performance Profile from the STROBE - STATUS panel.

  Enter "P" to the left of a completed measurement request number on the STROBE - STATUS panel. STROBE displays the STROBE - PRODUCE A PERFORMANCE PROFILE panel, where you can specify options for the Performance Profile.

Figure 2-2 shows the options on the STROBE - PRODUCE A PERFORMANCE PROFILE panel.

Figure 2-2. STROBE - PRODUCE A PERFORMANCE PROFILE Panel

```
-------------------- STROBE - PRODUCE A PERFORMANCE PROFILE -------------------
OPTION ===>  F

     B  - Background processing    F  - Foreground processing

ENTER BLANKS TO VIEW A DATASET SELECTION LIST
SAMPLE DATA SET NAME ===> WPANAC.WPAJEA.S001D001
                UNIT ===> WPAANY    VOLUME ===> WPA002

SPECIFY PROFILE REPORT PARAMETERS: (Y or N)
  Detail Reports   ===> Y    Tailor Reports   ===> Y    Indexing   ===> N

PROFILE REPORT FORMAT ===> N  ((W)ide or (N)arrow)

NUMBER OF COPIES FOR BACKGROUND REPORTS ===>

Specify a data set name to save a copy of the STROBE Profile Report:
DATA SET NAME ===> 'WPAGFL.IMSV5.PROF'

        UNIT ===> WPAANY    VOLUME ===> WPA001


Specify a SYSIN data set containing parameters for the Reporter:
DATA SET NAME ===>
```

Controlling the Level of Detail in the Performance Profile

By specifying certain STROBE options, you can either compress or expand the level of detail of the reports in the Performance Profile.

You can control the size of the reports by specifying certain parameters when you create the Performance Profile. For example, on a number of reports, you can specify a minimum percentage of activity that must occur for a procedure or cylinder to appear as an individual line on the report. If the minimum percentage is not attained, contiguous procedures or cylinders are condensed on a single line in the report. The lower the number you specify, the greater the level of detail in the reports.

The following reports have minimum percentage options:

- Program Usage by Procedure
- DASD Usage by Cylinder
- Transaction Usage by Control Section
- the Attribution reports

You can also break the reports into manageable sections by specifying the report resolution — the number of bytes to be considered a codeblock for detailed reporting. Once again, the lower the number you specify, the greater the level of detail in the reports. For more information, see *STROBE MVS Concepts and Facilities*.

Summary and Next Step

This chapter described the procedures necessary for measuring your CSP application and producing an indexed Performance Profile. The next chapter describes several of the pertinent Performance Profile reports and shows you how to analyze the reports to identify performance improvement opportunities.

Where to Find More Information

Refer to the following table for additional information on using STROBE and the CSP Feature.

For more information on...	**Refer to...**
Determining if the measurement session is valid	"Verifying the Measurement Session Information" on page 3-1
Deciding which Performance Profile reports to analyze	"Choosing Between Execution and Wait Reports" on page 3-2
Analyzing Performance Profile execution reports	"Finding Significant Activity in the Application" on page 3-3
Analyzing Performance Profile wait reports	"Additional Reports To Examine" on page 3-7
Learning more about STROBE	*STROBE MVS User's Guide*
Learning more about Performance Profiles	*STROBE MVS Concepts and Facilities*

Chapter 3.
Analyzing the Performance Profile for a CSP Application

This chapter describes the process for analyzing a Performance Profile for a CSP application. The following sections instruct you in

- verifying measurement session information
- choosing between execution and wait reports
- finding significant activity in the application
- using other Performance Profile reports to identify performance improvement opportunities

For more information on analyzing a Performance Profile, see the *STROBE MVS User's Guide*.

Overview

The example reports in this chapter refer to a Performance Profile produced after a measurement of a CSP-generated COBOL application running under a TSO session accessing a DB2 database. The reports featured in this chapter are part of an indexed Performance Profile; therefore, some reports are slightly different than those described in *STROBE MVS Concepts and Facilities*. To view reports not featured in this chapter, see *STROBE MVS Concepts and Facilities*.

Before You Begin

Before using this chapter, you should have produced a Performance Profile using one of the three methods described in Figure 3-2 on page 3-4. You will analyze either a printed or an online version of your Performance Profile, depending on the options you specified when creating the Profile. For information on browsing your Performance Profile, see "Browsing Performance Profiles" in the *STROBE MVS User's Guide*.

Verifying the Measurement Session Information

Start analyzing the Performance Profile by examining the Measurement Session Data report. This report describes the environment during a measurement session and provides some overall measurement statistics.

To verify that the Performance Profile is valid, examine 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 your application appears 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. 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.

The Measurement Session Data report in Figure 3-1 on page 3-3 displays a .69 RUN MARGIN OF ERROR PCT and a .96 CPU MARGIN OF ERROR PCT, indicating that the measurement session produced a valid Performance Profile.

Note: The CPU MARGIN OF ERROR PCT is typically higher than the RUN MARGIN OF ERROR PCT because it is based on only the number of CPU samples processed, whereas the run time margin of error is based on both CPU and WAIT samples.

Choosing Between Execution and Wait Reports

Usually, a performance improvement opportunity exists in either reducing wait time or curbing CPU consumption. 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 of the CPUs were active) and WAIT TIME PERCENT fields.

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 steps. If the value for CPS TIME PERCENT is high, you should examine the following execution reports to determine opportunities for improvement in CPS time:

- Program Section Usage Summary
- Program Use by Procedure
- Most Intensively Executed Procedures
- Transaction Summary
- Transaction Usage by Control Section
- Attribution of CPU Execution Time

If the value for WAIT TIME PERCENT is very high, 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
- Attribution of CPU Wait Time

Refer to *STROBE MVS Concepts and Facilities* for more information on any of these execution or wait reports.

Figure 3-1. Measurement Session Data Report

```
                                        ** MEASUREMENT SESSION DATA **

-------- JOB ENVIRONMENT --------      ------- MEASUREMENT PARAMETERS ------      ----- MEASUREMENT STATISTICS ----
 PROGRAM MEASURED   -    IKJEFT01       ESTIMATED SESSION TIME  -       2 MIN       CPS TIME PERCENT -          52.00
 JOB NAME           -      WPAJJN       TARGET SAMPLE SIZE      -      20,000       WAIT TIME PERCENT -          48.00
 JOB NUMBER         -    TSU32047       REQUEST NUMBER   (A)    -         468       RUN MARGIN OF ERROR PCT  -    .69
 STEP NAME   -  $WPAPCSP.$WPAPCSP       FINAL SESSION ACTION    -        QUIT       CPU MARGIN OF ERROR PCT  -    .96

 DATE OF SESSION    -    02/20/97       MODULE MAPPING BASELINE -           0       TOTAL SAMPLES TAKEN -      20,000
 TIME OF SESSION    -    10.38.05                                                  TOTAL SAMPLES PROCESSED -  20,000
                                                                                  INITIAL SAMPLING RATE- 166.67/SEC
                                        --------- REPORT PARAMETERS ---------      FINAL SAMPLING RATE  - 166.67/SEC
 SYSTEM -             ESA SP5.2.0
 DFSMS              -       1.2.0       REPORT RESOLUTION    -        2 BYTES      SESSION TIME -    3 MIN 18.67 SEC
 SUBSYSTEM   -         DB2 3.1.0       SORTSIZE             -        999,999      CPU TIME    -     1 MIN 14.81 SEC
 DB2 APPLICATION    -     EPOTA42       LINES/PAGE           -             60      WAIT TIME    -    1 MIN 05.84 SEC
 CPU MODEL          -   3090-600S                                                 STRETCH TIME -    0 MIN 59.02 SEC
 SMF/SYSTEM ID      -  SESA/MVSAQ       DASD= 2.0%  ATTR= 0.0%
                                                                                  SRB TIME  -       0 MIN  4.42 SEC
 REGION SIZE BELOW 16M  -  6,208K                                                 SERVICE UNITS -             10723
 REGION SIZE ABOVE  -      8,192K                                                 PAGES IN-      0     OUT-       0
                                                                                  PAGING RATE  -           0.00/SEC
 PTF LVL - 2.5.0/FS000000/FS000000                                                EXCPS -         91        0.46/SEC
 STROBE TAPE NUMBER -  000-S00DSK

 SAMPLE DATA SET  - WPAJJN.TSO.EPOTA42.WPAJJN.S001D001
```

The CPS TIME PERCENT of 52.00 in Figure 3-1 indicates that you should first examine the execution reports to determine what is using CPU time. Though the WAIT TIME PERCENT indicates that the wait reports are also worth investigating, this chapter examines only the execution reports. See the section "Additional Reports To Examine" on page 3-7 for further information regarding wait reports. The following section describes the specific reports you should analyze to identify opportunities for performance improvement in your CSP application.

Finding Significant Activity in the Application

To find significant activity in the application, first examine the Program Section Usage Summary and Most Intensively Executed Procedures reports.

The Program Section Usage Summary report shows the distribution of CPU time used by modules and control sections. After determining what programs or system services are consuming the most resources, examine the Program Usage by Procedures report for those modules to identify possible performance improvement opportunities.

The Most Intensively Executed Procedures report also targets users of CPU time, but lists only the 10 heaviest users. It shows whether service modules or user programs are responsible for the majority of the CPU usage. When CPU time is high, this report highlights likely candidates for analysis in the Program Usage by Procedures report.

This section examines the Program Section Usage Summary report, the Most Intensively Executed Procedures report, and the Program Usage by Procedures report.

Analyzing the Program Section Usage Summary Report

The Program Section Usage Summary report (Figure 3-2) displays the percentage of CPU time used by each program module and subsystem that was active during the measurement session. The module name .SYSTEM identifies the CPU time consumed by system service modules.

Figure 3-2. Program Section Usage Summary Report

```
STROBE* PERFORMANCE PROFILE                          CSP PROFILE                          12/21/94            PAGE   9
                                                ** PROGRAM SECTION USAGE SUMMARY **
 MODULE  SECTION 16M   SECT FUNCTION                CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:   .96%
   NAME  NAME    <,>   SIZE                         SOLO     TOTAL         .00      9.50     19.00     28.50     38.00

 .SYSTEM  .COBLIB            COBOL LIBRARY SUBROUTINE  13.79     13.79        .**************
 .SYSTEM  .CSPLIB            CSP LIBRARY SUBROUTINE     5.17      5.17        .*****
 .SYSTEM  .DB2               DB2 SYSTEM SERVICES       10.34     10.34        .**********
 .SYSTEM  .IRLM              RESOURCE LOCK MANAGER       .86       .86        .
 .SYSTEM  .PRIVATE           PRIVATE AREA               1.72      1.72        .*
 .SYSTEM  .SVC               SUPERVISOR CONTROL        26.72     26.72        .****************************
                                                       -----     -----
 .SYSTEM  TOTALS             SYSTEM SERVICES           58.60     58.60

 EP0TA42  EP0TA42  >   33608                          37.93     37.93        .****************************************
 EP00A42  EP00A42  >   31560                           1.72      1.72        .*

 EP42G    .COBLIB       1096 COBOL LIBRARY SUBROUTINE    .86       .86        .
 EP42G    EP42G        63510                             .86       .86        .
                                                       -----     -----
 EP42G    TOTALS   >   64606                           1.72      1.72

                                                       -----     -----
 PROGRAM IKJEFT01 TOTALS                              100.00    100.00
```

An initial evaluation of the Program Section Usage Summary report in Figure 3-2 indicates that you should examine the following two areas:

- System service routines .COBLIB and .CSPLIB. Though the .SVC and .DB2 modules also have high CPU time percentages, the library subroutines usually offer more opportunity for performance improvement. In this application, it may be possible to reduce or eliminate the number of calls made by the CSP and COBOL code to the library subroutines.
- Application program module EP0TA42, section EP0TA42. This module consumes 37.93 percent of the total CPU time, significantly more than any other module.

If the Program Section Usage Summary report does not clearly reveal the modules you should look at, or if you want to verify the best modules to investigate, examine the Most Intensively Executed Procedures report, as described in the following section.

Analyzing the Most Intensively Executed Procedures Report

The Most Intensively Executed Procedures report (Figure 3-3) lists, in descending order, the ten heaviest users of CPU time during the measurement session. For your CSP application, line items are totaled and listed as a single CSP process. If you are specifically looking for CSP processes, you can easily identify them by looking at the Line Number column. The key to the numbers listed in this column is as follows:

- an asterisk (*) preceded by only blank spaces indicates a CSP I/O process
- an asterisk (*) preceded by a statement number indicates a process with CSP application logic
- a statement number without an asterisk (*) indicates a non-CSP process

In Figure 3-3, the procedure using the most CPU time, at 29.31 percent, is the CSP process EP00P10 in module EP0TA42. The line number for this procedure, 94*, indicates that this process includes CSP application logic.

Figure 3-3. Most Intensively Executed Procedures Report

```
STROBE* PERFORMANCE PROFILE                          CSP PROFILE                          12/21/94            PAGE   8

                                        ** MOST INTENSIVELY EXECUTED PROCEDURES **
MODULE     SECTION     LINE      PROCEDURE/FUNCTION              STARTING   PROCEDURE   CPU TIME PERCENT   CUMULATIVE PERCENT
 NAME       NAME     NUMBER          NAME                        LOCATION    LENGTH      SOLO     TOTAL       SOLO      TOTAL

EPOTA42    EPOTA42      94* EP00P10                              003680       1676     29.31     29.31      29.31     29.31
.SVC       SVC 093              TSO/TGET-TPUT                                          20.69     20.69      50.00     50.00
.COBLIB    IGZCPAC        IGZCXDI  DOUBLE PRECISION DIVIDE                            13.79     13.79      63.79     63.79
EPOTA42    EPOTA42     124* DISPLAY_EMPLOYEE                    003D1C       2946      4.31      4.31      68.10     68.10
.CSPLIB    ELARPRTM       ELAASDSP DISPLAY SERVICES                                    4.31      4.31      72.41     72.41
.SVC       SVC 122              EXT.SVC ROUTER-TYPE 2                                  3.45      3.45      75.86     75.86
EPOTA42    EPOTA42     266* STMT_GRP                            004872       4876      1.72      1.72      77.58     77.58
EPOTA42    EPOTA42      82* CPU_LOOP                            003552        274      1.72      1.72      79.30     79.30
.SVC       SVC 047              TIMER MANAGER/SET TIMER                                1.72      1.72      81.02     81.02
.PRIVATE                        PRIVATE AREA                                           1.72      1.72      82.74     82.74
```

The next step in improving application performance is to identify the CSP statements in the process EP00P10 causing the high usage. To do so, examine the Program Usage by Procedure report.

Analyzing the Program Usage by Procedure Report

The Program Usage by Procedure report identifies the distribution of CSP usage within the generated COBOL program. The Line Number column differentiates CSP processes from COBOL procedures. The key to the numbers is as follows:

- a statement number followed by an asterisk (*) indicates a CSP process or a CSP processing statement
- a statement number without an asterisk (*) indicates a COBOL procedure or verb
- an asterisk (*) preceded by blank spaces indicates a CSP I/O process or a STROBE-generated label (for example, FIRST CSP STATEMENT)

  Note: Statement names and numbers are not available for I/O processes, so STROBE lists only the process name in the Line Number column.

The format of the Program Usage by Procedure report in Figure 3-4 on page 3-6, which is slightly different than the standard Program Usage by Procedure report described in *STROBE MVS Concepts and Facilities*, is the result of an indexed Performance Profile. The format in this report shows the structure of CSP statements included in a process and COBOL statements running on behalf of a CSP statement. In this example, the report features module EP0TA42 and unidentified the CSP processes and statements that caused high CPU activity. The Most Intensively Executed Procedures report (Figure 3-3) indicated that process EP00P10 was consuming the most CPU time for this module.

Identifying CPU Usage

To identify CPU usage, first examine the lines between the first and last CSP statement. The lines within this range indicate activity executed on behalf of specific CSP statements and processes, as well as COBOL verbs. Use the following key to distinguish between them:

- **CSP process:** In the PROCEDURE NAME column, the text for a CSP process is left justified. For example, line 94* identifies the process EP00P10.
- **CSP statement:** In the PROCEDURE NAME column, the text for a CSP statement is indented under a CSP process. For example, the statements that run on behalf of process EP00P10 are LOOPCNT = and WKITEM =.

  Note: The equal sign (=) following LOOPCNT and WKITEM identifies them as CSP assignment statements, which specify the movement of results of an

expression to a target data item. In this example, the results of arithmetic operations are moved to LOOPCNT and WKITEM.

- **COBOL verbs:** In the PROCEDURE NAME column, the text for a COBOL verb is indented under a CSP statement. For example, a COBOL verb running on behalf of CSP statement WKITEM = is the COMPUTE verb (line 9272).

To calculate the total CPU percentage that STROBE associated with a CSP statement, add together the CPU percentages of all COBOL verbs indented under the statement. Likewise, to calculate the total CPU percentage for a CSP process, add together the CPU percentages for all CSP statements indented under the process.

In the Program Usage by Procedure report in Figure 3-4, the CSP statement WKITEM = has the highest percentage of CPU activity. The sum of all COBOL verbs attributed to this statement equals 21.55%. The percentage of CPU time for LOOPCNT is 7.76%. Together, the amount of CPU time attributed to process EP00P10 is 29.31%, which indicates that you should examine this process to improve application performance.

Figure 3-4. Program Usage by Procedure Report (Application)

```
STROBE* PERFORMANCE PROFILE                          CSP PROFILE                          12/21/94            PAGE  12
                                                  ** PROGRAM USAGE BY PROCEDURE **
 MODULE - EP0TA42                                     SECTION - EP0TA42
                                                      SOURCE LANGUAGE - CSP GENERATED IBM COBOL II                    VC32X26
  LINE     PROCEDURE          STARTING      INTERVAL  CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  .96%
 NUMBER      NAME             LOCATION        LENGTH   SOLO     TOTAL        .00      2.00      4.00      6.00      8.00

        DATA DIVISION         000000           13736    .00       .00      - .
     *  FIRST CSP STATEMENT
    82*  CPU_LOOP
    91*   LOOPCNT =
  9211     COMPUTE            0035A8             116    .86       .86        .****
  9214     MOVE               00361C              50    .00       .00      - .
  9218     CALL               00364E              36    .86       .86        .****
  9220     GO                 003672              72    .00       .00      - .
    94*  EP00P10
   104*   LOOPCNT =
  9256     COMPUTE            0036BA              28    .86       .86        .****
  9262     COMPUTE            0036D6             116   6.90      6.90        .**********************************
  9265     MOVE               00374A              86    .00       .00      - .
   105*   WKITEM =
  9272     COMPUTE            0037A0              36    .86       .86        .****
  9278     COMPUTE            0037C4               8    .00       .00        .
  9284     COMPUTE            0037CC             298   6.90      6.90        .**********************************
  9287     MOVE               0038F6              50    .00       .00      - .
  9291     COMPUTE            003928             184   7.76      7.76        .*************************************
  9294     MOVE               0039E0              50    .00       .00      - .
  9298     COMPUTE            003A12             120   4.31      4.31        .**********************
  9301     MOVE               003A8A              50    .00       .00      - .
  9305     CALL               003ABC              36   1.72      1.72        .********
  9307     GO                 003AE0             614    .00       .00      - .
   124*  DISPLAY_EMPLOYEE
   128*   WHILE
  9426     IF                 003D46              16   1.72      1.72        .********
   129*   LOOPCNT =
  9433     COMPUTE            003D56              28    .00       .00        .
  9439     COMPUTE            003D72             116   1.72      1.72        .********
  9442     MOVE               003DE6              86    .00       .00      - .
  9448     GO                 003E3C               8    .86       .86        .****
  9452     PERFORM            003E44            2692    .00       .00      - .
   266*  STMT_GRP
   273*   LOOPCNT =
 10067     COMPUTE            0048C8             116   1.72      1.72        .********
 10070     MOVE               00493C            4760    .00       .00      - .
     *  LAST CSP STATEMENT
 10467  EZESQL-ERROR-ROUTINE  005BD4             948    .00       .00      - .
 10564     MOVE               005F88               4    .86       .86        .****
 10565     MOVE               005F8C            9144    .00       .00      - .
                                                       -----     -----
 SECTION  EP0TA42   TOTALS                             37.91     37.91
```

Identifying Additional CPU Activity

To further identify CPU activity, examine the lines preceding FIRST CSP STATEMENT and following LAST CSP STATEMENT, which STROBE has not associated with specific CSP processes or statements. The CSP statements using these COBOL lines may be responsible

for high CPU usage. Look for COBOL paragraph names within the COBOL listing to see if you can identify a logical relationship to CSP processes.

Note: To report all CSP and COBOL statements associated with CPU activity, when creating the Performance Profile, specify the Program Usage by Procedure report compress percentage default value of zero (0) and report resolution (RESLTN) value of two (2). Refer to "Summary and Next Step" on page 3-9 and Chapter 4 of the *STROBE MVS User's Guide* for more information.

When determining the cause of CPU activity in process EP00P10, the Program Usage by Procedure report indicates that the multiple COMPUTE statements in the CSP process WKITEM = warrant further investigation. See "Reports That Show System Services Invoked by the Application" on page 3-7 for this analysis.

Additional Reports To Examine

This section describes several types of reports that help you find performance improvement opportunities. The reports are grouped into the following categories:

- reports that show system services invoked by the application
- reports that show wait time

Examine these sections to get details about specific reports that are appropriate for your analysis.

Reports That Show System Services Invoked by the Application

When system services are responsible for significant CPU use or wait time, examine the Attribution reports to identify the sites of invocation of selected system services. STROBE produces Attribution reports for both CPU execution time and CPU wait time.

Each Attribution report detail line identifies a location in a program from which the service routine was directly or indirectly invoked. The location is usually within a user-written routine and is defined by transaction, module, control section, and return address.

The Attribution of CPU Execution Time report shows solo and total CPU time spent in the service routine invoked from that location. The Attribution of Wait Time report shows the page and program wait time spent in the service routine invoked from that location.

The Attribution of CPU Execution Time report (Figure 3-6 on page 3-8), when used in conjunction with the Program Usage by Procedure report (Figure 3-5 on page 3-8), identifies the CPU usage in the service sections .COBLIB and .CSPLIB. The Program Section Usage Summary report (Figure 3-2 on page 3-4) attributed 19% of CPU activity to these sections, indicating potential performance improvement opportunities.

The Program Usage by Procedure report for these service sections (Figure 3-5 on page 3-8) indicates that for .CSPLIB, two control sections are responsible for 5.17% of the CPU time. However, in .COBLIB, control section IGZCXDI, defined as Double Precision Divide, is responsible for 13.79% of the CPU time.

Figure 3-5. Program Usage by Procedure Report

```
STROBE* PERFORMANCE PROFILE                          CSP PROFILE                          12/21/94            PAGE  10
                                                ** PROGRAM USAGE BY PROCEDURE **
         .SYSTEM     SYSTEM SERVICES                        .COBLIB   COBOL LIBRARY SUBROUTINE
MODULE   SECTION        FUNCTION             INTERVAL  CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  .96%
  NAME      NAME                              LENGTH   SOLO     TOTAL        .00      3.50      7.00     10.50     14.00

IGZCPAC  IGZCXDI   DOUBLE PRECISION DIVIDE       968  13.79     13.79        .************************************
                                                       -----     -----
         .COBLIB   TOTALS                            13.79     13.79

         .SYSTEM     SYSTEM SERVICES                    .CSPLIB   CSP LIBRARY SUBROUTINE
MODULE   SECTION        FUNCTION             INTERVAL  CPU TIME PERCENT       CPU TIME HISTOGRAM  MARGIN OF ERROR:  .96%
  NAME      NAME                              LENGTH   SOLO     TOTAL        .00      1.50      3.00      4.50      6.00

ELARPRTM ELAASDSP  DISPLAY SERVICES           11928   4.31      4.31        .****************************
ELARPRTM ELARSMEM  MAIN MEMORY FUNCTIONS        1648    .86       .86        .*****
                                                       -----     -----
         .CSPLIB   TOTALS                            5.17      5.17
```

To determine which modules invoked service IGZCXDI, refer to the Attribution of CPU Execution Time report (Figure 3-6). This report identifies module EP0TA42 as responsible for invoking service IGZCXDI. As the Attribution report shows, CSP statement 105*, WKITEM =, is causing high CPU activity.

Figure 3-6. Attribution of CPU Execution Time Report

```
STROBE* PERFORMANCE PROFILE                          CSP PROFILE                          12/21/94            PAGE  16

                                        ** ATTRIBUTION OF CPU EXECUTION TIME **
.COBLIB  IGZCPAC  IGZCXDI   DOUBLE PRECISION DIVIDE
 -------------------------WAS INVOKED BY--------------------   -----------------VIA---------------------  CPU TIME %
 XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                SOLO  TOTAL

          EP0TA42  EP0TA42    0038B2   105*    WKITEM =                                                    13.79 13.79
                                                                                                           ----- -----
                                                                                                           13.79 13.79
.CSPLIB  ELARPRTM ELAASDSP  DISPLAY SERVICES
 -------------------------WAS INVOKED BY--------------------   -----------------VIA---------------------  CPU TIME %
 XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                SOLO  TOTAL

          EP0TA42  EP0TA42    0049FC     *    DISPLAY_MENU        IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
          EP00A42  EP00A42    004360     *    DISPLAY_MENU        IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
          EP00B42  EP00B42    001D06     *    SHOW_HEADER         IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
          EP00B42  EP00B42    001DAC     *    FLOAT               IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
          EP00B42  EP00B42    001E52     *    TRAILER             IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
                                                                                                           ----- -----
                                                                                                           4.31  4.31
.CSPLIB  ELARPRTM ELARSMEM  MAIN MEMORY FUNCTIONS
 -------------------------WAS INVOKED BY--------------------   -----------------VIA---------------------  CPU TIME %
 XACTION  MODULE   SECTION    RETURN   LINE   PROCEDURE NAME       MODULE   SECTION  FUNCTION                SOLO  TOTAL

          EP00B42  EP00B42    001DAC     *    FLOAT               IGZCPAC  IGZCLNK  LINKAGE MANAGER-DYNAMIC   .86   .86
                                                                                                           ----- -----
                                                                                                            .86   .86
```

Controlling the way CSP performs mathematical operations often improves application performance. In the Program Usage by Procedure report (Figure 3-5 on page 3-8), CSP statement WKITEM = shows multiple COMPUTE statements, indicating that CSP math was specified when generating module EP0TA42. The parameter MATH(CSP) caused the generation of these COMPUTE statements and caused the application to invoke the COBOL Double Precision Divide routine, service section IGZCXDI. By selecting the MATH(COBOL) parameter, the application will generate only one COMPUTE statement and will not need to invoke the Double Precision Divide routine. This change will decrease CPU usage and will improve the performance of the application.

Reports That Show Wait Time

For a Performance Profile that indicates a high percentage of CPU wait time, examine the following reports:

- Wait Time by Module report
- Attribution of CPU Wait Time report

The Wait Time by Module report shows all the modules, control sections, pseudo-modules, and pseudo-sections in which STROBE found the application waiting. STROBE records the run-time percent attributed to paging, as well as the total run-time percent.

The Attribution of CPU Wait Time also identifies page and program wait time, attributing the wait time of service routines to the locations that directly or indirectly invoked the service routine.

Summary and Next Step

This chapter provided a base of tactical knowledge that will allow you to analyze the Performance Profile for your CSP application. You can apply the steps described for the execution reports to the wait reports to identify performance problems in all areas of your application. The next step is to use this knowledge to measure your application. After you identify and fix problems in the coding, measure the application again to test the effect of your changes.

Additional Information

Consult the following table for more information on analyzing Performance Profile reports.

For more information on...	**Refer to...**
Measuring your CSP application	"Measuring the Application" on page 2-2
Analyzing execution and wait reports	*STROBE MVS Concepts and Facilities*
Learning more about STROBE	*STROBE MVS User's Guide*
Interpreting Performance Profiles	*STROBE MVS Concepts and Facilities*

Index

A

B

C

D

F

I

T

U

V

W