Understanding monitoring facilities in IBM TXSeries for Multiplatforms

TXSeries for Multiplatforms White Paper

Vijayeendra Namasevi (nvijayee@in.ibm.com)
Raghavendran Srinivasan (raghavs1@in.ibm.com)
Janaki Sundar (janakisundar@in.ibm.com)

Introduction

Monitoring can be defined as a process involving regular observation and recording of ongoing activities in a system to detect and warn about changes that occur in the system. Any business system needs to be monitored for better capacity planning and for tracking resource usage.

Monitoring a system is an important and challenging task for anyone responsible for administering a system. To effectively use, tune, and manage a system, it is important that the system administrator understands the services provided by the monitoring facilities available in the system. A good monitoring facility allows its users to customize their systems to suit their requirements with minimal impact on performance.

This paper describes four monitoring facilities available in IBM® TXSeries® for Multiplatforms. The paper begins with a brief introduction of the monitoring facilities available in TXSeries and the categories of resources that can be monitored using these facilities. This is followed by a detailed explanation of each of these facilities and how they can be used to monitor various resources. Wherever appropriate, tuning examples are provided to facilitate better understanding of these facilities. However, these examples are not meant for direct use.

This paper is intended for users and administrators of TXSeries for Multiplatforms who are interested in understanding the features and methodology provided by the product for monitoring and tuning systems. It is assumed that the reader has a reasonable knowledge of TXSeries for Multiplatforms. Complete information about TXSeries for Multiplatforms is available at:

http://www-306.ibm.com/software/htp/cics/txseries/library/

Conventions used: References to 'TXSeries' and 'CICS®' stand for 'TXSeries for Multiplatforms'.

Monitoring facilities available in TXSeries

TXSeries generates a lot of information about its resources, which can be used for monitoring. It provides four monitoring facilities for the purpose. These are:

- CICS Monitoring Facility

  CICS Monitoring Facility (CMF) is a comprehensive customizable facility provided by TXSeries. It gives a complete view of the time spent by a transaction in the system across its lifetime. CMF contains about 100 pre-defined fields, which collect data. Each CMF record contains information from all of these fields for each transaction. Because monitoring all these fields can result in the usage of a large amount of disk space, CMF allows you to select specific fields for monitoring.

- Statistics

  Statistics are provided by CICS for monitoring. They provide an overall picture of a CICS region, unlike CMF, which gives details about the individual transactions in a region.

  Statistics assist the user in monitoring memory growth and resources such as transactions, programs, and files. Terminals and queues can also be monitored with the help of statistics.

- TXSeries Administration Console

  The TXSeries Administration Console is a Web-based tool for performing various administration functions using a GUI. Monitoring through this console was introduced in TXSeries V6.2. The console allows you to select the parameters for monitoring for a specific duration. The results are displayed, as and when required, on a Web browser. It can also display the collected data in a graph for easier analysis.

- Automatic monitoring of heap memory

  TXSeries provides a utility to detect heap memory leaks in application servers. A separate thread runs in the application server to monitor the growth of the heap memory. This utility is mainly for debugging purposes.

Categories of resources that can be monitored by these facilities

While TXSeries generates a lot of information that can be monitored by using these facilities, this paper explains monitoring for the following categories of resources:

- Monitoring storage
- Monitoring CPU
- Monitoring CICS resources
- Monitoring 3270 terminals
- Monitoring intersystem communication (ISC)

All the above resources can be monitored online or offline:

Online monitoring

Refers to observing the system when it is in an active state. Statistics is an example of an online monitoring facility.

Offline monitoring

Refers to collecting data over a period of time for future analysis. CMF is an example of offline monitoring.

The following sections explain each of the monitoring facilities in detail.

CICS Monitoring Facility

TXSeries provides a detailed offline task monitoring facility through the CICS Monitoring Facility (CMF). CMF helps a user to monitor the system at the transaction level. In addition, it can be used to detect resource contention and other constraints. It can therefore help in the studying and tuning the performance of a system.

CMF terminology

Event Monitoring Points: CMF collects information at certain points referred to as *Event Monitoring Points* (EMPs). Event Monitoring Points can be either system-defined or user-defined.

System-defined Event Monitoring Points: During the execution of a transaction, the information related to its resource usage is collected internally by the CICS system. When a transaction terminates, the collected information is dumped into the Transient Data Queues (TDQ) specified in the Monitoring Definitions. CICS collects this standard data automatically when monitoring is enabled. Such event monitoring points are known as *system-defined Event Monitoring Points*.

System-defined EMPs include a wide range of fields, such as time taken on a Structured File Server (SFS) operation, the time taken by the terminal manager, the time spent on a CICS application, the total elapsed time for a transaction, the time spent on the scheduler, and so on. Even though you cannot modify these EMPs, you can choose the classes of data to be monitored.

User-defined Event Monitoring Points: TXSeries allows you to collect other information in addition to the information that can be obtained through system-defined EMPs. The CICS Monitoring Facility allows you to define the EMPs in an application program. You can place special trace commands in your application program to collect data. Such points are called *user-defined Event Monitoring Points*. You can decide the actions to be performed at these points. CMF appends these records to the system-defined records.

CMF fields: The user and system EMPs are configured using pre-defined numbers, commonly known as CMF field IDs or just fields. A field ID is a number that corresponds to a particular EMP. For example, Field ID 1 is used to indicate the 4-byte transaction identifier.

> **Note:** The monitoring records are written only when the transaction completes execution. It is not possible to retrieve the records when the transaction is hung.

User-defined EMPs

This section explains the use of user-defined EMPS. However, it does not explain the programming details of user-defined EMPs. Rather it points out the different scenarios where these EMPs can be useful. For details about programming, see the topic that explains performance monitoring of a user program in the TXSeries Information Center.

User-defined EMPs help you to time the interval between two events within an application or within CICS. The number of times that an event has occurred can also be tracked. In order to make use of these features, you must write an event monitoring program. You must inform the CICS region about this program through an attribute in MD.stanza, which is explained in a later section. CICS also supplies a sample program cics_emp.c, which can be tailored to suit your requirements.

Application of user-defined EMPs

You can use the user-defined EMPs in several ways:

- You can program the EMPs to clock individual SQL queries in your application.
- You can use the EMPs to monitor a remote transaction. Although different programs are executed in the back end, CPMI is the only transaction recorded by the CMF report. In such cases, you can program this module to record each program name. These records are appended to the existing records.
- You can clock the time consumed by individual programs (for example, programs invoked through External Call Interface (ECI)). To achieve this, you can define the DPL user exit in the PD stanza and in the DPL user exit program; start and end times of each program can be recorded separately. You can employ StartClock and StopClock functions provided with TXSeries.
- You can use EMPs to monitor specific transactions of interest to you. For example, in the initialization function of the user module, you can make your program return CICS_EMP_RETURN_DISABLE for the task that does not interest you.

Classification of fields as groups in CMF

Based on the data collected by them, the CMF fields are internally arranged into groups by CICS. These groups are:

- CICSCICS: Comprises the group of fields that collects details pertaining to CICS
- CICSCOMS: Comprises communication fields
- CICSFILE: Comprises fields that collect file-related information
- CICSPROG: Comprises fields that takes care of information corresponding to programs
- CICSTASK: Comprises fields that collect task-related information

You can specify these groups as an alternative to specifying fields in Monitoring Definitions. A few basic fields can be set up irrespective of the fields to be monitored.

Basic fields in CMF

The following are some of the basic information fields in CMF, which can be set in all cases:

- CICS_EMP_FID_TRANS_ID (Field ID - 001): Gives the 4-byte transaction identifier defined in TD.stanza. The transaction is the basic unit of monitoring through CMF. This is a non-tunable information field. All the CMF data is based on this ID.
- CICS_EMP_FID_START_TIME (Field ID- 005): Gives the system time at which the transaction started.
- CICS_EMP_FID_STOP_TIME (Field ID – 006): Gives the system time at which the transaction stopped.

Note: For the entire list of fields and groups available in CMF, see Appendix B.

Enabling CMF

To configure CMF, complete the following procedure:

1. Configure an extrapartition queue in Transient Data Definitions (TDD) stanza.

   The following example defines an extrapartition TDQ, assuming a region with the name CMFREG.

   ```
   cicsadd -c tdd -r CMFREG MONQ DestType=extrapartition
   ExtrapartitionFile="MONQ.out" RecordType=variable_length
   ```

   In this example, `MONQ` is the TDD entry name and the extrapartition file is created in the data directory of the region.

   > **Note:** There is no direct option to limit the size of the TDQ file of CMF. You can limit the size of the TDQ file of CMF programmatically, by first disabling the CMF using EXEC CICS SET MONITOR OFF and then resetting the file size of the CMF data file. You can take a backup of the CMF data file before performing the reset.

2. Update the Monitoring Definitions (MD) stanza by issuing the following command:

   ```
   cicsupdate -c md -r CMFREG TDQ="MONQ"
   ```

   where `MONQ` refers to the TDD entry created in Step 1.

   If you want to monitor specific fields, use the **Include** attribute or the **Exclude** attribute of the MD stanza. You can use either field IDs or group names for the **Include** and **Exclude** attributes.

   - To monitor specific fields by using field IDs for the **Include** and **Exclude** attributes, see the following example, which configures the region to monitor only storage-related fields by using the appropriate field IDs.

     ```
     cicsupdate -c md -r CMFREG MonitorStatus=yes
     TDQ="MONQ"Include="001,002,005,006,029,030,033,054,061,087,
     008,095,108,202,203"
     ```

   - To monitor specific fields by using group names for the **Include** and **Exclude** attributes, see the following example, which monitors only communication-related fields by using appropriate group names.

     ```
     cicsupdate -c md -r CMFREG MonitorStatus=yes TDQ="MONQ"
     Include="CICSCOMS"
     ```

   If you want to create separate monitoring records for each input and output operation for conversational transactions, set the value of the Conversational parameter in the MD stanza to **yes**.

   To update the path of the user-defined monitoring program, use the **UserMonitorModule** attribute in MD.

3. Cold start the region.

   ```
   cicscp -v start region CMFREG StartType=cold
   ```

After the region is started, the MONQ TDQ (`MONQ.out` file) is updated with the monitoring data. When the region is active, monitoring can be enabled or disabled dynamically by using CEMT:

- `CEMT SET MONITOR ON`: Enables monitoring at run time
- `CEMT SET MONITOR OFF`: Disables monitoring at run time

Any change made through CEMT updates only the runtime database and is therefore valid only for the current session of CICS. Other MD attributes cannot be defined through CEMT. However, if you want to monitor the startup programs also, then the **MonitorStatus** attribute must be set to **yes** in the Monitoring Definitions (MD.stanza) before starting the region.

Formatting CMF output

CICS supplies a basic formatter **cicsmfmt** (Monitoring data formatter), for formatting the data collected by CMF, in the TDQ file. The formatted output consists of only a few limited fields such as transaction name, terminal name, start and end times of a transaction, time spent waiting for file I/O, program name, total number of file requests issued, data segment memory occupancy, and first abend code. To overcome this limitation, use any of the following methods:

- Use the sample formatter program (**cicsmfmt**) provided with CICS to write your own formatter. The required fields can be obtained in the output by this method.
- Download the formatter supplied by a vendor from http://www-1.ibm.com/support/docview.wss?rs=175&context=SSAL2T&q1=monitoring&uid=swg24009305&loc=en_US&cs=utf-8&lang=en. This formatter supports many options and formats all of the monitoring fields available in CMF.

This paper uses the above mentioned vendor-acquired formatter for all the examples and references. A sample output is shown in Figure 1.

```
root@ajnabi3 : 6.1.0.0 /tmp/cmfreport
# ./cmfreport -f MONQ.out -s 001,002,003,004,005,006,007,024 | more
 TRAN TERM Opr T    Start-TOD     Stop-TOD Elapsed-Time    TranSched
 CAIN          A 14:55:33.588 14:55:40.126        6.537         .717
 CRAB      AIX A 14:55:40.126 14:55:40.154         .028         .000
 CGWY          A 14:55:40.155 14:55:40.155         .000         .000
 CTDP          A 14:55:40.165 14:55:40.165         .000         .000
 CAGE          A 14:55:40.166 14:55:40.166         .000         .000
 CCIN      AIX S 14:56:19.928 14:56:19.931         .002         .000
 CCIN      AIX S 14:56:19.933 14:56:19.934         .001         .000
 CTIN      AIX S 14:58:56.011 14:58:56.013         .001         .000
 CTIN      AIX S 14:58:56.013 14:58:56.014         .001         .000
 CTIN      AIX S 14:58:56.013 14:58:56.015         .001         .000
 CTIN      AIX S 14:58:56.015 14:58:56.016         .001         .000
 CTIN      AIX S 14:58:56.016 14:58:56.017         .001         .000
 CTIN      AIX S 14:58:56.018 14:58:56.019         .001         .000
 CTIN      AIX S 14:58:56.020 14:58:56.021         .001         .000
 CTIN      AIX S 14:58:56.021 14:58:56.022         .001         .000
 CTIN      AIX S 14:58:56.023 14:58:56.024         .001         .000
 CTIN      AIX S 14:58:56.024 14:58:56.025         .001         .000
 CTIN      AIX S 14:58:56.026 14:58:56.028         .001         .000
 CTIN      AIX S 14:58:56.028 14:58:56.030         .001         .000
 CTIN      AIX S 14:58:56.031 14:58:56.032         .001         .000
 CTIN      AIX S 14:58:56.034 14:58:56.035         .001         .000
 CTIN      AIX S 14:58:56.038 14:58:56.040         .001         .000
 CTIN      AIX S 14:58:56.040 14:58:56.041         .001         .001
 CTIN      AIX S 14:58:56.042 14:58:56.043         .001         .003
 CTIN      AIX S 14:58:56.043 14:58:56.044         .001         .000
 CTIN      AIX S 14:58:56.043 14:58:56.045         .001         .000
 CTIN      AIX S 14:58:56.045 14:58:56.046         .001         .000
 CTIN      AIX S 14:58:56.048 14:58:56.050         .001         .000
 CTIN      AIX S 14:58:56.050 14:58:56.051         .001         .003
 CTIN      AIX S 14:58:56.051 14:58:56.052         .001         .004
 CTIN      AIX S 14:58:56.063 14:58:56.065         .001         .000
 CTIN      AIX S 14:58:56.065 14:58:56.066         .001         .018
 CTIN      AIX S 14:58:56.066 14:58:56.068         .001         .000
 CTIN      AIX S 14:58:56.068 14:58:56.070         .001         .000
 CTIN      AIX S 14:58:56.070 14:58:56.071         .001         .001
 CTIN      AIX S 14:58:56.071 14:58:56.073         .001         .003
 CTIN      AIX S 14:58:56.073 14:58:56.074         .001         .004
 CTIN      AIX S 14:58:56.074 14:58:56.075         .001         .006
 CTIN      AIX S 14:58:56.076 14:58:56.077         .001         .007
 JC01 U6T4 AIX S 14:58:56.037 14:58:56.078         .041         .000
 CTIN      AIX S 14:58:56.077 14:58:56.107         .029         .009
 CTIN      AIX S 14:58:56.106 14:58:56.119         .013         .000
 CTIN      AIX S 14:58:56.120 14:58:56.121         .001         .051
 CTIN      AIX S 14:58:56.121 14:58:56.122         .001         .052
 CTIN      AIX S 14:58:56.122 14:58:56.124         .001         .054
 CTIN      AIX S 14:58:56.124 14:58:56.125         .001         .055
 CTIN      AIX S 14:58:56.125 14:58:56.127         .001         .057
 KA10 X6T4 AIX S 14:58:56.107 14:58:56.260         .152         .000
 KA60 J6T4 AIX S 14:58:56.261 14:58:56.273         .012         .000
 F401 L6T4 AIX S 14:58:56.127 14:58:56.286         .159         .000
 KA50 T9T4 AIX S 14:58:56.273 14:58:56.288         .014         .000
 KA70 J6T4 AIX S 14:58:56.287 14:58:56.329         .041         .000
```

*Figure 1: A sample output from **cmfreport** command*

The output obtained from the formatter can be filtered based on your requirements, using commands such as **grep**. For example, the following command filters the fields for CEMT:

```
#. /cmfreport -f MONQ.out -s 001,002,003,004,005,006,007 | grep CEMT
```

This gives the following output:

```
[cmfreport] Selected: EMPs 7), Records 12 of 12), File Errors 0),
ABENDs (0) TaskFlags (0)
CEMT UDOT SUN T 11:28:14.256 11:28:23.510 9.254
```

Monitoring storage

TXSeries provides fields to monitor the memory usage for a given transaction. This section describes the fields associated with memory usage and the inferences that can be drawn from this data. Examples based on a few tests are also provided.

Inappropriate usage of memory in application programs or lack of system resources can result in performance degradation. Table 1 shows a few CMF fields related to storage (CICSSTOR group). The CMF variables listed in the table will help you to spot the memory constraints from a transaction perspective.

Table 1: CMF fields related to storage (CICSSTOR group)

Field name (Field ID)	**Description**
CICS_EMP_FID_PAGE_COUNT (029)	The number of page faults serviced that did not require any physical I/O activity.
CICS_EMP_FID_PAGE_IO_COUNT (030)	The number of page faults serviced that required physical I/O activity.
CICS_EMP_FID_DSEGMENT_SIZE (033)	The maximum amount of memory that a transaction uses in a data segment.
CICS_EMP_FID_GETMAIN_COUNT (054)	The total number of GETMAIN requests that a task issued.
CICS_EMP_FID_SWAP_COUNT (061)	The number of times that the task was swapped out of main memory.
CICS_EMP_FID_TSEGMENT_SIZE (087)	The maximum amount of memory used by the transaction in text segment.
CICS_EMP_FID_FREEMAIN_SIZE (088)	The total amount of memory that is freed using EXEC CICS FREEMAIN requests.
CICS_EMP_FID_DSEG_OCCUPANCY (095)	The data segment occupancy of the user transaction. This field is updated before GETMAIN and FREEMAIN requests and at transaction exit.
CICS_EMP_FID_TSEG_OCCUPANCY (108)	The text segment occupancy of the user task. This field is updated before a LINK, XCTL, or LOAD request and at task exit.
CICS_EMP_FID_FREEMAIN_CNT (202)	The total number of FREEMAIN requests that the transaction issued.
CICS_EMP_FID_GETMAIN_SIZE (203)	The total amount of memory that was obtained from GETMAIN requests.

A large value for PAGE_COUNT, PAGE_IO_COUNT, and SWAP_COUNT fields (Field IDs: 029, 030, 061) indicates a shortage of primary memory or a memory leak in the system or application, leading to performance degradation.

The following tips provide guidance about the usage of GETMAIN and FREEMAIN in a CICS application for proper utilization of memory.

- Memory pool leaks can be easily detected by observing the FREEMAIN and GETMAIN counts. The FREEMAIN and GETMAIN counts are updated with every EXEC CICS FREEMAIN and GETMAIN call.
- Use FREEMAIN calls explicitly to free the memory allocated by GETMAIN SHARED, to avoid the leaks in task-shared pool.
- Large values reported for the fields related to PAGE COUNTS can mean incorrect virtual memory settings in the OS, resulting in more page faults.
- DSEG_OCCUPANCY (Field ID: 095) signifies the amount of time that the allocated memory is part of the task in the physical memory. The utilization of main memory is inversely proportional to the value in the DSEG_OCCUPANCY field.
- Because FREEMAIN calls are time-consuming, memory allocated through GETMAIN can be reused instead of repeated GETMAIN and FREEMAIN calls.
- Consider enabling TransDump in TD stanza, if GETMAIN and FREEMAIN are used frequently in the applications. CICS will automatically initiate transaction dump on any storage violation.

The DSEGMENT_SIZE and TSEGMENT_SIZE fields (Field IDs: 033, 087) are related to the program control interface of CICS and provide information about program caching in memory. A large data segment for a program can result in performance degradation, which can be monitored with the mentioned fields. Typically, initialized and static local variables largely contribute to the size of the data segment.

Note:

1. The data segment memory of a program can be found on AIX using the **size –fv** command**.**
2. The value of monitoring through CMF can be fully appreciated after the bottlenecks are identified using statistics. For example, if you observe a growth in the task-shared pool using statistics, enabling CMF will help you to spot the task causing the memory leak.

Monitoring CPU

Table 2 lists the fields related to CPU usage provided by CMF and provides a description of each of these fields. These fields are considered for performance monitoring.

Table 2: Fields related to CPU usage provided by CMF

Field name (Field ID)	Description
CICS_EMP_FID_TASK_ET (007)	Time spent by a task during execution. This does not include the time spent in getting the task from the 3270 terminals to the scheduler, queuing in the scheduler, or getting the task from the scheduler to the application process.
CICS_EMP_FID_TASK_UT(008)	The CPU time during which the task was in user space when in execution.
CICS_EMP_FID_TASK_ST (211)	The CPU time during which the task was in kernel space while it was executing.
CICS_EMP_FID_SUT_CICSSPACE (216)	The system or user time of the CPU that is spent in the CICS space for the task. CICS space means the processing of EXEC CICS statements. System time is the time spent in CPU kernel space for the task. User time is that spent in the CPU user space for the task.
CICS_EMP_FID_ULM_TIME(217)	The elapsed time spent in monitoring ULM.

If the TASK_ET (Field ID: 007) field reports an unexpectedly high value, other fields described later, need to be investigated. The TASK_UT (ID: 008) field gives an idea about the CPU usage by individual transactions. You can tune the Transaction Definitions (TD) attribute **MaxTaskCPU** to prioritize the transactions as required. The **MaxTaskCPU** and **MaxTaskCPUAction** attributes in the RD stanza are meant to control the CPU time consumed by any transaction. For more details about **MaxTaskCPU** and **MaxTaskCPUAction** attributes, see the TXSeries Information Center.

Note: Setting MaxTaskCPU in the TD stanza overrides the value of MaxTaskCPU in the RD stanza.

Monitoring CICS resources

Table 3 lists the field names related to monitoring CICS resources and provides a description of each of these fields.

Table 3: Fields related to monitoring of CICS resources

Field name (Field ID)	Description
CICS_EMP_FID_ET_TS_IO (011)	Time spent waiting for Temporary Storage I/O.
CICS_EMP_FID_ET_TRANS_SCH (024)	The total elapsed time spent by the transaction in the scheduler.
CICS_EMP_FID_ET_SUSPENDED (027)	Elapsed time during which the task was voluntarily suspended.
CICS_EMP_FID_ET_FILE_IO (063)	Elapsed time spent waiting for file I/O.
CICS_EMP_FID_TASK_FLAGS**(**064**)**	A field that is used to hold information for signaling unusual conditions detected during the execution of a task.
CICS_EMP_FID_ET_TD_IO (101)	Elapsed time that the task spent waiting for Transient Data I/O.
CICS_EMP_FID_ET_EXCEPTION (103)	Elapsed time that the task waited for TS space or memory.
CICS_EMP_FID_WT_TRANS_SCH (221)	Elapsed waiting time in the transaction scheduler for an application server to become available.
CICS_EMP_FID_TT_TRANS_SCH (222)	Elapsed waiting time in the transaction scheduler for a TCLASS (TranClass) to become available.

In the table, the fields 011, 063, and 101 are I/O related.

Note the following:

- When the elapsed time spent waiting for TS I/O (Field ID: 011) is high, you need to tune the file manager. It is recommended that you use the main Temporary Storage Queue (TSQ) instead of the auxiliary TSQ, if your application logic permits. Using the main TSQ can considerably help reduce the TS_IO (11) time.
- Defining TSQs as non-recoverable reduces time and additional overheads involved in TSQ operations.
- Incorrect positioning of EXEC CICS ENQ and DEQ in a program, while working simultaneously on a queue, can also result in a higher TS I/O time.
- A high value for File I/O contention (Field ID: 063) calls for tuning the file manager or application. Tuning OpThreadPoolSize and BufferPoolSize in the SSD stanza can ease the read and write access to SFS files.
- The CICS_BROWSE_CACHE environment variable helps in faster file browsing. For details of this variable, see the TXSeries Information Center.
- Faster access is achieved by splitting the data across the files. Also, read-only files can be defined non-recoverable to reduce the File I/O time.
- A higher value for TD_IO (101) indicates an OS file system configuration issue or a user program limitation.

All the transactions are queued before they are executed by the CICS application server (cicsas). The field 024 gives the time spent by the transaction in the scheduler. This time is a sum total of the values of the CICS_EMP_FID_WT_TRANS_SCH (Field ID 221) and CICS_EMP_FID_TT_TRANS_SCH (Field ID 222) fields.

Note the following:

- A higher value reported by CMF for the 024 field with the major contribution from CICS_EMP_FID_WT_TRANS_SCH shows that the RD MaxServer might be insufficient and CICS is not finding enough cicsas to run the transaction soon after it is submitted. Increasing the MaxServer and MinServer values in the RD stanza can help reduce this schedule time.
- A higher value reported by CMF for the 024 field with the major contribution by CICS_EMP_FID_TT_TRANS_SCH signifies that the task has been waiting in TCLASS queue before getting scheduled in the cicsas queue. Tuning the TCLASS value in ClassMaxTasks of the RD stanza can improve the performance of this queue.

Monitoring terminal usage of the system

Table 4 lists the field names related to monitoring terminal usage of the system and provides a description of each of these fields.

Field 009 in table gives the amount of time spent by the transaction waiting on an RPC Terminal I/O request. This variable is significant if the RPC client, RPC-EPI client, or Telnet 3270 client is used to connect to the region. This value will always be **0** for the CTG client. This field is applicable for transactions of type CICS_EMP_FID_TRANS_TYPE=T.

Making the task pseudo-conversational for terminal-based I/O can largely help to reduce the time reported in field 009.

*Table 4: Fields related to terminal usage of the syst***em**

Field name (Field ID)	**Description**
CICS_EMP_FID_TRANS_TYPE (004)	Transaction type, which can be one of the following: • A, attached by Automatic Transaction Initiation (ATI) • C, second or subsequent part of a conversational task • D, attached by transient data trigger level • T, attached from a terminal • Z, second or subsequent part of a pseudo-conversational task
CICS_EMP_FID_ET_TERM_IO (009)	Total time that the task spent waiting for terminal I/O.

Field name (Field ID)	Description
CICS_EMP_FID_ET_TERM_MGR (015)	Total time spent by the task in the terminal manager. This elapsed time for terminal I/O processing in the application process includes the time spent waiting for a user response at a terminal. This is not the same as field 009, because the region cannot distinguish between times spent processing and waiting in the cicsterm process.

Monitoring intersystem communication (ISC)

Table 5 lists the field names for monitoring intersystem communication and provides a description of each of these fields.

In the table, field 208 refers to the amount of time spent waiting for the TCP/IP link. Increasing TCPProcessCount in the LD stanza can reduce the time spent waiting for the TCP/IP link. That is, the field gives the amount of time that the system took to create and allocate a conversation to the remote region. This time is also affected by the availability of resources in the remote region.

Table 5: Fields related to monitoring intersystem communication

Field name (Field ID)	Description
CICS_EMP_FID_ET_PPC_SNA_LINK (207)	Elapsed time that the task spent waiting on an SNA link.
CICS_EMP_FID_ET_TCP_LINK (208)	Time spent by the task waiting on a TCP/IP link.
CICS_EMP_FID_ET_LOCAL_SNA_LINK (218)	Time spent waiting on a local SNA link.

Performance impact of CMF

In general, a 10-15% overhead on performance is observed if all the fields of CMF are enabled for monitoring. However, enabling only the required fields for monitoring can substantially reduce the overhead. You can thus enable only the required monitoring fields. On completion of each transaction, monitoring records for the list of included fields are written to the disk. The frequency of writes to the disk would increase if there are too many short-running transactions. You can also consider creating a RAM disk or other alternatives for improved I/O performance.

CICS statistics

Statistics is an exhaustive and commonly used monitoring tool in TXSeries. Statistics deal with a CICS region on the whole, unlike CMF, which gives micro-level details pertaining to each transaction. This section explains some of the important information available through online statistics and a few tuning recommendations based on it.

Classification of statistics

Statistics can be online or offline, as shown in Table 6.

Table 6: Types of statistics

Online statistics	Offline statistics
CSTD: A CICS-supplied transaction, which provides details about the current state of a region.	• Interval statistics • End-of-day statistics • Requested statistics • Unsolicited statistics

The different categories of offline statistics are briefly explained in Table 7.

Table 7: Categories of offline statistics

Type of statistics	Description
Interval statistics	Collected at specified time intervals. CICS writes the collected data over the specified interval and resets the statistics values to **0**. Interval statistics are useful for analyzing activities during a particular period of time. For example, this can be very useful if you want to study the usage of resources and load during peak and off-peak hours. This type of statistics is user configurable.
End-of-day statistics	Collected at end of day or shutdown of a CICS system. End-of-day statistics are useful for studying trends and trouble spots. These statistics are dumped by CICS by default, irrespective of the user's settings.
Requested statistics	Collected to track the state of the system immediately when requested. These statistics are useful for analyzing temporary problems. You can control requested statistics.
Unsolicited statistics	Collected automatically for dynamic allocation and de-allocating the resources. These records are written about resources that are about to be deleted and with statistics that will otherwise be lost. This type of statistics is controlled by CICS and you have no control these statistics.

Note:

1. For more details about the categories of offline statistics, see the topic explaining collecting monitoring statistics, in the TXSeries Information Center.
2. For an explanation on the fields used in offline statistics, see the topic on statistics in the TXSeries Information Center.

Enabling statistics

By default, statistics are enabled for a region. The CSTD transaction, supplied with CICS, presents a snapshot of the current state of a region and its load. You can choose the resources to monitor from the options provided by CSTD. Figure 2 gives a snapshot of CSTD with options.

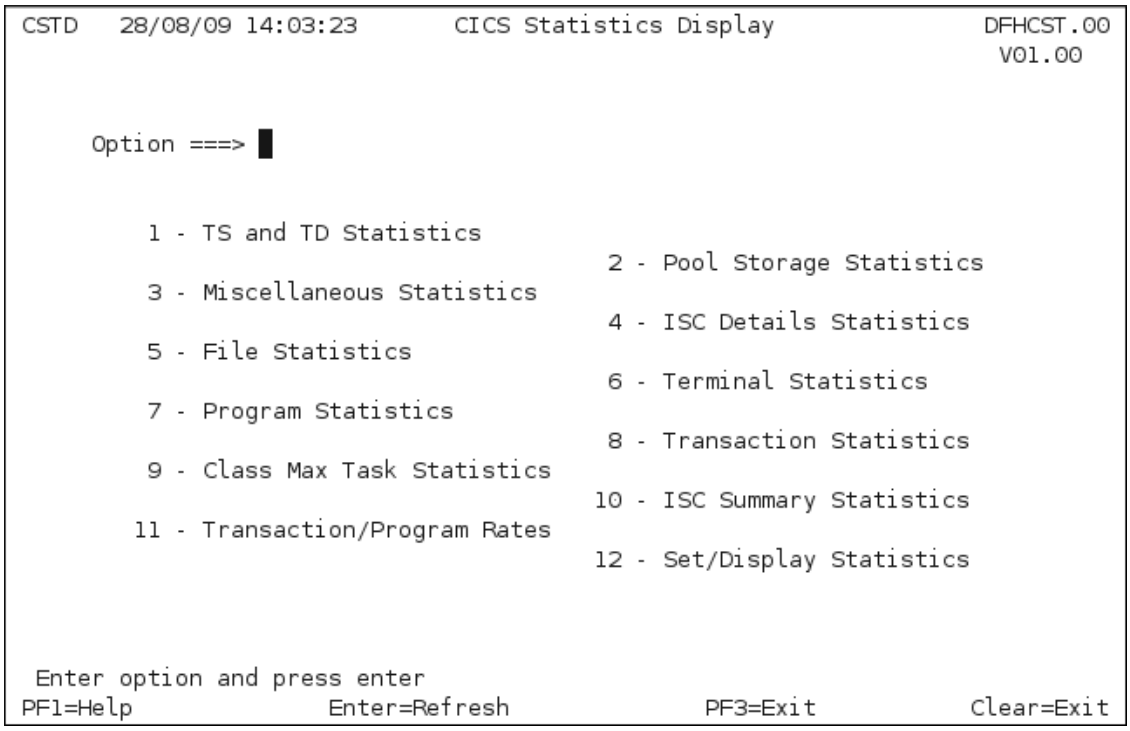

Figure 2: A sample CSTD screen showing a list of available options to monitor

Formatting statistics output

The output from statistics can be formatted using the sample formatter provided by CICS, **cicssfmt**. This formatter can be customized to meet your requirements. It formats all statistics records to the specified output file.

The following command is used to list interval statistics for intersystem communication management from 6 a.m. on May 1, 2008 until 6 p.m. on May 31, 2008 from the **statsfile** file.

```
cicssfmt –c ISCM –s 080501060000 –e 080531180000 –i statsfile
```

For a detailed description of the formatter, see the topic explaining **cicssfmt** in the TXSeries Information Center.

Monitoring storage

Statistics can be used to monitor the region pool and the task-shared pool. Online storage information can be accessed using CSTD option 2 or CST2 transaction directly.

Note the following:

- Region pool memory is extensively used for CICS internal purposes. For example, the number of active CICS application server (cicsas) processes in the

region directly impacts the usage of the region pool. Sufficient region pool memory needs to be allocated to handle the maximum number of cicsas specified in the RD stanza. This memory is also used to cache the runtime databases of a region. Hence, the usage of region pool can increase substantially with an increase in the number of resources defined for a region.

- Task-shared pool is used when the application has an EXEC CICS GETMAIN call with the SHARED option or it shares data through the COMMAREA. If the task-shared pool is observed to be continuously increasing over a period of time (which can be either observed through offline statistics or the Administration Console with monitoring enabled), you can find the leaking tasks by enabling the storage fields in CMF. Main memory temporary storage queues are also stored in task-shared pool.

Monitoring CICS resources

Statistics can be used to monitor CICS resources, which include files, Transient Data Queues (TDQ), Temporary Storage Queues (TSQ), programs, journals, and transactions.

You can monitor queue-related resources by using option 1 of CSTD. CST1 can be used as an alternative. This option provides details about the queue operations in the region since the last interval of collected statistics. Because this data summarizes the usage of queues in the region, it can provide very useful input for tuning.

The sample CSTD screen in Figure 3, which displays queue-related data, provides some guidance on this.

```
CST1   28/08/09 14:41:19       CICS Statistics Display                DFHCST.01

12:00:00 :Last Reset
                              Temporary Storage
                         Reads       Writes      Deletes
                  Main:  0000017090  0000042694  0000001783
                  Aux:   0000011946  0000037535  0000001760
            Aborts:        0000000000          I/O Errors:          0000000000
            Commits:       0000001764          Auth Errors:         0000000000
            Exhausted:     0000000000          # Aged:              0000000000
            Peak Num:      0000000000          # CAGE:              0000000000
            Current Num:   -000316961          Del Conflict:        0000000000
            Longest Queue: 0000000020          Remote Reqs:         0000000000

                           Transient Data
                            Reads       Writes
                     Extra:  0000000000  0000001511
                     Intra:  0000000000  0000000000
                     Remote: 0000000000  0000000000
                     Trigger: 0000000000  0000000000


Enter=Refresh PF3=Main Menu                      PF6=Start Autoupdate Clear=Exi
```

Figure 3: Sample CSTD screen displaying queue-related data

The screen provides information on both the main and auxiliary TSQs. As mentioned earlier, it is recommended that you use main TSQs rather than auxiliary TSQs, wherever the business logic permits.

Note the following:

- If main memory queues are largely used, you need to consider increasing MaxTaskSharedPool in the RD stanza. The 'Peak Num' field can help in tuning MaxTaskSharedPool.
- **Aborts** gives the number of times the data in recoverable TSQ has been rolled back. If this value is more than expected, you must analyze the region console and symrecs.
- If the memory is frequently exhausted, the task-shared pool size can be increased if the main TSQ is used. If SFS is used as the file manager, SFS data and log volume sizes can be increased. You can set TSQAgeLimit in the RD stanza to inform CICS to clean up unused TSQs.
- **Remote** gives the number of requests made for remote queues.

Option 5 in CSTD addresses file related statistical information. Figure 4 shows a sample of data displaying reads/writes in a file.

```
CST5   28/08/09 14:42:50       CICS Statistics Display              DFHCST.05

12:00:00 :Last Reset
                           File Statistics  (Item 0001 Total:   0025 )
Name                 Writes                Deletes                Opens
         Reads                 Browses               Updates              Closes
F100BASE 0000000000 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F100PTH1 0000000000 0000000000 0000000000 0000000274 0000000000 0000000 0000000
F100PTH2 0000000000 0000000000 0000000000 0000000284 0000000000 0000000 0000000
F20BASE  0000000000 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F20PTH1  0000000958 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F20PTH2  0000000197 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F21BASE  0000000000 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F21PTH1  0000000917 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F21PTH2  0000000203 0000000000 0000000000 0000000000 0000000000 0000000 0000000
F22BASE  0000000000 0000000000 0000000000 0000000000 0000000000 0000000 0000000
                                      Totals:
0000025  0000003781 0000000171 0000014233 0000002865 0000000020 0000000 0000000

                Display inactive files: Y
                                        Start display with file:

Enter=Refresh     PF3=Main Menu                      PF8=Forward     Clear=Exit
```

Figure 4: Sample of data displaying reads/writes in a file

Note the following:

- File related statistics in CSTD deal with the file operations for each file in the file system of the region. Higher values in these fields can affect performance considerably because of disk access.
- Caching the most accessed files can improve performance. Enable the CICS_BROWSE_CACHE environment variable to achieve faster browsing.

- Using the correct file type (KSDS, RRDS, or ESDS) can help to improve performance. If a lot of writes are involved with SFS as file manager, tuning the PrePages value in the FD stanza can adjust the allocation done on the disk for the file.
- Read-only files can be made non-recoverable using the **Recoverable** attribute of the FD stanza for better performance. You can switch on ErrorIsolation in the FD stanza while tuning. When you switch on ErrorIsolation, all the SFS errors are passed on to the application. Setting the parameter to **off** can improve the performance.
- A simple way to enhance the performance in an SFS system is to maintain the user files in a server which is different from the region FM and tune the value of BufferPoolSize and OpThreadPoolSize in the SSD stanza.

Options 7 and 8 of CSTD direct the user to Program and Transaction Statistics respectively:

- Program statistics tell you the number of runs of each program in the region. Frequently accessed programs (except MF COBOL and Java™ programs) can be made resident by setting the Resident flag in the PD stanza to **yes**, for enabling the program to be cached in the memory. The cacheable program size is limited by the ProgramCacheSize value in the RD stanza.
- Transaction statistics also display the number of runs for each transaction. Additionally, the number of abends related to task-private storage is also available, which might suggest that you look at the task-private pool allocated for the user application.

Monitoring intersystem communication (ISC)

Figure 5 shows a sample of data displaying ISC details. As will be seen from the figure, CSTD option 4 provides detailed information about the ISC load in a region. All the ISC facilities supported by TXSeries are dealt with. For example, the maximum number of requests, the number of requests transmitted, and the number of requests purged for outbound function shipping.

```
CST4   28/08/09 14:44:21       CICS Statistics Display                DFHCST.04

12:00:00 :Last Reset
                        Intersystems Communications Stats for: @OWW
                                                    Page 0001 Total:   0002

                                    In          Out         Forwarded
               File Control:        0000000000  0000000000  0000000000
               Transient Data:      0000000000  0000000000  0000000000
               Temporary Storage:   0000000000  0000000000  0000000000
               Interval Control:    0000000000  0000000000  0000000000
               DPL:                 0000000000  0000000000  0000000000
               Transaction Route:   0000000000  0000000000  0000000000
               Terminal Defs:       0000000000  0000000000
               DTP Conversations:               0000000000
                              Queued Function Ships
                          Max Queued:    0000000000
                          Curr Queued:   0000000000
                          Sent:          0000000000
                          Purged:        0000000000
                          Send attempts: 0000000000


Enter=Refresh     PF3=Main Menu                      PF8=Forward     Clear=Exit
```

Figure 5: Sample of data displaying ISC details

Note the following:

- If in CSTD, **Curr Queued** reports a continuously high value, the TCPProcessCount value in the LD stanza can be increased for CICS_TCP communication.
- If the protocol is ppc_tcp, tune the RPCListenerCount value in the RD stanza to increase the RPC processing count.

However, remote resource contention cannot be handled by tuning the parameters mentioned above. Enabling CMF can assist you in spotting the bottlenecks.

Monitoring through TXSeries Administration Console

The TXSeries Administration Console is a Web-based utility that helps you to manage a CICS system. Monitoring through the TXSeries Administration Console was introduced in TXSeries V6.2 for monitoring a CICS region through the browser. The console also presents a graphical representation of the collected data.

Terminology

Monitoring session. The Administration Console defines a *monitoring session* as a duration for which a region is being monitored. This session is divided into equal

intervals called *sampling intervals* specified in minutes with a minimum value of 1 minute.

Monitoring profile. A monitoring profile must be in place for a region that requires monitoring. A monitoring profile is a specification of resources and the associated attributes to be monitored. It must be ready when a monitoring session is started.

> **Note:** There can be only one active monitoring session, and hence, only one region can be monitored at a time through the Administration Console.

Enabling monitoring through TXSeries Administration Console

To enable monitoring for a region through the Administration Console, you must define the following attributes in Monitoring Definitions (the MD stanza):

- **Set administration console monitoring on?** Set this attribute to **yes**
- **TCP Port number for administration console monitoring** Set this attribute to a unique port number in the range 1025–65535

You must define the monitoring profile for the region by selecting the required resources and attributes, and also the sampling interval.

When the Administration Console monitoring is set to **yes**, a CICS-defined transaction CMBT starts with the region and continues to run in a dedicated application server until the region stops, or the transaction is explicitly forcepurged. The CMBT transaction collects the data configured in the profile and supplies the data to a Web listener.

GUI-based monitoring

The Administration Console presents several options for monitoring:

- System statistics such as region pool, task-shared pool, and task-private pool.
- Transaction and program statistics such as the number of transactions started, the number of programs run, the number of exceptions raised and similar information for selected transactions and programs.
- Various attributes of the transaction and the program.

At any point of time, the monitoring data can be viewed by using the **current view** option. With this option, you can track the current state of the region. You can stop the monitoring session at any time.

By enabling monitoring through the Web Administration Console, the data received from the monitoring region is automatically collected and saved at the intervals you specify for offline analysis. You can later view this data in either the data format or the graphical format.

One of the main advantages of monitoring using the Administration Console is that you can view the statistics data collected at regular intervals in the form of a graph. This pictorial view aids your understanding of the system.

> **Note:** For complete information on monitoring through the Administration Console, see the TXSeries Information Center.

Figure 6 shows a sample graphical output from the Administration Console showing the number of transactions started.

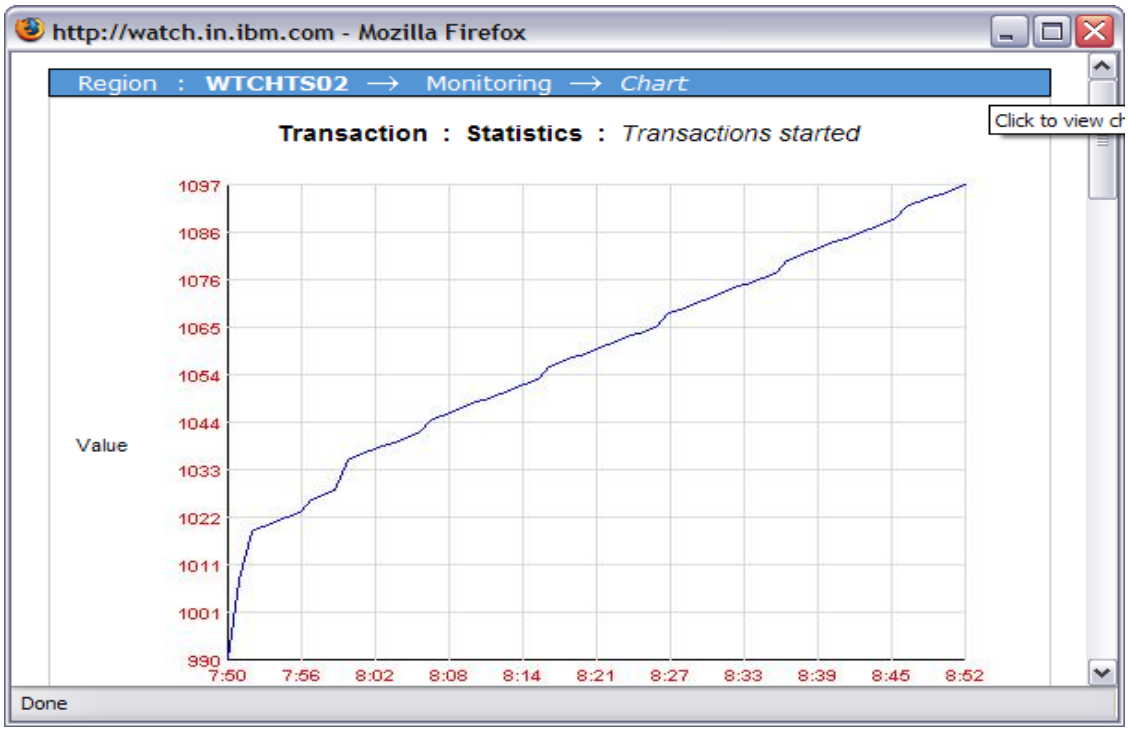

Figure 6: A sample graphical output showing the number of transactions started

Figure 7 shows the output from the Administration Console showing task-shared pool storage.

Figure 7: Output showing task-shared pool storage

Automatic monitoring of heap memory

TXSeries provides an autonomic feature to monitor the growth of heap memory in the cicsas process. The automatic monitoring of heap memory feature was introduced in TXSeries V5.1 and has been available since then. This monitoring facility can be a useful aid in warning about possible memory leaks in the cicsas process.

The main features of this facility are:

- When enabled, this feature constantly monitors the data segment in the cicsas process at regular intervals. This process of monitoring memory does not affect the normal run time, because it is done through a separate thread.
- This monitoring feature can be configured using the ServerMemCheckInterval and ServerMemCheckLimit variables in the RD stanza. ServerMemCheckInterval defines the time in seconds in which the cicsas process can record its data segment size. ServerMemCheckLimit defines the number of times that the check must be done before CICS evaluates and warns about the increase in memory. It is suggested that you make these checks as a minimum requirement when a cicsas process reports growth in heap memory. You can also check if there are any missing FREEMAIN calls in the application. In other words, scenarios in the application where memory is allocated but not released need to be found.

- Because cicsas is a hosting environment, all runtime libraries, such as compiler runtime and database client libraries, for example, reside in the cicsas process. Thus, when a cicsas process reports growth in memory, it might be because of the application, or because of the vendor-supplied software loaded into the cicsas process or CICS code. You can try to isolate the cause by reducing variables such as a database or a certain application. The heap memory can then be monitored for growth.
- The CICS_LEAKDEBUG feature can be enabled and reports can be generated, which can be useful in analyzing memory leaks.

INQUIRE APIs

INQUIRE APIs allow user-written programs to retrieve information about a single named resource. You can use these APIs to browse all the runtime definitions that the programs are authorized to access for a particular resource. The browsing command has a format with start and end operations enclosing the browse operation. The start operation indicates the start of the browse operation. The end operation ends the browse and frees any held resources. The following API is used to get information about a file that has a specified index name:

```
EXEC CICS INQUIRE FILE("FIL1") INDEXNAME(index);
```

INQUIRE can be used to obtain information in a wide range of categories. A few examples are:

- **INQUIRE PROGRAM** Gathers information about a program, map set, or a table
- **INQUIRE STATISTICS** Retrieves information about the accumulation and recording of CICS.
- **INQUIRE SYSTEM** Provides details about the local CICS region
- **INQUIRE TRANSACTION** Helps to understand the transaction details

CEMT INQUIRE is equivalent to the INQUIRE APIs that can be used in a program. It also provides the same set of options as the APIs to get information about resources.

For example, to determine if a transaction is in the in-doubt condition, CEMT INQUIRE TASK INDOUBT can be used. Figure 8 shows the output from CEMT INQUIRE TASK.

```
CST8   28/08/09 14:46:07       CICS Statistics Display              DFHCST.08

12:00:00 :Last Reset
                          Transaction Statistics  (Item 0001 Total:   0179 )
      # Run                       # Run                       # Run
 Name            S Abends   Name            S Abends   Name            S Abends
 CADB 0000000000 0000000    CAGE 0000000000 0000000    CAIN 0000000000 0000000
 CALF 0000000000 0000000    CARP 0000000000 0000000    CATS 0000000000 0000000
 CAUT 0000000000 0000000    CCIN 0000000001 0000000    CDCN 0000000000 0000000
 CEBR 0000000000 0000000    CECI 0000000000 0000000    CECS 0000000000 0000000
 CEDF 0000000000 0000000    CEMT 0000000001 0000000    CESF 0000000003 0000000
 CESN 0000000000 0000000    CFTS 0000000017 0000000    CGWY 0000000000 0000000
 CHAT 0000000004 0000000    CICE 0000000017 0000000    CIOD 0000000000 0000000
 CIOF 0000000000 0000000    CJDB 0000000000 0000000    CLAM 0000000017 0000000
 CLU0 0000000000 0000000    CMBT 0000000000 0000000    CMLV 0000000000 0000000
 CPMI 0000000000 0000000    CRAB 0000000000 0000000    CROW 0000000000 0000000
 Total Transaction:           0000000179
            # Run:           0000040280         St Abends:         0000000000

            Display inactive transactions: Y
                                  Start display with transaction:

Enter=Refresh     PF3=Main Menu                     PF8=Forward     Clear=Exit
```

Figure 8: Output from CEMT INQUIRE TASK

In case of transaction routing, a transaction can have a different name on the remote system. As a result the output of CEMT INQUIRE TRANSACTION on the local system might be different from that obtained on the remote system.

INQUIRE TERMINAL can be used to find out about an Advanced Peer-to-Peer Communication (APPC) session in SNA communication. The full connection details can be obtained using the INQUIRE CONNECTION command

> **Note:** For a complete reference on the INQUIRE API, see the section on CICS API command reference in the TXSeries Information Center.

Conclusion

This paper is an attempt to provide a handy guide to understanding the various monitoring facilities available in TXSeries. The applicability of the different monitoring facilities available in TXSeries differs depending on the circumstances. The specific monitoring facility applicable to your system, therefore, depends on the circumstances facing you. For instance, if you want information about the overall usage of a region in your system, you can use the statistics facility or the TXSeries Administration Console. If you want to view data in a graphical format, you can use the TXSeries Administration Console, but at the cost of a dedicated application server. If you need specific details for tuning your system at the transaction level, you can use CMF, which provides the necessary data that you might require. Hence, the use of a specific facility is entirely dependent on the system that is being tuned. It is therefore important that you have a clear understanding of all the facilities before deciding on which facility best suits your system.

Appendix A. Precautions to observe before tuning your system

Take the following precautions before tuning your system:

- Back up your configuration files before starting the tuning process. This helps you to recover the original configuration, in case of any problem caused by the changes made.
- Make one tuning change at a time and measure its effect.
- Identify and prioritize the major constraints in the system and then proceed with the tuning accordingly.
- Tuning is a continuous process, because the constraints vary with time.
- Tune to relieve only identified constraints. Tuning resources, which are not the primary cause of performance problems, has little or no effect on response time. Also, a system that is tuned beyond requirements can deliver the best performance. But such a system might require more maintenance than a mildly tuned or standard system.

Appendix B. Monitoring field descriptions

The following table lists the monitoring fields, identifies the group to which they belong, and provides a description of each field.

Field ID	Group	Description
001	CICSTASK	The name by which the system knows the transaction.
002	CICSTERM	The name by which the system knows the terminal.
003	CICSCICS	The name by which the system knows the operator.
004	CICSTASK	The transaction type, which can be one of the following: • A (Attached by automatic transaction initiation (ATI) • C (Second or subsequent part of a conversational task) • D (Attached by transient data trigger level) • T (Attached from a terminal) • Z (Second or subsequent part of a pseudo conversational task)
005	CICSCICS	The recording start time. This is the time when the task was attached or when the data recording was reset after a conversational or User Load Module (ULM) record was written.
006	CICSCICS	The recording stop time. This is the time when the task was detached or data recording was complete in support of a conversational or ULM record written.
007	CICSTASK	The elapsed time for which the task was running in the system. It is measured between monitoring points, and contains the time between the application server starting and stopping work on the task. This time does not include the time that is spent in getting from the CICS 3270 Terminal Emulator to the scheduler, queuing in the scheduler, or getting from the scheduler to the application process. This field contains times that are also accounted for in other fields, such as fields 027, 063, and 101.
008	CICSTASK	The CPU time during which the task was in user space while

Field ID	Group	Description
		it was running.
009	CICSTERM	The total time that the task spent waiting for terminal I/O.
010	CICSJOUR	The elapsed time that the task spent waiting for journal I/O.
011	CICSTevent monitoring point	The elapsed time that the task spent waiting for TS I/O.
015	CICSTERM	The total time that the task spent in the terminal manager. It is the elapsed time that is taken by terminal I/O processing in the application process. It includes time that is spent waiting for a user to respond at a terminal. This is not the same as field 009, because the region cannot distinguish between the time that is spent processing and the time that is spent waiting in the cicsterm process.
016	CICSTERM	The system time that the task spent in the terminal manager. System time is the time that is spent processing terminal requests in the CICS region. It does not include time that is spent processing in the cicsterm process.
024	CICSTASK	The total elapsed time that the transaction spent in the scheduler.
027	CICSTASK	The elapsed time during which the task was voluntarily suspended.
029	CICSSTOR	The number of page faults that occurred during the lifetime of the task that did not require I/O activity.
030	CICSSTOR	The number of page faults that occurred during the lifetime of the task that did require I/O activity.
031	CICSTASK	The TCA sequence number for the task.
033	CICSSTOR	The maximum amount of data memory that a task uses. This field is updated as a result of an EXEC CICS GETMAIN request.
034	CICSTERM	The number of messages that the task receives from the primary terminal facility.
035	CICSTERM	The number of messages that the task sends to the primary terminal facility.
036	CICSFILE	The number of file GETs that a task issued.
037	CICSFILE	The number of file PUTs that a task issued.
038	CICSFILE	The number of file BROWSEs that a task issued.
039	CICSFILE	The number of file ADDs that a task issued.
040	CICSFILE	The number of file DELETES that a task issued.
041	CICSDEST	The number of transient data GETs that a task issued.
042	CICSDEST	The number of transient data PUTs that a task issued.
043	CICSDEST	The number of transient data PURGEs that a task issued.
044	CICSTevent monitoring point	The number of temporary storage GETs that a task issued.
046	CICSTevent monitoring point	The number of auxiliary TS PUTs that a task issued.
047	CICSTevent monitoring point	The number of main TS PUTs that a task issued.

Field ID	Group	Description
050	CICSMAPP	The number of BMS map requests that a task issued.
051	CICSMAPP	The number of BMS IN requests that a task issued.
052	CICSMAPP	The number of BMS OUT requests that a task issued.
054	CICSSTOR	The total number of GETMAIN requests that a task issued.
055	CICSPROG	The number of LINK requests that a task issued.
056	CICSPROG	The number of XCTL requests that a task issued.
057	CICSPROG	The number of LOAD requests that a task issued.
058	CICSJOUR	The total number of journal output requests that a task issued.
059	CICSTASK	The number of START or INITIATE requests that a task issued.
060	CICSSYNC	The number of SYNCPOINT requests that a task issued.
061	CICSSTOR	The number of times that the task was swapped out of main memory.
063	CICSFILE	The elapsed time that was spent waiting for file I/O.
064	CICSTASK	A field that is used to hold information for signaling unusual conditions that are detected during the execution of a task. These bit flags are defined as follows: Bit 1: Out-of-phase clock start/stop detected: C (Conversational); D (ULM write request) Bit 5: Corrupt data storage area detected Bit 22: Maximum task condition detected Bit 23: Storage shortage condition detected
067	CICSTERM	The number of messages that the task received from the alternate terminal facility.
068	CICSTERM	The number of messages that the task sent to the alternate terminal facility.
071	CICSPROG	The name of the first program that was invoked at attach time.
083	CICSTERM	The number of characters that the task received from the primary terminal facility.
084	CICSTERM	The number of characters that the task sent to the primary terminal facility.
085	CICSTERM	The number of characters that the task received from the alternate terminal facility.
086	CICSTERM	The number of characters that the task sent to the alternate terminal facility.
087	CICSSTOR	The maximum amount of text memory that is used. This field is updated as a result of a LINK or XCTL API command.
088	CICSSTOR	The total amount of memory that is obtained from EXEC CICS FREEMAIN requests.
089	CICSCICS	The name by which the system knows the user.
090	CICSMAPP	The total number of BMS requests that the task issued.
091	CICSDEST	The total number of TD requests that a task issued.
092	CICSTevent monitoring point	The total number of TS requests that a task issued.

Field ID	Group	Description
093	CICSFILE	The total number of file requests that a task issued.
094	CICSSTOR	The total time that the task spent in program compression.
095	CICSSTOR	The data segment occupancy of the user task. This field is updated before GETMAIN and FREEMAIN requests and at task exit.
096	CICSTASK	Total elapsed time that is taken to execute the EXEC CICS statements in the task.
097	CICSCOMS	The fully qualified name by which the originating system or local terminal is known at attach time. This name is generated by the autoinstall user exit for a terminal that is attached to a CICS region. For terminals that are attached through Systems Network Architecture (SNA) links, this is the LUNAME of the remote system. So, for a transaction-routed terminal, where CICS is the application-owning region, this is the LUNAME of the terminal-owning region.
098	CICSTASK	Unique representation of the ID for the LUW for the current task.
101	CICSDEST	The elapsed time that the task spent waiting for TD I/O.
103	CICSCICS	The elapsed time that the task waited for TS space or memory.
108	CICSSTOR	The text segment occupancy of the user task. This field is updated before a LINK, XCTL, or LOAD request and at task exit.
109	CICSTASK	The priority of the transaction. The priority of a transaction determines which transactions get first use of resources when they become available, and how quickly the transaction is executed.
112	CICSCICS	One of the following conditions that is causing the monitoring record to be written: • C (Conversational) • D (ULM write request) • T (Task termination)
113	CICSPROG	The first abend code that the task recorded.
114	CICSPROG	The most recent abend code that the task recorded that is different from the first abend code that the task recorded.
115	CICSPROG	The elapsed time that was spent waiting for the program to be loaded.
200	CICSFILE	The elapsed time that was spent in the file manager (SFS).
202	CICSSTOR	The total number of FREEMAIN requests that the task issued.
203	CICSSTOR	The total amount of memory that was obtained from GETMAIN requests.
207	CICSCOMS	The elapsed time that the task spent waiting on an SNA link.
208	CICSCOMS	The elapsed time that the task spent waiting on a TCP/IP link.
209	CICSCOMS	The number of ISC messages that the task received.

Field ID	Group	Description
210	CICSCOMS	The number of ISC messages that the task sent.
211	CICSTASK	The CPU time during which the task was in kernel space while it was executing.
212	CICSTASK	The number of times that a context switch occurred because this process voluntarily gave up system resources before its time period was completely used.
213	CICSTASK	The number of times that a context switch occurred because a higher priority process thread was able to run or because the current process exceeded its allotted time.
214	CICSTASK	The number of signals that the task received.
215	CICSTASK	The number of file system I/O actions that occurred while the task was active. This number accounts only for real I/O; data that is supplied by the caching mechanism is charged only to the first process to read or write the data.
216	CICSTASK	The system or user time of the CPU that is spent in the CICS space for the task. CICS space means the processing of EXEC CICS statements. User time is the time that is spent in the user space of the CPU for the task. System time is the time that is spent in kernel space of the CPU for the task.
217	CICSUSER	The elapsed time that was spent in monitoring ULM.
218	CICSCOMS	The elapsed time that the task spent waiting on a local SNA link.
219	CICSTASK	The operating system Process ID (PID) of the process that is running the task.
220	CICSTASK	The elapsed time that the task was delayed because of EXEC CICS DELAY.
221	CICSTASK	The elapsed waiting time in the transaction scheduler for an application server to become available.
222	CICSTASK	The elapsed waiting time in the transaction scheduler for a TCLASS (TranClass) to become available.

Appendix C. Integration with Tivoli

The IBM Tivoli® Monitoring solution can be integrated with TXSeries. Tivoli scripts can be written to monitor a TXSeries region, SFS availability, and SFS page space utilization. A sample implementation is available for downloading.

The sample Tivoli solution for TXSeries is available at:

http://www-01.ibm.com/software/brandcatalog/portal/opal/details?catalog.label=1TW10TM5C

The sample SCRIPT data provider offers information such as indication of dump files, CICS region availability, SFS availability, SFS page space utilization, region transaction statistics, and region transaction performance data.

Users can alternatively develop their scripts and customize them to suit their requirements.

GC34-7115-00