

CICS Transaction Server for z/
OSバージョン 5 リリース 6

パフォーマンス・ガイド



注記

本書および本書で紹介する製品をご使用になる前に、[製品の特記事項](#)に記載されている情報をお読みください。

本書は、IBM® CICS® Transaction Server for z/OS®, バージョン 5 リリース 6 (製品番号 5655-Y305655-BTA)、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典：

CICS Transaction Server for z/OS
Version 5 Release 5
Performance Guide

発行：

日本アイ・ビー・エム株式会社

担当：

トランスレーション・サービス・センター

© Copyright International Business Machines Corporation 1974, 2020.

目次

本書について.....	vii
第 1 章測定、調整、およびモニター: 基本.....	1
24 ビット、31 ビット、および 64 ビット・アドレッシング.....	1
CICS トランザクション・フロー.....	2
パフォーマンス・データをモニターおよび収集するための CICS の機能.....	3
パフォーマンス・データをモニターおよび収集するための CICS ツール.....	3
パフォーマンス・データを取得するためのその他のツール.....	4
リソース測定機能 (RMF).....	4
Tivoli OMEGAMON XE for CICS on z/OS.....	5
IBM Z Decision Support.....	6
パフォーマンスのモニターおよび検討.....	6
モニターの活動および手法の確立.....	7
モニター・スケジュールの計画.....	8
パフォーマンス検討の典型的質問.....	10
CICS パフォーマンス分析の手法.....	12
パフォーマンス測定ツール.....	23
システムの調整.....	24
パフォーマンス・データを取得するための CICS ツール.....	25
パフォーマンス・データを取得するためのその他のツール.....	34
CICS のパフォーマンス制約の識別.....	47
ハードウェアの競合.....	48
設計上の考慮事項.....	49
応答時間の観察.....	50
ストレージ・ストレスの削減.....	52
DASD ページング・アクティビティーの削減.....	54
リソース競合の削減.....	55
リソース問題の解決.....	56
記憶保護違反の削減.....	58
パフォーマンス管理およびキャパシティー・プランニング.....	58
CICS トランザクションとハードウェア・リソースの関連付け.....	59
第 2 章 CICS システムのパフォーマンスの改善.....	61
CICS トランザクション・マネージャー: パフォーマンスおよび調整.....	63
最大タスク仕様 (MXT) の設定.....	63
トランザクション・クラス (MAXACTIVE) を使用してトランザクションを制御する方法.....	65
トランザクション・クラス・ページしきい値 (PURGETHRESH) の指定.....	65
タスクの優先順位付け.....	66
CICS ディスパッチャー: パフォーマンスおよび調整.....	67
オープン TCB 管理.....	68
インターバル制御値パラメーター: ICV、ICVR、および ICVTSD.....	70
MROBTCH.....	71
FORCEQR.....	71
PRTYAGE.....	72
SUBTSKS.....	72
TCB 統計.....	73
仮想記憶と実記憶: パフォーマンスおよび調整.....	74
CICS 仮想記憶.....	75
オンライン・システムの分割: 仮想記憶域.....	132
リンク・バック域 (LPA/ELPA) でのモジュールの使用.....	133

位置合わせマップまたは位置合わせなしマップの選択.....	133
常駐、非常駐、または一時としてのプログラムの定義.....	134
16 MB より上へのアプリケーション・プログラムの配置.....	135
トランザクション分離を使用する場合の実記憶域の割り振り.....	135
SNA パーシングを使用したサブプール 229 の拡張の制限.....	136
CICS のストレージ保護機能: パフォーマンスおよび調整.....	137
Language Environment での調整.....	137
GETMAIN および FREEMAIN アクティビティの最小化.....	138
Language Environment の、AMODE(24) プログラム用のランタイム・オプション.....	139
C++ の DLL の使用.....	139
Language Environment が一時データ・キュー CESE にダンプ出力を書き込むのに消費する時間の最小化.....	139
Java アプリケーション: パフォーマンスおよび調整.....	140
MVS および DASD: パフォーマンスおよび調整.....	140
ネットワーキングおよび z/OS Communications Server: パフォーマンスおよび調整.....	141
端末入出力域のサイズの設定.....	142
任意受信入力域サイズの設定.....	143
任意受信プールのサイズの設定.....	144
SNA での MVS 高性能オプションの使用.....	146
SNA トランザクション・フローにおける伝送数の調整.....	146
SNA チューニングを使用したラージ・メッセージのセグメント化.....	147
同時ログオンおよびログオフ要求数の制限.....	148
端末スキャン遅延の調整.....	149
出力端末データ・ストリームの圧縮.....	151
端末の自動インストールの調整.....	152
CICS MRO、ISC、および IPIC: パフォーマンスおよび調整.....	154
システム間セッションのためのキュー管理.....	157
トランザクション・クラス DFHTCLSX および DFHTCLQ2 を使用したストレージ使用の制御.....	159
MRO セッションの端末入出力域の長さ (SESSIONS IOAREALEN) の制御.....	159
要求のバッチ処理 (MROBTCH).....	160
ミラー・トランザクションの存続期間の延長 (MROLRM および MROFSE).....	161
シップされた端末定義の削除の制御 (DSHIPINT および DSHIPIDL).....	161
CICS VSAM およびファイル制御: パフォーマンスおよび調整.....	163
VSAM のチューニング: 一般的な目標.....	163
VSAM サブタスキングの使用.....	174
データ・テーブルの使用.....	175
カップリング・ファシリティ・データ・テーブルの使用.....	177
VSAM レコード・レベル共用の使用.....	186
スレッド・セーフなファイル制御アプリケーション.....	189
ファイル制御 API のコスト.....	190
パフォーマンスのためのデータベース管理.....	192
DBCTL パラメーターの設定.....	192
CICS Db2 接続機能の調整.....	192
パフォーマンスおよびメンテナンスのための許可 ID の選択.....	193
ロギング.....	194
同期点.....	195
CICS ロギングおよびジャーナリング: パフォーマンスおよび調整.....	195
CICS ログ・マネージャー.....	196
ログ・ストリーム・ストレージ.....	196
ジャーナル・レコード.....	199
ロガー環境のモニター.....	200
カップリング・ファシリティへのデータの書き込み: パフォーマンスの考慮.....	201
ログ・ストリーム数の定義: パフォーマンスの考慮.....	202
LOWOFFLOAD および HIGHOFFLOAD パラメーター.....	203
ステージング・データ・セットのサイズの調整.....	205
活動キーポイント頻度 (AKPFREQ).....	206
ログ延期インターバル (LGDFINT).....	207
DASD 専用ロギング.....	208

CICS 一時記憶域: パフォーマンスおよび調整.....	209
CICS 一時記憶域: 概説.....	209
一時記憶域キューの自動削除.....	211
主一時記憶域: モニターおよび調整.....	212
補助一時記憶域: モニターおよび調整.....	213
リカバリー可能 TS キューとリカバリー不能 TS キュー.....	214
CICS 一時データ (TD) 機能: パフォーマンスおよび調整.....	214
リカバリー・オプション.....	215
区画内一時データの考慮事項.....	217
区画外一時データの考慮事項.....	218
グローバル CICS エンキューおよびデキュー: パフォーマンスおよび調整.....	219
CICS モニター機能: パフォーマンスおよび調整.....	221
CICS トレース: パフォーマンスおよびチューニング.....	221
CICS セキュリティー: パフォーマンスおよび調整.....	223
VERIFY TOKEN および SIGNON TOKEN のための調整.....	223
CICS の起動時間およびシャットダウン時間: パフォーマンスおよび調整.....	224
始動プロシージャの改善.....	224
自動インストールのパフォーマンス.....	226
MVS 自動リスタート管理.....	226
CICS Business Transaction Services: パフォーマンスおよび調整.....	227
z/OS ワークロード・マネージャーによるチューニング.....	228
z/OS ワークロード管理で使用される用語.....	229
z/OS ワークロード・マネージャー操作のスパン.....	229
CICS 領域のパフォーマンス・ゴール.....	229
CICS ワークロードの種別規則の定義.....	230
サービス・クラスの定義.....	231
CICS パフォーマンス・パラメーターのサービス方針への適合.....	232
第 3 章 イベント処理パフォーマンスの改善.....	233
特記事項.....	237
索引.....	243

本書について

本書では、パフォーマンスの制約を識別し、操作可能な CICS システムとそのアプリケーション・プログラムを調整する方法について説明します。以下にリストする他の PDF は、CICS の特定の領域で発生する問題への対処方法について説明するもので、本書と共にそれらも参照することが必要な場合があります。(IBM Knowledge Center では、これらのすべての情報が「パフォーマンスの改善」という 1 つのセクションの下にまとめられています。) さらに「モニター・データ・リファレンス」および「統計リファレンス」という関連する 2 つの解説書の PDF が必要となる場合もあります。CICS TS V5.4 より前は、これらの 2 つの解説書の PDF は、「パフォーマンス・ガイド」に含まれていました。

CICS の領域のパフォーマンスについては、以下の PDF を参照してください。

- ONC/RPC インターフェースについては、「外部インターフェース・ガイド」を参照してください。
- Java™ および Liberty については、「CICS での Java アプリケーション」を参照してください。
- フロントエンド・プログラミング・インターフェースについては、「フロントエンド・プログラミング・インターフェース・ユーザズ・ガイド」を参照してください。
- DBCTL については、「IMS DB コントロール・ガイド」を参照してください。
- 共用データ・テーブルについては、「共用データ・テーブルの手引き」を参照してください。
- システム間パフォーマンスについては、「相互通信ガイド」を参照してください。

本書で使用している用語および表記について詳しくは、IBM Knowledge Center の [CICS 資料で使用されている表記規則および用語](#)を参照してください。

この PDF の作成日

この PDF は、2020 年 5 月 28 日に作成されました。

第1章 測定、調整、およびモニター: 基本

良好なパフォーマンスが得られると、システム・リソースが最大限に活用され、サービス・レベル・アグリーメントを効率的に実現するのに役立ちます。

このタスクについて

以下の時点で、CICS システムのパフォーマンスを考慮する必要があります。

- 新規システムのインストールを計画する
- 既存システムを検討する
- システムの主要な変更を計画する

以下の手順は、システムを調整するための基本ステップを示しています。

手順

1. 良好なパフォーマンスについて同意する。
2. パフォーマンス目標を設定し、それらの測定方法を決定する。
3. 実動システムのパフォーマンスを測定する。
4. 必要に応じてシステムを調整する。
5. システム・パフォーマンスのモニターを継続し、将来の制約を予測する。

24 ビット、31 ビット、および 64 ビット・アドレッシング

ビット世代がパフォーマンスに与える影響

zOS のアーキテクチャーでは、仮想記憶とアドレス・スペースを使用します。あるアドレス・スペースで実行されるプログラムは、そのアドレス・スペースに関連付けられたすべてのストレージを参照できます。元の MVS™ (zOS より前) のアーキテクチャーでは、ストレージ・アドレスが 24 ビット長と定義されていたため、16 MB のアドレス・スペースを各ユーザーに割り振ることができました。後にアーキテクチャーのアドレス可能度が 31 ビットに拡張されたことで、仮想記憶のアドレス可能度とアドレス・スペースのサイズが 16 MB から 2 GB に増えました。この新しいアーキテクチャーのために、ユーザーが既存のアプリケーション・プログラムを変更する必要はありませんでした。もともと 24 ビット・アドレッシングで実行するように設計された既存のプログラムのための互換性が用意されていました。その後、2000 年の IBM eServer™ zSeries メインフレームのリリースで、IBM はアーキテクチャーのアドレス可能度をさらに 64 ビットに拡張しました。64 ビット・アドレッシングにより、各アドレス・スペース (64 ビット・アドレス・スペースと呼ぶ) のサイズは最大 16 EB になりました (1 エクサバイトは 10 億ギガバイトよりもやや大きい)。

注: アドレス・スペースのサイズは最大 16 EB になりましたが、デフォルトでは z/OS はこれまでと同様に 2 GB サイズのアドレス・スペースを作成します。この制限をアドレス・スペースを超えるのは、その中で実行されるプログラムが 2 GB アドレスを超えて仮想記憶を割り振る場合のみです。その場合、z/OS は、ユーザーが使用できるストレージを 2 GB から 16 EB に増やします。プログラムを 64 ビット (2 GB 境界より上) ストレージ内で実行することはできません。64 ビット・ストレージは、2 GB より下のアドレス・スペースにロードされるプログラムのユーザー・データを保管する目的に制限されます。

z/OS および zSeries メインフレーム上で実行されるプログラムは、24 ビット、31 ビット、または 64 ビット・アドレッシングで実行できます (必要に応じてこれらを切り替えることが可能)。64 ビット・アーキテクチャーで使用する高位仮想記憶域をアドレス指定するためには、プログラムは 64 ビット固有の命令を使用し、64 ビット・アドレス・モード (AMODE(64)) で実行されなければなりません。アーキテクチャーには固有の 64 ビット命令が導入されていますが、プログラムは 31 ビット命令と 64 ビット命令の両方を必要に応じて使用できます。zOS 用語では、次のように呼びます。

- 24 ビット・ストレージ (16 M 以下) は、16 MB 境界より下のストレージと呼ばれます。
- 31 ビット・ストレージ (16 M から 2 GB まで) は、16 MB 境界より上のストレージと呼ばれます。
- 64 ビット・ストレージ (2 GB から 16 EB まで) は、2 GB 境界より上のストレージと呼ばれます。

CICS トランザクション・フロー

このセクションでは、CICS がどのようにトランザクションを処理するかについて説明します。

CICS でオンライン・セッションを開始するには、通常、CICS に対して自分の身元を証明するプロセスである「サインオン」から始めます。CICS にサインオンすると、特定のトランザクションを呼び出す権限が与えられます。サインオンしたら、使用する特定のトランザクションを呼び出します。CICS トランザクションは通常、1 文字から 4 文字のトランザクション ID (TRANSID) によって識別されます。この ID は、トランザクションの処理に使用する初期プログラムを指定する表に定義されています。

アプリケーション・プログラムは、プロセッサに接続されている直接アクセス・ストレージ・デバイス (DASD) 上のライブラリーに保管されています。それらは、システムの始動時にロードすることも、必要に応じて単純にロードすることもできます。プログラムがストレージに存在しても使用されていない場合、CICS はそのスペースを他の用途のために解放します。プログラムが次に必要になったときに、CICS は新しいコピーをライブラリーからロードします。

あるトランザクションを処理中に、システムが複数の端末からメッセージを受信することがあります。各メッセージについて、CICS はアプリケーション・プログラムをロードし (まだロードされていない場合)、それを実行するタスクを開始します。このようにして、複数の CICS タスクを同時に実行できます。

CICS は、タスクごとに別の制御スレッドを維持します。例えば、あるタスクが、ディスク・ファイルの読み取りや端末からの応答の取得を待機しているときには、CICS は他のタスクに制御を与えることができます。タスクは CICS タスク管理プログラムが管理します。

CICS は、マルチタスキングと、タスク自体からのサービス (オペレーティング・システムのサービスまたは CICS 自体のサービス) に対する要求を両方とも管理します。このプロセスによって、オペレーティング・システムがタスクのために要求を実行するのをそのタスクが待機している間に、CICS 処理を続けることができます。CICS が管理する各トランザクションは、実行する準備ができたものの中で優先度が最も高くなったときに、プロセッサの制御を与えられます。

アプリケーション・プログラムは実行中にさまざまな CICS 機能を要求して、端末との間でのメッセージ送信を処理したり、必要なファイルまたはデータベース・アクセスを処理したりします。アプリケーションが完了すると、CICS は端末をスタンバイ状態に戻します。

プログラム制御

1 つのトランザクション (タスク) はその処理の中で複数のプログラムを実行することができます。

プログラム定義には、CICS システム内のアプリケーションが使用するすべてのプログラムについて 1 つのエントリーが含まれています。各エントリーには、他の事柄と並んで、プログラムが作成された言語が含まれています。トランザクション定義にはシステム内のすべてのトランザクション ID について 1 つのエントリーが含まれています。各トランザクションに関して保持されている重要な情報は、ID およびトランザクションのために実行される最初のプログラムの名前です。

これらの 2 セットの定義と、トランザクションおよびプログラムは、次のように連携して機能します。

1. ユーザーがトランザクション ID を端末に入力します (もしくは前のトランザクションによって ID が決定されます)。
2. CICS はインストールされたトランザクション定義のリストでその ID を検索します。これにより、CICS はどのプログラムを最初に呼び出すかを認識します。
3. CICS はインストールされたトランザクション定義のリストでこのプログラムを検索し、その場所を見つけてロードします (プログラムがまだ主ストレージにない場合)。
4. CICS は、トランザクションと端末の特定の組み合わせに必要な制御ブロックを、両方の定義セットからの情報を使用して作成します。コマンド・レベル COBOL のプログラムの場合は、このプログラムの特定の実行のために作業用ストレージの専用コピーの作成も行われます。
5. CICS が制御をプログラムに渡すと、プログラムがこの端末用の制御ブロックを使用して実行を開始します。このプログラムは、必要であれば、インストールされたプログラム定義リスト内の他のプログラムに (トランザクションの実行中に) 制御を渡すことができます。

プログラム間で制御を渡すための CICS コマンドは 2 つあります。1 つは LINK コマンドで、COBOL の CALL ステートメントに似ています。もう 1 つは XCTL (制御権移動) コマンドで、COBOL にはこれに相当す

るものはありません。プログラムが他のプログラムにリンクしても、1つ目のプログラムは主ストレージにとどまります。もう一方の(リンク先)プログラムが終了して制御を渡すと、1つ目のプログラムが LINK 後のポイントから再開されます。リンク先のプログラムは、リンク元のプログラムより1つ下の論理レベルで実行されるものと見なされます。

これとは対照的に、あるプログラムが別のプログラムに制御を移した場合、1つ目のプログラムは終了したものと見なされ、2つ目のプログラムは1つ目のプログラムと同じレベルで動作します。2つ目のプログラムが終了すると、制御は1つ目のプログラムに戻されるのではなく、最後に LINK コマンドを実行したプログラムに戻されます。

CICS 自体がこのプロセスで最高のプログラム・レベルで、トランザクションの1つ目のプログラムがその下のレベルだと考えられることもあります。

CICS アプリケーション設計の正常な原則は、表示ロジックをビジネス・ロジックから分離します。プログラム間の通信は LINK コマンドを使用して実行され、それらのプログラム間のデータの受け渡しは COMMAREA で行われます。このようなモジュラー設計により、機能を分離できるだけでなく、新しい表現方式を使用して既存アプリケーションを非常に柔軟に Web 化できるようになります。

パフォーマンス・データをモニターおよび収集するための CICS の機能

CICS 統計、モニター、およびトレース機能を使用して、パフォーマンス・データを収集してモニターし、CICS システムを最適に調整するのに役立てることができます。

CICS 統計

CICS 統計は、CICS システムを恒常的にモニターするための最も単純で最も重要なツールです。この統計は、タスクに関係なく、CICS システム全体の情報を収集します。

詳しくは、「[CICS 統計の概要](#)」を参照してください。

CICS モニター

CICS モニターは、後でオフライン分析ができるよう、オンライン処理中に、ユーザーおよび CICS のすべてのトランザクションのパフォーマンスに関するデータを収集します。

詳しくは、「[CICS モニタリングの概要](#)」を参照してください。

CICS トレース

システム対話を含むより複雑な問題の場合、CICS トレースを使用して、CICS 管理モジュールを通過する CICS トランザクションの進行状況を記録できます。

CICS トレースは、特定の状態に至るイベントのヒストリーを提供します。

また、CICS トレース機能は、システムにおけるイベントの待機超過などのパフォーマンス上の問題、または非効率的なシステム・セットアップやアプリケーション・プログラム設計によって生じる制約の分析にも役立ちます。

詳しくは、「[CICS trace](#)」を参照してください。

パフォーマンス・データをモニターおよび収集するための CICS ツール

CICS ツールを使用して、CICS システムのチューニングに役立つパフォーマンス・データを収集およびモニターできます。

CICS Performance Analyzer for z/OS (CICS PA)

CICS Performance Analyzer は、CICS システムおよびアプリケーションのパフォーマンスに関する情報を提供するパフォーマンス・レポート作成およびパフォーマンス分析ツールです。CICS システムを効果的にチューニング、管理、および計画するのに役立ちます。CICS PA は、CICS および接続されているサブシステム (Db2®、IMS、MQ など) から収集されたデータを使用して、CICS システムおよびアプリケーションのパフォーマンスと統計に関する履歴情報を提供します。

CICS PA を使用して包括的なパフォーマンス・レポート作成およびパフォーマンス分析を行うことで、CICS システムの効率を評価し、システム・ボトルネックを除去し、スレッド・セーフ分析を含めてシステム・パフォーマンスをプロアクティブにチューニングできるようになります。250 を超えるカスタマイズ可能な定義済みのレポートと、ユーザー独自のレポートおよび抽出を作成するための機能が用意されています。

CICS Explorer® 用の CICS PA プラグインを使用すると、パフォーマンス・データが CICS Explorer に組み込まれるため、例えば Explorer で CICS リソースを右クリックして、そのリソースに関するパフォーマンス履歴情報を表示したりできます。

CICS PA の詳細については、[CICS Performance Analyzer for z/OS](#) を参照してください。

CICS Interdependency Analyzer for z/OS (CICS IA)

CICS Interdependency Analyzer は、CICS リソースを検出して分析し、それらのリソース間の関係を明らかにすることができる生産性向上ツールです。

CICS IA® は、CICS システム内の主要なリソース間のランタイム関係を動的に検出します。この検出は、アプリケーションをモニターして API および SPI コマンド (必要に応じて Db2、IMS、MQ、および COBOL 呼び出しも含む) を特定することで行われます。これにより、アプリケーションの全体像、対話、参照されるリソース、および相互関係がわかります。CICS IA を使用して、スレッド・セーフに関する考慮事項についてアプリケーションを分析し、全体的な実行効率を向上させることができます。

CICS IA は、そのデータを Db2 データベースに保管します。このデータに、CICS IA レポート作成コンポーネントを使用してオフラインでアクセスし、照会およびレポート作成を行うことができます。CICS IA には CICS Explorer 用のプラグインも用意されています。このプラグインを使用すると、例えば、Explorer で CICS リソースを右クリックして、そのリソースに関連するすべてのリソースを表示したりできます。

CICS IA のコマンド・フロー機能を使用すると、すべての EXEC CICS、SQL、MQ、および IMS 呼び出しをキャプチャーして日時順に表示できます。この機能を、TCB モード切り替えを検出する機能と一緒に実行すると、チューニングおよびスレッド・セーフ分析のためにアプリケーションの実行中のフローを診断する際に役立ちます。

CICS IA の詳細については、[CICS Interdependency Analyzer for z/OS](#) を参照してください。

パフォーマンス・データを取得するためのその他のツール

CICS には付属していない各種のツールを使用して、パフォーマンス関連の情報を提供すると、CICS システムを最適に調整するのに役立ちます。

IBM Redbooks® [ABCs of z/OS System Programming](#) には、キャパシティー・プランニング、パフォーマンス管理、RMF、および SMF に関する情報が記載されています。

リソース測定機能 (RMF)

リソース測定機能 (RMF) は、プロセッサ活動 (WAIT 時間)、入出力活動 (チャネルおよび装置使用量)、主記憶域活動 (要求およびスワップ・ページング統計)、およびシステム・リソース・マネージャー (SRM) 活動 (ワークロード) を説明する、システム全体のデータを収集します。

RMF は、システム活動をモニターして、パフォーマンスとキャパシティー・プランニング・データを収集する中央測定ツールです。RMF レポートの分析を行うと、ユーザー要求に対してシステムをチューニングするための基礎的データが得られます。それらのデータを リソース使用量の追跡に使用することもできます。

RMF は以下の活動を測定します。

- プロセッサ使用量
- アドレス・スペース使用量
- チャネル活動
 - 物理チャネル当たりの要求率およびサービス時間
 - 論理チャネルと物理チャネル間の関係
 - 論理チャネル・キュー項目数およびキューイングの理由

- 以下の装置の装置活動および競合
 - ユニット・レコード
 - グラフィックス
 - 直接アクセス・ストレージ
 - 通信装置
 - 磁気テープ
 - 文字読取装置
- 詳細なシステム・ページング
- 詳細なシステム・ワークロード
- ページおよびスワップ・データ・セット
- エンキュー
- CF 活動
- XCF 活動

RMF では、z/OS ユーザーは以下のことができます。

- システムの応答性の評価
 - ボトルネックの識別。 ページおよびスワップ・データ・セット活動に関連した 詳細なページング・レポートにより、仮想記憶環境の動作の状況をよく理解できます。
- チューニングの影響の検査
 - 結果を画面上で動的に、またはポストプロセッシング機能によって監視できます。
- キャパシティー・プランニング評価の実行
 - ワークロード活動報告書には、プロセッサ、入出力、および主記憶域サービスなどの 主要エレメントによって分解されたインターバル・サービスが含まれます。
 - リソース・モニター出力 (例えば、システム競合インディケーター、カテゴリーごとに分割された スワップアウト、ドメインごとの平均作動可能ユーザー) の分析は、ユーザー環境の理解および傾向の予測に役立ちます。
 - ポストプロセッシング機能により、ピーク・ロード期間の分析およびトレンド分析が 簡単になります。
- MVS がサポート できる大規模ワークロードおよび増大したリソースの管理
- オンライン・チャネル・パス使用量の識別および測定

RMF について詳しくは、IBM Redbooks 資料「[ABCs of z/OS System Programming](#)」および「[z/OS リソース測定機能 \(RMF\) ユーザーズ・ガイド](#)」を参照してください。

Tivoli OMEGAMON XE for CICS on z/OS

Tivoli® OMEGAMON® XE for CICS on z/OS は、複合 CICS システムのパフォーマンスおよび可用性をプロアクティブに管理するのに役立ちます。

Tivoli OMEGAMON XE for CICS on z/OS (OMEGAMON XE for CICS on z/OS) は、z/OS 管理システム上で実行されるリモート・モニター・エージェントです。パフォーマンス上の問題を予測するのを支援し、CICS 環境で重大なイベントが発生すると警告します。CICS 領域内のイベントが重大なポイントに達したときに警報するように、しきい値レベルとフラグを設定できます。

Tivoli Enterprise Portal 下で実行されている場合、IBM Tivoli OMEGAMON XE for CICS on z/OS は、CICS Transaction Server の一元管理点として機能し、CICS 領域内の問題を検出して防止するために必要な情報を収集する包括的な手段を提供します。Tivoli Enterprise Portal が収集するデータを表やグラフに表示して、管理対象の CICS 領域の状況を示します。

このデータを使用して、次のようなさまざまな作業が行えます。

- 信頼できる最新のデータを収集して分析し、より迅速に、適切な情報に基づいて運用上の決定を下せるようにする
- すべての CICS 領域を単一点から管理し、随時に問題を特定する

- さまざまな領域間でワークロードのバランスを取る
- 目標に照らしてパフォーマンスを追跡する

OMEGAMON XE for CICS on z/OS を使用すると、システム管理者はしきい値レベルおよびフラグを設定し、システムの状態がそのしきい値に達したときに警報を出すようにすることができます。以下は、拡張モニター機能です。

- しきい値に基づくユーザー定義および事前定義の状態で、異なるタイプのアラートを生成する
- すべての CICS 領域の状況を一覧表示する
- 1 つ以上の集中型ワークステーションから同時に複数の CICS 領域をモニターする機能

他の OMEGAMON XE モニター製品と一緒に、OMEGAMON XE for CICS on z/OS から提供されるデータ、分析、およびアラートを使用して、単一のコンソールからコンピューティング・エンタープライズ全体の正常性を把握することができます。

OMEGAMON XE for CICS on z/OS について詳しくは、[IBM Tivoli OMEGAMON XE for CICS on z/OS](#) を参照してください。

IBM Z Decision Support

IBM Z[®] Decision Support (旧称 Tivoli Decision Support for z/OS) は、CICS および他の IBM システムおよび製品のデータを収集し、分析する IBM 製品です。

IBM Z Decision Support によって生成されるレポートは、以下の目的のために役立ちます。

- システムの概要の入手
- サービス・レベルの維持の確認
- 可用性の確認
- パフォーマンスの調整
- キャパシティー・プランニング
- 変更および問題の管理
- アカウンティング

多くの既成のレポートを使用できます。特定の要求に合う独自のレポートを作成することもできます。

レポートにおいて、IBM Z Decision Support は、CICS モニターおよび統計のデータを使用します。IBM Z Decision Support は、MVS システムのデータ、および RMF、TSO、IMS、NetView[®] などの製品のデータも収集します。つまり、CICS と他のシステムのデータを共に表示したり、別々のレポートに表示したりできます。

レポートは、図、棒グラフ、円グラフ、タワー図表、ヒストグラム、面グラフ、およびその他のグラフ形式で表すことができます。IBM Z Decision Support は、データおよび詳細なフォーマット設定を、残りの作業を実行する IBM Graphic Data Display Manager (GDDM) に渡します。) IBM Z Decision Support は、GDDM が使用できないか、または出力装置がグラフをサポートしない場合は、文字グラフィックを使用して折れ線グラフおよびヒストグラムを作成することもできます。正確な数値が必要なレポートの場合は、テーブルやマトリックスなどの数値レポートがより適しています。

IBM Z Decision Support を使用して CICS のパフォーマンスを報告するには、IBM Z Decision Support CICS パフォーマンス・フィーチャーを使用します。詳しくは、[IBM Z Decision Support for Capacity Planning](#) を参照してください。

パフォーマンスのモニターおよび検討

ユーザーのニーズに最適な方針を実装して、CICS のパフォーマンスのモニター、測定、および分析を行うことができます。

パフォーマンス目標を設定し、CICS のパフォーマンスを分析するために、複数のモニター手法を使用できます。

モニターの活動および手法の確立

モニター活動およびモニター手法を含めた継続的な方針を確立すると、CICS 実動システムについて理解し、最適なパフォーマンスを確保して、予期しない問題を回避するのに役立ちます。

モニターは、データの収集と解釈によって、CICS 実動システムのパフォーマンスを目標に照らして定期的に確認することを説明する用語として使用されます。分析は、パフォーマンス低下の理由を調査するための手法を表します。チューニングは、この分析から生じる何らかの処置を表します。

モニターは、次のようなさまざまな理由で行われる継続的な活動です。

- システム・キャパシティを予測するためのトランザクション・プロファイル(つまり、ワークロードおよび量) および統計データを設定できる
- 比較データによって、パフォーマンス上の問題を回避するために早い段階で警告を出せる
- 以前、パフォーマンス上の問題に対応するために行ったチューニングを測定し検証できる

パフォーマンス・ヒストリー・データベース(例えば、36 ページの『[IBM Z Decision Support](#)』を参照)は、システム・パフォーマンスに関する問題の対処方法、および将来のチューニングを計画するための重要な情報源です。

モニターは、方針、手順、およびタスクの観点から説明できます。

方針には、以下の要素が含まれます。

- ワークロードの継続的または定期的要約。すべてのトランザクションまたは選択した 代表的トランザクションを追跡できます。
- 標準またはピーク負荷時のスナップショット。以下のために、ピーク・ロードをモニターします。
 - ピーク・ボリュームのときに制約と遅い応答がより明白になります。
 - 現行ピーク負荷は、将来の平均負荷の指標に適しています。

手順(優れた文書化の手続きなど)は、モニター方針とタスク間の管理リンクを提供します。

タスク(CICS トランザクションのタスク・コンポーネントと混同しないでください)には、以下が含まれます。

- 1 つ以上のツールの実行(23 ページの『[パフォーマンス測定ツール](#)』を参照してください)
- 出力の照合
- 傾向を知るための出力の検討

これらのタスクの責任をオペレーション担当者、プログラミング担当者、分析担当者の間に割り振ります。重要なリソースを識別し、これらのリソースの使用における傾向を強調する手順を設定します。

ツールはリソースを必要とするため、実動システムのパフォーマンスを低下させる場合があります。

新規アプリケーションおよびシステム全体の両方について、活動のピーク期間を重視します。予想したピークが実際のピークに一致することを確認するために、必要であれば最初にツールを頻繁に実行します。

多くの場合、すべての明細出力を保持することは実用的ではありません。要約レポートを、対応する CICS 統計と共にファイルし、通常の保護機能を使用して、合意された期間、ツールからの出力を保持します。

システム・パフォーマンスの 1 つか 2 つのスナップショットに基づくのではなく、長期間にわたって異なる時刻に収集されたデータを基にして、基本的な結論を出します。ピーク負荷を重視してください。異なるツールが異なる測定基準を使用するため、初期の測定値は、明らかに矛盾した結果を示す場合があります。

モニター手順を事前に計画します。手順には、使用するツール、使用する分析手法、それらの活動の運用範囲、および活動の実行頻度を記述する必要があります。

モニターの活動および手法の開発

方針に整合したデータを収集および分析するには、適切なツールとプロセスが整っている必要があります。モニターおよびパフォーマンス分析の主要計画を開発する場合は、以下のポイントを考慮します。

- モニター活動の主要スケジュールを確立します。オンライン・イベントのフィードバックや、毎日または定期的なデータ収集の指示を考慮に入れて、モニターと運用手順を調整します。
- 業務をシステム・パフォーマンスに関連付けて考慮します。例えば、アプリケーションの使用におけるトランザクション比率や変更の増加、および将来の傾向を考慮します。アプリケーションの異常終了、頻発する問題、過度の再試行など、パフォーマンス以外のシステム問題の影響を考慮します。
- モニターに使用するツールを決定します。データ収集に使用するツールには、動的モニター、毎日の統計収集、および詳細なモニターを行う機能が必要です。詳しくは、[8 ページの『モニター・スケジュールの計画』](#)を参照してください。
- 実行する分析の種類を考慮します。インストール・システムの管理のために既に設定済みの制御を考慮に入れます。モニター出力からどのデータを抽出するかを文書化し、データのソースおよび使用法を決定します。モニター・ツールによって提供されるフォーマット済みレポートは、データ量を編成するのに役立ちますが、場合によっては、データの抽出および縮小に役立つワークシートを設計します。
- 調査結果の検討を行う担当者のリストを作成します。モニター・データの分析結果や結論は、ユーザー連絡窓口グループおよびシステム・パフォーマンス専門担当者と共有する必要があります。
- チューニングの推奨から発生した、CICS システム 設計 に対する変更点を実装する方針を作成します。推奨事項をインストール管理手順に組み込みます。テストの規格や、実稼働環境に対して許可される変更の頻度などの項目を含めます。

パフォーマンス検討プロセスの計画

パフォーマンス・レビュー・プロセスの計画には、モニター手順の実装に必要なツールと分析のチェックリストが含まれます。モニター手順の簡単なスケジュールを確立します。パフォーマンス・レビュー・プロセスを作成するには、以下のタスクを実行します。

- 各タイプのタスクによって行われる CICS 要求のリスト作成。これにより、統計および CICS モニター機能レポートにおいて、調べる必要のある要求または リソース (高い頻度または高いコストのもの) を決定できます。
- 検討問題のチェックリストの作成。
- 新規アプリケーションのリソース 使用量およびシステム 負荷の見積もり。これによって、比較を開始するための最初の基準を設定できます。

モニター・スケジュールの計画

包括的なモニター計画には、さまざまなシステム活動をさまざまな時間間隔でスケジューリングすることが含まれます。このアプローチにより、CICS システムのパフォーマンスを測定および分析するための幅広いデータを収集できます。動的なモニターとスケジュールされたモニターの両方を計画します。

動的モニター

動的なモニターは、常時実行できる「現場での」モニターです。このタイプのモニターには、以下の活動が含まれます。

- パフォーマンス目標からの重大な短期の逸脱を発見するための 継続したシステム・オペレーションの監視。このアクティビティーのために、エンド・ユーザーのフィードバックは重要です。リソース測定機能 (RMF) を使用して、プロセッサ、チャンネル、カップリング・ファシリティ、および入出力装置の使用量に関する情報を収集することもできます。
- 状況情報の取得。オンライン実行中のシステム 処理に関する状況情報を取得できます。この情報には、キュー・レベル、アクティブな領域、アクティブな端末、および会話型トランザクション の数およびタイプが含まれます。マスター端末オペレーターによって起動された自動プログラムの助けを借りて、この情報を取得できます。実動サイクルの編成前の時 (メッセージのスケジュール前、一部のネットワークのシャットダウン時、またはピーク負荷時) に、このプログラムは、システム・リソース・レベルの、トランザクション処理の状況および測定値を収集できます。
- CICSplex® SM モニター・データの使用。CICSplex SM は、CICS のモニター機能で生成された情報を集計することができ、動的なモニター活動を支援します。データは、オンラインですぐに見ることができるため、トランザクションのパフォーマンスに関する 即時フィードバックを得ることができます。CICSplex SM で CICS のモニター情報を収集するには、CICS モニター機能がアクティブでなければなりません。

毎日のモニター

データを毎日モニターして、主要なシステム・パラメーターを測定して記録します。データの毎日のモニターは、通常、イベントのカウントおよび全体レベルの時間測定値によって構成されます。場合によっては、タイミング数はCICSシステム全体で平均されます。データを毎日モニターするには、一連のタスクを実行します。例えば、次のようになります。

- メッセージ、タスク、プロセッサ使用量、入出力イベント、使用されたストレージなどの項目の毎日の平均とピーク期間(通常1時間)の平均の両方を記録します。これらのイベントを主要なパフォーマンス目標と比較し、パフォーマンスの低下が発生していないかどうかを確認します。
- CICSの各実行の最後にCICSが提供する統計をリストします。提供されるデータに日付スタンプとタイムスタンプを付け、後で検討するためにそれをファイルします。例えば、安定したインストール・システムにおいて、週の終わりに毎日のデータを検討する場合があります。通常、どのタイプのモニター・データでも、収集頻度より低い頻度で検討を実行できます。問題があることがわかったら、検討の頻度を多くします。例えば、使用可能になったらすぐに毎日のデータを検討します。
- シャットダウン時以外のときに統計を提供するCICSのすべての機能に熟知している必要があります。主要機能は、端末からの呼び出し(カウンターがリセットされる場合とされない場合がある)と、時刻によって開始される自動要求です。
- 実行中に報告された発生事象について非公式メモをファイルします。例えば、統計内のギャップの原因になるCICSのシャットダウン、遅い応答時間に対するユーザーからの苦情、サービス休止となった端末、その他の重要な項目を含めます。このようなメモは、後に発見される可能性のある詳細なパフォーマンスの数値における不均衡を調整するときに役立ちます。
- CICSがアクティブであった期間のシステム・コンソール・ログを印刷します。また、同時バッチ活動の観点からCICSシステム・パフォーマンスの検討が必要になる場合のために、コンソール・ログのコピーをファイルします。
- 負荷に変動がある場合は、少なくとも1日の一定時間、23ページの『パフォーマンス測定ツール』で説明されているパフォーマンス分析ツールのいずれかを実行します。使用するツールによって作成されるレポートの要約をファイルします。
- 開発後レビューの段階で、使用頻度が一貫して高いと確認された項目をグラフに転記します。
- CICS統計、モニター・データ、およびRMFデータをIBM Z Decision Support データベースに収集します。

週次モニター

システム指向の目標やワークロード・プロファイルと比較するために、システムの運用に関する詳細な統計を定期的に収集します。週次のモニター・データでは、以下のステップを実行します。

- パフォーマンス・クラスをアクティブにしてCICSモニター機能を実行し、処理します。モニター機能は毎日実行する必要はない場合がありますが、定期的に実行し、ソートされた要約出力と詳細なレポートを保持することが重要です。このファシリティーを同じ曜日に実行するかどうかは、システム負荷の性質によって異なります。例えば、特定の曜日が他の曜日よりシステム負荷が大きい場合は、この日にモニターします。ただし、特にパフォーマンス・クラスをアクティブにした場合は、モニター機能を使用すると、負荷が増すことに注意してください。
- 負荷が毎日同じことが明らかな場合は、負荷を確認するのに十分な期間だけ毎日CICSモニター機能を実行します。実際にCICSの負荷が日によってほとんど変わらない場合は、同じ方法で、ログの同時バッチ負荷を検査します。バッチ負荷の検査により、週の特定の曜日におけるピーク・ボリュームや異常なトランザクションの混合による潜在的な問題を識別できます。最初の数週間のCICS統計の出力も役立つ情報を提供します。詳細なモニター・レポート出力を毎回検討する必要はない場合がありますが、統計やユーザーのコメントで提起された問題に対処するには要約データでは不十分な場合があるため、この出力を常に保持しておく必要があります。CICSモニター機能出力にラベルを付け、追加調査が必要な場合に備えて、合意した期間それを保持します。
- RMFを実行します。これは、入出力使用量、チャンネル使用量、およびその他の使用量を表示します。サマリー・レポートをファイルし、出力情報を同意期間だけ保存します。
- CICS統計、および問題レポートを検討します。
- 重要なパラメーターのグラフを検討します。重大レベルに近づいている項目がある場合は、パフォーマンス分析およびRMF出力で詳細を確認します。

- 将来の参照用の要約として、値を表にしたり、グラフ化したりします。
- 週ごとの IBM Z Decision Support、または CICS Performance Analyzer レポートを生成します。

月次モニター

傾向をモニターして評価します。これは、長期間にわたって定期的に追跡すると、より適切に反映されます。以下のリストに、月次ベースでデータをモニターするためのタスクをいくつか示します。

- RMF を実行します。
- RMF およびパフォーマンス分析のリストを検討します。リソースを使用し過ぎている兆候がある場合には、以前に同意した手順に従ってから (例えば、管理部に通知する)、さらにモニターを続けます。
- RMF 出力に日付スタンプとタイム・スタンプを付け、パフォーマンス上の問題が発生し始めたときに使用できるように、保持します。コンポーネント使用量の詳細な知識が重要になる場合は、その出力を使用して見積りもりの作成をすることもできます。RMF 出力によって、プロセッサ使用量、DASD の使用、およびページング率など、システム内のリソース使用量に関する詳細なデータを入手できます。
- 毎月、長期間の傾向を示す IBM Z Decision Support レポートを作成します。

将来のためのモニター

パフォーマンスを許容できる場合は、システム・パフォーマンス測定値が応答時間の問題を引き起こすようになる前に、その測定値をモニターし、パフォーマンス制約を予測する手順を設定します。効果的なモニター方式では、例外報告手順が重要です。複雑な実動システムでは、多くの場合、毎日包括的に検討できないほど多くのパフォーマンス・データがあります。パフォーマンス低下の主要要素は、経験によって識別できるため、それらの要素を最も詳しくモニターします。使用量およびこのプロセスに役立つその他の要因 (バッチ・スケジュールなど) の傾向を識別します。

パフォーマンス検討の典型的質問

パフォーマンス・データを検討するときには、チェックリストの基本として以下の質問を使用します。これらの質問の多くは、CICS Performance Analyzer または IBM Z Decision Support for z/OS などの、パフォーマンス報告パッケージによって答えることができます。

パフォーマンスと厳密に関連していない問題もあります。例えば、トランザクション統計が、異常条件プログラムの使用によるトランザクションの異常終了を高頻度で示している場合は、サインオン・エラーがある可能性があり、したがって端末オペレーターのトレーニング不足が考えられます。この状態はパフォーマンス上の問題ではありませんが、モニターによって提供できる追加情報の一例を示しています。

1. トランザクション・ワークロードには、どのような特徴がありますか？
 - a. 各トランザクション ID の使用頻度が変化しましたか？
 - b. 一日の特定の時間と別の時間では、混合は変化しますか？
 - c. これを検査するために、日中に統計を要求する頻度を増やす必要がありますか？

次の場合は、別の方法を採用する必要があります。

- すべてのメッセージが同じ初期のタスクおよびプログラム (ユーザー・セキュリティ・ルーチンの場合、初期の編集またはフォーマット設定、統計分析など) を通過するシステム
- 一連の長いメッセージ・ペアが、単一トランザクションによって反映される会話型トランザクション
- 実行される作業量が入力データに著しく依存するトランザクション

これらの場合は、CICS のプログラム統計、ファイル統計、またはその他の統計を適切に参照して、プログラムまたはデータ・セットの使用による機能を識別する必要があります。また、ユーザー・タグをモニター・データ (例えば、CICS モニター機能の場合のユーザー文字フィールド) に入れることもできます。これにより、CICS Performance Analyzer for z/OS または IBM Z Decision Support などの製品による分析の基礎として、モニター・データを使用することができます。

2. 通信回線の使用量はどのくらいですか？
 - a. CICS 端末統計が、各回線の端末において、メッセージ数の増加を示していますか？

- b. CICS パフォーマンス・クラス・モニター・レポートの平均メッセージ長が、トランザクション・タイプによって変化しますか？回線またはフィールド出力の数が入力データに依存するアプリケーションの場合、このことは簡単に起こります。
 - c. 端末エラー数は受け入れ可能ですか？端末エラー・プログラムまたはノード・エラー・プログラムを使用する場合、何らかの回線の問題がありますか？
3. DASD の使用量はどのくらいですか？
- a. ファイル制御に対する要求数は増加していますか？CICS は、行われた論理要求数を記録することに注意してください。物理入出力操作数は、索引の構成、および制御間隔およびバッファ割り振り当たりのデータ・レコード数によって異なります。
 - b. 区画内一時データの使用量は増加していますか？一時データには、キュー混合に依存する入出力操作数が含まれます。少なくとも、行われた要求数を検討して、以前の実行に比べて一時データがどうなっているかを確認します。
 - c. 補助一時記憶域の使用量は増加していますか？一時記憶域は制御間隔アクセスを使用しますが、制御間隔を書き出すのは、同期点またはバッファがいっぱいの場合のみです。
4. 仮想記憶域の使用量はどのくらいですか？
- a. 動的ストレージ域の大きさはどれくらいですか？
 - b. GETMAIN 要求数は、タスクの数およびタイプと矛盾していませんか？
 - c. 頻繁にストレージ不足 (SOS) 状態になりますか？
 - d. デッドロック・タイムアウト・インターバル (DTIMOUT) の有効期限後に消去されるタスクについて、何らかの問題が報告されましたか？
 - e. プログラム・ロード活動はどのくらいありますか？
 - f. モニター・レポート・データの、タスク・タイプごとの動的ストレージ使用は予想通りですか？
 - g. CICS のそれぞれの実行において、ストレージ使用量は同じ程度ですか？
 - h. 関数の最初の呼び出しが後続のものより長くかかることを示す問題レポートがありますか？この状態は、例えば特に IMS において、ロードされたプログラムがその後でデータ・セットを開く必要がある場合に発生する可能性があります。アプリケーション設計の変更によって問題を修正できますか？
5. プロセッサの使用量はどのくらいですか？
- a. モニター・レポートによって測定されたプロセッサ使用量が以前の監視と矛盾しませんか？
 - b. 実行する予定のバッチ・ジョブを正常に実行できますか？
 - c. CICS より高い優先順位で実行される機能の使用量が増えていますか？優先順位の低い領域のために、MVS リーダーおよびライター、MVS JES、および z/OS Communications Server が、CICS より上位、および全体の入出力より上位で実行している場合は、それらを含めてください。
6. カップリング・ファシリティーの使用量はどのくらいですか？
- a. 平均のストレージ使用量はどのくらいですか？
 - b. リンクの使用率はどのくらいですか？
7. 設計、コーディング、またはオペレーションのエラーを示す数値がありますか？
- a. リソースのうちで使用頻度の高いものがありますか？その場合、設計時にこの状態を予測しましたか？予測しなかった場合は、使用頻度の高さをトランザクションの使用頻度の高さによって説明できますか？
 - b. 使用頻度の高さは特定のアプリケーションに関連しますか？その場合は、増大またはピーク期間が計画された形跡がありますか？
 - c. ブラウズ・トランザクションが、予想された数より多くの要求を出していますか？言い換えれば、トランザクションによって出されたブラウズ要求の数が、ユーザーが出すと予想した数を上回っていますか？
 - d. CICS CSAC トランザクション (DFHACP 異常条件プログラムが提供する) は頻繁に使用されていますか？その場合、これは無効トランザクション ID が入力されているために発生しているのですか？

例えば、IBM 3270 端末において、トランザクション ID が小文字で入力されたが、大文字への入力の自動変換が指定されていない場合は、エラー信号が送られます。

対応する CSAC の数はないが、DFHACP プログラムの使用頻度が高い場合は、正しいオペレーター・サインオンなしにトランザクションが入力されていることを示す可能性があります。この状態は、一部の端末オペレーターがシステムの使用についてさらにトレーニングを受ける必要があることを示している場合があります。

加えて、以下のような CICS 統計の特定の項目を定期的に検討します。

- MAXTASK 限度に達した回数 (トランザクション・マネージャー統計)
- タスクのピーク数 (トランザクション・クラス統計)
- クッションの解放回数 (ストレージ・マネージャー統計)
- 記憶保護違反数 (ストレージ・マネージャー統計)
- 通知された RPL の最大数 (z/OS Communications Server 統計)
- ストレージ不足の数 (ストレージ・マネージャー統計)
- ストリングにおける合計待機数 (ファイル制御統計)
- DFHSHUNT ログ・ストリームの使用
- 補助記憶域を使い果たした回数 (一時記憶域統計)
- バッファ待機数 (一時記憶域統計)
- ストリング待機発生数 (一時記憶域統計)
- NOSPACE の発生回数 (一時データ・グローバル統計)
- 区画内バッファ待機数 (一時データ・グローバル統計)
- 区画内ストリング待機数 (一時データ・グローバル統計)
- MAXSOCKETS 限度に達した回数 (TCP/IP 統計)
- プール・スレッドの待機数 (Db2 接続統計)

システム障害およびその継続時間の影響および理由を検討します。障害が連続して発生する場合は、共通する原因がある可能性があります。

CICS パフォーマンス分析の手法

CICS のパフォーマンスを分析するには、いくつかの手法を使用できます。

パフォーマンス分析には、主に次の 4 つの使用目的があります。

- 現在のところはパフォーマンス上の問題はないが、パフォーマンスのさらなる向上を目指してシステムを調整する。
- 個々のスタンドアロンのトランザクションを、それらのトランザクションのドキュメンテーションの一部として、および将来トランザクションの振る舞いが変化し始めたときに比較できるように、特徴付けて調整する。
- システムが以前に確認した目標からずれるようになったので、どこに問題があり、なぜそれが起こっているのかを正確に知る。オンライン・システムをインストールしたときは効率的に稼働しているように思えたが、システム使用の特性が変化して、システムの実行がそれほど効率的ではなくなっている可能性がある。この非効率性は、通常、各種制御項目を調整することによって訂正することができます。どのような新規システムでも、それを稼働し続けた場合、通常は調整を行う必要が出てきます。
- システムがパフォーマンス目標を持っているか、または持っていない場合でも、深刻なパフォーマンス上の問題を抱えているように見える。

現在のパフォーマンスが要求を満たしていない場合は、システムを調整してください。システムを調整するには、以下の作業を実行する必要があります。

1. システムにおける主要な制約を識別する。
2. どのような変更であれば、他のリソースを犠牲にして、制約を軽減することができるのかを理解する。調整は、通常、あるリソースと別のリソースの間のトレードオフです。

3. より集中的に使用される可能性のあるリソースを判別する。
4. パラメーターを調整して、制約付きのリソースを軽減する。
5. 以下の基準の観点から、調整後のシステムのパフォーマンスを検討する。
 - 既存のパフォーマンス目標
 - これまでの進展
 - これまでの調整作業
6. パフォーマンスが受け入れ可能である場合はこの時点で停止する。そうでなければ、以下の処置のいずれか1つを行う。
 - 調整を続ける
 - 適切なハードウェア能力を追加する
 - システム・パフォーマンス目標を下げる。

調整タスクは、以下のようなフローチャート形式で表すことができます。

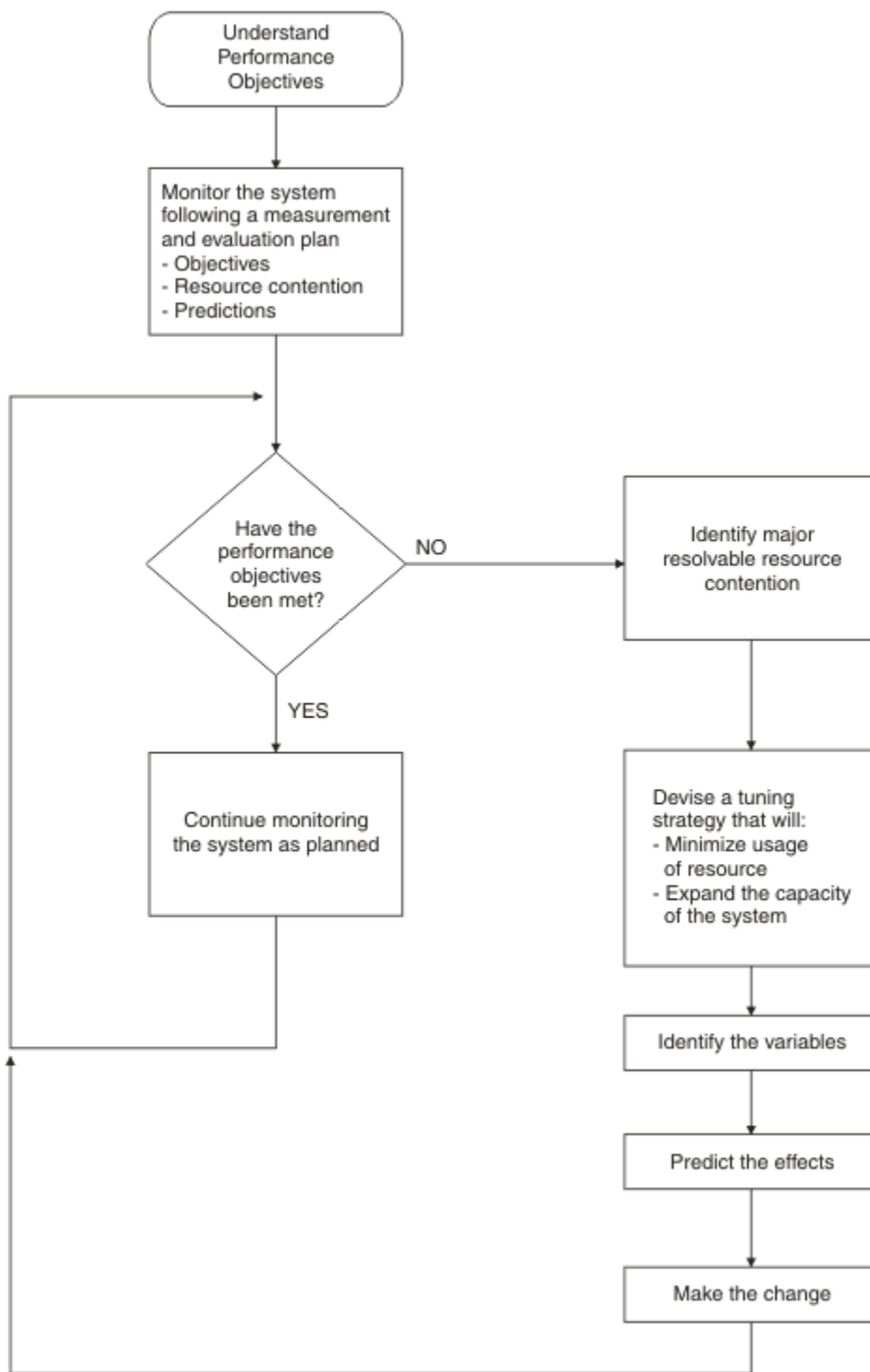


図 1. パフォーマンス調整のための規則を示したフローチャート

パフォーマンスの分析時の調査項目

特定の CICS 上の問題があると判断する前に、まず最初に必ずシステム全体をチェックしてください。プロセッサ使用量の合計、直接アクセス・ストレージ・デバイスのアクティビティ、およびページングをチェックしてください。パフォーマンスの低下は、アプリケーションの規模の増加、およびそれに対応したハードウェア・リソースの増加が釣り合っていないことによる場合がよくあります。その場合は、ハードウェア・リソースの問題を最初に解決してください。それでもまだ複数領域のための計画に従う必要がある場合があります。

以下に示した、少なくとも 3 つのレベルの情報が必要です。

1. CICS: 特定のリソースが過負荷であることを示唆している例外、キュー、およびその他の徴候がないか、CICS インターバルまたは 1 日の終わり統計を調べます。レポート期間が短いと、問題を分離できます。ソフトウェア・リソースとハードウェア・リソースを考慮してください。例えば、VSAM スtring またはデータベース・スレッド、ファイル、および TP 回線の使用率を検討します。コンソールおよび一時データ宛先 (CSMT や CSTL など) に送信される実行時のメッセージをチェックして、アプリケーション問題やネットワーク・エラーが引き続き発生していないかを確認します。

CICS Explorer や RMF などのツールを使用して、オンライン・システムをモニターし、パフォーマンスが低下している期間に関連するアクティビティを確認します。CICS Performance Analyzer や IBM Z Decision Support などのツールを使用して、CICS モニター機能の履歴を収集して分析し、パフォーマンスおよびリソースの使用例外および傾向を確認します。例えば、入出力をほとんどまたはまったく行わない、プロセッサ集中のトランザクションに注意してください。これらのトランザクションはプロセッサを独占することがあり、それにより、通常はバランスの取れたアクティビティ・プロファイルを持つ他のトランザクションで応答が不安定になることがあります。このようなトランザクションは、別の CICS 領域に分離する候補となる可能性があります。
2. MVS: SMF データを使用して、CICS パフォーマンスが低い期間と MVS システム内の他の同時アクティビティとの間の関連性を見つけます。過負荷状態にある装置およびパスを特定するには、RMF データを使用します。CICS 領域のページング率をモニターし、構成をサポートするのに十分な実記憶域があることを確認してください。
3. ネットワーク: システムで費やされる応答時間の比率は、ネットワークでの伝送遅延およびキューイングと比較した場合、小さなものです。Tivoli NetView for z/OS などのツールを使用して、ネットワーク内の問題および過負荷を確認します。自動ツールがない場合は、パフォーマンスが低下した、というユーザーの主観的意見に依存することになります。

CICS では、パフォーマンス上の問題は、応答時間が遅いか、またはリソースの使用が予想を上回り、その原因を説明できないかのいずれかです。一般的には、システムを少し詳細に調べて、なぜシステムでのタスクの処理に時間がかかり、なぜ特定のリソースが集中的に使用されているかを確認する必要があります。CICS の詳細な振る舞いを調べる最もよい方法は、CICS 補助トレースを使用することです。補助トレースは最も優れた方法ではありますが、これに切り替えると、その使用中は、既存のローパフォーマンスをさらに低下させてしまう可能性があることに注意してください。方法としては、まずタスク・アクティビティの状況を取得してタスク・トレースのみをリストし、それから特定のアクティビティ (特定のタスクまたは限定された時間間隔) にのみフォーカスします。例えば、応答時間の問題の場合、速度が遅いと判断されたあるタスクの詳細なトレースを調べることができます。さまざまな原因が考えられます。例えば、タスクが実行しようとしている処理がそのシステムにとって多すぎる場合です。システムの実記憶が制約されているか、または特定の機能に対する競合があるために多数の CICS タスクが待機している可能性があります。

パフォーマンス分析に役立つ情報源

統計機能や CICS モニター機能など、どのパフォーマンス測定ツールも問題の診断に役立つ可能性があります。各パフォーマンス・ツールは、モニター、単一トランザクション測定、および問題判別などそれぞれの目的に、ある程度は使用可能である、と考えてください。CICS 統計により、特定のリソースの使用頻度が高いことが明らかになることがあります。例えば、主記憶域内の一時記憶域の割り振りが大きいこと、タスク当たりのストレージ制御要求数が多いこと (おそらく 50 または 100)、あるいはプログラムの使用回数が多いこと (これは、プログラム制御 LINK が集中的に使用されていることを意味します) が見つかる場合があります。

統計と CICS モニターの両方から、CICS の実行中に例外的な VSAM 共用リソース状況が発生したことがわかる場合があります。統計は、String 待ち、VSAM 共用リソース待ち、GETMAIN 要求でのストレージ待ち、その他の待機を表示できます。待機もまた、CICS モニター機能例外クラス・レコードを生成します。

これらの状態は CICS 補助トレースでも明らかになりますが、十分に明白ではない場合があります。その他の情報源はトレース・データの調査を方向を示すのに役立ちます。また、CICS の停止の調査から有益なデータを得ることができる可能性もあります。停止が連続して発生する場合は、停止間の共通のつながりを調査します。

The QR TCB CPU Dispatch Ratio

TCB CPU ディスパッチ率は、累積ディスパッチ時間に対する累算 CPU 時間の割合を、パーセンテージで表したものです。CICS 環境では、この比率は QR TCB でのみ役に立ち、他の TCB では意味がありません。QR TCB CPU ディスパッチ率は、CICS ディスパッチャーによって要求されるプロセッサ・リソースの量と比較したときの、オペレーティング・システムおよびハードウェアによって QR TCB に割り当てられているプロセッサ・リソースの量を示します。

ある特定のインターバルでの高い比率は、CICS が QR TCB にタスクをディスパッチしたときに、その CICS タスクが完了するまで、プロセッサ・リソースがほとんど中断されずにオペレーティング・システムやハードウェアによって使用可能になっていたことを示します。この場合、CPU 時間は全体的な経過時間 (CICS ディスパッチ時間) と密接な相関関係にあります。

比率が低い場合は、CICS がプロセッサ・リソースを要求しても、オペレーティング・システム、ハードウェア、またはその両方の組み合わせによる物理プロセッサの待機が原因で、高頻度または長時間の遅延が生じたことを示しています。この場合、CPU 時間は全体的な経過時間よりかなり小さくなります。

QR TCB CPU ディスパッチ率は、以下のようないくつかの方法で報告されます。

- システム・メモリー・ダンプの使用:

CICS IPCS システム・メモリー・ダンプ・フォーマッターの DS キーワードは、Dispatcher Statistics - CICS TCB Mode Statistics セクションに比率を示します。

- CICS 統計 SMF レコードの使用:

CICS 統計ユーティリティー・プログラムの DFHSTUP は、Dispatcher Statistics セクションに比率を示します。

- サンプルの統計プログラム DFHOSTAT の使用:

サンプル・プログラムは、Dispatcher TCB Modes セクションに比率を示します。

- CICS メッセージ DFHDS0102 の使用:

DFHDS0102 メッセージは、QR TCB CPU ディスパッチ率を定期的に報告するので、CPU リソース不足の可能性を示す指標として使用できます。DFHDS0102 メッセージのデフォルト・インターバルは 5 分間ですが、システム初期設定パラメーター **INITPARM=(DFHQRCPU='nn')** を指定してインターバルを設定することもできます。**nn** は 01 から 59 の範囲の分数です。

QR TCB CPU ディスパッチ率は、CICS ディスパッチャー統計を使用して計算されます。CICS は、QR TCB がディスパッチされるたびに TCB によって使用される CPU 時間を集計します。システム・メモリー・ダンプ、CICS 統計の SMF レコード、またはサンプル統計プログラムを使用する場合、この比率は、統計が最後にリセットされてから集計された CPU 時間とディスパッチ時間を使用して計算されます。DFHDS0102 を使用する場合、この比率は、最後にこのメッセージが発行された以降での QR TCB の平均使用率となります。

ベスト・プラクティスとして、一連の計測を収集してお客様の環境での QR TCB CPU ディスパッチ率の通常範囲を判別し、それらの計測を使用して、この比率が通常動作からの変化を示しているかどうかを識別してください。

比率が低い一般的な理由

ビジー・システムでは、CICS 作業がプロセッサ・リソースを待機することは正常な動作なので、ディスパッチ率が 100% 未満であっても許容できます。この比率が低い値になると、CICS 領域でパフォーマンス上の問題 (トランザクション応答時間の悪化など) が発生する可能性があります。QR TCB CPU ディスパッチ率での低い値とは、通常 70 % 未満のことです。以下のいずれかの条件が発生すると、比率が低くなる可能性があります。

- LPAR がビジー状態である。CICS 領域は他のアドレス・スペースとの間で CPU の競合があるので、オペレーティング・システムは要求されたときにプロセッサ・リソースを割り振ることができません。
- LPAR の相応の割り当て量に達したか、その上限が設定されている。オペレーティング・システムが CICS QR TCB を論理プロセッサにディスパッチしましたが、ハードウェアは論理プロセッサを物理プロセッサにディスパッチできません。

- CICS が LPAR 内の制限されたリソースに依存している。LPAR は完全には使用できない可能性があります。オペレーティング・システムの制御により、CICS 領域で使用可能なプロセッサ・リソースの量が制限されています。
- アプリケーション・コードが非 CICS API 要求 (例えば MVS マクロ要求) を発行している。そのため、要求が完了するまで QR TCB がブロックされます。
- 過剰なシステム・ページングが行われている。

以下の条件によっても比率が低くなる場合があります。ただし、これらは正常な状態と見なされるので、詳しい調査は必要ありません。

- CICS システムの初期設定中である。CICS に制御が渡された直後に低い比率が検出されますが、これは正常な状態と見なされます。CICS は初期化中に多数の MVS システム・サービスを使用し、そのすべては QR TCB によって処理されるためです。
- 領域が端末専有領域であり、システム初期設定パラメーター **HPO** が YES に設定されている。この場合、VTAM® は到着する作業をサブタスク化して SRB に渡し、使用される CPU は別の場所に作業をルーティングするためのものだけになります。
- 領域内の非スレッド・セーフ・アプリケーションが VSAM RLS ファイルにアクセスすると、VSAM は SRB でファイル・アクセス要求を完了し、消費される CPU は QR TCB ディスパッチャー統計によって集計されません。詳しくは、[グループ DFHFILE 内のパフォーマンス・データの RLSCPUT フィールドの説明](#)を参照してください。

測定および評価計画の確立

一部のインストールの場合、測定および評価計画が適切であることがあります。測定および評価計画は、システムのパフォーマンスの測定、評価、およびモニターのための構造化された方法です。

測定および評価計画をセットアップするには、以下のステップを実行します。

1. 計画を立てる
2. 計画を検討する
3. 計画を実施する
4. 必要に応じて計画を修正してアップグレードする

計画を使用するには、以下の主要な活動を実行します。

- 以下を判別するために、定期的に情報を収集する。
 - 目標が満たされたかどうか
 - トランザクションのアクティビティー
 - リソースの使用率
- 情報を要約し、分析する。このアクティビティーには、以下を行います。
 - 指定された頻度で、ボリュームおよび平均をグラフにプロットする
 - 指定された頻度で、リソースの使用率をグラフにプロットする
 - 毎日のログに、異常な状態を記録する
 - ログおよびグラフを毎週検討する
- 目標が満たされなかった場合は、変更を行う、または推奨する。
- 過去、現在、および計画のトランザクション・アクティビティーおよびリソース使用率を関連付けて、その目標を引き続き満たしているかどうか、およびリソースが効率的な容量を超えて使用されていないかどうかを判断する。
- 非公式レポート、作成済みレポート、および毎月のミーティングで、関係者に情報を通知し続ける。

一般的な測定および評価計画には、記録する頻度および測定ツールに関する記述と共に、以下の項目が目標として含まれます。

- 部門別のボリュームおよび応答時間
- ネットワーク・アクティビティー:

- トランザクションの合計
- 毎秒のタスク
- トランザクション別の合計
- 毎時間のトランザクション・ボリューム (合計、およびトランザクション別)
- リソースの使用率の例:
 - DSA 使用率
 - CICS でのプロセッサ使用率
 - CICS およびシステムのページング率
 - チャンネル使用率
 - 装置の使用率
 - データ・セットの使用率
 - 回線使用率
- 異常状態:
 - ネットワーク問題
 - アプリケーション問題
 - オペレーターの問題
 - トランザクション・クラスへのエントリーのためのトランザクション・カウント
 - SOS の発生
 - 記憶保護違反
 - (通信ネットワークとは関係のない) 装置の問題
 - システム停止
 - CICS 停止時間

システムのパフォーマンスの査定

システムのパフォーマンスを判別するには、プロセッサ使用量、入出力比率、端末メッセージまたはデータ・セット・レコード・ブロック・サイズ、ページング率、およびエラー率のパフォーマンス測定が役立つことがあります。

プロセッサ使用量

この項目は、プロセッサのアクティブ度を反映しています。測定的主要な対象は中央処理装置ですが、37X5 通信コントローラーおよび端末制御装置も、集中的に使用された場合は、応答時間を増加させます。

入出力比率

これらの比率は、特定の期間に渡るディスク装置またはデータ・セットへのアクセス量を測定するものです。ここでも、受け入れ可能な比率は、ハードウェアの速度および応答時間要件によって変動します。

端末メッセージまたはデータ・セット・レコード・ブロック・サイズ

これらの要因は、入出力比率と組み合わせて、ネットワークまたは DASD サブシステムに対する現在の負荷に関する情報を提供します。

内部の仮想記憶制限の標識

これらの標識は、ストレージまたはバッファ拡張回数、システム・メッセージ、およびシステムの停止によるプログラムの異常終了など、ソフトウェア・コンポーネントによって変動します。CICS では、非常駐プログラムに対するプログラム取り出し、およびシステムのストレージ不足やストレス・メッセージは、この状態を反映しています。

ページング率

CICS は、実記憶域不足に敏感であり、ページング率はこの不足を反映しています。DASD への、受け入れ可能なページング率は、DASD の速度および応答時間の基準によって変動します。

エラー率

エラーはオンライン・システムのどのポイントでも発生します。エラーがリカバリー可能な場合には、そのエラーが気付かれずにいることがあります。このような場合は、そのエラーが発生しているリソースにさらに負荷をかけます。

システム状態およびアプリケーション状態の両方を調べてください。

システム状態

以下の状態を知ることにより、システム全体のパフォーマンスを評価するのに役立ちます。

- システム・トランザクション比率 (平均およびピーク)
- 内部応答時間および端末応答時間 (可能であれば、トランザクション比率と比較)
- 平均およびピークのトランザクション比率での作業セット
- 単位時間当たりのディスク・アクセスの平均数 (合計、チャンネルごと、および装置ごと)
- トランザクション比率と比較した、プロセッサの使用量
- トランザクション比率および実記憶と比較した、毎秒のページ不在の数
- (正味および実際の) 通信回線使用量
- アクティブな CICS タスク数の平均
- 停止の数およびその継続時間

アプリケーションの状態

アプリケーションの状態は、個々のトランザクション・タイプおよび全システムの両方について測定され、個々のアプリケーション・プログラムの振る舞いの推定値を提供します。メイン・トランザクションごとのデータとシステム全体の平均値を収集します。これには、次のデータが含まれます。

- トランザクション当たりのプログラム呼び出し回数
- CICS ストレージの GETMAIN および FREEMAIN 要求 (数および量)
- アプリケーション・プログラムおよびトランザクションの使用量
- ファイル制御 (データ・セット、要求のタイプ)
- 端末管理 (端末、入出力数)
- トランザクション・ルーティング (ソース、ターゲット)
- 機能シップ (ソース、ターゲット)
- その他の CICS 要求

パフォーマンス分析の方式

パフォーマンス分析のために 2 つの方式を使用できます。完全実動ロード下でのシステムの測定 (完全ロード測定)。これは、システム・ロードが高い条件下でのみ測定可能なすべての情報を取得します。単一アプリケーションのトランザクションの測定 (単一トランザクション測定)。この間、システムは他のいかなるアクティビティも実行しないようにする必要があります。

システムにはさまざまな問題があることが考えられるため、システムの動作を調査するために使用するオプションを推奨することはできません。問題の範囲が不確かな場合は、常に両方の方式を使用します。

しきい値を超えて、システムが限界負荷に近づくと、パフォーマンスが急激に低下することがよくあります。システムが完全にロードされている場合にのみ、さまざまな標識 (例えば、CICS のページング、ストレージ不足状態など) を確認できるため、通常は完全ロード測定のための計画を立てる必要があります。

IBM Redbooks 資料「[ABC's of z/OS System Programming, Volume 11](#)」には、パフォーマンス分析方式に関する詳細が記載されています。

パフォーマンス分析: 完全ロード測定

完全ロード測定を実行すると、システム内の潜在的な問題が浮かび上がります。完全ロード測定は、実動経験から、負荷がピークに達したときに測定を行うことが重要です。

多くのインストールでは、負荷がピークを迎えるのは、朝の約 1 時間の間、および昼の約 1 時間の間です。CICS 統計および各種パフォーマンス・ツールは、完全ロード測定のための貴重な情報を提供します。これらのツールによる全般的な結果の他に、CICS 補助トレースまたは RMF を約 1 分間アクティブにしておくことが役立ちます。

CICS 補助トレース

CICS 補助トレースは、完全負荷の下で発生する状況を見つけるのに使用することができます。例えば、すべての ENQUEUE 操作を、アプリケーション・プログラム内で即時に優先できない場合には、発行側のタスクが中断します。この状態が頻繁に発生する場合は、マスター・トランザクションを使用してシステムを制御しようとしても効果がありません。

トレースは、重いオーバーヘッドです。このオーバーヘッドを最小にするには、トレース選択オプションを使用します。

RMF

RMF 測定を行う場合は、バッチ・アクティビティーを行わないことをお勧めします。

完全ロード測定の場合、システム・アクティビティー 報告書および DASD アクティビティー 報告書が重要です。

完全ロード測定の最も重要な値は、以下のとおりです。

- プロセッサ使用量
- チャンネルおよびディスク使用量
- ディスク装置使用量
- プロセッサと、チャンネルおよびディスク・アクティビティーのオーバーラップ
- ページング
- 入出力の開始操作のカウンタおよび平均の入出力開始時間
- 応答時間
- トランザクション比率

プロセッサの負荷が 100% に近づくと、スループットが停滞し、応答時間が急激に上昇します。

相互作用する要因が多すぎるため、パフォーマンスにとって深刻なダメージがない場合に達成可能なシステム・ページング率を予測することは困難です。報告されたページング率を注意して見てください。短期間続く重大なページングが発生すると、応答時間の急激な上昇につながることに注意してください。

入出力開始操作のカウンタおよびそれぞれの平均の長さに関する以外に、システムが 1 台の装置だけで待機していないかを調べます。例えば、ディスクの場合、頻繁にアクセスされる複数のデータ・セットが、1 台のディスクにあり、アクセスが相互に干渉しあうということが発生することがあります。それぞれの場合において、データ・セットを再編成して、特定の装置で発生するシステム待ちを最小にできないかを調べます。

RMF DASD アクティビティー 報告書には、以下の情報が含まれます。

- すべてのディスク情報の要約
- システム番号別および領域別の、ディスク当たりの明細
- シーク・アーム移動の、ディスク当たりの配分
- アーム移動を伴う、または伴わないアクセスの、ディスク当たりの配分

DASD 制御装置の競合を表示するには、RMF モニター 1 で IOQ(DASD) オプションを使用します。

アーム移動を伴う場合と伴わない場合のアクセスの関係をチェックした後、例えば、定期的にアクセスされる頻度が多いデータ・セットを、別々のディスクに移動します。

比較一覧表

調整の変更を行う前と後に、比較一覧表を使用して、システム・パフォーマンスの重要な側面を評価することを検討してください。推奨する一覧表を以下に示します。

表 1. 比較一覧表					
測定項目		実行 A	実行 B	実行 C	実行 D
DL/I トランザクション	番号				
DL/I トランザクション	応答				
VSAM トランザクション	番号				
VSAM トランザクション	応答				
応答時間	DL/I				
応答時間	VSAM				
最も集中的に使用される トランザクション	番号				
最も集中的に使用される トランザクション	応答				
平均的に使用されるトラ ンザクション	番号				
平均的に使用されるトラ ンザクション	応答				
ページング率	システム				
ページング率	CICS				
DSA 仮想記憶	最大				
DSA 仮想記憶	平均				
タスク	ピーク				
タスク	最大時				
最も集中的に使用される DASD	応答				
最も集中的に使用される DASD	使用率				
平均的に使用される DASD	応答				
平均的に使用される DASD	使用率				
CPU 使用率					

このタイプの比較一覧表では、システムのピーク時に約 20 分間、同時に実行された TPNS、RMF、および CICS の間隔統計を使用する必要があります。また、以下の項目を識別する必要があります。

- DL/I データベースにアクセスする、端末向け DL/I トランザクションから選択する代表的なトランザクション
- VSAM ファイルを処理する端末向けトランザクションから選択する代表的なトランザクション
- 最も集中的に使用されるトランザクション
- 区画内一時データ宛先にデータを書き込む、平均的に使用される 2 つの非端末向けトランザクション
- システムで最も集中的に使用されるボリューム
- システムでの代表的な、平均的に使用されるボリューム

調整変更の前後で実行する CICS の実行ごとに比較一覧表を完成するには、以下のソースから数値を取得します。

- **DL/I トランザクション:** DL/I データベースにアクセスする端末向け DL/I トランザクションの選択を示します。
- **VSAM トランザクション:** VSAM ファイルを処理する端末向け トランザクションの選択を示します。
- **応答時間:** 外部応答時間は、TPNS 端末応答時間分析レポートから取得できます。内部応答時間は、RMF から取得できます。「DL/I」副見出しは平均の応答時間で、以前に選択した端末向け DL/I トランザクションの第 99 百分位数で計算されます。「VSAM」副見出しは平均の応答時間で、以前に選択した端末向け VSAM トランザクションの第 99 百分位数で計算されます。
- **ページング率 (システム):** RMF ページング・アクティビティ・レポートには、システムの非 VIO 非スワップ・ページインの合計に示された数値を、システムの非 VIO 非スワップ・ページアウトの合計に示された数値に加算した値が表示されます。この数値は、全システムの、毎秒のページング率の合計です。
- **タスク:** トランザクション・マネージャー統計 (CICS インターバルの一部、1 日の終わり、および要求された統計)。「ピーク」副見出しは、統計の「Peak Number of Tasks (ピーク・タスク数)」に示されている数値です。「最大時」副見出しは、統計の「Number of Times at Max. Task (最大タスク回数)」に示されている数値です。
- **最も集中的に使用される DASD:** RMF 直接アクセス装置のアクティビティ・報告書。これは、システムで最も集中的に使用されるボリュームに関連しています。「応答」副見出しは、選択したボリュームを表す「Avg. Resp. Time (平均応答時間)」列に示される数値です。「使用率」副見出しは、そのボリュームの「% Dev. Util. (% 装置使用率)」列に示される数値です。
- **平均的に使用される DASD:** RMF 直接アクセス装置のアクティビティ・報告書。これは、システム内の代表的な平均的使用量のボリュームに関連しています。「応答」副見出しは、選択したボリュームを表す「Avg. Resp. Time (平均応答時間)」列に示される数値です。「使用率」副見出しは、そのボリュームの「% Dev. Util. (% 装置使用率)」列に示される数値です。
- **プロセッサ使用率:** RMF プロセッサのアクティビティ・報告書。

この一覧表が最も役に立つのは、CICS システムの調整中に、変更の前後のパフォーマンスを比較する場合です。

パフォーマンス分析: 単一トランザクション測定

完全ロード測定を使用すると、トランザクション当たりのシステムの平均ロード時間を評価できます。ただし、このタイプの測定からは、単一トランザクションおよびその考えられるシステムの超過ロードの振る舞いに関する情報は得られません。例えば、9 つの異なるトランザクション・タイプがそれぞれ 5 つの入出力開始 (SIO) を発行し、10 番目のトランザクション・タイプが 55 個の SIO を発行した場合、トランザクション・タイプ当たり、平均 10 個の SIO になります。この状態では、複数のトランザクションを同時に開始しても問題はないはずです。ただし、10 番目のトランザクション・タイプのトランザクション率が増加すると、全体のパフォーマンスが低下する可能性があります。このタイプの問題を調べるには、単一トランザクション測定を実行します。

時々、既存の端末では応答時間が良好であるにもかかわらず、少数の端末を追加すると、パフォーマンスが容認できないほど低下することがあります。この場合、既存の端末にパフォーマンス上の問題が存在している可能性があり、負荷をかけることでその問題が強調されただけです。

このタイプの問題を調べるには、完全ロード測定と単一トランザクション測定を実行します。単一トランザクション測定では、バッチ領域が稼働していないときにこの測定を実行する必要があり、CICS にはテスト画面以外のいかなるアクティビティも存在しないようにする必要があります。リモート端末のポーリングを一時停止します。

実動システムまたは最終のテスト・システムで使用される既存のトランザクションをそれぞれ測定します。データ値を変えて、各トランザクションを 2 回から 3 回テストして、特に望ましくないデータの組み合わせを除外します。トランザクションの順序、および後の分析または解釈のための前提条件として、テストごとに入力された値を文書化します。

各単一トランザクションのテスト間には、数秒の休止をはさみ、トレースを読みやすくする必要があります。テストには、実動データベースまたはデータ・セットのコピーを使用します。100 個のレコードを含むテスト・データ・セットは、100,000 000 個のレコードを含む実動データ・セットと比較した場合、異なる動作を示すことがよくあるからです。

特に、多くのセグメントやレコードがデータベースまたはデータ・セットに追加された場合、データ・セットの状態は、パフォーマンス低下の原因になることがあります。データベースまたはデータ・セットは、短時間の間だけこの状態になっているので、再編成後は直接測定を行わないでください。測定で異常に多数のディスク・アクセスが明らかになった場合は、データを再編成し、さらに測定を実行して、データの再編成の効果を評価します。

1つだけの端末を使用する単一トランザクションの測定は、おそらく 40 または 50 の端末が使用されているときに発生する可能性のあるパフォーマンスの低下を明らかにするには、効率的なツールとはいえない場合があります。ただし実際的な経験からは、通常単一トランザクション測定が、妥当なコストで完全負荷の下でのパフォーマンスの低下を明らかにし、それを修正するための唯一の手段であることが分かっています。

これらの理由で、単一トランザクション測定はトランザクションの最終テスト段階で実行するのが理想的です。

- トランザクションの動作のエラーは、実動システムをロードせずに、実動を開始する前にすべて明らかにし、修正することができます。
- アプリケーションは測定段階で文書化されるので、その後の変更の効果を確認するのに役立ちます。

CICS 補助トレース

補助トレースは、CICS の標準機構であり、トランザクション・フローの概要を示してくれるので、それらを素早くかつ効率的に分析することができます。このトレースから、指定されたアプリケーションが予想通りに実行されているかどうかを確認できます。

多数のトランザクションを分析する必要がある場合は、最初のパスで、動作が予想に適合していないトランザクションを選択できます。

すべてのトランザクションが予想よりはるかに長く続いている場合は、アプリケーション・プログラミングまたはシステム実装に、システム全体に関わるエラーがある可能性があります。トランザクションをいくつか分析するだけでエラーを判別することができます。

ほんの少数のトランザクションが残っている場合、次にこれらのトランザクションを分析します。これらのトランザクションがパフォーマンス上の問題の大部分を引き起こしている可能性が高いためです。

パフォーマンス測定ツール

パフォーマンスを測定し、システム内の制約が発生する可能性のある場所を把握するために使用できるいくつかのツールがあります。

実動システムのパフォーマンスは、CPU、実記憶域、ISC リンク、カップリング・ファシリティ、およびネットワークなどのリソースの使用率によって異なります。さまざまなプログラムを作成して、これらすべてのリソースをモニターできます。これらのプログラムの多くは、現在、CICS または IMS などの IBM 製品の一部として提供されているか、別個の製品として提供されています。これらのトピックでは、実動システムのさまざまなコンポーネントに関するパフォーマンス情報を提供できる製品のいくつかを説明します。

これらのトピックの製品リストは、パフォーマンス・モニター・ツールを網羅した要約ではありませんが、これらのソースから提供されるデータは、大量の情報になります。このデータすべてをモニターするには、広範囲の作業が必要になります。また、提供される情報の小さなサブセットだけでも、制約を識別し、必要なチューニング処置を判別するには重要であるため、使用する特定の CICS システムのこの特定のサブセットを識別する必要があります。

また、次の 2 つの異なるツールが存在することを考慮してください。

1. 目標を満たしているかどうかを直接測定するツール
2. 目標を満たしていない内部の理由を調べる追加ツール

エンド・ユーザーの応答時間の目標を満たしているかどうかを直接測定するツールはありません。CICS 内のタスクの存続時間を同等のものと考えることができます。つまり、それは通常、応答時間に関連します。

遅い応答時間は、通常、CICS 内の長い存続時間に関連しますが、この相関は応答時間には他の貢献要因があるため、正確なものではありません。

目標を測定できるツールが必要なことは言うまでもありません。しかし、実際の目標を測定することが困難であるため、場合によっては、それを直接測定するのではなく、タスクの存続時間など、パフォーマンス目標に影響するいくつかの内部機能を調べるツールを選択することができます。

システムの経験を積むと、その特定のシステムで最も重要な特定の出来事、つまり、例外報告の基礎として使用できる出来事をよく理解できるようになります。したがって、重要なデータをモニターする方法として、チューニング・プロセスにとって本質的でないデータをフィルタリングする例外報告手順を準備する方法があります。この方法には、通常のパフォーマンス・データをフィルタリングする一方、例外を識別し報告できるようにするための制約を識別するパフォーマンス基準の標準を設定する作業が含まれます。これらの標準は、個別のシステム要件およびサービス・レベルの合意によって異なります。

ご使用のシステムの動作を完全に理解し、どこをチューニングすれば、全体のパフォーマンスが一番向上するのかを判別できるまでには、しばしば、大量のデータを収集する必要があります。チューニングを成功させるためには、分析ツールとそれが提供するデータに慣れることが基本です。

ただし、すべてのモニター・ツールは、処理時間を必要とすることに注意してください。典型的なコストは、CICS モニター機能 (パフォーマンス・クラス) の場合は 5% の追加プロセッサ・サイクルであり、例外クラスの場合は最大 1% です。CICS トレース機能のオーバーヘッドは、使用するワークロードによって大きく異なります。オーバーヘッドは 25% を超過する場合があります。

したがって、一般に、次のツールを以下に示す優先順位で使用することをお勧めします。

1. CICS 統計
2. CICS モニター・データ
3. CICS 内部および補助トレース

システムの調整

制約を具体的に識別すると、調整が必要なシステム・リソースがわかります。システム調整の 3 つの主要なステップは、容認できる調整のトレードオフの判別、システムの調整の変更、および調整の結果の検討です。

受け入れ可能な調整トレードオフの判別

調整のこつは、大まかに言えば、制約を検出して削除することです。ほとんどのシステムでは、パフォーマンスは単一の制約によって制限されます。ただし、パフォーマンスの改善中に制約を削除すると、必然的に別の制約が作成されるため、通常は一連の制約を削除する必要があります。一般に調整を行うと特定のリソースの負荷が軽減される代わりに、別のリソースの負荷が増大するため、特定の制約を解放すると必ず別の制約が作成されます。

システムは常に制約されます。制約を削除することはできません。ユーザーが実行できるのは、最も満足のいく制約を選択することだけです。システムの追加負荷を受け入れても、制約が厳しくならず、性能低下を引き起こさないリソースを検討してください。

システムの調整変更

調整プロセスの次のステップは、パフォーマンスを改善するために実際にシステムを変更することです。システムを調整する場合は、次の点に注意する必要があります。

- 調整とは、システムのリソース割り振りおよび可用性を少し変更して、応答時間を大幅に改善する技術です。
- 調整が常に有効であるとはかぎりません。システム応答が長すぎて、すべてのシステム・リソースの使用頻度が少ない場合、CICS 応答時間はほんのわずかしかが変更されません。(同じことが不正なリソースを調整した場合も言えます)。さらに、制約リソース (回線容量など) が最大限に使用されている場合は、容量を拡張するか、またはアプリケーションを再設計する (回線容量の場合はデータ転送量を削減する) 以外に解決策はありません。
- 調整のためだけの調整は行わないでください。識別された制約を軽減するように調整してください。パフォーマンス問題の主要原因でないリソースを調整すると、主要制約を軽減しないかぎり応答時間に効果

はまったく、あるいはほとんどなく、以降の調整作業がより困難になることがあります。大幅に改善する余地があるとすれば、それは応答時間の主要要因であるリソースのパフォーマンスを向上させることです。

- 一般に、主要制約 (特に応答時間に対する効果が大きな制約) を最初に調整します。最大の効果を持つ項目が最初に実行されるように、調整アクションを配置します。通常は、調整変更を 1 回行くと、パフォーマンス低下の原因となるパフォーマンス問題を解決することができます。その他のアクションは不要な場合があります。さらに、主要な方法でパフォーマンスを改善すると、ユーザーの不満が軽減され、より完全な方法で作業できるようになることがあります。ここでは、80/20 の規則が適用されます。システムを少し変更し、パフォーマンス問題の主要原因を解決するだけで、通常は、応答時間のうち、改善可能な量の大部分が改善されます。
- 調整変更は一度に 1 つずつ行ってください。一度に 2 つの変更を行うと、これらの効果が反対の方向に作用し、どちらが大きな効果があったかを判断しにくくなる場合があります。
- 割り振りまたは定義は少しずつ行ってください。例えば、システムの常駐プログラム数を削減する場合は、システム内のすべてのプログラムを RES=YES から RES=NO に一度に変更しないでください。このようにすると、フラグメント化によってストレージ使用量が増加したり、上位プログラムのロード・アクティビティによってプロセッサ使用量が増加して、応答時間が予想外に延びることがあります。一度に変更するプログラムが少ない場合は、使用頻度の低いプログラムから開始してください。このようにすると、全体的な結果を把握しやすくなります。

バッファーストリング設定、その他のデータ・セット・オペランド、トランザクションおよびプログラム・オペランド、およびオペランドをそれぞれ個別に指定できるすべてのリソースに対して、同じ規則が適用されます。同じ理由から、MXT など、タスク制限に割り当てられた値を極端に大きくしたり、小さくしたりしないでください。

- 調整プロセス中は、制約を継続的にモニターしてください。各調整を行うとシステムの制約が変更されるため、時間と共にこれらの制約が変化します。制約が変更された場合、古い制約はパフォーマンスを制限する効果がなくなるため、新しい制約について調整する必要があります。さらに、制約は日中の時刻によって異なる場合があります。
- フォールバック・プロシージャを適切に配置してから、調整プロセスを開始してください。前述のとおり、調整によっては、予期せぬパフォーマンス結果が得られることがあります。これによりパフォーマンスの低下が生じる場合は、元に戻して、他の方法を試す必要があります。以前の定義またはパス設計が保管されていない場合、これらを再定義して、システムを元の状態に戻す必要があります。これらの回復作業が実行されるまで、システムは低レベルで実行されます。以前のセットアップが再呼び出し可能な方法で保管されている場合は、不正な変更をバックアウトする作業が大幅に簡単になります。

調整結果の検討

各調整を行った後に、パフォーマンス問題として識別されたパフォーマンス測定値を検討して、意図したとおりのパフォーマンス変更が行われたことを確認し、この変更を定量化します。サービス・レベルの合意を満たすレベルまでパフォーマンスが改善された場合は、これ以上の調整は不要です。パフォーマンスが向上したにもかかわらず、まだ不十分な場合は、調査を行って、次に実行するアクションを判断したり、調整したリソースが引き続き制約となっているかを確認する必要があります。調整したリソースが制約となっていない場合は、新しい制約を識別して調整する必要があります。この場合は調整プロセスの最初のステップに戻り、十分なパフォーマンス・レベルが得られるまで、調整プロセスの次のステップを繰り返す必要があります。

パフォーマンス・データを取得するための CICS ツール

CICS 統計、モニター、およびトレース機能を使用して、パフォーマンス・データを収集してモニターし、CICS システムを最適に調整するのに役立てることができます。追加の情報ソースもリストします。

CICS 統計

CICS 統計は、CICS システムを恒常的にモニターするための最も単純で最も重要なツールです。この統計は、タスクに関係なく、CICS システム全体の情報を収集します。

詳しくは、「[CICS 統計の概要](#)」を参照してください。

CICS モニター

CICS モニターは、後でオフライン分析ができるよう、オンライン処理中に、ユーザーおよび CICS のすべてのトランザクションのパフォーマンスに関するデータを収集します。

詳しくは、「[CICS モニターのデータの収集および処理](#)」を参照してください。

CICS トレース

システム対話を含むより複雑な問題の場合、CICS トレースを使用して、CICS 管理モジュールを通過する CICS トランザクションの進行状況を記録できます。

CICS トレースは、特定の状態に至るイベントのヒストリーを提供します。

また、CICS トレース機能は、システムにおけるイベントの待機超過などのパフォーマンス上の問題、または非効率的なシステム・セットアップやアプリケーション・プログラム設計によって生じる制約の分析にも役立ちます。

詳しくは、[CICS trace](#) を参照してください。

他の情報ソース

これまでに説明した測定ツールは、現行のシステム・パフォーマンスを完全に評価するのに必要なすべてのデータを提供するわけではありません。それらのツールは、各リソースがどのように、またどのような条件の下で使用されているかについての情報は提供しません。また、データ収集時の既存のシステム構成に関する情報も提供しません。したがって、システムの情報を取得するには、できるだけ多くの手法を使用することが重要です。追加の情報ソースには、以下のものが含まれます。

- ハードウェア構成
- VTOC リスト
- LISTCAT (VSAM)
- インストールされているリソース定義
- リンク・パック域 (LPA) マップ
- CICS 中核のロード・モジュール相互参照
- SYS1.PARMLIB リスト
- z/OS ワークロード・マネージャー (WLM) サービス定義
- MVS システム・ロガー構成 - LOGR 結合データ・セット・リスト
- CICS アドレス・スペースのダンプ
- TCP/IP プロファイル・データ・セット

システム管理機能 (SMF)

z/OS システムは、タスクの存続期間中に特定のイベントが発生したときに、各タスクの統計データを収集します。システム管理機能 (SMF) は、収集した情報をシステム関連 (または、ジョブ関連) のレコードにフォーマット設定します。

システム関連の SMF レコードには、構成、ページング・アクティビティー、およびワークロードに関する情報が含まれます。ジョブ関連のレコードには、各ジョブ・ステップ、ジョブ、APPC/MVS トランザクション・プログラム、および TSO/E セッションのプロセッサ時間、SYSOUT アクティビティー、およびデータ・セット・アクティビティーに関する情報が含まれます。

SMF によって収集された情報は、以下のタスクを実行するときに役立ちます。

- ユーザーに対する課金
- 信頼性の報告
- 構成の分析
- ジョブのスケジューリング
- 直接アクセス・ボリューム活動の要約
- データ・セット活動の評価
- システム・リソース使用のプロファイル作成
- システム・セキュリティの管理

詳しくは、[z/OS MVS システム管理機能 \(SMF\)](#)を参照してください。

汎用トレース機能 (GTF)

GTF は、CICS トレース項目を記録するために使用できる MVS システムの部分です。

GTF を使用して CICS トレース項目を記録し、対話式問題管理システム (IPCS) を使用してレポートを作成できます。より一般的には、GTF は MVS システムの不可欠な部分であり、次のシステム・イベントをトレースします。つまり、入出力開始命令の DASD シーク・アドレス、システム・リソース・マネージャー (SRM) アクティビティ、ページ不在、入出力アクティビティ、および監視プログラム・サービスです。実行オプションは、トレースするシステム・イベントを指定します。

GTF は通常、短期間のシステム活動のモニターに使用するため、それに応じて実行する必要があります。

GTF に要する処理時間は、トレースするイベントの数によって大きく異なることがあります。すべての GTF トレースにおいて、EXEC ステートメントで TIME=YES オペランドを指定して、GTF レコードのタイム・スタンプを要求する必要があります。

レコードが失われないようにするために、GTF は、255 のディスパッチング優先順位 (DPRTY) で実行します。255 の DPRTY を指定していた場合に GTF レコードが失われたときは、実行ステートメントで、10 パッファより多い BUF オペランドを指定します。

CICS パフォーマンスの検討に通常必要なデータを入手するには、以下のオプションを使用できます。

```
TRACE=SYS,RNIO,USR      (VTAM)
TRACE=SYS                (Non-VTAM)
```

注: VTAM は、z/OS Communications Server として知られるようになりました。

システムによってディスパッチされた作業単位に関するデータ、および SVC や LOAD などのイベントを実行するために必要な時間の長さに関するデータが必要な場合、オプションは次のようになります。

```
TRACE=SYS,SRM,DSP,TRC,PCI,USR,RNIO
```

TRC オプションは、GTF がトレースしている他のタスクの GTF 割り込みを示す GTF トレース・レコードを生成します。このオプションの設定では、高いパーセントのプロセッサ・リソースが使用されるため、イベントの詳細な分析やタイミングが必要な場合に限ってその設定を使用してください。

GTF ではデータ削減プログラムは提供されません。データを、意味のある、管理可能な形式で抽出し、要約するには、データ削減プログラムを作成するか、使用可能ないずれかの提供プログラムを使用することができます。

詳細については、[z/OS MVS 診断 ツールと保守援助プログラム](#)を参照してください。

汎用トレース機能 (GTF) レポート

対話式問題管理システム (IPCS) で GTF データからレポートを作成できます。IPCS によって生成したレポートは、システムおよび個別ジョブの両方のパフォーマンスを評価するのに役立ちます。

IPCS は、ジョブとシステムのサマリー・レポート、および短縮された詳細トレース・レポートも作成します。サマリー・レポートには、MVS ディスパッチ、SVC 使用量、内容監視、I/O の数とタイミング、シーク分析、ページ不在、および GTF によってトレースされるその他のイベントに関する情報が含まれます。詳細トレース・レポートは、システム内でトランザクションを時間順に追跡するのに使用できます。

他のデータをマップするその他のレポートが使用可能です。

- 特定ボリュームのシーク・アドレス
- 特定ボリュームのアーム移動
- 区分データ・セット内のデータ・セットおよびメンバーに対する参照
- リンク・バック域 (LPA) のページ不在およびモジュール参照

GTF を実行する前に、トレースするイベントを計画する必要があります。入出力開始 (SIO) などの特定のイベントをトレースせず、SIO-I/O タイミングが必要な場合は、そのレポートに必要なデータを取得するために、トレースを再作成する必要があります。

モニターしているシステムの制御装置への代替パスがある場合は、レポート実行ステートメントに PATHIO 入力ステートメントを組み込む必要があります。PATHIO オペランドを指定しない場合は、代替パスをもった装置のレポートに複数の I/O 行ができます。1 つの行は基本装置アドレスであり、もう 1 つの行は 2

次装置アドレスです。このオペランドを指定しない場合は、その装置の合計数を取得するために、基本装置アドレスと代替装置アドレスの I/O 数を手動で結合する必要があります。

シーク・ヒストグラム・レポート

シーク・ヒストグラム・レポート (SKHST) は、そのボリュームにアーム競合があるかどうか、つまり、マップされているボリュームに長いシークが存在するかどうかを調べるのに役立ちます。2つのレポートが作成されます。最初のレポートは、特定アドレスへのシーク数を示し、次のレポートは、アームがシーク間で移動する距離を示します。これらのレポートは、将来、特定ボリュームを再編成する必要があるかどうかを調べるために、ボリューム・マップ・レポートを要求する必要があるかどうかを判別するときに使用できます。

ボリューム・マップ・レポート

ボリューム・マップ・レポート (VOLMAP) は、マップされているボリューム上のデータ・セット、およびそのボリューム上の各データ・セットに対するシーク活動に関する情報を表示します。このレポートは、区分データ・セットのメンバー、および各メンバーに実行されたシーク数もマップします。このレポートは、ボリューム上のデータ・セットおよび区分データ・セット内のメンバーを再編成して、その特定ボリューム上のアーム移動を減らすのに役に立ちます。

参照マップ・レポート

参照マップ・レポート (REFMAP) は、MVS™ のリンク・バック域 (LPA) におけるページ不在活動を示します。この参照はモジュール名別になっており、データ不在と命令不履行は区別されます。このレポートは、特定モジュールに対する参照数も示します。この参照は、GTF の I/O および EXT 割り込みトレース・イベントの保管 PSW のアドレスから選択されます。このレポートは、実記憶域を減らすか、または MVS のページング可能リンク・バック域で検出されるページ不在数を減らすために、現行 MVS パック・リストを変更する場合に役立ちます。

CICS Performance Analyzer for z/OS (CICS PA)

CICS Performance Analyzer (CICS PA) は、CICS システムおよびアプリケーションのパフォーマンスに関する情報を提供するレポート作成ツールで、CICS システムを効率的に調整、管理、および計画するのに役立ちます。

CICS PA は、以下の場合に役立ちます。

- システム・プログラマーが CICS システム全体のパフォーマンスを追跡し、自分たちが行ったシステム調整の結果を評価する。
- アプリケーション・プログラマーが、自分たちのアプリケーションのパフォーマンスおよびアプリケーションが使用するリソースを分析する。
- データベース管理者が、データベース・システム (IMS や Db2 など) の使用およびパフォーマンスを分析する。
- IBM MQ 管理者が、自分たちの IBM MQ メッセージング・システムの使用およびパフォーマンスを分析する。
- 管理者が、トランザクションが必要なサービス・レベルを満たしていることを確認し、傾向を測定して将来の要件および戦略を計画するのに役立てる。

CICS PA では、レポートおよび抽出データの要求と実行依頼を支援するために、ISPF メニュー方式ダイアログを提供しています。使用可能なレポートおよび抽出データは、カテゴリー別にグループ化されています。

- Performance reports (パフォーマンス・レポート)
 - List (リスト)
 - List extended (拡張リスト)
 - 要約
 - Totals (合計)
 - Wait analysis (待機分析)

- Transaction profiling (トランザクション・プロファイル)
- Cross-system work (システム間処理)
- トランザクション・グループ
- BTS
- Workload activity (ワークロード・アクティビティ)
- Transaction tracking list (トランザクション・トラッキング・リスト)
- Transaction tracking summary (トランザクション・トラッキング要約)
- Exception reports (例外レポート)
 - List (リスト)
 - 要約
- Transaction resource usage reports (トランザクション・リソース使用レポート)
 - File usage summary (ファイル使用の要約)
 - Temporary storage usage summary (一時記憶域使用の要約)
 - DPL usage summary (DPL 使用量の要約)
 - Transaction resource usage list (トランザクション・リソース使用量リスト)
- Statistics reports (統計レポート)
 - List (リスト)
 - Alert (アラート)
 - CICS Transaction Gateway
- Subsystem reports (サブシステム・レポート)
 - Db2
 - IBM MQ
 - OMEGAMON
- System reports (システム・レポート)
 - System logger (システム・ロガー)
- Extracts (抽出データ)
 - Cross-system work (システム間処理)
 - Performance (パフォーマンス)
 - Record selection (レコード選択)
 - HDB load (HDB 負荷)
 - System logger (システム・ロガー)
 - Statistics (統計)

さらに CICS PA は、CICS トランザクションのパフォーマンス・データと統計データを管理するのに役立つ履歴データベース (HDB) 機能も提供しています。HDB コンテナ・データ・セットは、SMF データの保管場所となり、CICS PA ダイアログから管理されます。以下のタイプの HDB を使用できます。

- パフォーマンス・リスト HDB

リスト HDB は、CMF パフォーマンス・クラス・データから構築されます。リスト HDB データ・セットでは、1 つのレコードが 1 つのトランザクションを表します。一般には、リスト HDB は最近のレポート・トランザクションのイベントを分析するために使用されます。

- パフォーマンス・サマリー HDB

要約 HDB は、CMF パフォーマンス・クラス・データから構築されます。要約 HDB データ・セットでは、1 つのレコードが、ユーザー指定の時間間隔の間のトランザクション・アクティビティの要約を表します。一般には、要約 HDB は、長期間の傾向分析およびキャパシティー・プランニングに使用されます。

- 統計 HDB

統計 HDB には、指定された時間間隔における、CICS 統計、サーバー統計、および CICS Transaction Gateway 統計が収集されます。

CICS Performance Analyzer for z/OS についての詳細は、[CICS Performance Analyzer](#) の資料を参照してください。

CICS PA ダイアログ

CICS PA ダイアログは、レポート要求を作成、保守、および処理依頼するときに使用します。また、これを使用することにより、CMF データについて理解していなくても、入力データを指定して、自分の要件に固有の要求を調整することができます。

このダイアログには、特別なカスタマイズまたはセットアップは不要です。即時にレポート作成を開始できます。

以下のステップで、レポート作成にこのダイアログを使用する方法を説明します。

1. CICS (および関連するその他の) システム、および各システムの SMF ファイルとログ・ストリームを定義します。システムの定義が完了したら、そのシステムに対してレポート作成を開始できます。取り込み機能を使用すると、このプロセスを高速で追跡できます。CICS PA は、SMF ファイルから CICS システムに関する情報を抽出し、ダイアログでその情報を使用できるようにします。独自の CMF ユーザー・フィールドを定義している場合は、MCT 定義を指定してください。このようにすると、ユーザー・フィールドを CICS PA レポートに組み込むことができます。以下のパネル・サンプルでは、CICS PA に定義されているいくつかの CICS システム、Db2 サブシステム、IBM MQ サブシステム、および MVS システム・ロガーを示します。

System Definitions					Row 1 from 8
Command ==> _____ Scroll ==> CSR					
Select a System to edit its definition, SMF Files and Groups.					
/	System	Type	Image	Description	SMF Files
-	MVS1	Image		Production MVS system	System
-	CICSP1	CICS	MVS1	CICS Production System 1	MVS1
-	CICSPTOR	CICS	MVS1	CICS Production TOR	MVS1
-	CICSPAOR	CICS	MVS2	CICS Production AOR	CICSPAOR
-	CICSPFOR	CICS	MVS2	CICS Production FOR	CICSPFOR
-	DB2P	DB2	MVS3	DB2 Production Subsystem	DB2P
-	MQSP	MQ	MVS4	MQ Production Subsystem	MQSP
-	MVS1LOGR	Logger	MVS1	System Logger for MVS1	MVS1

図 2. CICS PA: システム定義

レポート作成の目的で、IRC/MRO や ISC/APPC を介して接続されているシステムなど、関連する CICS システムは、グループ化することができます。例えば、CICS MRO システム (CICSPTOR、CICSPAOR、CICSPFOR、CICSPDOR) をあるグループに割り当てることにより、これらのシステムを単一のエンティティとして報告することができます。したがって、CICS PA のレポートには、MRO トランザクション・アクティビティのエンドツーエンドの全体像が示され、詳細な Db2 統計 (サブシステム DB2P の Db2 アカウンティング・データから取得されます) が組み込まれています。

2. レポート要求を作成、処理依頼、および保管するために、レポート・セットを定義できます。レポート・セットには、単一のジョブで実行するレポートのセットが含まれています。単に、必要なレポートを選択して、レポート要求を処理依頼するだけです。

31 ページの図 3 にレポート・セットを示します。使用可能なレポートがツリー構造 (フォルダー・スタイル) で示され、カテゴリ別にグループ化されています。レポート・カテゴリは、必要に応じて拡張および縮小表示することができます。アクティブ状況は、レポート要求を処理依頼したときに、レポート・セット内のどのレポートを実行するのかを制御します。

```

EDIT                                     Report Set - DAILY                               Row 1 of 45
Command ===> -----Scroll ===> CSR

Description . . . . Daily Reports for our production MRO system

Enter "/" to select action.

---      ** Reports **                      Active
-  ---  Options                          Yes
      ---  Global                          Yes
-  ---  Selection Criteria                  Yes
      ---  Performance                      Yes
      ---  Exception                        No
-  ---  Performance Reports                Yes
      ---  List                            Yes
      ---  List Extended                    Yes
      ---  Summary                          Yes
      ---  Totals                          Yes
      ---  Wait Analysis                    No
      ---  Transaction Profiling            No
      ---  Cross-System Work                No
      ---  Transaction Group                Yes
      ---  BTS                             No
      ---  Workload Activity                No
      ---  Transaction Tracking List        No
      ---  Transaction Tracking Summary     No
-  ---  Exception Reports                  No
      ---  List                            No
      ---  Summary                          No
-  ---  Transaction Resource Usage Reports No
      ---  File Usage Summary              No
      ---  Temporary Storage Usage Summary No
      ---  DPL Usage Summary               No
      ---  Transaction Resource Usage List No
-  ---  Statistics Reports                 No
      ---  Alert                           No
      ---  CICS Transaction Gateway         No
-  ---  Subsystem Reports                  No
      ---  DB2                             No
      ---  WebSphere MQ                     No
      ---  OMEGAMON                         No
-  ---  Performance Graphs                No
      ---  Transaction Rate                 No
      ---  Transaction Response Time        No
-  ---  Extracts                          Yes
      ---  Cross-System Work                Yes
      ---  Performance                      No
      ---  Record Selection                 No
      ---  HDB Load                         No
      ---  System Logger                    No
      ---  Statistics                       No
      ** End of Reports **

```

図 3. CICS PA: レポート・セット

レポート・セットには、CMF レコードをフィルター操作するために使用される選択基準を含めることができます。これにより、興味のある情報だけを組み込むようレポート作成を調整することができます。例えば、レポートを以下のものに制限する選択基準を指定することができます。

- 特定の日時範囲
 - 関連するトランザクション ID のグループ
 - 設定したしきい値を超えるトランザクション応答時間
3. レポートのフォーマットおよび内容を調整するために、Report Form (レポート・フォーム) を定義することができます。エディターを使用して、必要な CMF フィールドを選択することにより独自のレポートを設計できます。レポート作成用のほとんどの CMF フィールドを選択可能で、各 CMF フィールドの詳細な説明をダイアログから表示できます。レポート・フォームには選択基準を含めることができます。あるレポートが Report Form を指定していて、両方に選択基準が指定されている場合、レコードがレポートに組み込まれるのは、両方の選択基準セットを満たす場合のみです。

32 ページの図 4 に、ファイル制御統計を示すように調整された Report Form を示します。

```

EDIT LIST Report Form - FCLIST                      Row 1 of 16 More: >
Command ===> ----- Scroll ===> CSR
Description . . . . File Control List Form          Version (VRM): 730
Selection Criteria:
- Performance *                                     Page width . . 132

Field
/ Name +      Type      Description
-- TRAN                      Transaction identifier
-- USERID                      User ID
-- STOP      TIMET      Task stop time
-- RESPONSE                      Transaction response time
-- DISPATCH  TIME      Dispatch time
-- CPU        TIME      CPU time
-- FCWAIT     TIME      File I/O wait time
-- FCAMCT                      File access-method requests
-- FCADD                      File ADD requests
-- FCBROWSE                      File Browse requests
-- FCDELETE                      File DELETE requests
-- FCGET                      File GET requests
-- FCPUT                      File PUT requests
-- FCTOTAL                      File Control requests
-- EOR                      ----- End of Report -----
-- EOX                      ----- End of Extract -----

```

図 4. CICS PA: レポート・フォーム

4. パフォーマンス・データのリポジトリとして履歴データベース (HDB) を定義し、保守します。HDB と対照してレポートを生成するか、HDB データを Db2 のテーブルにエクスポートして、詳細な分析を行います。

CICS PA を使用した CICS パフォーマンスの分析

CICS PA は、以下に示した、CICS システムおよびアプリケーションのパフォーマンスの分析および調整に役立つレポートおよび抽出を提供します。

- 「パフォーマンス・リスト」、「拡張リスト」、および「要約」の各レポート。これらは、トランザクション・アクティビティの詳細な分析を提供します。
- 「パフォーマンス合計報告書」。CICS システム全体または個々のトランザクションの包括的なリソース使用分析を提供します。
- 「待機分析」レポート。このレポートは、待機時間別のトランザクションのアクティビティを要約します。トランザクション ID 別に、このトランザクションの中断の原因となるリソースが、コストが最も高くつくものから最も安いものの順に表示されます。このレポートでは、応答時間を遅くする可能性のあるシステム・リソースのボトルネックが強調表示されます。識別された問題のリソースに焦点を絞ることにより、より詳細な分析を実行することができます。
- 「Transaction Profiling Report」は、2 つの CMF パフォーマンス・クラス・データ・セットを比較します。例えば、2 つの異なる期間における特定の CICS アプリケーションのパフォーマンス・データや、2 つのシステム上にあるすべてのアプリケーションのパフォーマンス・データなどを比較できます。
- 「システム間作業報告書」。このレポートでは、接続されているシステム (MRO や APPC など) からの CMF レコードを結合して、統合された作業単位レポートが作成されます。
- 「システム間作業抽出」。同じ作業単位の CMF レコードが、CMF フォーマットで単一のレコードに統合されます。その後、抽出データ・セットを CICS PA で処理し、任意のレポートを作成できます。例えば、「開始トランザクション ID が TR01 であるマルチシステムの UOW をすべて要約する」レポートを作成できます。
- 「トランザクション・グループ」レポート。着信する処理要求の詳細リストを提供します。着信する同じ処理要求の下で CICS が実行するトランザクション (例えば、CICS Web サポート要求に対する CWXN および CWBA トランザクション) は、このレポートでまとめてグループ化されます。
- 「BTS 報告書」。このレポートは、CICS ビジネス・トランザクション・サービス・アクティビティの詳細リストを提供します。同一の CICS BTS プロセス ID (ルート・アクティビティ ID) を持つトランザクションは、このレポートでまとめてグループ化されます。
- 「作業アクティビティ」レポート。このレポートは、z/OS ワークロード・マネージャー (WLM) のサービスおよびレポート・クラス別にトランザクション応答時間分析を提供します。この情報を使用すること

により、CICS の観点から、CICS トランザクションが自分たちの応答時間目標をどの程度満たしているのかを理解することができます。「ワークロード・アクティビティ・リスト」レポートは、ネットワークの作業単位ごとに、単一または複数の CICS システムからの CMF パフォーマンス・クラス・データを相互に関連させるシステム間レポートです。重要なことは、このレポートが MRO および機能シップ・タスクをそれぞれの親タスクに結びつけ、応答時間に与えるそれぞれの影響を評価することができることです。

- 「トランザクション・トラッキング・リスト・レポート」。このレポートは、関連トランザクションのグループのパフォーマンス・データを提供します。これにより、トランザクション・フローの観点からトランザクション・パフォーマンスを監視および計測できます。このレポートは、プロセスがどのように 1 つのトランザクションやシステムから次のものに流れて戻ってくるかを示します。このレポートは、各発信元トランザクションとその従属 (グループ) トランザクションの CMF レコードを組み合わせます。
- 「トランザクション・トラッキング要約報告書」。このレポートは、各発信元トランザクションとその従属 (グループ) トランザクションの CMF レコードを組み合わせます。集計データは、発信元トランザクションのグループごとに 1 行ずつ表示されます。
- 「例外リスト」および「要約」レポート。これらのレポートは、CMF が記録した例外イベントの詳細な分析を提供します。
- 「トランザクション資源使用報告書」。CMF パフォーマンス・データおよび CMF リソース・クラス・データを処理して、ファイル、一時記憶域、分散プログラム・リンク (DPL) の使用の詳細な分析を提供します。
- 「統計アラート」。これは、CICS Transaction Server 統計または CICS Transaction Gateway 統計のフィールド値に関して、対象とする条件を定義することができます。次いでこれらの条件を使用して、SMF ファイルまたは履歴データベースに保管されている CICS 統計のレポートを作成することができます。
- 「CICS Transaction Gateway 報告書」。このレポートは、CICS Transaction Gateway 統計 SMF 111 レコードのレポートを作成します。使用可能なレポートは以下のとおりです。
 - アクティビティ
 - 使用量および容量
 - 構成
 - クライアント・ワークロード
 - CICS Workload
- 「Db2」レポートは、CICS CMF レコードおよび Db2 会計レコードを処理して、CICS システム別に Db2 使用の統合された、詳細ビューを作成します。このレポートを使用すると、単一のレポートで、CICS および Db2 リソースの使用統計を一緒に表示することができます。「Db2 リスト」レポートには、トランザクション別に Db2 アクティビティの詳細情報が示されます。「Db2 要約」レポートには、APPLID トランザクションおよびプログラム別に、Db2 が要約されます。
- IBM MQ レポートは、IBM MQ 会計 (SMF 116) レコードを処理して、IBM MQ 使用状況の詳細を CICS システム別に示すビューを作成します。IBM MQ リスト・レポートは、IBM MQ 会計レコードのトレースを示します。IBM MQ 要約レポートは、IBM MQ トランザクションの要約ビューを 2 つ示します。CICS トランザクション ID 別に IBM MQ システムおよびキューのリソースの使用状況を示すものと、IBM MQ キュー名別に、キューで処理されたトランザクションおよびリソースの使用状況を示すものです。
- 「OMEGAMON レポート」は、OMEGAMON XE for CICS (SMF 112) レコードを処理して、CICS トランザクションが Adabas、CA-Datcom、CA-IDMS、および Supra を使用方法を示す詳細なビューを作成します。DBMS タイプごとに、以下のレポートを要求できます。
 - トランザクション別のデータベース使用量を示す「リスト」レポート
 - トランザクション ID 別に集計されたデータベース使用量を示す「トランザクション要約」レポート。
 - データベース別に集計されたデータベース使用量を示す「データベース要約」レポート。
- 「システム・ロガー」レポートでは、CICS Transaction Server がシステム・ロガーのレコードを処理して、ロギング、リカバリー、およびバックアウト操作に使用するシステム・ロガーのログ・ストリームおよびカップリング・ファシリティ構造に関する情報が提供されています。このレポートは、調整変更およびログ・ストリームの影響の評価、および構造のパフォーマンスの問題の識別に役立てることができます。

- 「パフォーマンス・グラフ」レポート。このレポートは、トランザクション比率および応答時間をグラフィカルに表示します。
- 「抽出」データ・セットは、SMF データから作成され、さらに操作や分析を行うときに適しています。以下の抽出データを使用して、Db2 などの外部プログラムやスプレッドシート・プログラムなどの PC ツールにデータをインポートできます。

- CMF パフォーマンス・クラス・データ用の「パフォーマンス・データ抽出」
- システム・ロガー・データ用の「System Logger 抽出」
- CICS 統計用の「統計抽出」

「システム間作業抽出」については、このトピックの前の部分で説明されています。「レコード選択抽出」を使用すると、レポート作成を効率的に行うため、CICS PA によって処理されるデータ量を削減することができます。「HDB ロード」を使用すると、SMF データを履歴データベース (HDB) にロードできます。

「Report Form」を使用すると、レポートおよび抽出のフォーマットを調整することができます。例えば、フィールド、列の順序、およびソート・シーケンスを指定することができます。

「選択基準」を使用すると、レポートをフィルターに掛けて、例えば特定のトランザクション ID のみ、または特定の期間だけのデータを組み込むようにすることができます。

CICS Performance Analyzer for z/OS についての詳細は、[CICS Performance Analyzer](#) の資料を参照してください。

パフォーマンス・データを取得するためのその他のツール

CICS には付属していない各種のツールを使用して、パフォーマンス関連の情報を提供すると、CICS システムを最適に調整するのに役立ちます。

z/OS Resource Measurement Facility は、シスプレックス内のアクティビティーに関するデータを収集し、レポートを作成します。詳しくは、[z/OS リソース測定機能 \(RMF\) ユーザーズ・ガイド](#)を参照してください。

IBM Redbooks [ABCs of z/OS System Programming](#) には、キャパシティー・プランニング、パフォーマンス管理、RMF、および SMF に関する情報が記載されています。

リソース測定機能 (RMF)

リソース測定機能 (RMF) は、プロセッサ活動 (WAIT 時間)、入出力活動 (チャンネルおよび装置使用量)、主記憶域活動 (要求およびスワップ・ページング統計)、およびシステム・リソース・マネージャー (SRM) 活動 (ワークロード) を説明する、システム全体のデータを収集します。

RMF は、システム活動をモニターして、パフォーマンスとキャパシティー・プランニング・データを収集する中央測定ツールです。RMF レポートの分析を行うと、ユーザー要求に対してシステムをチューニングするための基礎的データが得られます。それらのデータを リソース使用量の追跡に使用することもできます。

RMF は以下の活動を測定します。

- プロセッサ使用量
- アドレス・スペース使用量
- チャンネル活動
 - 物理チャンネル当たりの要求率およびサービス時間
 - 論理チャンネルと物理チャンネル間の関係
 - 論理チャンネル・キュー項目数およびキューイングの理由
- 以下の装置の装置活動および競合
 - ユニット・レコード
 - グラフィックス
 - 直接アクセス・ストレージ
 - 通信装置

- 磁気テープ
- 文字読取装置
- 詳細なシステム・ページング
- 詳細なシステム・ワークロード
- ページおよびスワップ・データ・セット
- エンキュー
- CF 活動
- XCF 活動

RMF では、z/OS ユーザーは以下のことができます。

- システムの応答性の評価
 - ボトルネックの識別。 ページおよびスワップ・データ・セット活動に関連した 詳細なページング・レポートにより、仮想記憶環境の動作の状況をよく理解できます。
- チューニングの影響の検査
 - 結果を画面上で動的に、またはポストプロセッシング機能によって監視できます。
- キャパシティー・プランニング評価の実行
 - ワークロード活動報告書には、プロセッサ、入出力、および主記憶域サービスなどの 主要エレメントによって分解されたインターバル・サービスが含まれます。
 - リソース・モニター出力 (例えば、システム競合インディケーター、カテゴリーごとに分割された スワップアウト、ドメインごとの平均作動可能ユーザー) の分析は、ユーザー環境の理解および傾向の予測に役立ちます。
 - ポストプロセッシング機能により、ピーク・ロード期間の分析およびトレンド分析が 簡単になります。
- MVS がサポート できる大規模ワークロードおよび増大したリソースの管理
- オンライン・チャネル・パス使用量の識別および測定

RMF について詳しくは、IBM Redbooks 資料「[ABCs of z/OS System Programming](#)」および「[z/OS リソース測定機能 \(RMF\) ユーザーズ・ガイド](#)」を参照してください。

パフォーマンス・データを取得するために IMS で提供されているツール

IMS Performance Analyzer (IMS PA) および IMS プログラム分離 (PI) トレースを使用すると、CICS やオペレーティング・システムで使用される各種のアクセス方式および他のプログラムに関する情報をモニターできます。

IMS Performance Analyzer (IMS PA)

IMS Performance Analyzer は、IMS のデータベースおよびトランザクション・マネージャー・システム用のパフォーマンス分析およびチューニング援助プログラムです。 このプログラムは、IMS ログおよびモニター・データ (ファスト・パス・データを含む) を処理して、IMS システムの分析とチューニングに役立つ、包括的なパフォーマンス、使用量、および可用性レポートを提供します。

IMS PA:

- ログおよびモニター・データを使用して、アプリケーションおよび内部リソース使用率、プロセッサ使用量、全機能およびファスト・パス・データベース・アクティビティーを表示した、包括的な DBCTL レポートを作成します。
- IMS ログ・データを使用して、通過時間 (実際のシステム・パフォーマンス時間)、および IMS リソースの使用量と可用性に関する包括的な情報を作成します。
- 通過時間の時間間隔ごとの抽出データを作成し、それを外部プログラムが処理できるように、グラフにしたり、エクスポートしたりできます。PC にダウンロードすることもできます。
- 合計トランザクション・トラフィックおよび例外トランザクション (MSGQ またはファスト・パス) の抽出データを作成して、外部プログラムで直接インポートできるようにします。

- 単一 IMS システムからのログ、または SYSPLEX で稼働し、共用キューを使用する、複数 IMS サブシステムからのログを処理します。
- モニター・データを使用して、領域、リソース、プログラム、トランザクション、データベース、およびシステム全体の、詳細レベルおよび分析領域ごとに編成された、サマリーおよび分析レポートを作成します。

詳しくは、[IMS Performance Analyzer for z/OS](#) を参照してください。

IMS プログラム分離 (PI) トレース

プログラム分離 (PI) トレースは、特定データベースにアクセスするというタスクの性質によって発生するデータベース競合問題を指し示すことができます。

一度に 1 つのタスクしかレコードにアクセスできないため、他のタスクはレコードが解放されるまで待ちます。このため、競合数が多い場合は、応答時間が長くなります。このトレースは、IMS の一部です。PI トレース・レポートの形式については、[IMS 製品資料内の『システム管理』](#)を参照してください。

TCP/IP のモニター

TCP/IP は、物理的に分離されたコンピューター・システム間で使用される通信プロトコルです。幅広いネットワーク上に TCP/IP を実装できます。TCP/IP は、プロトコルの大きなファミリーであり、その最も重要なメンバーである伝送制御プロトコル (Transmission Control Protocol) とインターネット・プロトコル (Internet Protocol) の名前をとって命名されています。

インターネット・プロトコル (IP) は、ネットワーク層のプロトコルです。コネクションレス・データ伝送サービスを提供し、TCP とユーザー・データグラム・プロトコル (UDP) の両方をサポートします。データはリンクからリンクへと伝送され、呼び出し中にエンドツーエンド接続がセットアップされることはありません。データ伝送の単位はデータグラムです。

伝送制御プロトコル (TCP) は、トランスポート層のプロトコルです。アプリケーション間にコネクション指向のデータ伝送サービスを提供します。つまり、データ伝送を開始する前に接続が確立されます。TCP は UDP よりもエラー・チェック機能が優れています。

UDP は、トランスポート層プロトコルで、TCP の代替の 1 つです。アプリケーション間にコネクションレス・データ伝送サービスを提供します。UDP は TCP に比べてエラー・チェック機能が劣ります。UDP のユーザーがエラーに対応できるようにする必要がある場合、通信プログラムはエラー処理のための独自のプロトコルを設定する必要があります。高品質の伝送ネットワークを使用している場合は、UDP のエラーはあまり懸念する必要はありません。

TCP/IP について詳しくは、[インターネット、TCP/IP、および HTTP の概念](#)を参照してください。

TCP/IP 管理および制御を使用して、CICS で収集されたデータを保管できます。これにより、関連のタスクやリソースが使用可能でなくなった後のある時点で、オフラインでデータを調査することができます。また、TCP/IP 管理および制御を使用すると、TCP/IP ネットワークの CICSplex 全体のビューを取得し、以下の項目をリアルタイムで調べることができます。

- 特定の CICS 領域が使用している TCP/IP ネットワーク・リソース。
- TCP/IP ネットワーク経由で特定の CICS 領域と受け渡しされている処理。
- TCP/IP ネットワーク経由で CICSplex 全体に流れる分散トランザクションに関連付けられている CICS リソースおよびタスク。
- 分散トランザクションの発信元の CICS 領域。

TCP/IP 管理および制御を使用すると、接続問題やトランザクション遅延などの問題の診断、CICSplex 全体の処理の追跡、CICSplex のモニター、およびキャパシティー・プランニングに使用するための一定期間のシステム・データの収集を行うことができます。

IBM Z Decision Support

IBM Z Decision Support (旧称 Tivoli Decision Support for z/OS) は、CICS および他の IBM システムおよび製品のデータを収集し、分析する IBM 製品です。

IBM Z Decision Support によって生成されるレポートは、以下の目的のために役立ちます。

- システムの概要の入手

- サービス・レベルの維持の確認
- 可用性の確認
- パフォーマンスの調整
- キャパシティー・プランニング
- 変更および問題の管理
- アカウンティング

多くの既成のレポートを使用できます。特定の要求に合う独自のレポートを作成することもできます。

レポートにおいて、IBM Z Decision Support は、CICS モニターおよび統計のデータを使用します。IBM Z Decision Support は、MVS システムのデータ、および RMF、TSO、IMS、NetView などの製品のデータも収集します。つまり、CICS と他のシステムのデータを共に表示したり、別々のレポートに表示したりできます。

レポートは、図、棒グラフ、円グラフ、タワー図表、ヒストグラム、面グラフ、およびその他のグラフ形式で表すことができます。IBM Z Decision Support は、データおよび詳細なフォーマット設定を、残りの作業を実行する IBM Graphic Data Display Manager (GDDM) に渡します。) IBM Z Decision Support は、GDDM が使用できないか、または出力装置がグラフをサポートしない場合は、文字グラフィックを使用して折れ線グラフおよびヒストグラムを作成することもできます。正確な数値が必要なレポートの場合は、テーブルやマトリックスなどの数値レポートがより適しています。

IBM Z Decision Support を使用した CICS パフォーマンスのレポート

パフォーマンス・データを理解するには、まず最初に、インストールされた CICS 作業パフォーマンスを理解する必要があります。基本的なビルド・ブロックであるトランザクションごとに作業を分析します。トランザクションを類似リソースのカテゴリまたはユーザー要件にグループ化して、各カテゴリの特性について説明します。それぞれのトランザクションごとに CICS が実行する作業、および特定の期間に期待されるトランザクションの容量を理解してください。IBM Z Decision Support は、CICS で処理されるトランザクションに対して、さまざまなタイプのデータを表示できます。

CICS ユーザー・グループのサービス・レベル・アグリーメントは、計量可能な CICS 関連のリソースとサービスのいくつかの領域でコミットメントを定義します。CICS サービス・コミットメントは、以下のいずれかの領域に属します。

- 応答時間
- トランザクション・カウント
- 例外と機能不良
- 可用性

以下のトピックでは、特定の問題とシステム管理に関連した懸念事項、および IBM Z Decision Support CICS パフォーマンス・フィーチャーの使用方法について説明します。

IBM Z Decision Support によるパフォーマンス測定

IBM Z Decision Support (旧称 Tivoli Decision Support for z/OS) は、Db2 を使用するレポート・システムです。コンピューター・システムによってログ・データ・セットに書き込まれる使用率およびスループットの統計を処理するために使用できます。これを使用して、データを分析したり Db2 に保管して、さまざまなフォームで提出することができます。

IBM Z Decision Support は、基本製品およびシステム管理で使用されるオプション・フィーチャーで構成されます。

- **IBM Z Decision Support およびオプション・フィーチャー:**
- CICS パフォーマンス・フィーチャー
- IMS パフォーマンス・フィーチャー
- ネットワーク・パフォーマンス・フィーチャー
- システム・パフォーマンス・フィーチャー
- ワークステーション・パフォーマンス・フィーチャー

- iSeries パフォーマンス・フィーチャー
- アカウンティング・フィーチャー

IBM Z Decision Support 基本機能には、次のものがあります。

- 対話式システム生産性向上機能 (ISPF) を使用したダイアログのレポートと管理
- 独自の言語を使用した、ログ・データを読み取るためのコレクター機能
- フィーチャーで使用されるすべてのデータ・レコードのフィーチャー・マッピング (定義)

各フィーチャーには、以下のものがあります。

- ログ・データを Db2 テーブルに転送するための命令 (コレクター言語)
- Db2 テーブル定義
- レポート

IBM Z Decision Support データベースには、多くのソースのデータを含むことができます。例えば、システム管理機能 (SMF)、リソース測定機能 (RMF)、CICS、および情報管理システム (IMS) からのデータを 1 つのレポートに統合することができます。実際に、IBM Z Decision Support に標準ソース以外のログ・データを定義して、標準データから受信したデータと共にレポートできます。

IBM Z Decision Support CICS パフォーマンス・フィーチャーでは、CICS モニター機能 (CMF) と CICS 統計からのデータに基づいて、CICS Transaction Server のパフォーマンスを分析する場合に使用するレポートが提供されます。IBM Z Decision Support でレポートされる領域は、以下のとおりです。

- 応答時間
- リソース使用量
- プロセッサ使用量
- ストレージ使用量
- ボリュームとスループット
- CICS および Db2 アクティビティ
- 例外と機能不良
- キーとして作動する作業単位を使用した接続状態領域からのデータ
- CICS アベイラビリティ
- CICS リソース可用性

IBM Z Decision Support CICS パフォーマンス・フィーチャーは、CICS ユーザーの要求を満たすために必要なデータのみを収集します。そのデータを追加データ (環境データと呼ばれる) と結合して、さまざまなレポートで提出できます。IBM Z Decision Support は、環境データの保守用に管理ダイアログを提供します。39 ページの図 5 では、IBM Z Decision Support レポートに表示するためにデータがどのように編成されるかを説明します。

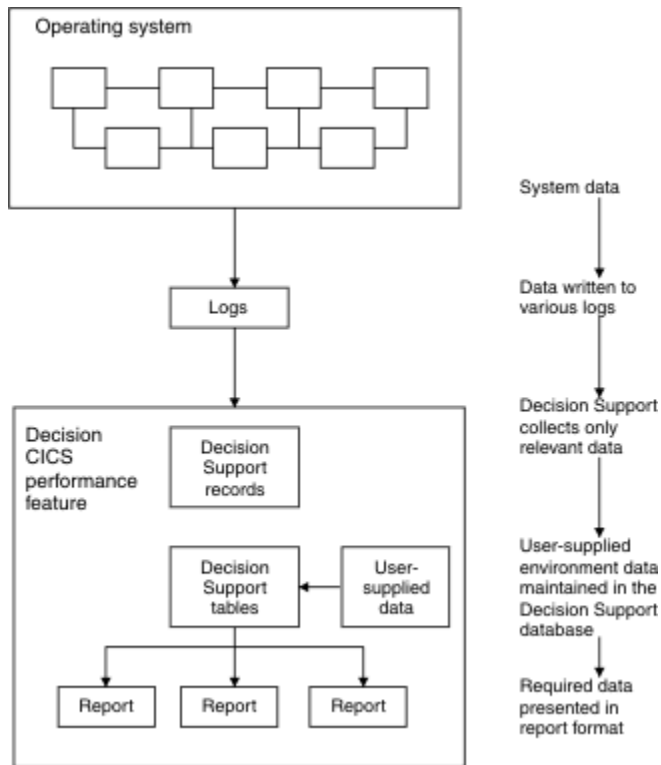


図 5. システム・パフォーマンス・データの編成および表示

IBM Z Decision Support for z/OS CICS パフォーマンス・フィーチャーは、以下のレコードを処理します。

CMF

- CICS Transaction Server パフォーマンス
- CICS Transaction Server 例外
- CICS Transaction Server アカウンティング、パフォーマンス、および例外

統計

- CICS Transaction Server 統計

モニター応答時間

応答時間は、トランザクションのアクティビティーの開始から終了までの合計時間を示し、これは中断時間とディスパッチ時間に細分されます。ディスパッチ時間にはサービス時間が含まれます。IBM Z Decision Support の CICS 応答時間レポートを使用して、CICS アプリケーションの内部応答時間を見ることができます。

応答時間レポートのエレメントを [39 ページの図 6](#) に示します。



図 6. CICS 内部応答時間エレメント

「IBM Z Decision Support 製品資料」で説明されているとおり、ネットワーク・パフォーマンス機能は、論理装置ごとに、SNA アプリケーション (例えば、CICS 領域) に対する合計、およびエンドツーエンド平均応答時間 (オペレーター通過時間) を表示するレポートを生成します。オペレーター通過時間は、ホスト通過時間とネットワーク通過時間で構成されます。これは、ネットワーク・パフォーマンス機能レポートにも表示されます。これらのレポートを使用すると、応答時間の問題をネットワークまたは CICS に切り分けることができ、問題に応じて対応できます。CICS の問題では、IBM Z Decision Support CICS パフォーマ

ンス・フィーチャー・レポートを使用して、応答時間の遅延の原因となっているアプリケーションを識別します。

プロセッサおよびストレージ使用量のモニター

応答時間が十分でない場合は、常にプロセッサまたはストレージの使用が非効率であることを示します。IBM Z Decision Support が提供するレポートは、CICS のパフォーマンス上の問題となるリソースを切り分ける手助けをします。

IBM Z Decision Support CICS パフォーマンス・フィーチャーの統計コンポーネント、および IBM Z Decision Support System Performance フィーチャーの MVS コンポーネントは、両方ともインストールされ、アクティブになっています。これらのレポートを使用して、CICS 領域で使用される トランザクション比率とプロセッサの分析に使用できます。

- ・ 月次レポート CICS Transaction Processor Utilization には、指定した日付の月次平均が表示されます。
- ・ 日時レポート CICS Transaction Processor Utilization には、指定した日付の日時平均が表示されます。

IBM Z Decision Support は、ストレージ使用量の分析に役立つレポートをいくつか作成します。例えば、CICS 動的ストレージ (DSA) 使用量レポートは、見出し「**Pagepool name**」、「**DSA (bytes)**」、「**Cushion (bytes)**」、「**Free storage (bytes)**」、「**Free storage (pct)**」、「**Largest free area**」、「**Getmains**」、および「**Freemains**」の下にページ・プール使用量を表示します。

CICS Dynamic Storage (DSA) Usage							
MVS ID = 'MV28'				CICS ID = 'IYCSTSK'			
Date: '2001-01-17' to '2001-01-18'							
Pagepool name	DSA (bytes)	Cushion (bytes)	Free storage (bytes)	Free storage (pct)	Largest free area	Getmains	Freemains
CDSA	524288	65536	299008	57	245760	2668	2470
ECDSA	5242880	131072	1122304	21	868352	1084154	1067000
ERDSA	11534336	262144	1130496	9	966656	710	16
ESDSA	0	0	0	0	0	0	0
EUDSA	2097152	0	2097152	100	1048576	73620	73620
RDSA	524288	65536	204800	39	122880	40	0
SDSA	262114	65536	249856	95	249856	12	6
UDSA	524288	65536	524288	100	262114	301922	301922

Tivoli Decision Support Report: CICS809

図 7. CICS 動的ストレージ (DSA) 使用量レポート

ボリュームおよびスループットのモニター

パフォーマンス上の問題が過度のページングであると思われる場合は、IBM Z Decision Support を使用して、RMF データを使用するページインをレポートできます。

CICS Transaction Server for z/OS, バージョン 5 リリース 6 はページに MVS サブタスクを使用し、MVS ページインは MVS タスクが停止実行する原因となるため、ページインの回数はパフォーマンスの懸念事項になります。ページアウトは CICS 処理の中休み中に発生するようにスケジュールされているので、問題はありません。

トランザクションのパフォーマンスの最良のインディケーターは、その応答です。トランザクション ID ごとに、CICS トランザクション・パフォーマンス明細報告書 (41 ページの図 8 内) に、合計トランザクション・カウントと応答時間の平均が表示されます。見出しは、「Tran ID (トランザクション ID)」、「Tran count (トランザクション・カウント)」、「Average resp time (sec) (平均応答時間 (秒))」、「Average CPU time (sec) (平均 CPU 時間 (秒))」、「Prog load reqs (avg) (プログラム・ロード要求数 (平均))」、「FC calls (avg) (FC 呼び出し回数 (平均))」、「Exceptions (例外)」、「Program storage bytes (max) (プログラム・ストレージ・バイト数 (平均))」、「Getmains < 16 MB (avg) (Getmains < 16 MB (平均))」、および「Getmains > 16 MB (avg) (Getmains > 16 MB (平均))」になります。このレポートを使用して、まずサービス・レベル目標を満たしていることを確認してください。まず最初に、平均応答時間の値が受け入れ可能であることを確認します。次に、トランザクション・カウントが決められた限界を超えていないことを確認します。トランザクションが適切なサービスのレベルを受信しない場合は、遅延の原因を判別する必要があります。

CICS Transaction Performance, Detail										
MVS ID = 'MV28' CICS ID = 'IYCSTSK'										
Date: '2001-01-17' to '2001-01-18'										
Tran ID	Tran count	Avg resp time (sec)	Avg CPU time (sec)	Prog load reqs (avg)	Prog loads (avg)	FC calls (avg)	Excep-tions	Program storage bytes (max)	Getmains < 16 MB (avg)	Getmains > 16 MB (avg)
QUIT	7916	0.085	0.017	0	0	18	0	74344	22	0
CRTE	1760	4.847	0.004	0	0	0	0	210176	1	0
AP00	1750	0.184	0.036	0	0	8	0	309800	66	0
PM94	1369	0.086	0.012	0	0	6	0	130096	24	0
VCS1	737	0.073	0.008	2	0	7	0	81200	14	0
PM80	666	1.053	0.155	1	0	62	0	104568	583	0
CESN	618	8.800	0.001	0	0	0	0	41608	0	0
SU01	487	0.441	0.062	4	0	126	0	177536	38	0
...										
GC11	1	0.341	0.014	1	0	2	0	37048	10	0
DM08	1	0.028	0.002	0	0	0	0	5040	3	0
=====	20359							=====		
								309800		

Tivoli Decision Support Report: CICS101

図 8. CICS トランザクション・パフォーマンス、明細報告書

CICS および Db2 パフォーマンス・データの結合

CICS と Db2 のパフォーマンス・データを組み合わせることにより、CICS トランザクションが原因で発生した Db2 アクティビティーを表示するレポートを作成できます。

CICS タスクごとに、CICS は LU6.2 作業単位 ID を生成します。また、Db2 も LU6.2 作業単位 ID を作成します。41 ページの図 9 には、タスクを識別するために、Db2 トークン (QWHCTOKN) を使用して、Db2 データが CICS パフォーマンス・データと関連付けられる方法が示されています。

DB2 accounting record



CICS performance-monitoring record

TRAN	USERID	NETNAME	UOWID	TCIOWT

図 9. CICS パフォーマンス・モニター・レコードと Db2 会計レコードとの相関

CICS レコードの NETUOWPX および NETUOWSX フィールドを、Db2 トークンと突き合わせると、CICS トランザクションによる Db2 アクティビティーを表示するレポートを作成できます。

例外および機能不良データのモニター

例外はモニターすべきイベントです。発生した場合にのみレポートに例外が表示されます。つまり、レポートにはヌル・カウントは表示されません。単一の例外は、必ずしもアラームの原因にはなりません。機能不良は、重大度 1、2、または 3 と共に例外として定義されます。

IBM Z Decision Support CICS パフォーマンス・フィーチャーは、以下の機能不良と例外に対して例外レコードを作成します。

- ストレージ待ち
- 主一時記憶域待ち
- ファイル・ストリング待ち
- ファイル・バッファ待ち
- 補助一時記憶域ストリング待ち
- 補助一時記憶域バッファ待ち
- トランザクション ABEND
- システム異常終了
- 記憶保護違反
- ストレージ不足状態
- z/OS Communications Server 要求のリジェクト
- 補助一時記憶域の入出力エラー
- 区画内一時データ・セットの入出力エラー
- 自動インストール・エラー
- MXT に達しました
- IRC および ISC のリンク・エラー
- ログ・ストリーム・バッファ・フル状況
- CREAD および CWRITE 故障 (データ・スペース問題)
- ローカル共有リソース (LSR) プール (ストリング待ち)
- LSR プール内のバッファ待ち
- SMF に書き込まれるエラー
- 一時データ・データ・セットにスペースがない
- 一時データ・ストリング待ち
- 一時データ・バッファ待ち
- トランザクション再始動
- transaction class reached (CMXT) のタスクの最大数
- 伝送エラー

43 ページの図 10 に、問題レポートの例を示します。ここには、「Severity (重大度)」、「Date (日付)」、「Time (時間)」、「Terminal operator ID (端末オペレーター ID)」、「User ID (ユーザー ID)」、「Exception ID (例外 ID)」、および「Exception description (例外記述)」の情報が示されています。

CICS Incidents						
DATE: '2001-01-17' to '2001-01-18'						
Sev	Date	Time	端末 operator ID	User ID	Exception ID	Exception description
03	2001-01-17	15.42.03	SYSTEM		TRANSACTION_ABEND	CICS TRANSACTION ABEND AZTS
03	2001-01-18	00.00.00	SYSTEM		TRANSACTION_ABEND	CICS TRANSACTION ABEND APCT
03	2001-01-18	17.37.28	SYSTEM		SHORT_OF_STORAGE	CICS SOS IN PAGEPOOL
03	2001-01-18	17.45.03	SYSTEM		SHORT_OF_STORAGE	CICS SOS IN PAGEPOOL
Tivoli Decision Support report: CICS002						

図 10. IBM Z Decision Support CICS 機能不良レポートの例

IBM Z Decision Support は、例外を情報管理システムに受け渡します。

作業単位レポート

CICS 複数領域操作 (MRO) またはシステム間連絡 (ISC) 環境では、ある領域 (またはプロセッサ複合体) から別の領域へ、またはその逆に、トランザクションをトレースできます。トレースのデータを使用すると、各領域のコンポーネント・トランザクションを個別に分析せずに、1 つの作業単位として、結合トランザクションの総合リソース要件を判別できます。

MRO または ISC シリーズのコンポーネント・トランザクションを結合する機能は、高精度のリソース・アカウンティングやチャージバック、さらに容量やパフォーマンス分析を可能にします。

43 ページの図 11 の CICS UOW Response Times レポートに、IBM Z Decision Support による CICS の作業単位の応答時間の表示例が示されています。見出しは、「Adjusted UOW start time (調整済み UOW 開始時刻)」、「Tran ID (トランザクション ID)」、「CICS ID」、「Program name (プログラム名)」、「UOW tran count (UOW トランザクション・カウント)」、および「Response time (sec) (応答時間 (秒))」です。

CICS UOW Response Times						
Time: '09.59.00' to '10.00.00'						
Date: 2001-01-18						
Adjusted UOW start time	Tran ID	CICS ID	Program name	UOW tran count	Response time (sec)	
09.59.25	OP22	CICSPROD	DFHAPRT	2	0.436	
	OP22	CICSPRDC	OEPCPI22			
09.59.26	AP63	CICSPRDE	APPM00	2	0.045	
	AP63	CICSPROD	DFHAPRT			
09.59.26	ARUS	CICSPROD	DFHAPRT	3	0.158	
	CSM5	CICSPRDB	DFHMIRS			
	ARUS	CICSPRDC	AR49000			
09.59.27	CSM5	CICSPRDB	DFHMIRS	4	0.639	
	CSM5	CICSPRDB	DFHMIRS			
	MQ01	CICSPROD	DFHAPRT			
	MQ01	CICSPRDD	CMQ001			
...						
Tivoli Decision Support report: CICS902						

図 11. IBM Z Decision Support CICS UOW 応答時間レポート

可用性のモニター

場合によっては、アプリケーションは、同じまたは異なる多くのリソースの可用性に依存します。したがって、可用性のレポートには、さまざまなソースからの複雑なデータの分析が必要です。

CICS アプリケーションのユーザーは、いくつかのタイプのリソースの可用性に依存します。

- CICS 領域を実行する中央側ハードウェアおよびオペレーティング・システム環境

- ユーザーの CICS 領域へのアクセスを介する、通信コントローラー、通信回線、および端末などのネットワーク・ハードウェア
- CICS 領域
- アプリケーション・プログラムおよびデータ。アプリケーション・プログラムは、いくつかの CICS 領域間に分散する場合があります。

IBM Z Decision Support は、すべてのデータが 1 つのデータベースにあり、使いやすくなっています。

CICS ワークロード・アクティビティ報告書

CICS は、トランザクション ID、関連の端末 ID、および各トランザクションの終了時の経過時間を記録します。さらに詳細なレポートが必要な場合は、IBM Z Decision Support のシステム・パフォーマンス・フィーチャーの MVS Performance Management (MVSPM) コンポーネントを使用します。

トランザクション・データは、CMF が作成する詳細情報ではなく、トランザクション統計のみが必要な場合に役立ちます。多くの場合、RMF が SMF type-72 レコードの一部としてこのデータを記録するため、このデータを処理するだけで十分です。そして、詳細データが必要な場合には、CMF の SMF レコードの分析（および記録）を予約することができます。IBM Z Decision Support システム・パフォーマンス・フィーチャーの MVSPM コンポーネントを使用して、このデータをレポートします。

MVS 5.1.0 以降をゴール・モードで実行する場合は、CICS パフォーマンスをワークロード・グループ、サービス・クラス、および期間で報告できます。この環境には、CICS の IBM Z Decision Support レポートの例がいくつかあります。44 ページの図 12 に、別のサービス・クラスとして機能するサービス・クラスを示します。このレポートは、MVS システムがゴール・モードで実行されている場合にのみ使用可能です。見出しは、「Workload group (ワークロード・グループ)」、「Service class (サービス・クラス)」、「Served class (対象クラス)」、「No of times served (処理回数)」、「No of transactions (トランザクションの数)」、および「No of times served per transaction (トランザクションごとの処理回数)」です。

MVSPM Served Service Classes, Overview					
Sysplex: 'SYSPLEX1' System: IP02					
Date: '2001-01-18' Period: 'PRIME'					
Workload group	Service class	Served class	No of times served	No of tx's	No of times served per tx
CICS	CICSREGS	CICS-1	15227	664	22.9
		CICS-2	6405	215	29.8
		CICS-3	24992	1251	20.0
		CICS-4	87155	1501	58.1
		CICSTRX	67769	9314	7.3
Tivoli Decision Support report: MVSPM79					

図 12. サービス・クラスとして機能する MVS Performance Management 概要レポートの例

45 ページの図 13 には、平均トランザクション応答時間傾向と、トランザクション状況がその傾向にどのように関与しているかを示します（さまざまなトランザクション状況を示す時間は、トランザクション状況サンプルに基づいて計算されるため、状況ごとに費やされた時間の厳密なレコードは必要ありません）。各トランザクション状況ごとに費やされた時間（グラフの陰影領域）を追加すると、平均実行時間になります。これは、平均応答時間より短くなります（グラフ上のライン）。応答時間と実行時間の差は、主に切り替え時間になります。そうした切り替え時間の例としては、トランザクションが処理のために別の領域に送信される時間があります。

このレポートは、MVS システムがゴール・モードで実行されており、サブシステムが CICS または IMS の場合に使用可能です。

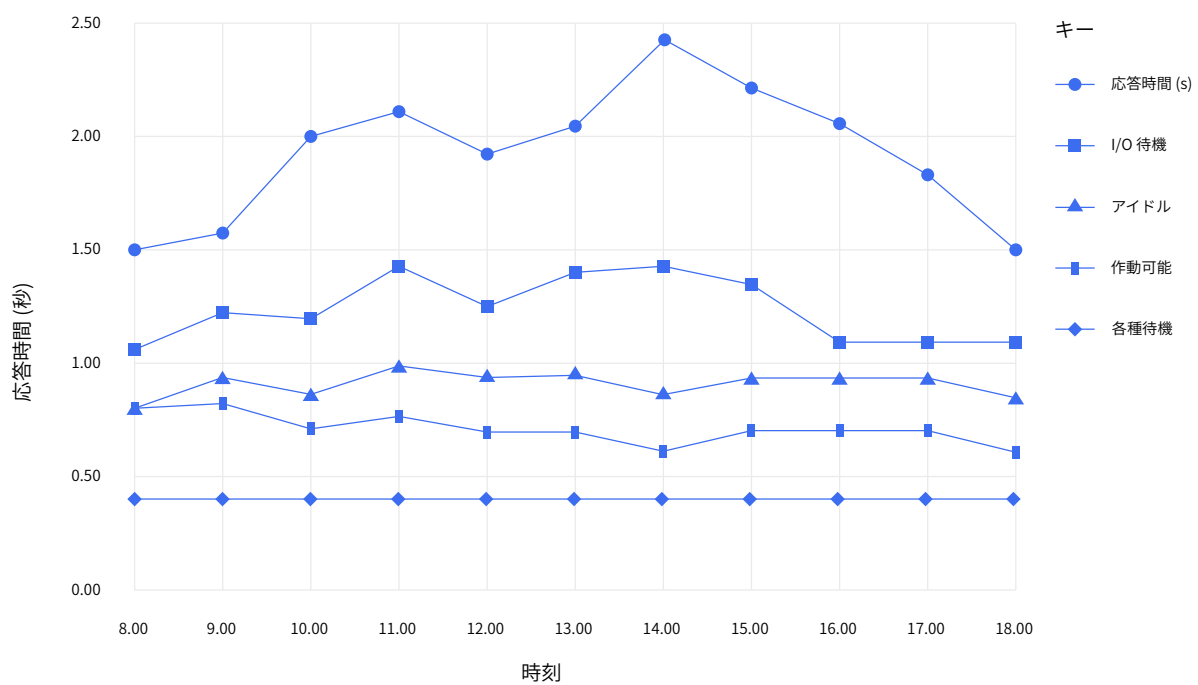


図 13. MVS Performance Management 応答時間明細、毎時傾向レポートの例

46 ページの図 14 は、さまざまなトランザクション状況が平均応答時間に与えた影響を示しています。このレポートは、MVS システムがゴール・モードで実行されており、サブシステムが CICS または IMS の場合に使用可能です。レポートは、「Workload group (ワークロード・グループ)」、「Service class/Period (サービス・クラス/期間)」、「Ph」、「MVS sys ID (MVS システム ID)」、および「Total state (合計状態)」に続いて、45 ページの図 13 にリストされた状態ごとに費やされる応答時間の割合 (パーセント) の情報を提供します。

MVSPM Response Time Breakdown, Overview																
Sysplex: 'SYSPLEX1' Subsystem: IP02																
Date: '2001-01-18' Period: 'PRIME'																
Other	Service	MVS	Total	Activ	Ready	Idle	Lock	I/O	Conv	Distr	Local	Netw	Syspl	Timer		
Workload	Misc															
wait	wait															
group	class	Ph	ID													
(%)	(%)			(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)	(%)

CICS	CICS-1	/1	BTE	CA0	6.6	0.0	0.0	0.0	0.0	0.0	6.5	0.0	0.0	0.0	0.0	0.0
0.0	0.0			C80	29.4	0.0	0.0	0.0	0.0	0.0	14.7	0.0	0.0	0.0	0.0	0.0
14.6	0.0			C90	3.8	0.4	1.3	1.5	0.0	0.2	0.5	0.0	0.0	0.0	0.0	0.0
0.0	0.0															

			*		13.3	0.1	0.5	0.5	0.0	0.1	7.2	0.0	0.0	0.0	0.0	0.0
4.9	0.0															

		/1	EXE	CA0	16.0	0.1	0.2	0.1	0.0	15.5	0.0	0.0	0.0	0.0	0.0	0.0
0.1	0.0			C80	14.9	0.1	0.1	0.1	0.0	3.7	0.0	0.0	0.0	0.0	0.0	0.0
11.0	0.0			C90	14.0	1.6	4.5	4.8	0.0	3.2	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0															

			*		14.9	0.6	1.6	1.7	0.0	7.4	0.0	0.0	0.0	0.0	0.0	0.0
3.7	0.0															

IMS	IMS-1	/1	EXE	CA0	20.7	0.4	0.7	0.0	0.0	0.0	19.6	0.0	0.0	0.0	0.0	0.0
0.0	0.0			C80	1.1	0.2	0.1	0.7	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0			C90	22.2	5.3	11.9	1.2	0.0	0.2	3.6	0.0	0.0	0.0	0.0	0.0
0.0	0.0															

			*		14.7	2.0	4.2	0.6	0.0	0.1	7.8	0.0	0.0	0.0	0.0	0.0
0.0	0.0															

Tivoli Decision Support report: MVSPM73

図 14. MVS Performance Management 応答時間の概説レポート明細の例

Tivoli OMEGAMON XE for CICS on z/OS

Tivoli OMEGAMON XE for CICS on z/OS は、複合 CICS システムのパフォーマンスおよび可用性をプロアクティブに管理するのに役立ちます。

Tivoli OMEGAMON XE for CICS on z/OS (OMEGAMON XE for CICS on z/OS) は、z/OS 管理システム上で実行されるリモート・モニター・エージェントです。パフォーマンス上の問題を予測するのを支援し、CICS 環境で重大なイベントが発生すると警告します。CICS 領域内のイベントが重大なポイントに達したときに警報するように、しきい値レベルとフラグを設定できます。

Tivoli Enterprise Portal 下で実行されている場合、IBM Tivoli OMEGAMON XE for CICS on z/OS は、CICS Transaction Server の一元管理点として機能し、CICS 領域内の問題を検出して防止するために必要な情報を収集する包括的な手段を提供します。Tivoli Enterprise Portal が収集するデータを表やグラフに表示して、管理対象の CICS 領域の状況を示します。

このデータを使用して、次のようなさまざまな作業が行えます。

- 信頼できる最新のデータを収集して分析し、より迅速に、適切な情報に基づいて運用上の決定を下せるようにする
- すべての CICS 領域を単一点から管理し、随時に問題を特定する
- さまざまな領域間でワークロードのバランスを取る
- 目標に照らしてパフォーマンスを追跡する

OMEGAMON XE for CICS on z/OS を使用すると、システム管理者はしきい値レベルおよびフラグを設定し、システムの状態がそのしきい値に達したときに警報を出すようにすることができます。以下は、拡張モニター機能です。

- しきい値に基づくユーザー定義および事前定義の状態で、異なるタイプのアラートを生成する
- すべての CICS 領域の状況を一覧表示する
- 1 つ以上の集中型ワークステーションから同時に複数の CICS 領域をモニターする機能

他の OMEGAMON XE モニター製品と一緒に使用すると、OMEGAMON XE for CICS on z/OS で提供されるデータ、分析、および警報は、単一のコンソールからコンピューター・エンタープライズ全体の全体ビューを作成するのに役立ちます。

OMEGAMON XE for Db2

Tivoli OMEGAMON XE for Db2 Performance Expert on z/OS は、単一の包括的な評価ツールです。Tivoli OMEGAMON XE for Db2 Performance Monitor on z/OS は、重大なパフォーマンス問題を解決するのに役立ちます。

Performance Expert

OMEGAMON XE for Db2 Performance Expert (PE) は、z/OS 上の Db2 環境用のパフォーマンス分析、モニター、およびチューニング・ツールです。この製品は、z/OS 上のすべての Db2 サブシステムおよびその他のリソース (IMS、MVS、CICS など) をモニターする IBM Tivoli OMEGAMON XE ファミリーの z Systems® 間モニター・ソリューションに統合され一体化されています。OMEGAMON XE for Db2 PE は、システムおよびアプリケーションのパフォーマンス・モニター、レポート作成、トレンド分析、チャージバック使用、およびバッファー・プール分析を簡素化し、サポートします。問題が検出されると、ユーザーに通知し、続行する方法を示します。

パフォーマンス・モニター

Tivoli OMEGAMON XE for Db2 Performance Monitor on z/OS では、z/OS 上の Db2 アプリケーションのパフォーマンスのモニター、分析、および最適化を、オンライン (問題が発生したときにリアルタイムで即時に警報する) およびバッチ (レポート形式) の 2 つの主要モードで実行できます。

Tivoli OMEGAMON XE for Db2 Performance Monitor on z/OS は、重大なパフォーマンス問題を解決するのに役立ちます。これを使用して、以下をモニターできます。

- 個別のデータ共用メンバー、またはデータ共用グループ全体。
- 並列照会環境で実行されるアプリケーション。並列タスクが別々のプロセッサで実行されている場合も可能です。
- 気付かないうちに進行する問題を調べ、将来的な問題を防止するための、短期的なパフォーマンス・ヒストリー。
- パフォーマンスを調整するための、データベース・ディスク、テーブル、テーブル・スペース、およびその他のエレメントのオブジェクト分析。

CICS のパフォーマンス制約の識別

CICS システム上の大きな制約は、多くの場合、ストレス状態や長い応答時間など、外部の症状によって示されます。一部の制約問題は CICS が解決できます。その他のものは手動で解決する必要があります。

過密なシステムでは、ローパフォーマンスのさまざまな症状が生じることがあります。例えば、直接アクセス・ストレージ・デバイス (DASD) のアクティビティーがスローダウンすると、次のような症状が発生することがあります。

- データ・セット・アクティビティーを実行するトランザクションが累積する
- スtring待ちが発生する
- システム内に待機状態のトランザクションが増える
- 仮想記憶に対する要求が増える
- 実記憶に対する要求が増える
- ページングの発生が増える
- タスク・ディスパッチャーが使用するプロセッサ能力のスキャン・タスク・チェーン数が増える
- タスク制約が発生する
- MXT またはトランザクション・クラス制限を超過する。必要な再試行回数が増えるため、プロセッサは追加の処理が必要になります。

その結果、システムではすべてのリソースの使用頻度が高まり、典型的なシステム・ストレス状態になります。この状態は、すべてのリソースに問題があることを示されているわけではありません。制約を見つける必要があることを示しています。制約を特定するには、タスクの存続時間に影響を与えている要因を検出する必要があります。

パフォーマンスが許容できない場合は、パフォーマンスの制約 (症状の原因) を特定して、それを調整できるようにする必要があります。

制限状態に対処している場合は、パフォーマンス制約が発生しているシステム内のさまざまなハードウェアおよびソフトウェアの場所を確認すると役立つ場合があります。

ハードウェアの競合

競合は、プロセッサ・サイクル、実記憶、データベース関連のハードウェア入出力操作、およびネットワーク関連のハードウェア操作で発生する可能性があります。

- プロセッサ・サイクル。トランザクションが 100 万より多くの命令を実行することは一般的ではありません。これらの命令を実行するには、トランザクションがシステム内の他のタスクおよびジョブと競合する必要があります。時々、これらのタスクとジョブは、ファイル入出力などの活動を待たなければなりません。トランザクションは、これらの時点でプロセッサの使用を止め、その活動が完了したら、再びプロセッサの使用のために競争しなければなりません。ディスパッチング優先順位は、どのトランザクションまたはジョブがプロセッサを使用できるかを判別し、バッチまたは他のオンライン・システムは、プロセッサへの優先アクセスの受け取りを終了するまでの応答時間に影響を与えます。オンライン・データベースにアクセスするバッチ・プログラムも、それぞれのディスパッチング優先順位が低い場合は、より長い期間を得るためにそれらのデータベースと連携します。使用量が高い場合は、プロセッサへのアクセス待ち時間はかなり長くなることがあります。
- 実記憶 (作業セット)。トランザクションは、プロセッサを得るために競合が必要であると同時に、トランザクションにはある程度の実記憶を与える必要もあります。実記憶不足は、CICS パフォーマンスにとって特に重要な場合があります。これは、実記憶を獲得する際に生じる通常のページ不在の結果が、同期入出力になるためです。CICS の基本設計は非同期です。つまり、CICS は複数のタスクからの要求を同時に処理し、プロセッサを最大限に利用します。ほとんどのページング入出力は、同期的であり、CICS が使用している MVS タスクは待機し、CICS のその部分は、ページ操作が完了するまではそれ以上の処理を実行できません。CICS 処理のすべてではありませんが、そのほとんどは単一の MVS タスク (ディスパッチャー統計では「QUASI」と呼ばれる) を使用します。
- データベース関連ハードウェア (入出力) 操作。トランザクションで必要な情報を提供するためにデータにアクセスしている場合、入出力操作は、プロセッサ、プロセッサ・チャンネル、ディスク制御装置、一連のディスク列のヘッド、およびデータが常駐する実際のディスク装置を経由します。これらの装置のいずれかを使用しすぎている場合は、データへのアクセス時間が著しく増加することがあります。このような過度の使用は、1つのデータ・セットまたはアクティブなデータ・セットの組み合わせに対するアクティビティーの結果です。エラー率も装置の使用量およびパフォーマンスに影響します。共用 DASD 環境では、プロセッサ間の競合もパフォーマンスに影響します。これによって今度は、トランザクションが実記憶域と仮想記憶域、および他のリソースと連携する時間が増加します。

大容量の中央ストレージや拡張ストレージ、および非常に大きなデータ・バッファを使用し、プログラムをストレージ内に保持すると、DB 入出力の競合を大幅に削減し、プロセッサ使用率を幾分減らすことができると同時に、内部応答時間に関して大きな利点が得られます。

- ネットワーク関連ハードウェア操作。トランザクションの入力および出力メッセージは、端末から制御装置、通信リンク、ネットワーク・コントローラー、プロセッサ・チャンネル、および最後にプロセッサを通過する必要があります。データにアクセスするために装置を使いすぎると応答時間に影響を与えるのと同様に、ネットワーク・リソースを使用しすぎるとパフォーマンスが低下することがあります。エラー率もパフォーマンスに影響します。場合によっては、出力メッセージの送達は、アクセスされているプロセッサ・リソースを解放するための前提条件であり、競合によってこれらのリソースが長期間連携することになる場合があります。

設計上の考慮事項

データ・セットの再編成と次の再編成の間の時間がパフォーマンスに影響することがあります。アクセスの効率は、データ・セットのフラグメント化が進むにつれ減少していきます。データ・セットの再編成と次の再編成の間の時間を短くすることにより、フラグメント化を最小に保つことができます。

パフォーマンスを制限する可能性がある要因には、以下のものがあります。

- データベース設計。データ・セットまたはデータベースは、それがサポートするアプリケーションの要求に合うように設計する必要があります。データ・セットへのアクセス・パターン (特に、パターンがランダムなのか順次なのか)、選択したアクセス方式、およびアクセスの頻度などの要因によって、最適なデータベース設計が決まります。物理レコードのサイズ、ブロック化因数、代替索引または2次索引の使用、データベース・セグメントの階層構造またはリレーショナル構造、データベースの編成 (HDAM、HIDAM など)、およびポインター配列などのデータ・セット特性はすべて、データベース・パフォーマンスの要因です。
- ネットワーク設計。この項目が応答時間の支配的要因になることがよくあります。それは、ネットワーク・リンクは、オンライン・システムのほとんどのコンポーネントよりもずっと低速であるためです。プロセッサ速度はナノ秒で測定され、回線速度は秒で測定されます。画面設計も、全体の応答時間にかなりの影響を与えることがあります。1200 バイトのメッセージを伝送するのに、比較的高速である 9600 ビット/秒のリンクで 1 秒かかります。このメッセージの 600 バイトが不要である場合は、その半分の応答時間は無駄になります。画面設計およびサイズの他に、回線上の端末の数、使用されるプロトコル (SNA、2 進同期)、および全二重または半二重機能などの要因がパフォーマンスに影響することがあります。
- 特定のソフトウェア・インターフェースまたはシリアル機能。オペレーティング・システム、端末アクセス方式、データベース・マネージャー、データ・セット・アクセス方式、および CICS はすべて、トランザクションの処理中に通信する必要があります。一度に特定のレベルの同時処理のみが発生することがあり、これもパフォーマンス制約の原因になることがあります。同時処理の例としては、SNA 任意受信プール (RAPPOOL)、VSAM データ・セット・アクセス (ストリング)、CICS 一時記憶域、CICS 一時データ、および CICS 相互通信セッションがあります。これらのそれぞれが、トランザクションの応答時間に対して単一またはマルチサーバー・キューイングの影響を及ぼし、タスクのスループットを低下させて、他のリソースと連携することができます。

SNA を使用する CICS システムのパフォーマンス制約を特定するための便利な手法の 1 つに、ユーザーの端末から発行される IBMTEST コマンドを使用する方法があります。この端末は、CICS とセッション中であってはなりませんが、z/OS Communications Server for SNA に接続されている必要があります。

SNA LU で、次のように入力します。

```
IBMTEST (n)(,data)
```

ここで、*n* は、データをエコー出力する回数で、*data* は、任意の文字ストリングで構成します。データを入力しなかった場合は、アルファベットおよび 0 から 9 の数字が端末に戻されます。このコマンドに対しては、SNA LU が応答します。

IBMTEST は、端末応答時間の内の z/OS Communications Server 部分についての概略の考え方をユーザーに提供するように設計されたエコー・テストです。応答時間の遅いシステムで、応答時間が高速の場合、z/OS Communications Server の部分は制約になる可能性はありません。応答時間が遅い場合は、z/OS Communications Server または SNA ネットワークに原因がある可能性があります。この種の演えきのプロセスは、一般的に、制約の分離に役に立つことがあります。

CICS とのセッションに入らないようにするには、LU ステートメントから APPLID= を除去するか、TERMINAL 定義から CONNECT=AUTO を除去する必要があります。

応答時間の観察

実動システムにおけるパフォーマンスの基本的な基準は応答時間です。良好なパフォーマンスは、ユーザー要件、使用可能な能力、システムの信頼性、およびアプリケーションの設計など、さまざまな要因によって異なります。あるシステムでの良好なパフォーマンスは、別のシステムではローパフォーマンスであることもあります。

簡単なデータ入力システムでは、優れた応答時間とは、暗黙で応答時間がサブミリ秒であることを意味します。通常の実動システムでは、良好な応答時間は、5 ミリ秒から 10 ミリ秒の範囲内にあります。科学計算に特化したシステム、または印刷システムでは、良好な応答時間は 1 分から 2 分になることもあります。

CICS システムのパフォーマンスが、システムの期待されている能力または要求されている能力に合致しているかどうかを確認する場合、この調査は、インストールに存在するハードウェア、ソフトウェア、およびアプリケーションに基づいている必要があります。

例えば、アプリケーションがデータベースへのアクセスを 100 回必要としている場合、応答時間が 3 ミリ秒から 6 ミリ秒であれば、かなり良好であると考えられます。ただし、アプリケーションが必要としているアクセスが 1 つだけの場合は、ディスク・アクセスの応答時間が 3 ミリ秒から 6 ミリ秒では、調査する必要があるでしょう。ただし、応答時間は、プロセッサの速度、および実動システムで実行されているアプリケーションの性質に依存します。

また、応答時間がどの程度一貫しているのかも注意して見る必要があります。変動が急激な場合は、異常なシステム動作を示しています。

通常、システムの応答時間はトランザクション比率の増加につれて変化し、最初はゆっくり、次に急速に低下します。典型的な曲線は、トランザクション比率の比較的小さな増加に対して、応答時間が突然劇的に増加するときに鋭い変化を示します。

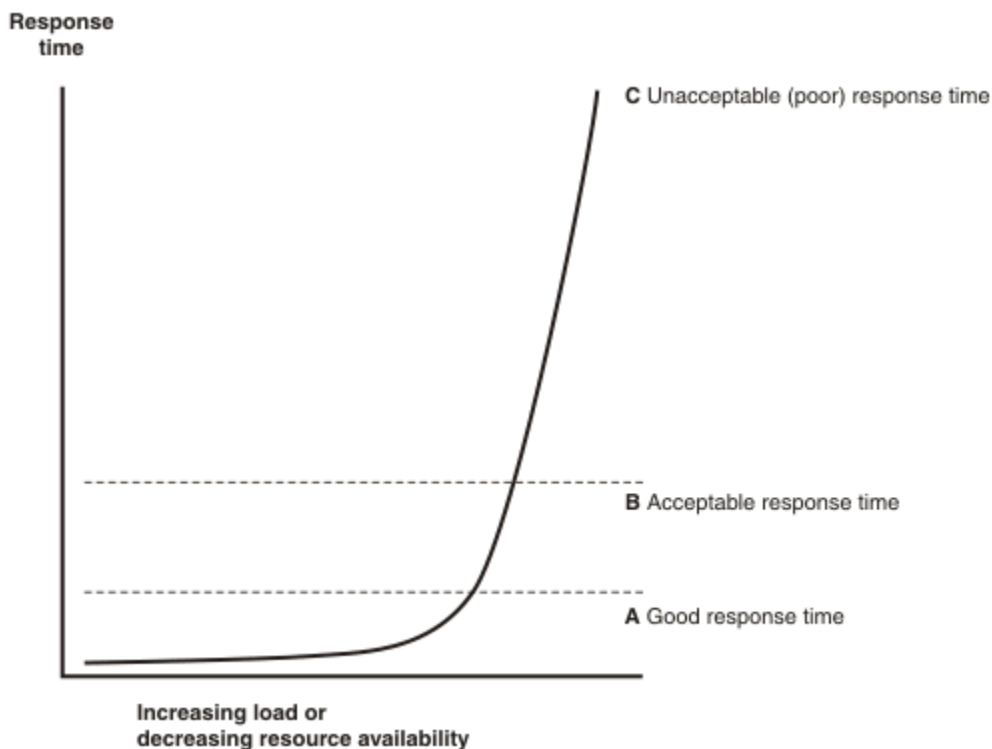


図 15. 負荷の増加に対する応答時間の影響を示したグラフ

安定したパフォーマンスを得るには、応答時間が劇的に増加するこの点よりも低くシステムの動作を保っておく必要があります。これらの環境では、DP 部門が担当している調整アクティビティーによって、ユーザー・コミュニティが深刻な影響を受ける可能性は低いので、これらの変更は、急がずに制御された方法で実行することができます。

応答時間は、キュー時間およびサービス時間で構成されていると考えることができます。サービス時間は一般には使用量には依存しませんが、キュー時間はそうではありません。例えば、使用量が 50% の場合、キュー時間はサービス時間にほぼ等しいと考えられ、使用量が 80% の場合、キュー時間はサービス時間の約 4 倍と考えられます。特定のシステムのサービス時間がシステム 応答のわずかな部分しか占めていない場合 (例えば、それがプロセッサの一部である場合)、80% の使用量は容認できる場合があります。これが、例えば通信回線におけるシステム 応答時間の比較的大きな部分を占めている場合は、50% の使用量は高いと考えられます。

端末から端末への応答時間を知ろうとする場合、ホストで実行しているいかなる手段またはツールから取得できる、最も一般的な「応答時間」は、「内部応答時間」であることを認識しておく必要があります。トレースが識別できるのは、ホスト内のソフトウェア、すなわち CICS およびその付随ソフトウェアが最初にインバウンド側でメッセージを「見た」とき、およびそれが最後にアウトバウンド側でメッセージを「見た」ときだけです。

内部応答時間からは、メッセージが、端末の制御装置、任意の速度の回線、(任意の) 通信コントローラーを経由し、(任意の) 通信アクセス方式を介して端末から取り出されるのにどれくらい時間がかかったのか、および読み取りを開始したチャンネル・プログラムが最終的に CICS に通知される前の遅延時間については、何も得ることができません。この場合、CICS がこの入力メッセージの処理を開始するのにかかる時間は含まれていません。端末管理が制御を再度取得する前、および端末管理がこの通知済みイベントを見つける前でさえ、CICS には行うべき作業が多く存在していた可能性があります。

同じことが、アウトバウンド側にも当てはまります。CICS 補助トレースは、いつアプリケーションがその要求を発行したかを認識していますが、それは、いつ端末管理がその要求を見つけたか、いつアクセス方式がそれをシップするか、いつコントローラーが装置に接続できるか、などとはほとんど関係がありません。

ローパフォーマンスの表面上の症状は、全体的に応答がよくないというものですが、累進的な初期の警告状態のセットを正しく解釈すれば、制約を見つけてそれを除去するという問題は簡単になります。

このトピックの情報は、CICS がシステムで実行されている唯一の主要プログラムであるという前提に基づいています。バッチ・プログラムまたはその他のオンライン・プログラムが CICS と共に同時に実行している場合は、CICS が、システム・リソースを均等に割り当てられていること、および他の領域からの干渉によって CICS のパフォーマンスが著しく低下していないことを確認してください。

遅い応答時間: 原因と解決策

次の表は、応答時間の 4 つのレベルを、重大度の降順に示しています。レベルごとに、主要原因を、提案しているソリューションの範囲と共に示しています。

最初のステップでは、[18 ページの『システムのパフォーマンスの査定』](#)で与えられているアドバイスに従って、原因をチェックします。正確な原因が特定できたら、適切な解決策を実装する方法について、[61 ページの『第 2 章 CICS システムのパフォーマンスの改善』](#)で情報を見つけることができます。

表 2. CICS 応答時間チェックリスト	
主要原因	解決策
レベル 1: すべてのトランザクションのすべての負荷で応答が遅い	
ページングのレベルが高い	作業セットを縮小するか、またはより多くの実記憶域を割り当てます。
主要リソースの使用量が非常に多い	システム・リソースの要件を再検討し、システムを再設計して、アプリケーション・エラーやリソースの競合がないか確認します。
レベル 2: 中および高負荷での低応答	
ページングのレベルが高い	作業セットを縮小するか、またはより多くの実記憶域を割り当てます。
プロセッサ使用量が多い	パス長さを短くするか、プロセッサの能力を上げます。

表 2. CICS 応答時間チェックリスト (続き)	
主要原因	解決策
DB またはデータ・セットの使用量が多い	データ・セットを再編成するか、データ転送を減らすか、能力を上げます。
通信ネットワークの使用量が多い	データ転送を減らすか、能力を上げます。
TP または入出力方式の制約	バッファの可用性を大きくします。
CICS 限界値を超える	オペランドを変更する、またはより多くのリソースを提供する、またはアプリケーションにエラーがないかチェックする。
レベル 3: 特定のトランザクションのみ低応答	
レベル 2 にリストされたものと共通の特性を示す	解決策はレベル 2 の場合と同じです。
回線または端末の使用量	能力を上げるか、データ転送を減らす、またはトランザクション論理を変更します。
データ・セット使用量	データ・セット配置バッファの割り振りを変更するか、エンキュー論理またはデータ・セット設計を変更します。
ストレージの使用量が多い	再設計するか、またはアプリケーションを調整します。
複数のトランザクションが同一のサブプログラムを使用する	再設計するか、またはアプリケーション・サブプログラムを調整します。
複数のトランザクションが同一のアクセス方式または CICS 機能を使用する	リソースを再割り振りするか、アプリケーションを変更して、問題の機能の使用を再評価します。
制限状態	リソースを再割り振りするか、またはアプリケーションを変更します。
レベル 4: 特定の端末が低応答	
適宜にネットワーク・ロードをチェックする	ネットワークの該当部分の能力を上げます。
オペレーター手法をチェックする	端末手順を修正します。
端末定義をチェックする	端末定義を再定義します。

ストレージ・ストレスの削減

ストレージ・ストレスは、いずれかの動的ストレージ域でフリー・スペースが不足しているときに発生します。

ストレージ・ストレスは、以下の状態の徴候である可能性があります。

- その他のリソース制約が原因で、通常よりも長い時間にわたって CICS タスクがストレージを占有する
- 突然に多数のタスクが生じて、使用可能なフリー・ストレージでは対応できなくなる
- アプリケーションの設計が悪く、必要なストレージ容量が極端に多い

CICS は、以下の方法でストレージ・ストレスを扱います。

- 使用されていない非常駐プログラムは、フリー・ストレージの可用性が減少すると、CICS が適切と判断したときに、最低使用頻度のものから徐々に削除されることがあります。新規タスクのディスパッチも、フリー・ストレージが危険なほど小さな量に近づくと、次第に速度が低下していきます。この自己調整アクティビティは、ストレージの管理コストを分散する傾向があります。全体としてのプログラムのロードはさらに多いかもしれませんが、プログラムの完全圧縮という重いオーバーヘッドは、重要な時間帯には発生しません。

- プログラムのロードおよび再ロードは MVS サブタスクで CICS が処理します。これによって、MVS イメージのプロセッサが使用可能な場合は、プログラム・ロードの一部としてページインが必要な場合でも、他のユーザー・タスクの処理を続行できます。
- ユーザーによるストレージ使用の実行時制御は、最大タスク仕様 (MXT) およびトランザクション・クラス制限を適切に使用することにより実現されます。ストレージに対する制限のない要求から生じるストレージ不足の状態を回避するには、これが必要です。

ストレージ不足の状態

CICS は、使用されていない非常駐プログラムがすべて削除された後でも、無条件 GETMAIN 要求を満たすのに十分なフリー・ストレージが存在していない場合にもみ使用するための、最低限のフリー・ストレージ・ページを予約しています。

ストレージ要求によって、動的ストレージ域の 1 つにおける連続した空きページの数それぞれのクッション・サイズよりも小さくなった場合、またはストレージ・クッションでも満足されない場合はいつも、クッション・ストレス状態が発生しています。詳細は、ストレージ・マネージャー統計 (「Times request suspended (要求が中断された回数)」, 「Times cushion released (クッションが解放された回数)」) に示されます。CICS は、多数のアクションによって、ストレージ・ストレス状態の軽減を試行します。これらのアクションが状態を軽減できない場合、またはストレス状態の原因が SOS のために中断しているタスクによるものである場合は、ストレージ不足状態が通知されます。これには、メッセージ DFHSM0131、DFHSM0133、または DFHSM0606 が伴います。

不要データ・セット名ブロックの除去

拡張 CICS 動的ストレージ域 (ECDSA) は、データ・セット名 (DSN) ブロックにも使用されます。CICS ファイル制御がオープンするデータ・セットごとに 1 つの DSN ブロックが作成され、ウォーム・リスタートまたは緊急再始動時にリカバリーされます。アプリケーションが、多数の一時データ・セット (すべて固有名を持つ) を作成した場合、DSN ブロックの数が、ストレージ不足の状態を引き起こす範囲まで増えることがあります。

作成されたすべてのデータ・セットが異なる名前を持つ、複数の一時データ・セットをアプリケーション・プログラムが使用する場合、これらの一時データ・セットは使用後にアプリケーション・プログラムによって削除されることが重要です。このコマンドを使用して CICS 領域から不要な一時データ・セットを除去する方法については、『[SET DSNAME](#)』を参照してください。

Language Environment® の、AMODE(24) プログラム用のランタイム・オプション

CICS の 2 つのデフォルトの Language Environment ランタイム・オプションは ALL31(ON) および STACK(ANY) です。つまり、Language Environment が使用可能になっている場合、Language Environment を必要とするプログラムはすべて、31 ビットのストレージをアドレッシングできる必要があります (つまり AMODE(31) である必要があります)。AMODE(24) プログラムを Language Environment の環境で実行するには、ALL31(OFF) および STACK(BELOW) を指定できます。ただし、すべてのプログラムがこれらのオプションを使用できるようにするために、これらのオプションをグローバルに変更した場合、大量のストレージが 16 MB 境界よりも下に配置され、これによりストレージ不足の状態が起こることがあります。

詳しくは、[92 ページの『動的ストレージ域のストレージ不足状態』](#)を参照してください。

タスクのページ

CICS タスクがその DTIMOUT 値よりも長い間中断している場合、RDO トランザクション定義で SPURGE=YES が指定されている場合はそのタスクをページできます。すなわち、このタスクは異常終了して、そのリソースは解放されます。これにより、他のタスクでそれらのリソースを使用できるようになります。このようにして CICS は、ストレージで事実上デッドロックになっている状態を解決しようとしています。

タスクのページが不可能な場合、または問題を解決しない場合、CICS は処理を停止します。その後、CICS 領域を取り消して再始動する必要があります。

DASD ページング・アクティビティーの削減

大量の DASD ページング・アクティビティーが生じると、トランザクションがシステムをパススルーする速度が低下することがあります。

ページングについて

プロセッサの仮想記憶域は、構成内で使用可能な中央ストレージのサイズをはるかに超える可能性があります。超過分はすべて補助記憶域 (DASD) に保守する必要があります。この仮想記憶域は、ページと呼ばれるアドレスのブロック内にあります。仮想記憶域の最近参照されたページだけが、物理中央ストレージのブロックを占有するよう割り振られます。中央ストレージにない、仮想記憶域のページが参照された場合は、そのページが DASD から取り込まれ、使用されていない、最低使用頻度のページが置き換えられます。

新規に参照されたページは、ページインされたといわれます。置き換えられたページが変更されていた場合は、ページアウトされる必要があります。

まず最初に注意する必要があるのはページイン率です。これは、ページイン・アクティビティーは同期的に発生するからです (すなわち、MVS タスクは、ページ不在が解決されるまで停止します)。ページアウト・アクティビティーは CICS 処理と並行して実行されているので、CICS のスループットにはあまり影響しません。

DASD からのページインの場合は、物理的入出力のための時間コストが発生し、プロセッサの使用量が増加します。

したがって、余分な DASD ページイン・アクティビティーは、トランザクションが CICS システムを通過する速度を低下させます。つまり、トランザクションは CICS を通過するのにより時間がかかり、CICS でオーバーラップするトランザクションが増え、そのため、さらに多くの仮想記憶域および物理ストレージが必要になります。

パフォーマンス上の問題が、過度のページングに関連していると思われる場合は、RMF を使用して、ページング率を取得できます。

CICS で MXT およびトランザクション・クラス制限を使用して、CICS のスループットを制御してください。この場合、並行トランザクションの数が少ないほど、必要とされる実記憶域や、発生するページングが少なくなり、トランザクションの数がより多い場合よりも高速に処理される、ということを基本においてください。

CICS システムが、トランザクション分離をアクティブにして稼働している場合、ストレージは 1 MB の倍数でユーザー・トランザクションに割り振られます。つまり、トランザクション分離を有効にした CICS システムの仮想記憶の所要量は非常に大きくなります。これが直接、接触のあった 4K バイト・ページにのみ影響を与えるページングに影響するわけではありません。ただし、ELSPA ではさらに多くの実記憶域が必要になります。トランザクション分離および実記憶域について詳しくは、[135 ページの『トランザクション分離を使用する場合の実記憶域の割り振り』](#)を参照してください。

DASD からの CICS ページング率は、どのくらいが理想的でしょうか。毎秒 1 ページイン未満が、CICS 領域のスループット能力を最大にするのに最適です。毎秒 5 ページイン未満であれば、おそらく受け入れ可能でしょう。最大 10 ページインが許容限度でしょう。毎秒 10 ページが限界で、それ以上になると、おそらく重大な問題になります。CICS のパフォーマンスは、ページングに関係する待機の影響を受けるがあるので、ページングは毎秒 5 ページから 10 ページを超えないようにする必要があります。

注: DASD からのページングに対する CICS システムの感度は、トランザクション比率、プロセッサのロード、および CICS タスクの内部存続期間の平均によって異なります。継続率、つまりアワー・オン・アワー率が毎秒 5 ページでも多すぎるシステムもあります。ピークのページングが 10 秒以上と見積もることができる場合、または、そのように設定すると容易にその数字の 4 倍になる可能性がある場合には特にそうです。

さまざまなプロセッサに対してどの程度のページング率であれば、過大といえるのでしょうか。また、これらのページング率はオペレーティング・システムによって異なるのでしょうか。過大なページング率は、アプリケーションに過大な遅延を引き起こす率と定義する必要があります。優先順位の高いページング監視プログラムが命令を実行し、アプリケーションにプロセッサを待機させることによって発生する影響は、アプリケーションの全遅延に関する限り、おそらく小さな考慮事項にすぎません。DASD 装置で

の待機が、全遅延の主要部分です。つまり、高ページング率というペナルティーは、プロセッサ・タイプとはほとんど関係がないことを意味します。

大量の中央ストレージを持つ潜在能力を、より多くのデータおよびプログラムをメモリーに保持することによって活用している場合は、CICS システムは、通常、はるかに良好な応答時間を、適度のプロセッサ使用率で提供することができます。

プログラムのロードおよびページング

CICS は、MVS サブタスクの下で MVS ロードを使用してプログラムをロードします。これにより、MVS のライブラリー検索機能を使用して、MVS 制御されたデータ・スペース内にプログラムのコピーを保持することにより、ほとんどの DASD 入出力をなくすることができます。

ページイン操作はページが取り出されるまで、それを必要としている MVS タスクを停止させます。ページを DASD から取り出す場合、これはかなり大きな影響を及ぼします。ページを取り出すことができる場合、その影響は比較的小さく、プロセッサ使用量がわずかに増加する程度です。

CICS ストレージへのプログラムのロードが、ページインの主要原因になることがあります。これは、CICS のメイン・アクティビティから切り離されたサブタスクの下で実行されるので、そのようなページインによって他の CICS アクティビティが停止することは、ほとんどありません。

リソース競合の削減

ストレス状態は、特定の制限条件に達しており、追加処理が必要であることを示します。関与するトランザクションは、リソースが解放されるまで待機する必要があります。

CICS システムで発生する可能性のある、主となる制限状態または制約には「[47 ページの『CICS のパフォーマンス制約の識別』](#)」でリストした項目が含まれます。

要約すると、制限状態は以下によって示すことができます。

- ・ 仮想記憶域の状態 (ストレージ不足または SOS)。CICS ストレージ・マネージャー統計内のこの項目は、CICS 領域への仮想記憶域スペースの割り振りに不足が生じていることを示しています。

ほとんどの環境では、より多くの仮想記憶域を割り振っても、本来はパフォーマンスが低下することはありません。何らかのエラー形態によってこの状態が引き起こされている場合は、その理由を突き止める必要があります。これには、ストレージ (一時記憶域を含む) を解放するためのアプリケーションの障害、プログラムまたはマップの不要な複数コピー、記憶保護違反、およびプログラムまたはハードウェアのエラーによって引き起こされた非常駐例外ルーチンの活発なアクティビティが含まれることがあります。

新規のアプリケーションはすべて、16MB 境界よりも上で実行するように作成する必要があります。

16MB 境界よりも上の動的ストレージ域は、31 ビット・アドレッシングの限界である 2GB 制限まで拡張することができます。16MB 境界よりも下の動的ストレージ域は、領域サイズ (16MB 未満) 未満に制限されています。

- ・ 到達した同時タスクの数 (MXT およびトランザクション・クラス制限) (トランザクション・マネージャー統計で示されます)。
- ・ 使用された z/OS Communications Server の任意受信 RPL の最大数 (z/OS Communications Server 統計で示されます)。
- ・ VSAM データ・セットのストリング待機や関連する状況 (ファイル制御統計で示されます)。

制限状態が発生する頻度をチェックします。一般には、次のようになります。

- ・ 制限状態が発生していない場合は、リソースを割り当てすぎです。リソースが高価でない場合、これは受け入れ可能ですが、リソースを割り当てすぎており、他でさらに使用される場合はそうではありません。
- ・ 制限状態が稀にしか発生しない場合は、特定のリソースの使用量は適切です。これは通常、システムが正常であることを示しています。
- ・ 頻繁に発生する場合 (トランザクションの 5% 以上)、通常これは、より明確な、ローパフォーマンスの徴候を回避するための処置を施す必要のある問題を、直接または間接的に示しています。頻度が約 10% よりも大きい場合は、迅速に何らかの処置を施す必要があります。それは、CICS それ自体が取る処置 (動的プログラム・ストレージ圧縮、ストレージ・クッションの解放など) は、パフォーマンスに対してかなりの影響を与えることがあるからです。

独自の処置には、以下のものがあります。

- エラーのチェック
- 他の領域に対してパフォーマンスを低下させない場合、制限の引き上げ
- 競合を除去するための、リソースのさらなる割り振り
- 競合に対するリカバリー使用のチェック

リソース問題の解決

この表では、リソース問題を示す症状、その原因、および解決策に関する情報を提供します。

以下のリソース問題を解決するための一般手順に従ってください。

1. 詳細なパフォーマンス分析によって、制約のタイプの診断が正しいことを確認してください。19 ページの『パフォーマンス分析の方式』では、さまざまな手法について説明しています。
2. パフォーマンス調整の一般的なアドバイスについては、24 ページの『システムの調整』をお読みください。
3. さまざまなソリューションの適用に関する詳細は、61 ページの『第 2 章 CICS システムのパフォーマンスの改善』の関連セクションを参照してください。
4. 仮想記憶域の活用を改善します。これには、以下が必要です。
 - 16 MB 境界よりも上またはハイパースペース記憶域内の、大きなデータ・バッファ
 - 16 MB 境界よりも上で実行するプログラム
 - 仮想記憶域の活用をサポートするための大量の実記憶域

このようなシステムは、DASD 入出力制約を最小化し、プロセッサ使用率を減らしながら、良好な内部応答時間を与えることができます。

代表的なリソース問題、その症状、および解決策

問題	症状	解決策
過度の DASD 入出力操作。DASD ストレージからモジュールを見つけて取り出すために必要な入出力操作の回数が過剰である。	<ul style="list-style-type: none">• 低応答時間 (応答時間の長さは、入出力操作の数によって異なり、バッチ・モードがアクティブの場合は応答時間は長くなります)。• DSA 使用率が高い。• ページング率が高い。• 頻繁に MXT 制限に到達してしまう。• SOS 状態が頻発する。	<ul style="list-style-type: none">• 入出力操作の数を減らす。• 残りの入出力操作を調整する。• Balance で、入出力操作の負荷のバランスをとる。
ネットワークのトランザクション応答時間が遅い。ネットワークの平均トランザクション応答時間が許容できないほど遅い。	<ul style="list-style-type: none">• 応答時間が遅い。• 回線上でアクティブになっている端末がほとんどない場合の応答は良好であるが、その回線で多くの端末がアクティブになっている場合は応答が遅い。• 内部応答時間と端末応答時間の差が大きい。	<ul style="list-style-type: none">• 回線使用率を削減する。• データ転送の遅延を小さくする。• ネットワークを変更する。

問題	症状	解決策
リモート・システムからの応答が遅い。接続されたリモート・システムからの応答時間がひどく遅い。	<ul style="list-style-type: none"> 接続領域に問題がある場合、SOS 状態または MXT が発生する。 問題が修正されているのに、CICS のリカバリーに時間がかかる。 	<ul style="list-style-type: none"> リモート・システムとの接続を使用するために発生するキューイングの量を制御する。 リモート・システムの応答時間を改善する。
仮想記憶域の過剰な使用。共通ストレージの過剰な使用が生じているか、ジョブまたはアドレス・スペースの終わりにストレージが解放されていない。	<ul style="list-style-type: none"> 応答時間が遅い。 同一プログラムが複数ロードされている。 プログラム・ライブラリーに対する入出力操作が増加する。 ページング率が高い。 頻繁な SOS 状態。 	<ul style="list-style-type: none"> CICS 用の仮想記憶域をより多く獲得するよう MVS システムを調整する (領域サイズを増やす)。 動的ストレージ域をより効率的に活用する。
不十分な実記憶域。プログラムが実記憶 (プロセッサ) に対し、可変長および高すぎる最大値を指定した要求を発行した。	<ul style="list-style-type: none"> ページング率が高い。 応答時間が遅い。 頻繁に MXT 制限に到達してしまう。 SOS 状態が頻発する。 	<ul style="list-style-type: none"> 実記憶に対する要求を削減する。 CICS 用の実記憶をさらに多く獲得するよう MVS システムを調整する。
過度のプロセッサ・サイクル。統合チャンネルでストレージ・バッファリングまたはサイクル・スチーリングが発生しているか、キュー検索の量が過度である。	<ul style="list-style-type: none"> 応答時間が遅い。 優先順位の低いトランザクションの応答が、非常に低速である。 優先順位の低い処理の完了が非常に遅い。 	<ul style="list-style-type: none"> CICS のディスパッチング優先順位を上げる。 オペレーティング・システムのジョブの相対的な優先順位を再評価する。 MVS 領域の数を減らす (バッチ)。 生産作業のためのプロセッサ使用率を減らす。 実際に必要な CICS 機能のみを使用する。 使用されていないトレースをすべてオフにする。 トレースの有効範囲を縮小するか、トレースの頻度を少なくして、トレースされるデータを最小化する。 高速のプロセッサを使用する。

パフォーマンス上の問題の解決について詳しくは、「[z/OS リソース測定機能 \(RMF\) ユーザーズ・ガイド](#)」を参照してください。

記憶保護違反の削減

CICS がストレージ保護およびトランザクション分離を使用可能にしている場合は、記憶保護違反を減少させることができます。

CICS が記憶保護違反を検出する可能性があるのは、重複ストレージ・アカウントिंग域 (SAA) または TIOA ストレージ・エレメントの初期 SAA が破壊された場合、またはユーザー・タスク・ストレージの先頭のストレージ・チェック・ゾーンまたはその末尾のストレージ・チェック・ゾーンが破壊された場合です。

記憶保護違反は、次の状態で発生することがあります。

- CICS が、TIOA ストレージ・エレメントに対する FREEMAIN 要求を通常処理しているときにエラーを検出し、重複する SAA の 2 つのストレージ・チェック・ゾーンおよび初期 SAA が同一でないことが判明した場合。
- CICS が、FREEMAIN コマンドの後に、ユーザー・タスク・ストレージの、あるエレメントのストレージ・チェック・ゾーンをチェックして、ユーザー・タスク・ストレージが関係するユーザー違反も検出した場合。

記憶保護違反が検出されると、内部トレース・テーブルに例外トレース項目が作成されます。メッセージ (DFHSM0102) が発行され、ダンプ・オプションがオンになっている場合は、その後に CICS システム・ダンプが実行されます。

記憶保護違反について詳しくは、[記憶保護違反の処理](#)を参照してください。

パフォーマンス管理およびキャパシティー・プランニング

パフォーマンス管理は、サービス・レベル・アグリーメント (SLA) または非公式のサービス目的に従って、既存のデータ処理リソースをモニターし、アプリケーションに割り振ることを意味します。キャパシティー・プランニングは、すべてのユーザーに対して将来のサービスのニーズを満たすように、コスト効率のよい方法で、十分なコンピューターのキャパシティーを計画するプロセスです。

パフォーマンス管理

パフォーマンス管理の目標は、現在のリソースを最大限に活用し、過剰な調整作業を行わずに現在の目的を達成することです。目的を定式化するために、サービス・レベル・アグリーメント (SLA) をセットアップできます。SLA は、次のような測定可能なパフォーマンス要因を客観的に記述した契約です。

- ネットワーク、入出力、プロセッサまたは全体の平均トランザクション応答時間
- トランザクション量
- システム可用性

パフォーマンス管理の基本的な部分は、トランザクション応答時間を測定し、コンポーネント別に内訳を出すことです。このプロセスにより、個々のトランザクションで調整を実行できる箇所がわかります。効果的なパフォーマンス管理のために、ワークロード・レベルでのリソース要件の測定に取り掛かる必要があります。ワークロードを分析することは、システムの動作や、ワークロードがどのように相互作用するのかを理解するうえで役立ちます。

パフォーマンス調整や管理に関する以下の IBM 製品資料を入手できます。

- [z/OS Resource Measurement Facility \(RMF\) ユーザーズ・ガイド](#)
- [z/OS MVS 初期設定およびチューニングガイド](#)
- [z/OS MVS 計画: ワークロード管理](#)

キャパシティー・プランニング

キャパシティー・プランニングでは、次のような問題を取り上げます。

- コンピューター・リソース (プロセッサ、ストレージ、入出力ネットワーク) のどのくらいの量が使用されているか。
- どのワークロードがリソースを消費しているか (ワークロードの配分)。
- 予想される成長率はどのくらいか。

- 現在のリソースの要求量がサービス・レベルに影響を与える時期はいつか。

収集するデータや行う予測は、z Systems ハードウェアのアップグレード・スケジュールの計画や、追加の機能拡張 (zIIP および zAAP 専用プロセッサのシステムへの追加など) をするのに役立ちます。

キャパシティー・プランニングについて詳しくは、IBM Redbooks 資料「[ABCs of z/OS System Programming](#)」(SG24-6327-01) を参照してください。

CICS トランザクションとハードウェア・リソースの関連付け

CICS モニターおよび統計で得られる情報を使用して、システム内のどのハードウェア・リソースが CICS トランザクションによって使われているかを把握します。キャパシティー・プランニング、および会計・請求処理のためにこのデータを使用できます。

このタスクについて

各 CICS トランザクションの SMF モニター・レコードは、CICS 領域が稼働している物理ハードウェア環境の CEC マシン・タイプと CEC 型式番号を識別します。CEC (中央電子処理装置) は CPC (中央演算処理複合システム) の同義語として広く使用されており、主ストレージ、1 つ以上の中央処理装置、タイマー、およびチャンネルを含む物理ハードウェアの集合を表します。さらに多くのモニター・フィールドを使用すると、トランザクションが zIIP または zAAP 専用プロセッサで費やすプロセッサ時間を計算できます。また、トランザクションが専用プロセッサで費やすことができた潜在的なプロセッサ時間も表示できます。

手順

- CICS ワークロードをシステム内の特定の CPC に関連付けるには、DFHTASK パフォーマンス・クラス・グループ内の CECMCHTP フィールドおよび CECMDLID フィールドの情報を使用します。

CECMCHTP は物理ハードウェア環境の CEC マシン・タイプを示し、CECMDLID は CEC 型式番号を示します。

この情報は、DFHSTUP 統計レポート・ユーティリティーおよび DFHOSTAT サンプル統計プログラムによって報告される、CICS 領域に関するモニター・ドメイン・グローバル統計にも含まれます。

ヒント: この情報を IBM Large Systems Performance Reference (LSPR) 比率と共に使用すると、CICS パフォーマンスおよび相対的プロセッサ能力を正確に査定できます (特に、z/OS ハードウェアのアップグレードを検討している場合)。LSPR 比率について詳しくは、[Large Systems Performance Reference for IBM z Systems](#)[®] を参照してください。

- CICS トランザクションによる zIIP 専用プロセッサまたは zAAP 専用プロセッサの実際の使用量および潜在的な使用量を計算するには、DFHTASK パフォーマンス・クラス・グループ内の CPUTONCP フィールドおよび OFFLCPUT フィールドの情報を次のように使用します。
 - フィールド 436 CPUTONCP は、標準プロセッサで費やされたタスク・プロセッサ時間の合計を示します。専用プロセッサで費やされたタスク・プロセッサ時間を計算するには、このフィールドに記録された時間を、フィールド 008 USRCPUT に記録された時間から差し引きます。
 - フィールド 437 OFFLCPUT は、専用プロセッサへのオフロードに適格であったものの、実際には標準プロセッサで実行されたタスク・プロセッサ時間の合計を示します。オフロードに適格でなかったタスク・プロセッサ時間の合計を計算するには、このフィールドに記録された時間を、フィールド 436 CPUTONCP に記録された時間から差し引きます。
 - 実際に専用プロセッサで費やされたか、または専用プロセッサで費やすのに適格だったタスク・プロセッサ時間の合計を計算するには、以下の式を使用します。

$$(\text{OFFLCPUT} + (\text{USRCPUT} - \text{CPUTONCP}))$$

- サブキャパシティー・ハードウェアでは、汎用 CP (GCP) はプロセッサ・モデルに合わせた速度に落として実行されますが、zIIP または zAAP の専用プロセッサはフルスピードで実行されます。したがって、サブキャパシティー・ハードウェアで実行する場合は、専用エンジンで費やされた CPU 時間を、同じトランザクションが GCP 上で実行された場合に費やされるはずの時間を表すように正規化する必要があります。

CICS が USRCPUT フィールド内に返す CPU 時間には、汎用 CP で費やされた合計 CPU 時間と、専用エンジンで費やされた CPU 時間を正規化したものが含まれます。CPUTONCP フィールドと OFFLCPUT フィールドは両方とも、専ら汎用 CP 上での実行に費やされた時間を表します。したがって、正規化を受けません。

LPAR の正規化係数は、IPL 時に固定されます。特定のプロセッサ・タイプの正規化係数を調べるには、SMF レコード内の以下のフィールドを使用してください。

- リソース測定機能 (RMF) - zAAP の場合は R723NFFI、zIIP の場合は R791NFFS を使用
- 共通アドレス・スペース処理 - zAAP の場合は SMF30ZNF、zIIP の場合は SMF30SNF を使用

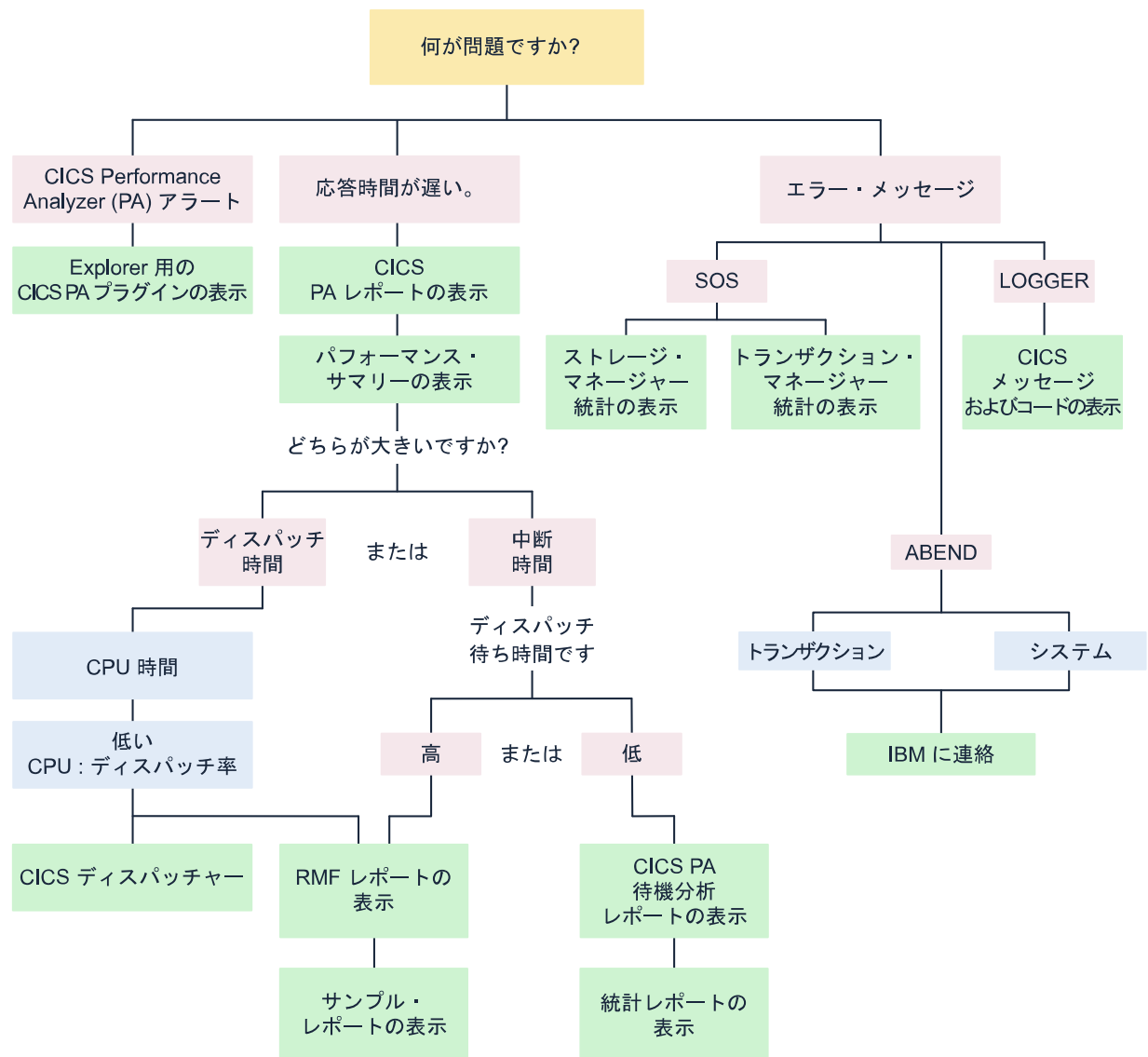
注: 正規化プロセスにより、タスク応答時間が USRCPUT フィールドに報告される時間よりも短くなるデータが生成されることがあります。

第 2 章 CICS システムのパフォーマンスの改善

チューニングは、CICS システムのパフォーマンスを改善するための主要な要因です。CICS パラメーターを通して各 CICS サブシステムを調整する前に、必ず DASD、ネットワーク、および MVS システム全体を調整する必要があります。システムの調整前に、CICS システムのパフォーマンスが予想どおりでない原因を理解しておく必要があります。

CICS システムのパフォーマンスについて懸念がある場合は、パフォーマンス・フロー・ダイアグラムをガイドとして使用して、CICS システムのパフォーマンスが予想どおりでない原因を理解し、システムの問題を解決してください。

例えば、システムの応答時間が遅すぎると判断した場合は、CICS Performance Analyzer (CICS PA) レポートおよびパフォーマンス要約を表示します。パフォーマンス要約および Performance Analyzer レポートを使用して、タスクのディスパッチ時間が長いのか、中断時間が長いのかを判断します。中断時間が長い場合は、ディスパッチ待ち時間が短いのか、長いのかを判断します。ディスパッチ待ち時間が短い場合は、レポート内の情報を使用してパフォーマンスの改善に役立ててください。



パフォーマンスの改善を支援するために、以下に示す CICS のさまざまな側面のチューニング・ガイドラインを表示できます。

- [175 ページの『データ・テーブルの使用』](#)
- [67 ページの『CICS ディスパッチャー: パフォーマンスおよび調整』](#)
- [74 ページの『仮想記憶と実記憶: パフォーマンスおよび調整』](#)
- [137 ページの『CICS のストレージ保護機能: パフォーマンスおよび調整』](#)
- [137 ページの『Language Environment での調整』](#)
- [140 ページの『Java アプリケーション: パフォーマンスおよび調整』](#)
- [140 ページの『MVS および DASD: パフォーマンスおよび調整』](#)
- [141 ページの『ネットワーキングおよび z/OS Communications Server: パフォーマンスおよび調整』](#)
- [154 ページの『CICS MRO、ISC、および IPIC: パフォーマンスおよび調整』](#)
- [163 ページの『CICS VSAM およびファイル制御: パフォーマンスおよび調整』](#)
- [192 ページの『パフォーマンスのためのデータベース 管理』](#)
- [195 ページの『CICS ロギングおよびジャーナリング: パフォーマンスおよび調整』](#)
- [209 ページの『CICS 一時記憶域: パフォーマンスおよび調整』](#)
- [214 ページの『CICS 一時データ \(TD\) 機能: パフォーマンスおよび調整』](#)
- [219 ページの『グローバル CICS エンキューおよびデキュー: パフォーマンスおよび調整』](#)
- [221 ページの『CICS モニター機能: パフォーマンスおよび調整』](#)
- [221 ページの『CICS トレース: パフォーマンスおよびチューニング』](#)
- [223 ページの『CICS セキュリティー: パフォーマンスおよび調整』](#)
- [224 ページの『CICS の起動時間およびシャットダウン時間: パフォーマンスおよび調整』](#)
- [227 ページの『CICS Business Transaction Services: パフォーマンスおよび調整』](#)
- [ワークロード管理](#)
- [CICS をモニターするための RMF の使用](#)

CICS トランザクション・マネージャー: パフォーマンスおよび調整

CICS トランザクション・マネージャー・ドメインは、トランザクション関連のサービスを提供します。

このドメインで提供されるサービスは、以下に使用されます。

- タスクの作成
- タスクの終了
- タスクのページ
- タスクに関する照会
- トランザクション定義の管理
- tranclass 定義の管理

トランザクション・マネージャー・ドメインは、他の CICS コンポーネントがトランザクション関連のサービスを実装できるトランザクション環境も提供します。

トランザクションについて詳しくは、[トランザクション統計](#)を参照してください。

最大タスク仕様 (MXT) の設定

MXT システム 初期設定パラメーターは、CICS システム内の同時ユーザー・タスクの総数を制限します。

MXT パラメーターは、ストレージ不足 (SOS) 状態を回避するために、制約されないリソース要求、特に仮想記憶域の使用を制御します。このパラメーターはカーネル・スタック・セグメントに割り振られるストレージの量にも影響を与え、リソースの競合、キューの長さ (これによりプロセッサの過度の使用を回避できる)、および実記憶域の使用量を制御します。

MXT の値は、CICS アドレス・スペース内でのストレージの使用に影響します。他のユーザーが使用できる十分なストレージを動的ストレージ域 (DSA) に確保する必要があります。

MXT パラメーターはディスパッチに適格なユーザー・タスク数を制御します。**MXT** は始動時に設定するか、**SET SYSTEM** コマンドを使用して設定することができます。**MXT** が設定されると、カーネルおよびディスパッチャーは、**MXT** 値で指定された数のユーザー・タスクを同時に作成できることを保証するために、十分な量の制御ブロックを事前に割り振ろうとします。この事前割り振り用のストレージの大部分は、CDSA または ECDSA から取得されます。ただし、各タスクに少量の MVS ストレージ (ユーザー・タスクごとに、31 ビット・ストレージ (16 MB 境界より上) で約 256 バイト、24 ビット・ストレージ (16 MB 境界より下) で約 32 バイト) が必要になります。**MXT** 値は z/OS REGION サイズ、および設定された DSA サイズ制限 (**DSALIM** および **EDSALIM** パラメーター) と相互に関連しています。[82 ページの『CICS ストレージの制限の設定』](#)を参照してください。

MXT システム初期設定パラメーターのデフォルト値は 250、最小設定は 10、最大設定は 2000 です。最初に、システムで必要とする同時ユーザー・タスクの総数に **MXT** を設定します。

MXT の設定値が大きすぎると、システム・リソース (プロセッサ、実記憶域、および仮想記憶域) が制約されたりリソース競合 (ファイル・ストリングまたはバッファ、Db2 スレッド、ENQ など) の競合が発生したりする場合に、スループットと応答時間に悪影響が出ることがあります。また、起動時の **MXT** の設定値が大きすぎる場合、CICS は使用可能な仮想記憶域に合わせて、最大タスク数を強制的により小さな値にします。

逆に、**MXT** の設定値が小さすぎる場合も、システム・リソース (プロセッサ、実記憶域、および仮想記憶域) が制約されなくても、過度のキューイング遅延のためにスループットと応答時間に悪影響が出ることがあります。

CICS 領域のパフォーマンスをモニターして、トランザクションの応答時間およびその他の時間コンポーネント (ディスパッチ時間や中断時間など) が許容できるものであるかどうかを確認します。一部のシステムでは、**MXT** の設定値を大きくし過ぎると、リソース競合が増加して、トランザクションの付加的な遅延が発生するレベルにまで至ることがあります。DFHOSTAT および DFHSTUP ユーティリティ・プログラムからのトランザクション・マネージャーのグローバル統計を使用して、**MXT** 値をモニターできます。

HTTP 接続のパフォーマンス・チューニングが有効な場合に領域が容量に達すると、HTTP 要求を CICS の **MXT** にキューイングする (これには内部処理とストレージ要件が含まれる) 代わりに、要求は TCP/IP バックログの CICS 外部にキューイングされます。HTTP 要求のサージが **MXT** を超過する (トランザクションのマネージャー・グローバル統計の **XMGPQT** フィールドによって外部化される) ことはありませんが、**MXT** に達する回数 (トランザクション・マネージャーのグローバル統計の **XMGTAMXT** フィールドとして外部化される) は増える可能性があります。これは、CICS がバックログ・キューからの各要求を処理するときに領域がストレス下に置かれる場合に発生する可能性があります。要求を処理するタスクが接続されると、CICS は **MXT** に戻ります。**MXT** のレベルがいったん低下すると、CICS は次の要求を受け取り、そのトランザクションによって CICS が **MXT** に戻される可能性があります。これは **MXT** を増やす必要があることを示しているではありません。現行値が CICS を無制限のリソース要求から正しく保護していることを示しています。

さらに、DFHTASK グループ内の以下のパフォーマンス・データ・フィールドは、トランザクションの存続期間中のタスク・ロードと、トランザクションのパフォーマンスとの間の関係を把握するのに便利です。

- **CURTASKS** は、ユーザー・タスクが接続された時点での、システム内の現在アクティブなユーザー・トランザクション数を記録します。
- **MAXTASKS** は、ユーザー・タスクが接続された時点での、CICS 領域のタスクの最大数に関する現在の設定を記録します。
- **MXTDELAY** は、**MXT** 値に達したために遅延が発生している場合に、初回ディスパッチ待ちの経過時間を記録します。

MXT 値を、CICS の稼働中に変更するには、**SET SYSTEM MAXTASKS** コマンドを使用します。CICS の実行中に **MXT** 値の設定値が大きすぎる場合、エラー・メッセージ「CEILING REACHED」が表示されます。CICS トランザクション・マネージャーの統計は、**MXT** の上限に達した回数を示します。

注: **MAXOPENTCBS** または **MAXXPTCBS** システム初期設定パラメーターが指定されていない場合は、**MXT** パラメーターによって **MAXOPENTCBS** および **MAXXPTCBS** パラメーターも設定されます。

重要: **MXT** 値を変更する前に、[オープン TCB プール](#)にある情報を検討してください。

トランザクション・クラス (MAXACTIVE) を使用してトランザクションを制御する方法

トランザクション・クラスは、システム内の CICS タスク数を制限するメカニズムを提供します。タスクを複数のトランザクション・クラスに分散し、各トランザクション・クラス内でディスパッチできる最大タスク数を制御することにより、タスク間のリソース競合を制御したり、タスク接続時にディスパッチできると CICS がみなすタスク数を制限することができます。

トランザクション・クラス定義の MAXACTIVE 属性を使用すると、リソースを大量に消費するユーザー、重要性の低いタスク (例えば「おはよう」ブロードキャスト・メッセージ) などの特定のタスク・セットを制御して、プロセッサ時間やストレージを他のタスクに割り振ることができます。MXT システム初期設定パラメーターと組み合わせると、トランザクション・クラスはトランザクションの混合を制御します。つまり、1 つタイプのトランザクションが CICS を独占しないようにします。特に、負荷の大きいタスク数、特定のデータ・セットまたはディスク・ボリュームの負荷、および回線上のプリンター負荷を制限することができます。例えば、トランザクション・クラスを使用してタスクを分離したり、すべてのユーザー・タスクを個別のクラスに収容することができます。推奨クラスは、単純な照会、複雑な照会または短いブラウズ、長いブラウズ、短い更新、長い更新です。非会話型タスクと会話型は分離してください。再入力不可能なコードを単一スレッド化する必要がある場合は、設定に ENQ を使用します。

トランザクション・クラスは、特に大量のリソースを消費するが、MAXACTIVE の上限を頻繁に超えることではないタスクに使用すると便利ことがあります。通常のタスクの場合や、特定のタスク内で機能を直列化するなどの設計上の理由がある場合は、トランザクション・クラスを使用しないでください。これらのクラスをインプリメントする代わりに、アプリケーション設計を検討する必要があります。

CICS トランザクション・クラスの統計は、トランザクション・クラス内のアクティブ・トランザクション数が MAXACTIVE 値 (Times MaxAct) に達した回数を示します。CICS は独自に使用する 2 つのトランザクション・クラス (DFHTCLSX および DFHTCLQ2) を定義します。これらのクラスの効果について詳しくは、[159 ページの『トランザクション・クラス DFHTCLSX および DFHTCLQ2 を使用したストレージ使用の制御』](#)を参照してください。

トランザクション・クラス・ページしきい値 (PURGETHRESH) の指定

トランザクション・クラス定義の PURGETHRESH 属性は、新規に作成されたにもかかわらず、関連するトランザクション・クラスの MAXACTIVE 制限に達したために開始できなかったタスク数を制限します。これらのタスクは、クラスのメンバーシップを取得するまで、トランザクション・マネージャー・ドメインによって優先順位の順にキューに入れられます。

このタスクはストレージを少しだけ占有しますが、キューが非常に長くなった場合は、CICS がストレージ不足になり、リカバリー時間が長くなることがあります。TRANCLASS メカニズムで大量のトランザクション負荷を制御するシステムは、キューによって処理が滞ることがあります。キュー内のタスクは、MXT メカニズムでは考慮されません。MXT システム初期設定パラメーターは、既にシステムに承認されているタスクの総数を TRANCLASS の制約内に制限します。

トランザクション・クラスで開始を待機しているタスクのキュー長は、該当するクラスの PURGETHRESH 属性によって制限されます。新規トランザクションによって制限に到達した場合、このトランザクションは終了し、異常終了コード AKCC が表示されます。制限到達前にキューに入れられていたタスクは、実行可能になるまで引き続き待機することができます。

PURGETHRESH 属性を指定する必要があるのは、トランザクション・クラス内のトランザクション負荷が大きい場合のみです。これは、端末専有領域 (TOR) および複数のアプリケーション専有領域 (AOR) を使用するシステムの場合や、AOR に関連したトランザクション・クラスがその AOR を使用するトランザクション数の制御に使用される場合に該当します。このような構成では、AOR の速度が低下または停止し、関連するトランザクション・クラスが AOR 内で処理を完了できないタスクで一杯 (MAXACTIVE で定義された値まで) になります。その後、新規トランザクションがキューに入れられ、トランザクション・ボリュームに応じて、CICS DSA の使用可能なストレージをすべて占有できるように数分以内にキューを拡張することができます。

キュー内の各エントリーのサイズは、トランザクション・サイズ (256 バイト) と、インターバル制御エレメント (ICE) セキュア・ストレージ拡張 (2108 バイト) と、トランザクションへのすべての端末入力を保持する TIOA サイズとの合計です。キューは、TOR にインストールされた TRANCLASS ごとに 1 つずつ、いくつでも設定できます。キューの合理的なサイズ・ページしきい値の推定値は、トランザクションが開始するまでにユーザーが待機する最大時間に、TRANCLASS 内トランザクションの最大到着レートを掛けて算出できます。最大長のキューがストレージを大量に占有できないようにしてください。

CICS がトランザクションが不必要に異常終了することがあるため (CPU 負荷の変動が原因で AOR がスローダウンする場合など)、PURGETHRESH キューイング制限を低い値に設定しないでください。TRANCLASS の PURGETHRESH 属性は、該当するトランザクション・クラスのキュー制限を設定する場合に使用します。デフォルトのアクションは、キュー長を制限しません。

各トランザクション・クラスのキュー長をモニターするには、CICS トランザクション・クラス統計を使用する必要があります。トランザクション・クラスごとに多くの統計が保持されます。これらは、キュー長をモニターするための最も役立つ統計です。

XMCP1

キューのサイズが PURGETHRESH 制限に達したために AKCC によって異常終了したトランザクション数

XMCPQT

キュー内のピーク・トランザクション数

XMCTAPT

キューのサイズが PURGETHRESH 制限に達した回数

CSMT ログで AKCC 異常終了数をモニターできます。AKCC 異常終了は、キュー制限に達した期間を示します。どの制限に達したかを判別するには、異常終了メッセージ内のトランザクション・コードをトランザクション・クラスに関連付ける必要があります。

タスクの優先順位付け

優先順位付けは、ディスパッチ中に特定のタスクを優先させる方法です。優先順位は、TERMPRIORITY)、TRANSACTION 定義内のトランザクション (PRIORITY)、および外部セキュリティー・マネージャー (ESM) のユーザー・セグメントの優先順位フィールド (OPPRTY) で指定されます。

全体的な優先順位を判別するには、指定されたタスクの 3 つの定義すべての優先順位を合計します (優先順位の最大値は 255 です)。

```
TERMPRIORITY+PRIORITY+OPPRTY <= 255
```

PRTYAGE システム 初期設定パラメーターの値はディスパッチ順序にも影響します。例えば、デフォルト値 **PRTYAGE=1000** を指定すると、作動可能キュー上のタスクの優先順位は、1000 ms 経過するごとに 1 上がります。タスクのディスパッチング優先順位はディスパッチの準備ができるたびに、クロック・タイムおよび定義済み優先順位に基づいて再評価されます。ディスパッチの準備ができたばかりの優先順位が n+1 のタスクは、通常、優先順位が n のタスクよりも先にディスパッチされます。ただし、これは、優先順位が n の最後のタスクがディスパッチできるようになってから、PRTYAGE ミリ秒が経過しなかった場合にかぎられます。したがって、優先順位が低いタスクは、ビジー・システムでは優先順位の高い多くのタスクよりも先に処理されることがありますが、最終的には 1 つのディスパッチの作動可能キューの一番上に配置されます。PRTYAGE の値が低いほど、早くタスクがディスパッチされます。特定のトランザクションがビジー期間中に優先順位が高いトランザクションの背後にスタックされる場合を除き、PRTYAGE は通常デフォルト値のままにしておく必要があります。

注：非端末トランザクションは、TXD からのトランザクション優先順位、およびオペレーター優先順位に基づく優先順位の値で接続されるのに対し、端末管理に基づくタスクはトランザクション優先順位のみで接続されます。端末管理に基づくタスクが最初にディスパッチされるときに、オペレーター優先順位と端末優先順位が追加されます。この理由から、端末ベースのタスクと非端末ベースのタスクは、同じトランザクション・クラスを介して管理してはなりません。かなりビジーなシステムでは、非端末ベースのトランザクションの安定したストリームは、他の端末管理ベースのトランザクションより優先される可能性があります。

優先順位付けはブラウズ・タスク、および多数のプロセッサ時間を使用するタスクに役立ちます。入出力制約タスクは CPU を必要な量だけ取得して、次の読み取り/書き込み待機に移行することができます。CPU 集中タスクは、集中度が低いタスクよりも優先順位が高くなります。優先順位付けはすべての CICS システムにインプリメントできます。優先順位付けは、アクティビティーが低いシステムよりも高いシステムで重要です。優先順位を慎重に選択することにより、全体的なスループットや応答時間を改善できます。優先順位付けを行うと、特定のリソース制約トランザクションのリソース使用量を最小にすることができます。優先順位付けを行うと、優先順位が低いタスクの応答時間が増加し、トランザクション・クラス定義の MXT および MAXACTIVE 属性の調整効果が変動することがあります。

優先順位は端末入力メッセージの処理順には影響しないため、トランザクション・マネージャーに接続するまでの待機時間にも影響しません。優先順位付けは3つの定義セット(端末、トランザクション、およびオペレーター)で判別されるため、システム内で多くのトランザクションを追跡するプロセスには時間がかかることがあります。CICS 優先順位付けは、オペレーティング・システムの優先順位付けのような割り込み駆動方式ではなく、作動可能キューの位置を判別します。つまり、タスクがプロセッサの制御下に入ると、CICS ディスパッチャーを呼び出す CICS コマンドが発行されるまで、タスクはこの制御から解放されません。プロセッサ制約タスクのディスパッチ後に、CICS 要求が頻繁に発生しない場合は、CICS が長期間停止することがあります。そのため、優先順位付けをインプリメントするのは、トランザクション・クラス定義の MXT および MAXACTIVE 属性の調整が不十分であると判明した場合に限定してください。

優先順位付けは多用しないでください。使用するにしても、トランザクション・クラス定義の MXT および MAXACTIVE 属性を使用してタスク・レベルを調整したあとに限定してください。すべてのタスクを同じ優先順位に設定してから、例外に基づいて、およびシステム内の特定の制約に従って、一部のトランザクションの優先順位を上下させることを推奨します。タスクの存続時間が長くなり、ディスパッチのオーバーヘッドが大きくなることを容認できるのでない限り、低速のタスクには優先順位を設定しないでください。低速のタスクはどのような場合でも低速であり、入出力を待つ必要があるたびに制御を引き渡します。小さい優先順位値および差異を使用して、トランザクションの優先順位を中心に設定を行ってください。個人でなく制御オペレーター・タスクに優先順位を設定するか、あるいは少なくとも、特定の物理端末(制御オペレーターが周囲を移動できる端末)ではなく制御オペレーターのサインオン ID に優先順位を設定します。

大量のリソースを使用するタスクに高い優先順位を付けることを検討してください。ただし、優先順位付けの効果をシステム全体で慎重にモニターして、このタイプのトランザクションを多数ロードしても他のトランザクションがロックアウトされないようにする必要があります。また、システム・リソースへのエンキューを引き起こして、他のトランザクションのロックアウトの原因となるトランザクションにも、高い優先順位を付けることを検討してください。このようにすると、これらのトランザクションは短時間で処理されて、リソースを解放します。以下にいくつかの例を示します。

- 論理リカバリーでの区画内一時データの使用
- 頻繁に使用されるレコードの更新
- 自動ロギング
- データ入力など、高速なアプリケーション応答時間を必要とするタスク

次のようなタスクには、低い優先順位を付けることを検討してください。

- ブラウズ・アクティビティーが長いタスク
- 入出力アクティビティーが最小のプロセス中心タスク
- 次のような、端末相互作用が不要なタスク
 - 自動起動タスク(端末の宛先が定義された、ゼロより大きいトリガー・レベルを持つ一時データ区画内キューを使用する場合を除く)
 - バッチ更新制御クラス

トランザクション優先順位を直接測定することはできません。間接的な測定には、次の情報を使用できます。

- タスク優先順位
- 監視対象トランザクションの応答
- プロセッサ、ストレージ、およびデータ・セット入出力の全体的な使用量

CICS ディスパッチャー: パフォーマンスおよび調整

ディスパッチ間隔を指定することによって、CICS ディスパッチャーのパフォーマンスを調整できます。ディスパッチ間隔を指定するには、インターバル制御値のシステム初期設定パラメーター、およびその他のパラメーター(FORCEQR、MROBTCH、PRTYAGE、SUBTSKS など)を設定します。

ディスパッチャー統計の詳細については、[ディスパッチャー TCB モード・レポート](#)を参照してください。

オープン TCB 管理

オープン・トランザクション環境 (OTE) は、準再入可能タスク制御ブロック (QR TCB) で待機問題を起こすことなく、CICS アプリケーション・コードが CICS アドレス・スペース内で非 CICS サービス (CICS API の有効範囲外の機能) を使用できる環境です。

オープン・トランザクション環境を利用するアプリケーションは、QR TCB ではなく独自のオープン TCB で実行されます。CICS は、QR TCB ではサブディスパッチングを実行しますが、オープン TCB では実行しません。オープン TCB で実行されているアプリケーションが呼び出した非 CICS サービスが TCB をブロックした場合、その TCB のブロッキングは他の CICS タスクには影響を与えません。オープン・トランザクション環境を利用するためのアプリケーションの作成について詳しくは、[マルチスレッド化: 再入可能なプログラム、準再入可能なプログラム、およびスレッド・セーフ・プログラム](#)を参照してください。

TCB モード

それぞれのオープン TCB モードには、そのモード特定の目的を示す 2 文字の ID があり、CICS によってそれぞれ異なる方法で扱われます。

L8 モードの TCB と L9 モードの TCB

これらの TCB は、以下のように使用されます。

- L8 TCB は、PROGRAM リソース定義で API(OPENAPI) として定義されている CICS キー・アプリケーション・プログラムに使用されます。
- L8 TCB は、PROGRAM リソース定義で CONCURRENCY(REQUIRED)、API(CICSAPI) として定義されている CICS キー・アプリケーション・プログラムに使用されます。
- L8 TCB は、ENABLE PROGRAM コマンドの OPENAPI オプションを使用して使用可能にされたタスク関連ユーザー出口 (TRUE) を介して、プログラムがリソース・マネージャーにアクセスする必要がある場合に使用されます。タスク関連のユーザー出口は常に CICS キーで実行されます。
- L8 TCB は、z/OS UNIX System Services ファイルに保管されている文書テンプレートおよび HTTP 静的応答に CICS がアクセスするときに使用されます。
- L8 TCB は、Web サービス要求、および CICS キーで実行されて OPENAPI として定義されている XML CICS プログラムの構文解析に使用されます。
- L9 TCB は、ユーザー・キーで実行されて OPENAPI として定義されているアプリケーション・プログラムに使用されます。

CICS は OPENAPI タスク関連ユーザー出口と共に作動するので、以下の製品に接続したときに L8 TCB を使用します。

- CICS-MQ アダプターを使用した IBM MQ
- CICS Db2 接続機能を使用した Db2。オープン・トランザクション環境内の CICS Db2 接続機能スレッド TCB について詳しくは、[概要: スレッドの働き](#)を参照してください。
- CICS DBCTL Database Adapter Transformer (DFHDBAT) を使用した、IMS バージョン 12 以降。[スレッド・セーフ・プログラミングにより CICS IMS アプリケーションがオープン・トランザクション環境 \(OTE\) を使用できるようにする](#)を参照してください。

その他の IBM プロダクト、例えば、IP CICS Sockets および z/OS Integrated Cryptographic Service Facility (ICSF) も、OPENAPI 対応のタスク関連ユーザー出口を使用できます。IP CICS ソケットの管理について詳しくは、[z/OS Communications Server: IP CICS ソケット・ガイド](#)を参照してください。

CICS-ICSF 接続機能について詳しくは、[z/OS Cryptographic Services ICSF System Programmer's Guide](#)を参照してください。

SP モードの TCB と S8 モードの TCB

これらの TCB は、CICS が、DFHDDAPX XPI インターフェースを使用して LDAP への SSL 接続および要求を管理するために使用します。S8 TCB は、単一エンクレープ (SP TCB が所有し、SSL キャッシュも含む) で実行されます。

S8 TCB は、CICS ディスパッチャーによって管理される SSL プールに含まれています。各 SSL 接続は、この SSL プールの S8 TCB を使用します。SSL 接続のための CICS 処理はすべて、S8 TCB で行われます (S8 TCB と SO TCB の切り替えは行われません)。Web サーバー HTTP 接続タスク (デフォルトでは CWXN) は、すべてのデータが送受信されるまで S8 TCB にとどまります。

S8 TCB は、SSL プールからタスクに割り振られますが、SSL ハンドシェークや LDAP 要求などの機能を実行するために必要な期間だけロックされます。この機能が完了した後、TCB は解放され、再使用のために SSL プールに戻されます。

注: 送受信するメッセージが非常に大きい場合は、タスクがランナウェイ制限に達して終了させられることがあります。タスクの異常終了を回避するためには、Web サーバー HTTP 接続トランザクション (デフォルトでは CWXN)、Web サーバー別名トランザクション、および Web クライアント API コマンドの SEND、RECEIVE、CONVERSE を発行するトランザクションのトランザクション RUNAWAY 値を大きくしなければならない場合があります。

UNIX System Services (USS) では、**MAXTHREADS** パラメーターおよび **MAXTHREADTASKS** パラメーターを使用して、USS プロセスが所有できる pthread の数を制限できます。各 SSL TCB には、1 つの pthread と 1 つの MVS タスクが必要です。そのため、これらの USS パラメーターの値は **MAXSSLTCSB** システム初期設定パラメーターの値を上回っていることを確認する必要があります。**MAXTHREADS** または **MAXTHREADTASKS** に十分大きな値を設定していない場合、CICS が SSL TCB を接続しようとしている間にこれらの制限のどちらかに達すると、CICS は DFHDSIT からエラー・メッセージ DFHDS0002 重大エラー・コード X'0137' を発行します。

TP モードの TCB と T8 モードの TCB

これらの TCB は、JVM サーバーが Java プログラムの要求を処理するために使用します。JVM サーバーは、単一の JVM 内で Java アプリケーションの複数の同時要求を処理できるランタイム環境です。TP モードの TCB は、Language Environment エンクレープと T8 TCB のプールを所有しています。CICS 領域で実行されている各 JVM サーバーは、1 つの TP TCB と 1 つ以上 (ただし、256 を超えない) の T8 TCB を持っています。T8 TCB は、適切な JVM サーバーの THRD プールからタスクに割り振られますが、システム処理を実行するために必要な期間だけロックされます。T8 TCB は、JVM サーバー間で共用されません。

各 T8 TCB には、1 つの pthread と 1 つの MVS タスクが必要です。CICS 領域に許可される T8 TCB の最大数は 2000 です。z/OS UNIX では、**MAXTHREADS** パラメーターおよび **MAXTHREADTASKS** パラメーターを使用して、z/OS UNIX プロセスが所有できる pthread の数を制限できます。そのため、これらのパラメーターの値は T8 TCB の最大数を上回っていることを確認する必要があります。

MAXTHREADS または **MAXTHREADTASKS** に十分大きな値を設定していない場合、CICS が T8 TCB を接続しようとしている間にこれらの制限のどちらかに達すると、CICS は DFHDSIT からエラー・メッセージ DFHDS0002 重大エラー・コード X'0137' を発行します。JVM サーバーのスレッド制限について詳しくは、[JVM サーバーのスレッド限度の管理](#)を参照してください。

X8 モードの TCB と X9 モードの TCB

これらの TCB は両方とも、XPLINK オプションを使用してコンパイルされた C プログラムおよび C++ プログラムを実行するために使用されます。X8 TCB は CICS キーのプログラムに使用され、X9 の TCB はユーザー・キーのプログラムに使用されます。XPLink プログラムの各インスタンスは、1 つの X8 TCB または X9 TCB を使用します。XPLink の使用についての詳細は、『[XPLink and C and C++ programming](#)』を参照してください。

オープン TCB プール

CICS は、プール内のオープン TCB を管理します。1 つのプールには、同じ目的のために使用されるオープン TCB が含まれています。1 つのプールは、タスクに割り振られたいくつかの TCB と、アプリケーションによって既に解放され、再利用可能な状態になっているその他の TCB とで構成されることがあります。

CICS は、プールに対して設定された制限まで、各プールにオープン TCB を作成または接続できます。各プールに許可された TCB の最大数は、以下のように設定されます。

- **MAXOPENTCBS** システム初期設定パラメーターが指定された場合、それによってオープン TCB プールの値が設定されます。MAXOPENTCBS システム初期設定が指定されない場合、CICS は CICS 領域に対して指定されたタスクの最大数 (MXT 値) に基づいて、L8 および L9 モードのオープン TCB プールの制限を自動的に設定します。このとき、公式 $(2 * \text{MXT 値}) + 32$ を使用します。MAXOPENTCBS パラメーターをユーザー自身が明示的に設定する方法については、[MAXOPENTCBS](#) を参照してください。
- **MAXSSLTCSB** システム初期設定パラメーターは、SSL TCB プールの値を指定します。

- **MAXTHRDTCBS** は、JVM サーバー THRD TCB プールの値を指定します。それぞれの JVMSERVER リソース上の JVM serverTHREADLIMIT の値に予約されるスレッドの数は、自動的にスレッド数に 1 を加えた値として計算され、最大限度は 2000 です。
- **MAXXPTCBS** システム 初期設定パラメーターが指定された場合、それによって XP TCB プールの値が設定されます。**MAXXPTCBS** システム 初期設定が指定されない場合、CICS は X8 および X9 モードの XP TCB プールの制限を、CICS 領域に対して指定されたタスクの最大数 (MXT 値) と等しい値に自動的に設定します。**MAXXPTCBS** パラメーターをユーザー自身が明示的に設定する方法については、[MAXXPTCBS](#) を参照してください。

アプリケーションが、オープン TCB を必要とする要求を行った場合、CICS は最初に、該当するプールで再使用可能な適切な TCB を見つけようとします。CICS は、TCB が正しいサブスペースに対するものである場合のみ、正しいモードの使用可能な TCB と要求をマッチングさせることができます。CICS は、正しいサブスペース用のフリー TCB に適切に一致するものが見つからない場合、プールの制限に達していなければ、新規 TCB を接続します。

CICS タスクでは、X8 および X9 TCB を必要な数だけ使用でき、これらの TCB はプログラムが終了するまでのみ保持されます。ただし、各 CICS タスクでは、最大で 1 つの L8 TCB と 1 つの L9 TCB が許可され、L8 TCB と L9 TCB は、割り振られた時点からタスクが終了するまで保持され、必要に応じて将来の要求に再使用されます。タスクが終了すると、TCB は解放され、CICS はそれらを別のタスクに割り振ったり、破棄したりすることができます。

時として、CICS は、アプリケーションの要求に一致するものが見つけれず、プールの制限に達してしまうことがあります。このような場合、CICS は、プール内の誤ったサブスペースまたはモードを持つフリー TCB を破棄し、それを正しいモードおよびサブスペースを持つ TCB で置き換えることによって、要求を満たすことがあります。この手法は、スチーリングと呼ばれます。スチーリングは、オープン TCB のタイプによってはパフォーマンス・コストが高くなる可能性があるため、CICS は妥当と思われる場合はスチーリングを回避します。CICS は、過剰な TCB 管理および TCB スチーリング・アクティビティの統計を、CICS ディスパッチャー TCB モードおよび TCB プール統計に維持します。

TCB の数がプールの制限に達し、スチールするためのフリー TCB がいない場合は、TCB がフリーになるまで、またはプールの制限が増加するまで、タスクは OPENPOOL 待ちで中断されます。

ストレージへの影響を最小限に抑えるために、CICS は、現在のニーズに照らして各プール内のオープン TCB の数のバランスを取ろうとします。CICS は、プール内でフリー TCB を検出した場合、徐々にそれらを切り離して過剰な数を減らし、過剰な TCB が使用していたリソースを解放します。

MAXSSLTCBS

DFH0STAT および DFHSTUP ユーティリティー・プログラムからのディスパッチャーの TCB 統計を使用して、SSL プール内の S8 TCB をモニターすることができます。TCB の最大数は、**MAXSSLTCBS** システム 初期設定パラメーターで設定されます。

SSL のパフォーマンスを向上させたい場合は、ディスパッチャー・レポートを使用して、S8 TCB 待ちのタスクが多数存在するかどうかを調べることができます。同時に、キューに入っているタスクの数も調べます。両方のフィールドで多数のタスクが報告された場合は、S8 TCB の最大数を増やします。キューに入っているタスクの数はわずかであるが、待ちタスク数が多い場合は、S8 TCB の数を増やすかどうかはユーザーが決定できます。S8 TCB の数を 1 つまたは 2 つ増やすことにより、待ちタスク数に効果があり、キューに入っているタスクの数を減らすことができ、ストレージに大きなオーバーヘッドが発生することもあります。

設定可能な S8 TCB の最大数は、1024 です。ただし、多数の S8 TCB を設定した場合も、ストレージの使用量によりパフォーマンスに影響を与えることがあります。CICS がストレージを使い尽くすと、TCB の接続障害が発生します。この障害は、S8 TCB モードの統計に関するディスパッチャー・レポートで報告されます。

インターバル制御値パラメーター: ICV、ICVR、および ICVTSD

インターバル制御値 (ICV) は、新規の値を設定するためにシステム 初期設定テーブル (SIT) で指定されるか、またはオーバーライドによって指定されます。ICV パラメーターを正しく設定すると、使用率の低い CICS 領域のプロセッサの使用量を削減することにより、パフォーマンスの改善に役立つことがあります。

CICS には、3 つのタイプのインターバル制御値パラメーターがあります。

インターバル制御値 (ICV)

ICV システム 初期設定パラメーターは、処理を再開するトランザクションがない場合に、CICS がオペレーティング・システムに制御を解放する最大時間 (ミリ秒) を指定します。この時間間隔は、100 ミリ秒から 3600000 ミリ秒の範囲の整数を指定できます (最大 60 分までの間隔を指定します)。標準的な運用の範囲は 100 ミリ秒から 2000 ミリ秒です。

ランナウェイ・タスクのインターバル制御値 (ICVR)

ICVR システム 初期設定パラメーターは、デフォルトのランナウェイ・タスク時間をミリ秒単位で 10 進数として指定します。ゼロ、または 250 から 2700000 の範囲の数値を、250 の倍数で指定できます。CICS は、250 の倍数でない値を切り捨てます。これは、RUNAWAY=SYSTEM で定義されたトランザクションによって使用される RUNAWAY 間隔です。

端末スキャン遅延のインターバル制御値 (ICVTSD)

以前のリリースで **ICVTSD** システム 初期設定パラメーターは、アプリケーションから出されるある種の端末出力要求の処理オーバーヘッドを分散させる目的で、CICS がどれほど速やかに要求を処理するかを制限するために使用されていました。範囲は 0 から 5000 ミリ秒です。非 SNA ネットワークで CICS システムを使用しているような場合には、ゼロ以外の値を指定することが適切でした。しかし、SNA および IPIC ネットワークでは、応答時間を短縮し、仮想ストレージを最大限に利用するためには、ICVTSD を 0 (デフォルト) に設定するのが適切です。

MROBTCH

MROBTCH システム 初期設定パラメーターは、通知の前に 1 つのバッチで累積できる領域内のイベントの数を指定します。

それらの要求を処理できるようにするために、その領域が開始されます。複数領域操作 (MRO) 要求のバッチ処理には、いくつかの非 MRO イベントが含まれます。

- VSAM 物理入出力完了イベント
- サブタスク要求完了 (大部分は VSAM)
- DBCTL を介して実装されている DL/I 要求の完了

MROBTCH パラメーターの値は 1 から 255 の範囲です。デフォルト値は 1 です。このバッチ処理のメカニズムを使用すると、CICS 内のディスパッチ・リソースの使用を複数のタスクに分散させることができます。この値が 1 より大きく、CICS がシステム待機状態にある場合、指定された数のイベントが発生するまで、CICS はディスパッチを通知されません。イベントには、接続システムまたは DASD 入出力からの MRO 要求、および CHANGE_MODE 処理が含まれます。これらのイベントの場合、CICS は以下の条件が満たされた後でディスパッチされます。

- 現在のバッチが満杯になる (イベント数が **MROBTCH** に等しい)
- ICV 間隔が満了する

ICV パラメーターで指定する時間間隔は、システムに過度の遅延が生じないように十分低くする必要があります。

使用率の低い期間中は、**MROBTCH** の値が 1 より大きいと、トランザクションの応答時間が増えることがあります。FCIOWAIT 値の増加により、入出力要求を出すトランザクションで遅延が生じる可能性があります。**MROBTCH** がパフォーマンスに与える影響について、詳しくは [160 ページの『要求のバッチ処理 \(MROBTCH\)』](#) を参照してください。

FORCEQR

FORCEQR システム 初期設定パラメーターは、スレッド・セーフとして定義されたすべての CICS API ユーザー・アプリケーション・プログラムを、準再入可能プログラムとして指定されたプログラムと同様に、CICS 準再入可能 (QR) タスク制御ブロック (TCB) 下で実行することを CICS で強制するかどうかを指定します。

ユーザー・プログラムが準再入可能として定義されている場合、CICS は常に CICS QR TCB 下でプログラムを呼び出します。マルチスレッド化コンテキストでの準再入可能プログラムに対する要件は、そのプログラムを複数の TCB 上で同時に実行する場合ほどは厳格ではありません。CICS は、アプリケーション・プログラムが一貫性のある条件を保証するように、それが再入可能であることを要求します。実際には、アプリケーション・プログラムは真に再入可能ではない場合がありますが、CICS では「準再入可能性」を予想

しています。このことは、アプリケーション・プログラムは、それに制御が渡されているときは、入力時
も、各 **EXEC CICS** コマンドの前と後も、一貫性のある状態である必要があることを意味しています。こ
うした準再入可能性により、アプリケーション・プログラムの各呼び出しは、前の実行や、複数の **CICS** タ
スクによるプログラムの同時マルチスレッド化による影響を受けないことが保証されます。

CICS 準再入可能ユーザー・プログラム (アプリケーション・プログラム、ユーザー置き換え可能モジュール、
グローバル・ユーザー出口、およびタスク関連ユーザー出口) は、**QR TCB** 下で **CICS** ディスパッチャ
ーによって制御が渡されます。この **TCB** 下で実行されているプログラムは、そのプログラムが **CICS** 要求
時に制御を解放するまでは他の準再入可能プログラムは実行できないことが保証され、制御を解放した時
点でユーザー・タスクは中断状態になりますが、プログラムはまだ「使用中」のままになります。その後、
同じプログラムを別のタスクのために再呼び出しすることができます。つまり、アプリケーション・プロ
グラムは複数のタスクによって同時に使用中にすることができます。ただし、実際には一度に 1 つのタス
クしか実行できません。

OTE を使用するためにスレッド・セーフとして定義されたプログラムを使用するアプリケーション (**CICS**
Db2 アプリケーションなど) を実行しているときは、1 つ以上のプログラムがスレッド・セーフでない場
合、問題が起きることがあります。**FORCEQR** システム初期設定パラメーターを使用して、すべてのアプリ
ケーションを強制的に **QR TCB** に入れることができます。

アプリケーションを強制的に **QR TCB** に入れる方法は、問題を調査する間、アプリケーションをサービス
休止にする余裕がない実動領域で役立ちます。

このパラメーターのデフォルトは **FORCEQR=NO** です。これは、**CICS** はプログラム・リソース定義の
CONCURRENCY 属性を尊重することを意味します。一時的な手段として、スレッド・セーフとして定義さ
れたプログラムに関連する問題を調査して解決する間だけ **FORCEQR=YES** に設定することができます。す
べての問題が解決したら、**FORCEQR=NO** にリセットすると、すべてのプログラムは **OTE** 下でオープン **TCB**
の使用を再開します。

FORCEQR パラメーターは、現行の **CICS** プログラミング・インターフェースに制限されているすべてのア
プリケーション・プログラム (**API(CICSAPI)** を指定するプログラム) に適用されます。このパラメーター
は、以下のプログラムには適用されません。

- **JVM** 内で実行される **Java** プログラム
- **XPLINK** を使用している **C** または **C++** プログラム
- **OPENAPI** プログラム
- **CONCURRENCY(REQUIRED)** で定義されたプログラム

FORCEQR パラメーターは、タスク関連ユーザー出口、グローバル・ユーザー出口、またはユーザー置き換
え可能モジュールとして使用されない、スレッド・セーフとして定義されたすべてのプログラムに適用さ
れます。

PRTYAGE

システム初期設定パラメーター **PRTYAGE** は、システム内のタスクの優先順位繰り上げの速度を決定する値
に設定できます。

CICS 内のタスクの優先順位は、タスクがディスパッチされる順序を決定します。タスクは、1 から 255 の
優先順位の値を持つことができます。特定のタスクの最初のディスパッチが遅すぎる場合、優先順位を高
い値に変更すると、ディスパッチ時間が短縮されます。**CICS** システム・タスクの優先順位は制御できませ
ん。**PRTYAGE** の値の調整は、タスクの優先順位を制御するのではなく、**CICS** がタスクの優先順位を設定す
る方法のみを制御します。**PRTYAGE** の値を変更すると、タスクがディスパッチされる速度に影響を与えま
す。

SUBTSKS

SUBTSKS システム初期設定パラメーターは、**CICS** が並行モードでタスクを実行するために使用するタスク
制御ブロック (**TCB**) の数を指定します。

SUBTSKS パラメーターの値は 1 か 0 のどちらかです。1 の値は、サブタスキングをオンにし、0 の値は、
サブタスキングをオフにします。

0 の値 (**SUBTSKS** のデフォルト値) を使用すると、CICS は準再入可能 (QR) TCB 下で稼働し、すべてのアプリケーションを QR TCB 下で実行します。この値では、CICS はリソース所有モード TCB 下でファイルを開いたり閉じたりするタスクも実行します。

このパラメーター値を 1 に設定すると、CICS はリソース所有 TCB および QR TCB 下で稼働し、追加の TCB (並行モード TCB) を使用してシステム・サブタスキングを実行します。

TCB 統計

タスク制御ブロック (TCB) ディスパッチャー統計は、最後に統計がリセットされた後に各 CICS TCB によって消費された CPU 時間の量を報告します。

CICS 統計が最後にリセットされてからのおよその時間を計算するには、「Accum time in MVS wait (MVS 待機の累積時間)」と「Accum time dispatched (ディスパッチされた累算時間)」の値を加算します。各 CICS TCB の使用量の割合を計算するには、「Accum CPU time/TCB (累算 CPU 時間/TCB)」の値を、CICS 統計が最後にリセットされてからの時間 (直前の部分で計算した値) で除算します。

「Accum CPU time/TCB (累算 CPU 時間/TCB)」値には、キャプチャーされない時間は含まれません。そのため、この計算を使用すると、過密使用中の CICS TCB であっても 100% 使用中をかなり下回ります。この計算で CICS 領域の使用率が 70% より高い場合は、その領域のキャパシティーに近づきつつあります。ただし、70% という計算結果は概算にすぎない可能性があります。領域のキャパシティーは、実行中のワークロード、ワークロード内のアクティビティーの混合状態、および現在使用中の CICS のリリースなどの要因によって異なります。リソース測定機能 (RMF) を使用して正確な測定値を取得してから計算で使用したり、モニター・システムで RMF を使用したりできます。詳細については、[z/OS リソース測定機能 \(RMF\) レポート分析](#) を参照してください。

注: 「Accum Time Dispatched (ディスパッチ累積時間)」は、CPU 時間の測定値ではありません。MVS は、CICS が意識しなくても、すべての I/O アクティビティーや高い優先順位の領域など、より高い優先順位の作業を実行できます。

TCB モードは、以下のとおりです。

QR

準再入可能モードの TCB が常に 1 つあります。これは、準再入可能 CICS コードおよび非スレッド・セーフ・アプリケーション・コードの実行に使用されます。

FO

ファイル占有 TCB が常に 1 つあります。ユーザー・データ・セットのオープンとクローズに使用されます。

RO

リソース占有 TCB が常に 1 つあります。RO TCB は、プログラムのロードに使用されます。ただし、そのプログラムをロードするコマンド (EXEC CICS LOAD、XCTL、または LINK) が、オープン TCB で現在実行中のアプリケーションによって発行された場合を除きます。そのような場合は、RO TCB の代わりにオープン TCB がプログラムのロードに使用されます。RO TCB も、CICS データ・セットのオープンとクローズや、RACF 呼び出しの実行などのタスクにも使用されます。

CICS ロダー・ドメイン・グローバル統計は、RO TCB 上で行われたプログラム・ロード操作の数と、それらに費やされた時間を記録します。これらの値を、プログラム・ロード操作の数と時間の全体の統計と比較して、RO TCB の代わりにオープン TCB 上で行われたプログラム・ロード操作の比率を知ることができます。

CO

オプションの並行モード TCB は、VSAM 要求などの他の CICS アクティビティーと並列に安全に実行できるプロセスに使用されます。値 0 または 1 を使用してシステム初期設定パラメーター **SUBTSKS** を定義し、CO TCB の有無を指定します。

D2

D2 モードの TCB は、Db2 保護スレッドの停止に使用されます。保護スレッドは通常のページ・サイクルで停止します。または、ユーザーが **DSNC DISCONNECT plan-name** コマンドを実行すると、計画の保護スレッドはすぐに停止します。

SZ

単一オプション SZ モードの TCB は FEPI インターフェースによって使用されます。

RP

単一オプション RP モードの TCB は、ONC/RPC 呼び出しに使用されます。

EP

EP モードの TCB は、CICS 領域でイベント処理を実行するために使用されます。TCB は、イベントを適切な EP アダプターにディスパッチするか、システム・イベントのフィルター操作を延期します。

L8

OPENAPI オプションによって使用可能に設定済みで、かつ EXECKEY=CICS で定義されているプログラムをタスクが呼び出す場合、または OPENAPI オプションによって使用可能に設定済みのタスク関連ユーザー出口プログラムをタスクが呼び出す場合、そのタスクは自分専用の L8 モードの TCB を持ちます。L8 TCB が使用されるのは、CICS が CICS-MQ アダプターを使用して WebSphere® MQ バージョン 6 以降に接続する場合と、CICS が DB2® バージョン 8 以前に接続する場合です。

L9

タスクに専用の L9 モードの TCB があるのは、OPENAPI オプションを指定して使用可能にされていて、かつ EXECKEY=USER で定義されているプログラムをタスクが呼び出す場合です。

SO

SO モードの TCB は、TCP/IP のソケット・インターフェースに対する呼び出しに使用されます。

SL

SL モードの TCB は、聴取ソケット・セット上のアクティビティを待つために使用されます。

S8

タスクがシステムの Secure Sockets Layer (SSL) を使用する必要がある場合、タスクは S8 TCB を使用します。また、タスクが DFHDDAPX XPI インターフェースを介して LDAP を使用する必要がある場合にも、タスクは S8 TCB を使用します。この TCB は、SSL の折衝中、または LDAP 要求の際にのみ使用されます。完了すると、TCB は解放され、SSL プールに戻って再使用されます。

SP

SP モードの TCB は、ソケット pthread 専有タスクに使用されます。この TCB は、S8 TCB の SSL プールを管理し、SSL キャッシュを含む Language Environment エンクレープを所有します。

T8

JVMSEVER で稼働中の Java アプリケーションは、T8 TCB を使用します。T8 は、CICS が Db2 以降に接続する場合、Java アプリケーションからの DB2 バージョン 9 要求にも使用されます。

TP

TP モード TCB は、Language Environment エンクレープ、JVM、THRD TCB プール、および JVM サーバーの T8 TCB を所有および管理します。

X8

タスクに専用の X8 モードの TCB があるのは、XPLINK コンパイラー・オプションでコンパイルされていて、かつ EXECKEY=CICS で定義されている C または C++ プログラムをタスクが呼び出す場合です。CICS-Db2 タスク関連ユーザー出口は、それが CONCURRENCY(REQUIRED) および API(CICSAPI) を使用して実行される場合、X8 TCB を使用する可能性があります。

X9

タスクに専用の X9 モードの TCB があるのは、XPLINK コンパイラー・オプションでコンパイルされていて、かつ EXECKEY=USER で定義されている C または C++ プログラムをタスクが呼び出す場合です。

仮想記憶と実記憶: パフォーマンスおよび調整

z/OS システム内および CICS 領域内の仮想記憶と実記憶の使用について理解し、ストレージの使用についてパフォーマンスのモニターおよび調整を開始します。

手順

1. z/OS アドレス・スペース内の仮想記憶と実記憶の配置方法および管理方法について理解します。
z/OS 内のアドレス・スペースについては、以下の z/OS 資料を参照してください。
 - ・ アドレス・スペース内の 24 ビットおよび 31 ビット・ストレージ (2 GB 境界より下) については、[「z/OS MVS 初期設定およびチューニングガイド」](#)の『仮想記憶域の概要』を参照してください。

- アドレス・スペース内の 64 ビット (2 GB 境界より上) ストレージについては、『[z/OS MVS Programming: Extended Addressability Guide](#)』の『[64 ビット・アドレス・スペースの使用](#)』を参照してください。
- 2. z/OS アドレス・スペースが CICS 領域によって使用される場合の仮想記憶の配置および管理の方法を理解します。
CICS が z/OS アドレス・スペースを使用する方法については、[75 ページの『CICS 仮想記憶』](#)を参照してください。
- 3. z/OS パフォーマンス・ツールおよびモニター・ツール (z/OS リソース測定機能 (RMF) など) を使用して、ご使用の z/OS システム全体の仮想記憶の使用をモニターおよび測定します。
RMF の概要については、[34 ページの『リソース測定機能 \(RMF\)』](#)を参照してください。
詳しくは、[z/OS リソース測定機能 \(RMF\) ユーザーズ・ガイド](#)を参照してください。
- 4. 以下の機能を使用して、各 CICS 領域内の仮想記憶の使用および動的ストレージ域 (DSA) のサイズをモニターおよび測定します。
 - CICS ストレージ・マネージャー統計
 - サンプル統計プログラム DFHOSTAT で作成されたレポート
 - ローダー・ドメインおよびストレージ・ドメインの CICS 定様式ダンプ
- 5. ご使用の z/OS システム全体にわたるシステムの使用を調整します。
RMF を使用している場合、CICS およびストレージに関する RMF レポートの説明、および調整の方針については、[z/OS リソース測定機能 \(RMF\) レポート分析](#)を参照してください。
- 6. 資料のこのセクションに示されている方法および推奨を使用して、各 CICS 領域内のストレージの使用を調整します。
予想と最も異なっていると思われるストレージの領域を重点的に調整してください。
- 7. CICS 領域で Java アプリケーションを使用する場合は、[JVM サーバーのストレージ要件の計算](#)を参照してください。

CICS 仮想記憶

各 CICS 領域は、独自の z/OS アドレス・スペースで稼働します。z/OS アドレス・スペース内で使用可能なストレージは、複数の異なる領域に分割されています。

[76 ページの図 16](#) は、z/OS アドレス・スペース内で使用可能なストレージの概要を示しています。この仮想記憶の理論上の上限は極めて高くなっていますが、実記憶域に対する実質的な制限があります。そのため、z/OS では、各アドレス・スペースに対して、そのアドレス・スペースで使用できるストレージの量を制限する **REGION** パラメーターと **MEMLIMIT** パラメーターが適用されます。

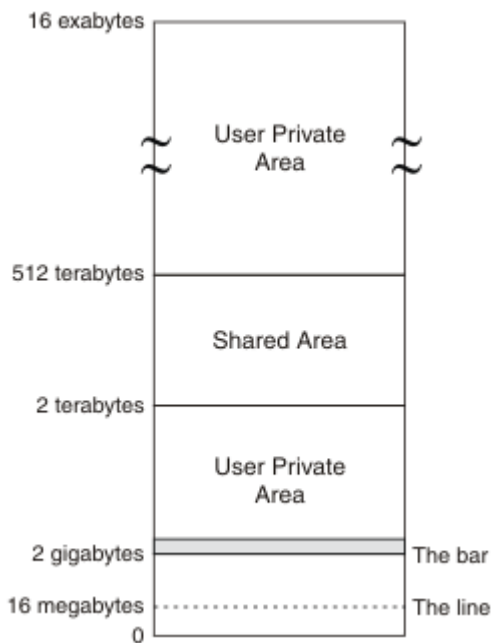


図 16. z/OS アドレス・スペース

CICS は、z/OS アドレス・スペースの 3 つの領域内の仮想記憶域を使用および管理します。

16 MB 境界より下のストレージ (0 MB から 16 MB)

この領域のストレージは、24 ビット・ストレージです。

16 MB アドレスより下のアドレスは 24 ビット・アドレッシングによってアクセスされ、プログラムは AMODE 24 以上で実行される場合にこのストレージを使用できます。16 MB アドレスは境界と呼ばれるので、24 ビット・ストレージは 16 MB 境界より下のストレージとも呼ばれます。

16 MB 境界より上のストレージ (16 MB から 2 GB)

この領域のストレージは、31 ビット・ストレージです。

16 MB アドレスより上かつ 2 GB アドレスより下のアドレスは、31 ビット・アドレッシングによってアクセスされ、プログラムは AMODE 31 以上で実行される場合にこのストレージを使用できます。16 MB アドレスは境界と呼ばれるので、31 ビット・ストレージは 16 MB 境界より上のストレージとも呼ばれます。

2 GB アドレスより下の仮想記憶域をユーザー専用領域から分離する領域は、境界と呼ばれます。24 ビット・ストレージと 31 ビット・ストレージは、2 GB より下のストレージにあり、総称して 2 GB 境界より下のストレージと呼ぶことができます。

2 GB 境界より上のストレージ (4 GB から理論上の 16 エクサバイト)

この領域のストレージは、64 ビット・ストレージです。

2 GB アドレスより下の仮想記憶域をユーザー専用領域から分離する領域は、境界と呼ばれ、64 ビット・ストレージは 2 GB 境界より上のストレージとも呼ばれます。

2 GB 境界より上のストレージは、4 GB から 2 テラバイトまでのユーザー専用領域、2 テラバイトから 512 テラバイトまでの共用記憶域、およびその共用記憶域の終わりから 16 エクサバイトまでのユーザー専用領域で構成されています。

2 GB 境界より上のアドレスは 64 ビット・アドレッシングによってアクセスされ、AMODE 64 で実行されるプログラムはこのストレージを使用できます。

ストレージの各専用領域では、仮想記憶域は以下の目的に使用されます。

CICS 動的ストレージ域

動的ストレージ域は、CICS、アクセス方式、および CICS 内で実行されるアプリケーションのストレージ所要量を提供するために使用されます。78 ページの『CICS 動的ストレージ域』を参照してください。

MVS ストレージ

MVS ストレージは、オペレーティング・システムが領域関連のサービスを実行するために使用可能です。[119 ページの『64 ビット MVS ストレージ』](#) および [119 ページの『2 GB より下の MVS ストレージ』](#) を参照してください。

注：この情報および以下のトピックでは CICS と共にインストールされる他の製品について言及していますが、これは本書の作成時点で有効なものです。CICS と共にインストールされる他の製品については、ご使用の製品のバージョンに関する情報を常に確認してください。

CICS 領域サイズ

CICS が実行されるアドレス・スペースの仮想記憶域の量は、z/OS **REGION** パラメーターおよび **MEMLIMIT** パラメーターで指定します。

- z/OS **REGION** パラメーターでは、24 ビットおよび 31 ビット・ストレージ、つまり、2 GB 境界より下のストレージの量に対する要求を指定します。最大 2047 MB のストレージを要求できますが、2 GB より下の MVS ストレージを必要とする領域関連のサービス用に十分なストレージを残す必要があります。
- z/OS **MEMLIMIT** パラメーターでは、CICS 領域の 64 ビット (2 GB 境界より上) ストレージの制限を指定します。CICS 領域には、10 GB 以上の **MEMLIMIT** 値が必要です。z/OS の **MEMLIMIT** のデフォルト値は 2 GB です。

新規リリースの CICS にアップグレードする場合は、**REGION** パラメーターおよび **MEMLIMIT** パラメーターの設定を再評価してください。また、新規リリースの z/OS、または CICS 以外のサブシステムをインストールする場合も、設定を再評価してください。CICS 内の変更は、CICS DSA の 24 ビット、31 ビット、および 64 ビット・ストレージの要件を変える可能性があります。他の製品に対する変更は、CICS DSA の外部の MVS ストレージの要件を変える可能性があります。

CICS の実行中に、CICS 領域の **REGION** または **MEMLIMIT** の値を変更することはできません。CICS 領域の次の始動時に、新しい値を指定できます。説明は、[CICS 領域のアドレス・スペース・ストレージ制限の設定](#) を参照してください。

REGION パラメーターは、さまざまな方法で指定して特定の量のストレージを要求することも、すべての使用可能な 24 ビットまたは 31 ビット専用ストレージを要求することもできます。2 GB 境界より下に得られる領域サイズは、予測不能ことがあります。z/OS メッセージ IEF374I は、z/OS が CICS 領域に割り当てる 2 GB 境界より下のストレージの総量を報告します。このメッセージの VIRT=nnnK 部分は 24 ビット・ストレージを示し、EXT=nnnK 部分は 31 ビット・ストレージを示します。CICS サンプルの統計プログラム DFHOSTAT は、この情報を含む報告書を生成します。RMF を使用して、ストレージの使用をさらに詳細にモニターすることもできます。

REGION 値を増やすことを計画している場合は、以下の点に留意してください。

- 24 ビットおよび 31 ビット・ストレージ内の高専用領域のストレージ要件に注意してください。このストレージの一部は、z/OS Communications Server およびその他のプログラムによって使用されます。CICS に割り振られる 24 ビットまたは 31 ビット・ストレージを増やすと、高専用領域の項目に使用可能なストレージが減ります。これらの項目は、ローカル・システム・キュー域 (LSQA)、スケジューラー作業域 (SWA)、およびサブプール 229 と 230 です。これらサブプールが不足すると、S80A、S40D、および S822 異常終了の原因となります。高専用領域および LSQA については、[123 ページの『高専用領域』](#) を参照してください。
- **REGION** 値を増やす場合は、CICS システム初期設定パラメーター **DSALIM** と **EDSALIM** の値を必要に応じて増やすことを念頭に置いてください。そうしないと、CICS は追加のストレージを使用できません。

REGION パラメーターおよび **MEMLIMIT** パラメーターについての情報、およびそれらを z/OS アドレス・スペースに適用する方法についての詳細は、以下の z/OS 情報を参照してください。

- アドレス・スペース内の 24 ビットおよび 31 ビット・ストレージ (2 GB 境界より下のストレージ) については、[『z/OS MVS 初期設定およびチューニングガイド』の『仮想記憶域の概要』](#) を参照してください。
- アドレス・スペース内の 64 ビット (2 GB 境界より上) ストレージについては、[『z/OS MVS Programming: Extended Addressability Guide』の『64 ビット・アドレス・スペースの使用』](#) を参照してください。
- [『z/OS MVS JCL Reference』の『REGION パラメーター』](#)。
- [『z/OS MVS JCL 解説書』の『MEMLIMIT パラメーター』](#)。

CICS 領域に必要な仮想記憶域の総量が増えた場合は、CICS によって要求される監視プログラム呼び出し (SVC) ダンプに割り振られるスペースの量、および使用可能な補助記憶域の量を検討することが必要な場合があります。SVC ダンプ・データ・セットの管理については、[z/OS MVS 診断 ツールと保守援助プログラム](#)を参照してください。補助記憶管理については、[z/OS MVS 初期設定およびチューニングガイド](#)を参照してください。

CICS 動的ストレージ域

動的ストレージ域 (DSA) は、CICS タスクにトランザクションを実行するためのストレージを提供します。この領域は CICS の操作に不可欠なものです。24 ビット・ストレージの DSA は、CDSA、UDSA、SDSA、および RDSA です。31 ビット・ストレージの DSA は、ECDSA、EUDSA、ESDSA、ERDSA、および ETDSA です。64 ビット・ストレージ内の DSA は、GCDSA、GUDSA、および GSDSA です。

動的ストレージ域は、MVS ストレージ・サブプールから取得する仮想記憶ページから作成されます。動的ストレージ域では、CICS はストレージを CICS サブプール単位で配置します。サブプールは、動的ストレージ域から一度に 1 ページずつ、必要に応じて動的に獲得されます。個々のサブプールが使用するストレージは、CICS ストレージ・マネージャー統計のドメイン・サブプール統計に表示されます。

CICS は、DSA ストレージをエクステント単位で管理します。個々の DSA は、1 つ以上のエクステントで構成されます。

- 24 ビット・ストレージ・エクステントは通常、256 KB の倍数で割り振られます。ただし、トランザクション分離が動作中である場合は、UDSA は 1 MB エクステント単位で割り振られます。
- 31 ビット・ストレージ・エクステントは、1 MB の倍数で割り振られます。
- 64 ビット・ストレージ・エクステントは、1 GB の倍数で割り振られます。

所有する DSA のみが割り振られたエクステントを使用でき、特定のエクステントを同時に複数の DSA 間で共用することはできません。

DSA 用のストレージは、CICS キー・ストレージ、ユーザー・キー・ストレージ、または読み取り専用キー 0 保護ストレージから割り振ることができます。各 DSA に割り振られるストレージのタイプは、CICS 領域の **STGPROT** および **RENTPGM** システム初期設定パラメーターの設定によって異なることがあります。

- CDSA、ECDSA、ETDSA および GCDSA 用のストレージは、常に CICS キー・ストレージから割り振られます。
- **STGPROT** システム初期設定パラメーターは、CICS 領域でストレージ保護を行うかどうかを指定します。

STGPROT=YES を指定するか、システム初期設定パラメーターをデフォルトにすると、ユーザー・アプリケーション用の CICS 動的ストレージ域のストレージは、ユーザー・キー・ストレージから割り振られます。これらの DSA は、UDSA、SDSA、EUDSA、ESDSA、GUDSA、および GSDSA です。**STGPROT=NO** を指定した場合、これらの DSA のストレージは CICS キー・ストレージから割り振られます。

- **RENTPGM** パラメーターは、CICS が読み取り専用キー 0 保護ストレージから読み取り専用 DSA を割り振るかどうかを指定します。

RENTPGM=PROTECT を指定した場合、読み取り専用 DSA が読み取り専用キー 0 保護ストレージから割り振られます。これらの DSA は、RDSA および ERDSA です。**RENTPGM=NOPROTECT** を指定した場合、これらの DSA のストレージは CICS キー・ストレージから割り振られます。

動的ストレージ域が小さすぎると、プログラム圧縮が増加し、さらに重大になると、ストレージ不足 (SOS) 状態が発生します。CICS ストレージ・マネージャー統計を使用して、仮想記憶に対するプレッシャーを調べることができます。この統計では、CICS がストレージ不足状態になった回数が報告されます。

24 ビット・ストレージ内の DSA: CDSA、UDSA、SDSA、および RDSA

16 MB 境界より下の CICS 動的ストレージ域 (DSA) は、24 ビット・ストレージにあります。これらのストレージ域には、総称名はありません。CICS システム初期設定パラメーター **DSALIM** は、これらの動的ストレージ域の合計サイズに対する制限を指定します。

システム初期設定時に、**DSALIM** 値で指定された量のストレージが、保証されたストレージとして割り振られます。このストレージ内で、CICS は以下の動的ストレージ域を自動的に管理します。ユーザーは個別のサイズを指定する必要はありません。

CDSA (CICS DSA)

すべての再入不可 CICS キー RMODE(24) プログラム、24 ビット・ストレージ内のすべての CICS キー・タスク存続期間ストレージ、および 24 ビット・ストレージに常駐する CICS 制御ブロック用のストレージ域。CDSA は、常に CICS キー・ストレージから割り振られます。

UDSA (ユーザー DSA)

24 ビット・ストレージ内のすべてのユーザー・キー・タスク存続期間ストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、UDSA はユーザー・キー・ストレージから割り振られます。**STGPROT=NO** を指定した場合、UDSA は CICS キー・ストレージから割り振られます。

SDSA (共用 DSA)

再入不可ユーザー・キー RMODE(24) プログラム用のストレージ域、およびプログラムで SHARED オプションを指定して 24 ビット・ストレージに対して CICS GETMAIN コマンドを発行して取得されるストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、SDSA はユーザー・キー・ストレージから割り振られます。**STGPROT=NO** を指定した場合、SDSA は CICS キー・ストレージから割り振られます。

RDSA (読み取り専用 DSA)

24 ビット・ストレージ内のすべての再入可能プログラムおよびテーブル用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **RENTPGM=PROTECT** を指定した場合 (これはデフォルトです)、RDSA は読み取り専用キー 0 保護ストレージから割り振られます。**RENTPGM=NOPROTECT** を指定した場合、RDSA は CICS キー・ストレージから割り振られます。

31 ビット・ストレージ内の DSA: ECDSA、EUDSA、ESDSA、ERDSA、および ETDSA

16 MB 境界より上 (16 MB より上、ただし 2 GB より下) の CICS 動的ストレージ域のグループは、総称して拡張動的ストレージ域 (EDSA) と呼ばれます。このストレージは、31 ビット・ストレージです。CICS システム初期設定パラメーター **EDSALIM** は、これらの動的ストレージ域の合計サイズに対する制限を指定します。

システム初期設定時に、**EDSALIM** 値で指定された量のストレージが、保証されたストレージとして割り振られます。このストレージ内で、CICS は以下の動的ストレージ域を自動的に管理します。ユーザーは個別のサイズを指定する必要はありません。

ECDSA (拡張 CICS DSA)

すべての再入不可 CICS キー RMODE(ANY) プログラム、31 ビット・ストレージ内のすべての CICS キー・タスク存続期間ストレージ、および 31 ビット・ストレージに常駐する CICS 制御ブロック用のストレージ域。ECDSA は、常に CICS キー・ストレージから割り振られます。

EUDSA (拡張ユーザー DSA)

31 ビット・ストレージ (16 MB 境界より上) 内のすべてのユーザー・キー・タスク存続期間ストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、EUDSA はユーザー・キー・ストレージから割り振られます。**STGPROT=NO** を指定した場合、EUDSA は CICS キー・ストレージから割り振られます。

ESDSA (拡張共用 DSA)

再入不可ユーザー・キー RMODE(ANY) プログラム用のストレージ域、およびプログラムで SHARED オプションを指定して 31 ビット・ストレージに対して CICS GETMAIN コマンドを発行して取得されるストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、ESDSA はユーザー・キー・ストレージから割り振られます。**STGPROT=NO** を指定した場合、ESDSA は CICS キー・ストレージから割り振られます。

ERDSA (拡張読み取り専用 DSA)

31 ビット・ストレージ内のすべての再入可能プログラムおよびテーブル用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **RENTPGM=PROTECT** を指定した場合 (これはデフォルトです)、ERDSA は読み取り専用キー 0 保護ストレージから割り振られます。**RENTPGM=NOPROTECT** を指定した場合、ERDSA は CICS キー・ストレージから割り振られます。

ETDSA (拡張信頼 DSA)

31 ビット・ストレージに常駐するセキュリティ関連 CICS 制御ブロック用のストレージ域。ETDSA は、常に CICS キー・ストレージから割り振られます。

64 ビット・ストレージ内の DSA: GCDSA、GUDSA、および GSDSA

2 GB 境界より上の CICS 動的ストレージ域は、総称して 2 GB 境界より上の動的ストレージ域 (GDSA) と呼ばれます。このストレージは、64 ビット・ストレージです。z/OS **MEMLIMIT** パラメーターは、GDSA を含めて、CICS 領域内の 64 ビット・ストレージを制限します。

z/OS オペレーティング・システムによって CICS アドレス・スペースに割り当てられる **MEMLIMIT** 値が、CICS 領域内の 64 ビット・ストレージの上限を制御します。この 64 ビット・ストレージには GDSA と、GDSA 外部の CICS 領域内の MVS ストレージの両方が含まれます。

これに対して、**DSALIM** 値で指定される 24 ビット・ストレージおよび **EDSALIM** 値で指定される 31 ビット・ストレージは、CICS DSA にのみ関係します。

GDSA には、保証された量のストレージは事前割り振りされません。GDSA には、以下の動的ストレージ域が含まれます。

GCDSA (2 GB 境界より上の CICS DSA)

64 ビット (2 GB 境界より上) ストレージのすべての CICS キー・タスク存続期間ストレージ用、および 64 ビット・ストレージを使用する CICS 機能用のストレージ域。89 ページの『[64 ビット・ストレージを使用する CICS 機能](#)』を参照してください。GCDSA は、常に CICS キー・ストレージから割り振られます。

GUDSA (2 GB 境界より上のユーザー DSA)

64 ビット (2 GB 境界より上) ストレージ内のすべてのユーザー・キー・タスク存続期間ストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、GUDSA はユーザー・キー・ストレージから割り振られます。STGPROT=NO を指定した場合、GUDSA は CICS キー・ストレージから割り振られます。

GSDSA (2 GB 境界より上の共用 DSA)

SHARED オプションを使って 64 ビット・ストレージを取得するために **EXEC CICS GETMAIN64** コマンドを発行してプログラムが獲得するすべてのストレージ用のストレージ域。CICS 領域に対してシステム初期設定パラメーター **STGPROT=YES** を指定した場合 (デフォルト)、GSDSA はユーザー・キー・ストレージから割り振られます。STGPROT=NO を指定した場合、GSDSA は CICS キー・ストレージから割り振られます。

ストレージ保護

CICS はオペレーティング・システムで使用可能なストレージ保護機能を使用します。これによって CICS コードおよび制御ブロックが、ユーザーのアプリケーション・プログラムによって上書きされないようにすることができます。これを実現するため、ユーザー・アプリケーション・プログラム用、および CICS コードと制御ブロック用に、それぞれ別個のストレージ・キーを伴う別個の動的ストレージ域 (DSA) が割り振られます。ストレージ域へのアクセスは、アクセス・キーが対象のストレージ域のキーと一致しない限り許可されません。

ほとんどの CICS コードおよび制御ブロック用に割り振られるストレージは、CICS キー・ストレージと呼ばれ、ユーザー・アプリケーション・プログラム用に割り振られるストレージはユーザー・キー・ストレージと呼ばれます。

CICS キー・ストレージとユーザー・キー・ストレージのほかに、読み取り専用 DSA (RDSA および ERDSA) と呼ばれる動的ストレージ域を分離するために、CICS ではユーザー・キー 0 ストレージも使用されます。ERDSA は、RENT 属性および RMODE(ANY) 属性を使用してリンク・エディットされた適格な再入可能 CICS プログラムおよびユーザー・アプリケーション・プログラムに使用されます。RDSA は、RENT 属性および RMODE(24) 属性を使用してリンク・エディットされた適格な再入可能 CICS プログラムおよびユーザー・アプリケーション・プログラムに使用されます。読み取り専用 DSA 用のキー 0 ストレージの割り振りには、**DSALIM** および **EDSALIM** システム初期設定パラメーターで指定された、他の DSA と同じストレージ制限が適用されます。

ストレージ保護機能の使用はオプションです。ストレージ保護に関連したシステム初期設定パラメーターでオプションを使用して、この機能を使用可能にすることができます。例えば、これらのパラメーターを使用して、以下の項目を定義または制御できます。

- 共通作業域のストレージ・キー (**CWAKEY**)
- 端末管理テーブル・ユーザー域のストレージ・キー (**TCTUAKEY**)
- ストレージ保護グローバル・オプション (**STGPROT**)

- ・読み取り専用プログラム・ストレージ・キー・オプション (**RENTPGM**)
- ・トランザクション分離オプション (**TRANISO**)

ストレージ保護、トランザクション分離、およびコマンド保護では、ユーザー・アプリケーション・コードからストレージが保護されます。ユーザー・コードが実行されない領域、つまり、純粋な端末専有領域 (TOR) または純粋なファイル専有領域 (FOR) には、利益は追加されません (分散プログラム・リンク (DPL) 要求が機能シップされない場合)。

共通作業域 (CWA)

共通作業域 (CWA) とは、すべてのユーザー・アプリケーションがアクセスできる CICS 領域内のストレージ域のことです。**WRKAREA** システム初期設定パラメーターを使用して、この作業域のサイズを決定します。サイズは最大で 3584 バイトまで指定できます。

WRKAREA パラメーターを省略した場合、デフォルトでは、CICS は 512 バイトの CWA を割り振ります。

CWAKEY パラメーターで、CWA のストレージ・キーを指定します。

この作業域は CICS 領域内のすべてのトランザクションが使用できるため、ストレージ・キーが、すべてのトランザクションで CWA を使用するのに適したものであることを確認する必要があります。ユーザー・キーで実行されるトランザクションが書き込み権限を必要とする場合、CWA 用のユーザー・キー・ストレージを指定する必要があります。指定しないと、そのようなトランザクションはストレージ保護例外 (ASRA 異常終了) で失敗します。デフォルトでは、CICS は CWA にユーザー・キー・ストレージを取得します。これを CICS キー・ストレージに変更する場合は、それを決定する前に、すべてのプログラムによるこのストレージの使用を検討する必要があります。

CWA が書き込み権限のないアプリケーションによって上書きされないように保護することができます。この場合、CWA への書き込みアクセスを正当に必要とするすべてのアプリケーションが CICS キーで実行されるのであれば、CWA に対して CICS キー・ストレージを指定できます。

端末管理テーブル・ユーザー域

端末管理テーブル・ユーザー域 (TCTUA) は、端末管理テーブルの端末入力 (TCTTE) に関連した任意指定のストレージ域です。TCTUA は、アプリケーション・プログラム用に使用可能です。**TCTUAKEY** システム初期設定パラメーターを使用して、TCTUA 用のストレージ・キーを CICS 領域に対してグローバルに指定します。

デフォルトでは、CICS はすべての TCTUA に対してユーザー・キー・ストレージを取得します。CICS 領域内の TCTUA の使用を検討し、これが正当であると確信できる場合にのみ、TCTUA に対して CICS キーを指定してください。TCTUA に対して CICS キー・ストレージを指定すると、ユーザー・キー・アプリケーションはどの TCT ユーザー域にも書き込むことができません。

SNA LU の場合、TYPETERM リソース定義の USERAREALEN 属性を使用して、TCTUA が必要であることを指定します。USERAREALEN 属性は、TYPETERM リソース定義を参照するすべての端末の TCTUA サイズを決定します。

順次端末の場合、定義が端末管理テーブル (TCT) に追加され、サイズは DFHTCT TYPE=TERMINAL 項目および TYPE=LINE 項目の **TCTUAL** パラメーターを使用して定義されます。TCTUAL パラメーターについては、[Terminal control table \(TCT\)](#) を参照してください。

ストレージ保護グローバル・オプション

STGPROT システム初期設定パラメーターの指定により、CICS 領域でストレージ保護を使用するかどうかを制御できます。デフォルトでは、CICS はストレージ保護を使用します。

STGPROT=YES を指定するか、システム初期設定パラメーターをデフォルトにすると、ユーザー・アプリケーション用の CICS 動的ストレージ域のストレージは、ユーザー・キー・ストレージから割り振られます。これらの DSA は、UDSA、SDSA、EUDSA、ESDSA、GUDSA、および GSDSA です。**STGPROT=NO** を指定した場合、これらの DSA のストレージは CICS キー・ストレージから割り振られます。

ストレージ保護なし (**STGPROT=NO**) で CICS 領域を実行することは、ユーザー・トランザクションを実行しない純粋な端末専有領域 (TOR) に適しています。

トランザクション分離グローバル・オプション

CICS トランザクション分離は、CICS ストレージ保護の上に構築され、ユーザー・トランザクションを相互に保護できるようにします。**TRANISO** システム 初期設定パラメーターを使用して、トランザクション分離を CICS 領域に対してグローバルに指定できます。

ユーザー・トランザクションごとに個別にストレージ・キーおよび実行キーを指定できるほかに、CICS では、トランザクション間の保護を提供するために、トランザクションのユーザー・キー・タスク存続期間ストレージを分離することを指定できます。これを行うには、TRANSACTION リソース定義の **ISOLATE** オプションを使用します。

トランザクション分離は、64 ビット・ストレージには適用されません。

読み取り専用ストレージ・オーバーライド・オプション

CICS は、読み取り専用 DSA (RDSA および ERDSA) 用のストレージを、MVS 読み取り専用ストレージから取得します。**RENTPGM** システム 初期設定パラメーターで **NOPROTECT** を指定すると、RDSA および ERDSA の読み取り専用ストレージの選択をオーバーライドできます。

CICS ローダーは、適格なモジュールを RDSA および ERDSA に自動的にロードします。つまり、モジュールが RENT 属性、および ERDSA の場合は RMODE(ANY) を使用してリンク・エディットされている場合です。アプリケーション・プログラムにブレークポイントを設定する開発援助プログラム・パッケージを使用しているなどの理由で、こうしたモジュールを読み取り専用ストレージにロードしたくない場合は、**RENTPGM=NOPROTECT** を指定できます。**RENTPGM=NOPROTECT** を指定しても、CICS は読み取り専用 DSA を割り振りますが、RDSA および ERDSA 用に、読み取り専用ストレージの代わりに CICS キー・ストレージを取得します。

RENTPGM=NOPROTECT によるオーバーライドは、開発領域に対してのみ適切です。実動 CICS 領域では、**RENTPGM=PROTECT** により、RDSA および ERDSA 内のモジュールに対して適切なレベルの保護が提供されます。

CICS ストレージの制限の設定

z/OS **REGION** パラメーターおよび **MEMLIMIT** パラメーターを使用して、CICS 領域のストレージに対する制限を設定します。CICS **DSALIM** および **EDSALIM** システム 初期設定パラメーターを使用して、24 ビット および 31 ビット・ストレージ内の CICS 動的ストレージ域 (DSA) の合計ストレージを制限します。

このタスクについて

z/OS **REGION** パラメーターおよび **MEMLIMIT** パラメーターは、CICS 領域が稼働するアドレス・スペースの仮想記憶の量を指定します。

- **z/OS REGION** パラメーターを使用して、CICS 領域用に要求する 24 ビット (16 MB 境界より下) および 31 ビット (16 MB 境界より上) ストレージの量を指定します。
- **z/OS MEMLIMIT** パラメーターを使用して、CICS 領域の 64 ビット (2 GB 境界より上) ストレージの量を制限します。

MEMLIMIT 値で指定された 64 ビット・ストレージには、2 GB 境界より上の CICS 動的ストレージ域 (GDSA) と、GDSA 外部の CICS 領域内の MVS ストレージが含まれます。

このパラメーターの詳細については、[77 ページの『CICS 領域サイズ』](#)を参照してください。

REGION 値で指定されたストレージには、CICS システム 初期設定パラメーター **DSALIM** および **EDSALIM** で指定されたストレージが含まれます。これらのパラメーターは、CICS が CICS DSA 用のストレージを割り振ることができる全体的な制限を決定します。

- **DSALIM** パラメーターを使用して、24 ビット (16 MB 境界より下) ストレージ内の CICS DSA の全体的な制限を設定します。
- **EDSALIM** パラメーターを使用して、拡張動的ストレージ域 (EDSA)、つまり 31 ビット (16 MB 境界より上) ストレージ内の CICS DSA の全体的な制限を設定します。

CICS は、個々の動的ストレージ域を自動的に割り振ります。ユーザーはそれぞれのサイズを指定する必要はありません。CICS は、必要が生じると、個々の動的ストレージ域のサイズを変更します。

- CICS は、**DSALIM** で設定された制限内で 24 ビット・ストレージに DSA を割り振ります。

- CICS は、**EDSALIM** で設定された制限内で 31 ビット・ストレージに DSA を割り振ります。
- CICS は、**MEMLIMIT** で設定された制限内で 64 ビット・ストレージに DSA を割り振ります。

手順

- z/OS パラメーター **REGION** の設定の見積もりおよび変更を行うには、[83 ページの『REGION の見積もりおよび設定』](#)を参照してください。
- z/OS パラメーター **MEMLIMIT** の設定の見積もりと変更、および実行中の CICS システムでの値の確認を行うには、[87 ページの『MEMLIMIT の見積もり、確認、および設定』](#)を参照してください。
- CICS システム 初期設定パラメーター **DSALIM** の設定の見積もりおよび変更を行うには、[84 ページの『DSALIM の見積もり、確認、および設定』](#)を参照してください。
- CICS システム 初期設定パラメーター **EDSALIM** の設定の見積もりおよび変更を行うには、[85 ページの『EDSALIM の見積もり、確認、および設定』](#)を参照してください。

REGION の見積もりおよび設定

z/OS **REGION** パラメーターは、CICS アドレス・スペースが使用できる 24 ビットおよび 31 ビット・ストレージ (2 GB 境界より下のストレージ) の量を制限します。この値には、専用領域内の 2 GB 境界より下のすべてのストレージ (24 ビット・ストレージ内の 16 KB システム領域を除く) と、LSQA などの高専用域内の項目が含まれます。

このタスクについて

2 GB より下の z/OS アドレス・スペース内のストレージ域についての説明は、[「z/OS MVS 初期設定およびチューニングガイド」の『仮想記憶アドレス・スペース』](#)を参照してください。

CICS 領域用に 2 GB 境界より下に最大 2047 MB のストレージを要求できますが、MVS が高専用領域で使用するための十分なストレージが 2 GB 境界より下に残されていることを確認する必要があります。高専用領域内の項目は、ローカル共用キュー域 (LSQA)、スケジューラー作業域 (SWA)、およびサブプール 229 と 230 です。これらの項目は、24 ビット (16 MB 境界より下) ストレージと 31 ビット (16 MB 境界より上) ストレージの両方に存在します。31 ビット・ストレージ内の LSQA は、拡張 LSQA と呼ばれます。このストレージの一部は制御ブロック用に使用され、一部は z/OS Communications Server プログラムおよびその他のプログラムによって使用されます。

REGION に指定する値の中には、以下のタイプのストレージが含まれます。

- 24 ビット・ストレージ内の CICS DSA。これらの DSA 用のストレージは、**DSALIM** システム 初期設定パラメーターによって制限されます。
- 31 ビット・ストレージ内の CICS DSA。これらの DSA 用のストレージは、**EDSALIM** システム 初期設定パラメーターによって制限されます。
- CICS カーネルによって使用されるストレージ。
- MVS GETMAIN 要求によって取得される MVS ストレージ。
- CICS ディスパッチャー
- CICS ストレージ・マネージャー
- CICS ロック・マネージャー

CICS の実行中に、CICS 領域の **REGION** 値を変更することはできません。CICS 領域の次の始動時に、新規の値を指定できます。

手順

1. **REGION** パラメーターの最大値を判別するには、以下を実行します。
 - a) RMF または別のストレージ・モニターを使用して、専用領域のサイズを調べます。
 - b) 次の公式を適用してください。

最大可能 REGION =
専用領域のサイズ

- システム領域のサイズ (16K)
- (LSQA + SWA + サブプール 229 および 230)

高専用領域および LSQA の説明と、高専用領域内の項目のサイズの見積もりについて詳しくは、[123 ページの『高専用領域』](#)を参照してください。

- c) 安全のために、**REGION** パラメーターに対してこの最大値の 80% または 90% を超える値を使用しないでください。

CICS の上部と高専用領域の下部の間に一定量のフリー・ストレージを維持すると役立ちます。

システムが静的あるか、あまり変更されない場合、この数値の最大 90% を使用します。システムが動的であるか、頻繁に変更される場合は、80% にするのが適しています。

2. ご使用のストレージのニーズを満たすために必要な **REGION** パラメーターの値を見積もるには、以下のストレージ域の見積もりを加算します。

- **DSALIM** システム初期設定パラメーターで指定された、16 MB 境界より下の CICS DSA 用の 24 ビット・ストレージ域。 [84 ページの『DSALIM の見積もり、確認、および設定』](#)を参照してください。
- **EDSALIM** システム初期設定パラメーターで指定された、16 MB 境界より上の CICS DSA (EDSA) 用の 31 ビット・ストレージ域。 [85 ページの『EDSALIM の見積もり、確認、および設定』](#)を参照してください。
- CICS DSA の外部の CICS カーネルによって使用される少量のストレージ。 [118 ページの『CICS カーネル・ストレージ』](#)を参照してください。
- CICS DSA の外部の MVS GETMAIN 要求によって取得される MVS ストレージ。 [119 ページの『2 GB より下の MVS ストレージ』](#)を参照してください。

3. **REGION** パラメーターの指定について、およびユーザーの要求に応じて z/OS が割り振るストレージの量について詳しくは、[『z/OS MVS JCL Reference』の『REGION パラメーター』](#)を参照してください。

実行中の CICS 領域の **REGION** 値を変更することはできません。

REGION は、次のように設定することができます。

- CICS JCL の JOB ステートメントで **REGION** パラメーターを指定できます。この場合、ジョブの各ステップは要求されたスペース量で実行されます。
- CICS の EXEC ステートメント (プログラム実行の行) で **REGION** パラメーターを指定できます。この場合、各ステップは独自のスペース量で実行されます。さまざまなステップで必要とされるスペース量が大きく異なる場合は、JOB ステートメントの代わりに EXEC ステートメントを使用してください。例えば、CICS のシャットダウン後に補助トレース・データ・セットを印刷するための追加のジョブ・ステップを使用している場合 (DFHIVPOL インストール検査手順の場合のように)、EXEC ステートメントを使用できます。
- z/OS インストール・システム出口 IEFUSI は、指定する **REGION** 値を制限することがあります。IEFUSI については、[『z/OS MVS Installation Exits』の『IEFUSI - 手順開始出口』](#)を参照してください。

DSALIM の見積もり、確認、および設定

DSALIM システム初期設定パラメーターは、CICS が 24 ビット・ストレージ (16 MB 境界より下) に常駐する個別の動的ストレージ域 (DSA) を割り振ることができる、ストレージの総量の上限を指定します。ご使用のシステムで 24 ビット・ストレージが制約されている場合は、**DSALIM** パラメーターに対して CDSA と UDSA の合計に相当する値を設定してください。より大きい **DSALIM** 値を許容できる十分な仮想記憶がある場合は、ここに示す公式を使用できます。

このタスクについて

DSALIM 値の正確な見積もりは、あまり重要ではありません。**DSALIM** 値は、予想される所要量より少し小さい値ではなく、少し大きい値を指定することをお勧めします。実行中のシステムからデータを取得した後で、**DSALIM** パラメーターをより小さい値に調整できます。

他のサブシステム問題が生じないようにするために、CICS DSA 外部の 24 ビット・ストレージ内の MVS ストレージの要件を把握しておいてください。MVS ストレージを使用するオペレーティング・システム・コンポーネントについて詳しくは、[119 ページの『2 GB より下の MVS ストレージ』](#)を参照してください。

DSALIM の最小値は 2 MB、デフォルト値は 5 MB です。最大 **DSALIM** 値は 16 MB です。CDSA、RDSA、および SDSA のエクステント・サイズの増分値は 256 KB です。トランザクション分離がアクティブな場合、UDSA のエクステント・サイズは 1 MB であるため、各 UDSA エクステントは 1MB 単位になるように調整する必要があります。トランザクション分離がアクティブでない場合、割り振りは 256 KB エクステント単位で行われます。

CICS は、必要なときに、24 ビット・カーネル・スタック・ストレージを割り振ります。タスクは、24 ビット・スタック・ストレージが必要になるたびに、4 KB 拡張スタック・セグメントを取得します。CICS は、使用可能な 24 ビット・スタック・ストレージが他にない場合にタスクが使用できる、24 ビット拡張スタック・セグメントの予約プールを事前割り振りします。カーネル・ストレージについての詳細は、[118 ページの『CICS カーネル・ストレージ』](#)を参照してください。

手順

- 実行中の CICS 領域に現在適用されている **DSALIM** 値を確認するには、次のいずれかの方法を使用します。
 - CICS Explorer: 「**Global Dynamic Storage Areas (グローバル動的ストレージ域)**」ビュー
 - CICSplex SM: 「**Dynamic storage areas - CICSDSA (動的ストレージ域 - CICSDSA)**」ビュー
 - CEMT: **CEMT INQUIRE DSAS** または **CEMT INQUIRE SYSTEM**
 - CICS SPI: **INQUIRE SYSTEM**
- 適切な **DSALIM** 値を見積もるには、以下の手順を使用します。
 - a) 十分な **DSALIM** 値を指定するだけの十分な仮想記憶がある場合は、次の公式を使用して、値を見積もります。
$$\text{CDSA} + \text{UDSA} + \text{SDSA} + \text{RDSA}$$

計算に含まれる各コンポーネントの値を、256 KB 境界に切り上げます。
 - b) 現在のインストール・システムの **DSALIM** 値が必要以上に大きい場合は、次の公式を使用して **DSALIM** 値を見積もります。
$$\text{ピーク CDSA 使用} + \text{ピーク UDSA 使用} + \text{ピーク SDSA 使用} + \text{ピーク RDSA 使用}$$

計算に含まれる各コンポーネントの値を、256 KB 境界に切り上げます。
- CICS 領域の **DSALIM** 値を変更するには、CICS の実行中に、以下のいずれかの方法を使用します。
 - CICS Explorer: 「**Global Dynamic Storage Areas (グローバル動的ストレージ域)**」ビュー
 - CICSplex SM: 「**Dynamic storage areas - CICSDSA (動的ストレージ域 - CICSDSA)**」ビュー
 - CEMT: **CEMT SET DSAS** または **CEMT SET SYSTEM**
 - CICS SPI: **SET SYSTEM**

タスクの結果

24 ビット・ストレージ内の CICS DSA にフリーのエクステントがない場合は、CICS は **DSALIM** の縮小を実装できません。ストレージ・マネージャーは、エクステントが使用可能になると、新規の **DSALIM** 値に達するまで、エクステントに対して MVS FREEMAIN 要求を適用します。

DSALIM を削減すると、ストレージ不足状態が発生することがあります。

EDSALIM の見積もり、確認、および設定

EDSALIM システム初期設定パラメーターは、CICS が 31 ビット (16 MB 境界より上) ストレージに常駐する個別の拡張動的ストレージ域 (EDSA) を割り振ることができる、ストレージの総量の上限を指定します。

EDSALIM パラメーターは、他の領域、特に MVS ストレージを考慮した上で、可能な限り大きい値に設定してください。

このタスクについて

EDSALIM に指定できる最大値は、以下の要因によって制限されます。

- CICS ジョブまたはプロシーチャーの **z/OS REGION** パラメーターで CICS 領域に指定したサイズ。

EDSALIM の値は、**REGION** の値より小さくなくてはなりません。

- 31 ビット・ストレージに対する MVS GETMAIN 要求を満たすために必要な、CICS DSA 外部の MVS ストレージの量。MVS ストレージを使用するオペレーティング・システム・コンポーネントについては、[119 ページの『2 GB より下の MVS ストレージ』](#)を参照してください。

EDSALIM 値の正確な見積もりは、あまり重要ではありません。推奨されるアプローチは、以下のとおりです。

- 最初は、予想される所要量より少し大きい **EDSALIM** 値を指定します。デフォルトの設定値 800 MB なら、CICS 領域は程々のワークロードを実行できます。
- ピーク・ロード付近でシステムを実行しながら、EDSA 内の各 CICS DSA の使用をモニターします。
- 実行中の CICS システムで **EDSALIM** 値を調整します。

最大可能 **EDSALIM** 値 (例えば、最大許容領域サイズ) を指定しないようにしてください。最大可能限度を使用すると、問題の解決が難しくなるまで、仮想記憶の不足について警告を受け取らない可能性があります。

CICS ストレージ・マネージャー統計を見ると、現在の EDSA の使用に関する情報を入手できます。ストレージ・マネージャー統計、動的ストレージ域、およびタスク・サブプール内の DSA サイズに関する情報を参照してください。自動 DSA サイジングを使用することによって、CICS が要求に応じて DSA のサイズを動的に変更するので、個々の DSA に対する正確なストレージ見積もりが不要になります。

EDSALIM の最小値は 48 MB です。これは、CICS 領域を始動するために最小限必要な値です。**EDSALIM** のデフォルト値は 800 MB です。最大 **EDSALIM** サイズは 2047 MB です。これは、2 GB から 1 MB を減算した値です。ECDSA、EUDSA、ESDSA、ERDSA、および ETDSA のエクステント・サイズは、1 MB です。

カーネル・スタック・ストレージも EDSA から割り振られます。カーネル・ストレージについての詳細は、[118 ページの『CICS カーネル・ストレージ』](#)を参照してください。

トランザクション分離をアクティブにして稼働する CICS 領域 (**TRANISO** システム初期設定パラメーターを使用して設定) の場合、トランザクション分離機能はいくつかの仮想記憶域の割り振りを 16 MB 以上に増やすことに注意してください。

トランザクション分離がアクティブである場合、CICS はユーザー・キー・タスク保存期間ストレージを 1 MB (トランザクション分離がアクティブであるとき、EUDSA のストレージ割り振りの最小単位は 1 MB です) の倍数で 16 MB 以上に割り振ります。ただし、MVS ページング・アクティビティーは、使用された (参照された) ストレージにのみ影響を与え、1 MB の割り振りの未使用部分はページングされません。

トランザクション分離なしで稼働する CICS 領域の場合、CICS はユーザー・キー・タスク存続期間ストレージを 64 KB の倍数で 16 MB 以上に割り振ります。

MVS が各サブスペースの実記憶域からページおよびセグメント・テーブルを作成するため、サブスペース・グループ機能は、より多くの実記憶域を使用します。CICS が必要とする実記憶域は、トランザクション負荷によって変わります。指針として、システムの各タスクには 9 KB の実記憶域が必要であり、一時にシステムに存在できる同時タスクの数 (**MXT** システム初期設定パラメーターで制御) で乗算した数である必要があります。

手順

- 実行中の CICS 領域に現在適用されている **EDSALIM** 値を確認するには、次のいずれかの方法を使用します。
 - CICS Explorer: 「**Global Dynamic Storage Areas (グローバル動的ストレージ域)**」ビュー
 - CICSplex SM: 「**Dynamic storage areas - CICSDSA (動的ストレージ域 - CICSDSA)**」ビュー
 - CEMT: **CEMT INQUIRE DSAS** または **CEMT INQUIRE SYSTEM**
 - CICS SPI: **INQUIRE SYSTEM**
- 適切な **EDSALIM** 値を見積もるには、以下の手順を使用します。
 - a) ご使用の CICS 領域の **MXT** 値を確認します。

MXT システム 初期設定パラメーターには CICS システム・タスクは含まれていません。また、必要以上に大きい値に設定されている場合もあります。**EDSALIM** 値を計算するための最も安全な見積もりは、**MXT** を同時アクティブ・タスクの数として想定することです。

- b) ご使用の CICS 領域の **TRANISO** システム 初期設定パラメーターの設定を確認します。

EUDSA の場合、**TRANISO** パラメーターが CICS 領域に対して NO に設定されている場合は、同時アクティブ・タスク当たり 64 KB を見込むことができます。**TRANISO** パラメーターが YES に設定されている場合は、同時アクティブ・タスク当たり 1 MB を見込むことができます。

ご使用のアプリケーションで、**TRANISO** パラメーターを NO に設定した状態でタスク当たり 64 KB を超える量を使用していたり、**TRANISO** パラメーターを YES に設定した状態でタスク当たり 1 MB を超える量を使用している場合は、それに応じて公式を調整してください。公式を調整する場合は、64 KB または 1 MB の倍数を使用してください。

- c) 十分な **EDSALIM** 値を指定するだけの十分な仮想記憶がある場合は、以下のいずれかの公式を使用して、値を見積もります。

計算に含まれる各コンポーネントの値を、エクステントのサイズである 1 MB 境界に切り上げて、フラグメント化および部分的に使用されたエクステントを考慮に入れます。

TRANISO=NO の場合:

$$\text{ECDSA} + \text{EUDSA} + \text{ESDSA} + \text{ERDSA} + \text{ETDSA} + (64 \text{ K} \times \text{MXT})$$

TRANISO=YES の場合:

$$\text{ECDSA} + \text{EUDSA} + \text{ESDSA} + \text{ERDSA} + \text{ETDSA} + (1 \text{ MB} \times \text{MXT})$$

- d) 現在のインストール・システムの **EDSALIM** 値および **MXT** 値が不必要に大きい場合は、以下のいずれかの公式を使用して、**EDSALIM** 値を見積もります。

計算に含まれる各コンポーネントの値を、エクステントのサイズである 1 MB 境界に切り上げて、フラグメント化および部分的に使用されたエクステントを考慮に入れます。

TRANISO=NO の場合:

$$\begin{aligned} & \text{ピーク ECDSA 使用} + \text{ピーク EUDSA 使用} + \text{ピーク ESDSA 使用} + \text{ピーク ERDSA 使用} + \text{ピーク} \\ & \text{ETDSA 使用} - \text{タスク・サブプール内の EUDSA ピーク・ページ・ストレージ} + (64 \text{ K} \times \text{タスクのピーク数}) \end{aligned}$$

TRANISO=YES の場合:

$$\begin{aligned} & \text{ピーク ECDSA 使用} + \text{ピーク EUDSA 使用} + \text{ピーク ESDSA 使用} + \text{ピーク ERDSA 使用} + \text{ピーク} \\ & \text{ETDSA 使用} - \text{タスク・サブプール内の EUDSA ピーク・ページ・ストレージ} + (1 \text{ MB} \times \text{タスクのピーク数}) \end{aligned}$$

- CICS 領域の **EDSALIM** 値を変更するには、CICS の実行中に、以下のいずれかの方法を使用します。

- CICS Explorer: 「**Global Dynamic Storage Areas (グローバル動的ストレージ域)**」ビュー
- CICSplex SM: 「**Dynamic storage areas - CICSDSA (動的ストレージ域 - CICSDSA)**」ビュー
- CEMT: **CEMT SET DSAS** または **CEMT SET SYSTEM**
- CICS SPI: **SET SYSTEM**

タスクの結果

EDSALIM を小さめに指定すると、システムがストレージ不足になり、CICS コマンドを発行して制限値を大きくすることができなくなる場合があります。この状態の場合は、CICS Explorer または CICSplex SM を使用して、**EDSALIM** 値を増やしてください。

CICS DSA 内にフリーのエクステントがない場合は、CICS は **EDSALIM** の縮小を実装できません。ストレージ・マネージャーは、エクステントが使用可能になると、新規の **EDSALIM** 値に達するまで、エクステントに対して MVS FREEMAIN 要求を適用します。

MEMLIMIT の見積もり、確認、および設定

z/OS **MEMLIMIT** パラメーターは、CICS アドレス・スペースで使用できる 64 ビット (2 GB 境界より上) のストレージの量を制限します。このストレージには、2 GB 境界より上の CICS 動的ストレージ域 (総称 GDSA) と、GDSA 外部の CICS 領域内の MVS ストレージが含まれます。

このタスクについて

CICS には、10 GB の **MEMLIMIT** 値が必要です。アプリケーションまたは JVM でさらに使用する場合は、**MEMLIMIT** をそれより大きな値にする必要があります。**MEMLIMIT** 値が 10 GB 未満の状態では CICS 領域を始動しようとすると、メッセージ DFHSM0602 が発行され、ダンプ・コード KERNDUMP のシステム・ダンプが生成されて、CICS は終了します。

CICS の実行中に、CICS 領域の **MEMLIMIT** 値を変更することはできません。CICS 領域の次の始動時に、新規の **MEMLIMIT** 値を指定できます。

CICS は、新しい GDSA エクステンツが割り振り可能かどうかを調べるために **MEMLIMIT** を検査するときに、z/OS とは異なる計算を使用します。CICS は、現在の使用量を、64 ビット・メモリー・オブジェクトに割り振られた量 (使用可能ストレージと隠しストレージの両方を含む) として計算します。

手順

- **MEMLIMIT** を設定するときは、2 GB 境界より上の CICS SOS を避けるようにする必要があります。実際のピーク使用量が明確になったら、**MEMLIMIT** を減らすことができます。
- 実行中の CICS 領域に現在適用されている **MEMLIMIT** 値を確認するには、次のいずれかの方法を使用します。
 - CICS Explorer: 「**Regions (領域)**」ビューまたは「**Global Dynamic Storage Areas (グローバル動的ストレージ域)**」ビュー
 - CICSplex SM: 「**Dynamic storage areas - CICSDSA (動的ストレージ域 - CICSDSA)**」ビューまたは「**CICS regions - CICSrgn (CICS 領域 - CICSrgn)**」ビュー
 - CEMT: **CEMT INQUIRE DSAS** または **CEMT INQUIRE SYSTEM**
 - CICS SPI: **INQUIRE SYSTEM**
 - DFHOSTAT が生成する Storage Above 2 GB のレポートには、さまざまな CICS 統計フィールドからの 64 ビット使用に関する情報が示されます。SMSMEMLIMIT - (SMSLVABYTES/1048576) は、GDSA を拡張できる量 (GB 単位になるように端数切り捨て) を示します。これは、「**MEMLIMIT minus allocated to Private Memory Objects**」(**MEMLIMIT** から専用メモリー・オブジェクトへの割り振り量を減算) として報告されます。
- CICS 領域の適切な **MEMLIMIT** 値を見積もるには、CICS 領域内で使用される 64 ビット・ストレージを使用する機能のストレージ要件を加算します。

CICS 領域および関連するストレージ・サブプール内の 64 ビット・ストレージを使用する CICS 機能のリストについては、「[パフォーマンスの改善](#)」の『[64 ビットのストレージを使用できる CICS 機能](#)』を参照してください。

CICS は、CICS 64 ビット・サブプール用のストレージを内部に 2 次割り振りする前に、1 GB の倍数の GDSA エクステンツを割り振っておく必要があります。CICS 内部トレース・テーブルは常に 1 GB のサイズで割り振られます。

CICS ストレージ・マネージャー統計に、実行中の CICS 領域の GDSA 内の各 CICS サブプールで使用されているストレージが表示されます。GDSA 内のサブプールについて詳しくは、[GDSA の CICS サブプールおよび GSDSA の CICS サブプール](#)を参照してください。

- z/OS の **MEMLIMIT** 値を変更する場合、またはセットアップする CICS 領域に適用される値を調べる場合は、「z/OS MVS Programming: Extended Addressability Guide」の『[Limiting the use of memory objects](#)』を参照してください。

MEMLIMIT は、以下のいずれかの方法で設定できます。

- **MEMLIMIT** 値は、CICS JCL の JOB ステートメント、または CICS の EXEC ステートメント (プログラム実行の行) で指定できます。
- CICS 領域に固有の **MEMLIMIT** 値がない場合は、z/OS SMFPRMxx PARMLIB メンバーで設定された **MEMLIMIT** 値、またはシステム・デフォルトが適用されます。
- z/OS インストール・システム出口 IEFUSI は、他の任意の **MEMLIMIT** 値をオーバーライドできます。

次の例は、プログラム実行の行で設定された **MEMLIMIT** 値を示しています。

```
//CICS EXEC PGM=DFHSSIP,PARM='SI',REGION=0M,MEMLIMIT=10G
```

64 ビット・ストレージを使用する CICS 機能

64 ビット・ストレージを使用する CICS 機能、および各機能に必要なストレージの量をリストしています。CICS 領域に適切な **MEMLIMIT** 値を見積もる場合に、この情報を使用できます。

次の表は、64 ビット・ストレージを使用する CICS 機能と関連 CICS ストレージ・サブプールをリストして、各機能が必要とするストレージ量を示したものです。64 ビット・ストレージを必要とする CICS 領域で使用する機能を特定できます。次に、それらの機能のストレージ要件を調べ、その情報を使用して、z/OS **MEMLIMIT** パラメーターの適切な値を見積もることができます。

表 3. 64 ビット・ストレージを使用する CICS 機能			
CICS 機能	ストレージの使用	ストレージの CICS サブプール	注
アプリケーション・コンテキスト・データ制御ブロック	アプリケーションのエントリー・ポイントとして定義される各プログラムごとに 220 バイト。	PGPSTE64	
関連データ制御ブロック	アクティブ・タスクごとに約 1 KB。	MN_ADCS	
CICS 管理クライアント・インターフェース (CMCI)	ストレージ所要量の見積もりの詳細については、 CMCI のストレージ要件の見積もり を参照してください。	WU_64	保存された結果およびメタデータのためにストレージが使用されます。
CICSplex SM API 結果セット	結果セットのサイズはアプリケーションによって異なります。	CPSM_64	結果セットについては、 結果セットの操作 を参照してください。
コンソール・キュー処理	サブプール当たり 1 MB	CQCQ_TR CQCQ_XT	
コンテナおよびチャンネル	トランザクション当たり MEMLIMIT 値の 5% に制限されます。	PGCSDB	チャンネルが有効範囲外になるか、アプリケーションがコンテナまたはチャンネルを削除するまで、ストレージは使用中のままです。
イベント処理		EP_64	イベント・キャプチャー・キュー内の項目の制御ブロック用にストレージが使用されます。
内部トレース・テーブル	最小 16 KB 最大 1 GB デフォルト 12288 KB (12 MB) TRTABSZ システム初期設定パラメーターによって制御されます。	CICS DSA の外部の MVS ストレージ	CICS は最大サイズのメモリー・オブジェクトを割り振りますが、未使用領域を隠しストレージとして定義します。
JVM サーバー	-Xmx および -Xcmsx の値を除いて、ストレージの量は可変です。	CICS DSA の外部の MVS ストレージ	

表 3. 64 ビット・ストレージを使用する CICS 機能 (続き)			
CICS 機能	ストレージの使用	ストレージの CICS サブプール	注
ローダー制御ブロック	可変	LD_APES LD_CPES LD_CSECT	使用されるストレージは、CICS 領域でのプログラム・ロード操作数に応じて異なります。
インストール済みアプリケーション・エレメント (IAE) 用のローダー制御ブロック	IAE ごとに 368 バイト	LD_IAES	インストール済みアプリケーション・エレメントはそれぞれ、プラットフォーム上にデプロイされたアプリケーションの特定のバージョンを表します。
主一時記憶域	最小 1 MB 最大 32 GB デフォルト 64 MB TSMINLIMIT システム初期設定パラメーターによって制御されます。	TSDTN TSMIN TSMN0064 TSMN0128 TSMN0192 TSMN0256 TSMN0320 TSMN0384 TSMN0448 TSMN0512 TSMN0576 TSMN0640 TSMN0704 TSMN0768 TSMN0832 TSMN0896 TSMN0960 TSMN1024 TSQUEUE TSTSI	TSMINLIMIT は、使用できる最大ストレージを指定します。これは MEMLIMIT 値の 25% に制限されます。CICS 統計は、実際の使用を表示します。
管理対象プラットフォーム	各ユーザー・タスクに対して 1072 バイト	DFHMPMDR DFHMPPMB DFHMPTAS	DFHMPMDR および DFHMPPMB サブプール内のストレージが、インストールされたポリシーの制御ブロックに使用されます。ポリシーを定義したバンドルが無効にされるまで、使用中になります。 DFHMPTAS サブプール内のストレージには、タスク用のポリシー・カウンターが含まれます。 何らかの ポリシーが CICS 領域にインストールされている場合、すべてのユーザー・タスクに割り振られます。タスクが完了するとこのストレージは解放されます。

表 3. 64 ビット・ストレージを使用する CICS 機能 (続き)			
CICS 機能	ストレージの使用	ストレージの CICS サブプール	注
メッセージ・テーブル	英語のメッセージ・モジュールの場合は、最小 3 MB。 ユーザー・メッセージ・テーブルがロードされる場合は、1 MB を追加します。 ロードされる追加言語ごとに 3 MB を追加します。	CICS DSA の外部の MVS ストレージ	英語のメッセージ・モジュールは常にロードされます。
Node.js アプリケーション	可変	CICS DSA の外部の MVS ストレージ	
ソケット・ドメイン	送信される HTTPS データのコピーを保管します。ストレージのサイズは、送信する HTTPS メッセージのサイズによって異なります。	SOGNRL64	このストレージのサブプールが使用されるのは、HTTPS メッセージを送信する前に、ソケット・ドメインがそのメッセージをコピーする必要がある場合です。
静的データ項目制御ブロック	可変	MPSTAT	イベント・アクションを含むポリシー規則で静的データ・キャプチャー項目を定義する場合に使用します。
ストレージ割り振り制御ブロック	可変	CICS DSA の外部の MVS ストレージ	使用されるストレージは、システム内のストレージ割り振り活動の量に応じて異なります。例えば、エレメント・チェーニングを保持し、多数の小さなレコードを持つサブプールには、より多くのストレージが必要です。
トランザクション・ダンプ・トレース・テーブル	最小 16 KB 最大 1 GB デフォルト 1024 KB TRTRANSZ システム 初期設定パラメーターによって制御されます。	CICS DSA の外部の MVS ストレージ	CICS は、トランザクション・ダンプが生成されるときにのみ、このストレージを取得します。
Web ドメイン	HTTP 本文を保管します。使用されるストレージのサイズは、処理される HTTP 本文のサイズによって異なります。	WB64GNRL	可能な場合、Web ドメインはこのサブプールを使用して、HTTP 本体を 64 ビット・ストレージに格納します。
Web ドメイン	HTTP アウトバウンド・メッセージ用の内部バッファ・ストレージ。ストレージのサイズは、送信する HTTP メッセージのサイズによって異なります。	WBOUTB64	Web ドメインで 64 ビット・ストレージからバッファおよび制御ブロックを取得するために使用します。

表 3. 64 ビット・ストレージを使用する CICS 機能 (続き)			
CICS 機能	ストレージの使用	ストレージの CICS サブプール	注
z/OS XML System Services (XMLSS) パーサー入出力バッファ		ML64GNRL	さまざまな機能 (CICS Web サービスを含む) が、XML 構文解析のためにパーサーを使用します。

DSA サイズ制限

個々の動的ストレージ域 (DSA) のサイズを設定することは、推奨されません。また、通常は、その必要はありません。ただし、一部の DSA のサイズは、**CDSASZE**、**UDSASZE**、**RDSASZE**、**ECDSASZE**、**EUDSASZE**、**EDDSASZE**、および **ERDSASZE** システム初期設定パラメーターを使用して設定することが可能です。

例えば、**CDSASZE** は CICS 動的ストレージ域 (CDSA) のサイズを設定し、**ECDSASZE** は拡張 CICS 動的ストレージ域 (ECDSA) のサイズを指定します。これらのパラメーターのデフォルト値は 0 で、これは DSA のサイズを動的に変更できることを示します。ゼロ以外の値を指定した場合、DSA のサイズは固定になります。

DSA サイズの複数の値を組み合わせて指定しても残りの DSA 用に十分なスペースが許可されない場合、CICS は初期化に失敗します。

- 24 ビット・ストレージ (16 MB 境界より下) の DSA に使用可能なストレージの制限は、**DSALIM** システム初期設定パラメーターによって指定されます。サイズを設定していない 24 ビット・ストレージでは、DSA ごとに少なくとも 256K を許可する必要があります。
- 31 ビット・ストレージ (16 MB 境界より上で、2 GB 境界より下) の DSA に使用可能なストレージの制限は、**EDSALIM** システム初期設定パラメーターによって指定されます。サイズを設定していない 31 ビット・ストレージでは、DSA ごとに少なくとも 1 MB を許可する必要があります。

64 ビット・ストレージ内の DSA、つまり 2 GB 境界より上の DSA (GDSA) については、個々にサイズを設定することはできません。

DSA 限界のコーディング規則

DSA 限界のサイズは、バイト数、キロバイト数、またはメガバイト数として指定できます。

値がキロバイトの整数を表すことを示すには、接尾部として文字 K を使用します。値がメガバイトの整数を表すことを示すには、接尾部として文字 M を使用します。例えば、2 MB は、2048K または 2M としてコーディングできます。(1 KB = 1024 バイト、1 MB = 1024 KB = 1048576 バイト。)

指定する値が 256 KB (**DSALIM** の場合) または 1 MB (**EDSALIM** の場合) の倍数でない場合、CICS は値を次の倍数に切り上げます。

メガバイト数の小数部を指定することはできません。バイト数またはキロバイト数でサイズをコーディングする必要があります。92 ページの表 4 に、いくつかの例を示します。

表 4. バイト数、キロバイト数、およびメガバイト数での DSA 限界値の例

コーディング					
バイト数	2097152	3145788	3670016	4194304	4718592
キロバイト数	2048K	3072K	3584K	4096K	4608K
メガバイト数	2M	3M	-	4M	-

動的ストレージ域のストレージ不足状態

動的ストレージ域 (DSA) の限界が小さすぎる場合、CICS 領域は定期的にストレージ不足状態になります。CICS は、可能な場合、通常の操作を再開するのに十分なストレージをリカバリーできるまで、システム・アクティビティを抑制します。CICS のメッセージおよび統計を使用して、ストレージ不足状態になった時期とそれが解消された時期をモニターします。

CICS は、ストレージ不足状態になる前に、ストレージに対するプレッシャーを解決しようとします。CICS で DSA 内のスペースが不足し始めた場合、その状態はストレージ・ストレス状態と呼ばれます。CICS は、

可能であれば、使用されていないプログラムを削除したり (プログラム圧縮)、あらゆる文書テンプレートのキャッシュ・コピーを削除したり、他の DSA 内のフリー・エクステントを探したりするなどのアクションを実行します。こうしたアクションではストレージ・ストレス状態を解決できない場合、CICS は DSA の SOS 状態を宣言します。

SOS 状態では、CICS は作業を制限するためのステップを実行し、既に進行中の作業を処理するために必要なストレージを確保できるようにします。CICS は、新規の入力メッセージ域の獲得を防止し、CICS システム・モジュールからのすべての ATTACH 要求を延期します。作業を制限することにより、CICS 領域のパフォーマンスが低下します。極端な場合には、SOS 状態の結果、ストレージのデッドロックにより異常終了することもあります。

SOS 状態になると、以下のいずれかのメッセージが発行されます。

- DFHSM0131 (24 ビット・ストレージの場合)
- DFHSM0133 (31 ビット・ストレージの場合)
- DFHSM0606 (64 ビット・ストレージの場合)

SOS 状態は、動的ストレージ域の CICS 統計にも記録されます (「Times went short on storage (ストレージ不足になった回数)」)。CICS コマンドの **CEMT INQUIRE SYSTEM**、**EXEC CICS INQUIRE SYSTEM**、および **CEMT INQUIRE DSAS** を使用して、SOS 状態について照会することができます。

SOS 状態を監視する場合、まず影響を受けたストレージが 24 ビット、31 ビット、64 ビットのいずれであるかを調べます。

- 24 ビット・ストレージの場合、24 ビット・ストレージ内の DSA の制限ができるだけ高く設定されているかどうかを確認します。必要であれば、CICS の実行中に **DSALIM** パラメーターを変更できます。
- 31 ビット・ストレージの場合、拡張動的ストレージ域の制限ができるだけ高く設定されているかどうかを確認します。必要であれば、CICS の実行中に **EDSALIM** パラメーターを変更できます。
- 64 ビット・ストレージの場合、CICS 領域に十分な 64 ビット・ストレージがあるかどうかを確認します。必要であれば、z/OS **MEMLIMIT** 値を変更できますが、変更は CICS 領域の次の始動時にしかできません。

これらの制限の変更については、82 ページの『[CICS ストレージの制限の設定](#)』を参照してください。

CICS は、各 DSA 内に、ストレージ・クッションと呼ばれる連続した仮想記憶領域を予約しています。ストレージ・クッションは、無条件の GETMAIN 要求を満たすのに十分なフリー・ストレージが DSA 内にない場合にのみ使用されます。ストレージ・ストレス状態では、このストレージ・クッションによってストレージのデッドロックを回避できる場合があります。動的ストレージ域に関する CICS ストレージ・マネージャー統計に、CICS がクッションからストレージを使用する必要があった回数が表示されます。要求が DSA 内の残りのすべてのストレージよりも大きくて、クッションのストレージでも不十分な場合があります。この理由で要求が中断状態になった場合、その中断状態も動的ストレージ域の CICS ストレージ・マネージャー統計に表示されます。

24 ビットおよび 31 ビット・ストレージのストレージ不足状態

24 ビットまたは 31 ビット・ストレージ内の個別の DSA (例えば、CDSA) に追加のストレージが必要な場合、CICS ストレージ・マネージャーは追加のエクステントをその DSA に割り振ります。追加のエクステントは、**DSALIM** または **EDSALIM** の制限 (該当する方) に達するまで、必要に応じて獲得できます。可能なすべてのエクステントが割り振られると、CICS は別の DSA 内でフリー・エクステントを検索し、必要としている DSA にそれを再配置します。CICS が 1 つの DSA からエクステントを除去し、それを別の DSA に割り振るためには、そのエクステント内のすべてのページが空きになっていなければなりません。つまり、サブプールに割り振られているページがあってはなりません。

DSALIM または **EDSALIM** の制限に近づき、使用可能なフリー・エクステントまたはフリー・エクステントがほとんどない場合、プログラム圧縮が起動されることがあります。プログラムが入っている DSA が個別に評価され、プログラム圧縮が必要かどうかが判別されます。ロード可能プログラムの比率が適切なシステムでは、プログラム圧縮が仮想記憶圧縮の指標です。

CICS では、以下の条件のすべてに該当する場合、24 ビットまたは 31 ビット・ストレージ内の DSA がストレージ不足状態であると見なします。

- 他の DSA からそれ以上のエクステントを割り振りまたは再配置できない。
- プログラム圧縮を試行済みである。
- 削除に適しており、使用中でない、非常駐プログラムをすべて削除済みである。
- 削除に適している文書テンプレートのすべてのキャッシュ・コピー。[文書テンプレートのキャッシングおよびリフレッシュ](#)を参照してください。
- ストレージ・クッションからのストレージを使用中である (つまり、フリー・ページ数がクッションのページ数より少ない)、または要求を収容できる大きさの連続したストレージ域がないために 1 つ以上の要求が中断状態になっている、あるいはこれらの状態の両方に該当する。

64 ビット・ストレージのストレージ不足状態

64 ビット・ストレージの場合、CICS は、CICS アドレス・スペース用に使われている 64 ビット・ストレージの総量を追跡します。このストレージには、2 GB 境界より上の動的ストレージ域 (GDSA) と、GDSA 外部の CICS 領域内の MVS ストレージの両方が含まれます。

ストレージ・クッションのストレージが使用されている場合、または十分な大きさの連続したストレージ域がないために 1 つ以上の要求が中断状態になっている場合、あるいはこれらの状態の両方に該当する場合、CICS は SOS 状態と見なします。割り振り済みの 2 GB 境界より上のすべてのストレージと新規エクステントのサイズの合計が **MEMLIMIT** 値を超える場合は、64 ビット・ストレージ内の DSA に対して追加のエクステントを割り振ることはできません。

CICS ストレージ・マネージャー統計に、64 ビット・ストレージの使用量が表示されます。CICS ストレージ・マネージャーの動的ストレージ域の統計に、GDSA 内の DSA のストレージ使用量が表示されます。関連のある統計には、以下のものが含まれます。

- Current GDSA active (アクティブな現行の GDSA)
- Peak GDSA active (アクティブなピーク GDSA)
- Number of IARV64 CONVERT(FROMGUARD) failures (IARV64 CONVERT(FROMGUARD) 失敗の数)
- Current GDSA allocated (割り振られている現在の GDSA)
- Peak GDSA allocated (割り振られたピーク GDSA)
- Times cushion released (クッションの解放回数)
- Times went short on storage (ストレージ不足になった回数)

IARV64 CONVERT(FROMGUARD) 失敗は、64 ビット・ストレージの要求が失敗したことを示します。要求をバッキングするための十分な補助記憶域がシステム内にないため、要求が失敗した可能性があります。また、CICS ストレージ・マネージャーが制御しないコンポーネント (例えば、JVM サーバー) が多くのストレージを割り振りすぎて、ストレージ・マネージャーが影響を受けたために、要求が失敗した可能性もあります。CICS では、GDSA 外部のコンポーネントのストレージの割り振りが原因でのストレージに対するプレッシャーは解決できないため、CICS 統計を使用して、こうした問題を特定する必要があります。

ストレージ不足状態の回避

CICS 動的ストレージ域およびストレージ・クッションの使用を最適化し、ストレージ不足状態が発生しないようにするには、以下の原則に従ってください。

手順

- システムにおける同時トランザクションの数が小さいほど、仮想記憶の使用量が減ります。例えば、物理入出力を最小限に抑えることによって、トランザクションの内部応答時間を改善できれば、仮想記憶の使用量を減らすことができます。
- アプリケーション・プログラムで大きい GETMAIN 要求を行わないようにします。
ストレージ・クッションには、連続した大きいブロックのストレージに対する要求を満たせる十分な大きさがない場合があります。
- 必要な場合にのみ、プログラムを常駐として定義します。
CICS は、DSA 内のスペースを再使用するために常駐プログラムを削除することはできません (プログラムが使用中でない場合でも)。

- CICS ストレージ・マネージャー統計を使用して、ストレージ・クッションの解放およびストレージ要求の中断をモニターできます。これらの問題が頻繁に発生する場合は、原因を調査してください。
必要な場合は、ユーザー・タスクの最大数を削減し (**MXT** システム 初期設定パラメーターを使用して)、主記憶域を使用するタスク数を減らします。
- 妥当な数のトランザクションを **SPURGE(YES)** として定義し、**DTIMOUT** 値を指定するようにします。
このように定義されたトランザクションのみ、それらが **DTIMOUT** 値より長くストレージを待っていた場合に、**SOS** 状態時にページできます。ページ可能なトランザクションが少なすぎる場合、CICS システム内のストレージがデッドロック状態になることがあります。

関連タスク

ストレージ不足状態の分析

ストレージ不足 (**SOS**) の問題分析は、システムが **SOS** になっている状態のダンプを取得することから開始します。

サブプール・ストレージのフラグメント化によって生じるストレージ不足状態の修正

DSALIM または **EDSALIM** の制限を増やしても、24 ビット・ストレージまたは 31 ビット・ストレージでストレージ不足状態が発生することがあります。この状態では、CICS 自己調整メカニズムを有効にすることが必要な場合があります。また、対応する **SIT** オーバーライドを使用して、個別の **DSA** のサイズを修正することも可能です。

ストレージ不足状態の分析

ストレージ不足 (**SOS**) の問題分析は、システムが **SOS** になっている状態のダンプを取得することから開始します。

このタスクについて

ストレージ不足状態は、以下の理由で起こる可能性があります。ストレージ不足状態を分析するには、このタスクを実行してください。

1. その他のリソース制約が原因で、通常必要な時間よりも長い時間にわたって CICS タスクがストレージを占有している
2. 大量のタスクのために、使用可能なフリー・ストレージでは対応できない
3. アプリケーションの設計が悪く、必要なストレージ容量が極端に多い

手順

1. メッセージ **DFHSM0131**、**DFHSM0133**、または **DFHSM0606** が発行されたときにダンプを生成する項目をダンプ・テーブルに設定します。
例えば、初めてメッセージ **DFHSM0131** が発行されたときにダンプを生成するには、次のコマンドを使用します。

```
CEMT SET SYDUMPCODE(SM0131) SYSDUMP MAXIMUM(1) ADD
```

2. ダンプを取得するときは、以下の **IPCS** コマンドを入力します。
 - a) **IPCS** コマンド **VERBX CICS730 'SM=3'** を使用して、**SM** 制御ブロックをフォーマットします。
 - b) **IPCS** コマンド **VERBX CICS730 'LD=3'** を使用して、**LD** 制御ブロックをフォーマットします。
3. 統計間隔が完了する直前に、**DFHOSTAT** を実行します。
例えば、統計間隔が 1 時間である場合、59 分の時点で **DFHOSTAT** を実行します。**DFHOSTAT** は、ストレージの要約として有用な情報を示します。サブプール別の詳細はありません。詳細については、[サンプルの統計プログラム DFHOSTAT](#) を参照してください。
4. 収集した情報で、**DSA** の集計を調査し、ストレージ不足になっている **DSA**、およびその他の **DSA** 内のフリー・スペースの量をメモします。
フリー・スペースの量は、各 **DSA** のそれぞれのエクステントごとに指定されています。**UDSA** または **CDSA** のいずれかが頻繁にストレージ不足になりますが、**SDSA** には大容量のフリー・ストレージがあります。

また、大量の冗長プログラム・ストレージ (RPS) の兆候を探します。これはストレージ不足状態の原因になることがあります。冗長プログラム・ストレージは、ドメイン・サブプール集計およびローダー・ドメイン集計で確認できます。

例

この例に示すダンプは、UDSA がストレージ不足状態のときに抽出されたものです。

24 ビット・ストレージ (16 MB 境界より下) のストレージ・エクステントは、UDSA の場合を除いて、常に 256 KB の倍数で割り振られます。トランザクション分離がアクティブな場合、UDSA のエクステント・サイズは 1 MB であるため、各 UDSA エクステントは 1MB 単位になるように調整する必要があります。トランザクション分離がアクティブでない場合、割り振りは 256 KB エクステント単位で行われます。CDSA、RDSA、および SDSA では 256 KB エクステント単位 (UDSA では 1 MB エクステント単位) での一定量のフラグメント化を考慮に入れる必要があります。

31 ビット・ストレージ (16 MB 境界より上) のストレージ・エクステントは、1 MB の倍数で割り振られます。

64 ビット・ストレージ (2 GB 境界より上) のストレージ・エクステントは、2 GB の倍数で割り振られます。

それぞれのエクステントには、関連するページ・プール・エクステント (PPX) およびページ割り振りマップ (PAM) があります。

SDSA エクステントの検査では、大容量のフリー・スペースを持ついくつかのエクステントが表示されています。例えば、00700000 から始まって 0073FFFF まで実行するエクステントには、4 KB のみ割り振られ、252 KB がフリーになっています。

Extent list:	Start	End	Size	Free
	00700000	0073FFFF	256K	252K

DSA エクステント集計では、00700000 のエクステントの PPX が 09F0A100 にあり、関連する PAM が 09F0A150 にあることが示されています。PAM の検査では、1 ページのみが割り振られ、そのページが X'7A' の ID を持つサブプールに属していることが示されています。

Start	End	Size	PPX_addr	Acc	DSA
00700000	0073FFFF	256K	09F0A100	C	SDSA
PPX.SDSA 09F0A100 Pagepool Extent Control Area					
0000	00506EC4	C6C8E2D4	D7D7E740	40404040	*.&>DFHSMPPX *
0010	E2C4E2C1	40404040	09A1BA68	071B3EA0	*SDSA
0020	00040000	00700000	0073FFFF	071B5EE0	*.....*
0030	00000000	09F0A150	00000040	0710A268	*.....0.&;... ..s.*
0040	0003F000	00000000	00000000	00000000	*..0.....*
PAM.SDSA 09F0A150 Page Allocation Map					
0000	00000000	00000000	00000000	00000000	*.....*
0010	-	002F	LINES SAME AS PREVIOUS		
0030	00000000	0000007A	00000000	00000000	*.....*

ドメイン・サブプール集計では、SDSA について、X'7A' の ID に関連付けられているサブプールを判別できます。このダンプでは、7A は、サブプール ZCTCTUA の ID です。CICS を複数回実行する場合には、同一の ID に依存しないようにしてください。これは、ID が ADD_SUBPOOL が発行される順番に割り当てられるためです。

==SM: UDSA Summary (first part only)	
Size:	512K
Cushion size:	64K
Current free space:	56K (10%)
* Lwm free space:	12K (2%)
* Hwm free space:	276K (53%)
Largest free area:	56K
* Times nostg returned:	0
* Times request suspended:	0
Current suspended:	0
* Hwm suspended:	0
* Times cushion released:	1

Currently SOS:	YES
==SM: SDSA Summary (first part only)	
Size:	4352K
Cushion size:	64K
Current free space:	2396K (55%)
* Lwm free space:	760K (17%)
* Hwm free space:	2396K (55%)
Largest free area:	252K
* Times nostg returned:	0
* Times request suspended:	0
Current suspended:	0
* Hwm suspended:	0
* Times cushion released:	0
Currently SOS:	NO

次のタスク

1. ご使用の CICS システムのストレージ制限を検討します。[82 ページの『CICS ストレージの制限の設定』](#)を参照してください。
2. 24 ビット・ストレージの SOS 状態の場合、**DSALIM** パラメーターが可能な限り大きい値に設定されているかどうかを調べます。[84 ページの『DSALIM の見積もり、確認、および設定』](#)を参照してください。
3. 31 ビット・ストレージの SOS 状態の場合、**EDSALIM** パラメーターが可能な限り大きい値に設定されているかどうかを調べます。[85 ページの『EDSALIM の見積もり、確認、および設定』](#)を参照してください。
4. 64 ビット・ストレージの SOS 状態の場合、z/OS **MEMLIMIT** パラメーターが適切な値に設定されているかどうかを調べます。[87 ページの『MEMLIMIT の見積もり、確認、および設定』](#)を参照してください。
5. 最大タスク指定 (**MXT** パラメーター) などのオプションの使用を検討し、プログラムを常駐として定義して、全体的なストレージ要件を低く抑えます。これらの設定を変更すると、タスクのスループットが制限される場合があります。16 MB より上で実行されるプログラムを使用することにより、16 MB より下のストレージ制約を削減することもできます。また、LPA の使用により、LDNUCRO で使用されるストレージの量が約 100 KB 削減されます。
6. z/OS の調整の可能性、および CICS 外部のその他の調整の可能性を検討します。ご使用の CICS 領域を分割する方法も検討します。
7. CICS 自己調整メカニズムを使用可能にしたり、適切な SIT オーバーライドを使用して 1 つ以上の個別 DSA のサイズを修正することを検討します。説明は、[97 ページの『サブプール・ストレージのフラグメント化によって生じるストレージ不足状態の修正』](#)を参照してください。

関連タスク

ストレージ不足状態の回避

CICS 動的ストレージ域およびストレージ・クッションの使用を最適化し、ストレージ不足状態が発生しないようにするには、以下の原則に従ってください。

[サブプール・ストレージのフラグメント化によって生じるストレージ不足状態の修正](#)

DSALIM または **EDSALIM** の制限を増やしても、24 ビット・ストレージまたは 31 ビット・ストレージでストレージ不足状態が発生することがあります。この状態では、CICS 自己調整メカニズムを有効にすることが必要な場合があります。また、対応する SIT オーバーライドを使用して、個別の DSA のサイズを修正することも可能です。

[サブプール・ストレージのフラグメント化によって生じるストレージ不足状態の修正](#)

DSALIM または **EDSALIM** の制限を増やしても、24 ビット・ストレージまたは 31 ビット・ストレージでストレージ不足状態が発生することがあります。この状態では、CICS 自己調整メカニズムを有効にすることが必要な場合があります。また、対応する SIT オーバーライドを使用して、個別の DSA のサイズを修正することも可能です。

このタスクについて

自己調整メカニズムおよび SIT オーバーライドは、**DSALIM** または **EDSALIM** の制限を増やしてもストレージ不足問題が完全に解消されない場合にのみ使用してください。

管理対象エクステントへの割り振りを行った結果、GETMAIN 要求を満たすには不十分なストレージのブロックがエクステント内に生じることがあります。サブプールや DSA は動的な特性を持っているため、エクステント・ストレージが再使用されるにつれて、この状態は解消されるものと思われます。影響を受ける DSA に対して SIT オーバーライドを使用して初期 DSA サイズを指定すると、CICS は指定された量までの連続したエクステントを予約して、ストレージのブロックを除去します。

ヒント: MAPS を MAPS として定義します。MAPS をプログラムとして定義した場合は、LDNUC ではなく LDRES にロードされます。LDRES は SDSA の一部で、フラグメント化にはより影響を受けやすくなります。

手順

1. ローカル・カタログにレコードを追加して、ストレージ・マネージャー・ドメイン・サブプール用に CICS 自己調整メカニズムを使用可能にできます。
CICS 提供のユーティリティー・プログラム DFHSMUTL を使用してサブプール・レコードを処理する方法についての詳細は、[ローカル・カタログ・ストレージ・プログラム \(DFHSMUTL\)](#) を参照してください。
2. 対応する SIT オーバーライド (**CDSASZE**、**UDSASZE**、**SDSASZE**、**RDSASZE**、**ECDSASZE**、**EUDSASZE**、**ESDSASZE**、および **ERDSASZE**) を使用して、1 つ以上の個別の DSA のサイズを修正することができます。

これらのオーバーライドについて詳しくは、[システム初期設定パラメーターの説明と要約](#)を参照してください。

使用する値を判別するには、以下のプロセスを実行してください。

- a) DFHOSTAT 出力を収集して、統計間隔中の各 DSA によるストレージの使用を示す情報を入手します。
- b) 数日間 CICS 統計を検討します。

統計から、サブプール・レベルおよび DSA レベルで使用されるストレージ量を定義するために使用できる情報が得られます。エクステントの使用量は、追加され、解放されたエクステント数で表示されます。

DFHOSTAT で提供される DSA 情報に加えて、割り振られていた DSA を含むそれぞれのサブプールの結果が提供されます。統計を集計中の場合には、1 日の終わり統計は、最後に統計が収集されて以降のデータのみを提供します。

関連タスク

[ストレージ不足状態の回避](#)

CICS 動的ストレージ域およびストレージ・クッションの使用を最適化し、ストレージ不足状態が発生しないようにするには、以下の原則に従ってください。

[ストレージ不足状態の分析](#)

ストレージ不足 (SOS) の問題分析は、システムが SOS になっている状態のダンプを取得することから開始します。

CICS サブプール

各動的ストレージ域において、ストレージはサブプールに配置されます。CICS サブプールは、適用可能な動的ストレージ域から一度に 1 ページずつ、必要に応じて動的に獲得されます。

ほとんどの CICS サブプールは、31 ビット (16 MB 境界より上) ストレージまたは 64 ビット (2 GB 境界より上) ストレージです。24 ビット (16 MB 境界より下) ストレージ内のサブプールは、使用可能なスペースが限られているため、より注意してモニターする必要があります。

個々のサブプールは、静的または動的です。これらのサブプールの中には、調整できない静的 CICS ストレージが含まれているものがあります。すべてのサブプールのストレージ・サイズは、4 KB の倍数に切り上げられます。この切り上げ係数は、サブプールのサイジング、または調整やその他の変更後のストレージ・サイズ変更の評価に含めてください。

CICS ドメイン・サブプールの統計には、動的ストレージ域のサブプールのサイズおよび使用に関する有用な情報が含まれています。以下のトピックでは、各動的ストレージ域内のサブプールをリストし、その使用について説明します。この情報を使用して、個々のサブプールでの過剰な使用についての考えられる原因を特定することができます。

CDSA の CICS サブプール

CICS 動的ストレージ域 (CDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 5. CDSA の CICS サブプール	
サブプール名	説明
AP_TCA24	タスク・データ・ロケーション・オプションが BELOW に設定されている場合、TCA のストレージ。
DFHAPD24	24 ビット・アプリケーション・ドメイン・ストレージの汎用サブプール。
DFHTDSDS	実際の一時データ SDSCI 用のストレージ。各 SDSCI には 24 ビット・ストレージ (16 MB 境界より下) に常駐する DCB が含まれています。
DHPDPOOL	文書ハンドラー・ドメインによって使用される区分データ・セットの DCB 用のストレージ。
FC_DCB	BDAM ファイルの DCB のストレージ。定義されているそれぞれのファイルごとに、104 バイト必要です。
FCCBELOW	実 VSWA および先行読み取りデータ・バッファのストレージ。それぞれの VSWA は、120 バイトのストレージが必要です。先行読み取りデータ・バッファの最大数は、以下のように計算されます。 (ストリングの数) x (最大レコード長) x (ファイル数)。
KESTK24	単一の 2 KB 24 ビット (16 MB 境界より下) スタック・セグメント。これは、すべてのタスクで共用されるダミー・スタック・セグメントです。24 ビット・スタック・ストレージを使用する必要があるタスクは、サブプール KESTK24E から拡張スタック・セグメントを取得します。
KESTK24E	24 ビット・スタック・ストレージを使用する必要があるタスクが取得する、4 KB 24 ビット (16 MB 境界より下) 拡張スタック・セグメント。CICS は、サブプール内に使用可能なストレージが他にない場合にタスクが使用できる、24 ビット拡張スタック・セグメントの予約プールを事前割り振りします。
LD_JFCB	ローダー・ドメインのジョブ・ファイル制御ブロックのストレージ。
LDNRS	RESIDENT である CICS 中核、およびマクロ・テーブル用のストレージ。CICS 中核は約 192 KB で、テーブルのサイズは計算可能です。プログラムは、EXECKEY (CICS) として定義され、再入可能オープンなしで RMODE(24) でリンク・エディットされます。
LDNUC	RESIDENT ではない CICS 中核、およびマクロ・テーブル用のストレージ。CICS 中核は約 192 KB で、テーブルのサイズは計算可能です。プログラムは、EXECKEY (CICS) として定義され、再入可能オープンなしで RMODE(24) でリンク・エディットされます。
SMCONTRL	制御クラス・ストレージに対する GETMAIN 要求を満たします
SMSHARED	24 ビット共用ストレージ。例えば、RMI グローバル作業域、モニター中のトランザクションの存続期間中の EDF ブロック、およびその他の制御ブロックです。
SMSHRC24	SHARED_CICS24 クラス・ストレージの多数のブロックが使用。
SMT24	境界および端末入出力域用のストレージ。16 MB 境界から上には配置できません。ストレージ要件は、システム内の端末の数と回線トラフィックによって決まります。サブプールは、RAPOOL、RAMAX、TIOAL サイズ、および MRO セッション数を削減することによって調整できます。
SZSPFCAC	FEPI z/OS Communications Server ACB 作業域。
TRUBELow	16 MB 境界から下のタスク関連ユーザー出口プール。
XMG24	トランザクション・マネージャーが使用する汎用ストレージ。
ZCSETB24	16 MB 境界から下のアプリケーション制御バッファ。

表 5. CDSA の CICS サブプール (続き)

サブプール名	説明
ZCTCTUA	TCTTE ユーザー域用のストレージ。これは、次の DSA、すなわち SDSA、ECDSA、CDSA、または ESDSA のいずれか 1 つに配置できます。配置場所は、システム 初期設定パラメーター TCTUALOC=ANY BELOW およびシステム 初期設定パラメーター TCTUAKEY=CICS USER によって制御されます。最大サイズは、端末定義の USERAREALEN オペランドで指定できます。端末定義について詳しくは、 TERMINAL ターミナル を参照してください。

SDSA の CICS サブプール

共用動的ストレージ域 (SDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 6. SDSA の CICS サブプール

サブプール名	説明
APECA	イベント制御域のストレージ。
DFHAPU24	16 MB 境界から下のアプリケーション・ドメイン・ストレージの汎用サブプール。
LDPGM	動的にロードされたアプリケーション・プログラム (RMODE (24)) のストレージ。このサブプールの予想サイズは、前のリリースを参考にし、LDPGMRO を考慮に入れることによって予測できます。サブプール・サイズは、31 ビット・プログラムを使用することによって削減できることがあります。再入可能ではありません。
LDRES	常駐アプリケーション・プログラム (RMODE (24)) のストレージ。このサブプールの予想サイズは、前のリリースを参考にし、LDRESRO を考慮に入れることによって予測できます。サブプール・サイズは、31 ビット・プログラムを使用することによって削減できることがあります。再入可能ではありません。
OSCOBOL	COBOL マージ・ロード・リスト (MLL) 制御ブロックおよびそのエクステントの割り振りで使用。このサブプールは、ストレージ 1 ページの初期割り振りよりも大きく占有することはできません。
SMSHRU24	SHARED_USER24 クラス・ストレージの多数の制御ブロックで使用。
ZCTCTUA	TCTTE ユーザー域用のストレージ。これは、次の DSA、すなわち SDSA、ECDSA、CDSA、または ESDSA のいずれか 1 つに配置できます。配置場所は、システム 初期設定パラメーター TCTUALOC=ANY BELOW およびシステム 初期設定パラメーター TCTUAKEY=CICS USER によって制御されます。最大サイズは、端末定義の USERAREALEN オペランドで指定できます。端末定義について詳しくは、 TERMINAL ターミナル を参照してください。

RDSA の CICS サブプール

読み取り専用動的ストレージ域 (RDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 7. RDSA の CICS サブプール

サブプール名	説明
LDNRSRO	RESIDENT で、リンク・エディットされた REENTRANT および RMODE(24) であった、プログラム定義された EXECKEY(CICS) のストレージ。
LDNUCRO	RESIDENT ではなく、EXECKEY(CICS) として定義され、RMODE(24) および REENTRANT でリンク・エディットされたプログラムのストレージ。
LDPGMRO	RESIDENT ではなく、EXECKEY(USER) として定義され、RMODE(24) および REENTRANT でリンク・エディットされたプログラムのストレージ。
LDRESRO	RESIDENT で、リンク・エディットされた REENTRANT および RMODE(24) であった、プログラム定義された EXECKEY(USER) のストレージ。

ECDSA の CICS サブプール

拡張 CICS 動的ストレージ域 (ECDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 8. ECDSA の CICS サブプール	
サブプール名	説明
>LGJMC	ログ・マネージャー・ドメインのジャーナル・モデル・リソース・エントリー用のストレージ。
AITM_TAB	自動インストール端末形式 (AITM) テーブル項目サブプール (DFHAITDS)。
AP_TCA31	タスク・データ・ロケーション・オプションが ANY に設定されている場合、TCA のストレージ。
AP_TXDEX	TXD テーブルのアプリケーション・パーツのストレージ。
APAID31	境界から上の AID のストレージ。
APBMS	BMS が使用するストレージ。
APCOMM31	COMMAREA のストレージ。ストレージ要件は、指定した COMMAREA のサイズと、アプリケーションの同時ユーザー数によって決まります。
APDWE	非タスク据え置き作業エレメントのストレージ。
APICE31	境界から上の ICE のストレージ。
APURD	リカバリー単位記述子 (URD) と非タスク据え置き作業エレメント (DWE) を含むサブプール。
ASYNCBUF	ソケット・ドメインの非同期操作が使用するバッファ。
BAGENRAL	ビジネス・アプリケーション・マネージャー・ドメインの汎用サブプール
BAOFBUSG	ビジネス・アプリケーション・マネージャー・ドメインが使用するバッファ・ストレージ。
BAOFT_ST	ビジネス・アプリケーション・マネージャー・ドメインのアクティビティが使用するストレージ。
BR_BFBE	ブリッジ機能のブロック拡張のストレージ。
BR_BFNB	ブリッジ機能の名前ブロックのストレージ。
BR_BMB	ブリッジ・メッセージ・ブロックのストレージ。
BR_BSB	ブリッジ開始ブロックのストレージ。
BRGENRAL	ブリッジが使用する汎用サブプール
BRNSBLK	ブリッジ番号スペースに使用されるストレージ。
BRNSFBLK	ブリッジ・ファイルが使用するストレージ。
BRPC	ブリッジ 1 次クライアントが使用するストレージ。
BRVS	ブリッジ仮想端末が使用するストレージ。
BRVSCA	ブリッジ仮想画面文字属性が使用するストレージ。
BRVSXA	ブリッジ仮想画面拡張属性が使用するストレージ。
CCNV_BCE	文字変換バッファ・チェーン・エレメントのストレージ。
CCNV_CCE	文字変換チェーン・エレメントのストレージ。
CCNV_TRT	文字変換の変換テーブルのストレージ。これらのテーブルは、変換チェーン・エレメントによってアドレス指定されます。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
CCNVG_AN	文字変換のアンカー・ブロックのストレージ。
COLARAY	Web 制御ブロックのアレイ・ストレージのストレージ。
CQCQ_AN	コンソール・キュー管理のアンカー・ブロックのストレージ。
CQCQ_CB	コンソール・キュー管理のコマンド入力バッファのストレージ。
DBCTL	サブプールには、DBCTL タスク関連ユーザー出口プログラム DFHDBAT の起動時に、RMI が使用する TIE ブロックが含まれています。TIE は 120 バイト長で、このタスク関連ユーザー出口用ローカル・タスク作業域が TIE に追加されます。DFHDBAT の場合には 668 バイト長です。このサブプールは、DBCTL の使用時のみ存在します。サブプールは、DBCTL スレッドを制限することによって、または最大タスク (MXT) やトランザクション・クラスを使用することによって調整できます。
DBDBG	DBCTL グローバル・ブロックのストレージ。
DCTE_EXT	すべての区画外キュー定義のストレージ。
DCTE_IND	すべての間接キュー定義のストレージ。
DCTE_INT	すべての区画内キュー定義のストレージ。
DCTE_REM	すべてのリモート・キュー定義のストレージ。
DDAPSESS	LDAP セッションの状態制御ブロックのストレージ。
DDAPSRCH	LDAP 検索結果のバッファ。
DDBROWSE	ディレクトリー・マネージャー・ブラウズ要求トークンのストレージ。
DDGENRAL	ディレクトリー・マネージャー制御ブロック一般情報のストレージ。
DDS_BFBE	BFBE テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_BFNBB	BFNB テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_DCTE	DCTE テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_DHT1	DHT1 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_DHT2	DHT2 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_DSN	DSN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_D2AC	D2AC テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_D2CS	D2CS テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_D2EN	D2EN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_D2TN	D2TN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
DDS_D2TT	D2TT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_ECCS	ECCS テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_ECEV	ECEV テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_ECSC	ECSC テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_EPAD	EPAD テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_FCT	FCT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_ISIA	ISIA テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_ISIN	ISIN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_JVMD	JVMD テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_MLRL	MLRL テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_MLXT	MLXT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_MQII	MQII テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_MQIN	MQIN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_NQRN	NQRN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_P IPL	PIPL テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_PPT	PPT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_PTPO	PTPO テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_PTST	PTST テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_PTT	PTT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_REFE	REFE テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_RLBN	RLBN テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。

表 8. ECDSA の CICS サブプール (続き)	
サブプール名	説明
DDS_RTXD	RTXD テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_SCAC	SCAC テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_SERV	SERV テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_SOCI	SOCI テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_SOSI	SOSI テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_TCL	TCL テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_TPNM	TPNM テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_TXD	TXD テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_USD1	USD1 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_USD2	USD2 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_USD3	USD3 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_USD4	USD4 テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_WBST	WBST テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_WBUR	WBUR テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_WSRD	WSRD テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_WURS	WURS テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_W2AT	W2AT テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DDS_W2RL	W2RL テーブルのディレクトリー・マネージャー・ディレクトリー・エレメントのストレージ。
DFHAPDAN	16 MB 境界から上だが 2 GB 境界から下のアプリケーション・ドメイン・ストレージの汎用サブプール。
DFHD2CSB	CICS/Db2 アダプターによって作成された Db2 スレッドを表す制御ブロックのストレージ。
DFHD2ENT	DB2ENTRY 定義を表す制御ブロックのストレージ。
DFHD2PKG	Db2 PACKAGESET 定義を表す制御ブロックのストレージ。

表 8. ECDSA の CICS サブプール (続き)	
サブプール名	説明
DFHD2TRN	DB2TRAN 定義を表す制御ブロックのストレージ。
DFHECCD	イベント・キャプチャー・データのストレージ。
DFHECCS	イベント・キャプチャー仕様ブロックのストレージ。
DFHECDQE	イベント・キャプチャー据え置きフィルター・キュー・エレメントのストレージ。
DFHECEVB	イベント・キャプチャー・イベント・バインディング・ブロックのストレージ。
DFHECFP	イベント・キャプチャー・イベント・フィルター述部ブロックのストレージ。
DFHECSC	イベント・キャプチャー・システム・イベント呼び出しのストレージ。
DFHECSF	イベント・キャプチャー・システム・フィルター述部のストレージ。
DFHEPAC	イベント・キャプチャー・イベント・アダプター構成データのストレージ。
DFHMPGEN	管理対象プラットフォームのアンカー・ブロック (MPA) と障害アダプター (MPPFA) 制御ブロックの割り振りに使用されます。
DFHMPMOD	管理対象プラットフォーム・モデル (MPMOD) 制御ブロックの割り振りに使用します。
DFHMPPPB	管理対象プラットフォーム・ポリシー (MPPPB) 制御ブロックの割り振りに使用します。
DFHTDG31	一時データ汎用ストレージおよび制御ブロック。ストレージ要件は、バッファとストリングの数、および指定されている制御間隔サイズによって決まります。
DFHTDIOB	区画内一時データ入出力バッファ。ストレージ要件は、区画内一時データ・セットの制御間隔サイズをバッファ数で乗算した数によって決まります。
DFHTDWCB	一時データ待機エレメントのストレージ。
DHCACHE	文書テンプレートのキャッシュ・コピーのストレージ。
DHDBB	文書ブックマーク・ブロックのストレージ。
DHDCR	文書制御レコードのストレージ。
DHDDB	文書データのストレージ。
DHDOA	文書アンカー・ブロックのストレージ。
DHFSPATH	HFS パス・テンプレート拡張のストレージ。
DHGENRAL	文書マネージャー・ドメインの汎用サブプール。
DHSTB	文書シンボル・テーブルのストレージ。
DHTLPOOL	文書ハンドラー・テンプレート記述子のストレージ。
DLI	サブプールには、EXEC DL/I タスク関連ユーザー出口プログラム DFHEDP の起動時に、RMI が使用する TIE ブロックが含まれています。TIE は 120 バイト長で、このタスク関連ユーザー出口用ローカル・タスク作業域が TIE に追加されます。DFHEDP の場合には 4 バイト長です。このサブプールは、EXEC DL/I の使用時のみ存在します。サブプールは、DBCTL スレッドを制限することによって、または最大タスク (MXT) やトランザクション・クラスを使用することによって調整できます。
DMSUBPOL	一般的に使用するドメイン・マネージャー・サブプール。
DP_GENRL	DP ドメインの制御ブロックのストレージ。
DPLA	デバッグ・プロファイルのインスタ・リンク・リストのためのアンカー・ブロックのストレージ。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
DPLE	デバッグ・プロファイルのインスタ・リンク・リスト内のエレメントのストレージ。
DPLP	パターン・マッチングに使用されるデバッグ・プロファイル内のエレメントのストレージ。
DPTA	DP ドメインに必要なトランザクション・インスタンス状態データのストレージ。
DS_STIMR	ディスパッチャー・ドメイン STIMER トークンのストレージ。
DS_TCB	ディスパッチャー・ドメイン TCB のストレージ。
DS_VAR	ディスパッチャー・ドメイン可変長サブプール。
DSBROWSE	ディスパッチャー・ブラウズ要求トークンのストレージ。
EC_GENRL	EC ドメインの制御ブロックのストレージ。
EJMI	エンタープライズ Bean メソッド情報。
EJOSGENS	エンタープライズ Bean 汎用サブプール。
EJOSTSKS	エンタープライズ Bean タスク・サブプール。
EJSPBFBC	エンタープライズ Bean のブラウザー制御ブロックのストレージ。
EJSPBVIC	エンタープライズ Bean 制御ブロックのストレージ。
EJSPCFBC	CorbaServers のブラウザー制御ブロックのストレージ。
EJSPCFIC	CorbaServers の制御ブロックのストレージ。
EJSPCOMM	エンタープライズ Bean のアンカー・ブロックのストレージ。
EJSPDFBC	配置 JAR ファイルの Web ブラウザー制御ブロックのストレージ。
EJSPDFIC	配置 JAR ファイルの制御ブロックのストレージ。
EJSPGVNC	エンタープライズ Bean の永続的ストレージのストレージ。
EJSPTVNC	エンタープライズ Bean のトランザクション関連ストレージのストレージ。
EJSTGENS	エンタープライズ Bean 統計の制御ブロックのストレージ。
EMBRB	イベント・マネージャー・ブラウズ・ブロックのストレージ。
EMEVA	イベント・マネージャー・イベント・プール・アンカーのストレージ。
EMEBB	イベント・マネージャー・イベント・ブロックのストレージ。
EMGENRAL	イベント・マネージャー・ドメインの汎用サブプール
EP_GENRL	EP ドメインの制御ブロックのストレージ。
EPADA	イベント処理アダプター管理のストレージ。
EPADI	EP アダプター・セット内の EP アダプター名のストレージ。
EPADT	イベント処理アダプター・セット管理のストレージ。
FC_ABOVE	実 VSWA および先行読み取りデータ・バッファのストレージ。それぞれの VSWA は、120 バイトのストレージが必要です。先行読み取りデータ・バッファの最大数は、以下のように計算されます。 (ストリングの数) x (最大レコード長) x (ファイル数)
FC_ACB	VSAM ファイルの ACB のストレージ。VSAM ファイルごとに 1 つの ACB (80 バイト) があります。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
FC_BDAM	BDAM ファイル制御ブロックのストレージ。それぞれの BDAM ファイルには、96 バイトのストレージが必要です。
FC_DSNAM	データ・セット名ブロックのストレージ。それぞれのファイルには、120 バイトのストレージを使用するデータ・セット名ブロックが必要です。
FC_FCPE	ファイル制御プール・エレメントのストレージ。
FC_FCPW	ファイル制御 CFDT プール待機エレメントのストレージ。
FC_FCUP	作業プール・ブロックのファイル制御 CFDT ユニットのストレージ。
FC_FLAB	ファイル制御永続アクセス・ブロックのストレージ。
FC_FLLB	ファイル制御ロック・ロケータ・ブロックのストレージ。
FC_FRAB	ファイル要求アンカー・ブロック (FRAB) のストレージ。ファイル制御要求を発行したトランザクションごとに 1 つの FRAB があります。FRAB は、タスクの終了まで保存されています。現在未使用の FRAB のフリー・チェーンがあります。
FC_FRTE	<p>ファイル要求スレッド・エレメント (FRTE) のストレージ。タスクのそれぞれのアクティブ・ファイル制御要求ごとに、1 つの FRTE があります。以下の条件に適合する場合には、ファイル制御要求に 1 つの FRTE があります。</p> <ul style="list-style-type: none"> • VSAM スレッドをまだ終了させていない場合。例えば、ENDBR を未発行の参照。 • リカバリー可能ファイルを更新した状態で、同期点がまだ発生していない場合。 • 今後解放する必要がある READ-SET ストレージを保持している場合。 <p>現在未使用の FRTE のフリー・チェーンがあります。</p>
FC_RPL	ファイル制御の要求パラメーター・リストのストレージ。
FC_SHRCT	ファイル制御 SHRCTL ブロックのストレージ。これらのブロックは 8 個あり、それぞれは VSAM LSR プールを記述します。
FC_VSAM	VSAM ファイルのファイル管理テーブル (FCT) 項目のストレージ。
FCB_256	256 バイトの長さのファイル制御バッファ。これらのバッファは、最大レコード長が 256 バイト以下のファイルに対して発行されたファイル制御要求が使用します。
FCB_512	512 バイトの長さのファイル制御バッファ。これらのバッファは、最大レコード長が 256 プラス 1 バイトから 512 バイトまでの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_1K	1 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 512 プラス 1 バイトから 1 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_2K	2 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 1 KB プラス 1 バイトから 2 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_4K	4 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 2 KB プラス 1 バイトから 4 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_8K	8 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 4 KB プラス 1 バイトから 8 KB までの間のファイルに対して発行されたファイル制御要求が使用します。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
FCB_16K	16 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 8 KB プラス 1 バイトから 16 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_32K	32 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 16 KB プラス 1 バイトから 32 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_64K	64 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 32 KB プラス 1 バイトから 64 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_128K	128 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 64 KB プラス 1 バイトから 128 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_256K	256 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 128 KB プラス 1 バイトから 256 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_512K	512 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 256 KB プラス 1 バイトから 512 KB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_1M	1 MB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 512 KB プラス 1 バイトから 1 MB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_2M	2 MB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 1 MB プラス 1 バイトから 2 MB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_4M	4 MB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 2 MB プラス 1 バイトから 4 MB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_8M	8 MB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 4 MB プラス 1 バイトから 8 MB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCB_16M	16 KB 長のファイル制御バッファ。これらのバッファは、最大レコード長が 8 MB プラス 1 バイトから 16 MB までの間のファイルに対して発行されたファイル制御要求が使用します。
FCSTATIC	ファイル制御静的ストレージ。
ICUS	内部制御エレメント (ICE) セキュア拡張のストレージ。
IE_GENRL	IE ドメインの制御ブロック。
IECCB	IE ドメインの会話制御ブロック。
IECSB	IE ドメインのクライアント状態ブロック。
IFGLUWID	VSAM IFGLUWID 領域。
IIGENRAL	IIOP ドメイン汎用サブプール。
IIMBR	IIOP ドメイン要求モデル・ブラウズ・ブロック。
IIMDB	IIOP ドメイン要求モデル・ブロック。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
IS_GENRL	IS ドメインの制御ブロックのストレージ。
ISAQ	IS 割り振りキュー・エレメントのストレージ。
ISCB	インストールされた IPCONN のインスタンスを記録するために使用される、IS 制御ブロックのストレージ。
ISQA	IS キュー接続制御ブロックのストレージ。
ISRD	IS リモート削除要求のストレージ。
ISSB	それぞれが ISCB サブプールに関連付けられた、IS セッション・ブロックのストレージ。
ISSS	IS セッション・セットのストレージ。
KEANCHOR	ストレージ・マネージャーのドメイン・アンカー。
KESTK31	28 KB 31 ビット (16 MB 境界より上) スタック・セグメント。MXT ごとに 1 つと、実行中のそれぞれの動的システム・タスクごとに 1 つあります。
KESTK31E	8 KB 31 ビット (16 MB 境界より上) 拡張スタック・セグメント。少なくとも、MXT 制限で指定されている 10 のタスクごとに 1 つあります。
KETASK	カーネル・タスク項目のストレージ。
LD_CDE	ローダー・ドメインのダミー CDE のストレージ。
LD_CNTRL	ローダー・ドメインの汎用制御情報のストレージ。
LD_LDBE	ローダー・ドメインの LDBE (ローダー・ブラウズ・エレメント) 制御ブロックのストレージ。
LD_LDWE	ローダー・ドメインの LDWE (ローダー中断作業エレメント) 制御ブロックのストレージ。
LD_PLIBE	ローダー・ドメインのプログラム・ライブラリー・エレメント・ストレージのストレージ。
LDENRS	RESIDENT である拡張 CICS 中核、および 31 ビット・マクロ・テーブル用のストレージ。拡張 CICS 中核は、約 50 KB です。REENTRANT オプションが指定されず、EXECCKEY(CICS) で定義され、RMODE(ANY) でリンク・エディットされたプログラム。
LDENUC	RESIDENT ではない拡張 CICS 中核、および 31 ビット・マクロ・テーブル用のストレージ。拡張 CICS 中核は、約 50 KB です。REENTRANT オプションが指定されず、EXECCKEY(CICS) で定義され、RMODE(ANY) でリンク・エディットされたプログラム。
LGBD	ログ・マネージャー・ドメインのログ・ストリーム名、ジャーナル名、およびジャーナル・モデル・ブラウズ・トークン用のストレージ。
LGGD	ログ・マネージャー・ドメインの明示的に開かれた汎用ログ用のストレージ。
LGGENRAL	ログ・マネージャー・ドメインの汎用サブプール。
LGJI	ログ・マネージャー・ドメインのジャーナル名エントリー用のストレージ。
LGSD	ログ・マネージャー・ドメインのログ・ストリーム・データ・エントリー用のストレージ。
LGUOW	ログ・マネージャー・ドメインの作業単位データ・エントリー用のストレージ。
LI_PLB	言語インターフェースのプログラム言語ブロックのストレージ。それぞれのプログラムごとに、制御が最初にそのプログラムに渡されるときに 1 つ割り振られます。
L2GENRAL	ログ・マネージャー・ドメイン汎用サブプール。
L2OFL2BL	ログ・マネージャー・ドメインのロッガー・ブロック・エントリー用のストレージ。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
L2OFL2BS	ログ・マネージャー・ドメイン・ロガーのブラウズ可能ストリーム・オブジェクト用のストレージ。
L2OFL2CH	ログ・マネージャー・ドメイン・ロガーのチェーン・オブジェクト用のストレージ。
L2OFL2SR	ログ・マネージャー・ドメイン・ロガーのストリーム・オブジェクト用のストレージ。
MDTTABLE	CICS Web インターフェースから送信された BMS マップの MDT フィールド属性テーブル。
ML_GENRL	ML ドメインの汎用ストレージ。
MN_CNTRL	モニター制御ブロック一般情報のストレージ。
MN_TIMAS	モニター制御ブロックの ID モニター領域のストレージ。
MN_TMAS	モニター制御ブロックのトランザクション・モニター領域のストレージ。
MN_TRMAS	モニター制御ブロックのリソース・モニター領域のストレージ。
MQM	WebSphere MQ 通信ストレージ。
MRO_QUEUE	MRO 作業キュー・マネージャーが使用。
MROWORKE	MRO 作業キュー・マネージャー・エレメントが使用。
NQEAS	NQ ドメイン・キュー・エレメント領域のストレージ。
NQGENRAL	NQ ドメインが使用する汎用サブプール
NQPOOL	NQ ドメイン・エンキュー・プールのストレージ。
NQRNAMES	NQRN ディレクトリー項目のストレージ。
OTGENRAL	OT ドメインが使用する汎用サブプール。
OTISINST	OTS トランザクションの未完了状態のストレージ。
OVERLAPD	重複フィールドのマージ用のストレージ。
PGCHCB	チャンネル制御ブロックのストレージ。このストレージには、チャンネルを説明するヘッダー情報が含まれます。
PGCPCB	チャンネル・コンテナ・プール制御ブロックのストレージ。このストレージには、コンテナのセットを説明するヘッダー情報が含まれます。
PGCPCBCH	チェーニングされたコンテナ・プール制御ブロックのストレージ。
PGCRBB	チャンネル・コンテナのブラウズ用ストレージ。
PGCRCB	チャンネル・コンテナ制御ブロックのストレージ。このストレージには、コンテナごとのヘッダー情報が含まれます。
PGCSCB	チャンネル・コンテナ・セグメントのストレージ。
PGGENRAL	汎用プログラム・マネージャー・ドメイン・サブプールのストレージ。
PGHM RSA	プログラム・ハンドル・マネージャー COBOL レジスター保管域のストレージ。
PGHTB	プログラム・マネージャー・ハンドル・テーブル・ブロックのストレージ。
PGJVMCL	JVM クラス名のストレージ。
PGLLE	プログラム・マネージャー・ロード・リスト・エレメントのストレージ。
PGPGWE	プログラム・マネージャー待機エレメントのストレージ。

表 8. ECDSA の CICS サブプール (続き)	
サブプール名	説明
PGPTE	プログラム・マネージャーのプログラム定義のストレージ。
PGPTA	プログラム・マネージャー・トランザクション関連情報のストレージ。
PI_GENRL	パイプライン・マネージャー (PI) ドメインの汎用ストレージ。
PI_POLCY	現在は使用されていません。
PI_PRSER	現在は使用されていません。
PINODEBL	パイプライン・オブジェクトのストレージ。
PIPEINST	パイプライン・オブジェクトのストレージ。
PITKDAT	コンテキスト・トークンのパイプライン・トークン・データのストレージ。
PITKPOOL	パイプライン・トークンのストレージ。
PITXMAST	Web Services Atomic Transaction (WS-AT) マスター制御ブロックまたは PI ドメイン・トランザクション制御ブロックのストレージ。
PR_TABLE	PRT からの PTE のストレージ。
PTWSB	プール・トークンの汎用ストレージ。
RCLELEM	Web 行列エレメント・リストのストレージのストレージ。
RCTABLE	Web テーブル・ストレージ。
RLGENRAL	リソース・ライフサイクル 汎用サブプール。
RMGENRAL	リカバリー・マネージャー 汎用サブプール。
RMOFRMLK	リカバリー・マネージャー・リンク・オブジェクトのストレージ。
RMOFRMUW	リカバリー・マネージャー作業単位オブジェクトのストレージ。
ROWARAY	Web 行配列のストレージ。
RS_FILEL	領域状況 (RS) ドメイン・ファイル・リスト・ストレージ。
RS_GENRL	RS ドメインの制御ブロックのストレージ。
RUNTRAN	トランザクションを実行するトランザクション・マネージャー・サブプール。
RUTKPOOL	再使用可能トークン・クラスのサブプール。
RXGENRAL	RX ドメインの汎用サブプール。
RZGENRAL	要求ストリーム・ドメインの汎用サブプール。
RZOFRSNR	要求ストリーム通知要求のストレージ。
RZOFRSRG	要求ストリーム登録オブジェクトのストレージ。
RZOFRZRS	要求ストリーム・オブジェクトのストレージ。
RZOFRZTR	要求ストリーム・トランスポートのストレージ。
SHGENRAL	スケジューラー・サービス・ドメインの汎用サブプール。
SHOFSHRE	スケジューラー・サービス要求オブジェクトのストレージ。
SJGENRAL	SJVM ドメインの汎用サブプール。
SJJ8TCB	SJVM ドメインの TCB のストレージ。
SMSHRC31	SHARED_CICS31 クラスの多数の制御ブロックのストレージ。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
SMTP	回線および端末入出力域。ストレージ所要量は、システム内の端末の数と回線トラフィックによって決まります。サブプールは、RAPOOL、RAMAX、TIOAL サイズ、および MRO セッション数を削減することによって調整できます。
SOCKET	ソケット・オブジェクトのストレージ。
SOCKPOOL	ソケット・プール・ストレージ。
SOCKSSL	ソケットに関連した SSL データのストレージ。
SOGENRAL	ソケット・ドメイン汎用サブプール。
SOLTE	ソケット・ドメイン・リスナー端末項目のストレージ。
SOSTE	ソケット・ドメイン・ソケット端末項目のストレージ。
SOTBR	ソケット・ドメイン TCIPSERVICE ブラウズ・ブロックのストレージ。
SOTDB	ソケット・ドメイン TCIPSERVICE ブロックのストレージ。
SOTKPOOL	ソケット・ドメイン・ソケット・トークンのストレージ。
STSUBPOL	統計ドメイン・マネージャー・サブプール。
SZSPFCCD	FEPI 接続制御サブプール。
SZSPFCCM	FEPI 共通域サブプール。
SZSPFCCV	FEPI 会話制御サブプール。
SZSPFCDS	FEPI 装置サポート・サブプール。
SZSPFCNB	FEPI ノード初期設定ブロック・サブプール。
SZSPFCND	FEPI ノード定義サブプール。
SZSPFCPD	FEPI プール記述子サブプール。
SZSPFCPS	FEPI プロパティ記述子サブプール。
SZSPFCRP	FEPI 要求パラメーター・リスト・サブプール。
SZSPFCRQ	FEPI 要求サブプール。
SZSPFCSR	FEPI サロゲート・サブプール。
SZSPFCTD	FEPI ターゲット記述子サブプール。
SZSPFCWE	FEPI 作業エレメント・サブプール。
SZSPVUDA	FEPI データ域サブプール。
TA_GENRL	現在は使用されていません。
TASKASOC	ソケット・ドメイン・タスク関連オブジェクトのストレージ。
TD_TDCUB	すべての一時データ CI 更新制御ブロックのストレージ。
TD_TDQUB	すべての一時データ・キュー更新制御ブロックのストレージ。
TD_TDUA	すべての一時データ UOW アンカー制御ブロックのストレージ。
TFUS	TCTTE セキュア拡張のストレージ。
TGODR	トランザクション・グループの起点データ・レコード用のストレージ。
TIA_POOL	タイマー・ドメイン・アンカー・サブプール。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
TIQCPPOOL	タイマー・ドメインの高速セル・サブプールが含まれています。
TSBRB	一時記憶域 (TS) ブラウズ・ブロックのストレージ。
TSBUFFRS	一時記憶域入出力バッファ。ストレージ要件は、以下によって指定されています。 (TS 制御間隔サイズ) x (TS バッファ数)。アプリケーション・プログラムによる一時記憶域の使用は、一時記憶域管理ブロックに関連するサブプール数のサイズに影響します。
TSGENRAL	TSGENRAL サブプールが使用するストレージの量。量は、バッファとストリング数、および一時記憶域データ・セット用に定義されている制御間隔サイズによって決まります。
TSMBR	一時記憶ブラウズ・ブロックのストレージ。
TSMDB	一時記憶域モデル・ブロックのストレージ。
TSQAB	TS キュー・アンカー・ブロックのストレージ。
TSQOB	TS キュー所有権ブロックのストレージ。
TSQUB	TS キュー更新ブロックのストレージ。
TSTSS	TS セクション記述子のストレージ。
TSTSX	TS 補助項目記述子のストレージ。
TSW	TS 待機キュー・エレメントのストレージ。
UE_EPBPL	ユーザー出口プログラム・ブロック (EPB) のサブプール。
USIDTBL	接続セキュリティ・ユーザー ID テーブル項目 (LUIT) のストレージ。詳しくは、ISC/IRC 接続時間エンタリ統計の解釈 を参照してください。
WBGENRAL	CICS Web サポート用汎用サブプール。
WBOUtbND	アウトバウンド HTTP バッファのストレージ。
WBPAThN1	短いパス名の URI マップ・ストレージに使用されるパス・ノード・エレメントのストレージ。
WBPAThN2	長いパス名の URI マップ・ストレージに使用されるパス・ノード・エレメントのストレージ。
WBPAThUR	URI パス名の URI マップ・ストレージに使用されるストレージ。
WBRQB	Web 要求オブジェクトのストレージ。
WBS	IPIC プロトコルに使用されるインバウンド Web セッション・ブロックのストレージ。
WBURIMAP	URI マッピング・エレメントのストレージ。
WBURIXT1	URI マッピング・エレメント拡張 (短い) のストレージ。
WBURIXT2	URI マッピング・エレメント拡張 (長い) のストレージ。
WBWRBR	Web 要求ブラウズ・ブロックのストレージ。
WBVHOST	URI 仮想ホスト・エレメントのストレージ。
WEB_STA	Web 状態関連ストレージ。
WEBELEM	Web 出力エレメント・リストのストレージ。
WEBHTML	Web HTML バッファのストレージ。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
WEBINB	着信データ用 Web ドメイン・ストレージのストレージ。
WEB327B	Web ドメイン 3270 バッファ・ストレージ。
W2ATOMSE	Web 2.0 Atom サービス・エレメントのストレージ。
W2ATOMX1	Web 2.0 Atom サービス拡張のストレージ。
W2ATOMX2	Web 2.0 Atom サービス拡張のストレージ。
W2GENRAL	Web 2.0 ドメインの汎用サブプール。
XMGENRAL	トランザクション・マネージャー用汎用サブプール。
XMTCLASS	トランザクション・マネージャー tranclass 定義のストレージ。
XMTRANSN	トランザクション・マネージャー・トランザクションのストレージ (システム内のトランザクションごとに 1 つ)。
XMTXDINS	トランザクション・マネージャー・トランザクション定義。
XMTXDSTA	トランザクション・マネージャー・トランザクション定義。
XMTXDTPN	トランザクション・マネージャーのトランザクション定義 TPNAME ストレージ。
ZC2RPL	アクティブ・タスク用複写 RPL のストレージ。z/OS Communications Server 端末に関連付けられているそれぞれのアクティブ・タスクには、304 バイト必要です。
ZCBIMG	BIND イメージのストレージ。
ZCBMSEXT	端末用 BMS 拡張のストレージ。各端末、サロゲート、ISC セッション、およびコンソールごとのサブプール・ストレージ要件は 48 バイトです。
ZCBUF	非 LU 6.2 バッファ・リストのストレージ。
ZCCCE	コンソール制御エレメントのストレージ。それぞれのコンソールごとに、48 バイト必要です。
ZCGENERL	端末管理の汎用サブプール。
ZCLUCBUF	LU 6.2 SEND および RECEIVE バッファ・リストのストレージ。
ZCLUCEXT	LU 6.2 拡張のストレージ。ストレージ要件は、それぞれの LU 6.2 セッションごとに 224 バイトです。
ZCNIBD	NIB 記述子のストレージ。それぞれの端末、サロゲート、ISC セッション、およびシステム定義には、96 バイトのストレージが必要です。
ZCNIBISC	拡張 NIB、および ISC の OPNDST および CLSDST 中の応答のストレージ。それぞれの同時ログオンおよびログオフには、448 バイトのストレージが必要です。同時要求の最大数は、セッション数によって制限されています。ストレージは、セッション数を削減することによって調整できる場合があります。
ZCNIBTRM	端末に対する OPNDST および CLSDST の際の拡張 NIB のストレージ。それぞれの同時ログオンおよびログオフには、192 バイトのストレージが必要です。同時要求の最大数は、端末数によって制限されています。ストレージは、端末数を削減することによって調整できる場合があります。
ZCRAIA	RECEIVE ANY 入出力域のストレージ。
ZCRPL	アクティブ・タスク用 RPL のストレージ。z/OS Communications Server 端末に関連付けられているそれぞれのアクティブ・タスクには、152 バイト必要です。

表 8. ECDSA の CICS サブプール (続き)

サブプール名	説明
ZCSETB	16 MB 境界より上だが 2 GB 境界より下のアプリケーション制御バッファのストレージ。
ZCSKEL	リモート端末項目のストレージ。それぞれのリモート端末定義には、32 バイトのストレージが必要です。
ZCSNEX	TCTTE サインオン拡張のストレージ。各端末、サロゲート、セッション、およびコンソールごとのストレージ要件は 48 バイトです。
ZCTCME	モード項目のストレージ。それぞれのモード項目には、128 バイトのストレージが必要です。
ZCTCSE	システム項目のストレージ。それぞれのシステム項目には、192 バイトのストレージが必要です。
ZCTCTTEL	大容量端末項目のストレージ。それぞれの定義済み端末、サロゲート・モデル、および ISC セッションには、504 バイトのストレージが必要です。
ZCTCTTEM	中容量端末項目のストレージ。それぞれの IRC バッチ端末ごとに、400 バイトのストレージが必要です。
ZCTCTTES	小容量端末項目のストレージ。それぞれの MRO セッションおよびコンソールごとに、368 バイトのストレージが必要です。
ZCTPEXT	TPE 拡張。
ZCTREST	端末管理のトランザクション再始動サブプール。
ZCTCTUA	TCTTE ユーザー域用のストレージ。これは、次の DSA、すなわち SDSA、ECDSA、CDSA、または ESDSA のいずれか 1 つに配置できます。配置場所は、システム初期設定パラメーター TCTUALOC=ANY BELOW およびシステム初期設定パラメーター TCTUAKEY=CICS USER によって制御されます。最大サイズは、端末定義の USERAREALEN オペランドで指定できます。端末定義について詳しくは、 TERMINAL ターミナル を参照してください。

ESDSA の CICS サブプール

拡張共用動的ストレージ域 (ESDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 9. ESDSA の CICS サブプール

サブプール名	説明
DFHAPUAN	16 MB 境界から上だが 2 GB 境界から下のアプリケーション・ドメイン・ストレージの汎用サブプール。
IE_BUFF	インバウンドおよびアウトバウンド・メッセージを処理するときに使用される、IE ドメイン・バッファ。
IIBUFFER	IIOP ドメイン・バッファ・サブプール。
IS_BUFF	メッセージ・データを IS セッション・ブロック用に保持するために使用される、IS バッファのストレージ。
LDEPGM	拡張 (31 ビット) 動的ロード・アプリケーション・プログラムおよび EXECKEY(USER) で定義されたプログラムのストレージ。
LDERES	拡張 (31 ビット) 常駐アプリケーション・プログラムのストレージ。
SJSCCHS	Java 仮想マシン・ドメイン (SJ ドメイン) クラス・キャッシュ用のストレージ。
SJSJPTE	SJ ドメイン・プロファイル・テーブル項目用のストレージ。

表 9. ESDSA の CICS サブプール (続き)	
サブプール名	説明
SJSJTCB	SJ ドメイン TCB 使用のストレージ。
SJUSERKY	SJ ドメイン・ユーザー・キー・ストレージ。
SMSHRU31	SHARED_USER31 クラス・ストレージの多数の制御ブロック、RMI グローバル作業域、モニター中のトランザクションの存続期間中の EDF ブロック、およびその他の制御ブロック用に使用。
WEBINB	Web 3270 バッファ・ストレージ。
ZCTCTUA	TCTTE ユーザー域用のストレージ。これは、次の DSA、すなわち SDSA、ECDSA、CDSA、または ESDSA のいずれか 1 つに配置できます。配置場所は、システム初期設定パラメーター TCTUALOC=ANY BELOW およびシステム初期設定パラメーター TCTUAKEY=CICS USER によって制御されます。最大サイズは、端末定義の USERAREALEN オペランドで指定できます。端末定義について詳しくは、 TERMINAL ターミナルを参照してください。

ERDSA の CICS サブプール

拡張読み取り専用動的ストレージ域 (ERDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 10. ERDSA の CICS サブプール	
サブプール名	説明
LDENRSRO	RESIDENT である拡張 CICS 中核、および 31 ビット・マクロ・テーブル用のストレージ。拡張 CICS 中核は、約 1850 KB です。このサブプールの内容は、再入可能な状態でリンクされている必要があります。
LDENUCRO	RESIDENT ではない拡張 CICS 中核、および 31 ビット・マクロ・テーブル用のストレージ。拡張 CICS 中核は、約 1850 KB です。このサブプールの内容は、再入可能な状態でリンクされている必要があります。
LDEPGMRO	拡張 (31 ビット) 動的ロード・アプリケーション・プログラムのストレージ。このサブプールの内容は、再入可能な状態でリンクされている必要があります。
LDERESRO	拡張 (31 ビット) 常駐アプリケーション・プログラムのストレージ。このサブプールの内容は、再入可能な状態でリンクされている必要があります。

ETDSA の CICS サブプール

拡張信頼動的ストレージ域 (ETDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 11. ETDSA の CICS サブプール	
サブプール名	説明
USGENRAL	ユーザー・ドメインの汎用サブプール。
USRTMQUE	USRDELAY を待機中のユーザーのキュー・エレメントのストレージ。それぞれのキュー・エレメントは 16 バイトです。
USUDB	ユーザー・データ・ブロックのストレージ。ストレージ要件は、固有ユーザーごとに 128 バイトです。
USXDPOOL	ユーザー・ドメイン・トランザクション関連データのストレージ。それぞれの実行中のトランザクションごとに、32 バイト必要です。
XSGENRAL	セキュリティ・ドメインの汎用サブプール。
XSXMPPOOL	セキュリティ・ドメイン・トランザクション関連データのストレージ。それぞれの実行中のトランザクションごとに、56 バイト必要です。

GCDSA の CICS サブプール

2 GB 境界より上の CICS 動的ストレージ域 (GCDSA) 内のサブプールのリストを、それぞれの使用法と共に記載します。

表 12. GCDSA の CICS サブプール	
サブプール名	説明
CPSM_64	System Management Single Server (SMSS) 環境での CICSplex SM API 結果セット用のストレージ
CQCQ_TR	コンソール・キュー処理トレース・テーブルのストレージ。
CQCQ_XT	コンソール・キュー・トランザクション・エントリー・テーブルのストレージ。
DFHAPD64	64 ビット・アプリケーション・ドメイン・ストレージの汎用サブプール。
DFHMPMDR	管理対象プラットフォーム・モデル規則 (MPMODR) 制御ブロックの割り振りに使用します。
DFHMPPMB	管理対象プラットフォーム・ポリシー修飾子 (MPPMB) 制御ブロックの割り振りに使用します。
DFHMPPRB	管理対象プラットフォーム・ポリシー規則 (MPPRB) 制御ブロックの割り振りに使用します。
DFHMPTAS	管理対象プラットフォーム・タスク存続時間ストレージ (MPTAS) 制御ブロックの割り振りに使用します。
EP_64	CICS イベント処理に使用される、イベント・キャプチャー・キュー内の項目の制御ブロック用のストレージ。
LD_APES	ローダー・ドメインのアクティブ・プログラム・エレメント (APE) 制御ブロック。
LD_CPES	ローダー・ドメインの現在のプログラム・エレメント (CPE) 制御ブロック。
LD_CSECT	ローダー・ドメインの CSECT リスト・ストレージ。
ML64GNRL	z/OS XML System Services (XMLSS) パーサーの入出力用のバッファ。
MN_ADCS	関連データ制御ブロック用のストレージ。
MPSTAT	イベント・アクションを含むポリシー規則で定義された静的データ・キャプチャー項目の制御ブロック用のストレージ。
PGCSDB	チャンネル・コンテナ・セグメントのストレージ (セグメント・ヘッダーを含む)。
PGPTE64	アプリケーション・コンテキスト・データ制御ブロック用のストレージ。
SMSHRC64	SHARED_CICS64 クラス・ストレージの多数の制御ブロックで使用。
SOGNRL6	ソケット・ドメインの HTTPS データ用のストレージ。
TSDTN	一時記憶域 (TS) デジタル・ツリー・ノード。
TSICDATA	TS インターバル制御エレメント。
TSMN	主一時記憶域のストレージ。
TSMN0064	ヘッダーを含めた長さが 64 バイト以下の主一時記憶域の項目の固定長エレメント。ヘッダーの長さは 8 バイトです。
TSMN0128	主一時記憶域の項目の 128 バイト固定長エレメント。
TSMN0192	主一時記憶域の項目の 192 バイト固定長エレメント。
TSMN0256	主一時記憶域の項目の 256 バイト固定長エレメント。
TSMN0320	主一時記憶域の項目の 320 バイト固定長エレメント。

表 12. GCDSA の CICS サブプール (続き)	
サブプール名	説明
TSMN0384	主一時記憶域の項目の 384 バイト固定長エレメント。
TSMN0448	主一時記憶域の項目の 448 バイト固定長エレメント。
TSMN0512	主一時記憶域の項目の 512 バイト固定長エレメント。
TSMN0576	主一時記憶域の項目の 576 バイト固定長エレメント。
TSMN0640	主一時記憶域の項目の 640 バイト固定長エレメント。
TSMN0704	主一時記憶域の項目の 704 バイト固定長エレメント。
TSMN0768	主一時記憶域の項目の 768 バイト固定長エレメント。
TSMN0832	主一時記憶域の項目の 832 バイト固定長エレメント。
TSMN0896	主一時記憶域の項目の 896 バイト固定長エレメント。
TSMN0960	主一時記憶域の項目の 960 バイト固定長エレメント。
TSMN1024	主一時記憶域の項目の 1024 バイト固定長エレメント。
TSQUEUE	TS キュー記述子。
TSTSI	TS 項目記述子。
WB64GNRL	Web ドメイン (WB) の HTTP データ用のストレージ。
WBOU64	Web ドメイン (WB) のアウトバウンド用の汎用サブプール。
WU_64	CMCI の保持される結果およびメタデータ用のストレージ。
XMG64	64 ビット・ストレージの汎用サブプール。

GSDSA の CICS サブプール

2 GB 境界より上の共用動的ストレージ域 (GSDSA) にあるサブプールを、それぞれの使用法と共にリストします。

表 13. GSDSA の CICS サブプール	
サブプール名	説明
DFHAPU64	2 GB 境界より上のアプリケーション・ドメイン・ストレージの汎用サブプール。
SMSHRU64	SHARED_USER64 クラス・ストレージの多数のブロックが使用。

CICS カーネル・ストレージ

CICS カーネル・ストレージでは、CICS が、システムおよびユーザー・タスクを CICS の実行中を通して管理するために必要な制御ブロックおよびデータ域で構成されています。このストレージの大部分は、CICS DSA から割り振られます。このストレージの少量のストレージは、MVS ストレージから割り振られています。

カーネルは、2つのタスク、静的タスクおよび動的タスクを識別します。静的タスクのカーネル・ストレージは、事前割り振りされ、MXT メカニズムによって制御されるタスクが使用します。動的タスクのストレージは事前割り振りされず、MXT 値によって制御されないシステム・タスクなどのタスクが使用します。動的タスクのストレージは事前割り振りされないため、カーネルは、タスクの接続時に動的タスクの接続に必要なストレージを取得するために GETMAIN コマンドを使用する必要があります。

静的タスクの数は、現在の MXT 値によって異なります。MXT+1 個の静的タスクがあります。静的タスクのストレージは、必ず CICS DSA から GETMAIN によって取得されます。MXT が減らされると、超過している静的タスク数分のストレージが再び解放されます。

CICS 初期設定の早い段階で、カーネルは 8 つの動的タスク用にストレージを割り振ります。このストレージは、MVS から GETMAIN によって取得され、常に内部 CICS タスクが使用できます。次に、動的タスク

用のこれ以外のすべてのストレージが、必要に応じて CICS DSA から割り振られます。通常、動的タスクの終了時に、関連するストレージは解放されます。

1 つのタスクに対してタスクの初期化時に CICS が割り振るストレージは、静的タスクも動的タスクも同じで、以下のようになります。

- 1576 バイトのカーネル・タスク項目
- 28K 31 ビット・スタック

割り振られるストレージは、すべて 16 MB 境界より上にあります。CICS では、タスクの初期化時に各タスクに対して 24 ビット・スタック (16 MB 境界より下) は割り振らなくなりました。

タスクの初期化時に割り振られるストレージに加えて、カーネルは 16 MB 境界の上と下の両方に、拡張スタック・セグメントのプールも割り振ります。

- 各 31 ビット拡張スタック・セグメント (16 MB 境界より上) のサイズは、8 KB です。すべてのタスクは、割り振られた 31 ビット・スタック・ストレージがオーバーフローした場合、これらの拡張スタック・セグメントを使用できます。CICS は、いくつかの 31 ビット拡張スタック・セグメントを含むプールを事前割り振りします。この数は、現在の MXT 値を 10 で除算して算定されます。
- 各 24 ビット拡張スタック・セグメント (16 MB 境界より下) のサイズは、4 KB です。タスクは、24 ビット・スタック・ストレージが必要になるたびに、これらの拡張スタック・セグメントを取得します。CICS は、使用可能な 24 ビット・スタック・ストレージが他にない場合にタスクが使用できる、24 ビット拡張スタック・セグメントの予約プールを事前割り振りします。

カーネルが、GETMAIN を使用して CICS DSA からストレージを取得する場合には、以下のサブプールが使用されます。

CDSA の KESTK24E

4 KB 拡張スタック・セグメント、24 ビット

ECDSA 内の KESTK31

28 KB スタック・セグメント、31 ビット

ECDSA 内の KESTK31E

8 KB 拡張スタック・セグメント、31 ビット

ECDSA 内の KETASK

1576 バイト・カーネル・タスク項目

64 ビット MVS ストレージ

64 ビット MVS ストレージは、オペレーティング・システムが領域関連のサービスを実行するために使用可能です。

アドレス・スペース内の 64 ビット (2 GB 境界より上) ストレージについては、『[z/OS MVS Programming: Extended Addressability Guide](#)』の『[64 ビット・アドレス・スペースの使用](#)』を参照してください。

領域内で Java プログラムを実行する場合、CICS は z/OS 上の 64 ビット JVM を使用します。CICS の制御下で実行される各 JVM に、64 ビット MVS ストレージが割り振られます。

64 ビット MVS ストレージを使用するその他の CICS 機能については、[89 ページの『64 ビット・ストレージを使用する CICS 機能』](#)を参照してください。

2 GB より下の MVS ストレージ

2 GB より下の MVS ストレージは、オペレーティング・システム・マクロ、または領域によって発行された SVC に応答して、オペレーティング・システムが領域関連サービスを実行するのに使用できます。

例えば、Java、Node.js、VSAM、DL/I、または Db2 などの言語およびコンポーネントは、制御ブロックを作成するストレージを取得するために、MVS GETMAIN 要求を発行します。これらの要求は、2 GB より下の MVS ストレージから対応されます。

MVS ストレージは、動的ストレージ域および他の CICS ストレージ要件が満たされた後に残る量のストレージです。2 GB より下の MVS ストレージのサイズは、CICS の実行時の MVS GETMAIN 要件によって決まります。この領域は、ファイルのオープンで主に使用します。

2 GB より下の MVS ストレージは、以下の項目を収容するために使用されます。

- オープン・データ・セットまたは他のオペレーティング・システム機能用に必要な制御ブロックおよびデータ域
- リンク・パック域 (LPA) に既に常駐していないアクセス方式ルーチンのプログラム・モジュール
- COBOL および PL/I プログラムの共用ルーチン

2 GB より下の MVS ストレージ内の仮想記憶には、4 つの主要なエレメントがあります。16 MB より下の各ストレージ域は、16 MB より上で重複しています。

- 16 MB から下の共通域
- 16 MB から下の専用域
- 16 MB から上の拡張共通域
- 16 MB から上の拡張専用域

CICS が他の製品を使用する場合のストレージ

VSAM バッファおよび大部分の VSAM ファイル制御ブロックは、16 MB より上にあります。VSAM バッファは、ローカル共用リソース (LSR)、または非共用リソース (NSR) を使用するように定義された CICS データ・セット用です。VSAM LSR プールは、そのプールを使用するように指定された最初のファイルが開かれたときに 16 MB より上に動的に作成され、そのプールを使用する最後のファイルが閉じられたときに削除されます。それぞれのオープンされたデータ・セットごとに、入出力ブロック (IOB) やチャネル・プログラムなどの項目用に、この領域に一定の量のストレージが必要です。

データ・テーブルとして定義されているファイルは、そのテーブルに含まれているレコード、およびそれらのファイルへのアクセスを可能にする構造用に、16 MB から上のストレージを使用します。

待機順次アクセス方式 (QSAM) ファイルは、この領域に一定量のストレージが必要です。一時データは、一時データ・キューのタイプごとに、16 MB より上の別々のバッファ・プールを使用します。ストレージは、一時データ・キュー・リソースがインストールされるときに、一時データ・キュー・リソース用のバッファ・プールから取得されます。一時データは、16 MB より上のバッファ・プールも使用します。このプールには、区画外キューのオープンまたはクローズ時に、QSAM が使用する区画外一時データ・キュー定義のセクションがコピーされます。

CICS DBCTL は、DBCTL スレッドを使用します。DBCTL スレッドは、CICS アドレス・スペースで指定されます。ただし、これらのスレッドには、CICS アドレス・スペースの高専用域内のストレージ要件があります。CICS が Db2 を使用する場合、Db2 スレッドごとに MVS ストレージが割り振られます。

MVS ストレージの制限

2 GB より下の MVS ストレージは、物理的には領域内のどこに配置されてもよく、場合によっては、CICS 領域より上に配置される場合もあります。領域は、その領域の上のこの MVS ストレージ域にインストールで設定された IEALIMIT まで、またはデフォルト値まで拡張されます。IEALIMIT について詳しくは、[z/OS MVS 導入システム 出口](#)を参照してください。この拡張は、オペレーティング・システム GETMAIN 要求が発行されたとき、領域内の MVS ストレージがいっぱいになったとき、および要求がその領域から上の MVS ストレージ域で満たされたときに発生します。

2 GB より下の両方の MVS ストレージ域がいっぱいになると、GETMAIN 要求は失敗し、異常終了するか、その要求が条件要求だった場合には、不良の戻りコードを戻します。

2 GB より下の MVS ストレージの量は、CICS 領域の実行中を通して、ストレージに対する要求を十分に満たす量でなければなりません。MVS ストレージを使い果たさないようにし、かつ、MVS ストレージを割り振り過ぎることのないように注意してください。

2 GB より下の MVS ストレージのサイズは、動的ストレージ域、カーネル・ストレージ域が必要とするストレージの割り振り後に領域に残ったストレージと、IMS/VS および DBRC モジュール・ストレージです。MVS ストレージで必要な量を領域内で使用可能にするには、適切な DSA サイズを指定することが重要です。

CICS システムの動的な特質のため、MVS ストレージへの要求は一日を通して、タスクの数の増加、またはデータ・セットのオープンやクローズによって変化します。MVS ストレージのこのような動的な使用によってフラグメント化が発生するため、これを補うために、追加ストレージを割り振る必要があります。詳しくは、[JVM サーバーのストレージ要件の計算](#)を参照してください。

MVS 共通域

MVS 共通域には、多数の中核、キュー、リンク・パック、共通サービス、およびストレージ域が含まれます。

以下の領域が MVS 共通域を構成します。

- 中核および拡張中核
- システム・キュー域 (SQA および ESQA)
- リンク・パック域 (PLPA、MLPA、および CLPA)
- 共通サービス域 (CSA および ECSA)
- 接頭部ストレージ域 (PSA)。

PSA を除く、共通域のこれらのエレメントはすべて、16 MB より上で重複します。

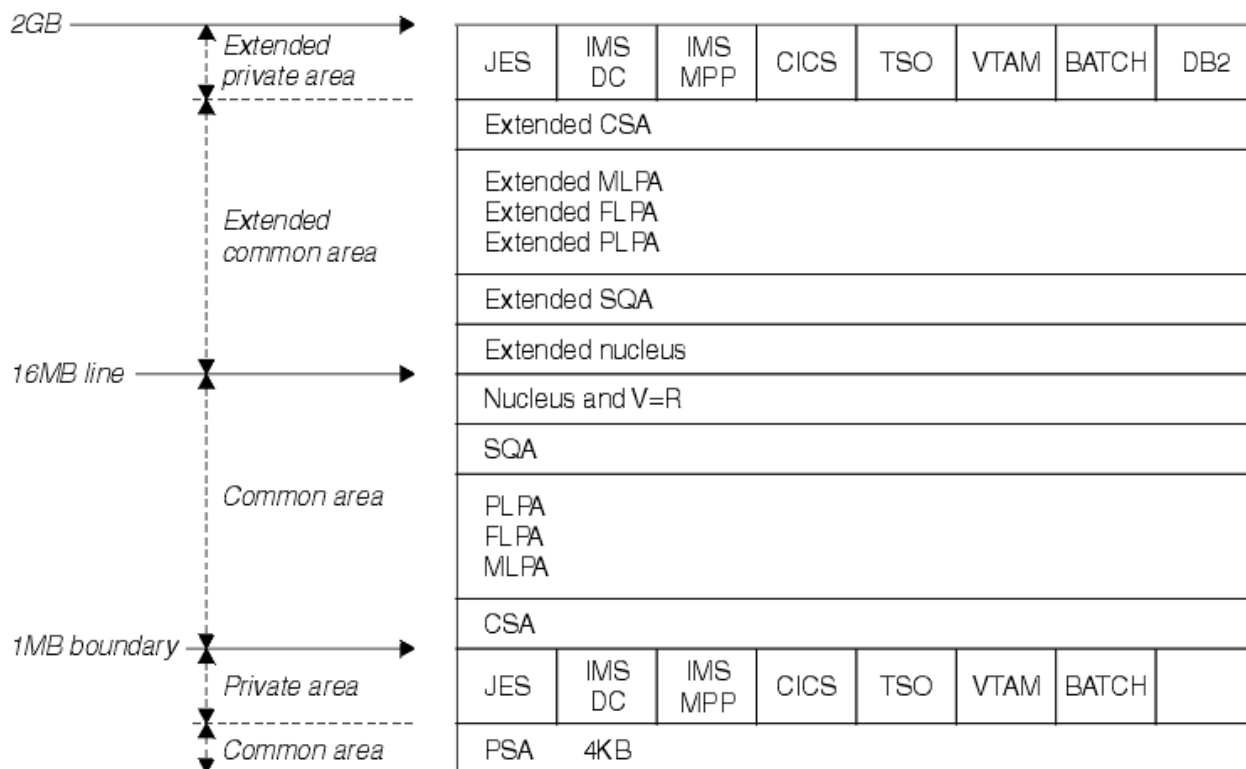


図 17. 仮想記憶マップ

MVS 中核および MVS 拡張中核

MVS 中核および MVS 拡張中核は、中核ロード・モジュール、およびその中核の拡張を含む静的領域です。ただし、この領域のサイズはインストールの構成によって変化しますが、MVS の再 IPL を使用せずに変更することはできません。

16 MB より下の中核域には、ページ枠テーブル項目は含まれません。中核域のサイズは、4 KB 境界で切り上げられます。また、中核域は 16 MB マップの先頭に配置され、これとは反対に、拡張中核は 16 MB のすぐ上に配置されます。

システム・キュー域 (SQA) および拡張システム・キュー域 (ESQA)

この領域には、システム全体に関係するテーブルおよびキューが含まれています。この領域の内容は、インストール時の構成およびジョブ要件に強く依存しています。

仮想記憶の総量、専用仮想記憶アドレス・スペースの数、およびインストール・システム・パフォーマンス仕様テーブルのサイズは、システムが SQA を使用するときに影響する要因です。SQA の初期割り振りサイズは、64 KB 境界に切り上げられます。ただし、SQA は共通システム域 (CSA) に 4 KB の増分で拡張できます。

SQA の割り振りが大きすぎる場合は、仮想記憶が永続的に無駄になります。割り振りが小さすぎる場合には、必要であれば CSA に拡張します。ストレージに制約があるシステムでは、割り振りがやや小さめの方が推奨されています。割り振りは、フリー・ストレージの量を確認することによって決定します。拡張 SQA の割り振りが小さすぎる場合には、拡張 CSA に拡張します。拡張 SQA および拡張 CSA が共に完全に使用されると、システムはストレージを 16 MB 境界から下の SQA および CSA から割り振ります。このストレージの割り振りは、最終的にシステム障害を引き起こすことがあります。このため、拡張 SQA および拡張 CSA は、大きめに割り振ることが推奨されています。

リンク・バック域 (LPA) および拡張リンク・バック域 (ELPA)

リンク・バック域 (LPA) には、システムによって共用されるすべての共通再入可能モジュールが含まれます。

リンク・バック域 (LPA) には、以下の利点があります。

- モジュールの 1 つのコピーを共用することによって実記憶域が節約できる。
- 保護: LPA コードは、キー 0 プログラムによる場合であっても上書きできない。
- モジュールがブランチ可能になるため、パス長さが削減される。

MVS が、MRO または ISC で CICS を使用する場合は、2 MB の LPA で十分であることは証明されています。このサイズは、IBM が LPA を出荷したときの変更を加えていないサイズです。このサイズがより大きい場合は、CICS には利点がないモジュールが、LPA にロードされています。これらは、バッチ・ユーザーおよび TSO ユーザーに利点がある SORT、COBOL、ISPF、およびその他のモジュールです。ユーザーは、現在得られている利点が、使用する仮想記憶に見合う価値があるかどうかを評価する必要があります。モジュールが除去されている場合は、そのモジュールを実行する領域のサイズを、そのモジュールに合わせて増加する必要があるかどうかを確認してください。

ページング可能リンク・バック域 (PLPA) には、監視プログラム呼び出しルーチン (SVC)、アクセス・メソッド、その他読み取り専用システム・プログラム、およびシステムのユーザー間で共用するインストールで選択した読み取り専用再入可能ユーザー・プログラムが含まれています。インストールでシステム生成中に選択されたオプションの機能および装置によって、追加のモジュールが PLPA に追加されます。

変更済みリンク・バック域 (MLPA) には、PLPA の拡張であるモジュールが含まれています。MLPA は、PLPA 内のモジュールを変更する場合、IPL 時に変更できます。このため、IPL 時にリンク・バック域の作成 (CLPA) オプションは必要ありません。

共通サービス域 (CSA) および拡張共通サービス域 (ECSA)

CSA および ECSA には、すべてのアクティブな仮想記憶アドレス・スペースによってアドレッシング可能なページング可能システム・データ域が含まれています。

例えば、これらのサービス域には、IMS、ACF/SNA、JES3 のバッファまたは実行可能モジュールが含まれています。CSA および ECSA には、サブシステムを定義するために使用する制御ブロックも含まれます。制御ブロックは、TSO 入出力制御 (TIOC)、イベント通知機能 (ENF)、およびメッセージ処理機能 (MPF) などの領域用に作業用ストレージを提供します。システム構成およびアクティビティが増加すると、ストレージ要件も増加します。

CICS では、データ転送のためではなく、制御ブロックを保管するためにのみ、複数領域操作 (MRO) で ECSA を使用します。仮想記憶間機能が使用されている場合は、ECSA 使用量は、以下の量に限定されます。

- リソースが獲得されていてサービス中であるか、あるいは解放されているかにかかわらず、IRC (領域間通信) がオープンである場合、セッションごとに 40 バイト
- MRO に組み込まれているアドレス・スペースごとに 4 KB

加えて、CICS MRO が領域間バッファに使用するストレージの量は、終了時に CSMT の宛先に対して発行される DFHIR3794 メッセージで詳述されます。

CICS は、IMS の ECSA および共用データ・テーブルも使用します。

静的システムの場合は、割り振られていない CSA の量は、割り振られている CSA の総量の約 10% にすべきです。動的システムの場合は、最適な値は 20% です。SQA とは異なり、CSA が完全に使用されている場合は、拡張する場所はなく、再 IPL が必要になる場合があります。

接頭部ストレージ域 (PSA)

PSA には、プログラム状況ワード (PSW) などのプロセッサ特定の状況情報が含まれています。プロセッサごとに 1 つの PSA があります。ただし、すべての PSA は、その特定のプロセッサから確認できるように、仮想記憶位置 0 MB から 4 KB にマップされます。

MVS では、PSA は別の領域で処理されます。拡張共通域には PSA はありません。

専用領域および拡張専用域

ユーザーのアプリケーション・プログラムから使用可能なそれぞれの仮想アドレス・スペース内のユーザー専用領域部分は、そのプログラムの領域と呼ばれます。16 KB システム領域を除き、専用域内のそれぞれのストレージ域には、拡張専用域に対応する領域があります。

専用領域には、以下の領域が含まれています。

- ローカル・システム・キュー域 (LSQA)
- スケジューラー作業域 (SWA)
- サブプール 229 および 230 (リクエスト保護キー域)
- 16 KB システム領域 (イニシエーターが使用)
- プログラムの実行およびデータの保管を行う専用ユーザー領域。

121 ページの図 17 で、MVS の仮想記憶マップを参照してください。

専用領域のユーザー領域のサイズは、専用領域全体のサイズ (接頭部ストレージ域 (PSA) の上端から共通サービス域 (CSA) の先頭、または CSA の終端まで) から、LSQA、SWA、サブプール 229 と 230、およびシステム領域のサイズ (例えば 220 KB) を差し引いたサイズまでの任意のサイズにすることができます。リカバリ終了管理 (RTM) 処理を可能にするために、領域は 420 KB を差し引いたサイズにすることが推奨されます。

セグメント・サイズは 1 MB です。したがって、CSA は一番近い MB に切り上げられます。専用領域の増分は、1 MB です。

高専用領域

アドレス・スペースの先頭の領域は、CICS では特に使用されません。オペレーティング・システムが領域とその要件をサポートするために必要な情報と制御ブロックが入っています。

高専用領域は、以下の 4 つの領域で構成されています。

- LSQA
- SWA
- サブプール 229
- サブプール 230

高専用領域の通常のサイズは、ジョブ制御ステートメントの数、システム・ログへのメッセージ、オープンされているデータ・セットの数、および USS 共用ライブラリー領域の使用量によって異なります。

この領域で使用されるスペースの合計は、ジョブ・ステップの終了時に、IEF374I メッセージの SYS=nnnnK というラベルのフィールドに示されます。16 MB より上の高専用領域を参照する、2 番目の SYS=nnnnK が発行されています。この情報は、サンプルの統計プログラム DFHOSTAT でも示されています。

この領域のサイズを削減することはできません。ただし、LSQA およびサブプール 229 の共用ライブラリー領域の使用量は削減できます。共用ライブラリー使用の最適化について詳しくは、[z/OS 共用ライブラリー領域の調整](#)を参照してください。

サブプール 229 は、CICS が z/OS Communications Server に対してオープン受信を発行していない場合に、z/OS Communications Server がインバウンド・メッセージを保管する場所です。このサブプールが使用されているかどうかを判別するには、CICS のシャットダウン後に取得された CICS 統計を使用します。シャットダウン統計で通知された RPL の最大数を SIT の RAPOOL 値と比較します。これらの値が等しい場合は、サブプール 229 がメッセージをステージングするために使用されている可能性が高いため、RAPOOL 値を増やす必要があります。

場合によっては、高専用領域内のストレージの使用方法が原因で S80A 異常終了が発生することがあります。少なくとも以下の 2 つの事項を考慮してください。

- SNA などのアクセス方式による MVS サブプール 229 および 230 の使用。

SNA と VSAM は、サブプール 229 と 230 の要求がストレージ不足になったことを検出することがあります。これらの要求は条件付きのため、ジョブ・ステップの S80A 異常終了は発生しません (例えば CICS)。

- ジョブ・ステップの開始中の LSQA および SWA ストレージの使用に関連する MVS オペレーティング・システム自体。

MVS イニシエーターによる LSQA および SWA ストレージの使用は、CICS が MVS START コマンドを使用して開始されたのか、既存のイニシエーターとアドレス・スペースの一部のジョブ・ステップとして開始されたのかによって異なります。MVS START コマンドで CICS を開始することは、領域境界から上のスペース内のフラグメント化を最小化するために有効です。CICS が、直前に開始されたイニシエーターのアドレス・スペースで開始されたジョブ・ステップの場合は、LSQA および SWA ストレージの割り振られ方によっては、使用可能な仮想記憶が明らかに縮小することがあります。これは、フラグメント化が進んだためです。

領域境界から上のストレージは、MVS イニシエーター (LSQA および SWA)、およびアクセス・メソッド (サブプール 229 および 230) が使用できるようになっている必要があります。

ユーザー指定の領域サイズから上のスペースのフラグメント化を最小化するには、MVS START コマンドを使用して CICS を開始することを検討してください。使用可能なストレージをより効果的に使用することができ、S80A 異常終了の発生を防止できる可能性があります。

MVS 中核、MVS 共通システム域、および CICS 領域のユーザーが選択するサイズは、LSQA、SWA、およびサブプール 229 と 230 の使用可能なストレージの量に影響します。MVS 中核と共通システム域のサイズと境界は、簡単に変更できるものではありません。LSQA、SWA、およびサブプール 229 と 230 のスペースを増やすには、領域サイズを削減することが必要な場合があります。

サブプールおよび専用ストレージ割り振りの管理について詳しくは、『[z/OS MVS Programming: Authorized Assembler Services Guide](#)』の『[Virtual storage management](#)』を参照してください。

ローカル・システム・キュー域 (LSQA)

この領域には、通常、ストレージおよびコンテンツ・スーパービジョンの制御ブロックが含まれています。この領域には、オペレーティング・システムのリリース・レベルに応じて、サブプールの 233、234、235、253、254、または 255 が含まれます。

LSQA の合計サイズは、ロード済みプログラム、タスク、およびアドレス・スペース内のその他のサブプールの数とサイズに応じて異なるため、計算が困難です。指針として、LSQA 領域は、通常、残りの CICS アドレス・スペースの複雑さに応じて、40 KB から 170 KB の間で稼働します。

ストレージ管理ブロックは、これらのサブプール内の空き領域と割り振り領域を記述する専用域内のストレージ・サブプールを定義します。この定義は、サブプール・キュー・エレメント (SPQE)、記述子キュー・エレメント (DQE)、およびフリー・キュー・エレメント (FQE) などの項目で構成されています。

コンテンツ・マネージメント制御ブロックは、タスク制御ブロック (TCB)、さまざまな形式の要求ブロック (RB)、内容ディレクトリー・エレメント (CDE)、およびその他のアドレス・スペース内にタスクとプログラムを定義します。

CICS DBCTL では、DBCTL スレッド用に LSQA ストレージが必要です。それぞれの DBCTL スレッドごとに、9 KB から **MAXTHRED** 値まで許可されます。

スケジューラー作業域 (SWA)

スケジューラー作業域 (SWA) は、サブプール 236 および 237 で構成されています。ここには、ジョブとステップ自体に関する情報が含まれています。ステップのジョブ・ストリームに現れるほとんどすべては、ここである種の制御ブロックを作成します。

通常、この領域は、DD ステートメントの数の増分で増加すると考えることができます。サブプール 236 と 237 内のストレージの配布は、オペレーティング・システム・リリース、および動的割り振りが使用されているかどうかによって変化します。これらのサブプール内のストレージの合計量は、100 から 150 KB で始まり、割り振られたデータ・セットごとにおよそ 1 から 1.5 KB 増加します。

SWA 制御ブロックのサブセットは、オプションで 16 MB から上に常駐できます。JES2 および JES3 には、これを制御するパラメーターがあります。これを個々のジョブ・ベースで行う必要がある場合は、SMF 出口である IEFUJV が使用できます。

サブプール 229

このサブプールは、主にメッセージのステージングで使われます。JES は、システム・ログに印刷するメッセージ、JCL メッセージ、および SYSIN/SYSOUT バッファ用はこのプールを使われます。

通常は、SYSIN と SYSOUT データ・セットの数、およびシステム・ログ内のメッセージ数に応じて、40 KB から 100 KB の値が許可されます。

サブプール 230

このサブプールは、セグメント化されたメッセージのインバウンド・メッセージ・アセンブリー用に、z/OS Communications Server によって使われます。ここでのデータ管理は、すべてのオープンされているデータ・セット用にデータ・エクステント・ブロック (DEB) を保持します。

通常は、オープンされているデータ・セットの数が増加すると、サブプール 230 のサイズが増加します。開かれているデータ・セットごとに、40 KB から 50 KB の初期値から始めて、300 バイトから 400 バイトまで許可されます。

CICS DBCTL では、DBCTL スレッド用にサブプール 230 ストレージが必要です。それぞれの DBCTL スレッドごとに、3 KB から **MAXTHRED** 値まで許可されます。

領域を超える MVS ストレージ

領域を超える MVS ストレージは、領域の先頭から高専用領域の下部の間に残っているストレージです。通常は、異常終了の場合に備えて、終了ルーチンが使用できるように、200 KB から 300 KB のフリー・ストレージが維持されています。

このフリー・ストレージが、リカバリー終了管理 (RTM) 処理のために不足している場合には、アドレス・スペースは S40D 異常終了で終了します。この場合には、ダンプは生成されません。

この領域は非常に動的な場合があります。高専用領域が増加すると、この領域まで拡張され、CICS 領域は、**IEALIMIT** で指定されている値までこの領域の上に拡張します。

CICS の MVS ストレージのモニター

MVS ストレージの使用を最適化し、ストレージ不足 (SOS) 状態による領域の異常終了を回避するために、24 ビットおよび 31 ビットのユーザー領域ストレージに対して CICS モニター・サポートおよび SOS 待機機能を使用できます。

126 ページの図 18 では、CICS が 24 ビット (ユーザー領域) および 31 ビット (拡張ユーザー領域) のストレージを使用する方法を図で示しています。CICS は、最初に CICS 動的ストレージ域 (DSA) を領域ストレージから事前割り振りし、このストレージを管理します。その他のストレージは、CICS、z/OS コンポーネント、ペンダー・プログラム、またはアプリケーションが z/OS GETMAIN 要求または STORAGE 要求 (このトピックでは GETMAIN と呼ばれる) を発行する際に使われます。ストレージが不足しているときに無条件 GETMAIN が発行されると、領域が異常終了します。

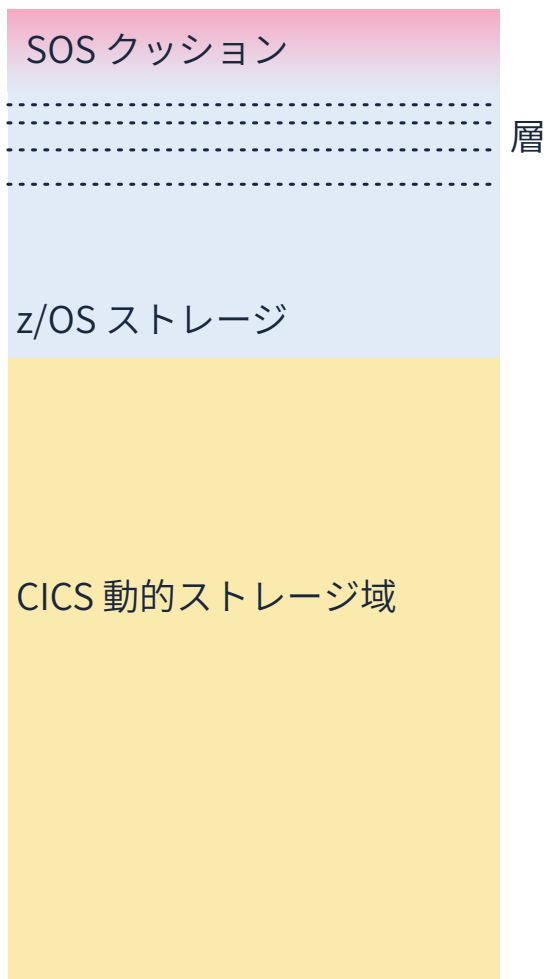


図 18. CICS のストレージ使用法を例示する図

この問題が生じる可能性を減らすために、24 ビットおよび 31 ビットの MVS ストレージで領域が SOS 状態になったかを判別するために CICS モニター・システム・タスクによって使用される SOS しきい値を定義できます。CICS は、CICS アドレス・スペースで使用可能な未割り振りの 24 ビットおよび 31 ビットの MVS ストレージの状態を定期的にモニターし、デフォルトで、未割り振り MVS ストレージ内の変動、および 24 ビットおよび 31 ビットのストレージで検出された SOS イベントを通知するコンソール・メッセージを発行します。

これらのコンソール・メッセージ、および ストレージ・マネージャー: グローバル統計に含まれている 24 ビットおよび 31 ビットの MVS ストレージ統計は、24 ビットおよび 31 ビットの MVS ストレージにおける潜在的なストレージ問題からの保護に役立ち、SOS イベントを防止するためのアクションを柔軟に実行できるようにします。

仕組み: CICS MVS ストレージ・モニター

CICS は、CICS アドレス・スペースで使用可能な未割り振りユーザー領域 (24 ビット) および拡張ユーザー領域 (31 ビット) の MVS ストレージの状態を定期的にモニターします。

モニター・システム・タスクは、24 ビット・ストレージおよび 31 ビット・ストレージの SOS しきい値を使用して、CICS が SOS 状態にあるかどうかを判別します。しきい値は、未割り振りストレージの合計量と、24 ビット・ストレージおよび 31 ビット・ストレージ内で使用可能な最大未割り振り連続ストレージの量です。未割り振りの 24 ビットまたは 31 ビットのストレージがいずれかのしきい値以下の場合、それは SOS と見なされます。デフォルトのしきい値、およびユーザー定義のしきい値の指定方法については、130 ページの『CICS MVS ストレージ・モニターのセットアップ』を参照してください。

モニター・システム・タスクは、アクティブまたはパッシブの 2 つのモードのいずれかで実行できます。モニター・タスクがアクティブ・モードの場合、しきい値の違反が発生すると、CICS は 24 ビット・ストレージまたは 31 ビット・ストレージでの SOS 状態について警告する次のいずれかのメッセージを発行し、

24 ビットまたは 31 ビットのストレージしきい値に違反しなくなるまで SOS メッセージが定期的に発行されます。

DFHSM0144W The CICS region is short on 24-bit MVS unallocated storage.

DFHSM0149W The CICS region is short on 31-bit MVS unallocated storage.

層のモニター

モニター・タスクは、24 ビットおよび 31 ビットの MVS ストレージを複数の層に分割します。各層は前の層より小さく、残りの未割り振り MVS ストレージが SOS しきい値に近づくほど小さくなります。以下の図で例を示します。

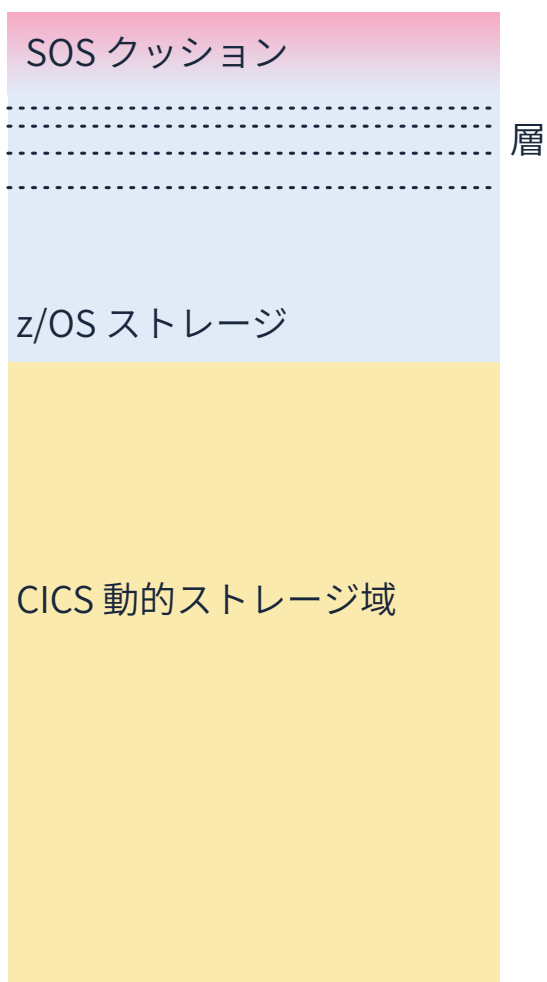


図 19. CICS のストレージ使用法を例示する図

層レベルが大幅に変更されると、モニター・タスクは、アクティブ・モードの場合に、現在の未割り振りストレージのサイズを示すコンソール・メッセージを発行します。

DFHSM0148I メッセージは、24 ビット MVS ストレージの値を示します。

```
DFHSM0148I IYK2ZDL1 MVS 24-bit unallocated storage: Total 4,660K, Largest contiguous area 4,528K.
```

図 20. サンプルの DFHSM0148I メッセージ

DFHSM0153I メッセージは、31 ビット MVS ストレージの値を示します。


```
DFHSM0153I IYK2ZDL1 MVS 31-bit unallocated storage: Total 1,254,176K, Largest contiguous area 1,254,112K.
```

図 21. サンプルの DFHSM0153I メッセージ

これらの情報メッセージは、システムの初期設定時、および層レベルが大幅に変更されたときに、残っている未割り振りストレージの量を示すために発行されます。CICS が SOS 状態になるときも、SOS メッセージに対する補足情報として発行されます。

モニター・サンプリング間隔

未割り振りストレージが SOS しきい値に近づくと、CICS は、ストレージ問題をタイムリーに検出して報告するためにより短い間隔が必要であると判断した場合、モニター間隔を短縮します。未割り振りストレージが SOS しきい値から遠ざかると、CICS は、モニター間隔をデフォルト値またはユーザー指定の初期値まで増やすことがあります。この状態では、より短い間隔でのモニターは必要ないためです。

ストレージ保護の制限と範囲

24 ビットおよび 31 ビットの MVS ストレージをモニターすることにより、ある程度の保護が提供され、潜在的な問題に関する警告が出されることがあります。ただし、24 ビットおよび 31 ビットのストレージの問題をすべて防止できるとは期待しないでください。このモニター機能が提供できる保護には制約があることに注意してください。例えば、モニター・タスクがアクティブ・モードの場合、ストレージのモニター・タスク・サンプリング間隔の間に 24 ビットおよび 31 ビットのストレージが大量に消費されて CICS アドレス・スペースが SOS になると、警告は次のモニター間隔まで出されません。

仕組み: SOS 待機機能

CICS タスクが新しいオープン TCB を必要とする場合、この CICS タスクの処理は、CICS アドレス・スペースで未割り振り MVS ストレージが使用可能かどうかの影響を受けます。ユーザー領域ストレージまたは拡張ユーザー領域ストレージでストレージ不足 (SOS) イベントが発生した場合、デフォルトで、CICS は新しいオープン TCB が必要な場合にタスクを中断します。CICS 管理ストレージのみを使用するタスク (ほとんどのタスク) は影響を受けません。

SOS 待機機能は、デフォルトで有効になっていて、フィーチャー切り替えによって制御されます。[131 ページの『CICS SOS 待機機能のセットアップ』](#)を参照してください。

新しいオープン TCB を必要とするタスクを CICS が処理する方法

CICS タスクが新しいオープン TCB を取得する必要がある場合、CICS はまず CICS アドレス・スペースで使用可能な未割り振り MVS ストレージを検査し、1 つ以上の SOS しきい値に違反しているかどうかを判断します。SOS しきい値は、モニター・システム・タスクで使用されるものと同じしきい値です。SOS しきい値について詳しくは、[125 ページの『CICS の MVS ストレージのモニター』](#)および [130 ページの『CICS MVS ストレージ・モニターのセットアップ』](#)を参照してください。

ユーザー領域または拡張ユーザー領域でストレージ不足が生じた場合、CICS はユーザー・タスクを中断します。中断状態のタスクは、MVS_Stor の待ち状態となり、SOS ウェイター・キューに追加されます。SOS 状態が解決されると、タスクを続行できます。中断状態のタスクはパージ可能になり、DTIMEOUT を適用できるようになります。

影響を受けるユーザー・タスクのパフォーマンス・クラス・モニター・フィールド SMMVSSWT にゼロ以外の値が設定されています。

CICS が SOS ウェイター・キュー内の中断状態のタスクを処理する方法

CICS は、SOS ウェイター・キュー内の最後のタスクを定期的に再開して、ユーザー領域や拡張ユーザー領域がまだストレージ不足かどうかを判別します。SOS 状態が解決された場合、タスクは SOS ウェイター・キューから除去されて、続行を許可されます。SOS 状態が続いている場合、タスクは再び中断されます。

このプロセスは、SOS ウェイター・キュー内に残っているタスクがなくなるまで続きます。

中断されたタスクは、パージされた場合や DTIMEOUT のためにタイムアウトになった場合も、SOS ウェイター・キューから除去されます。

ユーザー領域または拡張ユーザー領域がストレージ不足であるために最初のタスクがウェイター・キューに追加されると、モニター・システム・タスクによってまだ発行されていない場合、SOS メッセージが発行されます。

考慮事項

ユーザー領域または拡張ユーザー領域ストレージで SOS イベントが発生したためにタスクを中断すると、CICS アドレス・スペースで MVS ストレージが不足することを回避できます。ただし、この方法では問題が発生する可能性もあります。例えば、中断されるタスクがエンキューを保持している場合は、他のタスクが完了できなくなり、作業がキューに入れられます。

この動作が望ましくない場合は、[131 ページの『CICS SOS 待機機能のセットアップ』](#)の指示に従って SOS 待機機能を無効にすることができます。

MVS 未割り振りストレージで SOS 状態に対応する方法

DFHSM0144W または DFHSM0149W メッセージを受け取ると、ユーザー領域 (24 ビット) ストレージまたは拡張ユーザー領域 (31 ビット) ストレージでのストレージ不足 (SOS) 状態を解決するためのアクションを行う必要があります。

以下のいずれかの初期アクションを実行することを検討してください。

- 影響を受ける領域を避けて作業をルーティングします。
- CICS ポリシーまたは自動化を作成して、z/OS WLM Health 値を 0 に設定します。
- 待機中のタスクを取り消すか、タスクがタイムアウトになるのを待ちます。
- 影響を受ける領域で実行できる作業量を減らします。
- 影響を受ける領域でオープン TCB の数を減らします。
- JVM サーバーをシャットダウンします。

SOS イベントが解決されると、以下のメッセージが発行されます。

DFHSM0145I The CICS region is no longer short on 24-bit MVS unallocated storage.

DFHSM0150I The CICS region is no longer short on 31-bit MVS unallocated storage.

ベスト・プラクティス

デフォルトの SOS しきい値は、24 ビット MVS ストレージが主に TCB に使用される場合、ほとんどのユーザーに有効となるはずです。

最初は、SOS 状態が検出されてもタスクが待機しないように、SOS 待機機能を無効にしておくことができます。そして単にデフォルトのしきい値でモニターし、24 ビットまたは 31 ビットのストレージの問題が存在する可能性があるかどうかを判別します。

ストレージの問題がない場合は、SOS しきい値を十分に低い値に設定し、SOS 待機機能を再び有効にしてみてください。このとき、通常は、CICS が SOS になる可能性が低いのでタスクは中断されません。それでも SOS イベントが発生した場合は、CICS が反応してタスクを中断し、24 ビットまたは 31 ビットのストレージがさらに消費されないようにします。

SOS 待機機能を無効または有効にする方法について詳しくは、[131 ページの『CICS SOS 待機機能のセットアップ』](#)を参照してください。

起こりうる SOS 状態を回避するために、24 ビットおよび 31 ビットのストレージ統計とコンソール・メッセージを継続的にモニターする必要があります。CICS 領域がいずれかの SOS しきい値に近づいていることが統計に示されている場合は、CICS アドレス・スペースのストレージ不足を回避する手段を実行してください。

MVS
Storage

```
-----
region                                User region      Extended user

Last monitor sample time.....: 02/05/2020  16:39:53      02/05/2020
16:39:53
State.....: NORMAL
NORMAL
Current unallocated total.....: 832K
6680K
LWM unallocated total.....: 28K
6680K
Current unallocated largest contiguous area....: 804K
6260K
LWM unallocated largest contiguous area.....: 28K
6260K
Last date and time SOS.....: 02/05/2020
16:27:57
Current tasks waiting because SOS.....:
0 0
Peak tasks waiting because SOS.....:
1 0
Total waits because SOS.....:
2 0
Time tasks waited because SOS.....: 00:03:08.5650
00:00:00.0000
```

図 22. MVS ストレージに関するサンプル DFHSTUP レポート

CICS 構成に応じて、ストレージの問題を防ぐための多数のオプションがあります。例えば、CICS アドレス・スペースで使用可能な 24 ビットまたは 31 ビットのストレージの量を増やすことができます。このオプションが可能でない場合は、クローン領域を使用して、影響を受ける CICS 領域の一部のワークロードをそこに転送することを検討してください。

CICS MVS ストレージ・モニターのセットアップ

MVS 未割り振りストレージでストレージ不足 (SOS) 状態が発生する可能性を減らすために、ユーザー領域 (24 ビット) ストレージおよび拡張ユーザー領域 (31 ビット) ストレージで CICS が SOS になったかを判定するための SOS しきい値を定義できます。

このタスクについて

フィーチャー切り替えを構成することにより、SOS しきい値およびモニター・サンプリング間隔を指定できます。CICS フィーチャー切り替えについて詳しくは、[機能切り替えの指定](#)を参照してください。

手順

- フィーチャー切り替えを指定して、**モニター・サンプリング間隔を設定します。**

```
com.ibm.cics.mvssm.mon.interval={0|60,1-60}
```

0

モニター・システム・タスクをパッシブ・モードに設定します。メッセージは発行されません。

number

1 から 60 の範囲の秒数を指定します。デフォルトは 60 です。モニター・システム・タスクはアクティブ・モードです。

注：

未割り振りストレージが SOS しきい値に近づくと、CICS は、ストレージ問題をタイムリーに検出して報告するためにより短い間隔が必要であると判断した場合、モニター間隔を短縮します。未割り振りストレージが SOS しきい値から遠ざかると、CICS は、モニター間隔をデフォルト値またはユーザー指定の初期値まで増やすことができます。この状態では、より短い間隔でのモニターは必要ないためです。

- SOS しきい値を設定します。**

未割り振り MVS ストレージには、4 つの SOS しきい値があります。

デフォルトの SOS しきい値は、MVS 24 ビット・ストレージが主に TCB に使用される場合、ほとんどのユーザーに有効となるはずです。

無条件 GETMAIN を発行するアプリケーションまたはベンダー製品がある場合は、これらの設定のサイズを増やすことができます。無条件 GETMAIN について調べることは困難です。ただし、CICS は SOS メッセージを発行し、実行中のシステムでの MVS ストレージの使用状況を示す統計を提供します。ある期間におけるストレージ使用量の変動を調べることで、発行される GETMAIN の最大サイズを見積もることができます。ストレージ不足のリスクを減らすには、これらの GETMAIN を受け入れるように SOS しきい値を調整します。

関連するフィーチャー切り替えを設定することで、デフォルト以外の値を設定できます。すべての値は Kb 単位となります。無効な値が指定された場合、代わりにデフォルトが使用されます。

注：

- 以下の情報は、モニター・システム・タスクがアクティブ・モードであることを前提としています。
- 未割り振り MVS ストレージは 4K ページ単位で割り振られるため、4 未満のしきい値を指定すると、使用可能な未割り振りストレージがない場合にのみ、CICS はそのしきい値を違反したと見なします。

未割り振り 24 ビット MVS ストレージの最小合計量

このしきい値を指定するには、以下のフィーチャー切り替えを使用します。

```
com.ibm.cics.mvssm.sos24.minavailable.total={64,1-1024}
```

未割り振り 24 ビット・ストレージの合計量がこの値以下の場合、CICS では 24 ビット・ストレージが不足していると思われて、DFHSM0144W が発行されます。

注：指定された最小合計量が最小連続ストレージ値より小さい場合、その最小連続ストレージ値が、最小合計量しきい値として使用されます。

未割り振り 24 ビット MVS ストレージで使用可能な最小連続ストレージ

このしきい値を指定するには、以下のフィーチャー切り替えを使用します。

```
com.ibm.cics.mvssm.sos24.minavailable.contiguous={32,1-1024}
```

最大の連続未割り振り 24 ビット・ストレージ域がこの値以下の場合、CICS では 24 ビット・ストレージが不足していると思われて、DFHSM0144W が発行されます。

未割り振り 31 ビット MVS ストレージの最小合計量

このしきい値を指定するには、以下のフィーチャー切り替えを使用します。

```
com.ibm.cics.mvssm.sos31.minavailable.total={128,1-16384}
```

未割り振り 31 ビット・ストレージの合計量がこの値以下の場合、CICS では 31 ビット・ストレージが不足していると思われて、DFHSM0149W が発行されます。

注：指定された最小合計量が最小連続ストレージ値より小さい場合、その最小連続ストレージ値が、最小合計量しきい値として使用されます。

未割り振り 31 ビット MVS ストレージで使用可能な最小連続ストレージ

このしきい値を指定するには、以下のフィーチャー切り替えを使用します。

```
com.ibm.cics.mvssm.sos31.minavailable.contiguous={64,1-16384}
```

最大の連続未割り振り 31 ビット・ストレージ域がこの値以下の場合、CICS では 31 ビット・ストレージが不足していると思われて、DFHSM0149W が発行されます。

CICS SOS 待機機能のセットアップ

CICS SOS 待機機能は、デフォルトで有効になっています。これにより、ユーザー・タスクが新しいオープン TCB を必要とするとき、ユーザー領域 (24 ビット) または拡張ユーザー領域 (31 ビット) のストレージで CICS がストレージ不足 (SOS) 状態になると、ユーザー・タスクは待機します。これらの TCB を接続すると、24 ビットまたは 31 ビットのストレージ要件が増える可能性があります。

SOS イベント中にタスクを待機することが、CICS アドレス・スペースが SOS になることよりも望ましくない場合は、以下のフィーチャー切り替えを設定して、この機能を無効にすることができます。

```
com.ibm.cics.mvssm.sos.wait=false
```

このフィーチャー切り替えが有効な場合、CICS は 24 ビット・ストレージおよび 31 ビット・ストレージに問題がないと見なすので、タスクは中断されません。

この機能を再度有効にする場合は、true を指定するか、デフォルトを使用します。

CICS フィーチャー切り替えについて詳しくは、[機能切り替えの指定](#)を参照してください。

オンライン・システムの分割: 仮想記憶域

CICS システムが使用可能な仮想記憶を増やすには、システムを複数の個別のアドレス・スペースに分割することができます。システムを分割すると高可用性も提供され、システムを各プロセッサで並行して稼働できるようになるため、マルチプロセッサの複合システムを最大限に活用できます。ほとんどの CICS システムは分割できます。

より多くの仮想記憶域を取得するように CICS を調整するには、MVS を調整してから、CICS を調整します。MVS の共通仮想記憶域を調整したあとも、単一アドレス・スペース内で CICS を実行できない場合は、CICS ワークロードをマルチアドレス・スペースに分割することを検討する必要があります。新規のアドレス・スペースは、追加の実記憶を必要としますが、CICS 領域の分割によって得られる仮想記憶の節約がかなり大きくなる可能性があります。CICS システムの分割は、アプリケーションの機能別、CICS の機能別 (ファイル所有領域や端末所有領域など)、または 2 つの機能の組み合わせによって行えます。

通常のインストールでは、CICS ワークロードを複数の独立アドレス・スペースに分割すると、ワークロードを容易に定義でき、リソース共有が不要になるため、便利です。アプリケーション・サブシステムとそれに関連する端末、プログラム、およびデータ・セットを容易に分離できる場合は、単一の CICS アドレス・スペースを複数の独立したアドレス・スペースに分割するのが合理的です。これらのスペースは、相互作用のない自律型領域です。

CICS システムを完全に分割し、2 つの部分間に通信が必要ないようにできる場合、オーバーヘッドおよび計画が削減されます。新しいシステムでデータ、プログラム、または端末を共用する必要がある場合は、CICS 相互通信を使用できます。IPIC (IP 相互接続) 接続、SNA 経由 ISC (SNA 経由のシステム間連絡) 接続、または MRO (複数領域操作) を使用して、CICS 領域を相互に接続できます。CICS 相互通信方式および各方式で使用可能な機能 (トランザクション・ルーティングや機能シップなど) についての説明は、[CICS 相互通信の紹介](#)を参照してください。

また、CICS 領域の追加コピーを作成し、CICS 相互通信を使用して、領域間のトランザクション・ルーティングを提供することも検討できます。別の仮想記憶域が必要な場合は、AOR を複数の別の CICS コピーに分割するなどの方法が合理的です。システムを部分的に、または完全に分割すると、未使用の常駐プログラムを削除して、各領域に必要な仮想記憶域のサイズを小さくすることができます。未使用のプログラムを削除すると、関連の DSA のサイズが削減されます。

CICS 相互通信は追加のプロセッサ・サイクルを使用し、それが応答時間およびプロセッサ時間に影響を与えることがあります。相互通信のコストは、接続タイプ (IPIC、MRO、または SNA 経由 ISC)、およびその接続を介して使用する相互通信機能によって異なります。さまざまな相互通信方式および機能に関するパフォーマンスの考慮事項については、[154 ページの『CICS MRO、ISC、および IPIC: パフォーマンスおよび調整』](#)を参照してください。

CICS システムを分割する場合は、特定のパラメーター (MXT など) の調整が必要な場合があります。機能シップを使用する MRO システムでは、所要時間の長いタスクは、MXT およびその他のパラメーター (例えば、ファイル・ストリング番号、仮想記憶割り振り) もさらに調整が必要な場合があります。

MRO の使用を計画している場合は、MVS リンク・バック域 (LPA) を使用して、CICS コードまたはアプリケーション・コードを共用することを検討してください。LPA は、実記憶 (仮想記憶ではなく) およびその他の非 CICS アドレス・スペースの節約になることに注意してください。CICS 内の適格モジュールに LPA を使用するかどうかは、システム初期設定パラメーター (LPA=YES) で制御されます。このパラメーターは、CICS に LPA 内のモジュールを検索するように指示します。LPA の使用について詳しくは、[133 ページの『リンク・バック域 \(LPA/ELPA\) でのモジュールの使用』](#)を参照してください。

リンク・パック域 (LPA/ELPA) でのモジュールの使用

一部の CICS 管理およびユーザー・モジュールは、リンク・パック域 (LPA) または拡張リンク・パック域 (ELPA) に移動できます。CICS のコピーが複数稼働しているシステムでは、この移動により、複数のコピーで同じ CICS 管理コード・セットを共用できます。

LPA または ELPA にコードを配置すると、次のようないくつかの利点があります。

- ユーザー・アプリケーションによってコードが破壊されないように保護されます。LPA または ELPA は保護ストレージであるため、これらのプログラムの内容を変更することはほとんど不可能です。
- プログラム・モジュールに LPA または ELPA を使用すると、パフォーマンスを改善し、実記憶域に対する要求を軽減することができます。同じプロセッサの複数のアドレス・スペースで、同じリリースの CICS コピーが複数稼働している場合、アドレス・スペースごとに CICS 中核モジュールにアクセスする必要があります。これらのモジュールは各アドレス・スペースにロードするか、または LPA や ELPA で共用することができます。LPA または ELPA でモジュールを共用すると、作業セットを削減できるため、実記憶域 (ページング) に対する要求も軽減されます。
- プライベート域のストレージ要件を軽減するには、丸めによって作成された LPA または ELPA 内の未使用ストレージを次のセグメントに慎重に割り振ります。

モジュールを LPA または ELPA に配置するには、オペレーティング・システムの IPL が必要です。メンテナンス要件も考慮する必要があります。テスト・システムおよび実動システムが LPA モジュールまたは ELPA モジュールを共用している場合は、新規メンテナンスをテストするときに、LPA または ELPA モジュールを取り外してテスト・システムを実行しても構いません。

LPA に多くのモジュールを配置しすぎると、サイズが極端に大きくなるという欠点が生じることがあります (この欠点は ELPA では生じません)。CSA とプライベート域間の境界はセグメント境界上にあるため、境界が 1MB 下方に移動することがあります。ELPA のサイズは通常問題となりません。

LPA に使用するモジュールを選択するには、LPAUMOD という名前の SMP/E USERMOD を使用します。これにより、LPA または ELPA に適格なモジュールが示されます。この USERMOD を使用すると、ご使用の LPA ライブラリーにモジュールを移動できます。複数の CICS アドレス・スペースを持つすべてのユーザーは、すべての適格モジュールを ELPA に配置する必要があります。

LPA=YES はシステム初期設定テーブル (SIT) 内で指定する必要があります。LPA=NO を指定すると、新規バージョンの CICS プログラム (新規リリースなど) を使用してシステムをテストしてから、コードを実動システムに移動することができます。その後は、新規バージョンのテスト中も、実動システムは LPA 内のモジュールを引き続き使用できます。

その他の制御 (PRVMOD システム初期設定パラメーター) を使用すると、特定のモジュールを LPA で使用しないように、明示的に除外することができます。

LPA でのモジュールのインストールについては、[MVS リンク・パック域での CICS モジュールのインストール](#)を参照してください。

位置合わせマップまたは位置合わせなしマップの選択

基本マッピング・サポート (BMS) で使用される CICS マップは、位置合わせ、または位置合わせなしとして定義できます。位置合わせマップでは、BMS DSECT 内の BMS データ・フィールドに関連する長さフィールドは、常にハーフワード境界に対して位置が合わせられます。位置合わせなしマップでは、長さフィールドはマップ DSECT 内の直前のデータ・フィールドの直後に配置されます。位置合わせマップは AMAP オプションでコンパイルされ、位置合わせなしマップは、MAP オプションでコンパイルされます。位置合わせマップと位置合わせなしマップは、組み合わせて使用することができます。

位置合わせなしマップでは、BMS DSECT の長さフィールドがハーフワードに対して位置合わせされる保証はありません。この場合、一部の COBOL および PL/I コンパイラーがプログラム内に余分なコードを生成し、このような長さフィールドの内容が参照または変更されると、その内容をハーフワードに対して位置合わせされた作業域との間でコピーします。

マップの位置合わせを指定すると、BMS DSECT のサイズが増加し (最悪の場合、マップ・データ・フィールドごとに 1 バイトが埋め込まれ)、マップ処理中の BMS の内部パス長がわずかに増加します。したがって、最適な方法は、使用中のコンパイラーが非効率なアプリケーション・プログラム・コードを生成する場合を除き、位置合わせなしマップを使用することです。

COBOL では、位置合わせなしマップは非同期構造を生成します。PL/I では、位置合わせなしマップはマップ DSECT 定義を位置合わせなし構造として生成します。対照的に、位置合わせマップは、COBOL では同期化構造を、PL/I では位置合わせ構造を生成します。

CICS では、BMS マップは常にグループ (マップ・セット) 内に生成されます。マップ・セット全体を位置合わせまたは位置合わせなしとして定義する必要があります。また、各言語で記述されたアプリケーション・プログラムでマップを使用することもできます。これらの場合は、プログラムの組み合わせに最適なオプションを選択し、位置合わせマップと位置合わせなしマップの両方に対する要件がある場合は、ALIGNED オプションを選択します。

マップ DSECT を変更すると、参照元のすべてのアプリケーション・プログラムの再アセンブリーや再コンパイルも必要になるため、例えば、位置合わせから位置合わせなしに、マップを変換することは避けてください。

マップ位置合わせは、マップをアセンブルするときに定義されます。位置合わせマップは SYSPARM(A) オプションを使用します。BMS=ALIGN/UNALIGN システム初期設定パラメーターは、使用中のマップ・タイプを定義します。

マップおよびマップ・セットが画面定義機能 (SDF II) ライセンス・プログラムを使用して定義されている場合は、マップおよびマップ・セットの位置合わせオプションも指定できます。

マップ位置合わせの重要性は、多数のフィールドを含む画面を処理するプログラムを調べるとわかります。まず、マップ位置合わせオプションを指定しないで BMS DSECT を生成した場合、次にこのオプションを指定して BMS DSECT を生成した場合で、プログラムを再コンパイルしてください。リンケージ編集マップで指定されたプログラム・サイズが、オプションを指定した場合に大幅に縮小した場合は、できるかぎり位置合わせマップを使用してください。

常駐、非常駐、または一時としてのプログラムの定義

プログラム、マップ・セット、および区画セットは RESIDENT(NO|YES) および USAGE(NORMAL|TRANSIENT) として定義できます。プログラムは RELOAD(NO|YES) として定義できます。

CSD で定義されたすべてのプログラムは、最初の使用時に CDSA、RDSA、SDSA、ECDSA、ERDSA、または ESDSA にロードされます。RELOAD(YES) プログラムを共用したり、再使用することはできません。RELOAD(YES) が定義されたプログラムは、明示的な EXEC CICS FREEMAIN の後にのみ、削除されます。USAGE(TRANSIENT) プログラムは共用できますが、使用回数がゼロになると削除されます。RESIDENT(NO) プログラムは、使用回数がゼロになると削除に対して適格になります。DSA ストレージが制約されてくるにつれて、CICS ローダー・ドメインは、頻繁に使用されないプログラムを最初に削除して、これらのプログラムを徐々に削除します。

RESIDENT(YES) プログラムは通常削除されません。NEWCOPY を任意のプログラムに対して実行すると、次の参照時に新規コピーがロードおよび使用され、古いコピーの使用回数がゼロになった場合に、古いコピーが削除に対して適格になります。

CICS のウォーム・スタート時に、各常駐プログラム・サブプールの最初のフリー域が割り振られます。この領域のサイズは、直前の CICS シャットダウン中に記録された、現在ロード中のすべての常駐プログラムの合計長に基づいて決まります。常駐プログラムがロードされると、CICS はこのプログラムを初期フリー域に適合させようとします。このプログラムが適合しない場合は、初期フリー域の外部にロードされ、初期フリー域内のスペースは、他の (より小さな) 常駐プログラムがロードされないかぎり割り振り解除されたまま残ります。この状態は、常駐プログラムが直前のロード以降 (直前の CICS シャットダウンまで) にサイズが増加した場合に発生します。問題のプログラムが大きい場合は、初期フリー域の大量の未使用ストレージが常駐プログラム用に割り振られるため、ストレージ問題が発生することがあります。

使用中でないプログラムは最低使用頻度 (LRU) に基づいて削除されるため、特定のプログラムを永続的に常駐化して優先的に処理する特定の理由が存在しないかぎり、これらのプログラムは RESIDENT(NO) として定義します。LRU アルゴリズムは、時間経過によるプログラム使用量の変動を自動的に考慮します。

したがって、使用頻度の高い非常駐のプログラムは常駐化される可能性が大きくなる一方で、使用頻度が少ない期間中、常駐プログラムは永続的に占有する仮想記憶域を浪費する可能性があります。

16 MB 境界より上で実行するように記述されたプログラムでは、仮想記憶域が制約を受けない十分な大きさの EDSALIM を指定します。

プログラムが大きい、または頻繁に更新されてサイズが大きくなる場合は、このプログラムを非常駐として定義し、PLTPI 処理中に HOLD オプションを指定して LOAD を発行することを検討してください。

次のいずれかの理由で、プログラムを RESIDENT として定義することができます。

- ストレージのフラグメント化を避けるため。このようなプログラムはすべてストレージの 1 つのブロックに入れられるからです (ただし、プログラムの新規コピーは除きます)。
- 潜在的に危険な状態 (例えば、CEMT) を処理するプログラムの場合。
- DFHRPL または動的プログラム LIBRARY に重度の競合がある場合。ただし、通常は、データ・セット配置または他の DASD 調整、あるいは MVS ライブラリー・ルックアサイド機能を使って MVS データ・スペース内にプログラム・コピーを維持することにより、競合を処理します。

16 MB より上へのアプリケーション・プログラムの配置

CICS は RMODE(ANY) アプリケーション・プログラムを EDSA に保持します。EDSA は、16 MB より上、ただし 2 GB より下 (16 MB 境界より上) の MVS 拡張仮想記憶域内にあります。プログラムに関連する作業域は、境界より上に常駐することもあります。

64 ビット、31 ビット、および 24 ビットのアドレッシング・モード (AMODE) プログラムの間で LINK または XCTL を実行することができます。プログラムを 31 ビットまたは 64 ビット・アドレッシング・モード・プログラムに変換して、16 MB より上、ただし 2 GB より下の拡張専用域に移動することができます。プログラムを 16 MB より上、ただし 2 GB より下に移動すると、16 MB より下の仮想記憶域がその分だけ解放され、別の目的に使用できるようになります。

LPA または拡張リンク・パック域 (ELPA) のプログラムを使用する方法については、[133 ページの『リンク・パック域 \(LPA/ELPA\) でのモジュールの使用』](#)を参照してください。

マルチアドレス・スペースが採用されている場合は、CICS が必要とするプログラムは既にロードされていて、実記憶域使用量は最小化されているため、拡張プライベート域よりも ELPA を使用の方が便利です。

トランザクション分離が使用可能な状態で CICS システムを実行した場合は、トランザクションおよびアプリケーション・プログラムを 16 MB 境界より上に移動することによりパフォーマンスを改善できます。この場合、プログラム作業域は、4 KB のページ・サイズを持つ UDSA からでなく、1 MB のページ・サイズを持つ EUDSA から取得されます。この機能は、16 MB 境界までの仮想記憶域が必要で、かつ十分な実記憶域が存在する場合に便利です。16 MB 境界より上の仮想記憶域を使用する目的は、16 KB より下のスペースを他の目的に使用できるようにすることにあるため、プログラムが 16 MB より上、ただし 2 GB より下に移動された場合は、実記憶域に対する要求が全体的に増加します。

COMMAREA が、別々のアドレッシング・モードで実行されているプログラムの中で受け渡しされる場合、以下の制限が適用されます。

- AMODE(31) プログラムから AMODE(24) プログラムに渡される COMMAREA は、AMODE(24) プログラムで処理できる必要があるため、31 ビット・アドレスを含んでいてはなりません。
- AMODE(64) プログラムから AMODE(31) プログラムに渡される COMMAREA は、AMODE(31) プログラムで処理できる必要があるため、64 ビット・アドレスを含んでいてはなりません。
- AMODE(64) プログラムから AMODE(24) プログラムに渡される COMMAREA は、AMODE(24) プログラムで処理できる必要があるため、64 ビットまたは 31 ビット・アドレスを含んでいてはなりません。

16 MB 境界より上に常駐するプログラムは、リンク・エディットの MODE ステートメントの AMODE(31),RMODE(ANY) オプションを使用してリンク・エディットを行う必要があります。

トランザクション分離を使用する場合の実記憶域の割り振り

トランザクション分離がアクティブの場合、実記憶域に関するコストが発生します。十分な実記憶域が割り振られていない場合、ページング問題が発生する可能性があり、それがパフォーマンスに影響を与えることがあります。コストは、システムで使用中のサブスペースの数、および **EDSALIM** パラメーターのサイズによって異なります。

EUDSA のページ・サイズは 1 MB なので、トランザクション分離がアクティブの CICS システムの場合、**EDSALIM** の値が非常に大きくなる可能性があります。この仮想記憶域には実記憶域を使用するページおよびセグメント・テーブルをマップする必要があるため、実記憶域の使用量が增大することがあります。

実記憶域は **EDSALIM** 値に仮想記憶域をマップするために使用されるだけでなく、サブスペースでも必要となります。例を以下に示します。

- 各サブスペースには 2.5 ページが必要で、1 ページは 4 KB の実記憶域を意味します。
- システム内のトランザクションごとに固有のサブスペースが必要な場合 (トランザクション定義 TASKDATAKEY(USER) および ISOLATE(YES))、必要な実記憶域は **MXT** 値 x 2.5 ページになります。
- システム内のトランザクションごとに EUDSA のストレージが 1 ページ必要な場合は (1 MB ページ)、ストレージをマップするために 1 つのページ・テーブルが必要です。実記憶域は **MXT** 値 x 1 ページです。
- さらに 3 ページが必要です。したがって、実記憶域の合計は、**MXT** 値 x (1 + 2.5 ページ) + 3 ページとなります。
- この実記憶域はすべて、ELSQA から割り振られます。

実記憶域の使用量に関する値は、トランザクション分離がアクティブでない CICS システムに必要な値よりも大きくなります。CICS の実記憶域の所要量は、常にトランザクション負荷によって変動します。ガイドラインとして、システム内の各タスクには 9 KB の実記憶域が必要です。この数値に、一度にシステム内に存在できる同時タスクの数 (**MXT** システム初期設定パラメーターで設定) を乗算します。

SNA ペーシングを使用したサブプール 229 の拡張の制限

バッチ・タイプ端末が CICS トランザクションのデータ処理速度よりも高速にデータを送信する場合は、サブプール 229 を拡張できます。2 次側から 1 次側へのペーシング (別名インバウンド・ペーシング) を使用すると、所定のバッチ端末のサブプール 229 で待機できるデータ・サイズが制限されます。PACING パラメーターは Network Control Program (NCP) から端末へのトラフィック・フローを制御し、プロセッサ・アクティビティーには影響しません。VPACING パラメーターは、ホストと NCP 間のトラフィック・フローを制御します。

CICS APPL ステートメントの VPACING パラメーターは、1 つのセッション内で z/OS Communications Server アプリケーション・プログラムに対して、肯定応答 (ペーシング応答) を戻す必要なしに別の SNA 論理装置が送信できるメッセージの数を決定します。ホストは VPACING パラメーターの定義に従って、データ・パス情報単位 (PIU) を送信します。グループ内の最初の PIU は RH 内のペーシング標識を伝達します。この PIU が NCP で処理されると、NCP は同じペーシング標識を持つホストに応答を送信して、新規ペーシング・グループを要求します。つまり、端末への x 個の PIU ごとに、およびプリンターへの y 個の PIU ごとに、NCP からホストにペーシング応答トラフィックが流れる必要があるため、トラフィック・ボリュームに基づいて、ホスト・アクティビティーが大幅に増加することがあります。

通常、VPACING パラメーターは NCP バッファ不足によってホストと NCP 間のフロー・ボリュームを制御する必要が生じた場合にインプリメントされます。プロセッサに対する影響を小さくするには、VPACING パラメーターを NCP の許容値まで増加させます。

ほとんどのプリンターでは、バッファ容量を受信データの印刷速度と一致させるために、PACING パラメーターが必要です。一部のグラフィックス・アプリケーションの場合のように、1 つの LU への膨大なデータ送信を制限する必要がないかぎり、端末は通常ペーシングを必要としません。端末にペーシングを使用すると、応答時間が低下します。PACING パラメーターと VPACING パラメーターを併用すると、応答時間が低下し、プロセッサ・アクティビティーが増加して、ネットワーク・トラフィックが増大します。

「ランナウェイ」トランザクションが SNA ネットワークにメッセージをフラッディングして、大量のバッファ・ストレージが必要となる事態を避けるために、すべての端末で PACING および VPACING パラメーターを指定します。端末に SEND コマンドを発行する間にトランザクションがループすると、IOBUF (CSA ストレージ) と NCP バッファが満杯になって処理がスローダウンし、CSA 不足状態になることがあります。

通常、データ・トラフィックは規制されずに流れる一方で、過剰な量のデータがネットワークに流入してデータの通常フローを損なわないようにするために、PACING パラメーターおよび VPACING パラメーターは十分高く指定してください。

2 次側から 1 次側へのペーシングの場合、次の方法でコーディングする必要があります。

- 2 次側アプリケーション・プログラムによって示される LOGMODE 入力内で SSNDPAC=nonzero 値を指定
- 2 次側アプリケーションの APPL 定義で VPACING=nonzero 値を指定

使用される値は、VPACING パラメーターでコード化されます。これらの値のいずれかがゼロである場合、ペーシングは発生しません。

CICS 領域を定義する APPL ステートメントでは VPACING を指定し、バッチ装置を定義する LU ステートメントでは SSNDPAC パラメーターにゼロ以外の任意の値を指定します。装置のコンポーネント記述マニュアルを参照して、装置がこの形式のペーシングをサポートしていることを確認してください。

CICS のストレージ保護機能: パフォーマンスおよび調整

ストレージ保護に関連する機能は、ストレージ保護、トランザクション分離、およびコマンド保護です。これらの機能はユーザー・アプリケーション・コードからストレージを保護します。

ストレージ保護

CICS コード および制御ブロックがユーザー・アプリケーションによって上書きされないよう保護します。

トランザクション分離

トランザクション・データが他のユーザー・トランザクションによって上書きされないように保護します。

コマンド保護

CICS を使用して更新する必要がある EXEC CICS インターフェースを使用してアプリケーション・プログラムがストレージを CICS に渡さないようにします。ただし、アプリケーションはストレージを更新することはできません。

ストレージ保護、トランザクション分離、およびコマンド保護では、ユーザー・アプリケーション・コードからストレージが保護されます。ユーザー・コードが実行されない領域、つまり、純粋な端末専有領域 (TOR) または純粋なファイル専有領域 (FOR) には、利益は追加されません (分散プログラム・リンク (DPL) 要求が機能シップされない場合)。

トランザクション分離とアプリケーション

トランザクション分離を使用する場合は、タスクの割り振りサブスペースに対してストレージのページを活動化する必要があります。ストレージがサブスペースに対して活動化される前に取り出し保護されるため、タスクはそのストレージにアクセスできません。ストレージがタスクに割り振られているサブスペースに対して活動化されると、タスクはこのストレージに対して読み取りおよび書き込みアクセスを持ちます。CICS では、ユーザー・キー・タスク存続時間ストレージの新規ページを取得するためにユーザー・タスクが GETMAIN コマンドを呼び出すごとに、サブスペースに対してユーザー・ストレージを活動化する必要があります。サブスペースに対してストレージを活動化するときにはパフォーマンス・コストの一部が関連するため、アクティビティーを最小限にする必要があります。

16 MB 境界を超えないストレージは、4 KB の倍数単位で活動化されます。この境界を超えるストレージは、1 MB の倍数単位で活動化されます。そのため、完全に 16 MB 境界より上で実行されるユーザー・タスクが必要とする活動化操作は 1 回だけです。

RMODE(ANY) を使用してプログラムをリンク・エディットし、それらを DATALOCATION(ANY) として定義してください。すべてのトランザクションを TASKDATALOC(ANY) として定義することによって、ストレージ・アクティベーション数を削減する必要があります。

16 MB 境界より下のストレージを取得する必要がある場合は、サイズの小さい複数回の GETMAIN 要求ではなく 1 回の GETMAIN 要求ですべてのストレージを取得することによってパフォーマンスを向上することができます。そうすることで、ストレージ活動化操作数を最小限にすることもできます。

詳しくは、[MVS サブスペース](#)を参照してください。

Language Environment での調整

CICS で Language Environment を使用している場合は、いくつかの調整アクションによってパフォーマンスを最適化できます。Language Environment が CICS アドレス・スペースでアクティブになっている場合は、COBOL や PL/I などのネイティブ言語のランタイム・ライブラリーは不要です。つまり、CICS は、すべての言語のランタイムとの間には 1 つのインターフェースしか持っていません。

Language Environment について詳しくは、[プログラミング言語と言語環境](#)を参照してください。

GETMAIN および FREEMAIN アクティビティーの最小化

Language Environment を使用してプログラムを実行する場合のパフォーマンスを改善する 1 つの方法は、Language Environment が使用するストレージの管理に必要な GETMAIN および FREEMAIN 要求の数を少なくすることです。

Language Environment に代わって CICS が実行する GETMAIN および FREEMAIN 要求の数を最小化する場合には、次のシステム初期設定パラメーターを使用することができます。

- AUTODST
- RUWAPPOOL

これら 2 つのオプションは、どのような組み合わせでも一緒に使用できます。

これらの機能を使用する利点を確認するには、機能のいずれかまたは両方がアクティブになっているときに CICS ストレージ・レポートを実行して、領域内の GETMAIN および FREEMAIN 要求の数を表示し、その結果を前の実行結果と比較します。

AUTODST: Language Environment の自動ストレージ調整

CICS システム初期設定パラメーター AUTODST を YES に設定することにより、オプションで、CICS に対して Language Environment の自動ストレージ調整機能を活動化することができます。この機能がアクティブになっている場合は、Language Environment は各メインプログラムの実行をモニターし、プログラムがアクティブになっている間にプログラムに何らかの追加ストレージを割り振る必要があったかどうかを記録します。

各プログラムの実行の最後に、何らかの追加ストレージを割り振る必要があった場合は、Language Environment はこの情報を保存します。次回、プログラムが実行されたときに、Language Environment はこの追加のストレージ分を含めるように初期ストレージ割り振りを増やします。このプロセスは、CICS が実行する必要がある GETMAIN および FREEMAIN 要求の数を最小化するのに役立ちます。

自動ストレージ調整は、多くの動的呼び出しを発行するプログラムにとって特に役立ちます。そのようなプログラムはすぐに、それぞれの初期ストレージ割り振りを超えてしまうことがあるからです。また自動ストレージ調整により、個々の COBOL プログラムに対して手動でストレージを調整する必要もなくなります。

ただし、一度 Language Environment がプログラムの初期ストレージ割り振りを増やしてしまうと、それが減らされることはありません。プログラムを実行するのに、きわめて大量のストレージを必要とする場合（おそらくユーザーがほとんど使用されることのないプログラムの機能を活動化した場合）、このストレージ量が、以降のすべてのプログラム実行に割り振られます。そのため、まれなケースではありますが、自動ストレージ調整が、一部のプログラムに対して過度のストレージ割り振りを行ってしまふことがあります。

Language Environment ストレージ調整のユーザー出口 CEECSTX を使用すると、自動ストレージ調整メカニズムの振る舞いを変更することができます。このユーザー出口を使用すると、特定のプログラムに対して自動ストレージ調整を使用可能または使用不可にすることができます。これは、実行ごとにストレージに対する要求が大幅に変わるアプリケーションの場合に役に立つことがあります。また、自動ストレージ調整では、初期ストレージ割り振りの開始値を指定することができます。この開始値を使用すると、自動ストレージ調整プロセス中に Language Environment が割り振るストレージの最大量を制限することができます。

これまでに CEECSTX ユーザー出口を Language Environment ストレージ調整方式として使用したことがある場合は、ユーザー出口がなくても、自動ストレージ調整メカニズムは同じ機能を提供することに気付かれるかもしれません。主記憶域の調整方式としてどちらのメカニズムを使用するかを決定する必要があります。なぜなら、自動ストレージ調整を使用して CICS を実行している場合、CEECSTX ユーザー出口の機能が制限されるからです。自動ストレージ調整がストレージ割り振りをモニターするのに対し、ストレージ調整のユーザー出口 CEECSTX は、ユーザー・アプリケーション・プログラムが実際に使用するストレージをモニターします。それにもかかわらず、自動ストレージ調整に伴うオーバーヘッドは、CEECSTX 出口に基づく調整方法よりも小さくなります。また、自動ストレージ調整がトランザクションによって呼び出される初期プログラムごとに調整を行うのに対し、CEECSTX 出口は、それが入力として使用するテーブルに含まれるプログラムに対してのみ調整を行います。つまり、自動ストレージ調整の場合は、より多くのプログラムがストレージを使用するほど、そのストレージを調整することによる利点が大きくなります。

CEECSTX について詳しくは、[z/OS Language Environment カスタマイズ](#)を参照してください。

RUWAPPOOL: 実行単位作業域プール

システムのパスの長さは、Language Environment が起動する CICS アプリケーションが **EXEC CICS LINK** 要求を発行するときに増加します。Language Environment が起動した同一プログラムに対して **EXEC CICS LINK** 呼び出しを繰り返し行くと、実行単位作業域 (RUWA) に対する GETMAIN および FREEMAIN 要求が複数出されます。

システム 初期設定パラメーター RUWAPPOOL (YES) を使用すると、タスクの初期設定中に実行単位作業域が作成されます。このプールは、Language Environment が起動するプログラムが必要とする RUWA を割り振るために使用されます。これにより、Language Environment によって起動されるプログラムに対して多くの **EXEC CICS LINKS** を実行するタスク内の GETMAIN および FREEMAIN 要求の数が少なくなります。

RUWAPPOOL システム 初期設定パラメーターについて詳しくは、[RUWAPPOOL](#) を参照してください。

Language Environment の、AMODE(24) プログラム用のランタイム・オプション

CICS のデフォルトの Language Environment ランタイム・オプションは ALL31(ON) および STACK(ANY) です。つまり、Language Environment が使用可能になっている場合、Language Environment を必要とするプログラムはすべて、31 ビットのストレージをアドレッシングできる必要があります (つまり AMODE(31) である必要があります)。

AMODE(24) プログラムを Language Environment 対応の CICS 領域で実行するには、16 MB 境界よりも下で実行する必要のあるプログラムに対して、ALL31(OFF) および STACK(BELOW) を指定する必要があります。ただし、すべてのプログラムがこれらのオプションを使用するようにこれらのオプションをグローバルに変更する場合は、大量のストレージが 16 MB の境界よりも下に割り振られ、これにより、ストレージ不足の状態が発生することがあります。ALL31(OFF) オプションを使用すると、Language Environment は 16 MB 境界よりも上および下の両方に、RUWA などのいくつかの制御ブロックを獲得することになるので、重複する制御ブロックを管理するためには、追加の GETMAIN および FREEMAIN 要求が必要になります。

当該のプログラムがトランザクションによって起動される初期プログラムである限り、ALL31(OFF) を指定する必要はありません。Language Environment は、自動的に正しいアドレッシング・モードでエンクレープ (プログラム) 用のストレージを獲得するからです。例外は AMODE(31) プログラムで、これは動的に AMODE(24) プログラムを呼び出します。この場合、動的に呼び出された AMODE(24) プログラムは ALL31(OFF) を指定する必要があります。

C++ の DLL の使用

各ダイナミック・リンク・ライブラリー (DLL) が最初にロードされたときの初期設定のコストは、その DLL が獲得する書き込み可能な静的領域のサイズによって決定できます。この書き込み可能領域から不要な項目を除去することにより、初期設定コストを削減できます。

DLL を使用している場合は、以下の点を考慮する必要があります。

- `#pragma variable (x,NORENT)` の指定。これにより、テーブルなどのいくつかの読み取り専用変数がコード域に配置されます。
- `#pragma strings (readonly)` の指定。これは、変更可能であることがリテラル・ストリングのデフォルトである C コードに対して機能します。C++ は既に、デフォルトでリテラル・ストリングを読み取り専用として持っています。
- 大規模領域を決定するために、プリリンカー・マップを調べる。例えば @STATICC が見つかった場合、ストリングや静的変数などの無名の書き込み可能な静的オブジェクトを持っています。

Language Environment が一時データ・キュー CESE にダンプ出力を書き込むのに消費する時間の最小化

Language Environment のランタイム・オプション TERMTHDACT は、Language Environment が未処理エラーで生成する診断出力のタイプおよび量を制御します。

TERMTHDACT(DUMP)、TERMTHDACT(TRACE)、TERMTHDACT(UADUMP)、または TERMTHDACT(UATRACE) を使用すると、実稼働環境において、かなり大きなオーバーヘッドが生じることがあります。これらを設定すると、大量のトレースバック、および Language Environment のダンプ・データが、CESE 一時データ・キューに書き込まれることがあります。

アプリケーション環境にトレースバックまたは CEEDUMP が不要な場合は、TERMTHDACT(MSG) を使用して、フォーマット設定された CEEDUMP が CICS 一時データ・キュー CESE に書き込まれるときのパフォーマンスのオーバーヘッドを取り除いてください。アプリケーションがトレースバックまたは CEEDUMP を必要としている場合は、TERMTHDACT の CICSDDS オプションを指定して、Language Environment の診断出力を、CESE 一時データ・キューにではなく、CICS ダンプ・データ・セットに送ってください。

Java アプリケーション: パフォーマンスおよび調整

CICS 領域を分析し、調整すると、Java アプリケーションとそれらのアプリケーションが実行される JVM のパフォーマンスを改善できます。

Java アプリケーションのパフォーマンスの改善について詳しくは、[Java のパフォーマンス改善](#) を参照してください。

CICS 領域で実行される Java ワークロードの管理と調整に CICS 統計を使用する方法について詳しくは、[JVM サーバー統計](#) を参照してください。

MVS および DASD: パフォーマンスおよび調整

MVS 下の仮想記憶の CICS のチューニングは、z/OS のシステム・チューニング、z/OS Communications Server SNA のチューニング、CICS チューニング、および VSAM チューニングなど、いくつかの要素に依存します。チューニングはトップダウン式の活動であるため、CICS のチューニングの前に、既に z/OS のチューニングが十分に行われていることを確認する必要があります。

仮想記憶の制約を減らして軽減するには、各種トランザクションの所要時間を短縮します。つまり、タスクの所要時間を短縮するようにします。z Systems ハードウェアをアップグレードすると、タスクの所要時間を短縮するためのファスト・パスになります。

- 高速なプロセッサをインストールすることにより、現行の命令の実行時間が短縮され、これにより、同じ時間で処理できるトランザクションが増えるため、タスクの所要時間 (内部応答時間) を短縮することができます。
- ページインが頻繁に発生する場合 (秒あたりのページインが 5 から 10 より大きい場合、CICS のパフォーマンスに影響)、実記憶の追加により、ページング・サブシステムの待ち時間を短縮することができます。
- 高速な DASD をインストールすると、入出力の完了待ち時間を短縮することができます。また、ページング操作、データ・セットの索引検索、またはデータ・セットのバッファ検索にかかる待ち時間を短縮することにより、プロセッサにおけるタスクの所要時間を短縮することもできます。

特に入出力の問題がないか調べるため、z/OS システムに以下の兆候が現れていないかどうかを確認します。

- サービス・レベル目標に達していない。
- ユーザーが応答時間について苦情を言う。
- 入出力標識がストレスの兆候を示す。つまり、重要なワークロードに関する高い DEV DLY または USG が、直接 Monitor III レポートに表示される。

詳しくは、[z/OS リソース測定機能 \(RMF\) レポート分析](#) で入出力アクティビティの分析に関するセクションを参照してください。

MVS では、MVS パフォーマンス・グループのストレージを分離することができるため、実記憶の特定の範囲を CICS アドレス・スペースに予約して、実行中に CICS アドレス・スペースにかかるタスク制御ブロック (TCB) 時間に基づいてアドレス・スペースのページ・レートを制御することができます。

DASD ドライブ、ストリング、およびチャネルの CICS データを分離して、システム内の他の DASD アクティビティから CICS が受ける入出力コンテンションを最小化することができます。このように DASD を分離した場合、CICS オンライン・システムによる入出力アクティビティは、CICS のパフォーマンスに重大な影響を及ぼすほどではありません。

ここまでは、(ストレージの分離と DASD 共用を除き)、独立した単一の CICS アドレス・スペースを実行する CICS システムを中心に説明をしてきました。すべての MVS アドレス・スペースは、最大のサブシステムに共通の要件により定義されます。2 つ以上のプロセッサのワークロードを MVS イメージに結合する場合は、単一イメージのプロセッサで実行する個々のサブシステムの仮想記憶要件に留意する必要があります (仮想記憶についての詳細は、75 ページの『CICS 仮想記憶』を参照)。以下のワークロードを単一イメージ MVS システムに結合する場合の仮想記憶への影響について検討してください。

1. CICS および多数の (100 以上) TSO ユーザー
2. CICS および大規模 IMS システム
3. CICS および 5000 から 7500 の SNA LU

CICS では、その特性上、大容量の専用領域が必要ですが、こうした他のサブシステムに関する大規模システムの共通要件を満たすと、大容量の専用領域が使用不可能になる場合があります。オペレーティング・システム、SNA、VSAM、および CICS のチューニング後にアドレス・スペースの要件がこの使用可能なサイズを超えている場合は、次の 3 つのオプションのいずれかにより CICS を分割できます。

1. 複数領域操作 (MRO)
2. システム間連絡 (ISC)
3. 複数の独立アドレス・スペース

大規模な新しいアプリケーションを追加したり、SNA ネットワークのサイズを大幅に増加した場合、仮想記憶の要求が増えます。実動システムに実装する前に、これらを分析する必要があります。十分な分析を行い、システムの仕様を検討することにより、仮想記憶の制約を受ける環境に新しいアプリケーションを追加した場合に発生するパフォーマンスの問題を避けることができます。必要な準備を行わないと、大規模アプリケーションの実装および実動システムの大きな変更を行った後で、大きな負荷につながる問題が発生する可能性があります。これらの症状として、以下があります。

- 応答時間の遅れ
- ストレージ不足
- プログラムの圧縮
- ページング・アクティビティの増加
- 十分にテストされた多くのアプリケーションに新しい症状が出て突然、異常終了する
- S80A および S40D 異常終了
- S822 異常終了
- DFHRPL 連結または動的ライブラリー連結の入出力アクティビティの大幅増

このセクションでは以降、MVS 下の CICS のパフォーマンス向上のために使用できる技法について説明します。

ネットワークングおよび z/OS Communications Server: パフォーマンスおよび調整

SNA ネットワークおよび論理装置 (LU) のパフォーマンスは、さまざまな方法で調整できます。

このセクションでは、次のトピックについて説明します。

- https://ut-ilnx-r4.hursley.ibm.com/ts42_latest/help/topic/com.ibm.cics.ts.performance.doc/topics/dfht34d.html
- [143 ページの『任意受信入力域サイズの設定』](#)
- [144 ページの『任意受信プールのサイズの設定』](#)
- [146 ページの『SNA での MVS 高性能オプションの使用』](#)
- [146 ページの『SNA トランザクション・フローにおける伝送数の調整』](#)
- [SNA チューニングを使用したラージ・メッセージのセグメント化](#)
- [148 ページの『同時ログオンおよびログオフ要求数の制限』](#)
- [149 ページの『端末スキャン遅延の調整』](#)
- [151 ページの『出力端末データ・ストリームの圧縮』](#)
- [端末の自動インストールの調整](#)

端末入出力域のサイズの設定

TYPETERM RDO リソース定義の **IOAREALEN** 属性は、トランザクションに渡される端末入出力域のサイズを指定します。マクロ・リソース定義が使用されている場合、DFHTCT TYPE=REMOTE マクロの **TIOAL** パラメーターによって TIOA のサイズを指定することもできます。

TYPETERM RDO リソース定義の **IOAREALEN** 属性の構文は、({0/value1},{0/value2}) です。この設定は、すべてのトランザクションの最初の入力メッセージにのみ使用されます。最小サイズを定義する 1 つの値は、非 SNA 装置に使用され、最小サイズと最大サイズの両方を指定する 2 つの値は、SNA 装置に使用されます。

ATI(YES) を指定した場合、少なくとも 1 バイトの **IOAREALEN** 値を指定する必要があります。

効果

IOAREALEN に value1,0 を指定した場合、value1 は、**RECEIVE** コマンドを発行した場合にアプリケーション・プログラムに渡される端末入出力域の最小サイズです。入力メッセージのサイズが value1 を超えると、アプリケーション・プログラムに渡される領域は、入力メッセージのサイズになります。

value1, value2 を指定した場合、value1 は、**RECEIVE** コマンドを発行した場合にアプリケーション・プログラムに渡される端末入出力域の最小サイズです。入力メッセージのサイズが value1 を超えるごとに、CICS は value2 を使用します。入力メッセージ・サイズが value2 を超えると、ノード異常条件プログラムが例外応答を端末に送信します。

制限

IOAREALEN (value1) 値または DFHTCT TYPE=REMOTE マクロの **TIOAL** パラメーターの値が、ネットワークのほとんどの端末入力に対して大きすぎると、実記憶が浪費される場合があります。ただし、**IOAREALEN** (value1) または **TIOAL** がほとんどの初期端末入力よりも小さいと、過剰な GETMAIN 要求が発生することがあり、追加のプロセッサ要求がなされる場合があります (**IOAREALEN** (value1) または **TIOAL** がゼロでない場合)。

推奨

IOAREALEN (value1) または **TIOAL** は、端末の平均入力メッセージ長さよりもやや大きな値に設定してください。**IOAREALEN** または **TIOAL** に指定できる最大値は、32767 バイトです。

ゼロ以外の値が必要な場合は、最も一般的に発生する入力メッセージのサイズを指定してください。64 バイトの倍数から 21 を引いた値が SAA 要件に適合し、オペレーティング・システムのページ使用が適切に行われます。

z/OS Communications Server では、インバウンド・チェーニングを使用する場合、2 つの値を指定できます。最初の値は端末の通常のチェーン・サイズの長さ、2 番目の値はチェーンの最大サイズです。タスクに対して示される TIOA の長さは、メッセージ長および TIOA に指定されたサイズによって異なります。次の例を参照してください。

Where x is any number of bytes, the following applies.

Without chain assembly:

If the TIOA size is specified as	20x
and the message length is	15x
then the TIOA acquired is	20x

If the TIOA size is specified as	20x
and the message length is	25x
then the TIOA acquired is	25x

With chain assembly:

If Value1 size is	20x
and Value2 size is	25x, then
if the length of a message is	15x
the TIOA acquired is	20x
and if the message length is	22x
the TIOA acquired is	25x

図 23. メッセージ長および端末入出力域の長さ

value1 は、例えば、端末の表示画面のサイズに合わせるなど、大きすぎる値を指定しないようにしてください。この領域は入力専用です。READ に SET を使用して指定した場合、アプリケーションでは出力域に同じポインターが使用されます。

value1 には小さすぎる値を指定しないようにしてください。チェーン・アセンブリーに追加の処理時間が必要になったり、インバウンド・チェーニングを使用しない場合にデータが失われたりするからです。

一般に、ゼロの値が最適です。値をゼロにすると、ストレージが最適に使用され、2 番目の GETMAIN 要求が除去されます。端末で自動トランザクション開始 (ATI) を使用する場合、1 バイトの最小サイズが必要です。

SNA 装置の 2 番目の値は、端末のストリーミングを防ぐために使用されるため、ネットワークで起こりうる最大の端末入力よりもやや大きくします。この 2 番目の値よりも大きなメッセージが発生すると、端末に否定応答が戻され、端末メッセージは廃棄されます。

モニター

RMF および NetView パフォーマンス・モニター (NPM) を使用すると、ネットワーク内のストレージ使用量およびメッセージ・サイズを表示できます。

任意受信入力域サイズの設定

システム初期設定パラメーター **RAMAX** は、各 SNA 任意受信操作に割り振られる入出力域サイズ (バイト) を指定します。**RAMAX** システム初期設定パラメーターは、LU に z/OS Communications Server SNA アクセス方式を使用するいずれのネットワークでも使用することができます。

これらのストレージ域は、任意受信入力域 (RAIA) と呼ばれ、SNA からのトランザクションの最初の端末入力の受信に使用されます。SNA からの入力はすべて、要求/応答単位 (RU) で着信します。

RAIA のストレージは 16 MB 境界を超えて置かれ、CICS 端末管理プログラムにより CICS の初期化中に割り振られます。このストレージは CICS ジョブ・ステップの実行期間にわたって割り振られたままになります。このストレージのサイズは、**RAPOOL** および **RAMAX** システム初期設定パラメーターの積となります。

効果

SNA は、着信 RU をいずれも初期の任意受信入力域 (サイズ **RAMAX**) に配置しようと試みます。この領域が十分な大きさでない場合、SNA は問題を示すメッセージを作成し、収容できずに待機している残りのバイト数を示します。

RAMAX は、CICS が任意受信コマンドで直接に受け入れられる RU の最大サイズです。CICS は、SNA から示された RU の全体サイズをこの制限に突き合わせて比較します。余分なサイズがある場合は、SNA はこれを保管し、CICS は 2 番目要求で残りを取得します。

RAMAX が小さい場合は、RAIA に取られる仮想記憶を減らすことができます。ただし、SNA が RAIA に収まらないデータを取得しようと再試行して、プロセッサ使用が増える結果となります。

多くの場合、**RAMAX** のデフォルト値の 256 バイトが適切な値です。多くの着信 RU がこの値よりも大きいことがわかっていれば、システムに応じて **RAMAX** の値を大きくできます。

個々の端末には、この装置から送られる RU のサイズを決めるパラメーターが別に用意されています。**RAMAX** は、少なくとも、頻繁に使用される端末の最大の **SENDSIZE** 属性と同じ大きさにするのが妥当です。

制限

RAMAX 値が高いと、実記憶が浪費されることがあります。**RAMAX** 値が低すぎると、残りデータを受信するための追加バッファの取得に余分なプロセッサ時間が必要になります。

推奨

RAMAX は、CICS が発行する各任意受信要求に割り振られた入出力域のサイズ (バイト) に設定してください。最大値は 32767 です。ほとんどの入力 は 256 バイトであるため、このサイズがデフォルト値として指定されています。

RAMAX は、CICS システム入力メッセージよりも若干大きく設定します。システムのメッセージ長の分布が分かる場合は、ほとんどの入力メッセージに対応するよう値を設定します。

いずれの場合も、**RAMAX** に必要なサイズに考慮する必要があるのは、メッセージの最初 (または唯一) の RU のみです。このため、SNA チェーニングを使用して送信されるメッセージでは、そのチェーン全体の長さに基づいて **RAMAX** を設定することを要求するのではなく、これを構成要素 RU のサイズにのみ基づいて **RAMAX** を要求することになります。

パイプラインは長さ超過データを処理できないため、パイプライン端末には、**RAMAX** 値を RUSIZE (CINIT の) より小さく指定しないでください。

任意受信入力域は、ストレージの固定長サブプールから取得されます。このような 2 つの領域を 4 KB の 1 ページに収めるには 2048 が適当なサイズのように思われますが、各ページで使用できるのは 4048 バイトのみであるため、1 ページに収まるのは 1 領域だけになります。ページ・ヘッダーを含む 2 領域を 1 ページに入れるには、サイズ 2024 を定義します。

モニター

ネットワーク内の RU またはチェーンのサイズは、SNA 行またはバッファのトレースにより識別できます。

任意受信プールのサイズの設定

RAPOOL システム初期設定パラメーターは、CICS が処理する、z/OS Communications Server for SNA からの同時任意受信要求の数を指定します。

RAPOOL は、常時存在する任意受信バッファの数を決定します。したがって、z/OS Communications Server for SNA に多数の入力が同時に発生した場合、z/OS Communications Server はすべてのメッセージを CICS バッファに直接入れることができ、別の場所に保管する必要がなくなります。最初のオペランド (value1) は、HPO 以外のシステム用で、第 2 オペランド (value2) は HPO システム用です。

非 HPO 用のオペランドの HPO の値は、『**RAPOOL**』に示す式に従って派生します。HPO システム用の第 2 オペランド (value2) は、式によって最小限の調整で使われます。

効果

最初に、端末またはセッションからのタスク入力が SNA アクセス方式により受信され、CICS で任意受信要求が未解決な場合は、CICS へと渡されます。

それぞれの任意受信要求について、SNA 要求パラメーター・リスト (RPL)、任意受信制御エレメント (RACE)、および任意受信入力域 (RAIA) が保留されます。RAIA 値は、**RAMAX** で指定します (RAIA の考慮事項については、143 ページの『任意受信入力域サイズの設定』を参照してください)。SNA の任意受信操作に保留された領域の合計は、次のとおりとなります。

(最大 RAIA サイズ + RACE サイズ + RPL サイズ) * RAPOOL

HPO=YES の場合、RACE および RPL 共に 16 MB 境界を超えて配置されます。

一般的に、**RAPOOL** に指定された値までの入力メッセージはすべて、端末管理タスクの一度のディスパッチで処理されます。任意受信要求の処理は短い操作であるため、ときには **RAPOOL** 値の指定を超えるメッセージが端末管理の一度のディスパッチで処理されることがあります。このような状況は、端末管理プログラムが処理を終了する前に任意受信要求が完了し、SNA から追加のメッセージがある場合に発生します。

指定されたプールは、トランザクションの最初の端末メッセージまたはタスクを開始するための最初の入力 SNA 任意受信処理にのみ使用されます。**RAPOOL** は、会話型タスクや出力のためのその他の入力には影響を与えません。その他の入力は、SNA 固有受信要求で処理されます。

SNA は、任意受信入力域に関連したイベント制御ブロック (ECB) をポストします。次に CICS は、タスク処理の準備ができた端末入出力域 (TIOA) にデータを移動します。これで、RAIA が再使用できるようになります。

RAPOOL の重要度は、CICS システムの環境によって異なります。例えば、HPO が使用される場合には、**RAPOOL** は重要です。

制限

RAPOOL 値の設定が低すぎる場合、端末メッセージが端末管理プログラムの最も早いディスパッチで処理されないことがあり、アクティビティーの多い期間にはトランザクションの遅延が生じる可能性があります。例えば、デフォルト値を使用して、5 つの端末入力でタスクを開始する必要がある場合、少なくとも SNA 任意受信要求を完了してデータと RPL をコピーするために必要な時間だけ、3 つのタスクに遅延が生じる可能性があります。一般的に、すべての任意受信処理の 5 % から 10 % 以内を **RAPOOL** の上限として設定し、十分なストレージがある場合は、**RAPOOL** の上限をなしに設定します。

RAPOOL 値の設定が高すぎる場合は、過剰な仮想記憶が使用されることがありますが、このストレージはページ固定でなく、したがってページアウトされるため、実記憶に影響することはありません。

推奨

場合によっては、CICS が大半の時間は未使用になる多数の RAIA を持つよりは、時々生じるピーク時のメッセージを SNA が独自の領域に保管する方が無駄のない場合があります。

また、CICS が発行した任意受信が受け入れられると、すぐに任意受信要求を再発行する場合があります。SNA 内の追加のメッセージを取り入れるため、同じエレメントが繰り返し再使用されます。

CICS は、RPL の n の z/OS Communications Server **VTAM RECEIVE ANY** を保守します。 n は、**RAPOOL** 値、または **MXT** 値から現在アクティブなタスクの数を引いたもののいずれか小さい方の値です。

RAPOOL には、必要とする固定の要求パラメーター・リスト (RPL) の数を指定します。これが **MXT** でなければ、CICS はこれら各 RPL に対する任意受信要求を保持します。必要とする RPL の数は、システムで予期されるアクティビティー、トランザクションの平均存続時間、および指定された **MXT** により異なります。

設定する **RAPOOL** 値は、システム初期設定テーブル (SIT) のセッションの数、端末の数、および **ICVTSD** 値 (149 ページの『端末スキャン遅延の調整』を参照) により異なります。最初は、非 HPO システムに対して **RAPOOL** をピークのローカル・トランザクション率/秒に自動インストール率を足した値の 1.5 倍に設定します。この値は、CICS SNA 統計の分析や、到達した最大 RPL に値をリセットすることにより調整が可能です。**RAPOOL** の値には MRO セッションは含まれないため、この値は、アプリケーション所有領域またはファイル所有領域 (AOR または FOR) で低い数に設定します。

HPO システムの場合、**RAPOOL** システム初期設定パラメーターの **value2** に指定する場合は、通常は小さい値 (≤ 5) で十分です。例えば **RAPOOL=20** は **RAPOOL=(20)** または **RAPOOL=(20,5)** に指定することにより、同じ効果を得ることができます。

モニター

CICS SNA 統計には、端末管理プログラムの一度のディスパッチでポストされた RPL の最大数、および RPL の最大に達した回数値が含まれます。この最大値は、端末管理プログラムが一度のディスパッチで RPL を再使用できる場合、**RAPOOL** 値より大きくすることができます。詳細については、[z/OS Communications Server 統計の解釈](#)を参照してください。

SNA での MVS 高性能オプションの使用

SNA 要求の処理のために、MVS 高性能オプション (HPO) を使用できます。HPO の目的は、z/OS Communications Server を通るトランザクション・パスの長さを短縮することです。

HPO および監視プログラム呼び出し (SVC) の使用は、システム初期設定テーブル (SIT) で指定されます。デフォルトの SVC 数を許容できる場合は、システムの調整は必要ありません。

効果

HPO は MVS により入出力操作に対して行われる検証機能の一部をバイパスし、サービス要求ブロック (SRB) スケジューリングをインプリメントします。このバイパスによって、命令のパス長さが短縮され、SRB スケジューリングによって、z/OS Communications Server 操作の MVS イメージ上の一部の並行処理が可能になります。この効果により、HPO はマルチプロセッサ環境では役立ちますが、シングル・プロセッサ環境では効果は得られません。

制限

HPO を使用するには、CICS に許可が必要です。MVS では、ユーザー作成のモジュールがいずれかの CICS システム初期化ルーチンを置き換え、許可モードで実行される可能性があるため、MVS の整合性に関するリスクが伴います。このリスクは、CICS SDFHAUTH データ・セットを保護する RACF によって削減することができます。

HPO の使用により、プロセッサ時間が節約され、実記憶または仮想記憶要件や入出力の競合が増すことがなくなります。HPO の問題の 1 つは、検証の不足により機密漏れが発生する可能性があることです。

推奨

入念に検査されたアプリケーションを使用するすべての実動システムで、HPO を使用できます。アプリケーションに透過的で、機能に制約をもたらすことなく、z/OS Communications Server を通るパス長さを短縮します。z/OS Communications Server の場合、検証が削減されることでメッセージの整合性が失われることはありません。

モニター

HPO を直接測定することはできません。機能しているかどうかを検査する 1 つの方式は、HPO をオンにした状態 (SIT オプション) とオフにした状態のプロセッサ使用量を詳細に測定することです。ワークロードに応じては、多くの違いは見られないかもしれません。また、HPO をオンにして SRB スケジューリングにわずかな増加が見られる場合があります。

RMF は、プロセッサ使用に関する一般情報を提供できます。SVC トレースには、HPO の使用状況が示されます。

新規アプリケーションまたは CICS コード (新規リリースまたは PUT) の初期テストに使用されたシステムで HPO を使用する場合は注意が必要です。パス長さの縮小は、多くが z/OS Communications Server における制御ブロック検証コードのバイパスにより行われます。未検証のコードにより、CICS が z/OS Communications Server に渡す制御ブロックが破壊され、未検証のアプリケーションにより機密漏れが生じる可能性があります。

SNA トランザクション・フローにおける伝送数の調整

CICS 内で、**MSGINTEG** オプションおよび **ONEWTE** オプションを使用することにより、ネットワーク内の端末と、z/OS Communications Server および NCP 通信プログラムとの間で交換される通信の要求および応答を制御することができます。これらのオプションは、Communications Server を使用するすべての CICS システムで使用できます。

オンライン・リソース定義 (RDO) を使用する場合、PROFILE 定義で **MSGINTEG** オプションおよび **ONEWTE** オプションを使用して、保護を指定できます。 **MSGINTEG** オプションは、SNA 論理装置 (LU) でのみ使用されます。 PROFILE リソースの定義方法については、[PROFILE resources](#) を参照してください。

効果

システム・ネットワーク体系 (SNA) のオプションの 1 つに、CICS と端末間のメッセージ交換を確定または例外応答のモードで行うかを指定できます。 確定応答モードでは、端末と CICS の両方が相互に 1 対 1 でメッセージの受信を確認します。

SNA でも、同期データ・リンク制御 (SDLC) によりメッセージを送達することができ、通常は確定応答を必要としません。 メッセージの整合性 (**MSGINTEG**) を指定することにより、指定されたセッションが確定応答モードで処理されます。

通常の場合は、CICS と端末間のセッションが例外応答モードで処理されます。

このため、以下のオプションがあります。

- **MSGINTEG** を指定しない
- **MSGINTEG** を指定する (確定応答を強制するよう要求する)

SNA では、トランザクションはブラケット内に定義されます。 開始ブラケット (BB) コマンドはトランザクションの開始を定義し、ブラケット終了 (EB) コマンドはそのトランザクションの終了を定義します。 あらかじめ CICS 側でメッセージがトランザクションの最後であることを識別していない場合には、トランザクションが終了するときに最後のメッセージとは別に EB を送信する必要があります。 EB は SNA コマンドで、メッセージと共に送信することができ、端末への必須送信を 1 つ減らすことができます。

トランザクションに 1 回書き込み操作 (**ONEWTE**) オプションを指定すると、そのトランザクションでは端末に 1 つだけの出力メッセージが送信され、CICS はそのメッセージと共に EB を送ることができます。

ONEWTE が指定された場合は、1 つの出力メッセージのみ許可され、2 番目のメッセージが送られると、トランザクションは異常終了します。

CICS が端末メッセージと共に EB を送信できるようにするもう 1 つの方法として、プログラムの最後の端末管理または基本マッピング・サポート **SEND** コマンドで **LAST** オプションを指定します。 複数の **SEND** コマンドを使用できますが、**LAST** オプションはプログラムの最後の **SEND** に指定する必要があります。

3 つ目の (最も一般的な) 方法は、**SEND without WAIT** を最後の端末通信として発行することです。 メッセージはタスク終了の一部として送信されます。

制限

MSGINTEG オプションにより、端末に追加の送信が行われます。 トランザクションが CICS にとどまる期間が長くなり、仮想記憶とリソース・アクセスが結合されます (最初のエンキュー)。 トランザクションでメッセージの配達を確認するには、**MSGINTEG** が必要です。

MSGINTEG を指定した場合、端末から応答を受信するまで TIOA はストレージ内に残ります。 このオプションにより、ストレージを必要とする期間が延びることから、CICS 領域の仮想記憶要件が増す場合があります。

モニター

MSGINTEG オプションおよび **ONEWTE** オプションの使用は、Communications Server トレースから、端末と CICS の間の交換を調べることによって、特に要求/応答ヘッダー (RH) の内容を調べることによってモニターできます。

SNA チェーニングを使用したラージ・メッセージのセグメント化

システム・ネットワーク体系 (SNA) では、端末メッセージのチェーニングが可能であり、大きなメッセージを小さいパーツに分割して、複数のメッセージを論理的に単一メッセージとして扱うことができます。 チェーニングは、チェーニングに対応したタイプの z/OS Communications Server SNA LU を使用するシステムで使用できます。

チェーニング特性は、**SENDSIZE**、**BUILDCHAIN**、および **RECEIVESIZE** の各属性を使用して指定します。

通常は、各端末のハードウェア要件で、入力チェーン・サイズと特性が決められます。**BUILDCHAIN** 属性および **RECEIVESIZE** 属性には、装置の属性に依存するデフォルト値があります。出力チェーンのサイズは **SENDSIZE** 属性により指定されます。

効果

通常の LU タイプ 0、1、2、および 3 装置の場合、ネットワーク制御プログラム (NCP) もメッセージを 256 バイトのブロックに分割するため、**SENDSIZE** 値をゼロにすると、出力チェーニングの処理による影響がなくなります。このタイプのローカル装置には値 0 または 1536 が必要です。

システム間連絡 (ISC) セッションに **SENDSIZE** 属性を指定する場合、この属性は、他のシステムの **RECEIVESIZE** 属性に一致する必要があります。**SENDSIZE** 属性または **TCT BUFFER** オペランドは、送信される SNA エLEMENT のサイズを制御します。**RECEIVESIZE** に一致させて、その ELEMENT を受信できる同じサイズの対応するバッファが存在するようにする必要があります。

BUILDCHAIN(YES) を指定した場合、CICS は ELEMENT の完全なチェーンをアセンブルしてから、それをアプリケーションに渡します。**BUILDCHAIN(YES)** を指定しない場合、RU はそれぞれ、アプリケーションの個々の任意受信へと渡されます。SNA/3270 では、**BUILDCHAIN(YES)** を指定しない場合には BMS は正しく機能しません。

最大サイズの 32 KB を超える大きなインバウンド・ELEMENT を処理する場合、**BUILDCHAIN** 属性または **CHNASSY** オペランドを使用することはできません。複数の RU を個別に使用する必要があります、これによりシステム内のトランザクション存続時間を延長します。

制限

低い **SENDSIZE** 値を指定した場合、この設定によって追加の処理が発生します。単一の論理メッセージを複数パーツに分割するために、実記憶と仮想記憶が使用されます。

端末装置によっては、チェーニングが必要になる場合があります。出力チェーニングにより、表示画面にフリッカーが発生する場合があります、これはユーザーを混乱させることがあります。また、チェーニングにより、追加の z/OS Communications Server サブタスクおよび STARTIO 操作が必要になり、z/OS Communications Server と NCP の間に追加の入出力処理による影響が生じます。これらの影響は、適切な ACF/SNA リリースで、ラージ・メッセージのパフォーマンス強化オプション (LMPEO) を使用することによって解消されます。

推奨

RECEIVESIZE 値は、IBM 3274 接続のディスプレイ 端末の場合は 1024、IBM 3276 接続のディスプレイ 端末の場合は 2048 です。これらの値により、プロセッサ使用を最小限に抑えると同時に、適切な回線特性を得ることができます。

モニター

チェーニングの使用とチェーニング・サイズは、z/OS Communications Server トレースを調査することによって判別できます。CICS 内部および補助トレース機能を使用することができます。VIO ZCP トレースにチェーン・ELEMENT が表示されます。NetView パフォーマンス・モニター (NPM) などのネットワーク・モニター・ツールには、この情報を得ることができるものがあります。

同時ログオンおよびログオフ要求数の制限

OPNDLIM システム初期設定パラメーターは、CICS で処理される z/OS Communications Server の同時ログオンおよびログオフ要求の数を定義します。このパラメーターは、z/OS Communications Server を端末アクセス方式として使用する CICS システムで使用できます。

OPNDLIM パラメーターは、すべてのユーザー・コミュニティが、例えば、昼休みなどに同時にログオン/ログオフするような場合にも役立ちます。

このパラメーターにより、同時ログオン OPNDST およびログオフ CLSDST 要求の数が制限されます。この値が小さいほど、プロセスのオープン/クローズに必要なストレージの量は少なくなります。このパラメーターについて詳しくは、[OPNDLIM システム初期設定パラメーター](#)を参照してください。

同時ログオンおよびログオフではそれぞれ、その処理の間、CICS 動的ストレージ域にストレージを必要とします。

効果

ログオンが CICS CONNECT=AUTO 機能または z/OS Communications Server LOGAPPL 機能により自動的に行われる場合は、CICS の始動時または再始動時に多数のログオンが生じる可能性があります。

自動ログオン機能が必要な場合、LOGAPPL 機能は 2 つの利点を提供します。CONNECT=AUTO 機能に比べて、z/OS Communications Server に必要なストレージが約 3500 バイト少なくて済みます。また、CICS の初期化時だけでなく、z/OS Communications Server に対して装置がアクティブになるたびに、端末が CICS にログオンされて戻されます。

制限

OPNDLIM に指定した値が低すぎると、CICS 内の実記憶域と仮想記憶域の要件が減り、z/OS Communications Server のバッファ要件も削減される場合がありますが、セッションの初期化および終了にかかる時間が長くなります。

推奨

最初はデフォルト値を使用し、ご使用の環境に必要なストレージが多すぎることや、起動時間が長すぎるものが統計で示された場合に調整してください。

OPNDLIM の値は、単一の z/OS Communications Server 回線に接続された論理装置 (LU) の数以上に設定してください。

モニター

ログオンおよびログオフ・アクティビティーは、CICS または測定ツールによって直接には報告されませんが、z/OS Communications Server トレースまたは z/OS Communications Server 表示コマンドで示される情報を使用して分析することができます。

端末スキャン遅延の調整

端末スキャン遅延 (**ICVTSD**) システム初期設定パラメーターは、CICS が端末出力要求の処理を試みる頻度を決定します。

ICVTSD システム初期設定パラメーターは、ミリ秒単位で定義します。コマンド **CEMT** または **EXEC CICS SET SYSTEM SCANDELAY (nnnn)** を使用して、**ICVTSD** の値をリセットしてください。

適度にアクティブなシステムでは、ゼロ以外の **ICVTSD** は実質的に ICV に置き換わります。これは、次の端末管理テーブル (TCT) の完全スキャン (非 SNA) または出力要求の送信 (SNA) までの時間が、オペレーティング・システムの待機時間に影響を与える主要な要因であるためです。

ICVTSD パラメーターは、アクティビティーが非常に少ない CICS システムを除き、すべてのケースで使用することができます。

一般的に、**ICVTSD** 値は、端末管理プログラムが以下の要求の処理を待つ時間を定義します。

- WAIT を指定した非 SNA LU 入出力要求
- タスク終了まで据え置かれた非 SNA 出力
- 自動トランザクション開始 (ATI) 要求
- アプリケーション・タスク・アクティビティーが多い CICS システムにおける SNA LU 管理 (出力要求処理を含む) この最後のケースは、CICS によるアクティブ・タスクのスキャン方法から生じるものです。

CICS の非 SNA システムでは、遅延値は、アプリケーション端末要求後、TCT スキャンを実行するまでに端末管理プログラムが待機する時間を指定します。この値は、端末管理要求の関連処理におけるバッチおよ

び遅延を制御します。アクティビティの少ないシステムでは、端末管理プログラムのディスパッチングを制御します。

SNA ネットワークでの効果

SNA ネットワークでは、**ICVTSD** 値を低くしても完全な TCT スキャンは行われません。これは、SNA LU との入出力はアクティブなキュー・チェーンから処理され、これらの端末入力のみがスキャンされるためです。

要求のバッチ処理によりプロセッサ時間は短縮しますが、応答時間が長くなります。CICS SNA システムでは、特に MVS 高性能オプション (HPO) が使用されている場合、端末管理プログラムが SNA 要求処理をいかに迅速に完了するかに影響を与えます。

SNA LU については、CICS はブラケット・プロトコルを使用して、端末が現在トランザクションに接続されていることを指示します。ブラケットは、トランザクションの開始時に開始され、トランザクションが終了すると終了します。トランザクションごとに端末に 2 つの出力、つまり、データ送信の出力とトランザクションの終了時にブラケット終了を含む出力が存在します。実際には、1 つの出力のみ送信されます (WAIT を伴う **WRITE/SEND** と確定応答の場合を除く)。CICS は、次の端末管理要求または終了時まで出力データを保持します。メッセージとブラケット終了または方向転換 (次の要求が **READ/RECEIVE** の場合) を同じ出力メッセージ (PIU) で送ることにより、プロセッサ・サイクルと回線使用率を節約します。システムがビジーな状態になると、端末管理のディスパッチ回数が減り、**ICVTSD** に指定された値への依存度が増すようになります。CICS は延長された期間におよび SNA にブラケット終了を送信できないため、トランザクションの存続期間を延長することができます。そのタスクに割り当てられたストレージを保持できる期間が延び、CICS 動的ストレージ域全体に必要な仮想記憶量を増すことができます。**ICVTSD** をゼロに設定すると、この効果をなくすることができます。

非 SNA ネットワークでの効果

ICVTSD は、非 SNA の完全 TCT スキャンの頻度を制御する大きな要因です。アクティブ・システムでは、完全スキャンは各 **ICVTSD** 期間に約 1 回実行されます。出力メッセージを送信する前の追加の遅延は平均で、この期間の約半分とします。

非 SNA ネットワークでは、LU からの入力の着信などの他の理由により部分的なスキャンが行われ、その回線の出力はすべて同時に処理されます。このため、非 SNA ネットワークでは、通常は 0.5 から 1 秒までの値が適切な設定です。

CICS は **ICVTSD** 主導のスキャンがない場合は、まず最初にアプリケーション・タスクをスキャンします。使用率の高いシステムでは、**ICVTSD** の値を高くしすぎると、入出力メッセージが非常に遅れる場合があります。

すべてのネットワークでの効果

ICVTSD パラメーターは、システム初期設定テーブル (SIT) で変更するか JCL パラメーターの指定変更により変更することができます。仮想記憶の制約問題がある場合は、**ICVTSD** に指定された値を減らしてください。値をゼロにすると、端末管理タスクのディスパッチ回数が最も多くなります。多数の非 SNA LU も存在する場合、この値では非生産的なプロセッサ・サイクル数が増すことがあります。この場合は、値を 100 から 300 ミリ秒にするのが適切です。ただし、純粋な SNA 環境では、平均的なトランザクションのパス長さが短くない限り、処理への影響は重要ではありません。応答時間と仮想記憶の使用を最適化するために、**ICVTSD** をゼロに設定してください。

制限

z/OS Communications Server (SNA 用) システムでは、値が低い場合、アクティブ・キュー TCTTE チェーンのスキャン処理への影響が増しますが、通常これは小さな考慮要因です。高容量システムで高い値に設定した場合、タスクの存続期間が増し、そのタスクが所有するリソースの結合期間が長くなります。これは、重要な考慮事項になることがあります。

ICVTSD の値を低い、ゼロ以外の値に設定すると、CICS がディスパッチされる頻度が増すことがあり、これによりパフォーマンス・モニターの処理への影響が増します。

推奨

ICVTSD の値を領域出口時間間隔 (ICV) (これもシステム初期設定テーブルにある) よりも小さく指定します。SNA LU およびコンソールのみを含む環境では、ワークロードに多数の短いトランザクションが含まれる場合以外は、この値をゼロにします。

SNA LU のみの環境で **ICVTSD=0** を入力するのは、端末アクティビティーが少なくタスク・アクティビティーが多い CICS ワークロードの場合、推奨されません。端末アクティビティーの少ない期間は、CSTP のディスパッチに遅延が生じる場合があります。**ICVTSD=100-500** を設定すると、CSTP が定期的にディスパッチされるようになり、この影響は解決されます。非 SNA システムでは、小ネットワーク (端末数が 1 から 30) の場合にのみ値をゼロに指定してください。

「純粋な」SNA ではないほとんどのシステムでは、この範囲を 100 ミリ秒から 1000 ミリ秒の領域に設定してください。**ICVTSD** は、300 ミリ秒から 1000 ミリ秒の範囲で変更しても応答時間には大きな影響を与えませんが、値を大きくするとプロセッサ・アクティビティーへの影響が少なくなります。**ICVTSD** を 1000 ミリ秒より大きくしてもプロセッサ使用はそれ以上改善されず、応答時間が長くなる場合があります。

ICVTSD を小さくした場合、プロセッサ・リソースが十分であれば、応答時間をわずかに縮小することができます。250 ミリ秒より小さく設定した場合は、応答時間の向上はユーザーが実感するほどではありませんが、プロセッサ使用に対する効果が増します。

絶対最小レベルは、「純粋な」SNA ではないシステムの場合は、約 250 ミリ秒です。また、非常に高性能で高出力の「純粋な」SNA の場合は、このレベルは 100 ミリ秒です。

モニター

RMF を使用して、タスクの期間とプロセッサ要件をモニターします。ディスパッチャー・ドメイン統計に **ICVTSD** の値が報告されます。

出力端末データ・ストリームの圧縮

出力メッセージに対し、CICS には、出力データ・ストリーム全体にアクセスするユーザー出口があります。データ・ストリームを端末に送信する前に、余分な文字をデータ・ストリームから除去するようユーザー・コードを作成できます。

z/OS Communications Server for SNA 装置では、端末メッセージの圧縮に使用されるグローバル・ユーザー出口は **XZCOUT1** です。プログラミング情報については、[SNA 実効ページ・セット・モジュール出口 \(XZCIN, XZCOUT, XZCOUT1, および XZIQUE\)](#) を参照してください。

不要な文字の比率が大きい場合、この圧縮技法により応答時間が大幅に改善されることがあります。それは、通信リンクが通常、ネットワークの最も遅いパスであるためです。

制限

出口コードの処理には追加のプロセッサ・サイクルが必要であり、出口ロジックのコーディングも必要です。圧縮出口を使用すると、SNA のストレージ要件が削減され、回線の伝送時間が短縮されます。

推奨

最も簡単な操作は、3270 タイプ装置のデータ・ストリームで、特にブランクなど余分な文字をアドレス反復シーケンスに置き換える方法です。

注：アドレス反復シーケンスは、一部のタイプの 3270 クラスター・コントローラーでの処理が早くありません。場合によっては、代わりの方法により優れたパフォーマンスを得ることができます。例えば、一連のブランクにアドレス反復シーケンスを送る代わりに、ERASE を送って、バッファ・アドレス設定シーケンスを送り、ブランク領域をスキップします。この方式は、バッファにブランクの代わりにヌルを使用できる場合に機能します。

この他に、伝送データ量を減らす方法として、出力データ・ストリームの保護フィールドの変更データ・タグをオフにします。この方法では、次の入力メッセージでこれらの文字をプロセッサに転送して戻す必要がなくなります。ただし、これを行う前に、フィールドへのアプリケーションの依存性を確認する必要があります。

個別のシステムではデータ圧縮の機会には他にもありますが、システムの設計を十分に調査してから実行する必要があります。

モニター

出力端末データ・ストリームの内容は、SNA トレースで調べることができます。

端末の自動インストールの調整

自動インストールの処理中、CICS は、拡張 CICS 動的ストレージ域 (ECDSA) の制御サブプールからストレージを取得し、各自動インストール要求を処理します。

取得される仮想記憶の量は、CINIT 要求単位の長さにより決められ、これは LU タイプごとに異なります。LU 6.2 端末からの一般的な自動インストール要求の場合、取得される動的仮想記憶の量は 120 から 250 バイトです。

自動インストール処理における CICS リソースの主要な消費者は、自動インストール・タスク (CATA) 自体です。何らかの理由で自動インストール・プロセスが通常処理で期待される速度で進まない場合は、システムが CATA トランザクション・ストレージでいっぱいになっている可能性があります。

最大同時自動インストール

AIQMAX システム初期設定パラメーターは、自動インストールで同時にキューに入れることのできる装置の最大数を指定します。

AIQMAX 値は、自動インストール可能な装置の総数を制限するものではありません。

再始動遅延パラメーター

AIRDELAY システム初期設定パラメーターは、自動インストール端末定義を CICS により始動時に保持するかどうかを指定します。

再始動遅延の値は *hhmmss* として指定され、デフォルトは 000700 (7 分) です。この遅延は、端末が緊急再始動後に 7 分以内に CICS にログオンしない場合、その端末入力削除スケジュールの対象となることを意味します。

再始動遅延をゼロに設定すると、CICS は、自動インストール端末入力を緊急再始動時にグローバル・カタログから再インストールしないよう指定します。この場合、CICS は、端末の自動インストール中にカタログに端末入力を書き込みません。この設定により、以下の処理におけるパフォーマンスに良い影響を与えます。

Autoinstall

入出力アクティビティーを除去することにより、自動インストールのパス長さが短縮されてプロセッサへの集約度が高まります。このため、一般的に端末の自動インストールにかかる時間が短縮されます。ただし、CATA の優先度が高く入出力アクティビティーを待つ必要がないことから、他のタスクの応答時間が若干増す場合があります。

緊急再始動およびウォーム・リスタート

自動インストール端末入力削除がカタログに書き込まれない場合には、CICS が緊急再始動時にグローバル・カタログ・データ・セットから復元するエントリが少なくなります。このため、自動インストール端末が多数ある場合、再始動遅延をゼロに設定することにより、再始動時間を短縮できます。

通常シャットダウン

CICS は、通常シャットダウン時にグローバル・カタログ・データ・セットから AI 端末入力を削除します (ただし、これらがカタログに書き込まれておらず (*AIRDELAY=0*)、端末が削除されていない場合)。再始動遅延をゼロに設定した場合、CICS は自動インストール時に端末入力をカタログに書き込まず、これらは削除されません。この設定により、通常シャットダウンの時間が短縮されます。

トラッキングが完了していないために一部の端末ユーザーを再度ログオンさせる手間、および再始動遅延をゼロに設定することの利点を考慮する必要があります。キャッチアップは数分で済むため、このようなテークオーバーが発生することはほとんどありません。

削除遅延パラメーター

AILDELAY システム初期設定パラメーターにより、端末のログオフ後に、自動インストール端末入力を使用可能な状態にする期間を制御できます。デフォルト値はゼロで、端末入力が端末のログオフ後すぐに削除されるようスケジュールされます。これ以外の場合、CICS は TCTTE の削除をタイマー・タスクとしてスケジュールします。

一般的に、多数の自動インストール端末が一日の間にログオン/ログオフされる場合、削除遅延をゼロ以外の値に設定すると、CICS のパフォーマンスを改善できます。ただし、この設定により、未使用の自動インストール端末入力ストレージが、削除遅延間隔が期限切れとなるまで他のタスクにより解放されなくなります。このパラメーターにより、ストレージの存続時間が自動インストール端末の存続時間と静的に定義された端末の存続時間の間にある端末を効率よく定義することができます。

削除遅延をゼロ以外の値に設定することにより、再始動遅延の値に応じてさまざまな効果を得ることができます。

ゼロ以外の再始動遅延。再始動遅延がゼロ以外の場合、CICS は自動インストール端末入力をグローバル・カタログに書き込みます。

削除遅延もゼロ以外の場合、CICS は端末が再度ログオンした場合に再使用できるよう、端末入力を保持します。この設定により、以下のアクティビティーが除去されます。

- 仮想記憶の端末入力の削除
- カタログおよびリカバリー・ログへの入出力
- 端末が再度ログオンした場合の端末入力の再作成

再始動遅延ゼロ。再始動遅延がゼロの場合、CICS は、削除遅延にどのような値が指定されていても自動インストール端末入力をグローバル・カタログに書き込みません。

削除遅延がゼロ以外の場合、CICS は端末が再度ログオンした場合に再使用できるよう、端末入力を保持します。この遅延により、仮想記憶の端末入力を削除したり、端末が再度ログオンしたときに端末入力を再作成したりすることによる処理への影響を減らすことができます。

効果

3 とおりの方法で、自動インストールの処理によりリソース使用を制御できます。

1. トランザクション・クラス制限を使用して、同時に存在する自動インストール・タスク数を制限します (『65 ページの『[トランザクション・クラス \(MAXACTIVE\) を使用してトランザクションを制御する方法](#)』を参照)。
2. CATA および CATD トランザクションを使用して、自動インストール端末を動的にインストールおよび削除します。多数の装置が自動インストールされている場合、**MXT** システム初期設定パラメーターに達したり、CICS のストレージが不足することにより、シャットダウンが失敗する可能性があります。シャットダウン障害の考えられる原因を回避するには、CATD トランザクションをそれ自身のクラス内に置き、同時 CATD トランザクション数を制限することを考慮します。
3. **AIQMAX** を指定して、自動インストール用にキューに入れることができる装置の数を制限します。この設定により、他のいくつかの異常イベントの結果として生じた、自動インストール・プロセスによる仮想記憶の異常消費から保護されます。

この限度に達した場合、**AIQMAX** システム初期設定パラメーターは、CICS による LOGON および BIND 処理に影響を与えます。CICS は、z/OS Communications Server に対して、LOGON 要求と BIND 要求を CICS に渡さないよう要求します。z/OS Communications Server は、CICS がさらに LOGON および BIND を受け入れ可能であることを示すまで、これらの要求を保持します (キューに入れられた自動インストール要求を CICS が処理したときに起こります)。

推奨

AIQMAX 制限により自動インストール・プロセスのスローダウンが顕著な場合は、これを引き上げてください。CICS システムにストレージ不足の兆候が見られる場合は、**AIQMAX** 制限を下げてください。可能な場合は、**AIQMAX** システム初期設定パラメーターを通常の運用中に到達した値よりも高い値に設定します。

設定を (再始動遅延=0) および (削除遅延= hmmmss>0) にすると、プロセッサおよび DASD の使用率が最も効率的になります。ただし、この効率性は仮想記憶を犠牲にして獲得されます。それは、TCT エントリが遅延期間が終わるまで削除されないためです。

パフォーマンス全般と仮想記憶使用において、再始動遅延と削除遅延の両方の値をゼロにするのが、多くのシステム全体に最適な設定です。

再始動遅延がゼロよりも大きい場合 (カタログがアクティブ状態)、自動インストールのパフォーマンスはグローバル・カタログの定義 (DFHGCDD) の影響を受けます。VSAM により使用されるデフォルト・バッファ仕様は、アクティビティの多いシステムでは十分ではないことがあります。

ログオンおよびログオフ時に非常に多数のメッセージが一時データに送られるため、これら出力宛先のパフォーマンスも考慮してください。

モニター

自動インストール統計を定期的に調査して、通常の運用時の自動インストール率をモニターします。

CICS MRO、ISC、および IPIC: パフォーマンスおよび調整

複数領域操作 (MRO)、SNA 経由のシステム間連絡 (SNA 経由 ISC)、および IP 相互接続 (IPIC) の各接続を使用すると、CICS システムが相互に通信し、リソースを共用することが可能になります。パフォーマンスは、接続で使用する相互通信機能および接続の管理による影響を受けます。

MRO、SNA 経由 ISC、および IPIC 接続を使用すると、以下の CICS 相互通信機能が使用可能です。

- 機能シップ
- 分散トランザクション処理
- 非同期処理
- トランザクション・ルーティング
- 分散プログラム・リンク

CICS 相互通信機能についての説明は、[CICS 相互通信の紹介](#)を参照してください。

CICS ISC/IRC 統計では、相互通信セッションおよびミラー・トランザクションの使用頻度が示されています。z/OS Communications Server SNA トレース、SVC トレース、および RMF から追加情報が得られます。

各トランザクションが多数の相互通信要求を行う場合、通常は機能シップがプロセッサ使用の最も大きな要因になります。得失分岐点を構成するトランザクション当たりの要求数は、要求の性質によって異なります。

分散トランザクション処理 (DTP) および非同期処理はどちらも、さまざまな要求を 1 回の交換でバッチ処理できるため、多くの場合、相互通信の最も効率的な機能です。ただし、DTP には、この機能を使用するよう特別に設計されたアプリケーション・プログラムが必要です。

ほとんどの場合、トランザクション・ルーティングには、システム間で 1 つの入力および 1 つの出力を伴うので、追加のプロセッサ使用量は最小になります。

MRO

複数領域操作 (MRO) は、一般にはシステム間通信 (ISC) よりプロセッサの使用量が少なくなります。SVC のパス長が、SNA の複数システム・ネットワーク機能を使用する場合よりも短くなるからです。CICS MRO では、長時間実行されるミラー・トランザクションおよびファスト・パス変換プログラムが提供され、プロセッサの使用量をさらに削減できます。

CICS システム間に十分な数の MRO セッションを定義して、予想されるトラフィック・ロードに対応できるようにします。実記憶および仮想記憶でのコストの増加は最小限であり、タスクの存続時間が短縮されるため、総合的な効果としてストレージの節約が見込めます。ISC/IRC 統計を調べて ([ISC/IRC システムおよびモード・エントリ統計を参照](#))、キューに入れられている割り振りがなかったことを確認します。また、すべてのセッションが使用されていることも確認します。ただし、定義された MRO セッション数が多すぎると、関連する ECB のテストに使用されるプロセッサ時間が著しく増加することがあります。

MRO によるトランザクション・ルーティングのみが必要な場合、プロセッサ使用量は比較的小さくなります。数値はリリースおよびシステムに依存しますが (例えば、仮想記憶間ハードウェアを使用しているかどうかによって異なります)、合計コストはメッセージ・ペアあたり 15 KB から 30 KB 命令の範囲と想定できます。トランザクション全体に対する割合でみると、この値はほんのわずかです (通常は 10% 以下)。通常、各トランザクションにはさらに多くの CICS 間フローがあるため、MRO 機能シップのコストは非常に大きくなる可能性があります。このコストは、各 CICS システム間のリソースの処理方法によって大きく異なります。

MRO は応答時間およびプロセッサ時間に影響することがあります。遅延は、1 つの CICS システムから次の CICS への要求を受け取る時に発生します。この遅延が発生する原因は、一方の CICS システムの CICS 端末制御が他方から送信された要求を検出してから、その要求を処理する必要があるためです。また、ユニプロセッサを使用している場合、MVS は 2 つの CICS システムのディスパッチングを調整する必要があります、これは追加の WAIT/DISPATCH プロセッサの使用および遅延を意味しています。

長時間実行されるミラー・タスクを確立する場合は、システム初期設定パラメーター **MROLRM=YES** を指定します。このようにすると、アプリケーションが作業単位内で多くの機能シップ要求を作成する場合に、ミラー・トランザクションとの通信を再確立する時間が短縮されます。

MRO を使用する場合、MVS 仮想記憶間サービスを使用により、SVC 処理のための一部のプロセッサの使用を除去できます。仮想記憶間サービスでは、制御ブロック用に (データ転送用にではなく) MVS 共通システム域 (CSA) ストレージを使用します。これも利点の 1 つになる可能性があります。ただし、MVS では、仮想記憶間サービスを使用するアドレス・スペースはスワップ不能である必要があることに注意してください。

ISC

MVS イメージ間で ISC が使用されている状態では、XCF/MRO を使用してください。CICS では、MVS システム間カップリング・ファシリティ (XCF) を使用して、トランザクション・ルーティング、機能シップ、および分散プログラム・リンク用に MVS イメージ間の MRO リンクをサポートします。また、LU6.1 プロトコルで目的に十分対応できる場合は、分散トランザクション処理に XCF/MRO を使用することもできます。XCF/MRO が消費するプロセッサのリソースは、ISC よりも小さくなります。

ISC ミラー・トランザクションを優先順位付けすることができます。CSMI トランザクションはデータ・セット要求用、CSM1 は IMS システムとの通信用、CSM2 はインターバル制御用、CSM3 は一時データおよび一時記憶域用、そして CSM5 は IMS DB 要求用です。これらの機能の 1 つが特に重要な場合は、その機能を残りの機能よりも優先させることができます。この優先順位付けは MRO の場合には効果的ではありません。それは、接続されたミラー・トランザクション・サービスは、接続されている間ほどの MRO 要求も処理するからです。

ISC 機能がシステムをフラグディングする傾向がある場合は、SNA VPACING ファシリティを使用してこれを制御できます。複数セッション (SNA 並列セッション) を指定し、システム間のマルチパスを許可すると、スループットが増加します。CICS では、LU6.2 セッションで SNA サービス・クラス (COS) テーブルを指定し、ネットワーク内の ISC トラフィックの優先順位を付けることができます。

MRO および ISC を使用した領域間通信のパフォーマンス・コスト

これらのトピックの表を使用すると、特定の CICS API 呼び出しにかかる相対的な処理時間を比較したり、全体的な処理時間に影響を及ぼすその他の要因のいくつかを調べることができます。また、これらの表は、パフォーマンスを考慮するとき、アプリケーション設計に関する決定を行う場合に役立ちます。トランザクションの時間を計算するには、インストールおよびアプリケーションに該当する項目を見つけ、それらの値を加算します。

これらの数値を処理する前に、以下の考慮事項に注意してください。

- 呼び出しごとのコストは、IBM で内部的に使用されているトレース・ツールから取得された 1 K またはミリ秒単位の命令カウントとして文書化されています。1 つの命令のそれぞれの実行は、カウントが 1 になります。他の命令よりも多くのマシン・サイクルを使用する命令に対して、加重係数が追加されません。
- 測定は CICS 領域内の単一のトランザクションをトレースすることによって行われるため、I/O の待機などの待機によってフル MVS WAIT となります。このコストは、この文書で報告されている数値に含まれています。ビジー・システムでは、ディスパッチャが行う必要のある処理を発見する可能性が高いため、フル MVS WAIT をとる可能性は低減されます。

- ・パフォーマンスを判断する場合、使用されている方法が異なるため、以前公開された数値とこの資料の数値を比較しないでください。

トランザクション・ルーティングのパフォーマンス・コスト

MRO XM	MRO XCF (CTC 経由)	MRO XCF (CF 経由)	ISC LU6.2
37.0	43.0	66.0	110.0

機能シップのパフォーマンス・コスト (MROLRM=YES)

タイプ	MRO XM	MRO XCF (CTC 経由)	MRO XCF (CF 経由)
開始/終了環境	13.2	13.2	13.2
各機能シップ要求	9.0	23.4	48.4
同期点フロー	9.0	23.4	48.4

注：

- ・これらのコストは、長期実行ミラーを持つ CICS システムに関連しています。
- ・ISC LU6.2 は、**MROLRM=YES** をサポートしていません。
- ・開始/終了環境には、セッションの割り振り、ミラー・トランザクションの開始、ミラー・トランザクションの停止、およびセッションの割り振り解除のコストが含まれます。

例えば、ローカル・ファイル・アクセスから MRO XM にマイグレーションし、トランザクションごとに 6 つの機能シップを要求する場合、追加コストは次のように計算できます。

$$13.2(\text{開始/終了}) + (6(\text{要求}) \times 9.0(\text{要求コスト})) + 9.0(\text{同期点}) = 76.2 \text{Initiate}^{\circ}$$

機能シップのパフォーマンス・コスト (MROLRM=NO)

長期実行ミラーがない場合、機能シップの読み取り要求によって、セッションの割り振りおよびミラーの初期化と終了のコストが発生します。ただし、保護リソース (READ UPDATE または WRITE など) に対して行った最初の変更によって、セッションとミラーが同期点まで保持されるようになります。

MRO XM	MRO XCF (CTC 経由)	MRO XCF (CF 経由)	ISC LU6.2
21.4	35.0	59.9	115.0

IPIC

CICS 提供のミラー・プログラム DFHMIRS は、スレッド・セーフ・プログラムとして定義されています。サポート対象の CICS 機能の場合、IPIC 接続を介してのみ、リモートの CICS 領域はスレッド・セーフ・ミラー・トランザクションを使用し、可能な場合はいつでも L8 オープン TCB で要求を実行します。IPIC 接続を使用してリモート CICS システム上の機能に対するコマンドを発行するスレッド・セーフ・アプリケーションの場合、TCB の交換が減少するため、他の相互通信方式に比べてアプリケーションのパフォーマンスが向上します。オープン TCB を使用することで、CICS 領域間のスループットが大幅に改善する可能性もあります。

アプリケーションによっては、長期実行ミラーを使用することによっても著しいパフォーマンスの向上が得られる場合があります。IPIC は、IPCONN の MIRRORLIFE 属性をサポートします。これを使用してミラー・タスクの存続期間およびセッションの保持時間数を指定することにより、効率が改善され、パフォーマンスが向上する可能性があります。

IPIC は、CICS TS 4.2 以降の領域間での LINK コマンドのスレッド・セーフ処理をサポートします。使用するスレッド・セーフ・プログラムから出される DPL 要求が、IPIC 接続を使用して別の領域に送信される場合は、動的ルーティング・プログラムを変更してスレッド・セーフの標準に合わせてコーディングすることで、パフォーマンスを改善できることがあります。

IPIC 接続を介して機能シップのファイル制御、一時データ、および一時記憶域の要求を行うと、要求されたリソースの場所を気にせずに CICS アプリケーション・プログラムを実行できます。IPIC 接続を使用するファイル制御要求と一時記憶域要求の機能シップは、CICS TS 4.2 以降の領域間においてスレッド・セーフです。IPIC 接続を使用する一時データ要求の機能シップは、CICS TS 5.1 以降の領域間においてスレッド・セーフです。最良のパフォーマンスを得るには、ファイル制御、一時データ、および一時記憶域要求のためにリモート CICS 領域で呼び出されるグローバル・ユーザー出口プログラムはすべて、スレッド・セーフ・プログラムとして使用可能にする必要があります。

IPIC 接続を使用して機能シップされたファイル制御要求の場合、オープン・トランザクション環境のパフォーマンス上の利点を得るには、ファイル所有領域でシステム初期設定パラメーター **FCQONLY=NO** を指定する必要があります。

システム間セッションのためのキュー管理

システム間リンクがシステムに追加されると、リモート・システムのパフォーマンスが低下するため、システム間リンクがトランザクション要求に十分に応答できない可能性があります。

ローパフォーマンスは、リソース不足や過負荷などの長期間にわたる状態、またはメモリー・ダンプの取得中などの一時的な状態のいずれかが原因であることがあります。どのような場合であっても、この問題のために、要求側のシステムで、長いキューが発生する危険性があります。

CICS では、以下のためのメカニズムが提供されています。

- システム間セッションを使用するためのトランザクション・キューの間、リソースを使い過ぎから要求側のシステムを保護する。
- リモート・システムの問題の検出。CICS は、システム間接続の問題を示すメッセージを発行でき、パラメーターは、問題が発生した時点、または問題がなくなった時点の判別に使用される基準を制御します。

次の 2 つのメカニズムがあります。

1. 接続リソース定義の QUEUELIMIT および MAXQTIME パラメーター。

QUEUELIMIT パラメーターは、セッションが解放されるのを待っている割り振り処理において、キューに入れることのできるトランザクションの数を制限します。トランザクションは、既にその制限に達しているキューを結合しようとしても、拒否されます。

MAXQTIME パラメーターは、応答していないように見える接続上での空きセッションを待っている、キューに入れられた割り振り要求の待ち時間を制御します。キューの処理速度が、新規の割り振りがそのキューの先頭に到達するのに、指定された時間よりも長くかかることを示している場合は、キュー全体がページされます。

2. XZIQUE ユーザー出口。割り振り要求がキューに入れられようとしたとき、または疑わしい問題が発生した後に、最初に割り振り要求が続いたときに、このユーザー出口に制御が与えられます。XZIQUE 出口もキューを制御できます。または、この出口を使用して、独自のより高度な制御を追加することができます。

どちらのメカニズムも、割り振りを発行したアプリケーション・プログラムに対しては効果は同じで、SYSIDERR 状態に戻されます。動的ルーティング・プログラムには、割り振り要求のキューの状態を示すための戻りコードも提供されます。

MRO および APPC システム間キューの管理用 XZIQUE 出口では、XZIQUE 出口、およびこの出口と CICS のそれ以外のもの (アプリケーション・プログラムおよび動的ルーティング・プログラムを含む) との関係に関するプログラミング情報が提供されています。

関連する統計

接続統計を使用すると、CICS システム間環境の問題を検出できます。

CICS は、接続ごとに以下を記録します。

- キューで接続を待っている割り振りの数、およびこの数のピーク値。(接続統計の「未解決割り振りのピーク数」。)

この統計を使用すると、使用しているシステム内で、接続に対して通常どの程度のキューイングが発生するのかを確認することができます。大きなキューが時折発生する場合は、そのキューを制御してください。

い。十分なセッション数が定義されていますか?には、接続に適切なセッション数を設定するためのアドバイスが多く記載されています。

キューの制御メカニズムごとに、CICS は各接続に対して以下の統計を記録します。

- キューが大きくなりすぎたために拒否された割り振りの数
- スループットが低すぎたために、キューがパージされた回数
- スループットが低すぎたためにパージされた割り振りの数

ISC/IRC システムおよびモード・エントリー統計の解釈には、これらの統計の説明およびその他の接続統計の説明も記載されています。

問題へのアプローチ方法および推奨事項

システム間リンクを使用しようとして待っているタスクの数を制御する場合は、キュー制限メカニズムを使用してください。

キューの長さがその最大になっても、システム内の MXT スロットを使いすぎないようにするには、この制御を使用する必要があります。トランザクションを必要なリモート領域に対応するクラスに分離できる場合は、TRANCLASS 定義の MAXACTIVE 設定も使用できます。

通常の実行中に自由に使用できるようにするために、十分な数のシステム間セッションを用意してください。セッション定義が余分なストレージを占有することはありません。たいていは、トランザクション・ストレージの占有が、そのセッション用の追加のストレージを超えています。セッションの数は、接続を使用する可能性のある、システム内のトランザクションのピーク数に対応している必要があります。使用されているセッションの最大数は、接続の端末統計から確認することができます。すべてのセッションが使用されている場合、接続統計には、割り振りがキューに入った回数が、要求の総数と比較して示されます。

問題のないシステムにおいてさえ、どの時点においても、アクティブになっているトランザクションの数には相当の変動があり、実際のピーク数は、使用システムのピーク時の数分に渡る平均を超える可能性があります。実際のピークよりも、平均を使用する必要があります。キューイング・メカニズムは、短期間の変動を処理することを意図したものであり、短期間キューが存在していても、懸念するような原因にはなりません。

キューの開始は、接続の応答速度のモニターを開始するためのシグナルとして、キュー制限メカニズムによって使用されます。大きな問題が生じるまでキューが形成されることがない場合、検出メカニズムが反応していません。システム内に常にキューが存在する場合、診断を誤る可能性があります。

キューの制限は、おおよそセッションの数と同サイズにセットする必要があります。つまり、多くの接続で、その累積したキュー容量が MXT に達する場合は、MXT によって課せられる制限内にセットする必要があります。この後者の場合、接続へのキュー・スロットの割り振りがさらに動的になるよう、キューの長さを制限する独自の方法 (ZXIQUE を使用する) を設計します。

MAXQTIME パラメーターは、潜在的な問題が発生した場合に予想されるユーザーの最大応答待ち時間を反映した値に設定できます。MAXQTIME パラメーターは、低いキュー制限と組み合わせて低い値に設定してはなりません。このようにすると、検出基準の感度が高くなるためです。

設定のモニター

キュー制御メカニズムによって拒否された割り振りの数は、モニターする必要があります。その数が多すぎる場合は、システムに対する要求を満たすだけのリソースが不足している、つまり調整が不十分である可能性があります。

キューがパージされた回数は、リモート・システムで重大な問題が発生した回数を示している必要があります。リモート・システムが応答しなくなっているのにパージが発生しない場合は、MAXQTIME パラメーターの設定値を調べてください。設定値が高すぎ、反応していない可能性があります。問題の発生を示す頻度が多すぎ、リモート・システムの応答時間の変動によって誤ったアラームが発生する場合は、このパラメーターが低すぎるか、QUEUELIMIT 値が低すぎる可能性があります。

トランザクション・クラス DFHTCLSX および DFHTCLQ2 を使用したストレージ使用の制御

RDO グループ DFHISCT 内の DFHTCLSX と DFHTCLQ2 を使用して、CLS1、CLS2、および CLQ2 の各トランザクションを実行するために CICS が使用するストレージの量を制御します。

効果

これらのタスクは、APPC 会話を獲得し (CLS1/2)、MRO および APPC 接続のための作業単位を再同期化するために必要となるアクティビティーを実行します。通常、タスクの数は多くないので制御は不要です。ただし、CICS システムに多くの接続定義がある場合は、始動時のシステムの初期設定の結果、または **SET VTAM OPEN**、あるいは **SET IRC OPEN** コマンドの結果、これらの接続が同時に獲得される可能性があります。

注: VTAM は、現在は z/OS Communications Server と呼ばれています。

実装方法

システム定義はオプションです。リソース・グループ DFHISCT をインストールし、それらを活動化します。提供時の DFHTCLSX および DFHTCLQ2 内の **MAXACTIVE** パラメーターの値は 25 です。この値で、システムがストレージ不足の状態に達してしまうのを防ぐのに十分な制御を提供します。タスク CLS1 および CLS2 はそれぞれ、12KB の動的ストレージを必要とし、CLQ2 タスクは最大 17KB を必要とします。ページしきい値はゼロ以外の数値に設定しないでください。また **MAXACTIVE** パラメーターは 0 に設定しないでください。どちらの値も、CICS がシステム間機能に必要なタスクを実行するのを妨げる可能性があります。

MAXACTIVE 値を低すぎる値に設定しないでください。ネットワーク遅延またはネットワーク・エラーが原因で、TCLASS 内のタスクの 1 つが待機し、後続のトランザクションによる TCLASS の使用をブロックする可能性があるためです。低い値に設定すると、多数の接続があるシステムでは、シャットダウン時間が長くなる可能性もあります。

MRO セッションの端末入出力域の長さ (SESSIONS IOAREALEN) の制御

MRO 機能シップには、SESSIONS 定義属性 **IOAREALEN** が使用されます。この属性は、MRO リンクで伝送されたメッセージの処理に使用される、端末入出力域 (TIOA) の長さを制御します。これらの TIOA は、16 MB 境界よりも上に配置されます。

IOAREALEN 値は、他の CICS システムに伝送されるメッセージ (すなわち、出力メッセージ) を作成するために使用される TIOA の長さを制御します。値は 2 つ指定できます (value1 および value2)。value1 は、MRO 接続に対して定義されている各セッションで使用される TIOA の初期サイズを指定します。メッセージのサイズが value1 を超えた場合、CICS はさらに大きな TIOA を獲得して、そのメッセージを収容します。1 つの値のみが必須です。ただし、value2 が指定されている場合、CICS はメッセージのサイズが value1 を超過すると value2 を使用します。

値がゼロの場合、CICS は、出力メッセージのサイズと CICS 要件のための 600 バイトを加えたストレージ域を取得します。IOAREALEN 値が指定されていない場合は、デフォルト値の 4 KB に設定されます。

利点

IOAREALEN 属性は、MRO トランザクション・ルーティングまたは機能シップのいずれかのセッション定義で使用できます。MRO トランザクション・ルーティングでは、値によって TIOA の初期サイズが決まりますが、MRO 機能シップ環境の場合は、値を調整する機会が何度かあります。

制限

IOAREALEN の値が、MRO リンク上で伝送されるほとんどのメッセージにとって大きすぎる場合は、実記憶および仮想記憶が無駄になることがあります。IOAREALEN がほとんどのメッセージよりも小さいか、またはゼロの場合は、FREEMAIN および GETMAIN 要求が発生しすぎて、プロセッサ要件を追加する必要があります。

推奨

最適のストレージおよびプロセッサ使用状況を得るには、セッションが定義されている MRO を介して伝送される、最も一般的に見られるフォーマット済みのアプリケーション・データの長さよりも **IOAREALEN** を少しだけ大きくします。

効率的なオペレーティング・システムのページングを得るには、CICS 要件のための 600 バイトを追加し、合計を 64 バイトの倍数に切り上げます。64 バイトの倍数 (またはそれ未満) から、CICS 要件のための 600 バイトを引いたものが、オペレーティング・システムのページの使用として適していることが保証されます。

実装方法

TIOA サイズは、SESSIONS 定義の **IOAREALEN** 属性で定義できます。

要求のバッチ処理 (MROBTCH)

領域内のいくつかのイベントは、通知の前に、MROBTCH システム初期設定パラメーターで指定されている数に達するまで (または ICV がタイムアウトになるまで)、バッチに累積できます。

次に、それらの要求を処理できるようにするために、その領域が開始されます。MRO 要求のバッチ処理には、以下に示したいいくつかの非 MRO イベントが含まれています。

- VSAM 物理入出力の完了
- CO TCB (たいていは VSAM で、SUBTSKS=1 と指定されている場合) 上のサブタスクとして実行される要求の完了
- DBCTL を介して実装されている DL/I 要求の完了

厳密に言えば、バッチ処理は領域ではなく、TCB に適用可能です。MROBTCH は、「準再入可能」モードの TCB にしか適用されません。

MROBTCH のデフォルト値の変更の効果

バッチ処理を行わない場合 (MROBTCH=1、つまりこれがデフォルトです) と比較して、MROBTCH=n を設定すると、以下のような効果が得られます。

- その TCB の待機および通知のためのプロセッサ使用量が、最大 $[(n-1)*100/n]\%$ 節約されます。例えば、n=2 の場合は 50% の節約、n=3 の場合は 66% の節約、n=6 の場合は 83% の節約を達成できる可能性があります。
- 平均コストは、バッチ処理された要求ごとに、 $(n+1)/2$ と平均着信時間の積になります。
- 応答時間が長くなると、並行トランザクションの平均数が増えるので、仮想記憶全体の使用量が増える可能性があります。
- ピーク使用時の負荷が重いシステムでは、使用中のリソースを得るためのキューイングの当然の結果として、何らかのバッチ処理が発生することがあります。1 よりも大きく、値の小さい MROBTCH を使用すると、ピーク時とオフピーク時の応答時間の差は減少する可能性があります。

MROBTCH を 6 よりも大きくすることはお勧めしません。プロセッサの節約量の追加分が減少していき、それを上回る応答時間の増加に見合わなくなるからです。

使用率が低い期間の間、妥当な応答時間を維持するには、ICV の MROBTCH の値を小さくしておく必要があります。

適切なバッチ値の設定

許容できる応答時間の低下量に応じて、MROBTCH をさまざまな値に設定することができます。特定のワークロードに対して適切なバッチ値に到達するには、CICS Explorer の CICS-SM パースペクティブ (「操作」>「Regions (領域)」ビュー>「Region attributes (領域属性)」>「MRO Batch requests (MRO バッチ要求)」) または EXEC CICS SET SYSTEM MROBTCH を使用してください。

EXEC CICS システム・プログラミング・コマンドについてのプログラミング情報は、[システム・コマンド](#)を参照してください。

遅い期間の間は、バッチが完了していなくても ICV が無条件に領域をディスパッチし、遅延を最小にしてくれます。この場合は、各領域で ICV を 500 ミリ秒に設定してください。

ミラー・トランザクションの存続期間の延長 (MROLRM および MROFSE)

MROLRM システム初期設定パラメーターは、MRO 機能シップ環境における作業負荷のパフォーマンスに大きな影響を与えることがあります。

MROLRM=NO を設定すると、リカバリー可能リソースに対する最初の要求、またはファイル制御のブラウズ開始が受信されるまで、機能シップされた要求ごとにミラー・トランザクションが接続または切り離されます。そのような要求が受信された後は、呼び出し側のトランザクションが同期点に達するまで、ミラー・トランザクションはセッションに接続された状態を維持します。

機能シップ要求の受信領域で MROLRM=YES を設定すると、呼び出し側のトランザクションが同期点に達するまで、ミラー・トランザクションは、最初の要求による MRO セッションに接続された状態を続けます。このオプションを指定すると、以下に示したように、システムによって異なる影響が生じます。

- 一部のシステムでは、トランザクションあたりのプロセッサの使用率に大きな改善が見られます。このようなシステムは、それぞれが複数の VSAM 呼び出しを伴う照会トランザクション、または多くの呼び出しの後にいくつかの更新を伴うトランザクションのパーセンテージがかなり大きいシステムである可能性があります。
- 一部のシステムでは、パフォーマンスに差が見られません。IMS を使用している作業負荷、または VSAM 更新またはブラウズ・アクティビティーをよく利用しているトランザクションが、このカテゴリーに該当します。
- 一部のシステムは、同期点で余分なフローがあるために、機能が低下することがあります。この例としては、非常に単純な照会トランザクションの作業負荷を伴うシステムがあります。

一般に、MROLRM=YES を設定することをお勧めします。

フロントエンド領域に MROFSE=YES を設定すると、バックエンド領域のミラー・タスクが同期点の後に終了しなくなります。バックエンド領域のミラー・タスクは、フロントエンド・タスクが終了したときにのみ終了します。

フロントエンド領域で MROFSE=YES を使用することは、機能シップ要求に対する長時間実行のタスクが使用されることがあるときは推奨されません。これは、未使用のとき、他のタスクへの割り振りに SEND セッションを使用できないためです。さらに、タスクが終了するか機能シップ要求を発行するまで、バックエンド領域との接触が失われたときに接続が解放されることを防止します。

シップされた端末定義の削除の制御 (DSHIPINT および DSHIPIDL)

トランザクション・ルーティング環境では、端末定義は、端末占有領域 (TOR) からアプリケーション専有領域 (AOR) にシップできます。

AOR でシップされた端末定義は、以下の場合に冗長になります。

- 端末ユーザーがログオフする。
- 端末ユーザーが、AOR に送られるトランザクションの使用を停止する。
- ユーザーがサインオンした TOR がシャットダウンする。
- 自動インストールされた端末定義をリカバリーせずに TOR が再始動し、自動インストール・ユーザー・プログラム DFHZATDX が新規の端末 ID のセットを、端末の同じセットに割り当てる。

シップされた端末定義が冗長になった場合は、それを削除できます。長期にわたって続くシップされた端末定義は、一般にはストレージ問題を起こしません。それらの定義が占有するストレージの量が比較的小さいからです。ただし、セキュリティなど、別の考慮事項があります。セキュリティ上の理由で、シップされた冗長な端末定義が AOR に存続することは許可されない場合があります。

CICS 提供のトランザクション CRMF は、AOR 内のシップされた端末定義を定期的にスキャンし、冗長と判断した、シップされた端末定義にフラグを立てます。冗長な定義が識別された場合は、CICS 提供のトランザクション CRMD が呼び出されて、それらを削除します。この処理を CICS タイムアウト削除メカニズムといいます。

システム初期設定パラメーター **DSHIPINT** および **DSHIPIDL** は、冗長なシッパされた端末定義が存続を許されている期間、およびシッパされた端末定義が冗長でないかをテストする頻度を制御します。

効果

DSHIPIDL システム初期設定パラメーターは、シッパされた端末定義が、削除対象としてフラグが立てられる前に非アクティブ状態に留まっていられる期間を決定します。

DSHIPINT システム初期設定パラメーターは、CRMF トランザクションの呼び出しと次の呼び出しの間の時間間隔を決定します。CRMF は、シッパされた端末定義をすべて調べて、**DSHIPIDL** によって指定された時間間隔よりも長い間アイドルになっていた端末定義を判別します。CRMF が冗長な端末定義を識別した場合は、CRMD を起動して、それらを削除します。

利点

CRMF/CRMD 処理が最も効率的なのは、かなりの時間の間アイドル状態にあるシッパされた端末定義が AOR 内に存在している可能性のある、トランザクション・ルーティング環境においてです。

実装

シッパされた端末定義を、削除対象としてフラグを立てられる前にアイドル状態にしておくことができる最大時間は、CICS システム初期設定パラメーター **DSHIPIDL** で指定されます。アイドルの定義が存在していないかをテストするためのスキャンと次のスキャンの間隔は、CICS 初期設定パラメーター **DSHIPINT** で指定されます。

これらのパラメーターは両方とも調整できます。タイムアウト削除メカニズムの次の呼び出しまでのインターバルを修正した場合、そのインターバルは、このコマンドが最後に起動された時間からでも、CICS の始動時間からでもなく、このコマンドが発行された時間から開始することに注意してください。

モニター

CICS 端末自動インストール統計は、**DSHIPINT** および **DSHIPIDL** パラメーターの現在の設定、作成および削除された、シッパされた端末定義の数、およびシッパされた端末定義のアイドル時間に関する情報を提供します。

制限

DSHIPINT 値は、シッパされた端末定義が削除される前に、その端末定義がどれくらいアイドルの状態で存続しているかを決定する際の重要な要因です。

シッパされた端末定義が CRMF/CRMD 処理によって削除された後に、端末ユーザーが次にトランザクションを TOR から AOR に送付したときは、端末定義を再シッパする必要があります。**DSHIPIDL** 値は、シッパされた端末定義がトランザクション間に頻繁に削除されるような低い値に設定してはなりません。そのような処理を行うと、シッパされた端末定義の削除の場合だけでなく、次のトランザクションが送付されたときの、以降の再インストールの場合の CPU 処理コストも発生することがあります。

DSHIPINT に大きな値を選択すると、シッパされた端末定義の存続時間の長さに影響が出ることを考慮してください。シッパされた端末定義が、削除前にアイドル状態を続けている期間は、**DSHIPINT** 値の半分の平均だけ延長されます。これが起こるのは、**DSHIPIDL** パラメーターによって設定されたアイドルの端末に対する制限を端末が超えた後、CRMF が端末定義をアイドルであると識別してその端末定義にフラグを立て、CRMD がそれを削除するスケジュールが立てられる前に、(**DSHIPINT** インターバルの半分の間) その端末を待機させる必要があるからです。**DSHIPINT** インターバルが **DSHIPIDL** インターバルよりも著しく長い場合 (これは、**DSHIPINT** のデフォルト値である 120000、および **DSHIPIDL** のデフォルト値である 020000 が受け入れられている場合です)、**DSHIPINT** は、シッパされた端末定義が削除される前に、その端末定義がどれくらいアイドルの状態で存続しているかを決定する際の重要な要因になります。

推奨

DSHIPIDL には小さすぎる値を割り当てないでください。シッパされた端末定義が占有するストレージは、通常は問題になりません。したがって、(セキュリティーなど) 他の考慮事項によって **DSHIPIDL** の値

をさらに短くする必要があると示されていない限り、最大のアイドル時間として2時間を指定しているデフォルト値は妥当な値です。

アイドルの、シップされた端末定義を段階的に削除するのか、一度に削除するのかを決定してください。本来 CRMF 処理による CPU のオーバーヘッドは無視できるため、DSHIPIDL に適切な値が選択されている場合は、DSHIPINT に小さい値を指定して、コストを低く抑えることができます。CRMF が比較的頻繁に実行するように DSHIPINT に対して小さい値を指定すると、アイドルの端末定義はより小さいバッチで識別されるので、それらの定義を削除するのに必要な CRMD 処理は、長時間にわたって行われます。

DSHIPINT の値が大きくなると、特にデフォルト値である 12 時間が受け入れられている場合は、CRMF がかなりの数のアイドルの端末定義を識別するので、CPU は、CRMD 処理をかなり集中的に行う必要があります。CICS 領域での活動があまり活発でない期間にこのタイプの処理が行われるようにするには、INQUIRE/SET/PERFORM DELETSIPPED コマンドを使用すると、CRMF トランザクションを起動する時期をスケジュールするのに役立ちます。

CICS VSAM およびファイル制御: パフォーマンスおよび調整

このセクションでは、VSAM およびファイル制御に関連したパフォーマンス・チューニングについて説明します。

VSAM のチューニング: 一般的な目標

チューニングにより、十分なレベルのシステム・サービスを受け入れ可能なコストで実現します。十分なサービスは、VSAM の場合、適当なバッファを用意して物理的入出力を最小化し、同時にデータ・セットに対して複数の操作を並行して行うことによって得ることができます。

追加バッファの割り振りとデータ・セットの並行操作を行うためのコストには、バッファおよび制御ブロックに必要な追加の仮想記憶と実記憶があります。

VSAM データ・セットのパフォーマンスは、ローカル共用リソースを使用するか、非共用リソースを使用するかなど、複数の要因の影響を受けます。CICS データ・セットを圧縮してストレージを解放し、パフォーマンスを改善する場合は、CICS が稼働していない間に圧縮を行う必要があります。CICS のシャットダウンを避けるため、LIBRARY リソースを使用してデータ・セットを簡単にオフラインにし、連続可用性に影響することなく圧縮を行います。詳しくは、[動的プログラム LIBRARY リソースの使用](#)を参照してください。

チューニングのもう 1 つの要因はロッキングです。入力順データ・セット (ESDS) を使用すると、複数のタスクを使用してレコードを追加する場合に、パフォーマンスが低下する場合があります。これは、レコードの追加では、書き込みを実行するために排他的な add-to-end ロックが必要なためです。

ファイルとデータ・セットは、次のように区別されます。

- ファイルは、インストールされた CICS ファイル・リソース定義および VSAM ACB により定義されたデータ・セットのビューを示します。
- データ・セットは VSAM の範囲を意味し、関連の代替索引パスを含めた基本クラスターが含まれます。

ローカル共用リソース (LSR) または非共用リソース (NSR)

VSAM バッファおよびストリングに対してローカル共用リソース (LSR) を使用するか、非共用リソース (NSR) を使用するかをファイルごとに決定する必要があります。

特定の VSAM データ・セットにアクセスするために開かれるすべてのファイルは通常、同じリソース・タイプを使用する必要があります。

VSAM 制御間隔 (CI) へのアクセス

LSR と NSR の重要な違いは、VSAM 制御間隔 (CI) への同時アクセスにあります。

- LSR では、ストレージには CI のコピーが 1 つだけ存在し、2 番目の要求は、最初の処理が完了するまでキューに入れる必要があります。LSR では、複数の読み取り操作が同じバッファへのアクセスを共用することが許可されます。
- NSR では、ストレージ内に CI の複数のコピーを持つことができます。1 つ (ただし 1 つだけ) のストリングで CI を更新し、その他のストリングでは同じ CI の異なるコピーを読み取ることができます。

ただし、更新ではバッファを排他使用する必要があり、前の更新または前の読み取りが完了するまでキューに入れる必要があります。また、読み取りでは、すべての更新が終了するまで待つ必要があります。したがって、同時ブラウズ操作および同時更新操作を行うトランザクションは、NSR では正常に実行されても、LSR では 2 番目の操作が最初の操作の完了を待つことができずにデッドロックが発生する可能性があります。

CICS モニター機能を使用すると、各ユーザー・タスクの排他制御待ち時間のパフォーマンス・データを取得することができます。DFHFILE グループ内のパフォーマンス・データ・フィールド 426、FCXCWTT により、タスクが VSAM 制御間隔の排他制御を待機した経過時間がわかります。

制御間隔のサイズ (CI)

データ・セットの CI のサイズは、CICS に指定されるパラメーターでなく、VSAM AMS によって定義されます。ただし、制御間隔へのアクセスを提供する CICS システムのパフォーマンスに大きく影響します。

一般に、直接入出力は、データ CI が小さい場合にやや高速で実行され、一方、順次入出力は、データ CI が大きい場合により高速になります。NSR ファイルの場合、小さいデータ CI を使用しながら追加バッファを割り当て、順次入出力をチェーニングおよびオーバーラップすることにより、中間的な解決が可能です。ただし、追加のデータ・バッファはすべて、順次入出力を行う最初のストリングに割り当てられます。

VSAM は、制御域が最大サイズのときに最も効率的に機能します。データ CI を索引 CI より大きく設定します。このため、標準の CI サイズは、データの場合は 4 KB から 12 KB、索引の場合は 1 KB から 2 KB となります。

通常はファイルのデータ CI のサイズを指定しますが、対応する適当な索引 CI を VSAM で選択することができます。例外として、キー圧縮が VSAM で予想したよりも効率的でない場合があります。この場合、VSAM が選択する索引 CI サイズが小さすぎる場合があります。非常に高レートで制御域 (CA) 分割が行われ、DASD スペースが十分利用されていない可能性があります。この問題が疑われる場合は、より大きい索引 CI を指定してください。

LSR の場合、CI サイズを標準化すると利点が得られる場合があります。この標準化により、ファイル間でのバッファ共用を増やし、バッファの総数を減らすことができます。これとは逆に、ファイルに固有の CI サイズを与えて、同じプールを使用する他のファイルを含むバッファを競合しないようにすることも可能です。

CI サイズは、512 バイト、1 KB、2 KB など、4 KB の倍数に設定してください。26 KB や 30 KB のような異常な CI サイズは使用しないでください。CI サイズを 26 KB にしても、物理ブロック・サイズが 26 KB になるわけではありません。物理ブロック・サイズは装置依存であるため、この場合の物理ブロックはおそらく 2 KB になります。

ESDS ファイルの考慮事項

ESDS ファイルの **STRINGS** 値を選択する場合、パフォーマンス上の特別な考慮事項があります。

ESDS を追加専用ファイルとして使用する場合 (ファイル末尾にレコードを追加するため書き込みモードでのみ使用する)、ストリング数を 1 にすることをお勧めします。ストリングの数を 1 よりも大きくすると、複数のタスクが同時に ESDS に書き込みを行おうとした場合に排他制御が競合するため、パフォーマンスに大きく影響します。

ESDS を書き込みと読み取りの両方にする場合 (書き込みがアクティビティの 80%)、1 つのファイルを書き込み用、もう 1 つを読み取り用に、2 つのファイル定義を定義してください。

バッファの数

LSR と NSR の大きな違いは、VSAM によるバッファの割り振りおよび共用方法です。

LSR プール内で 1 サイズを占めるバッファのセットをサブプールと呼びます。ファイル制御ファイルには、最大で 255 個の個別の LSR プールを使用します。また、LSR プールへのデータ・セットの分散方法も決定する必要があります。CICS には、データおよび索引レコード用に個別の LSR バッファ・プールがあります。データ・バッファのみを指定した場合、1 セットのバッファのみ作成されて、データおよび索引レコードの両方に使用されます。各サブプールのバッファの数は、LSRPOOL 定義の DATA および INDEX パラメーターにより制御されます。正確な数を指定することも、CICS で数を計算することもできます。

NSR ファイルまたはデータ・セットは、バッファと制御ブロックの独自のセットを持ちます。ファイルの STRINGS パラメーターで指定された同時アクセスをサポートするために、各ファイルに対して十分なバッファを用意する必要があります。VSAM では NSR に対してこの要件を強制しています。NSR は、トランザクション分離を使用するトランザクションではサポートされません。NSR ファイルを使用するファイル制御コマンドは、スレッド・セーフではありません。

詳しくは、[166 ページの『LSR および NSR のバッファおよびストリングの数』](#)を参照してください。

ストリング数

次に、各ファイルおよび各 LSR プールでサポートされる同時アクセスの数を決定します。

VSAM ストリングを指定する必要があります。ストリングは、VSAM データ・セットに対する要求で、データ・セット内での位置決めを要求します。指定された各ストリングにより、作成される VSAM 制御ブロック (プレースホルダーを含む) の数が決定されます。

特定のファイルについてストリングの数を決める場合は、同時タスクの最大数を考慮してください。CICS コマンド・レベルでは、複数の要求を特定タスクの特定のデータ・セットに対して未解決とすることができないため、それ以上の同時要求にストリングを使用しても意味がありません。

詳しくは、[166 ページの『LSR および NSR のバッファおよびストリングの数』](#)を参照してください。

効果

LSR では、以下の効果による大きな利点が得られます。

- バッファとストリングを共用するため、仮想記憶をより有効に使用する。
- バッファ検索の向上によるパフォーマンスの改善により、入出力操作を削減する。
- ストレージに存在する CI のコピーは 1 つだけとなり、読み取り安全性が向上する。
- 使用中ファイルへの割り振りバッファが増し、参照回数の多い索引の制御間隔がバッファに保持されることによる自己調整。
- 同期ファイル要求および UPAD 出口の使用。LSR ファイルの CA および CI 分割により、サブタスクまたはメインタスクのどちらも待機することはありません。VSAM は、物理的入出力を待ちながら UPAD 出口を受け取り、CA/CI 分割の間、他の CICS 作業の処理が続行されます。

NSR ファイルのファイル制御要求は非同期式に行われますが、この場合でも、CICS メインタスクまたはサブタスクが分割中に停止します。

- トランザクション分離のサポート。

NSR では、以下の効果が得られます。

- 特定のデータ・セットを優先してチューニング。
- 順次操作のパフォーマンスを向上。

推奨

以下のいずれかの状態がある場合を除いて、すべての VSAM データ・セットで LSR を使用してください。

- ファイルがアクティブでありながら、ファイルが大きいなどの理由で、検索を行う機会がない。
- 追加の索引バッファの割り振りによってハイパフォーマンスが要求される。
- 追加のデータ・バッファによって高速な順次ブラウズまたは大量挿入が要求される。
- ファイルに制御域 (CA) 分割が想定され、CA 分割の高速化のため、追加のデータ・バッファが割り振られる。

LSR プールが 1 つだけの場合、ストリングを競合している状態のときは、同じプールを使用して特定のデータ・セットを他から分離することができません。分離できるのは、固有の CI サイズを指定して、バッファを競合する場合だけです。一般的に、1 つの大きなプールを使用して実行することにより、自己調整作業は増えます。複数のプールを使用することによって、使用中ファイルをその他のファイルより分離したり、ハイパフォーマンスのグループに追加のバッファを割り振ることができます。非常にアクティブなファイルは、NSR を使用する代わりに LSR サブプール内の唯一のファイルとして設定することにより、

正常にバッファ検索される機会が増え、入出力も減らすことができます。また、複数のプールを使用することにより、プールごとの 255 スtringの制限を緩和することもできます。

制限

同じ基本データ・セットのすべてのファイル (ファイル定義に DSNSHARING(MODIFYREQS) が指定された読み取り専用ファイルを除く) は、同じ LSR プールを使用するか、すべて NSR プールを使用する必要があります。

SERVREQ=REUSE ファイルは、LSR を使用できません。

LSR および NSR のバッファおよびストリングの数

バッファおよびストリングの数は、各ファイルに LSR または NSR のどちらを使用するかを決定するのに影響を与えることがあります。

LSR および NSR のバッファ数

VSAM によるバッファの割り振りおよび共用の方法には、LSR と NSR の間に次のようないくつかの重要な相違があります。

LSR

CICS で LSR パラメーターを計算する方法は簡単ですが、LSR プールを必要とする最初のファイルを開くときに、プールを作成するための追加の処理が発生します。CICS で LSR プールを計算するには、以下の要因を考慮してください。

- CICS は、プールを使用するよう指定されたすべてのファイルについて VSAM カタログを読み取る必要があります。
- CICS が計算を実行する時点で、関連するデータ・セットがマイグレーション済みである場合には、処理が増えます。CICS が LSR プールに関連した各データ・セットの VSAM カタログを読み取ることができるようにするには、各データ・セットを再呼び出しする必要があります。
- 単一の再呼び出しにより再呼び出しを行うタスクに大きく遅延が生じるだけでなく、同期操作によって、同じ TCB で CICS が実行する他のアクティビティが遅れる結果となります。

こうした遅延は、CICS データ・セットがマイグレーションされないように SMS ストレージ・クラスおよびマイグレーション・ポリシーを設計することにより、避けることができます。データ・セットのマイグレーション基準の設定については、「[z/OS DFSMSHsm ストレージ管理](#)」を参照してください。

CICS は、再呼び出しが必要な場合、情報メッセージ DHFC0989 を出力して、以降の遅延がエラー状態でないことを伝えます。

- CICS により計算される LSR プールは、各バッファごとに実サイズを指定して調整することはできません。
- LSR では、ストリングにも特定のファイルやデータ・セットにもバッファの事前割り当ては行われません。VSAM は、バッファを再使用する必要がある場合、参照頻度が最も低いバッファを選択します。ストリングは常にすべてのデータ・セット間で共用されます。LSR を使用する場合、ディスクへの読み取りを発行する前に、まず VSAM はバッファをスキャンして、必要な制御間隔が既にストレージ内に存在するかどうかをチェックします。存在する場合は、読み取りを発行する必要がないことがあります。このバッファの検索により、入出力を大幅に削減できます。
- LSR ファイルは、バッファの共通プールおよびストリングの共通プール、つまり、入出力操作をサポートする制御ブロックを共用します。その他の制御ブロックはファイルを定義し、各ファイルまたはデータ・セットに固有です。

LSR プールのサイズを変更する場合は、変更を行う前と行った後に CICS 統計を参照してください。これらの統計には、バッファの検索によって結果を得られた VSAM 読み取りの比率が変化したかどうかを示されます。

データ・バッファと索引バッファを別々に保持することをお勧めします。LSRPOOL を定義する場合、その LSRPOOL 定義によってデータ・バッファと索引バッファを別個に定義できます。CICS に LSRPOOL の作成を許可すると、データ・バッファと索引バッファは分離されません。

適切なサイズのバッファを指定してください。必要なサイズのバッファが存在しない場合、VSAM は次に大きいバッファ・サイズを使用します。

NSR

- ファイルの **STRINGS** パラメーターで指定された同時アクセスをサポートするために、各ファイルに対して十分なバッファを用意する必要があります。実際に、VSAM では NSR に対してこの要件を強制しています。
- ファイル定義の **DATABUFFERS** パラメーターおよび **INDEXBUFFERS** パラメーターを使用して、NSR のデータ・バッファおよび索引バッファの数を指定します。十分な索引バッファを指定することが重要です。KSDS が 1 つだけの制御域で構成されており、したがって、索引 CI が 1 つだけの場合は、**STRINGS** に等しい最小限の索引バッファ数で十分です。しかし、KSDS がこの値よりも大きい場合には、すべてのストリングで最低でも最上位の索引バッファを共用できるよう、少なくとも 1 つは追加の索引バッファを指定する必要があります。さらに索引バッファを増やすと、索引入出力がある程度削減されます。
- **DATABUFFERS** は最小値となる **STRINGS + 1** に設定します。ただし、順次操作で入出力のオーバーラップおよびチェーニングを有効にすることが目的である場合、あるいは CA 分割を高速化するために追加バッファを提供する必要がある場合を除きます。
- ファイルがベースへの代替索引パスである場合は、代替索引とベースのバッファに使用される **INDEXBUFFERS** (ベースが KSDS の場合) と **DATABUFFERS** の設定を同じにします (170 ページの『CICS による LSR プール・パラメーターの計算』を参照してください)。NSR では、データ・バッファの最小数は **STRNO + 1** で、最小の索引バッファ数 (KSDS および代替索引パスの場合) は **STRNO** です。各ストリングには、1 つのデータ・バッファと 1 つの索引バッファが事前に割り振られ、1 つのデータ・バッファが CA 分割用に予約されます。追加のデータ・バッファがある場合、これらのバッファは最初の順次操作に割り当てられます。また、これらのバッファは、チェーニングされた入出力操作を許可することにより、VSAM CA 分割の高速化にも使用できます。追加の索引バッファがある場合は、それらがストリング間で共用され、上位索引レコードの保持に使用されるため、物理的入出力が減る可能性があります。

注: 必ず、デッドロックを回避するようにトランザクションを設計し、プログラムしてください。詳しくは、[トランザクション・デッドロック](#)を参照してください。

ストリング数

VSAM では、並行したファイル操作ごとに 1 つ以上のストリングを必要とします。更新以外の要求 (READ または BROWSE など) の場合、ベースを使用するアクセスは 1 つのストリングを必要とします。代替索引を使用するアクセスには、2 つのストリングが必要です (1 つは代替索引上の位置を保持するため、もう 1 つは基本データ・セット上の位置を保持するために必要です)。アップグレード・セットを含まない更新要求の場合でも、ベースに 1 つのストリング、パスに 2 つのストリングを必要とします。アップグレード・セットが含まれる更新要求の場合、ベースに $1+n$ 個のストリングが必要であり、パスに $2+n$ 個のストリングが必要です。ここで、 n は、アップグレード・セット内のメンバーの数です。VSAM では、位置を保持するために、アップグレード・セット・メンバーごとに 1 つのストリングが必要です。同時要求ごとに、VSAM はアップグレード・セットの処理に必要な n 個のストリングを再使用できます。アップグレード・セットは順次に更新されるからです。

直接読み取りのような単純な操作では、1 つ以上のストリングが即時に解放されます。しかし、更新読み取り、大量挿入、またはブラウズ要求では、対応する更新、アンロック、またはブラウズ終了要求が実行されるまで、1 つ以上のストリングが保持されます。

CICS と VSAM による **STRNO** パラメーターの解釈は、コンテキストに応じて異なります。

- LSR プール定義 (LSRPOOL) における同等の **STRINGS** パラメーターは、VSAM BLDVRP マクロの **STRNO** パラメーターと同じ意味を持ちます。これはつまり、リソース・プールに割り振られるストリングの絶対数を表します。LSR プールに基本データ・セットのみ含まれる場合を除き、処理可能な同時要求の数は、指定された **STRINGS** 値よりも小さくなります。
- ファイル定義における同等の **STRINGS** パラメーターは、NSR ファイルの VSAM ACB における **STRNO** パラメーターと同じ意味を持ちます。つまり、処理可能な未解決 VSAM の同時要求の実際の数です。代替索引パスまたはアップグレード・セットが使用される場合、これをサポートするため VSAM が割り振るストリングの実際数は、指定された **STRINGS** 値よりも大きくなる場合があります。

LSR の場合、ストリングの正確な数を指定するか、CICS で数を計算することが可能です。LSR プール定義に指定された数は、プール内のストリングの実際の数です。CICS でストリングの数を計算する場合、その数は RDO ファイル定義のプールの **STRINGS** から派生します。これは、NSR の場合と同様に、このプールを同時要求の実際の数として解釈します。

サポートする必要がある同時の読み取り、ブラウズ、更新、大量挿入の要求などの数を決定する必要があります。

ファイルへのアクセスが、ブラウズを行わない読み取り専用の場合、多数のストリングは必要ありません。ストリングは 1 つだけで十分なこともあります。読み取り操作では、要求の期間だけ VSAM ストリングを保持しますが、同じ CI に対する更新操作が完了するのを待つ必要がある場合があります。

一般に、ブラウズまたは更新を使用する場合、**STRINGS** は最初に 2 または 3 に設定し、CICS ファイル統計を定期的にチェックして、発生する wait-on ストリングの比率を確認する必要があります。通常は、ファイル・アクセスの最大 5% までの wait-on ストリングを許容できると考えられます。NSR ファイルの場合、wait-on ストリングをゼロのままにしないでください。

CICS は、ファイルおよび LSR プールの両方でストリング使用を管理します。LSR または NSR のいずれを使用する場合でも、CICS はファイルごとに同時 VSAM 要求の数を、ファイル定義に指定された **STRINGS** に制限します。また、CICS は、LSR ごとにも、プール内のストリングで処理可能な数より多くの同時要求が VSAM に対して行われないう制限します。更新時のアップグレード・セットの処理に追加のストリングが必要とされる場合、CICS は先に read-for-update 時に追加ストリングを予約しておきます。使用可能なファイルまたは LSR プール・ストリングが十分にない場合、要求タスクはそれらが解放されるまで待機します。

CICS モニター機能を使用すると、各ユーザー・タスクの VSAM ストリング待ち時間のパフォーマンス・データを得ることができます。DFHFILE グループ内のパフォーマンス・データ・フィールド 427、FCVSWTT により、タスクが VSAM ストリングを待機した経過時間がわかります。CICS LSR プール統計からは、ストリング数、ストリングを待機した要求の数、および同時にアクティブであったストリングの最大数がわかります。

特定のファイルについてストリングの数を決める場合は、同時タスクの最大数を考慮してください。CICS コマンド・レベルでは、複数の要求を特定タスクの特定のデータ・セットに対して未解決とすることができないため、それ以上の同時要求にストリングを使用しても意味がありません。

異なるタイプのタスクにストリングを分散する場合、トランザクション・クラスも役立ちます。トランザクション・クラスの制限を使用すると、個別のタイプの VSAM 要求を発行するトランザクションを制御できます。また、VSAM ストリングを使用できるタスク・タイプの数制限することにより、ストリングのサブセットを他の用途で使用できるように残しておくことができます。

すべてのプレースホルダー (置き換え) 制御ブロックは、プールを共用するすべてのデータ・セットに関連した最大のキーに十分な長さのフィールドを含む必要があります。大きなキー (基本または代替) を持つ 1 つの非アクティブ・ファイルを、多数のストリングを含む LSR プールに割り当てると、非常に多くのストレージが使用される場合があります。

LSR の VSAM 仕様

LSR に対する VSAM バッファ割り振りおよびストリング設定を定義します。LSR のリソース百分位数および最大キー長を指定します。

LSR の VSAM バッファ割り振りの定義

ローカル共用リソース (LSR) を使用するファイルの場合、使用するバッファの数は、ファイルによって明示的には指定されません。ファイルは、LSR プールの適当なサイズのバッファを共用します。プール内のバッファ数は、CICS システム定義データ・セット (CSD) のファイル定義内の **BUFFERS** パラメーターを使用して明示的に指定することも、CICS に計算させることもできます。

CICS ファイル制御内の VSAM LSR ファイルを使用する CICS システムでは、**BUFFERS** パラメーターを使用してください。これにより、LSR プールに固有のバッファを正確に定義することができます。バッファの数は、パフォーマンスに大きく影響する場合があります。バッファを使用することにより、複数の並行操作が可能です (対応した数の VSAM ストリングがある場合)。また、バッファ検索が成功する可能性が増し、物理的な入出力操作を減らすことができます。

最適なバッファ割り振りを行うには、検索による入出力の節約を増やすことと、実記憶域所要量を増やすことの間でのトレードオフが必要です。この最適値は、索引に使用するバッファとデータに使用するバッファで異なります。LSR の最適なバッファ割り振りは、同じファイルで NSR を使用する場合のバッファ割り振りよりも小さくなります。

これらのパラメーターによる影響は、トランザクションの応答時間およびデータ・セットとページングの入出力率によってモニターできます。この設定値は、ファイルおよび LSRPOOL 統計の両方に影響します。CICS ファイル統計には、VSAM データ・セットのデータ・セット・アクティビティが表示されます。VSAM カタログおよび RMF には、データ・セット・アクティビティ、入出力の競合、スペース使用、および制御間隔 (CI) サイズを示すことができます。

注：z/OS V2.2 以降、VSAM には、特定の VSAM 要求で使用可能なバッファがない場合に LSR プール用の追加のバッファを割り振る動的バッファ追加機能があります。CICS にとっては、LSR プールの即時拡張を許可するよりも要求を再試行するほうが望ましいため、CICS LSR プールに対する動的バッファ追加は有効化されません。

CICS には、LSR バッファの割り振りの調整に使用できる統計メトリックとモニター・メトリックが用意されています。バッファ待機の結果としてタスクがキューに入れられた回数については、[ファイルに関する統計](#)を参照してください。強制的に待機させられているトランザクションと待機時間を調べるには、[例外モニター・データ](#)を参照してください。

LSR の VSAM スtring 設定の定義

STRINGS パラメーターは、String の数の決定に使用され、LSR プールに対して可能な並行操作の数が決まります (使用可能なバッファがあることが前提)。**STRINGS** パラメーターは、VSAM データ・セットを使用する CICS システムで使用できます。

String の数は、CSD のファイル定義の **STRNO** パラメーターにより定義され、これにより特定のファイルに対する並行アクティビティが制限されます。

LSR を使用するファイルに関連した **STRINGS** パラメーターは以下の影響があります。

- 特定のファイルに対して可能な同時要求の数を指定します。
- CICS によって、LSR プールに対する String およびバッファの数の計算に使用されます。
- VSAM LSR プールの **STRINGS** 値として使用されます。
- CICS により、VSAM short-on-strings 条件を避けるため、プールへの要求の制限に使用されます (要求ごとに必要な String の数は CICS が計算することに注意してください)。
- 1 より大きい値は、書き込みモードで排他使用される ESDS ファイルのパフォーマンスに悪影響を与えます。String の数を 1 より大きくすると、競合の排他制御を解決するコストが、String を待つコストを上回ります。排他制御が戻されるたびに、GETMAIN がメッセージ領域に発行され、VSAM への 2 番目の呼び出しが行われて、制御間隔の所有者が取得されます。

最大で 255 個の String がプールごとに許可されます。**STRINGS** パラメーターによって、各ファイル入力の応答時間が変化します。CICS LSR プール統計からは、String 数、String を待機した要求の数、および同時にアクティブであった String の最大数がわかります。DFHFILE グループ内の CICS パフォーマンス・データ・フィールド 427、FCVSWTT により、各ユーザー・タスクが VSAM String を待機した経過時間がわかります。

CICS 統計の String 数を検討することにより、String には 2 レベルのチェックが使用可能であることがわかります。1 つはデータ・セット・レベルで (DFHSTUP レポートの [ファイル制御統計](#)を参照)、もう 1 つは共用リソース・プール・レベルです (DFHSTUP レポートの [LSR プール統計](#)を参照)。

LSR の最大キー長の指定

CSD のファイル定義の **KEYLENGTH** パラメーター、または LSR プール定義の **MAXKEYLENGTH** パラメーターは、LSR プールで使用される最大のキー・サイズを指定します。**KEYLENGTH** パラメーターは、VSAM データ・セットを使用する CICS システムで使用できます。CSD のファイル定義で **KEYLENGTH** パラメーターを使用して、最大キー長を明示的に指定します。あるいは、CICS に VSAM カタログから最大キー長を判別させるようにします。

KEYLENGTH パラメーターにより、LSR プールで使用可能な最長キーのスペースを持つプレースホルダー制御ブロックが作成されます。指定された **KEYLENGTH** が小さすぎると、これより長いキー長のファイルに対する要求が拒否されます。LSR プールを使用するファイルの最大キーの長さ以上に常になるように、キー長を設定してください。

LSR のリソース百分位数の指定

LSR プール定義の **SHARELIMIT** パラメーターは、CICS が計算する値に適用するバッファーおよびストリングの割合を指定します。 **SHARELIMIT** パラメーターは、VSAM データ・セットを使用する CICS システムで使用できます。 **SHARELIMIT** パラメーターは、LSR プール定義で指定されます。

プールに対して **BUFFERS** パラメーターと **STRINGS** パラメーターの両方が指定されている場合、**SHARELIMIT** パラメーターは無視されます。 **SHARELIMIT** は、LSR プールの初期設定時に割り振られるファイルに対してのみ適用することができ、プール内の最初のファイルが開かれるときに適用されます。したがって、LSR プールには、常に 10 進 **STRINGS** と **BUFFERS** を指定することをお勧めします。

CICS による LSR プール・パラメーターの計算

プールに対して LSR パラメーターを指定していない場合、CICS によって必要なバッファーおよびストリングが計算されます。

この計算を行うために、CICS はインストールされたすべてのファイル・リソース定義をスキャンし、そのプールを使用するように指定されたファイルを見つけます。それぞれのファイルで、以下の値が使用されます。

- CICS ファイル・リソース定義から:
 - **STRINGS** パラメーターに指定されたストリングの数
- VSAM カタログから:
 - これらの各ファイルの索引レベル
 - 制御間隔 (CI) サイズ
 - ベースのキー長、パス (代替索引パスからアクセスする場合)、およびアップグレード・セットの代替索引

ユーザーがバッファーのみ、またはストリングのみを指定した場合、CICS は、バッファーおよびストリングのうち、指定されていない方を計算します。

以下の情報を参考に、必要なバッファー数を計算できます。特定のファイルでは、複数のバッファー・サイズを必要とする場合があります。各ファイルについて、CICS は、以下のコンポーネントに必要なバッファー・サイズを算定します。

- データ・コンポーネント
- 索引コンポーネント (KSDS の場合)
- 代替索引のデータ・コンポーネントおよび索引コンポーネント (代替索引パスの場合)
- アップグレード・セット (存在する場合) 内の各代替索引のデータ・コンポーネントおよび索引コンポーネント

各ファイルのバッファー数は、次のように計算されます。

- ベースおよび代替索引のデータ・コンポーネントについて = (ファイル・リソース定義項目の **STRINGS**) + 1
- ベースおよび代替索引の索引コンポーネントについて = (ファイル・リソース定義項目の **STRINGS**) + (索引内のレベルの数) - 1
- アップグレード・セット内の各代替索引のデータ・コンポーネントおよび索引コンポーネントそれぞれについて、バッファー 1 つ

プールを使用するすべてのファイルについてこの計算が完了すると、各サイズのバッファーの総数が、さらに次のように計算されます。

- 数値が、50% または LSRPOOL 定義の **SHARELIMIT** で指定されたパーセンテージのどちらかまで削減されます。 **SHARELIMIT** パラメーターが優先されます。
- バッファ数値の最小値は 3 なので、それより小さい場合は 3 に増やされます。
- 数値が、最も近い 4 KB 境界に切り上げられます。

CICS は、ストリングの数を計算するために、各ファイルの同時要求の処理に必要なストリングの数を、以下の値の合計として算定します。

- 基本部分として **STRINGS** パラメーター値
- 代替索引の **STRINGS** パラメーター値 (代替索引パスの場合)
- アップグレード・セットがある場合は、ストリングの個数である n (n は、アップグレード・セットのメンバーの数)。

注: LSR プールが CICS によって計算され、データ・セットが階層ストレージ・マネージャー (HSM) によりアーカイブ済みである場合、LSR プールを必要とする最初のファイルを開くときに、データ・セットが 1 つずつ必要になるため、CICS システムの起動時間がかなり長引くことがあります。CICS は、必要なカタログ情報を取得しますが、データベースは開きません。このため、データベースは実際にはアーカイブされたままとなります。この問題は、領域を再度開始すると再発し、データ・セットが開かれるまで解決されません。

すべてのファイルのストリング数が集計されると、バッファの総数はさらに以下のように計算されます。

- 合計が、50% または LSRPOOL 定義の **SHARELIMIT** パラメーターで指定されたパーセンテージのどちらかまで削減されます。 **SHARELIMIT** パラメーターが優先されます。
- 合計が、255 (VSAM によってプールに許可されるストリングの最大数) まで削減されます。
- 合計が、特定のファイルに指定された最大 **STRINGS** 値まで増やされます。

CICS により計算されるパラメーターは、CICS 統計に示されます。

注: z/OS V2.2 以降、VSAM には、特定の VSAM 要求で使用可能なバッファがない場合に LSR プール用の追加のバッファを割り振る動的バッファ追加機能があります。CICS にとっては、LSR プールの即時拡張を許可するよりも要求を再試行するほうが望ましいため、CICS LSR プールに対する動的バッファ追加は有効化されません。

CICS には、LSR バッファの割り振りの調整に使用できる統計メトリックとモニター・メトリックが用意されています。バッファ待機の結果としてタスクがキューに入れられた回数については、ファイルに関する統計を参照してください。強制的に待機させられているトランザクションと待機時間を調べるには、例外モニター・データを参照してください。

RLS モードから LSR モードへのデータ・セットの切り替え

データ・セットを RLS モードから非 RLS モードに切り替える必要がある場合があります (例えば、バッチ更新中に読み取り専用 LSR モードに切り替える場合)。この切り替えの結果、明示的に定義されず CICS がデフォルト値を使用して作成する LSR プールで、プールの作成後に LSR モードに切り替えられたファイルをサポートするのに十分なリソースがない状態になる可能性があります。

適切なリソースの不足によりファイルのオープンに障害が発生するのを避けるには、ユーザーは、CICS がデフォルト値を使用して LSR プールのサイズを計算する際に RLS モードで開かれたファイルを含めるように指定することができます。CICS が計算する値を使用して作成される LSR プールに RLSACCESS(YES) で定義されたファイルを組み込むには、このシステム初期設定パラメーターに RLSTOLSR=YES (RLSTOLSR=NO がデフォルト) を指定します。

このパラメーターについて詳しくは、RLSTOLSR システム初期設定パラメーター を参照してください。

データ・セット名の共用

データ・セット名 (DSN) の共用は、すべての VSAM データ・セットのデフォルトです。これは VSAM ACB で MACRF=DSN として指定されています。これにより VSAM は、同じ基本データ・セット・クラスターに (パスとして、または基本のクラスターに直接) 関連するすべてのファイルで必要とされるストリングとバッファ用の単一の制御ブロック構造を作成します。VSAM は、2 番目以降のファイルのオープン時に接

続を行います。DSN 共用が指定されている場合にのみ、VSAM は同じデータ・セットを処理していることを認識します。

この単一構造は、以下の利点を提供します。

- 1 つの VSAM データ・セットを更新する複数のアクセス制御ブロック (ACB) に対して VSAM 更新の整合性を保つ。
- CICS 領域内で複数の更新ブロックを許可したまま、VSAM の共用オプション 1 または 2 を使用できる。
- 仮想記憶域を節約する。

NSR と LSR の両方を使用するファイルの場合、DSN 共用はデフォルトです。このデフォルトの唯一の例外として、読み取り専用 (*READ=YES* または *BROWSE=YES*) として指定され、ファイル・リソース定義に *DSNSHARING(MODIFYREQS)* を持つファイルを開く場合があります。CICS のこのオプションにより、DSN 共用を抑止することで、特定のファイル (インストールされたファイル・リソース定義により表される) を、異なる LSR プール内または NSR 内の同じデータ・セットの他のユーザーから分離することができます。CICS は、更新、追加、または削除オプションを持つファイルに対しては、このパラメーターを無視します。これは、2 つのファイル制御ファイル項目が同じデータ・セットを同時に更新していた場合、VSAM が更新の整合性を保つことができないためです。

NSRGROUP パラメーターは、DSN 共用に関連しています。これは、同じ VSAM 基本データ・セットを参照するファイル・リソース定義をグループ化するために使用されます。NSRGROUP=name は、LSR を使用するデータ・セットには影響を与えません。

DSN 共用 NSR ファイルのグループの最初のメンバーを開くときに、CICS は VSAM に対し、グループ内のすべてのファイル項目に割り振るストリングの総数を、ACB 内の **BSTRNO** 値で指定する必要があります。VSAM は、開かれる最初のデータ・セットがパスであるか、ベースであるかに関係なく、この時点で制御ブロック構造を作成します。CICS は、同じ **NSRGROUP** パラメーターを共用するすべてのファイルの **STRINGS** 値を加算して、オープン時に使用される **BSTRNO** の値を計算します。

NSRGROUP パラメーターを指定しない場合、以後の処理にとって不十分なストリング数を持つ VSAM 制御ブロック構造が作成されることがあります。パフォーマンス上の理由から、この構造は避けてください。このような場合、VSAM は動的ストリング追加機能呼び出し、必要な数のストリング用の追加の制御ブロックを提供します。この追加のストレージは、CICS の実行が終了するまで解放されません。

代替索引の考慮事項

UPGRADE 属性を使用して定義された各代替索引について、VSAM は基本クラスターが更新されると自動的に代替索引をアップグレードします。

NSR の場合、VSAM は基本クラスターに関連付けられたバッファの特殊なセットを使用します。このセットは、2 つのデータ・バッファと 1 つの索引バッファで構成され、基本クラスターに関連付けられた各代替索引に対して順次に使用されます。VSAM 操作のこの部分は、調整できません。

LSR の場合、VSAM は適切なサブプールからバッファを使用します。

代替索引がアップグレード・セット内に存在することを VSAM に指定するときは注意が必要です。新しいレコードが追加されるか、既存のレコードが削除されるか、あるいは変更された属性キーを使用してレコードが更新されるたびに、VSAM はアップグレード・セット内の代替索引を更新します。この更新には、追加の処理と追加の入出力操作が含まれます。

追加の物理的入出力が生じる状況

多数の物理的入出力操作が生じ、応答時間および関連するプロセッサのパス長さの両方に影響を与える可能性がある状況には、以下のものが含まれます。

- KSDS が SHROPT 4 で定義されている場合、直接読み取りはすべて、索引バッファとデータ・バッファの両方のリフレッシュを引き起こします (最新のコピーを確保するため)。
- CICS が ENDREQ を発行することになるシーケンスはすべて、その操作に関連したすべてのデータ・バッファを無効にします。この状態が発生する可能性があるのは、*get-update* (以降の更新なしの)、ブラウズ (*no-record-found* 応答を伴うブラウズ開始でも)、*mass-insert*、またはプログラムからの *get-locate* を終了する場合です。操作がプログラムによって明示的に終了されない場合、CICS は同期点またはタスク終了時に操作を終了します。

- ストリング数よりもデータ・バッファ数が多い場合、ブラウズ開始により、少なくとも半数のバッファが即時にチェンニングされた入出力に参加します。ブラウズが短時間の場合、追加の入出力は不要です。

その他の VSAM 定義パラメーター

フリー・スペース・パラメーターは、制御間隔 (CI) 分割および制御域 (CA) 分割の数の削減に役立つため、慎重に選択してください。レコードが VSAM データ・セット全体に挿入される場合、各 CI にフリー・スペースを含めるのが適切です。挿入が集中する場合は、各 CA にフリー・スペースが必要です。すべての挿入がファイル内の少数の位置で行われる場合は、VSAM に CA を分割させることができ、フリー・スペースを指定する必要はまったくありません。

VSAM データ・セットの末尾にレコードを追加しても、CI や CA の分割は行われません。末尾以外の箇所に順次レコードを追加すると、分割が行われます。低い値のダミー・キーを持つ空ファイルは、分割が減る傾向があります。高い値のキーでは、分割の数が増えます。

NSR の VSAM 仕様

NSR の VSAM ストリング設定の定義および NSR の VSAM バッファ割り振りの定義を行います。

NSR の VSAM バッファ割り振りの定義

非共用リソース (NSR) を使用するファイルの場合、**INDEXBUFFERS** および **DATABUFFERS** パラメーターにより VSAM 索引バッファおよびデータ・バッファを定義します。

INDEXBUFFERS および **DATABUFFERS** パラメーターは、CSD のファイル定義で定義されます。これらは、VSAM ACB パラメーターに正確に対応し、**INDEXBUFFERS** は索引バッファの数、**DATABUFFERS** はデータ・バッファの数となります。

• 効果

バッファの数は、パフォーマンスに大きく影響する場合があります。バッファを使用することにより、複数の並行操作が可能で (対応した数の VSAM ストリングがある場合)、効率よく順次操作と制御域 (CA) 分割を行うことができます。上位索引レコードに追加バッファを用意することにより、物理的入出力操作を減らせます。

16 MB 境界を超えるバッファ割り振りは、ほとんどの CICS システムにおける仮想記憶要件の大きな割合を占めます。

• 制限

これらのパラメーターは、VSAM データ・セットに指定されたストリングに対して十分でない場合には、VSAM によって指定変更が可能です。最大仕様は 255 です。これよりも大きな値を指定した場合は、自動的に 255 に減らされます。DD ステートメントで **AMP** 属性を指定することによって、VSAM ストリングおよびバッファの指定変更を行ってはなりません。

• 制限

これらのパラメーターによる影響は、トランザクションの応答時間およびデータ・セットとページングの入出力率によってモニターできます。CICS ファイル統計には、VSAM データ・セットに対するデータ・セットのアクティビティが示されます。VSAM カタログおよび RMF には、データ・セット・アクティビティ、入出力の競合、スペース使用、および制御間隔 (CI) サイズを示すことができます。

NSR の VSAM ストリング設定の定義

STRINGS は、ファイルおよびファイルが関連する VSAM 基本クラスターに対する並行操作の数の決定に使用します。

STRINGS パラメーターは、CICS ファイル制御に VSAM NSR ファイルを使用する CICS システムで使用します。

ストリングの数は、CSD の CICS ファイル定義の **STRINGS** パラメーターにより定義されます。これは、ACB 内の VSAM パラメーターに対応します。ただし、基本ファイルが VSAM データ・セットに対して初めて開かれる場合を除きます。この場合は、CICS で累算された **BSTRNO** 値が、ACB の **STRNO** 値として使用されます。

- 効果

NSR を使用するファイルの **STRINGS** パラメーターには以下の影響があります。

- 特定のファイルに対して可能な非同期の同時要求の数を指定します。
- VSAM ACB で **STRINGS** として使用されます。
- これは、**BASE** パラメーターと合わせて、VSAM **BSTRNO** 値を計算するために使用されます。
- 1 より大きい値は、書き込みモードで排他使用される ESDS ファイルのパフォーマンスに悪影響を与えます。ストリング数が 1 より大きいと、各ストリングについてバッファーを無効にするコストが、ストリングを待つコストを上回り、VSAM EXCP 要求の数が大きく増加する可能性があります。

ストリングは、ほとんどの CICS システムにおける仮想記憶要件の大きな割合を占めます。CICS では、このストレージは 16 MB 境界を超えて置かれます。

- 制限

最大で 255 個のストリングを ACB の **STRNO** または **BSTRNO** 値として使用することができます。

- モニター

STRINGS パラメーターによって、応答時間が変化します。DFHFILE グループ内の CICS パフォーマンス・データ・フィールド 427、FCVSWTT により、各ユーザー・タスクが VSAM ストリングを待機した経過時間がわかります。CICS LSR プール統計からは、ストリング数、ストリングを待機した要求の数、および同時にアクティブであったストリングの最大数がわかります。RMF には、DASD サブシステムの入出力の競合を示すことができます。

VSAM サブタスキングの使用

オブションの並行 (CO) モード TCB は、VSAM 要求などの他の CICS アクティビティーと並列に安全に実行できるプロセスに使用されます。CO TCB が存在すべきかどうかを指定する数値 (0 と 1) を持つように、SIT キーワード **SUBTSKS** が定義されています。システム初期設定パラメーター **SUBTSKS=1** は、サブタスキングを使用するよう定義します。

サブタスキングは、VSAM を使用する CICS システムで活用できます。

サブタスキングは、シングル・プロセッサによる制限があるが MVS イメージ内の他のプロセッサに予備能力のある領域のマルチプロセッシング・システムでのみ使用してください。他の状況で使用した場合、複数タスクのディスパッチングによって、スループットが低下する場合があります。

効果

サブタスクは、マルチプロセッサにおける単一の CICS システムの最大スループットを増すことを目的としています。ただし、タスク間の通信により、プロセッサ使用率の合計は増加します。

サブタスクで入出力が行われると、NSR プールで制御間隔 (CI) または制御域 (CA) 分割など、CICS 領域の停止の原因になるような応答時間の延長があった場合、追加の TCB のみが停止します。これにより、ファイルに多数の CA 分割がある領域のスループットが高まりますが、サブタスクの使用に関連した追加処理について慎重に評価する必要があります。

SUBTSKS=1 システム初期設定パラメーターを指定した場合、次のようなサブタスクへの影響が見られます。

- KSDS に対するすべての非 RLS VSAM ファイル制御 WRITE 要求がサブタスク化されます。
- 他のファイル制御要求は、すべてサブタスク化されません。
- 補助一時記憶域または区画内一時データの要求がサブタスク化されます。
- リソース・セキュリティ検査要求は、CICS メイン TCB (準再入可能モード) が約 70% のアクティビティーを超えるとサブタスク化されます。

制限

サブタスキングは、余分なサブタスク実行に追加のプロセッサ・サイクルが必要なことから、マルチプロセッサ MVS イメージにおいてのみスループットを向上できます。このため、この機能はユニプロセッ

サー (UP) では使用しないようお勧めします。これは、予備のプロセッサ能力を持つ複合システムで 1 つのプロセッサの最大能力に達した領域、あるいは頻繁に CI または CA 分割が行われる NSR ファイルを持つ領域でのみ使用してください。

大量の VSAM データ・セット・アクティビティーを含まない領域では (特に更新アクティビティー)、VSAM サブタスキングの効果はありません。

サブタスキング処理間での競合とマルチプロセッサ使用の向上により、アプリケーション・タスク経過時間が増減する場合があります。タスク関連の DSA 占有もこれに応じて増減します。

推奨

SUBTSKS=1 を指定するのは、2 つ以上のプロセッサを持つ MVS イメージで CICS システムが実行されており、領域内の CICS メイン TCB によるプロセッサ・ピーク使用率が 1 つのプロセッサの約 70% を超え、CICS アドレス・スペース内の大量の入出力アクティビティーがサブタスキングに 適格である場合にのみ限定してください。

この環境では、2 番目のプロセッサの能力を使用して、VSAM データ・セット、補助一時記憶域、および区画内一時データの入出力スケジューリング・アクティビティーを実行できます。

この CICS 領域の最大のシステム・スループットは、入出力サブタスクを使用することにより向上が可能ですが、サブタスクとトランザクション処理が行われる MVS タスク間での通信に追加の処理を要します。この追加処理は、CICS 領域がスループットの限界に達したか、達しつつある場合でなければ、正当化されません。

1 つ以上の AOR にトランザクションの大部分を、または排他的にルーティングしている TOR には、サブタスキングに適格な入出力はほとんどありません。このため、サブタスキングの対象としてはふさわしくありません。

AOR が適した候補となるのは、大量の VSAM 入出力が、FOR に機能シッされるのではなく、AOR 内で実行される場合だけです。

多くの場合に大量の VSAM 入出力を行うビジーな FOR についてはサブタスキングを検討してください (ただし、VSAM データ・セットの DL/I 処理はサブタスキングされません)。

SIT に FCQRONLY=NO が設定された、ローカル VSAM LSR または RLS を使用するスレッド・セーフ・アプリケーションに対する VSAM サブタスキングは、通常は推奨されません。スレッド・セーフ・ファイル制御アプリケーションでは、複数の L8 または L9 TCB を使用することにより、パフォーマンス上の利点が大きくなります。

モニター

CICS ディスパッチャー・ドメイン統計には、[ディスパッチャー TCB モード・レポート](#) に示される TCB のモードに関する情報が含まれます。

CMF データおよび CICS トレースをすべて使用できます。

データ・テーブルの使用

データ・テーブルを使用することにより、16 MB 境界を超える仮想記憶に保持されるテーブル内のデータ・レコードの作成、保守、および高速アクセスが可能になります。これにより、DASD 入出力およびパス長さリソースを減らして、パフォーマンスを大きく向上できます。データ・テーブルからレコードを検索するためのパスの長さは、VSAM バッファ内既に存在するレコードを検索するためのパスの長さより短くなります。

データ・テーブルは、CEDx トランザクションの [DEFINE FILE](#) コマンドまたは DFHCSDUP ユーティリティー・プログラムを使用して定義できます。

効果

データ・テーブルを使用すると、次の効果があります。

- 初期データ・テーブルのロード操作後、すべてのユーザー管理および読み取り専用の CICS 管理データ・テーブル (CMT) に対して、DASD 入出力を排除できます。

- CMT の DASD 入出力の減少は、READ/WRITE の比率に依存します。この比率は、データ・テーブルの実装の前に、ソース・データ・セットで発生した READ 呼び出しと WRITE 呼び出しの数の比率です。これらの縮小は、データ・テーブルの READ ヒット率にも依存します。これは、ソース・データ・セットに送られた要求数に対する、テーブルで満たされた READ 呼び出しの数です。
- CICS ファイル制御のプロセッサ使用量は、最大で 70% 削減できます。この削減は、ファイル設計とアクティビティに依存し、ここに示したのは一般的なガイドラインに過ぎません。実際の結果は、インストールごとに異なります。

CMT については、ソース・データ・セットとデータ・テーブルの変更の同期が保証されます。ファイルがリカバリー可能な場合、必要な同期は、既存のレコード・ロックによって既に実施されています。ファイルがリカバリー不能な場合は、CICS レコードのロックは行われず、すべての更新要求に対して、代わりに注釈ストリング位置 (NSP) メカニズムが使用されます。この処置は、場合によっては、追加の VSAM ENDREQ 要求によるパフォーマンスの影響がわずかで済みます。

推奨

データ・テーブルは、ファイル定義の 2 つの RDO パラメーター (**TABLE** および **MAXNUMRECS**) によって定義されます。他に変更は不要です。

1 つか 2 つの候補を選択して始めてください。最初に、CMT のリカバリーの考慮事項を簡素化することができます。

READ と WRITE の比率が高い CMT を選択します。この情報は、VSAM LISTCAT ジョブを実行することにより、CICS LSRPOOL 統計で見ることができます ([LSR プール統計トピック ページ](#)を参照)。

READ INTO を使用します。READ SET は内部処理がやや多くなるからです。

実記憶使用量をモニターしてください。既にシステムに実記憶の制約がある場合は、大きいデータ・テーブルによりページイン率が増す可能性があり、これが CICS のシステム・パフォーマンスに悪影響を与えることがあります。RMF など、標準のパフォーマンス・ツールを使用して、実記憶およびページング率を監視してください。

CMT の候補として、完全キーによる直接読み取りの比率が高いファイルを選択してください。

再始動時にリカバリーの必要がない更新アクティビティの比率が多いファイルは、ユーザー管理データ・テーブルに適しています。

ユーザー管理データ・テーブルでは、グローバル・ユーザー出口 XDTRD を使用して、レコードの変更と選択の両方が行えます。この処置により、ユーザー管理データ・テーブルに、アプリケーションに関連した情報のみを含むことができます。

ストレージ分離が指定されている場合、データ・テーブルで必要とされる追加ストレージを使用して、CICS で発生するページングが増加しないようにしてください。

CMT として定義されたものと VSAM ファイルとして定義されたものの 2 つのオープン・ファイルが、基礎となる同じ VSAM 範囲を参照するという状態 (両方が同じデータ・セット名を参照するなど) を回避することを試行してください。この状態では、VSAM ファイルは CMT である場合とほぼ同様に処理されます。つまり、それは CMT の利点と欠点の両方を持つことになります。利点は、他のファイル用に作成されたテーブルの読み取りおよび参照処理が格段に速くなることです。

VSAM ファイルのパフォーマンス上の欠点は、以下のとおりです。

- 更新では、ファイルとテーブルの両方を更新する必要がある。
- VSAM ファイルがベースではなくパスを参照する (つまり代替キーを使用する) 場合、高速読み取りの利点が失われる。
- VSAM ファイルの要求が QR タスク制御ブロック (TCB) に常に切り替えられて、オープン TCB では処理されない。

モニター

データ・テーブルの効率を評価するため、パフォーマンス統計が収集されます。これは、標準の CICS ファイル統計で使用できる統計と合わせて収集されます。

以下の情報が記録されます。

- テーブルからの読み取り試行数
- 失敗した読み取り試行数
- データ・テーブルの割り振りバイト数
- データ・テーブルにロードされたレコード数
- テーブルへの追加の試行数
- ロード中または API 経由でのテーブルへの追加中にユーザー出口により拒否されたレコードの数
- テーブルがいっぱいで (レコードの最大数に達して) レコードの追加に失敗した試行数
- 再書き込み要求によるテーブル・レコードの更新の試行数
- テーブルからのレコード削除の試行数
- 前回のオープン時からテーブルのレコード数が達した最大値。

統計には、実行中の更新の存在によって生じるなどした明らかな矛盾が見られる場合があります。

カップリング・ファシリティ・データ・テーブルの使用

カップリング・ファシリティ・データ・テーブル (CFDT) からのデータの保管および検索に使用される API は、ユーザー管理データ・テーブルに使用されるファイル制御 API に基づいています。

CFDT は、多くの点で共用ユーザー管理データ・テーブルに似ています。共用データ・テーブルについては、[Introduction to shared data tables](#) を参照してください。

CFDT は、CICS 領域に対し FILE 定義で以下のパラメーターを使用して定義されます。

- **TABLE(CF)**
- **MAXNUMRECS(NOLIMIT***number***(1 から 99999999))**
- **CFDTPOOL***(pool_name)*
- **TABlename***(name)*
- **UPDATERMODEL (CONTENTION|LOCKING)**
- **LOAD(NO|YES)**

MAXNUMRECS は、CFDT が保持することのできるレコードの最大数を指定します。

CFDT を開く最初の CICS 領域がファイルの属性を決定します。正常に開かれると、これらの属性は、カップリング・ファシリティ・リスト構造のデータにより CFDT と関連を保ったままになります。このテーブルまたはカップリング・ファシリティ・リスト構造が CFDT サーバー・オペレーター・コマンドにより削除または変更されない場合は、属性は CICS および CFDT サーバーの再始動後にも残ります。CFDT を開こうとする他の CICS 領域は、例えば同じ更新モデルを使用するなどして、CFDT の一貫した定義を持つ必要があります。

CFDT サーバーは、カップリング・ファシリティのリスト構造およびこの構造に含まれるデータ・テーブルを制御します。[Coupling facility data table server parameters](#) に記載されたパラメーターは、初期の構造サイズ、構造エレメント・サイズ、およびエントリーとエレメントの比率の指定方法を記述します。

データは、UMT と異なり、MVS イメージ内のデータ・スペースには保持されず、CICS 領域によって管理されますが、カップリング・ファシリティのリスト構造内に保持されます。制御は CFDT サーバー領域間で共用されます。CFDT へのアクセスを要求する CICS 領域は、同じ MVS イメージで実行される CFDT サーバー環境との通信を、MVS 許可の仮想記憶間 (AXM) サーバー環境を使用して行います。CICS 一時記憶域サーバーでも、これと同じ手法が使用されます。

CFDT は、非公式の共用データに対して有効です。用途としては、シスプレックス全体にわたる共用スクラッチパッド、電話番号の索引テーブル、および顧客リストからの顧客サブセットの作成があります。共用データ・テーブル、共用一時記憶域、または RLS ファイルなど、この種の共用データの既存の方式と比較すると、CFDT には明らかな利点があります。

- 修正のため頻繁にデータにアクセスする場合、CFDT は、機能シッ プした UMT 要求または RLS ファイル使用と比べて、パフォーマンスに優れています。

- CFDT 保持データは、CICS トランザクション内でリカバリー可能です。構造のリカバリーはサポートされませんが、作業単位の障害、CICS 領域の障害、CFDT サーバー障害、または MVS の障害時に、CFDT レコードを回復できます (すなわち、障害の発生時に実行中であった作業単位による更新がバックアウトされます)。このような回復機能は、共用一時記憶域にはありません。

ロック・モデルおよびコンテンション・モデル

カップリング・ファシリティ・データ・テーブルには、コンテンション・モデルとロック・モデルの 2 つのモデルがあります。

ロック・モデル。 カップリング・ファシリティのリスト構造に保持されたレコードは、データを保持するカップリング・ファシリティのリスト構造エレメントに関連した付加属性領域の更新により、ロック済みとマークされます。最初のアクセスで、データがロック済みでないと判断されると、レコードのロックには、追加のカップリング・ファシリティ・アクセスによるロック設定が必要です。

ただし、更新に競合がある場合、以下のイベント・シーケンスに示すように、追加してカップリング・ファシリティのアクセスが必要になります。

1. ロック競合が発生した要求は、最初は拒否される。
2. 要求側は、ロックされたレコードの付加属性領域を変更して、これを対象とすることを示す。この領域が、ロック待機側に対して 2 番目に行われる追加のカップリング・ファシリティ・アクセスです。
3. ロック所有者は、更新を拒否している。これは、レコードの付加属性領域が変更されており、CICS 領域で再読み取りして更新を再試行する必要があるためです。これにより、2 度の追加のカップリング・ファシリティ・アクセスが生じます。
4. ロック所有者は、ロック解除の通知メッセージを送信する。ロックが異なるサーバーから要求されている場合は、カップリング・ファシリティのアクセスにより、他のサーバーに通知メッセージが書き込まれ、カップリング・ファシリティ・アクセスがこれを読み取ります。

コンテンション・モデル。 競合更新モデルでは、エントリー・バージョン番号を使用して変更を追跡します。エントリー・バージョン番号は、レコードを更新するたびに更新されます。この変更により、更新要求では、レポートのコピーを取得してから後でレコードが変更されていないかをチェックすることができます。

更新の競合が発生すると、追加のカップリング・ファシリティ・アクセスが必要になります。

- レコードが変更されたことを検出した要求が最初に拒否され、CHANGED 要求が送られます。
- 応答を受け取ったアプリケーションは、要求を再試行するかを決定する必要があります。

コンテンション・モデルを使用する場合、例外条件 (CHANGED) は、アプリケーションに対し、更新読み取り後の再書き込みまたは更新読み取り後の削除を再試行する必要があることを伝えます。これは、再書き込みまたは削除を実行する前に、テーブル内のレコードのコピーが他のタスクにより更新されているためです。コンテンション・モデルでは、レコードをロックせず、レコードのテーブル・エントリーのバージョン番号を使用して、変更されていないかをチェックします。再書き込みまたは削除時に、最初の更新読み取りが行われたときとこのレコードのバージョンが同じでない場合には、CHANGED 条件が戻されます。

ロック・モデルでは、複数の更新が行われないよう、更新読み取り要求後にレコードがロックされます。

コンテンション・モデルの CFDT は、リカバリー不能です。ロック・モデルの CFDT は、リカバリー可能またはリカバリー不能である場合があります。リカバリー不能ロック・モデルの場合、CFDT のロックは、更新読み取りシーケンスが再書き込み、削除、またはアンロック要求により完了するまで保持されます (ただし、次の同期点までではありません)。作業単位に障害が発生した場合、変更はバックアウトされません。リカバリー可能なケースでは、ロックは同期点まで保持され、CFDT レコードは、作業単位の障害、CICS 領域の故障、CFDT サーバー障害、または MVS の障害時に回復が可能です。

更新モデルとリカバリーの使用による相対コストは、要求をサポートするために必要なカップリング・ファシリティのアクセスの量に関連します。コンテンション・モデルで必要とされるアクセスは最低限ですが、データが変更されると、この条件の処理に追加のプログラミングとカップリング・ファシリティのアクセスが必要です。ロック・モデルではカップリング・ファシリティへのアクセスがより多く必要ですが、要求を再試行する必要はありません。一方、コンテンション・モデルを使用した場合は、繰り返し再試行が必要になる場合があります。リカバリーでは、リカバリー・データがカップリング・ファシリ

ティーのリスト構造に保持されることから、さらに、カップリング・ファシリティーのアクセスを必要とします。

以下の表では、更新モデル別に、CFDT 要求タイプをサポートするために必要なカップリング・ファシリティーのアクセスの量を示しています。

表 14. 要求タイプおよび更新モデルごとのカップリング・ファシリティーのアクセス			
要求の説明	競合	ロック	リカバリー可能
オープン、クローズ	3	3	6
読み取り、ポイント	1	1	1
新規レコード書き込み	1	1	2
更新読み取り	1	2	2
アンロック	0	1	1
再書き込み	1	1	3
削除	1	1	2
キーによる削除	1	2	3
同期点	0	0	3
ロック WAIT	0	2	2
ロック POST	0	2	2
システム間 POST	0	待ちサーバーごとに 2	待ちサーバーごとに 2

カップリング・ファシリティー・データ・テーブル (CFDT) の定義およびカップリング・ファシリティー・データ・テーブル・サーバーの開始方法については、[カップリング・ファシリティー・データ・テーブル・プールの定義](#)を参照してください。

効果

CFDT の使用と機能シッ UMT について、異なる MVS メンバーのシスプレックスで実行される 2 つの CICS 領域間で比較テストしたところ、CFDT の使用により、全般の CPU 使用率が 40% 以上も減少しました。一般的に役に立つ情報を以下に上げます。

- 4094 バイト以下 (4096 K または 4 K、2 バイトの接頭部データを含む) の CFDT レコードへのアクセスは、同期カップリング・ファシリティー 要求として CFDT サーバーで処理されています。4 K バイトよりも大きなレコードの要求は非同期に行われます。これらの非同期アクセスによるコストは、CPU 使用と応答時間で若干増します。同じトランザクション率 (337/秒) で異なるレコード・サイズを比較したベンチマーク・テストでは、4 K 未満の CFDT ワークロードでは、UMT での同等条件より CPU 使用が 41.7% 減少しました。4 K を超える CFDT ワークロードでは、CPU 使用が 41.1% 減少しましたが、応答時間の低下は測定では認められませんでした。
- コンテンション・モデルの使用で必要なカップリング・ファシリティーのアクセスは最低限となりますが、CHANGED 条件を処理し再試行の必要があることから、利点を最大限得るには CHANGED 条件がほとんどない場合となります。これらの発生については、これに続く CICS 統計に報告されます。
- CFDT レコード長が 63 バイト以下の場合、レコード・データは、カップリング・ファシリティーのリスト構造のエントリー付加属性領域に保管され、競合更新モード使用時のパフォーマンスが向上します。
- リカバリー可能なロック・モデルは、CFDT 操作で最もコストを要するモードです。このモードでは、カップリング・ファシリティーへのアクセスがより多く必要であるだけでなく、CFDT サーバーがリソース・マネージャーとしても機能し、要求元の CICS 領域と更新のコミットを調整します。CFDT レコードの READ/UPDATE および REWRITE (トランザクション率は 168/秒) を使用したベンチマーク・テストでは、競合およびロック CFDT を使用したトランザクション間で CPU 使用状況に大きな差は見られませんでした。ただし、CFDT がリカバリー可能と定義された場合、同じトランザクションの CPU 使用状況は約 15% 増加しました。

推奨

CFDT の適切な用法を選択してください。例えばシステム間では、リカバリー可能スクラッチパッド・ストレージでは、共用 TS には必要とされる機能がなく、また、VSAM RLS でも過大な処理が生じます。

ラージ・ファイルでは、これを含む大量のカップリング・ファシリティ・ストレージが必要です。CFDT の候補としては、類似するファイルの方が適しています (アプリケーションが、CFDT に保持されたレコードの数を制御するよう作成されている場合は除きます)。

ロック・モデルを使用する追加コストは、コンテンション・モデルと比較して大きくありません。コンテンション・モデルを使用することにより、既存のプログラムを使用する場合にアプリケーションの変更が必要であることを考慮すると、ロック・モデルは恐らく、CFDT で最適な更新モデルです。カップリング・ファシリティのアクセスが問題であれば、これはコンテンション・モデルにより最小化できます。

リカバリーのコストは、CPU 使用とカップリング・ファシリティの使用で若干増します。

CFDT のサイジングには、拡張を許可してください。構造が占有することのできるカップリング・ファシリティ・ストレージ量は、**SETXCF ALTER** コマンドにより、関連するカップリング・ファシリティ・リソース管理 (CFRM) ポリシーに定義された最大値まで動的に増やすことができます。CFDT サーバーに定義された **MAXTABLES** 値では、拡張を許可する必要があります。このため、これは、当初の要件よりも大きい値に設定するよう検討してください。CFDT がいっぱいになった場合、CFDT オペレーター・コマンド **SET TABLE=name, MAXRECS=n** を使用して容量を増やすことができます。

CFDT の使用状況は、CICS と CFDT の統計および RMF で定期的にモニターします。構造のサイズが、これに含まれるデータの量に対して適切であることを確認してください。最大で 80% の使用が適切な量の目標です。CFRM ポリシー定義で最大のカップリング・ファシリティ・リスト構造サイズを、CFDT サーバー開始パラメーターで **POOLSIZE** パラメーターに指定した初期割り当てサイズより大きい値に定義します。この設定により、特別な状況で構造が満杯になった場合に、**SETXCF ALTER** コマンドを使用して構造を動的に拡大できます。

AXMPGANY ストレージ・プールに十分な大きさがあることを確認してください。このプールは、CFDT サーバーの REGION サイズを増すことにより増やすことができます。AXMPGANY ストレージが不足すると、CFDT サーバーで 80A 異常終了が発生する可能性があります。

モニター

CICS および CFDT サーバー共に、統計レコードを作成します。このレコードについては、『[Reference](#)』の『[カップリング・ファシリティ・データ・テーブル・プール・レポート](#)』を参照してください。

CICS ファイル統計は、それぞれの CFDT に対して発行されたタイプごとに各種の統計を報告します。また、CFDT がいっぱいになった場合、保持されるレコードの最大数および変更応答/ロック待機のカウンタも報告します。この最後の項目は、競合 CFDT において、CHANGED 条件が戻された回数の判別に使用できます。ロック CFDT の場合、要求レコードはロック済みであるため、このカウンタには、待機に対してなされた要求の回数が報告されます。

詳しくは、[データ・テーブル・レポート](#)を参照してください。

カップリング・ファシリティ・データ・テーブルの統計

カップリング・ファシリティ・データ・テーブル (CFDT) サーバーは、使用するカップリング・ファシリティのリスト構造およびサポートするデータ・テーブルの両方について包括的な統計を報告します。CFDT 領域内で AXM ルーチンによって使用されるストレージ (AXMPGLOW 領域および AXMPGANY 領域) についても報告します。このデータは SMF に書き込むことができます。また、定期的な間隔で自動的に、あるいはオペレーター・コマンドによって、CFDT サーバーのジョブ・ログに作成することもできます。

CFDT 統計は、最新のカップリング・ファシリティ要求によって戻される情報を基に計算されます。現在のサーバーが関連情報に最近アクセスしていなかった場合、統計の正確さは期待できません。テーブルの数およびリストの数は、サーバーがテーブルを開いたり閉じたりする都度更新されますが、その他の場合には更新されないこともあります。エレメントとエントリーのカウンタは、ほとんどのタイプのカップリング・ファシリティ・アクセス要求が正常に完了した場合に更新されます。

以下のコードに、CFDT により作成されたカップリング・ファシリティ統計の例を示します。

```
DFHCF0432I Table pool statistics for coupling facility list structure DFH
CFLS_PERFCFT2:
```


Structure:	Size	Max size	Elem size	Tables:	Current	Highest
	12288K	30208K	256		4	4
Lists:	Total	In use	Max used	Control	Data	
	137	41	41	37	4	
	100%	30%	30%	27%	3%	
Entries:	Total	In use	Max used	Free	Min free	Reserve
	3837	2010	2010	1827	1827	191
	100%	52%	52%	48%	48%	5%
Elements:	Total	In use	Max used	Free	Min free	Reserve
	38691	12434	12434	26257	26257	1934
	100%	32%	32%	68%	68%	5%

この例は、カップリング・ファシリティーのリスト構造で現在使用されているスペース量 (Size) および構造に定義された最大サイズ (Max size) を示します。構造サイズは、**SETXCF ALTER** コマンドを使用して増やすことができます。定義されているリストの数は、CFDT サーバーの **MAXTABLES** パラメーターにより決定されます。この例の構造では、最大で 100 のデータ・テーブルをサポートします (および制御情報につき 37 のリスト)。

各リスト・エントリーは、エントリー制御の固定長セクションと、データ・エレメントの変数番号から構成されます。これらエレメントのサイズは、構造が最初にカップリング・ファシリティーに割り振られた時に固定され、**ELEMSIZE** パラメーターによって CFDT サーバーに定義されます。エントリー制御とエレメント間のカップリング・ファシリティー・スペースの割り振りは CFDT サーバーにより自動的かつ動的に変更され、必要に応じてスペース使用状況が改善されます。

予約スペースは、構造がユーザー・データでいっぱいになった場合に、再書き込みおよびサーバー内部操作が機能するよう使用されます。

CFDT 領域で AXM 要求のサポートに使用されるストレージ量も報告されます。例えば、以下のような項目が含まれています。

AXMPG0004I	Usage statistics for storage page pool AXMPGANY:				
Size	In Use	Max Used	Free	Min Free	
30852K	636K	672K	30216K	30180K	
100%	2%	2%	98%	98%	
	Gets	Frees	Retries	Fails	
	3122	3098	0	0	
AXMPG0004I	Usage statistics for storage page pool AXMPGLOW:				
Size	In Use	Max Used	Free	Min Free	
440K	12K	12K	428K	428K	
100%	3%	3%	97%	97%	
	Gets	Frees	Retries	Fails	
	3	0	0	0	

CFDT サーバーは、AXMPGANY および AXMPGLOW ストレージ・プール用に自らの領域のストレージを使用します。CFDT 領域の 16 MB 境界を超える使用可能ストレージのほとんどは、AXMPGANY が管理します。AXMPGLOW は、24 ビット・アドレス・ストレージ (16 MB 境界未満) を参照し、CFDT 領域内のこのストレージのわずか 5% だけを扱います。CFDT サーバーにおけるこのようなストレージの要件は少量です。

ローカル共用リソース (LSR) または非共用リソース (NSR)

VSAM バッファおよびストリングに対してローカル共用リソース (LSR) を使用するか、非共用リソース (NSR) を使用するかをファイルごとに決定する必要があります。

特定の VSAM データ・セットにアクセスするために開かれるすべてのファイルは通常、同じリソース・タイプを使用する必要があります。

VSAM 制御間隔 (CI) へのアクセス

LSR と NSR の重要な違いは、VSAM 制御間隔 (CI) への同時アクセスにあります。

- LSR では、ストレージには CI のコピーが 1 つだけ存在し、2 番目の要求は、最初の処理が完了するまでキューに入れる必要があります。LSR では、複数の読み取り操作が同じバッファへのアクセスを共用することが許可されます。
- NSR では、ストレージ内に CI の複数のコピーを持つことができます。1 つ (ただし 1 つだけ) のストリングで CI を更新し、その他のストリングでは同じ CI の異なるコピーを読み取ることができます。

ただし、更新ではバッファを排他使用する必要があり、前の更新または前の読み取りが完了するまでキューに入れる必要があります。また、読み取りでは、すべての更新が終了するまで待つ必要があります。したがって、同時ブラウズ操作および同時更新操作を行うトランザクションは、NSR では正常に実行されても、LSR では 2 番目の操作が最初の操作の完了を待つことができずにデッドロックが発生する可能性があります。

制御間隔のサイズ (CI)

データ・セットの CI のサイズは、CICS に指定されるパラメーターでなく、VSAM AMS によって定義されます。ただし、制御間隔へのアクセスを提供する CICS システムのパフォーマンスに大きく影響します。

一般に、直接入出力は、データ CI が小さい場合にやや高速で実行され、一方、順次入出力は、データ CI が大きい場合により高速になります。NSR ファイルの場合、小さいデータ CI を使用しながら追加バッファを割り当て、順次入出力をチェーニングおよびオーバーラップすることにより、中間的な解決が可能です。ただし、追加のデータ・バッファはすべて、順次入出力を行う最初のストリングに割り当てられます。

VSAM は、制御域が最大サイズのときに最も効率的に機能します。データ CI を索引 CI より大きく設定します。このため、標準の CI サイズは、データの場合は 4 KB から 12 KB、索引の場合は 1 KB から 2 KB となります。

通常はファイルのデータ CI のサイズを指定しますが、対応する適当な索引 CI を VSAM で選択することができます。例外として、キー圧縮が VSAM で予想したよりも効率的でない場合があります。この場合、VSAM が選択する索引 CI サイズが小さすぎる場合があります。非常に高レートで制御域 (CA) 分割が行われ、DASD スペースが十分利用されていない可能性があります。この問題が疑われる場合は、より大きい索引 CI を指定してください。

LSR の場合、CI サイズを標準化すると利点が得られる場合があります。この標準化により、ファイル間でのバッファ共用を増やし、バッファの総数を減らすことができます。これとは逆に、ファイルに固有の CI サイズを与えて、同じプールを使用する他のファイルを含むバッファを競合しないようにすることも可能です。

CI サイズは、512 バイト、1 KB、2 KB など、4 KB の倍数に設定してください。26 KB や 30 KB のような異常な CI サイズは使用しないでください。CI サイズを 26 KB にしても、物理ブロック・サイズが 26 KB になるわけではありません。物理ブロック・サイズは装置依存であるため、この場合の物理ブロックはおそらく 2 KB になります。

LSR および NSR のバッファ数

VSAM によるバッファの割り振りおよび共用の方法には、LSR と NSR の間に次のようないくつかの重要な相違があります。

LSR

LSR プール内で 1 サイズを占めるバッファのセットをサブプールと呼びます。ファイル制御ファイルには、最大で 255 個の個別の LSR プールを使用します。また、LSR プールへのデータ・セットの分散方法も決定する必要があります。CICS には、データおよび索引レコード用に個別の LSR バッファ・プールがあります。データ・バッファのみを指定した場合、1 セットのバッファのみ作成されて、データおよび索引レコードの両方に使用されます。各サブプールのバッファの数は、LSRPOOL 定義の **DATA** および **INDEX** パラメーターにより制御されます。正確な数を指定することも、CICS で数を計算することもできます。

CICS で LSR パラメーターを計算する方法は簡単ですが、LSR プールを必要とする最初のファイルを開くときに、プールを作成するための追加の処理が発生します。CICS で LSR プールを計算するには、以下の要因を考慮してください。

- CICS は、プールを使用するよう指定されたすべてのファイルについて VSAM カタログを読み取る必要があります。
- CICS が計算を実行する時点で、関連するデータ・セットがマイグレーション済みである場合には、処理が増えます。CICS が LSR プールに関連した各データ・セットの VSAM カタログを読み取ることができるようにするには、各データ・セットを再呼び出しする必要があります。
- 単一の再呼び出しにより再呼び出しを行うタスクに大きく遅延が生じるだけでなく、同期操作によって、同じ TCB で CICS が実行する他のアクティビティが遅れる結果となります。

こうした遅延は、CICS データ・セットがマイグレーションされないように SMS ストレージ・クラスおよびマイグレーション・ポリシーを設計することにより、避けることができます。データ・セットのマイグレーション基準の設定については、「[z/OS DFSMSHsm ストレージ管理](#)」を参照してください。

CICS は、再呼び出しが必要な場合、情報メッセージ DHFC0989 を出力して、以降の遅延がエラー状態でないことを伝えます。

- CICS により計算される LSR プールは、各バッファーごとに実サイズを指定して調整することはできません。
- LSR では、ストリングにも特定のファイルやデータ・セットにもバッファーの事前割り当ては行われません。VSAM は、バッファーを再使用する必要がある場合、参照頻度が最も低いバッファーを選択します。ストリングは常にすべてのデータ・セット間で共用されます。LSR を使用する場合、ディスクへの読み取りを発行する前に、まず VSAM はバッファーをスキャンして、必要な制御間隔が既にストレージ内に存在するかどうかをチェックします。存在する場合は、読み取りを発行する必要がないことがあります。このバッファーの検索により、入出力を大幅に削減できます。
- LSR ファイルは、バッファーの共通プールおよびストリングの共通プール、つまり、入出力操作をサポートする制御ブロックを共用します。その他の制御ブロックはファイルを定義し、各ファイルまたはデータ・セットに固有です。

LSR プールのサイズを変更する場合は、変更を行う前と行った後に CICS 統計を参照してください。これらの統計には、バッファーの検索によって結果を得られた VSAM 読み取りの比率が変化したかどうかを示されます。

一般に、追加索引バッファーの検索から期待できる利益は多く、追加データ・バッファーの検索では少なくなります。このことも、1 つのサブプール内に索引 CI とデータ CI が混在しないよう、LSR のデータ CI および索引 CI のサイズを標準化する理由として挙げられます。

データ・バッファーと索引バッファーは LSRPOOL 定義で別々に指定されるため、CI サイズを使用してデータと索引の値を区別する必要はありません。

適切なサイズのバッファーを指定してください。必要なサイズのバッファーが存在しない場合、VSAM は次に大きいバッファー・サイズを使用します。

NSR

- ファイルの **STRINGS** パラメーターで指定された同時アクセスをサポートするために、各ファイルに対して十分なバッファーを用意する必要があります。実際に、VSAM では NSR に対してこの要件を強制しています。
- ファイル定義の **DATABUFFERS** パラメーターおよび **INDEXBUFFERS** パラメーターを使用して、NSR のデータ・バッファーおよび索引バッファーの数を指定します。十分な索引バッファーを指定することが重要です。KSDS が 1 つだけの制御域で構成されており、したがって、索引 CI が 1 つだけの場合は、**STRINGS** に等しい最小限の索引バッファー数で十分です。しかし、KSDS がこの値よりも大きい場合には、すべてのストリングで最低でも最上位の索引バッファーを共用できるよう、少なくとも 1 つは追加の索引バッファーを指定する必要があります。さらに索引バッファーを増やすと、索引入出力がある程度削減されます。
- **DATABUFFERS** は最小値となる **STRINGS + 1** に設定します。ただし、順次操作で入出力のオーバーラップおよびチェーニングを有効にすることが目的である場合、あるいは CA 分割を高速化するために追加バッファーを提供する必要がある場合を除きます。
- ファイルがベースへの代替索引パスである場合は、代替索引とベースのバッファーに使用される **INDEXBUFFERS** (ベースが KSDS の場合) と **DATABUFFERS** の設定を同じにします ([170 ページの『CICS による LSR プール・パラメーターの計算』](#)を参照してください)。NSR では、データ・バッファーの最小数は **STRNO + 1** で、最小の索引バッファー数 (KSDS および代替索引パスの場合) は **STRNO** です。各ストリングには、1 つのデータ・バッファーと 1 つの索引バッファーが事前に割り振られ、1 つのデータ・バッファーが CA 分割用に予約されます。追加のデータ・バッファーがある場合、これらのバッファーは最初の順次操作に割り当てられます。また、これらのバッファーは、チェーニングされた入出力操作を許可することにより、VSAM CA 分割の高速化にも使用できます。追加の索引バッファーがある場合は、それらがストリング間で共用され、上位索引レコードの保持に使用されるため、物理的入出力が減る可能性があります。

- NSR ファイルまたはデータ・セットは、バッファと制御ブロックの独自のセットを持ちます。

注: NSR は、トランザクション分離を使用するトランザクションではサポートされません。NSR ファイルを使用するファイル制御コマンドは、スレッド・セーフではありません。

必ず、デッドロックを回避するようにトランザクションを設計し、プログラムしてください。詳しくは、[トランザクション・デッドロック](#)を参照してください。

ストリング数

次に、各ファイルおよび各 LSR プールでサポートされる同時アクセスの数を決定します。

VSAM ストリングを指定する必要があります。ストリングは、VSAM データ・セットに対する要求で、データ・セット内での位置決めを要求します。指定された各ストリングにより、作成される VSAM 制御ブロック (プレースホルダーを含む) の数が決定されます。

VSAM では、並行したファイル操作ごとに 1 つ以上のストリングを必要とします。更新以外の要求 (READ または BROWSE など) の場合、ベースを使用するアクセスは 1 つのストリングを必要とします。代替索引を使用するアクセスには、2 つのストリングが必要です (1 つは代替索引上の位置を保持するため、もう 1 つは基本データ・セット上の位置を保持するために必要です)。アップグレード・セットを含まない更新要求の場合でも、ベースに 1 つのストリング、パスに 2 つのストリングを必要とします。アップグレード・セットが含まれる更新要求の場合、ベースに $1+n$ 個のストリングが必要であり、パスに $2+n$ 個のストリングが必要です。ここで、 n は、アップグレード・セット内のメンバーの数です。VSAM では、位置を保持するために、アップグレード・セット・メンバーごとに 1 つのストリングが必要です。同時要求ごとに、VSAM はアップグレード・セットの処理に必要な n 個のストリングを再使用できます。アップグレード・セットは順次に更新されるからです。

直接読み取りのような単純な操作では、1 つ以上のストリングが即時に解放されます。しかし、更新読み取り、大量挿入、またはブラウズ要求では、対応する更新、アンロック、またはブラウズ終了要求が実行されるまで、1 つ以上のストリングが保持されます。

CICS と VSAM による **STRNO** パラメーターの解釈は、コンテキストに応じて異なります。

- LSR プール定義 (LSRPOOL) における同等の **STRINGS** パラメーターは、VSAM BLDVRP マクロの **STRNO** パラメーターと同じ意味を持ちます。これはつまり、リソース・プールに割り振られるストリングの絶対数を表します。LSR プールに基本データ・セットのみ含まれる場合を除き、処理可能な同時要求の数は、指定された **STRINGS** 値よりも小さくなります。
- ファイル定義における同等の **STRINGS** パラメーターは、NSR ファイルの VSAM ACB における **STRNO** パラメーターと同じ意味を持ちます。つまり、処理可能な未解決 VSAM の同時要求の実際の数です。代替索引パスまたはアップグレード・セットが使用される場合、これをサポートするため VSAM が割り振るストリングの実際の数、指定された **STRINGS** 値よりも大きくなる場合があります。

LSR の場合、ストリングの正確な数を指定するか、CICS で数を計算することが可能です。LSR プール定義に指定された数は、プール内のストリングの実際の数です。CICS でストリングの数を計算する場合、その数は RDO ファイル定義のプールの **STRINGS** から派生します。これは、NSR の場合と同様に、このプールを同時要求の実際の数として解釈します。

サポートする必要がある同時の読み取り、ブラウズ、更新、大量挿入の要求などの数を決定する必要があります。

ファイルへのアクセスが、ブラウズを行わない読み取り専用の場合、多数のストリングは必要ありません。ストリングは 1 つだけで十分なこともあります。読み取り操作では、要求の期間だけ VSAM ストリングを保持しますが、同じ CI に対する更新操作が完了するのを待つ必要がある場合があります。

一般に、ブラウズまたは更新を使用する場合、**STRINGS** は最初に 2 または 3 に設定し、CICS ファイル統計を定期的にチェックして、発生する wait-on ストリングの比率を確認する必要があります。通常は、ファイル・アクセスの最大 5% までの wait-on ストリングを許容できると考えられます。NSR ファイルの場合、wait-on ストリングをゼロのままにしないでください。

CICS は、ファイルおよび LSR プールの両方でストリング使用を管理します。LSR または NSR のいずれを使用する場合でも、CICS はファイルごとに同時 VSAM 要求の数を、ファイル定義に指定された **STRINGS**= に制限します。また、CICS は、LSR ごとにも、プール内のストリングで処理可能な数より多くの同時要求が VSAM に対して行われないう制限します。更新時のアップグレード・セットの処理に追加のストリン

グが必要とされる場合、CICS は先に read-for-update 時に追加ストリングを予約しておきます。使用可能なファイルまたは LSR プール・ストリングが十分にない場合、要求タスクはそれらが解放されるまで待機します。CICS 統計には、ストリングの待機に関する詳細が示されます。

特定のファイルについてストリングの数を決める場合は、同時タスクの最大数を考慮してください。CICS コマンド・レベルでは、複数の要求を特定タスクの特定のデータ・セットに対して未解決とすることができないため、それ以上の同時要求にストリングを使用しても意味がありません。

異なるタイプのタスクにストリングを分散する場合、トランザクション・クラスも役立ちます。トランザクション・クラスの制限を使用すると、個別のタイプの VSAM 要求を発行するトランザクションを制御できます。また、VSAM ストリングを使用できるタスク・タイプを制限することにより、ストリングのサブセットを他の用途で使用できるように残しておくことができます。

すべてのブレースホルダー (置き換え) 制御ブロックは、プールを共用するすべてのデータ・セットに関連した最大のキーに十分な長さのフィールドを含む必要があります。大きなキー (基本または代替) を持つ 1 つの非アクティブ・ファイルを、多数のストリングを含む LSR プールに割り当てると、非常に多くのストレージが使用される場合があります。

ESDS ファイルの考慮事項

ESDS ファイルの **STRINGS** 値を選択する場合、パフォーマンス上の特別な考慮事項があります。

ESDS を追加専用ファイルとして使用する場合 (ファイル末尾にレコードを追加するため書き込みモードのみ使用する)、ストリング数を 1 にすることをお勧めします。ストリングの数を 1 よりも大きくすると、複数のタスクが同時に ESDS に書き込みを行おうとした場合に排他制御が競合するため、パフォーマンスに大きく影響します。

ESDS を書き込みと読み取りの両方にする場合 (書き込みがアクティビティの 80%)、1 つのファイルを書き込み用、もう 1 つを読み取り用に、2 つのファイル定義を定義してください。

効果

LSR では、以下の効果による大きな利点が得られます。

- バッファとストリングを共用するため、仮想記憶をより有効に使用する。
- バッファ検索の向上によるパフォーマンスの改善により、入出力操作を削減する。
- ストレージに存在する CI のコピーは 1 つだけとなり、読み取り保全性が向上する。
- 使用中ファイルへの割り振りバッファが増し、参照回数の多い索引の制御間隔がバッファに保持されることによる自己調整。
- 同期ファイル要求および UPAD 出口の使用。LSR ファイルの CA および CI 分割により、サブタスクやメインタスクが待機することはありません。VSAM は、物理的入出力を待ちながら UPAD 出口を受け取り、CA/CI 分割の間、他の CICS 作業の処理が続行されます。

NSR ファイルのファイル制御要求は非同期式に行われますが、この場合でも、CICS メインタスクまたはサブタスクが分割中に停止します。

- トランザクション分離のサポート。

NSR では、以下の効果が得られます。

- 特定のデータ・セットを優先してチューニング。
- 順次操作のパフォーマンスを向上。

推奨

以下のいずれかの状態がある場合を除いて、すべての VSAM データ・セットで LSR を使用してください。

- ファイルがアクティブでありながら、ファイルが大きいなどの理由で、検索を行う機会がない。
- 追加の索引バッファの割り振りによってハイパフォーマンスが要求される。
- 追加のデータ・バッファによって高速な順次ブラウズまたは大量挿入が要求される。

- ファイルに制御域 (CA) 分割が想定され、CA 分割の高速化のため、追加のデータ・バッファが割り振られる。

LSR プールが 1 つだけの場合、ストリングを競合している状態のときは、同じプールを使用して特定のデータ・セットを他から分離することができません。分離できるのは、固有の CI サイズを指定して、バッファを競合する場合だけです。一般的に、1 つの大きなプールを使用して実行することにより、自己調整作業は増えます。複数のプールを使用することによって、使用中ファイルをその他のファイルより分離したり、ハイパフォーマンスのグループに追加のバッファを割り振ることができます。非常にアクティブなファイルは、NSR を使用する代わりに LSR サブプール内の唯一のファイルとして設定することにより、正常にバッファ検索される機会が増え、入出力も減らすことができます。また、複数のプールを使用することにより、プールごとの 255 ストリングの制限を緩和することもできます。

制限

同じ基本データ・セットのすべてのファイル (ファイル定義に DSNSHARING(MODIFYREQS) が指定された読み取り専用ファイルを除く) は、同じ LSR プールを使用するか、すべて NSR プールを使用する必要があります。

SERVREQ=REUSE ファイルは、LSR を使用できません。

カップリング・ファシリティ・データ・テーブル

ここに示されている CPU 命令データは、9672-R55 システムを使用して取得したものです。

以下の 2 つのテーブルが示されています。

- 1 つ目は、カップリング・ファシリティへの同期アクセスとなるレコード長 (4K より小さい) のテーブルである。
- 2 つ目は、カップリング・ファシリティへの非同期アクセスとなるレコード長 (4K より大きい) のテーブルである。

非同期要求の場合は、処理により多くの CPU 時間がかかることに注意してください。また、応答時間も同期要求の場合より若干長くなります。4 K より小さいレコード長の API 呼び出しごとの CPU 命令は、以下のとおりです。

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	11.8	11.8	11.8
READ/UPDATE	12.0	22.2	22.4
REWRITE	19.5	24.0	33.0
WRITE	8.0	8.0	13.0
DELETE	7.0	11.0	16.5

4 K より大きいレコード長の API 呼び出しごとの CPU 命令は、以下のとおりです。

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	15.3	15.3	15.3
READ/UPDATE	15.0	25.7	25.9
REWRITE	23.0	27.5	36.5
WRITE	11.5	11.5	16.5
DELETE	10.5	14.5	20.0

VSAM レコード・レベル共用の使用

VSAM レコード・レベル共用 (RLS) は、DFSMS で導入され、CICS でサポートされてい VSAM データ・セット・アクセス・モードです。RLS により、多数の CICS 領域で実行される多くのアプリケーション間で、完

全な更新機能を使用して VSAM データを共用することができます。RLS では、VSAM データ・セットを共用する CICS 領域は、MVS シスプレックス内の 1 つ以上の MVS イメージ内に置くことができます。

RLS には、CICS 領域とバッチ・ジョブ間でデータ・セットを共用する場合の利点もあります。

RLS では、以下のコンポーネントが使用されます。

VSAM サーバー (サブシステム SMSVSAM)

このサブシステムは、独自のアドレス・スペースで稼働し、並列シスプレックス環境の各 MVS イメージ内で、CICS アプリケーション所有領域 (AOR) およびバッチ・ジョブで必要とされる RLS サポートを提供します。

SMSVSAM との CICS インターフェースは、アクセス制御ブロック (ACB) により行われ、CICS はこの ACB を登録して接続を開きます。ユーザー・アクションで接続を開く必要のある Db2 および DBCTL データベース・マネージャー・サブシステムとは異なり、システム初期設定パラメーターとして **RLS=YES** を指定すると、CICS の初期化時に SMSVSAM 制御 ACB が CICS によって自動的に登録されます。

CICS 領域は、ACB ファイルを RLS モードで開く前に、SMSVSAM に登録する制御 ACB を開く必要があります。各通常のファイル ACB は、引き続きファイル・アクセス要求のインターフェースとなります。

共用制御データ・セット

VSAM では、RLS 制御用にいくつかのデータ・セットが必要になります。VSAM 共用制御データ・セットは、論理区分された線形データ・セットです。これは、2 次エクステントによる定義が可能ですが、各データ・セットのすべてのエクステントは、同じボリューム上にある必要があります。

少なくとも 3 つの共用制御データ・セットを定義します。VSAM では、二重モードでの使用に 2 つのアクティブ・データ・セットを必要とし、いずれかのアクティブ・データ・セットで障害が発生した場合の予備として、3 つ目のデータ・セットが必要になります。

共用制御データ・セットの詳細およびこれを定義する JCL サンプルについては、「[z/OS DFSMSdfp Storage Administration](#)」を参照してください。

共通バッファ・プールおよび制御ブロック

非 RLS モードでアクセスされるデータ・セット用に、VSAM 制御ブロックおよびバッファ (ローカル共用リソース (LSR) プール) が各 CICS アドレス・スペースに置かれます。このため、これはバッチ・プログラムには使用できず、他の CICS 領域に対しても使用することができません。

RLS では、すべての制御ブロックおよびバッファは、SMSVSAM サーバーの関連データ・スペースに割り振られます。この構造により、各 MVS イメージに 1 つの大きなバッファ・プールが用意され、SMSVSAM サーバーに接続されたすべての CICS 領域、およびバッチ・プログラムによって共用できます。このデータ・スペース内のバッファは、自動的に作成され解放されます。

DFSMS には、**RLS_MAX_POOL_SIZE** パラメーターが用意されており、これは IGDSMSxx SYS1.PARMLIB メンバーで指定できます。この他には RLS には LSR プールのようなチューニング・パラメーターはありません。RLS バッファの管理は、完全に自動的に行われます。

入力順データ・セット (ESDS) で RLS を使用すると、複数の領域から複数のタスクを使用してレコードを追加するときに、データ・セットの可用性が低下する場合があります。これは、レコードの追加では、書き込みを実行するために排他的な add-to-end ロックが必要なためです。CICS 領域で ESDS への書き込みに障害が発生すると、CICS 領域を再始動するまで、データ・セットがロックされる場合があります。

CICS ファイルで RLS アクセス・モードを使用するには、以下のタスクを実行します。

1. 必要な共用制御データ・セットを定義します。
2. **RLS_MAX_POOL_SIZE** パラメーターを IGDSMSxx SYS1.PARMLIB メンバーで定義します。
3. RLS をサポートする MVS イメージで SMSVSAM サーバーが開始されていることを確認します。
4. システム初期設定パラメーター **RLS=YES** を指定します。このパラメーターにより、CICS 初期化時に制御 ACB を開いて、CICS が SMSVSAM サーバーに自動的に登録されます。RLS=NO を指定して CICS を開始した場合、後から動的に RLS サポートを使用可能にすることはできません。
5. RLS アクセス・モードで使用するデータ・セットが定義されていることを確認します。これには、アクセス方式サービス・プログラム (AMS) を使用して、IDCAMS DEFINE ステートメントに **LOG** および **LOGSTREAMID** パラメーターにより必要なリカバリー属性が定義されていることを確認します。これ

ら属性を使用せずに定義された既存のデータ・セットを使用する場合は、これらのデータ・セットを使用してデータを再定義します。

6. ファイル・リソース定義に **RLSACCESS(YES)** を指定します。

CICS では、VSAM ファイルにアクセスするために、3 つの異なるモードを使用できます。これらは、非共用リソース (NSR) モード、ローカル共用リソース (LSR) モード、およびレコード・レベル共用 (RLS) モードです。(CICS では VSAM グローバル共用リソース (GSR) アクセス・モードはサポートされません。) アクセス・モードは、データ・セット自体の特性ではなく、データ・セットを開く方式の特性です。つまり、ユーザーは、あるデータ・セットをある時は NSR モードで開き、また、ある時は RLS モードで開くことができます。非 RLS モードという用語は、CICS でサポートされる NSR または LSR アクセス・モードを示す一般用語として使用されています。混合モード操作とは、RLS モードで開いたデータ・セットを同時に異なるユーザーが非 RLS モードで開くことを指します。

データ・セットはその都度異なるモードで開くことができますが、VSAM スフィア内のすべてのデータ・セットは、通常同じモードで開く必要があります。スフィアは、ある VSAM 基本データ・セットに関連したすべてのコンポーネントの集合のことです - ベース、索引、代替索引、および代替索引パス。ただし VSAM では、CICS の制限を前提として、異なるアプリケーションによるスフィアでの混合モード操作が許可されています。

効果

記されたテストおよび測定は、キー順データ・セット (KSDS) で RLS を使用して行われたものです。このトピックで既に述べたように、RLS を入力順データ・セット (ESDS) で使用することは推奨されません。レコードを追加する際に、パフォーマンスと可用性に関する問題が発生する恐れがあるからです。

RLS は、MRO を使用してファイル専用領域 (FOR) に機能シップした場合に比べると、CPU コストが増加します。標準の DSW ワークロードを使用して CPU 使用を測定した比較は以下のとおりです。

- MRO 仮想記憶間 (XM) 接続でローカル・ファイル・アクセスから機能シップに切り替えると、単一の CPC でトランザクションあたり 7.02 ms 増加する結果となりました。
- MRO XM から RLS への切り替えでは、単一 CPC でトランザクションあたり 8.20 ms の増加です。
- 2 つの CPU を使用した XCF/MRO から RLS への切り替えでは、トランザクションあたり 2.39ms 縮小されました。
- 1 つの CPC を使用した RLS から 2 つの CPU を使用する RLS に切り替えた場合、違いは認められませんでした。

応答時間についてのパフォーマンス測定は以下のとおりです。

- MRO XM を使用した機能シップは RLS を上回りましたが、これは 1 つの MVS イメージ内での機能シップに限られ、複数の MVS イメージまたは複数の CPU を使用した並列シスプレックスを完全な形で使用することはできません。
- FOR を AOR と異なる MVS イメージで実行した場合、RLS は XCF/MRO による機能シップよりも優れます。

ただし、パフォーマンス測定だけでは不十分で、例えば次のような他の要因は考慮されていません。

- 同じ VSAM データを共用するアプリケーションが増えるにつれ、単独の FOR での負荷が増し、FOR がスループットのボトルネックとなる場合があります。FOR は、CICS 内部アーキテクチャーにより、ユーザー・タスクでは単一の TCB に制限され、通常は CICS 領域で複数の CPU を使用できません。
- FOR に接続する AOR が増えるにつれ、セッション管理が困難になります。

これら FOR のマイナスの側面は、FOR にはないスケーラビリティを持つ RLS により解決されます。

モニター

RLS アクセス・モードを VSAM ファイルで使用する場合、SMSVSAM と、ファイル制御要求を発行する CICS 領域が関連します。これを選択した場合、全体の状況を把握するために CICS と SMSVSAM の両方のパフォーマンスをモニターする必要があります。これには CICS パフォーマンス・モニター・データと SMSVSAM が書き込む SMF タイプ 42 レコードを組み合わせて使用します。

CICS モニター

RLS アクセスの場合、CICS は、以下を含むパフォーマンス・クラス・レコードを SMF に書き込みます。

- SMSVSAM SRB での RLS CPU 時間
- RLS 待ち時間

SMSVSAM SMF データ

SMSVSAM は、タイプ 42 レコード、サブタイプ 15、16、17、18、および 19 を書き込み、カップリング・ファシリティー・キャッシュ・セット、構造、ロック統計、CPU 使用などの情報を提供します。この情報は、RMF III ポストプロセッシング・レポートを使用して分析できます。

以下のコードは、SMSVSAM データのレポート取得に使用可能な JCL のサンプルです。

```
//RMFCF      JOB (accounting_information),MSGCLASS=A,MSGLEVEL=(1,1),CLASS=A
//STEP1      EXEC PGM=IFASMFDP
//DUMPIN     DD DSN=SYS1.MV2A.MANA,DISP=SHR
//DUMPOUT    DD DSN=SMSF,UNIT=SYSDA,
//           DISP=(NEW,PASS),SPACE=(CYL,(10,10))
//SYSPRINT   DD SYSOUT=*
//SYSIN      DD *
            INDD(DUMPIN,OPTIONS(DUMP))
            OUTDD(DUMPOUT,TYPE=000:255))
//POST       EXEC PGM=ERBRMFPP,REGION=0M
//MFPINPUT   DD DSN=SMSF,DISP=(OLD,PASS)
//SYSUDUMP   DD SYSOUT=A
//SYSOUT     DD SYSOUT=A
//SYSPRINT   DD SYSOUT=A
//MFPMSGDS   DD SYSOUT=A
//SYSIN      DD *
NOSUMMARY
SYSRPTS(CF)
SYSOUT(A)
REPORTS(XCF)
/*
```

CICS ファイル制御統計には、CICS 領域で発行されたファイル制御要求数に関する情報が含まれます。これはまた、RLS モードでアクセスされたファイルも示し、RLS タイムアウトの数や RLS ファイルの EXCP 数が示されます。SMSVSAM サーバーやそのバッファー使用、またはカップリング・ファシリティーへのアクセスに関する情報は含まれません。

スレッド・セーフなファイル制御アプリケーション

デフォルトでは、スレッド・セーフ・アプリケーションにより発行されるファイル制御のコマンドを CICS は強制的に実行します。システム初期設定パラメーター **FCQRONLY** の指定を **NO** に変更すると、ローカル VSAM LSR または RLS ファイルに対するファイル制御コマンドを L8 または L9 TCB で実行できます。

スレッド・セーフのファイル制御を使用すると、複数のプロセッサを使用できる CICS 領域のスループットが大きく向上することがあります。ファイル制御コマンドが発行されると現在 L8 または L9 の TCB 上で稼働しているタスクは QR TCB に切り替わることなく、そのまま L8 または L9 の TCB で稼働を続けます。高い並行性および優れたタスク・スループットにより、これらのタスクのパフォーマンスは向上します。ファイル制御コマンドを Db2 または IBM MQ 要求と結合したスレッド・セーフ・アプリケーションにおいて、プロセッサの縮小およびスループットの高速化が見られます。

スレッド・セーフなファイル制御から利点を得るには、アプリケーションが以下の要件を満たしている必要があります。

- プログラム・リソースが **CONCURRENCY(THREADSAFE)** または **CONCURRENCY(REQUIRED)** で定義されている。
- 発行されるファイル制御コマンドが、ローカル VSAM LSR または RLS ファイルに対するものである。
- ファイル制御コマンドを実行する CICS 領域に対して、システム初期設定パラメーター **FCQRONLY=NO** が指定されている。 **FCQRONLY=YES** がデフォルトです。

スレッド・セーフのファイル制御は、ファイルが CICS 領域でローカルと定義されており、VSAM LSR または RLS である場合の CICS 領域に役立ちます。ファイル制御の観点から、ファイル・タイプが混在している CICS 領域では、システム初期設定パラメーター **FCQRONLY=NO** を指定することを考慮します。その上

で、ローカル VSAM LSR または RLS ファイルにアクセスするプログラムは CONCURRENCY(THREADSAFE) で定義し、その他のファイル・タイプにアクセスするプログラムは CONCURRENCY(QUASIRENT) で定義します。CICS 領域内のファイルがローカル VSAM LSR または RLS でない場合は、デフォルトのシステム 初期設定パラメーター **FCQRONLY=YES** を使用します。

ファイル所有領域 (FOR) に機能シッブされる要求

アプリケーション所有領域 (AOR) からファイル所有領域 (FOR) にファイル制御要求を機能シッブする場合、**FCQRONLY** の設定を次のように選択します。

- CICS TS 4.2 以降で、TCP/IP 経由の IP 相互接続 (IPIC) 接続を使用する FOR の場合、これらの接続のパフォーマンスを最適化するために、**FCQRONLY=NO** を指定します。
- MRO リンク接続または SNA 経由の ISC 接続を使用する FOR の場合、これらの接続のパフォーマンスを最適化するために、**FCQRONLY=YES** を指定します。また、CICS TS 4.2 より前のすべての FOR に対して **FCQRONLY=YES** を使用します。

特定の AOR がすべてのファイル制御要求を FOR に機能シッブし、ローカル・ファイルがない場合には、その AOR に対してデフォルトの **FCQRONLY=YES** を使用できます。この領域ではスレッド・セーフな制御から利点は得られないためです。いくつかのローカル・ファイルを持つ AOR の場合は、その領域内のファイル・タイプに応じて **FCQRONLY** の設定を選択してください。

ファイル制御 API のコスト

読み取り操作の場合、DASD にアクセスする必要があるかどうかワークロードに依存しているため、VSAM I/O コストは含まれません。読み取り操作を完了するには、索引とデータの両方にアクセスする必要があります。索引またはデータがバッファ内にはない場合は、索引の各レベルおよびそれぞれのデータごとに入出力操作が必要です。

1K 命令カウント内のファイル・タイプごとの入出力命令の相対数は、以下のとおりです。

- 9.5 (キー順データ・セット (KSDS) の場合)
- 9.5 (入力順データ・セット (ESDS) の場合)
- 8.2 (相対レコード・データ・セット (RRDS) の場合)

READ

KSDS	ESDS	RRDS	データ・テーブル (CMT)
3.0	2.4	2.2	初回: 1.5 初回以降: 1.1

READ UPDATE

リカバリー可能ファイルおよびリカバリー不能ファイルは、READ UPDATE コストに含まれます。

表 15. リカバリー不能ファイル			
KSDS	ESDS	RRDS	
3.1	2.3	2.2	

リカバリー可能 READ UPDATE は、変更前イメージをログ・バッファに入れ、ログ・バッファは、その後 1 次ストレージに書き込まれない場合は REWRITE が完了する前に書き出されます。

KSDS	ESDS	RRDS
5.5	4.3	4.2

REWRITE

リカバリー可能ファイルおよびリカバリー不能ファイルは、REWRITE コストに含まれます。各 REWRITE には、データ VSAM I/O が関連付けられています。

表 16. リカバリー不能ファイル		
KSDS	ESDS	RRDS
10.2	10.1	10.1

リカバリー可能ファイルの REWRITE は、変更前イメージを含む ログ・バッファが書き出されていることを必要とします。READ UPDATE 以降にバッファがまだ書き出されていない場合は、ログ・バッファの書き込みのコストが発生します。変更前イメージが書き出されると、VSAM I/O が行われます。リカバリー可能リソースが更新された場合、トランザクションの終わりに、同期点に追加コストがかかります。195 ページの『同期点』を参照してください。

KSDS	ESDS	RRDS
10.4	10.3	10.3

WRITE

WRITE のコストには、リカバリー不能ファイルおよびリカバリー可能ファイルが含まれます。各 WRITE には、データ VSAM I/O が関連付けられています。索引は、制御域分割が発生した場合にのみ書き込まれる必要があります。

表 17. リカバリー不能ファイル		
KSDS	ESDS	RRDS
12.9	11.1	10.9

各 WRITE には、ファイルに既にレコードが存在していないかどうかを確認するために、隠れた READ が関連付けられています。バッファに索引、データ、またはその両方が存在しない場合、この隠れた READ によって I/O コストが発生する可能性があります。リカバリー可能ファイルに対する各 WRITE では、VSAM I/O が行われる前に、データ・イメージを含むログ・バッファが書き出されている必要があります。

リカバリー可能リソースが更新された場合、トランザクションの終わりに、同期点に追加コストがかかります。195 ページの『同期点』を参照してください。

表 18. リカバリー可能ファイル		
KSDS	ESDS	RRDS
14.9	13.1	12.9

DELETE

ESDS レコード・ファイルで削除を行うことはできません。

表 19. リカバリー不能ファイル	
KSDS	RRDS
12.5	11.5

リカバリー可能リソースが更新された場合、トランザクションの終わりに、同期点に追加コストがかかります。195 ページの『同期点』を参照してください。

表 20. リカバリー可能ファイル	
KSDS	RRDS
14.5	13.5

ブラウズ

STARTBR	READNEXT	READPREV	RESETBR	ENDBR
3.1	1.5	1.6	2.6	1.4

UNLOCK

EXEC CICS UNLOCK のパス長さは 0.7 です。

パフォーマンスのためのデータベース管理

パフォーマンスを改善するために、データベース管理のいくつかの側面を調整することができます。

DBCTL パラメーターの設定

DBCTL パフォーマンスを支援するには、いくつかのパラメーターが必要です。これには、DRA 始動テーブル (DFSPZP) で指定する **MINTHRD** と **MAXTHRD**、および DBCTL システム生成時または DBCTL 初期設定時に定義する DEDB パラメーター (**CNBA**、**FPBUF**、および **FPBOF**) が含まれます。

DBCTL パラメーターおよび CICS-DBCTL システムの調整について詳しくは、[スレッド数の指定](#)および [DEDB パフォーマンスおよびチューニングの考慮事項](#)を参照してください。

CICS Db2 接続機能の調整

CICS Db2 接続機能は Db2 とのマルチスレッド接続を実現します。CICS Db2 接続機能の DB2CONN、DB2ENTRY、および DB2TRAN 定義は、トランザクションおよびトランザクション・グループに基づいて許可属性およびアクセス属性を定義します。CICS と Db2 間のパフォーマンスを最適化するには、トランザクション・クラス制限、CICS の MXT システム・パラメーター、および DB2CONN と DB2ENTRY の THREADWAIT、TCBLIMIT、THREADLIMIT、および PRIORITY 属性を調整します。

CICS Db2 接続およびパフォーマンスの考慮事項については、以下のように、いくつかのトピックで追加情報が提供されています。

- [Defining the CICS Db2 connection](#) では、最適なパフォーマンスを実現する CICS Db2 接続を定義するための推奨事項について説明しています。
- [スレッドの作成、使用、および終了](#) には、スレッドの説明と、Db2 での THREADWAIT、TCBLIMIT、および THREADLIMIT の各パラメーターの使用についての説明があります。
- [CICS Db2 に関するアプリケーション設計と開発の考慮事項](#) には、アプリケーション設計に関する推奨事項があります。
- [Db2 にアクセスする CICS アプリケーションのチューニング](#) には、CICS Db2 アプリケーションの調整に関する推奨事項があります。

要約すると、CICS 接続機能を調整する目的は次のとおりです。

- 接続内のスレッド数を最適化します。

接続内のスレッドの総数、および各専用エン트리およびプールのスレッド数を最適化する必要があります。スレッド数が必要以上に大きい場合は、TCB をディスパッチするためのプロセッサ時間や、計画、データ、および制御ブロック用のストレージが余分に必要になります。定義されているスレッド数が十分でない場合は、応答時間が長くなります。

- 割り当てを最適化し、スレッドを再使用します。

スレッドを再使用すると、計画の割り振りや許可検査を含む、スレッドの作成および終了プロセスを回避できます。トランザクションが単純な場合、スレッドの作成および終了は処理時間のかなりの部分を占めます。スレッドの再使用は、CICS Db2 統計を使用して測定できます。

トランザクション・クラスを使用するか、または専用の DB2ENTRY (0 より大きな THREADLIMIT) に THREADWAIT=YES を指定して使用することにより、会話型トランザクションを制限します。このようにしないと、会話型トランザクションはプールに関連付けられます。会話型トランザクションにプールの使用を許可しないでください。

- プールおよびエントリーのスレッドの場合、PRIORITY パラメーターを使用して、サブタスク・スレッド TCB に割り当てる優先順位を選択します。

PRIORITY パラメーターは CICS メイン TCB (QR TCB) に関連する CICS オープン L8 スレッド TCB の優先順位を制御します。PRIORITY=HIGH、PRIORITY=LOW、および PRIORITY=EQUAL の 3 つのオプションがあります。

PRIORITY=HIGH が指定されている場合、トランザクションは CICS よりも高い優先順位で実行されるため、仮想記憶は節約され、ロックは解放され、その他のトランザクションのデッドロックまたはタイムアウトがなくなります。ただし、すべてのスレッドに PRIORITY=HIGH が指定されている場合、CICS は極端に低い優先順位で動作する可能性があります。例えば、複雑な SQL 呼び出しは Db2 での所要時間が長くなり、CICS TCB がディスパッチされなくなることがあります。

SQL 呼び出しの加重平均が最も大きいトランザクションに PRIORITY=HIGH を設定してください。最大加重平均は、トランザクション単位の SQL 呼び出し数にトランザクションの頻度を掛けた値に等しくなります。その他のトランザクションには、PRIORITY=LOW または EQUAL を設定してください。呼び出しあたりの CPU 使用量が大きい場合は、PRIORITY=HIGH を設定しないでください。

- スレッドごとのサインオン・プロセスを回避するか、または最小化するために、最適な許可計画を選択してください。
- DB2ENTRY 数を最小にします。ワイルドカードによる計画選択や動的な計画選択が関係する場合は、これを使用し、エントリー内で適切なトランザクションを組み合わせます。使用頻度の低いトランザクションが、デフォルトでプールに格納されるようにします。ただし、ワイルドカード文字を使用してトランザクション ID を定義すると、DB2ENTRY (トランザクション・グループを表さない) ごとに統計が収集されるため、トランザクション単位で CICS Db2 統計を収集することができなくなります。

Db2 テーブルと Db2 サブシステムの調整に関する情報や、Db2 アプリケーションを調整するときの一般的な考慮事項については、「[Db2 for z/OS 製品資料内の『Db2 のパフォーマンス管理』](#)」を参照してください。

パフォーマンスおよびメンテナンスのための許可 ID の選択

Db2 に接続するプロセス、またはサインオンするプロセスは、Db2 アドレス・スペースのセキュリティチェックに使用できる許可 ID という名前の短い Db2 ID を、1 つ以上提供する必要があります。すべてのプロセスは 1 次許可 ID を提供する必要があり、オプションとして 2 次許可 ID を 1 つ以上提供することもできます。スレッドを Db2 に取り込む CICS トランザクションはプロセスとみなされるため、許可 ID を提供する必要があります。

「*Db2 Guide*」の資料に、CICS トランザクションで使用されるスレッドが Db2 にサインオンするときに、このトランザクションから Db2 に渡される許可 ID の選択方法とセットアップ方法を記載しています。トランザクションの許可 ID は、トランザクションが使用するスレッドのリソース定義内の属性によって決定されます。この定義は、エントリー・スレッドの場合は DB2ENTRY 定義、プール・スレッドまたはコマンド・スレッドの場合は DB2CONN 定義です。

CICS トランザクションが使用する許可 ID のタイプを選択する場合は、パフォーマンスおよびメンテナンスに関する考慮事項に配慮する必要があります。

許可 ID に関するパフォーマンスの考慮事項

パフォーマンスの観点からは、AUTHTYPE 属性についてオプション USERID、OPID、TERM、TX、または GROUP を 1 つ選択した場合、Db2 スレッドを使用する任意の CICS トランザクションが、このスレッドを直前に使用したトランザクションと異なる許可 ID を持つ可能性が生じます。これにより、サインオン処理が発生します。SIGN オプションを選択した場合、または AUTHTYPE 属性でなく AUTHID 属性を使用した場合、CICS トランザクションは同じ許可 ID を持ちます。スレッドを使用しているトランザクションが同じ許可 ID を持つ場合、サインオン処理はバイパスされます。

しかし、オプション USERID、OPID、TERM、TX、または GROUP を使用するとパフォーマンスは低下しますが、より細かい Db2 セキュリティチェックが可能になります。例えば、トランザクションのスレッドに AUTHTYPE(USERID) を定義した場合、Db2 セキュリティチェックでは、トランザクションを使用している各ユーザーの CICS ユーザー ID が使用されます。トランザクションのスレッドに AUTHTYPE(SIGN) を定義した場合は、Db2 セキュリティチェックでは、CICS 領域全体に定義された SIGNID が使用されます。そのため、Db2 は、CICS 領域に Db2 リソースへのアクセスが許可されているかどうかだけを検査します。すべてのトランザクションに同じ許可 ID を設定するオプションの 1 つを使用する場合は、CICS トランザクシ

ン接続セキュリティーを使用して、トランザクションへのアクセスを制限する必要があります(「Db2 Guide」資料の『DB2 関連の CICS トランザクションへのユーザー・アクセスの制御』を参照)。

計画に対する代わりのソリューションは、Db2 の GRANT コマンドを使用して、計画の EXECUTE 権限を PUBLIC に与え、サインオン処理をバイパスさせることです。Db2 は変更された許可 ID を無視します。この方法では、CICS Db2 接続機能内で同じ処理が発生するため、一定の許可 ID およびトランザクション ID を使用する方法ほど効率的ではありません。このソリューションを使用すると Db2 内で計画のセキュリティー検査を実行できなくなるため、Db2 サブシステムのセキュリティーに関する考慮事項では、このソリューションの使用が禁止されていることがあります。

許可 ID に関するメンテナンスの考慮事項

メンテナンスの観点からは、許可 ID のオプション USERID、OPID、TERM、TX、または GROUP を使用する場合は、より多くの許可 ID に Db2 の権限を付与する必要があります。例えば、CICS トランザクションが Db2 で計画を実行する場合、このトランザクションのスレッドが AUTHTYPE(USERID) を使用して定義されていれば、トランザクションを使用できる各ユーザーのすべての CICS ユーザー ID に Db2 で計画を使用する権限を付与する必要があります。SIGN オプションを使用する場合、または AUTHTYPE 属性でなく AUTHID 属性を使用する場合は、より少数の許可 ID に権限を付与する必要があります。

ただし、前述のように、使用する許可 ID 数が制限されている場合は、Db2 自身のセキュリティー検査の精度が低下します。セキュリティーを優先する必要があるが、Db2 システムの高度な保守に不安があるという場合には、CICS ユーザーに 2 次許可 ID を設定する方法があります。「Db2 Guide」資料の『CICS トランザクション用に 2 次許可 ID を提供する』に、この方法を記載しています。RACF グループを作成して、CICS ユーザーをこの RACF グループに接続することができます。トランザクションで使用されるスレッドの DB2ENTRY 定義の GROUP 属性を使用して、RACF グループが Db2 に渡される 2 次 ID の 1 つになるようにします。その後、RACF グループに Db2 権限を付与します。CICS ユーザーの Db2 権限を削除するには、RACF グループからこれらの権限を解除します。このソリューションを使用する場合、Db2 セキュリティー検査で、各 CICS ユーザーに Db2 内のリソースへのアクセス権限があることを確認できますが、各 CICS ユーザー ID に権限を特に付与する必要はありません。

ロギング

ロギング・コストには、カップリング・ファシリティへの同期アクセスによって生じた可変コストのいくつかが含まれているため、ここではこれらのコストはミリ秒単位の CPU 時間として記述されています。

リカバリー可能リソースにアクセスするコストを確認すると、1 次ストレージへのログ・バッファの書き込みのコストは、API コストから分離されています。FORCE および NOFORCE は、システム・ログ・バッファへの書き込み操作の 2 つのタイプです。

- FORCE 操作では、ログ・バッファを書き出し、不揮発性にすることが要求されます。この要求を行うトランザクションは、プロセスが完了するまで中断されます。ログは即時に書き出されませんが、内部アルゴリズムを使用して据え置かれます。ログへの最初の強制書き込みによって、据え置かれたログ・フラッシュから時間の計測が始まります。ログオン強制を要求する以降のトランザクションは、そのデータをバッファに入れて、元の据え置き時間が経過するまで中断されます。これにより、ログ要求のバッファリングが許可され、ログ・バッファの書き込みのコストが数多くのトランザクション間で共有されることになります。
- NOFORCE 操作では、ログ・バッファにデータが入れられ、FORCE 操作が要求されたとき、またはバッファがいっぱいになったときに 1 次ストレージに書き込まれます。

ログ・バッファの書き込みのコストは、以下のいずれの状態が該当するかによって異なります。

- 書き込みがカップリング・ファシリティに対して同期している。
- 書き込みがカップリング・ファシリティに対して非同期的。
- ステージング・データ・セットが使用されている。
- DASD のみのロギングが使用されている。

カップリング・ファシリティへの同期書き込み

サイズが 4 K より小さい書き込みは、一般的に同期書き込みになります。同期書き込みでは、カップリング・ファシリティに直接アクセスする特別な命令を使用します。命令は、カップリング・ファシリティにアクセスして戻るまで継続されます。このアクセス時間は、「CF サービス時間」と呼ばれ、カップリング・ファシリティの速度およびカップリング・ファシリティへのリンクの速度の

両方によって異なります。CF サービス時間は、202 ページの図 26 に示されるように、RMF III を使用して モニターすることができます。同期書き込みの場合、CF サービス時間が変更されるとアクセスの CPU コスト も変更されます。このことは、非同期書き込みの場合には該当しません。

CF への非同期書き込み

非同期書き込みでは、同期書き込みで使用される命令と同じ命令は使用されません。非同期ログ書き込みを行う CICS タスクは、別のタスクに制御を渡し、操作はロガー・アドレス・スペースによって完了されます。

ロギングについて詳しくは、195 ページの『CICS ロギングおよびジャーナリング: パフォーマンスおよび調整』を参照してください。

同期点

同期点コストは、全体的なトランザクション・コストに要因として組み込む必要があります。同期点での処理量は、作業単位 (UOW) 時に関連するリソース・マネージャーのタイプの数によって異なります。このため、コストも異なる可能性があります。

通常、同期点では、UOW 時に関連するすべてのリソース・マネージャーを呼び出します。これらは、書き出される前にログ・バッファにデータを入れる必要がある場合があります。例えば、リカバリー可能一時データ (TD) は、同期点までログ・バッファへのデータの書き込みを据え置きます。リカバリー・マネージャー自体は、コミット・レコードをログ・バッファに入れて、強制書き込みを要求します。これらの理由から、同期点の正確なコストを判別することは困難ですが、以下の情報をガイドとして使用する必要があります。

同期点は、以下のように分割することができます。

パート	値
基本コスト	5.0
ログ・バッファへのコミット・レコードの書き込み	2.0
UOW で使用されるそれぞれの RM ごと	2.5
書き込みログ・バッファ	194 ページの『ロギング』を参照してください。

この表は、ローカル・リソースのみの場合の同期点コストを 1K 命令単位で示しています。分散リソースが更新された場合は、通信コストを追加する必要があります。

リカバリー可能リソースが更新された場合、次に示すように、コストはトランザクション終了コストのみになります。

トランザクション・コスト	アセンブラー	COBOL
終了	6.2	10.0

注: トランザクションの初期化コストは、トランザクション付加の開始 から CICS アプリケーション・コードの開始までの間で計算されます。リカバリー可能リソースが更新された場合、同期点コストを終了コストに追加する必要があります。

CICS ロギングおよびジャーナリング: パフォーマンスおよび調整

個々の CICS ログ・ストリームは、カップリング・ファシリティ内のログ構造、または CICS ログ・マネージャーがサポートする MVS システム・ロガーの DASD 専用オプションのいずれかを使用できます。ログ・マネージャーのパフォーマンスは、いくつかの方法で調整できます。

CICS ログ・ストリームで使用されるストレージのタイプについての詳細は、[CICS 用のロガー環境を定義する](#)を参照してください。

カップリング・ファシリティ内のログ構造を使用する場合、各ログ・ストリームの定義方法 (使用法に基づく) については、[カップリング・ファシリティまたは DASD 専用ログ・ストリーム](#)を参照してください。

い。カップリング・ファシリティおよび DASD 専用ログ・ストリームの相対的なパフォーマンスについては、194 ページの『ロギング』を参照してください。

カップリング・ファシリティを使用する場合は、スタンドアロン・モデルを使用できます。また、統合結合マイグレーション機能 (ICMF) を使用して、論理区画 (LPAR) でカップリング・ファシリティ・サービスを提供することもできます。つまり、カップリング・ファシリティおよび MVS は障害から独立していないため、ステージング・データ・セットを使用する必要があります。

パフォーマンスおよびロギングのための調整に関する付加的なアドバイスと例については、以下の資料とサブトピックを参照してください。

- IBM Redbooks 資料「[Systems Programmer's Guide to: z/OS System Logger](#)」(SG24-6898)。この資料には、z/OS System Logger についての詳細な説明が記載されています。また、CICS など、これを利用する製品で最適のパフォーマンスを得られるようにするためのセットアップ方法についても説明しています。
- IBM Redpaper [Performance Considerations and Measurements for CICS and System Logger](#) (REDP-3768)。この資料は、Redbook 資料の補足として書かれたもので、CICS と z/OS System Logger の間の対話についての付加的な説明が記載されています。また、さまざまな CICS および System Logger の構成例を掲載し、その調整プロセスについて説明しています。
- IBM サポート資料の「[Useful CICS Logger information](#)」。この資料は、CICS および z/OS System Logger のパフォーマンス評価およびトラブルシューティングについての 2 つのプレゼンテーションに対するリンクを提供しています。
- 『z/OS Management Facility Configuration Guide』の『システム・ロガー用結合データ・セットの定義』。
- 『z/OS MVS Setting Up a Sysplex』の『IXCMIAPU ユーティリティの使用例』。

CICS ログ・マネージャー

CICS ログ・マネージャーは、CICS の実行中にジャーナルを作成、制御、および検索するための機能を提供します。ジャーナルは、後でデータやイベントを再構成するために必要になる可能性がある情報を発生順に記録することを意図しています。例えば、監査証跡として使用したり、バックアップの目的でデータベースの更新、追加、および削除を記録したり、システム内のトランザクション・アクティビティを追跡したりするためにジャーナルを作成できます。

CICS ログ・マネージャーは、MVS システム・ロガーによって提供されるサービスを使用して、すべてのロギングおよびジャーナリングを制御します。CICS ログ・マネージャーは、以下をサポートします。

- CICS システム・ログ
- 順方向リカバリー・ログ
- ファイル制御操作および端末管理操作の自動ジャーナル
- ユーザー・ジャーナル

MVS システム・ロガーは、以下のサービスを提供します。

- メディア管理およびアーカイブ
- ログ・レコードへの直接アクセスおよび順次アクセスによるログ・データ可用性。

ログ・ストリーム・ストレージ

ログ・ストリームは、一連のデータ・ブロックです。各ログ・ストリームは、独自のログ・ストリーム ID (ログ・ストリーム名 (LSN)) によって識別されます。CICS システム・ログ、順方向リカバリー・ログ、およびユーザー・ジャーナルは、特定の MVS ログ・ストリームにマップされます。CICS 順方向リカバリー・ログとユーザー・ジャーナルは、一般ログと呼ばれ、システム・ログと区別されています。

各ログ・ストリームはデータの一連のブロックから成り、CICS ログ・マネージャーによって内部で次の 3 つの異なるタイプのストレージに区分されます。

1. 1 次ストレージ。ログ・ストリームに書き込まれた最新のレコードを保持します。1 次ストレージは、以下のいずれかで構成されます。
 - カップリング・ファシリティ内の構造。(カップリング・ファシリティを使用すると、異なる MVS イメージ内の CICS 領域が同じ一般ログ・ストリームを共用できます。) カップリング・ファシリティ

ーに書き込まれたログ・データは、データ・スペースまたはステージング・データ・セットのどちらかにもコピーされます。

- システム・ロガーと同じ MVS イメージ内のデータ・スペース。データ・スペースに書き込まれたログ・データは、ステージング・データ・セットにもコピーされます。

2. 補助ストレージ。ログ・ストリーム用の 1 次ストレージが満杯になると、古いレコードが自動的に補助ストレージにスピルします。補助ストレージはストレージ管理サブシステム (SMS) によって管理されるデータ・セットで構成されます。各ログ・ストリームは、ログ・ストリーム名 (LSN) によって識別され、独自のログ・データ・セットに書き込まれます。
3. 3 次ストレージ。アーカイブ・ストレージを形成し、階層ストレージ・マネージャー (HSM) ポリシーの指定に従って使用されます。オプションで、古いレコードを 3 次ストレージにマイグレーションすることができます。3 次ストレージは、DASD データ・セットまたはテープ・ボリュームのいずれかです。

198 ページの図 24 と 199 ページの図 25 は、CICS システム・ロガーによって使用されるストレージのタイプを示しています。

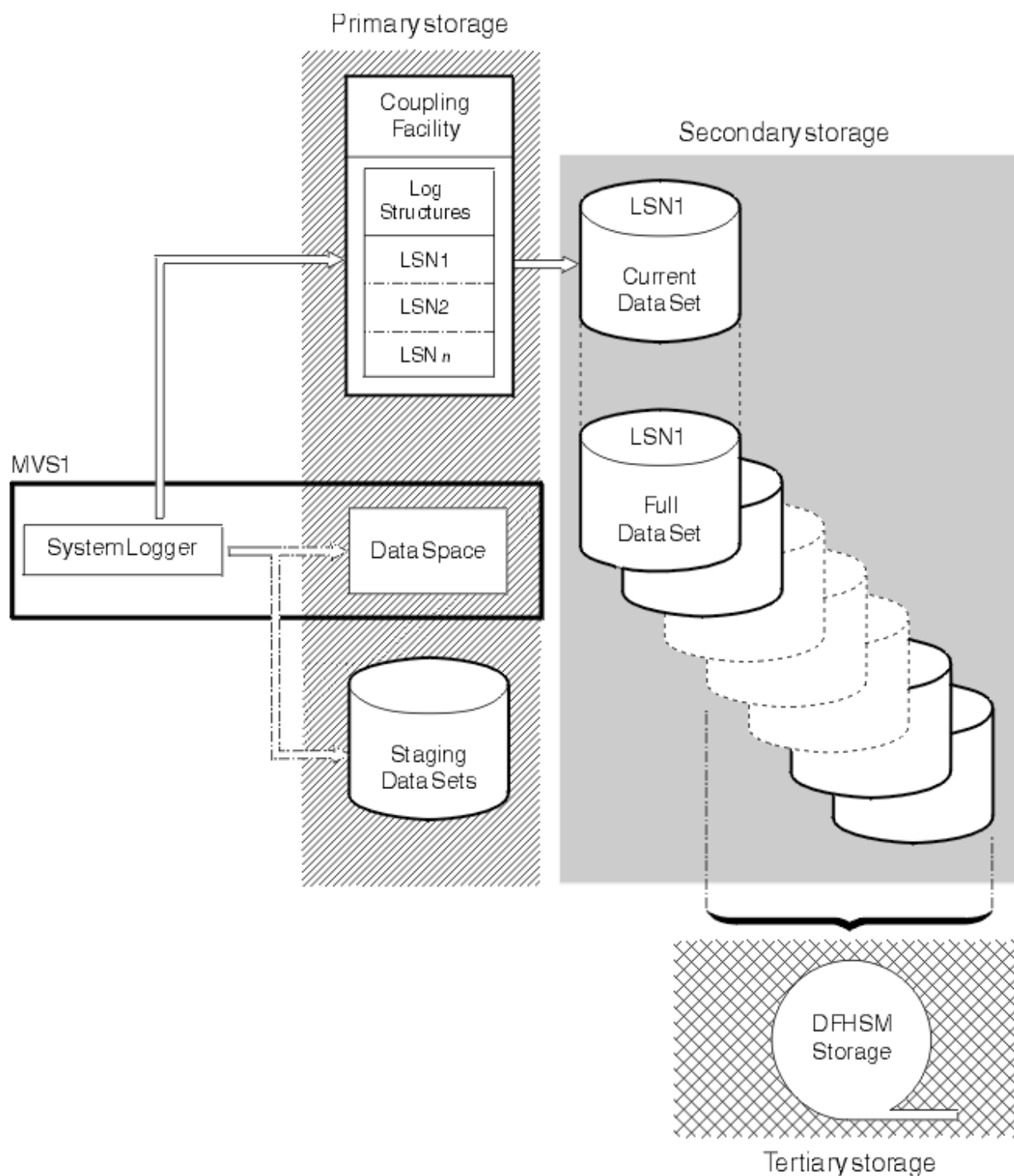


図 24. MVS システム・ログで使用するストレージのタイプ

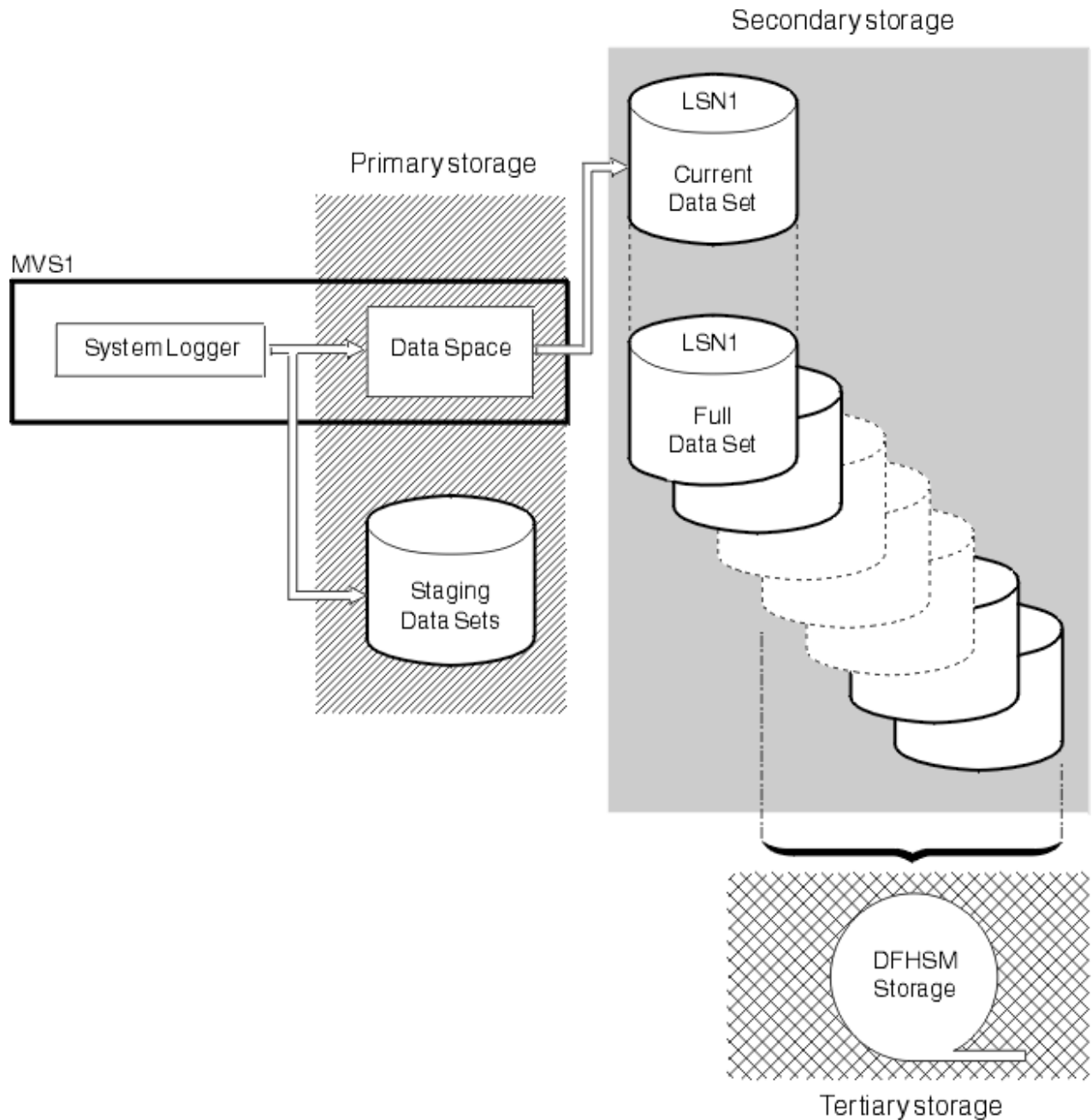


図 25. MVS システム・ロガーで使用するストレージのタイプ

ジャーナル・レコード

ジャーナル・レコードは、ユーザー・アプリケーション・プログラムから直接に、またはユーザー・アプリケーションの代わりに CICS 管理プログラムから、ログ・ストリームに書き込まれます。

ジャーナル・レコードは、ユーザー・アプリケーションから **WRITE JOURNALNAME API** コマンドを使用して書き込むことができます。アプリケーション・プログラムから **SET JOURNALNAME SPI** コマンドを使用して、ジャーナルを有効または無効にします。

ログ・ストリーム内のジャーナル・データへのアクセスは、MVS サブシステム・インターフェース (SSI) の **LOGR** を介して提供されます。バッチ・ジョブ JCL 内のログ・ストリーム用の DD で、**SUBSYS** パラメーターとサポートするオプションを指定すると、既存のユーザー・プログラムで一般ログ・ストリームを読み取ることができます。**SUBSYS** パラメーターで **LOGR** サブシステム名を指定した場合、**LOGR** は SSI でデータ・セットのオープン要求および読み取り要求をインターセプトし、それらをログ・ストリーム・アクセスに変換することができます。

SUBSYS パラメーターで指定されたオプションによって、一般ログ・ストリームのジャーナル・レコードは、2 とおり方法で表示されます。

- CICS/ESA 4.1 以前で使用されたレコード・フォーマット (旧ユーティリティー (COMPAT41 オプションで選択された) との互換性用)。
- CICS Transaction Server for z/OS フォーマット (ログ・レコード情報へのアクセスを必要とする新規またはアップグレードされたユーティリティー 用)。

CICS システム・ログ・レコードは、CICS Transaction Server for z/OS フォーマットのみ使用可能であるため、CICS Transaction Server for z/OS 以前のリリースでシステム・ログ・レコードを処理していたすべてのユーティリティーが、このフォーマットを処理するように変換されていることを確認する必要があります。

ジャーナル・レコードは、ユーザー作成のプログラムでオフラインで読み取ることができます。以下のように、プログラム・コードに特定のステートメントを組み込むことによって、そのようなプログラムに必要な DSECT を生成できます。

- 一般ログの CICS Transaction Server for z/OS フォーマットのレコードの場合、オフラインのユーザー作成プログラムは、INCLUDE DFHLGGFD ステートメントを組み込むことによってジャーナル・レコードをマップできます。このステートメントは、アセンブラー・バージョンの DSECT を生成します。
- COMPAT41 オプションを使用してフォーマットされたレコードの場合、オフラインのユーザー作成プログラムは、DFHJCR CICSYST=YES ステートメントを発行することによってジャーナル・レコードをマップできます。この結果、DFHJCRDS DSECT がプログラムに組み込まれます。

生成された DSECT は、COPY DFHJCRDS ステートメントによって CICS プログラム用に取得される DSECT と同じです。唯一の相違点は、フィールドの前に CICS ストレージ・アカウント領域がないことです。この DSECT は、ジャーナル・レコードを、CICS ストレージ域ではなく、ブロックに直接マップすることを意図しています。

ロガー環境のモニター

CICS は各ジャーナルおよびログ・ストリームに書き込まれたデータに関する統計を収集します。このデータを使用すると、単一領域のアクティビティーを分析できます。ただし、一般ログ・ストリームは複数の MVS イメージで共用できるため、MVS で生成される統計を調査する場合により便利です。

このタスクについて

MVS システム・ロガーは、接続された各ログ・ストリームの統計を含む SMF Type 88 レコードを書き込みます。MVS が SYS1.SAMPLIB で提供するサンプル・レポート・プログラムの IXGRPT1 は、そのまま使用することも、要件に合わせて変更することもできます。また、その他の SMF レポート・プログラムを使用することもできます。SMF タイプ 88 レコードおよびサンプル・レポート・プログラムについては、「[z/OS MVS システム管理機能 \(SMF\)](#)」を参照してください。

日常的にモニターする主なイベントは、次のとおりです。

- カップリング・ファシリティのログ・ストリームの場合は、「structure full」イベントの数
- DASD 専用ログ・ストリームの場合は、「staging data set full」イベントの数

これらのイベントが頻繁に発生する場合は、ロガーが着信データに匹敵する速度で補助記憶域にデータを書き込むことができず、CICS がデータを書き込めるようになるまで待機する原因となります。

手順

1. イベント・フル状態の結果発生する問題を解決するために、次のソリューションを考慮します。
 - a) 1 次ストレージのサイズ (つまりカップリング・ファシリティ構造のサイズ、または DASD 専用ログ・ストリームの場合はステージング・データ・セットのサイズ) を大きくして、ロガー・ロードのスパイクを平滑化します。
 - b) 多数のジャーナルまたは順方向リカバリー・ログを同じストリームにマージしないことにより、ログ・ストリームに書き込まれるデータを削減します。

- c) **HIGHOFFLOAD** しきい値 (システム・ロガーが 1 次ストレージからオフロード・データ・セットへのデータ・オフロードを開始する値) を小さくします。
- d) オフロード・データ・セットのサイズを検討します。発生する「DASD シフト」が多すぎて、新規データ・セットが割り振られることがないように、オフロード・データ・セットは十分大きな値にする必要があります。1 時間あたりの DASD シフト回数が 1 回以下にしてください。DASD シフト数をモニターするには、SMF88EDS レコードを使用します。
- e) デバイス I/O 統計を調べて、オフロード・データ・セットに使用される I/O サブシステムに競合が発生していないかを確認します。
- f) より高速な DASD デバイスを使用します。

最適な CICS システム・ログ・パフォーマンスを達成できるのは、CICS が不要になったログ・テール・データを、MVS システム・ロガーによって補助記憶域に書き込まれる前に削除できる場合です。このようになっているかをモニターするには、レポート・プログラムで SMF88SIB および SMF88SAB SMF タイプ 88 レコードの値を調べることができます。これらの値は、ログ・データに関連した役立つ情報を提供します。

SMF88SIB

DASD オフロード・データ・セットに最初書き込まれることなく、1 次ストレージから削除されたデータ。システム・ログ・ストリームの場合、この値は SMF88SAB の値に対して通常は大きくします。一般ログ・ストリームの場合、この値は通常ゼロにします。

SMF88SAB

DASD オフロード・データ・セットに書き込まれたあとに 1 次ストレージから削除されたデータ。システム・ログ・ストリームの場合、この値は SMF88SIB の値に対して通常は小さくします。一般ログ・ストリームの場合、この値は通常は大きくします。

注：SMF インターバルでは、1 次ストレージから削除されたバイトの合計数 (SMF88SIB と SMF88SAB の和) と、補助記憶域に書き込まれたバイトの合計数が一致しないことがあります。データはオフロード・データ・セットにのみ書き込まれ、**HIGHOFFLOAD** しきい値制限に達すると 1 次ストレージから削除されます。

2. SMF88SAB レコードの CICS システム・ログ値が一般に大きい場合は、次のようにしてください。
 - a) ログ・ストリームの MVS 定義で、RETPD=dddd が指定されていないことを確認します。MVS RETPD パラメーターについて詳しくは、[補助ストレージの管理](#)を参照してください。
 - b) 稼働時間の長いトランザクションによる、同期点を持たないリカバリー可能な更新が行われていないことを確認します。
 - c) 1 次ストレージのサイズを大きくします。
 - d) **HIGHOFFLOAD** しきい値を大きくします。
 - e) **AKPFREQ** システム 初期設定パラメーターの値を小さくします。

カップリング・ファシリティへのデータの書き込み: パフォーマンスの考慮

アプリケーション設計レベルで、カップリング・ファシリティに書き込まれる平均ブロック・サイズが CICS ログ・マネージャーのパフォーマンスに影響を与えることを考慮する必要があります。

カップリング・ファシリティに書き込まれているデータの平均ブロック・サイズが 4 KB 未満の場合、書き込み要求は同期で処理されます。操作は CICS と同期するだけでなく、カップリング・ファシリティへのアクセスに使用される命令も同期し、構造内にデータが配置されている間は、処理が実行されます。このため、高速プロセッサと低速カップリング・ファシリティを混在させるのは賢明ではありません。特定のカップリング・ファシリティへのアクセス時間が一定である場合に、同期アクセスを行うと、プロセッサが高速であるほど、要求で使用されるプロセッサ・サイクル数が多くなります。

カップリング・ファシリティに書き込まれているデータの平均ブロック・サイズが 4 KB よりも大きい場合、書き込み要求は非同期に処理されます。CICS タスクは制御を放棄し、書き込み要求が満たされた場合に、MVS システム・ロガーはイベント制御ブロック (ECB) をポストします。これにより、非同期要求は同期要求よりも完了に時間がかかることがあります。

必要な場合 (例えば、サブチャネルが使用中の場合)、サブシステムによって同期要求が非同期要求に変更されることがあります。変更された要求は、RMF III レポートに CHNGD として表示されます。[202 ページの図 26](#) は、RMF レポートから抽出されたデータで、カップリング・ファシリティ構造への同期書き

込みと非同期書き込みの数を示しています。このレポートには、システム名、要求の総数、および1秒間の平均要求数が記載されています。また、要求タイプごとに、要求数、全要求に対する要求数の割合、平均処理時間、および標準偏差が記載されています。

STRUCTURE NAME = LOG_FV_001			TYPE = LIST			
	# REQ			REQUESTS -----		
SYSTEM	TOTAL		#	% OF	-SERV	TIME(MIC) -
NAME	AVG/SEC		REQ	ALL	AVG	STD_DEV
MV2A	15549	SYNC	15K	95.3%	476.1	339.6
	27.87	ASYN	721	4.6%	3839.0	1307.3
		CHNGD	12	0.1%	INCLUDED	IN ASYN

図 26. カップリング・ファシリティへの同期および非同期書き込み数を示す RMF レポート

注: これは、カップリング・ファシリティ構造を使用するログ・ストリームにのみ適用されます。

ログ・ストリーム数の定義: パフォーマンスの考慮

カップリング・ファシリティ・スペースは、カップリング・ファシリティ・リソース管理 (CFRM) ポリシーによって、最大 255 個の構造に分割されます。複数のログ・ストリームで同じ構造を使用できます。同じようなサイズのデータ・レコードを書き込むアプリケーションで使用されるログ・ストリームは、同じ構造を共用するようにします。その理由は、構造定義の **AVGBUFSIZE** および **MAXBUFSIZE** パラメータで定義された値に関連します。

一般に、構造あたりのログ・ストリーム数が多いほど、CICS ログ・マネージャーの効率およびパフォーマンスに影響する各パラメータを調整することが困難になります。

カップリング・ファシリティ構造を定義する場合、その構造は2つの領域に分割されます。一方の領域はリスト・エントリを保持し、もう一方の領域はリスト・エレメントを保持します。

リスト・エレメントはロギングされたデータの集まりであり、256 バイト長または 512 バイト長です。リスト・エントリはリスト・エレメントに対する索引ポインターです。ログ・レコードにつき1つのリスト・エントリが存在します。ログ・レコードにつき少なくとも1つのエレメントが存在します。

65276 より大きな値の **MAXBUFSIZE** を定義した場合、データは 512 バイトのエレメントに書き込まれます。65276 以下の値の **MAXBUFSIZE** を定義した場合、データは 256 バイトのエレメントに書き込まれます。このパラメータの最大値は 65532 です。

リスト・エントリおよびリスト・エレメントで占有される領域の比率は、次の式で計算される値によって決まります。

$$\text{AVGBUFSIZE} / \text{element size}$$

計算された値は比率 $nn:1$ を表します (nn はエレメント・ストレージを、1 はエントリ・ストレージを表します)。最小の比率は 1:1 です。

この比率は、ロギング要件および動作が異なる多数のアプリケーションを組み合わせた場合には不適切なことがあるため、パフォーマンスに重大な影響を及ぼします。

エレメント/エントリ率および構造あたりのログ・ストリーム数

AVGBUFSIZE は構造レベルで設定され、構造全体に対する比率を示します。多数のアプリケーションがログ・ストリームに著しく異なるインターバルで、著しく異なるサイズのデータを書き込んでいる場合、一部のアプリケーションでは予期せぬ DASD オフロードが発生し、プロセッサの使用量が増えることがあります。

ログ・ストリームがまだ **HIGHOFFLOAD** しきい値に達していない可能性があるため、DASD オフロードは発生しないと想定されます。一般に、構造あたりのログ・ストリーム数が多いほど、ログ・ストリームを使用する特定のアプリケーションに対してエレメント/エントリ比率が不適切になる可能性が高まります。

各ログ・レコードはエントリを構造のリスト・エントリ領域に配置し、データはリスト・エレメント領域の1つまたは複数のエレメントとしてロードされます。リスト・エントリ領域が容量の 90% を超える場合は、すべてのログ・ストリームが DASD にオフロードされます。ログ・ストリームの現在の使用

率に関係なく、DASD オフロードが開始し、**HIGHOFFLOAD** しきい値と **LOWOFFLOAD** しきい値の差に等しいデータ量がオフロードされるまで継続します。

例えば、ログ・ストリーム A の使用率が 50% のみであっても、リスト・エン트리領域が容量の 90% を超えることがあります。**HIGHOFFLOAD** しきい値は 80%、**LOWOFFLOAD** しきい値は 60% です。ログ・ストリーム A が **HIGHOFFLOAD** しきい値に達していない場合、または **LOWOFFLOAD** しきい値に達していない場合でも、データはログ・ストリームの 20% がオフロードされるまでオフロードになります。これは、80% と 60% の差によります。オフロード処理が完了すると、ログ・ストリーム A は 30% の使用率になります (50% - 20%)。

したがって、ジャーナル書き込み要求の発行数が少ないアプリケーションで使用されるログ・ストリームは、同じ構造内の他のログ・ストリームを使用する別のアプリケーションによってジャーナル書き込み要求が何度も送信されるため、DASD にオフロードされることがあります。

ただし、複数のログ・ストリームが同じ構造を共用している場合、リスト・エン트리・ストレージの使用率が 90% に達するのは、すべてのログ・ストリームでロギング・アクティビティのサイズがほぼ同じである場合のみです。

動的再分割および DASD オフロードの頻度

カップリング・ファシリティ構造内のスペースは、構造に接続されたすべてのログ・ストリーム間で動的に分割されます。接続されているログ・ストリームが多いほど、DASD オフロードがより頻繁に発生する可能性があります。

ログ・ストリームをカップリング・ファシリティ構造に接続したり切断したりすると、構造は動的に再分割されます。つまり、構造内のスペースは構造に接続されたすべてのログ・ストリーム間で分割されます。接続されているログ・ストリームが多いほど、各ログ・ストリームに割り振られるスペースは少なくなります。ログ・ストリームのスペースが小さくなるということは、ログ・ストリームの **HIGHOFFLOAD** しきい値のパーセンテージに達する回数が増えることを意味するため、結果として、DASD オフロードの頻度が高まる可能性があります。

ほとんどの環境に対して、**MAXBUFSIZE** の値は 64000 が適切です。

MAXBUFSIZE を 65276 よりも大きな値に設定した場合、エレメント・サイズは 512 バイトです。512 バイトのエレメントの場合、スペースが使用されず、したがってログ・レコードの最終エレメント末尾への埋め込みによってスペースが浪費される可能性が高まります。この状態は、レコードがより大きく、システムがよりビジーであるほど発生しにくくなります。

AVGBUFSIZE および **MAXBUFSIZE** は、カップリング・ファシリティ構造を定義するために実行される IXCMIAPU プログラムで使用されるパラメーターです。詳しくは、『[z/OS MVS Setting Up a Sysplex](#)』の『[管理データ・ユーティリティ](#)』を参照してください。

構造のログ・ストリームへのデータ・トラフィックおよびログ・ストリームから DASD へのデータ・トラフィックをモニターする場合は、次の機能を使用できます。

- CICS ログ・ストリーム統計。これらの統計は、合計バイト値を合計書き込み値で割って計算できる「1 回の書き込みで書き込まれる平均バイト数」を示す値など、さまざまな統計情報を提供します。この情報は、**AVGBUFSIZE** の値を調整する場合に役立ちます。
- 合計エレメント数の値を合計エン트리数の値で割って計算できる「エン트리あたりのエレメント数」を示す値などの、RMF によって提供される統計。この統計を使用すると、ログ・ストリームのアクティビティをエレメント単位で確認できます。RMF は同様に、同期または非同期に処理された要求の比率を構造単位で通知します。同期に処理されたログ・ストリーム要求を保持する構造を、非同期に処理されたログ・ストリーム要求を保持する構造から分離することができます。
- SMF88 レコード。オフロードされたバイト数など、さまざまな統計情報を提供します。

ログ・ストリーム定義での LOWOFFLOAD および HIGHOFFLOAD パラメーター

ログ・ストリーム使用量 (カップリング・ファシリティまたはステージング・データ・セット内) がその **HIGHOFFLOAD** 制限に達すると、ログ・ストリームからのデータが DASD データ・セットにオフロードされることがあります。オフロードされるデータの量は、**LOWOFFLOAD** 制限を使用して決定されます。

HIGHOFFLOAD 制限は、ログ・ストリームが定義されている場合に指定されます。

この情報は、カップリング・ファシリティ構造を使用するログ・ストリームを使用している場合に関係があります。ただし、この手引きの大部分は DASD 専用ログ・ストリームにも適用されます。

DASD 専用ログ・ストリームの詳細については、[208 ページの『DASD 専用ロギング』](#)を参照してください。

システム・ログの場合、削除とマークされたすべてのレコードが物理的に削除されます。そのあとに、**LOWOFFLOAD** 制限に達しなかった場合は、**LOWOFFLOAD** に達するまで、最も古いアクティブ・レコードが DASD にオフロードされます。一般ログの場合、**LOWOFFLOAD** 制限に達するまで、最も古いデータが DASD にオフロードされます。

ログ・ストリームの **HIGHOFFLOAD** しきい値 (および環境によっては **LOWOFFLOAD** しきい値) に達しなかったにもかかわらず、ログ・ストリーム・データ・セットからデータがオフロードされることもあります。

- ステージング・データ・セットの **HIGHOFFLOAD** しきい値に達した場合。ステージング・データ・セットのサイズがログ・ストリームに比べて相対的に小さい場合は、ログ・ストリーム・データ・セットの **HIGHOFFLOAD** しきい値に達する前に、ステージング・データ・セットの **HIGHOFFLOAD** しきい値に達します。
- ログ・ストリームのリスト・エントリ領域が容量の 90% に達した場合。

これらの場合、ログ・ストリームからオフロードされたデータ量は次の式によって決まります。

(Current utilization or HIGHOFFLOAD, whichever is the greater) - LOWOFFLOAD

これはオフロードされたログ・ストリーム・データ・セットの割合です。

HIGHOFFLOAD および **LOWOFFLOAD** は、ログ・ストリーム・モデルおよび明示的に名前が指定された各ログ・ストリームを定義するために実行される IXCMIAPU プログラム用のパラメーターです。詳しくは、[『z/OS MVS Setting Up a Sysplex』の『管理データ・ユーティリティ』](#)を参照してください。

SMF88 レコードおよび RMF は、これらのパラメーターの調整に役立つさまざまな統計情報を提供します。

1 次システム・ログ

活動キーポイントが発生した場合、CICS は 1 次システム・ログのテールである DFHLOG を削除します。つまり、直前の活動キーポイントより古い完了済みの作業単位データは、削除されます。UOW が現在の活動キーポイント・インターバルでロギングを実行しなかった場合、直前の活動キーポイントよりも古い不完全な各作業単位データは、2 次システム・ログの DFHSHUNT に移動します。

DASD オフロードの頻度を最小にするには、現在の活動キーポイント・インターバル中に生成されるシステム・ログ・データ、および直前の活動キーポイントで削除されなかったデータが、常にカップリング・ファシリティ構造内に存在するようにしてください。このデータが DASD にオフロードされないようにするには、次の設定を使用することができます。

- **HIGHOFFLOAD** を 80 に設定します。
- **AKPFREQ** パラメーターに、例えば 4000 などの小さな値を指定して、活動キーポイント間で生成されるログ・データの量を最小にします。
- **LOWOFFLOAD** の値が、次の値の合計に必要なスペースよりも大きくなるようにします。

1. 1 つの完全な活動キーポイント・インターバル中に生成されたシステム・ログ・データ
2. 稼働時間が最も長いランザクションによって (同期点間で) 生成されたシステム・ログ・データ

次の式のどちらかを使用して、**LOWOFFLOAD** の値を計算します。

```
LOWOFFLOAD = ((trandur * 90) / (akpintvl + trandur)) + 10  
[where RETPD=0 is specified]
```

または

```
LOWOFFLOAD = (trandur * 90) / (akpintvl + trandur)  
[where RETPD=dddd is specified]
```

ここで、

- akpintvl は活動キーポイント間のインターバルです。この値はワークロードに従って変わるため、次のようにピーク・ワークロード・アクティビティーに基づいて計算します。

$$\text{akpintvl} = \text{AKPFREQ} / ((N1 * R1) + (N2 * R2) + (Nn * Rn))$$

ここで、

- N1、N2、... Nn は、トランザクションごとのトランザクション・レートです(トランザクション/秒)。
- R1、R2、... Rn は、各トランザクションによって書き込まれたログ・レコード数です。
- trandur は、通常のワークロードの一部として実行された、稼働時間が最も長いトランザクションの(同期点間の)実行時間です。

この期間が akpintvl 値よりも長い場合は、次のいずれかを実行できます。

- **AKPFREQ** の値を大きくして、akpintvl 値を大きくします(これにより、カップリング・ファシリティ構造のサイズが許容できないほど大きくなることはありません)。
- 同期点の頻度が高まるように、アプリケーション・ロジックを変更します。
- より短いトランザクション期間に基づいて構造サイズを計算し、稼働時間の長いトランザクションが使用されている場合でも DASD オフロードが発生するようにします。

DFHLOG LOWOFFLOAD パラメーター値の経験上の適正範囲は 40% から 60% です。値が小さすぎると、MVS ロガー・オフロード・プロセスがオフロード処理中に不要なログ・データの物理的な削除を完了した場合、ログ・データが 1 次記憶域から補助記憶域に物理的にオフロードされることがあります。逆に、値が大きすぎると、オフロード処理中に 1 次ストレージから解放されるスペースが少なくなるため、以降のオフロード処理の頻度が高まる場合があります。

式の計算結果が 40% から 60% の範囲に収まらない場合は、ワークロードの trandur または akpintvl の値が不適切であることがあります。

ログ・ストリームの定義値(**LOWOFFLOAD** など)は、MVS ロガー SMF 88 レコードからの統計などの情報を分析した後に検討してください。

一般ログ

順方向リカバリー・ログおよびユーザー・ジャーナルに関する推奨事項は、システム・ログに関する推奨事項と異なります。カップリング・ファシリティ構造にログ・データを保存する場合の推奨事項はありません。このようなデータを通常使用する場合に必要となるのは、小さな構造を使用して、データを迅速に DASD にオフロードすることのみです。このような場合は、デフォルト値である **HIGHOFFLOAD** は 80、**LOWOFFLOAD** は 0 を使用します。

ステージング・データ・セットのサイズの調整

MVS はカップリング・ファシリティに書き込まれたデータの 2 次コピーをデータ・スペースに保持し、エラー発生時にカップリング・ファシリティを再構築する場合に使用できるようにします。この条件は、カップリング・ファシリティが MVS の障害から独立している(別の CPC 内にあって、不揮発性である)かぎり、満たされます。

カップリング・ファシリティが同じ CPC 内にある場合、または揮発性ストレージを使用している場合、MVS システム・ロガーはステージング・データ・セットをサポートします。これにより、本来であればカップリング・ファシリティと MVS イメージの両方に影響する障害に対してぜい弱なログ・ストリーム・データのコピーを作成できます。

エレメント(ログ・レコードのグループ)はステージング・データ・セットに 4 KB のブロック単位で書き込まれます(ログ・ストリーム・データ・セットのような 256 バイト単位または 512 バイト単位ではありません)。

ステージング・データ・セットのサイズを調整する場合は、次の式を使用します。


```
staging data set size= (NR * AVGBUFSIZE rounded up to next unit of 4096)
```

NR はカップリング・ファシリティ構造を満杯にするレコード数です。この値は次の式で計算できます。

```
NR = coupling facility structure size / (AVGBUFSIZE rounded up to next element)
```

カップリング・ファシリティ構造およびステージング・データ・セットが同じレコード数を保持できるようにします。ステージング・データ・セットはログ・ストリームと同じオフロードしきい値の影響を受けます。そのため、できるだけオフロード・アクティビティが同じ頻度で実行するのが適切です。

ステージング・データ・セットのサイズは過小に見積もらないで、過大に見積もることを推奨します。最大レコード数を格納できるように (1 つの要素につき 1 つのレコードが存在するように) ステージング・データ・セットを計算するには、次の式を使用します。

エレメント・サイズは 512 バイトです。

```
maximum staging data set size = 8 * coupling facility structure size
```

エレメント・サイズは 256 バイトです。

```
maximum staging data set size = 16 * coupling facility structure size
```

DASD FastWrite 機能を使用して DASD キャッシュ内の保管データを表示し、調査してください (このデータをステージング・データ・セットに直接書き込まないでください)。これにより、必要なデータを短時間で検索することもできます。ただし、キャッシュがいっぱいの場合にデータをキャッシュに書き込むと、ステージング・データ・セットにもデータが書き込まれます。

活動キーポイント頻度 (AKPFREQ)

活動キーポイント頻度値 (AKPFREQ) は、CICS が活動キーポイントを書き込むまでに必要な CICS システム・ログ・ストリーム出力バッファへの書き込み要求数を指定します。キーポイントは、その時点でシステムで実行中のタスクのスナップショットです。

緊急再始動中に CICS が再度読み取る必要があるのは、キーポイントで識別されたタスクのレコードのみです。CICS は最初の活動キーポイント (直前に取得された活動キーポイント) が検出されるまで、システム・ログを逆方向に読み取ります。

キーポイントを取得すると、実行中のシステムにオーバーヘッドが発生します。

- AKPFREQ の設定値が大きすぎて、キーポイント頻度が低すぎる場合、キーポイントの書き込みによって、システムがほんの短い時間遅くなります。
- AKPFREQ の設定値が小さすぎて、キーポイント頻度が高すぎる場合は、緊急再始動時間が短縮されますが、活動キーポイントの処理数が増えるため、処理時間も長くなることがあります。

AKPFREQ 値はデフォルト値の 4000 に設定することを推奨します。AKPFREQ の設定を最適化すると、システム・ログ全体をカップリング・ファシリティに存続させることができます。

AKPFREQ 値を増やすと、システム・ログに必要な 1 次ストレージの量が増加します。AKPFREQ 値を小さくすると、次の効果があります。

- 再始動時間が短縮されることがあります。
- システム・ログに必要な 1 次ストレージの量が減少します。
- タスクの待機時間およびプロセッサ・サイクルが増加することがあります。
- ページングが増加することがあります。

最後の 2 つの効果は、システム・パフォーマンスに影響を与える可能性がありますが、大きな影響ではありません。

AKPFREQ 値をゼロに設定すると、緊急再始動にかかる時間が長くなります。この状態では、CICS は、シャットダウンが発生してシステム・ログが補助ストレージに予備的に格納されるまで、ログ・テールの削除を実行できません。活動キーポイントがないので、CICS はシステム・ログ全体を読み取る必要があるため、DASD オフロード・データ・セット内で予備のシステム・ログを検索する必要があります。

活動キーポイントの頻度は、AKPFREQ システム 初期設定パラメーターによって決まります。AKPFREQ は、CICS が稼働している間に、**CEMT SET SYSTEM AKP(value)** コマンドを使用して変更できます。

CICS ログ・ストリームのグローバル統計には、活動キーポイントの頻度に関する情報も含まれます。詳細については、[ログ・ストリーム・レポート](#)を参照してください。

キーポイントが取得されるたびに、メッセージ DFHRM0205 が CSMT 一時データ宛先に書き込まれます。

AKPFREQ および MRO

MRO 環境の場合、セッション割り振りアルゴリズムは番号が最小の空きセッションを、次の実行タスクで使用するために選択します。その結果、多数のセッションが定義されている場合 (おそらく、ピーク・ワークロード要件に対処するために)、番号の大きいセッションが処理の少ない期間中に頻繁に使用される可能性が小さくなります。

MRO 環境の場合、CICS は 2 フェーズ・コミットを最適化する「暗黙的な forget」プロセスを実装します。これは、MRO 接続のリモート・エンドでミラー・トランザクションがタスク終了処理を完了した場合に、このセッションに関する新規フローが着信すると、このタスクに関連するすべての情報が削除されることを意味します。通常、このフローは、MRO セッション割り振りアルゴリズムの結果、セッションで実行するために割り振られた次のタスクまたはトランザクションの最初のフローです。

トランザクションの着信レートが短期間に変動する場合は、暗黙的な forget 処理を待機している一部のミラー・トランザクションがしばらくの間存続できます。これは特に、このようなミラー・トランザクションがトランザクション着信レートのピーク期間中に番号の大きなセッションに割り振られ、現在は解除されている場合に発生します。

キーポイント・プログラムは、暗黙的な forget 処理を待機しているミラー・トランザクションに関連する作業単位など、永続的な作業単位を処理する場合に、検知しやすいプロセッサ容量を使用します。AKPFREQ 値が低い場合には、この機能が低下します。

AKPFREQ の設定を最適化すると、これらの永続的な多数の作業単位を通常のトランザクション処理アクティビティー中に完了できます。これにより、キーポイント・プログラムで使用されるプロセッサ処理が最小になります。そのため、AKPFREQ 値をデフォルト値より減らす場合は注意が必要です。

ログ延期インターバル (LGDFINT)

LGDFINT システム 初期設定パラメーターは、MVS システム・ロガーを起動するまで強制的なジャーナル書き込み要求を遅らせる期間を決定する場合に、CICS ログ・マネージャーで使用されるログ延期インターバルを指定します。

この値はミリ秒単位で指定します。一般的な CICS トランザクション・ワークロードのパフォーマンス評価では、5 ミリ秒の値を指定した場合に、応答時間と中央処理装置のコストが最適なバランスをとりました。

ログ延期インターバル値を変更すると、CICS パフォーマンスに悪影響を与える可能性があります。値が大きすぎると、MVS システム・ロガーを起動するまでの待機期間が延びるため、CICS トランザクション・スループットが低下します。

ログ遅延インターバルを短縮することにより CICS トランザクションのスループットが向上する例としては、強制的な多数のログ書き込みが発行されて、並行的なタスク・アクティビティーがほとんど発生していない場合があります。このようなタスクは、経過時間のほとんどをログ遅延期間の経過を待機するために費やします。このような場合、遅延期間中にバッファに追加されるその他のログ・レコードは少数であるため、ログ・バッファを書き込むために MVS システム・ロガー呼び出しを遅らせる利点は限定されています。

ログ遅延インターバルの有効範囲は 0 から 65535 ミリ秒ですが、ほとんどの場合、パラメーターを設定するときはデフォルトの 5 ミリ秒が最適なインターバルになります。

ログ遅延インターバルの値が 5 ミリ秒未満の場合は、IXGWRITE マクロを起動する前の CICS ログ・マネージャーの遅延が短縮されます。これによりトランザクション応答時間が改善されますが、CICS には所定の MVS システム・ロガー呼び出しへのジャーナル要求数が少なくなり、IXGWRITE マクロをより頻繁に起動しなければならないため、システムのプロセッサ・コストは増大します。

したがって、ログ遅延インターバル値を 5 ミリ秒より大きくすると、IXGWRITE マクロを起動する前に CICS では遅延期間が長くなるため、トランザクション応答時間が長くなります。ただし、独自のログ・デ

ータを MVS システム・ロガーに書き込む前に、より多くのトランザクションがこのデータを同じログ・バッファに書き込むことができるため、IXGWRITE 呼び出し時の全体的なプロセッサ・コストは小さくなります。

ログ遅延インターバルは **LGDFINT** システム初期設定パラメーターによって決まります。**LGDFINT** は CICS の実行中に、**CEMT SET SYSTEM[LOGDEFER(value)]** コマンドを使用して変更できます。

CICS ログ・ストリーム・グローバル統計は、ログ遅延インターバルに関する情報を収集します。詳しくは、[ログ・ストリーム・レポート](#)を参照してください。

DASD 専用ロギング

DASD 専用ログ・ストリームによって使用される 1 次ストレージは、MVS ロガーが所有するデータ・スペースと、ステージング・データ・セットで構成されます。パフォーマンスを向上させるために、DASD 専用ロギングを調整することができます。

カップリング・ファシリティー構造にデータは書き込まれません。ステージング・データ・セットを使用する場合、DASD 専用ログ・ストリームは DUPLEX(YES) COND(NO) を使用して定義されたカップリング・ファシリティー・ログ・ストリームと同様に機能します。

ステージング・データ・セットが **HIGHOFFLOAD** 制限に達すると、**LOWOFFLOAD** 制限に達するまで、データは削除されるか、またはオフロードされます。

DASD 専用ログ・ストリームおよびカップリング・ファシリティー・ログ・ストリームには、次の原則が適用されます。

- 現在の活動キーポイント・インターバル中に生成されたシステム・ログ・データ、および直前の活動キーポイントで削除されたデータが 1 次ストレージに保存されるように、システム・ログのサイズを変更します。
- システム・ログの場合、「ステージング・データ・セットが満杯」の状況を回避して、補助記憶域にオフロードされないようにします。

205 ページの『[ステージング・データ・セットのサイズの調整](#)』で説明されているように、DASD 専用ログ・ストリームのステージング・データ・セットのサイズ変更の基本原則は、カップリング・ファシリティー・ログ・ストリームのステージング・データ・セットの原則と同じです。開始点として取得した値を取り、ロガー環境をモニターして、ステージング・データ・セットのサイズを調整します。

次の式を使用してシステム・ログのステージング・データ・セットのサイズの開始点を計算します。この式はログ・ストリーム定義の **STG_SIZE** パラメーターで指定された値を計算します。つまり、サイズは 4 KB ブロックの個数として表されます。

ステージング・

DS サイズ [4K ブロック数] = (AKP 期間) * システム・ログのログ書き込み数/秒
ここで、

AKP 期間 = (CICS TS 390 AKPFREQ) / (バッファ格納数/秒)

ログ書き込み数/秒およびバッファ格納数/秒の値は、CICS 統計から取得できます。CICS Transaction Server リリースでは、ログ・ストリーム統計フィールドがこれらの統計を「書き込み要求」(LGSWRITES) および「バッファ付加」(LGSBUFAPP) として収集し、この合計を統計間隔の秒数で除算できます。

より正確なステージング・データ・セットのサイズの概算が必要な場合は、以下の文書を参照してください。

- IBM Redpaper [Performance Considerations and Measurements for CICS and System Logger](#) (REDP-3768)。この資料には、CICS と z/OS System Logger との間の対話についての説明が記載されています。また、さまざまな CICS および System Logger の構成例を掲載し、その調整プロセスについて説明しています。
- IBM Redbooks 資料「[Systems Programmer's Guide to: z/OS System Logger](#)」(SG24-6898)。この文書には、IXGRPT1 レポートを取得/使用して DASD 専用ログ・ストリームのステージング・データ・セットのサイズの概算をする方法が説明されています。(IXGRPT1 は、z/OS に提供されているサンプル・プログラムです。)

CICS 一時記憶域: パフォーマンスおよび調整

CICS 一時記憶域は、存続期間の短いデータを対象にしています。アプリケーションは、データを一時記憶域に、一時記憶域キュー内の一連の番号付き項目として書き込むことができます。CICS は、独自に使用するための一時記憶域キューもいくつか作成します。一時記憶域は、多くの CICS システムで頻繁に使用されます。

CICS の一時記憶域を調整する方法は、CICS 領域が使用可能な一時記憶域の場所によって異なります。一時記憶域は、CICS 領域内の主記憶域、VSAM データ・セット内の補助記憶域、または z/OS カップリング・ファシリティ内の共用一時記憶域プールに置くことができます。一時記憶域は、ローカル CICS 領域に関連付けることも、リモート・キュー所有領域 (QOR) に関連付けることもできます。一時記憶域の場所の概要については、[209 ページの『CICS 一時記憶域: 概説』](#)を参照してください。

主一時記憶域の場合、ストレージの使用をモニターし、**TSMINLIMIT** システム初期設定パラメーターを使用して、適切な制限を設定できます。主一時記憶域の調整について詳しくは、[212 ページの『主一時記憶域: モニターおよび調整』](#)を参照してください。

補助一時記憶域の場合、VSAM データ・セットをセットアップするとき、および CICS 一時記憶域の使用を調整するときに、いくつかの要因のバランスを取る必要があります。以下の要因は、補助一時記憶域のパフォーマンスに影響を与えます。

- データ・セットの制御間隔のサイズ
- CICS 領域内の VSAM バッファの数
- データ・セットへの入出力のための VSAM スtring の数

補助一時記憶域の調整について詳しくは、[213 ページの『補助一時記憶域: モニターおよび調整』](#)を参照してください。

可用性を改善し、動的トランザクション・ルーティングをサポートするために、共用一時記憶域プールをセットアップすることを検討してください。共用一時記憶域プールは、一時記憶域サーバーを必要としますが (通常は、シスプレックス内の各 z/OS イメージに 1 つのサーバーが必要です)、次のような多くの利点があります。

- 共用一時記憶域プールのために CICS 領域内のストレージは使用されません。
- 共用一時記憶域プールでは、トランザクション間の親和性が生じません。主記憶域または補助記憶域内のローカル一時記憶域キューでは、トランザクション間の親和性が生じることがあります。この場合、キューにアクセスするには、影響を受けるトランザクションを同じ領域で実行する必要があります。トランザクション間の親和性は、シスプレックス内の AOR 間でルーティングするワークロードの有効範囲を制限するで、パフォーマンスに影響を与えることがあります。
- リモート側のキュー所有領域と比べて、カップリング・ファシリティ内の共用一時記憶域プールにある一時記憶域キューへのアクセスは迅速です。
- プールごとに複数の一時記憶域サーバーを使用すれば、リモート・キュー所有領域の場合よりも可用性が向上します。1 つの一時記憶域サーバーまたは z/OS イメージに障害が起きた場合、トランザクションを異なる z/OS イメージ上の別のアプリケーション所有領域に動的にルーティングできます。

CICS 一時記憶域: 概説

CICS 領域用の一時記憶域を 3 つの場所 (主記憶域、補助記憶域、または z/OS カップリング・ファシリティ内の共用一時記憶域プール) にセットアップすることができます。

主記憶域

主一時記憶域は、CICS 領域内の 64 ビット (2 GB 境界より上) ストレージにあります。**TSMINLIMIT** システム初期設定パラメーターを使用して、一時記憶域キューが使用可能なストレージの量を指定します。

アプリケーションを実行する CICS 領域内のローカル主記憶域を使用することも、一時記憶域要求をリモート・キュー所有領域 (QOR) に機能シッすることもできます。

補助記憶域

補助一時記憶域は、DFHTEMP という名前の索引なし VSAM データ・セット内にあります。このデータ・セットのセットアップ時に、使用可能なスペースと追加のエクス Tent を定義します。VSAM データ・セットから制御間隔を使用できるようにするために、CICS 領域の一部の 31 ビット (16 MB 境界よ

り上) ストレージが VSAM バッファ用に使使されます。TS システム初期設定パラメーターを使用して、バッファの数を設定します。主一時記憶域と同様に、補助一時記憶域は、ローカル CICS 領域に関連付けることも、リモート・キュー所有領域に関連付けることもできます。

z/OS カップリング・ファシリティ内の共用一時記憶域プール

共用一時記憶域プール (TS プール) は、一時記憶域データ共用サーバー (TS サーバー) によって管理される z/OS カップリング・ファシリティ内にあります。各プールは、カップリング・ファシリティ内のリスト構造に対応しています。z/OS 内のカップリング・ファシリティ・リソース・マネージャー (CFRM) ポリシー定義ユーティリティを使用して、各一時記憶域プールのサイズを指定します。共用一時記憶域プールは、CICS 領域内のストレージを使用せず、アプリケーションはローカル CICS 領域から直接このプールにアクセスします。

アプリケーションが WRITEQ TS コマンドおよび READQ TS コマンドを使用して一時記憶域キューにアクセスすると、要求は CICS 一時記憶域ドメインによって処理され、適切なストレージ・ロケーションに一時記憶域キューが作成されて、データがその中に入れます。タスクは、一時記憶域キューのシンボル名を使用して、データを取り出すことができます。CICS 一時記憶域ドメインは複数の要求を同時に処理できますが、同じ一時記憶域キューに対する要求は直列化され、各要求が処理される期間、キューがロックされます。

TSMODEL リソース定義を使用して、CICS が一時記憶域キューを作成するために使用するモデルをセットアップします。各モデルでは、そのモデルに一致する名前を持つ一時記憶域キューの以下の属性を指定します。

- キューを保管する必要がある一時記憶域の場所
- 一時記憶域がローカル CICS 領域に関連付けられるか、リモート CICS 領域 (キュー所有領域など) に関連付けられるか
- キューが CICS によって自動的に削除されるかどうか (キューが一定期間未使用のまま残り、アプリケーションによって削除されない場合)。
- キューがリカバリー可能かどうか

210 ページの表 21 は、各場所の一時記憶域キューに対して選択できるストレージの使用と機能を要約しています。

表 21. 一時記憶場所の機能			
一時記憶場所	ストレージ・タイプ	自動キュー削除	リカバリー
主記憶域	CICS 領域内の 64 ビット・ストレージ	使用可能	使用不可
補助記憶域	VSAM データ・セット、およびバッファ用の CICS 領域内の 31 ビット・ストレージ	リカバリー不能キューの場合は使用可能	使用可能
共用一時記憶域プール	z/OS カップリング・ファシリティ	使用不可	CICS リカバリーは使用不可ですが、キューは永続的です (CICS 再始動による影響を受けません)

CICS は、独自に使用するための一時記憶域キューもいくつか作成します。これらのキューは、主一時記憶域に置くことも、補助一時記憶域に置くこともできます。例えば、CICS は、次のような目的で一時記憶域を使用します。

- 基本マッピング・サポート (BMS) ページングおよびルーティング
- メッセージのキャッシング
- インターバル制御
- CICS 実行診断機能 (EDF)
- ターゲット・システムが使用不可の間の MRO、ISC、および IPIC のローカル・キュー

CICS システム内の一時記憶域キューを表示する場合、文字 **、\$\$、X'FA' から X'FF'、CEBR、および DF で始まる名前を持つキューが CICS キューです。

一時記憶域キューの自動削除

CICS は、最近参照されていないリカバリー不能な一時記憶域キューを自動的に削除できます。この機能を使用するには、一時記憶域モデル (TSMODEL リソース定義) で適切な有効期限間隔を設定します。

自動削除により、アプリケーションで削除されなかった、不要になっている一時記憶域キューが占有していたストレージが解放されます。

一時記憶域モデルの有効期限間隔は、そのモデルに関連付けられている一時記憶域キューに適用されます。一時記憶域キューは、キューが作成される時点で TSMODEL リソース定義に存在する有効期限間隔を使用します。

デフォルトでは、有効期限間隔はゼロです。つまり、有効期限間隔は一時記憶域キューに適用されません。このようなキューは、自動削除に適格ではありません。

有効期限間隔は、最大 900,000 分 (すなわち 15,000 時間) まで、分単位で設定できます。CICS は、最も近い 10 分の倍数に切り上げた値を使用します。一時記憶域キューを使用するたびに、この間隔カウントが開始されます。有効期限間隔に達する前にキューが再使用されない場合、そのキューは CICS による自動削除の対象になります。1 つ以上の TSMODEL リソース定義に 1 つ以上の非ゼロの有効期限間隔が存在する場合、CICS は定期的に CICS 領域のスキャンを開始し、適格なキューを見つけます。CICS クリーンアップ・タスクが CICS 領域内の一時記憶域キューをスキャンし、自動削除の対象となるキューを削除します。

有効期限間隔は、以下の場所にある一時記憶域キューに適用されます。

- ローカル CICS 領域内の主一時記憶域。
- ローカル CICS に関連付けられたリカバリー不能な補助一時記憶域 (DFHTEMP データ・セット)。
- 共用一時記憶域プール内のキュー。

有効期限間隔は、以下のタイプの一時記憶域キューには適用されないため、CICS はそれらを自動的に削除しません。

- リカバリー可能として定義されている補助一時記憶域内のキュー。
- リモート CICS 領域内のキュー。CICS でリモート一時記憶域キューを削除するには、そのキューを所有する領域内の適切な TSMODEL リソース定義で有効期限間隔を指定します。
- CICS が独自に使用するために作成するキュー。
- どの一時記憶域モデルにも一致しないキュー。

TSMODEL リソース定義で有効期限間隔を変更した場合、そのモデルに一致する既存の一時記憶域キューは影響を受けません。これらのキューは、作成時に適用された有効期限間隔を引き続き使用します。非ゼロの有効期限間隔を持つすべての TSMODEL リソース定義が CICS 領域から削除された場合、CICS は有効期限切れの一時記憶域キューを検索するスキャンを停止します。

CICS クリーンアップ・タスクはスキャンを実行すると、メッセージ DFHTS1605 を発行します。このメッセージは、スキャンされた一時記憶域キューの数と削除されたキューの数を示します。クリーンアップ・タスクが異常終了した場合、メッセージ DFHTS0001 が発行され、CICS が再始動されるまで再実行されません。

システムが TSMMAINLIMIT に到達しており、TS 要求ロックが保留されるように TS キューへの書き込みが行われようとした場合には、CICS クリーンアップ・タスクでは一時ストレージ・キューは削除できません (TSMMAINLIMIT システム初期設定パラメーターを参照)。このような状況では、DFHTS1605 メッセージは、0 個のキューが削除されたと報告します。

TST ユーザーに対する自動削除

CICS 領域で、TSMODEL リソース定義と組み合わせて使用できる、一時記憶域テーブル (TST) を引き続き使用している場合、TST に TSAGE パラメーターが含まれている場合があります。TSAGE は、一時記憶域キューのエージングの期限を日数 (最大 512 日) で指定します。TST に非ゼロの TSAGE が含まれていて、CICS の緊急リスタートが行われた場合、CICS は指定されたインターバルの間参照されなかった一時記憶域

キューを削除します。それ以外のときは、TSAGE パラメーターが原因でキューの自動削除が行われることはありません。

主一時記憶域: モニターおよび調整

一時記憶域キューによって使用される CICS 領域内のストレージの量をモニターおよび制御することができます。

このタスクについて

CICS TS for z/OS バージョン 5.1 以降、主一時記憶域が 64 ビット・ストレージに配置され、以前の CICS リリースよりも使用可能なスペースが大きくなりました。主一時記憶域は、VSAM 入出力アクティビティーおよび一時記憶域サーバーとの通信を必要としません。ただし、主一時記憶域内の一時記憶域キューはリカバリー不能です。

CICS の一時記憶域統計に、主一時記憶域の使用に関する情報が表示されます。また、CICSplex SM または CICS コマンドを使用して、使用中の主一時記憶域の量および現在の制限を見することもできます。最大許容ストレージの 75% 以上が使用中になると、CICS はこの状態を通知するメッセージを発行します。

TSMMAINLIMIT システム初期設定パラメーターを使用して、一時記憶域キュー用に使用可能な CICS 領域内のストレージの量を指定します。ストレージの量を 1 MB から 32768 MB (32 GB) の範囲で指定できます。

ただし、z/OS パラメーター **MEMLIMIT** の設定も確認する必要があります。

手順

1. 一時記憶域モデルで、有効期限間隔を指定します。有効期限間隔を指定すると、CICS は、アプリケーションによって削除されない場合に、モデルに一致する一時記憶域キューを自動的に削除できます。
有効期限間隔について詳しくは、[一時記憶域キューの自動削除](#)を参照してください。
2. CICSplex SM、CICS コマンド、または CICS 統計を使用して、使用中の主一時記憶域の量をモニターします。
 - CICS 一時記憶域グローバル統計および要約統計に、主一時記憶域の使用量が **TSMMAINLIMIT** で設定された制限に達した回数、および主一時記憶域内のデータに使用された仮想記憶域のピーク量が表示されます。
 - TEMPSTORAGE リソースは、最大許容限度と比較して、使用中のストレージを表示します。
3. 主一時記憶域の高い使用量に関する CICS からのメッセージがないか調べます。

- 最大許容ストレージの 75% 以上が使用中になると、CICS はメッセージ DFHTS1601 を発行します。
- アプリケーションが、使用中の主一時記憶域の最大許容限度 (**TSMMAINLIMIT** 値) を超えるデータ項目を書き込もうとした場合、CICS はメッセージ DFHTS1602 を発行します。この状態では、スペースが使用可能になるまで、アプリケーションは主一時記憶域内の一時記憶域キューに書き込むことができません。

どちらかのメッセージが発行された場合は、以下のステップで説明するように、古い一時記憶域キューを削除するか、**TSMMAINLIMIT** 設定を増やすようにしてください。使用量が最大許容値の 70% を下回ると、CICS はメッセージ DFHTS1604 を発行します。

4. **TSMMAINLIMIT** 設定を変更する前に、z/OS パラメーター **MEMLIMIT** の現在の設定を確認してください。
一時記憶域キューに対して使用可能にするストレージの量は、**MEMLIMIT** 値の 25% を超えてはなりません。CICS の **MEMLIMIT** 値、および CICS 領域に現在適用されている **MEMLIMIT** の値を確認する方法については、87 ページの『[MEMLIMIT の見積もり、確認、および設定](#)』を参照してください。
5. オプション: 一時記憶域キュー用に使用可能なストレージの量を変更するには、**TSMMAINLIMIT** 設定を変更します。

稼働中の CICS システムで、**TSMMAINLIMIT** 設定を変更できます。

- **TSMMAINLIMIT** 設定を増やしたときに、新規の値が **MEMLIMIT** の値の 25% を超える場合、**TSMMAINLIMIT** は変更されないまま、メッセージ DFHTS1607 が発行されます。

- **TSMMAINLIMIT** 設定を減らした場合、CICS は許容ストレージ内に現在の使用量の上に 25% 以上のフリー・スペースを維持するようにして、一時記憶域の書き込み要求数がすぐに **TSMMAINLIMIT** 値に達しないようにします。値は以下のとおりに設定します。
 - 現在のフリー・スペースが 25% 未満の場合、**TSMMAINLIMIT** は変更されないままです。メッセージ DFHTS1606 が発行されます。
 - 新規の制限の 25% 以上がフリー・スペースである場合、選択した値に設定が引き下げられます。
 - 新規の制限の 25% 未満がフリー・スペースである場合、現在の使用率にその使用率の 33% を加算した値まで設定が引き下げられます。

TSMMAINLIMIT の値が変更された場合、CICS はメッセージ DFHTS1603 を発行して、新規の設定を示します。

タスクの結果

次の表は、主記憶域のコストを示しています。この例では、 n は削除される前にキューに入れられる項目数を表しています。

表 22. 主記憶域のコスト			
WRITEQ	REWRITE	READQ	DELETEQ
1.0	0.8	0.8	$0.71 + 0.23 \times n$

補助一時記憶域: モニターおよび調整

補助一時記憶域のパフォーマンスは、補助一時記憶域に対してセットアップする VSAM データ・セット DFHTEMP の特性の影響を受けます。また、CICS 領域に対して指定する VSAM バッファおよびストリングの数にも影響を受けます。

このタスクについて

CICS 一時記憶域統計は、補助一時記憶域の使用、バッファおよびストリングの使用、および入出力アクティビティに関する情報を表示します。データ・セットのパフォーマンスに関する追加情報については、RMF または VSAM カタログを参照してください。

補助 TS キューのコストの概算には、VSAM I/O コストが含まれません。1 つの VSAM I/O にかかるコストは約 11.5 K 命令で、以下の場合に発生します。

- いずれのバッファにも適合しない項目の書き込みを試行した場合。
- バッファ内にない項目を読み取る場合。
- DASD からの制御間隔の読み取り時に使用可能なバッファ・スペースがないとき、最低使用頻度を持つバッファを最初に書き出す必要がある場合。

このため、状況によっては、1 つの READQ によって 2 つの VSAM I/O のコストが発生する場合があります。

手順

以下のアクションは、補助一時記憶域のパフォーマンスに影響を与えることがあります。

- VSAM データ・セット DFHTEMP の設定時に制御間隔 (CI) サイズを指定します。アプリケーションで一時記憶域を使用する場合、または CICS 領域内の CICS の変更によって一時記憶域を使用する場合、制御間隔のサイズが引き続き適切であることを確認してください。

制御間隔のサイズより大きい項目を補助記憶域の一時記憶域キューに書き込んだ場合、CICS はその項目を処理しますが、パフォーマンスが低下することがあります。

制御間隔サイズについては、『[制御インターバル・サイズ](#)』を参照してください。

- DASD スペースをより効率的に使用するために、VSAM データ・セット DFHTEMP のセットアップ時に 2 次エクステントを指定できます。

新規データ用の十分なスペースを持つ制御間隔が DFHTEMP に残っていない場合、CICS は 2 次エクステントを使用します。一時記憶域データ・セットは、通常のアクティビティー用の十分な大きさの 1 次エクステントを 1 つと、例外状況用の複数の 2 次エクステントを持つように定義できます。

追加のエクステントの定義については、[複数のエクステントと複数のボリュームを使用したデータ・セットの定義](#)を参照してください。

- 補助一時記憶域のスペースが無駄にならないようにするために、一時記憶域モデルでリカバリー不能キューに対して有効期限間隔を指定してください。

有効期限間隔により、CICS はアプリケーションによって削除されない一時記憶域キューを自動的に削除します。未使用のキューが DFHTEMP 制御間隔から削除されると、CICS は残りのレコードを制御間隔の先頭に移動し、そのスペースを新規データのために使用します。古いキューを効率的に削除することにより、フリー・スペースがある制御間隔を見つけるために必要な時間を削減でき、2 次エクステントを使用する必要が少なくなります。

有効期限間隔について詳しくは、211 ページの『一時記憶域キューの自動削除』を参照してください。

- システム初期設定パラメーター **SUBTSKS=1** を指定した場合、CICS は並行 (CO) モード TCB で一時記憶域 VSAM 要求を実行します。これにより、スループットが増すことがあります。
- TS** システム初期設定パラメーターを使用して、CICS 領域内の補助一時記憶域の VSAM バッファおよびストリングの数を指定します。CICS 領域内の補助一時記憶域の使用頻度が高い場合は、これらの数を調整することができます。

バッファおよびストリングの数を増やすと、タスク待機および VSAM 入出力要求は削減できますが、CICS 領域のストレージ使用量も増えます。

リカバリー可能 TS キューとリカバリー不能 TS キュー

リカバリー可能 TS キューとリカバリー不能 TS キューの一時記憶域のコストは異なります。

リカバリー不能 TS キューにアクセスするコストとリカバリー可能 TS キューにアクセスするコストにおける主な差は、同期点時間で発生します。リカバリー可能キューの場合は、同期点時間で以下のイベントが発生します。

- 作業単位時に制御間隔が使用され、間隔が DASD にまだ達していない場合に VSAM I/O コストが発生した。
- 新規 DASD 制御間隔アドレスがログ・バッファに入れられた。リカバリー・マネージャーがこの作業を行うのにかかるコストは、約 2000 命令です。
- 強制ログ書き込みが要求されて、ログ・バッファが 1 次ストレージに書き込まれたときに同期点が完了する。

それぞれの表では、 n は削除される前にキューに入れられる項目数を表しています。

表 23. リカバリー可能 TS キュー			
WRITEQ	REWRITE	READQ	DELETEQ
1.4	1.9	1.0	$0.87 + 0.18 * n$

表 24. リカバリー不能 TS キュー			
WRITEQ	REWRITE	READQ	DELETEQ
1.3	1.8	1.0	$0.75 + 0.18 * n$

CICS 一時データ (TD) 機能: パフォーマンスおよび調整

一時データ (TD) は、CICS 内のさまざまな状況で使用され、さまざまなオプションがこの機能のパフォーマンスに影響を与える可能性があります。

一時データが使用される状況には、次のものがあります。

- ユーザー・タスクが行う要求 (後で処理するためにデータのキューを構築する要求など) の保守。

- CICS からの要求 (主に、印刷用のシステム・キューにメッセージを書き込む要求) の保守。これらの CICS メッセージを収集するために、一時データはインストール時にセットアップする必要があります。
- 区画内データを保持する DASD スペースの管理。
- キュー・トリガー・レベル指定および区画内宛先に書き込まれたレコードに基づいたタスクの開始。
- CICS 一時データ定義内で指定されているリカバリーのログギングの要求。
- 処理を行うためのオペレーティング・システム・アクセス・メソッドへの区画外要求の引き渡し。

制限

アプリケーション要件には、低いトリガー・レベル、物理または論理リカバリーが指定されていることがあります。これらの機能によってプロセッサ要件が増加します。特に複数のバッファが指定されている場合は、実および仮想記憶要件が増加します。

実装

一時データのパフォーマンスは、インストール済みの一時データ・リソース定義内の **TRIGGERLEVEL** および **RECOVSTATUS** オペランドの影響を受けます。

推奨

次の推奨事項は、QSAM 処理時の待機数の削減に役立つ場合があります。

- 物理プリンターを指定しないようにする。
- エクステント終了処理の結果発生する待ちを除去するために、可能な限り単一のエクステント・データ・セットを使用する。
- RESERVE アクティビティに 頻繁にまたは長期間従属するボリュームにデータ・セットを配置しないようにする。
- 頻繁に使用する数多くのデータ・セットを同じボリュームに配置しないようにする。
- BUFNO および BLKSIZE を選択して、CICS がデータを書き込んだり読み取る速度が、データがボリュームに転送される速度よりも低くなるようにする。例えば、非ブロック化レコードに対して可能な限り BUFNO=1 を指定しないようにします。
- デバイスに対して効果的な BLKSIZE を選択して、少なくとも 3 つのブロックが各トラックに保持されるようにする。

モニター

CICS 統計には、一時データのパフォーマンスが表示されます。CICS 一時データ統計は、書き込みまたは読み取りを行うレコード数を決定する場合に使用します。可変長レコードの長さを分散する方法を決定するには、アプリケーションに関する知識が必要となります。CICS 一時データ統計は、統計インターバルの間の各区画内一時データ・キューのピーク・サイズも示します。RMF または VSAM カタログには、データ・セットのパフォーマンスが示されています。

リカバリー・オプション

リカバリーは、一時データ・レコードがエンキューされる時間に影響を与えます。

以下の 3 つのオプションのうち 1 つを指定できます。

- *No recovery* (リカバリーなし)。リカバリーなしを指定すると、ログギングは行われず、リソースを保護するためのエンキューも行われません。
- *Physical recovery* (物理リカバリー)。区画内キューをシステム障害の直前の状況に復元する場合に物理リカバリーを指定します。パフォーマンスの考慮事項として、据え置き一時データ処理は存在しないため、自動タスク開始がすぐに開始されます。書き込まれたレコードは、別のタスクによってすぐに読み取られます。使い果たすと、制御間隔 (CI) はリリースされます。WRITEQ TD 要求ごとに、CI バッファは VSAM データ・セットに書き込まれます。

注: CICS 内でリカバリーを提供する他のすべてのリソースでは、論理リカバリーのみが提供されています。異常終了状態でバックアウトを使用すると、バックアウトから物理的にリカバリー可能な一時データと リカバリー不能な一時データが除外されます。

- *Logical recovery* (論理リカバリー)。 (システムが失敗するか、またはタスクが異常終了した場合に) 失敗したタスクを実行する前の状況にキューを 復元する場合に論理リカバリーを指定します。 このため、論理リカバリーは、他のリカバリー可能リソース (ファイル制御や一時記憶域など) に対して 定義されているリカバリーと同様に機能します。

要約すると、物理リカバリーでは、システムの障害時にレコードが復元されます。一方、論理リカバリーでは、タスクの失敗時にレコードの保全性が保証され、該当する一時データ・レコードが、レコードをエンキューするタスクの長さに結合されます。

一時データ・セットには、バッファは 32767 まで、ストリングは 255 まで指定可能であり、宛先に向けてシリアル処理が行われます。

宛先に高いトリガー・レベルを指定すると、少数のタスクがその宛先から開始されます。SIT で SUBTSKS=1 が指定されている場合、一時データはファイル・サブタスキングに参加します (174 ページの『VSAM サブタスキングの使用』を参照)。

リカバリー不能 TD キュー

リカバリー不能 TD キューには、関連したコストがあります。

WRITEQ	READQ	DELETEQ
1.5	1.3	1.3

注:

リカバリー不能 TD キューと論理的にリカバリー可能な TD キューとの主な差 は、同期点時間で発生します。同期点では、新規 TD キュー・アドレスがログ・バッファに入れられ、強制ログ書き込みが要求されます。データをバッファに入れるのにかかるコストは 2 K です。ログ・バッファをカップリング・ファシリティーに書き込むコストは、177 ページの『カップリング・ファシリティー・データ・テーブルの使用』で説明されています。

論理的にリカバリー可能な TD キュー

論理的にリカバリー可能な TD キューには、関連したコストがあります。

WRITEQ	READQ	DELETEQ
初回: 2.8 初回以降: 1.5	初回: 2.4 初回以降: 1.4	1.1

注:

リカバリー不能 TD キューと論理的にリカバリー可能な TD キューとの主な差 は、同期点時間で発生します。同期点では、新規 TD キュー・アドレスがログ・バッファに入れられ、強制ログ書き込みが要求されます。データをバッファに入れるのにかかるコストは 2 K です。ログ・バッファをカップリング・ファシリティーに書き込むコストは、177 ページの『カップリング・ファシリティー・データ・テーブルの使用』で説明されています。

物理的にリカバリー可能な TD キュー

物理的にリカバリー可能な WRITEQ 要求では、それぞれの要求ごとに、VSAM I/O の強制 およびカップリング・ファシリティー (CF) へのログ書き込みの強制が行われます。

WRITEQ	READQ	DELETEQ
19.7	初回: 9.3 初回以降: 8.8	8.7

区画内一時データの考慮事項

リカバリー不能な区画内一時データ・キューおよび論理的にリカバリー可能な区画内一時データ・キューの概算には、VSAM I/O コストが含まれません。

VSAM 入出力操作にかかるコストは、約 11.5 K で、次の状態で発生します。

- いずれのバッファにも適合しない項目の書き込みを試行した場合。
- バッファ内にない項目を読み取る場合。
- DASD からの制御間隔の読み取り時に、使用可能なバッファ・スペースがない場合。この状態が生じる場合、最低使用頻度を持つバッファを最初に書き出す必要があります。このため、状況によっては、1 つの READQ によって 2 つの VSAM 入出力操作のコストが発生する場合があります。

複数の VSAM バッファ

区画内一時データ (TD) をサポートするために複数のバッファとストリングを使用すると、単一のシステム全体のバッファ (およびストリング) を使用する場合に発生する可能性がある一時データ内の制約を除去できます。統計を使用すると、一時データの使用量に応じてシステムを調整できます。

要求をキューに入れる必要がある場合は、一時データ宛先に応じて直列にキューに入れられます。通常は、要求で必要となる制御間隔が使用中か、または同じキューや宛先に対する 1 つ以上の前の要求が待機状態にある場合に要求をキューに入れる必要があります。このような状態の場合、その他のキューや宛先に対する要求の保守が継続して行われます。

また、複数のバッファを使用すると、特定の要求によって要求される制御間隔がバッファ内で使用可能になる可能性があります。これにより、実行する必要がある実際の入出力要求 (VSAM 要求) が大幅に減少します。ただし、VSAM 要求は、物理および論理リカバリー要求から指示があると常に実行されます。

CICS が一時データに割り振るバッファ数は、**TD** システム初期設定パラメーターで指定します。デフォルトは 3 です。

複数のバッファのプロビジョンを使用すると、CICS は、ストレージ内に複数の VSAM 制御間隔 (CI) のコピー (または潜在的なコピー) を保持できます。異なるキューに対する複数の一時データ要求は、異なるバッファを使用して並行に保守できます。要求は、キュー名に応じて直列化されますが、グローバルではありません。複数のバッファを使用すると、必要な CI が既にストレージに存在する可能性が高くなり、新規データを保管するためにバッファをフラッシュする必要性が低くなるため、TD データ・セットに対する VSAM 要求の数を削減することができます。VSAM 要求は、リカバリーの考慮で必要な場合に発行されます。

複数のバッファを使用する利点は、インストール時の区画内一時データの使用量のパターンおよび範囲によって異なります。ほとんどのインストールの場合、デフォルト指定 (3 つのバッファ) で問題ありません。一時データの使用量が多い場合は、バッファの数を増やすことをお勧めします。バッファ統計には、適切な割り振りの決定に役立つ十分な情報が提供されます。通常、チューニングの目的は、必要なデータを保持するために使用可能なバッファがない場合にタスクが待機する回数を最小化することです。

この調整プロセスでは、一時データのパフォーマンスとストレージ要件の増加がトレードオフです。数多くのバッファを指定すると一時データ入出力が減少し、並行性が向上しますが、実記憶域の使用効率が悪くなる可能性もあります。また、バッファ数が多くてキュー数が少ない場合は、キューごとの内部バッファ検索に時間がかかることがあります。

バッファは、初期化時に ECDSA から取得できます。

複数の VSAM ストリング

CICS での並行入出力操作に関連して、一時データ (TD) プログラムは、バッファと VSAM TD データ・セットの間で実際の入出力が必要になるたびに、VSAM 要求を発行します。複数の VSAM ストリングを使用すると、複数の VSAM 要求を並行して実行できるため、バッファのサービスが高速になります。

VSAM 要求は、並行要求数が使用可能なストリング数を超えるとキューに入れられます。この制約は、使用可能なストリング数を最大数 255 まで増やすことによって除去できます。バッファ数を選択する場合、ストリング数の制限 255 を考慮する必要があります。バッファ数がストリング数よりも多い場合は、ストリング待機の可能性が増加します。

CICS が TD に割り振る VSAM スtring 数は、**TD** システム 初期設定パラメーターで指定します。CICS のデフォルトは 3 です。

論理リカバリー

ロギングおよびエンキューは、論理リカバリー・トランザクション (一時データ・キュー上で失敗するタスクのアクティビティの動的バックアウトなど) と共に発生します。論理リカバリーは、通常、何らかの理由でレコードのグループを処理する必要がある場合、または他のリカバリー可能リソースを同じタスク内で処理する場合に使用されます。

一時データ要求を処理する間、宛先キュー項目は、UOW が終了するまで、入力か出力、またはその両方 (キューが削除される場合) に対して最初の要求からエンキューされます。つまり、その期間中、他のタスクは同じ目的でキューにアクセスすることができないため、キューの状況の健全性が維持されます。

UOW の終了時 (同期点またはタスクの完了) に、同期点処理が行われ、キュー項目がログに記録されます。ページ要求が処理されます (UOW が行われる間、ページは、ページの準備ができていないキューにマークをつけます)。空の制御間隔は、一時データで使用するためにリリースされます。UOW が行われる間に達するトリガー・レベルによって、0 より大きいトリガー・レベルを持つキューに対して自動タスク開始が行われます。必要に応じて、バッファが VSAM データ・セットに書き込まれます。

キュー項目の DEQUEUE 関数が発生し、他のタスクによって入力または出力処理のキューがリリースされます。タスクによって書き込まれるレコードは、別のタスクで読み取ることができます。

ロギング・アクティビティ

物理リカバリーの場合、キュー項目は、READQ、WRITEQ、および DELETEQ コマンドの後のアクティビティ・キーポイント時間 (ウォーム・キーポイントを含む) にログに記録されます。

論理リカバリーの場合、キュー項目は、同期点およびアクティビティ・キーポイント時間 (ウォーム・キーポイントを含む) にログに記録されます。

区画内一時データの 2 次エクステント

区画内一時データの初期化時に、CICS は、データ・セットの 1 次エクステントがいっぱいになるまでフォーマット制御間隔によって設定された VSAM の空の区画内データを初期化します。複数のエクステントを使用してデータ・セットが定義されている場合は、必要に応じて追加の制御間隔がフォーマットされます。

2 次エクステントを使用すると、DASD スペースをより有効に使用できます。区画内データ・セットは、通常アクティビティ用に十分なサイズの 1 次エクステント、および例外状況 (アクティビティでの予期しないピーク) 用の 2 次エクステントを使用して定義することができます。

区画内一時データを過度に使用する場合に発生する可能性があるチャネルとアームの競合を削減または除去することができます。

区画外一時データの考慮事項

実際の区画外宛先は、CICS が QSAM PUT LOCATE または PUT MOVE コマンドを使用する対象の順次データ・セットです。

CICS 領域の待機が発生する理由とその回避方法

主なパフォーマンス要因として、オペレーティング・システムの待機が挙げられます。つまり、完全な CICS 領域は入出力が完了するまで待機します。長期間の待機は、以下の理由が原因で発生します。

- バッファ・スペースが使用可能でない。
- 2 次スペースの割り振り。
- ボリューム (エクステント) の切り替えが使用可能。
- データ・セットが動的にオープンまたはクローズされた。
- アプリケーションによってボリュームが強制終了された。
- データ・セットが物理プリンター上に定義されており、プリンターが用紙切れである。
- 同じボリューム上の別のデータ・セットに対して RESERVE コマンドが発行される。

このため、以下の方法を使用して CICS 領域での待機数を除去または最小化します。

- 出力データ・セットの十分なバッファリングおよびブロッキングを確保する。
- 最初に十分なスペースを割り振ることによってボリュームの切り替えを回避する。
- ピーク期間中の動的 OPEN または CLOSE アクションを回避する。

区画外一時データとユーザー・ジャーナルの比較

順次データ・セットを実装する代替方法は、CICS ユーザー・ジャーナルを使用する方法です。219 ページの表 25 では、これら 2 つの方法の相違点が要約されています。

表 25. 区画外一時データとユーザー・ジャーナル	
区画外 TD	ユーザー・ジャーナル
領域 (CICS) の待機	タスクの待機
バッファ・ロケーション: MVS ストレージ内	バッファ・ロケーション: DSA 内
バッファ数: 1 から 32767	2 バッファ
入力または出力	入力と出力の両方、タスクは待機
複数のタスクがアクセス可能	<ul style="list-style-type: none"> • 複数のタスクが出力にアクセス可能 • 排他制御下の単一のタスクが入力にアクセス可能

区画外 TD キューでの入出力コスト

区画外 TD キューのパフォーマンス・コストの概算には、I/O コストは含まれません。物理的な順次ファイルの入出力操作にかかるコストは、約 7K で、次の状態で発生します。

- いずれのバッファにも適合しない項目の書き込みを試行した場合。
- バッファ内にない項目を読み取る場合。
- DASD からのデータの読み取り時に、使用可能なバッファ・スペースがない場合。この状態が生じる場合、最低使用頻度を持つバッファを最初に書き出す必要があります。

このため、状況によっては、1 つの READQ によって 2 つの入出力操作のコストが発生する場合があります。

区画外 TD キューはリカバリー不能

WRITEQ	READQ
1.2	1.0

複数の宛先の出力を結合するための間接宛先の使用

TD キューの CSD 宛先内で CICS が必要な項目 (CSMT や CSSL など) に区画外データ・セットを指定しないようにするには、複数の宛先の出力を結合して単一の宛先にする間接宛先を使用することをお勧めします。これにより、ストレージ・スペースおよび内部管理オーバーヘッドを節約できます。

ただし、長い間間接チェーンを使用すると、重大なページングが発生する場合があります。

グローバル CICS エンキューおよびデキュー: パフォーマンスおよび調整

グローバル CICS エンキューおよびデキューは、CICS アプリケーション・プログラミング・インターフェースを拡張して、シスプレックスに含まれる指定した一連の CICS 領域間で指定したリソースへのアクセスを直列化するエンキュー・メカニズムを提供します。

CICS のグローバル・エンキューおよびデキューにより、トランザクション間の親和性を引き起こす大きな原因が取り除かれるため、グローバル・エンキューおよびデキューを使用すると、並列シスプレックスを効率良く活用することができ、パフォーマンス、キャパシティー、およびアベイラビリティが向上します。また、グローバル・エンキューおよびデキューを使用すると、トランザクション間の親和性のルール

を動的ルーティング・メカニズム (CICSplex SM など) に対して提供する必要性が低くなるため、並行シスプレックスを使用するためのシステム管理コストを削減することができます。

CICS は、z/OS のグローバル・リソースの逐次化を使用して、CICS のグローバル・エンキューおよびデキューに関係するリソースをシスプレックス全域で保護します。z/OS グローバル・リソースの逐次化について詳しくは、[z/OS MVS 計画: グローバル・リソース逐次化](#)を参照してください。

実装

ENQMODEL リソース定義を使用して、**EXEC CICS ENQ** および **EXEC CICS DEQ** コマンドがシスプレックス全体にわたるスコープを持つそれぞれの名前付きリソースを定義します。シスプレックス全体にわたるエンキューまたはデキュー機能を使用する必要のある CICS 領域には、必要な ENQMODEL リソースが定義され、インストールされていなければなりません。これを確実に行うために、CICS 領域が CSD を共用し、初期設定グループ・リストが同一の ENQMODEL グループを組み込む方法をお勧めします。ENQMODEL リソース定義の作成について詳しくは、[ENQMODEL リソース](#)を参照してください。

リソース名が動的に構成されるアプリケーションでは、事前に分からないため、エンキュー EXEC インターフェース・プログラム出口の XNQEREQ および XNQEREQC を使用して、適切な ENQMODEL リソース定義と一致するリソース名の先頭に文字を指定する必要があります。それらのユーザー出口について詳しくは、[エンキュー EXEC インターフェース・プログラム出口 XNQEREQ および XNQEREQC](#) を参照してください。

EXEC CICS ENQ および **EXEC CICS DEQ** コマンドがリソースに対して発行されると、CICS は、一致するインストール済みの ENQMODEL 定義の有無を確認します。エンキューの範囲を指定する一致 ENQMODEL リソースが存在する場合、CICS は情報を z/OS グローバル・リソースの逐次化に渡し、エンキューを管理します。z/OS グローバル・リソースの逐次化は、リソースに対してシスプレックス全体にわたる保護を提供します。

z/OS グローバル・リソースの逐次化には、リソースのスコープを指定するリソース名リスト (RNL) が含まれています。RNL 処理により、リソースのスコープが、CICS の ENQMODEL リソース定義内に指定されたスコープから変更されます。

z/OS のデフォルトでは、グローバル・リソースの逐次化は、エンキューおよびデキューの要求を探して適切な RNL を検索し、リソースのスコープを判別するために RNL を使用します。ただし、CICS のデフォルトでは、**NQRNL** システム初期設定パラメーターで指定されたとおり、すべてのエンキューおよびデキューの要求が **RNL=NO** を指定し、RNL 処理から除外されます。このアクションは、グローバル・リソースの逐次化が、CICS の ENQMODEL リソース定義で指定されているスコープのみを使用することを意味します。しかし、代わりの逐次化製品によってエンキュー要求が無視されることも意味します。これは、代わりの逐次化製品を使用している現行グローバル・リソースの逐次化環境外のシステムへのリソースの保護に影響を与えます。z/OS グローバル・リソースの逐次化が CICS からのエンキューおよびデキュー要求の RNL 処理を使用するようにする場合には、RNL 処理を実行する CICS 領域に対して、システム初期設定パラメーター **NQRNL=YES** を指定します。

グローバル・リソースの逐次化の RNL 処理について詳しくは、[z/OS MVS 計画: グローバル・リソース逐次化](#)を参照してください。

推奨

z/OS グローバル・リソースの逐次化は、システムをグローバル・リソースの逐次化システムに結合します。1 つ以上のシステムが相互にリング構成 (GRS=RING) で接続されるか、スター型構成 (GRS=STAR) でカップリング・ファシリティー・ロック構造に接続されます。グローバル・リソースの逐次化をスター型構成として初期設定すると、リソース・シリアライゼーションに関するすべての情報は、ISGLOCK カップリング・ファシリティー構造内に保持されます。グローバル名リソース上でリクエストがエンキューまたはデキュー命令を発行すると、グローバル・リソースの逐次化はカップリング・ファシリティーにアクセスします。

注：GRS=RING は重大なパフォーマンス制約をもたらす可能性があるため、この構成は十分注意して使用する必要があります。パフォーマンス上の理由のため、MVS イメージが 2 より大きい SYSPLEX では、グローバル・リソースの逐次化のスター型構成を使用します。

パフォーマンスの影響にはさまざまな原因がありますが、主に、要求がリングを完了することが遅延するために発生します。リング内の多数の MVS イメージを大きい値の RESMIL と組み合わせると、要求がリン

グを完了することが遅延します。エンキュー要求は、要求が元の MVS イメージに戻るまで許可できません。RESMIL (SYS1.PARMLIB の GRSCNF メンバー内) に対しては、0 の値、または 1 以下の値を使用してください。

CICS モニター機能: パフォーマンスおよび調整

CICS のモニター機能では、後でオフライン分析を行うために、オンライン処理中にすべてのユーザー提供および CICS 提供トランザクションのパフォーマンスに関するデータが収集されます。モニター・データは、パフォーマンスをチューニングしたり、ユーザーにリソースの使用料を課金する場合に役立ちます。CICS モニターによって生成されるレコードは、MVS システム管理タイプ 110 で、SMF データ・セットに書き込まれます。

CICS モニタリングの概要には、さまざまなタイプのモニター・データに関する情報があります。

パフォーマンスに関しては、パフォーマンス・クラス・データを収集することは相当なオーバーヘッドとなります。オーバーヘッドは、約 5% から 10% ですが、ワークロードによって異なります。MVS アドレス・スペースまたは RMF データは、CICS のモニター機能がアクティブであるかどうかに関係なく収集され、CICS のモニター機能を使用する場合に発生するパフォーマンス・オーバーヘッドが示されます。CICS モニター・ドメイン統計には、タイプごとに生成されるモニター・レコードの数が表示されます。

他の請求処理が存在しており、必要なパフォーマンス・データを収集する他の手段が存在しているため、アカウントング情報が必要ではない場合は、CICS のモニター機能をパフォーマンス・クラス・データの収集に使用しないでください。例外クラス・データも、必要でない場合は収集しないでください。

モニター・データを記録するとオーバーヘッドが発生しますが、システムを調整する場合は、パフォーマンスと例外の両方の情報が必要となります。調整を毎日処理しない場合は、CICS のモニター機能を常に実行する必要はありません。調整を行う場合は、パフォーマンス上の問題が通常発生するのはピーク・ポリシー時であるため、そのときに CICS のモニター機能を実行してください。

オーバーヘッドを削減するために、モニター・レコードのデータ圧縮がデフォルトとして設定されています。SMF データ・セットの過度の使用が潜在的な問題である場合は、モニター・レコードからフィールドを除外することを考慮してください。

CICS モニターの制御では、システム初期設定パラメーターを使用して CICS モニター機能オプションを設定する方法、および CICS の実行中にこれらのオプションを変更する方法について説明しています。

CICS トレース: パフォーマンスおよびチューニング

CICS トレースは、CICS トレース・ドメインによって処理され、アプリケーション・プログラムがさまざまなサービスのために CICS に対して行うすべての要求を記録します。ストレージおよび処理の要件は、記録されるトレース項目の数によって異なります。CICS トレースを使用すると、処理要件が大幅に増加します。ただし、CICS トレースを使用しない場合は、CICS 領域で使用可能な問題判別情報の量が減少します。

CICS では、トレースによって発生するプロセッサの使用を直接測定しません。RMF には、処理およびストレージ要件が示されています。補助トレースでは、トレース項目が補助記憶域に書き込まれ、入出力操作のために追加コストがかかります。補助トレースには 2 つのバッファが使用されますが、入出力をオーバーラップできる場合でも、ビジー・システムの場合は入出力率が非常に大きくなります。

CICS 領域で実行されるトレースの量を制御できます。トレース対象のトランザクションまたはコンポーネントと、収集されるトレース・データのレベルを制限できます。これらのオプションは、CICS の始動時に CICS システム初期設定パラメーターを使用して設定することも、CICS の実行中に CICS インターフェースを使用して設定することもできます。CICS 領域で行われるトレースの定義については、CICS trace を参照してください。

CICS は、例外条件を検出すると常に例外トレースを実行します。そのため、CICS トレースに設定した制限に関係なく、常に First Failure Data Capture が得られます。実動領域では、例えば、例外トレースは補助記憶域に書き込むが、それ以外のトレースは実行しないように、トレース・オプションを設定できます。その方法については、CICS exception tracing を参照してください。

CICS トレースによって生成されるトレース・データには、多数の宛先が可能です。これらの宛先の任意の組み合わせを、いつでもアクティブにすることができます。

- 内部トレース・テーブル
- 補助トレース・データ・セット
- MVS 汎用トレース機能 (GTF) データ・セット
- z/OS Unix システム・サービス内の JVM サーバー・トレース・ファイル

また、トランザクション・ダンプが作成された場合、CICS は内部トレース・テーブルをコピーして、トランザクション・ダンプ・トレース・テーブルを作成します。トレースの宛先を選択する方法については、[トレース宛先とトレース状況の設定を参照してください](#)。

内部トレース・テーブル: ストレージの使用

各 CICS 領域には、内部トレース・テーブルが常に存在していなければなりません。内部トレース・テーブルは、他のトレース宛先用のバッファとして使用されます。現在開始されているトレース宛先がまったくない場合でも、CICS は例外トレース項目を内部トレース・テーブルに書き込み、First Failure Data Capture を提供します。

CICS の始動時に、**TRTABSZ** システム初期設定パラメーターを使用して、内部トレース・テーブルのサイズを指定します。内部トレース・テーブルの最小サイズは 16 KB、最大サイズは 1 GB です。デフォルト・サイズは 12288 KB (12 MB) です。トレース・テーブルは、デバッグで必要となる項目を含むことができる大きさである必要があります。

CICS は、内部トレース・テーブル用に (CICS DSA の外部にある) MVS 64 ビット (2 GB 境界より上の) ストレージを取得します。

内部トレース・テーブルのサイズを変更する場合は、z/OS パラメーター **MEMLIMIT** の現在の設定値を確認してください。**MEMLIMIT** は、CICS アドレス・スペースが使用可能な 64 ビット・ストレージの量を制限します。**TRTABSZ** の設定値は **MEMLIMIT** の範囲内でなければならず、CICS 領域での 64 ビット・ストレージを他の用途で使用することも考慮に入れる必要があります。

CICS の **MEMLIMIT** 値、および CICS 領域に現在適用されている **MEMLIMIT** の値を確認する方法については、「パフォーマンスの改善」の『[MEMLIMIT の見積もり、確認、および設定](#)』を参照してください。z/OS での **MEMLIMIT** について詳しくは、「[z/OS MVS P プログラミング: 拡張アドレッシング機能ガイド](#)」の『[専用メモリー・オブジェクトの使用制限](#)』を参照してください。

トランザクション・ダンプ・トレース・テーブル: ストレージの使用

トランザクション・ダンプが作成されると、CICS は現在の内部トレース・テーブルをコピーして、トランザクション・ダンプ・トレース・テーブルを作成します。CICS は、トランザクション・ダンプを生成するときに、トランザクション・ダンプのトレース・テーブル用に 64 ビット・ストレージ (2 GB 境界より上のストレージ) で MVS ストレージを取得します。

TRTRANSZ システム初期設定パラメーターを使用して、トランザクション・ダンプ・トレース・テーブルのサイズを指定します。最小サイズは 16 KB、デフォルト・サイズは 1024 KB です。

CICS TS for z/OS、バージョン 4 リリース 2 より前には、トランザクション・ダンプ・トレース・テーブルは 31 ビット (16 MB 境界より上) ストレージにありました。31 ビット・ストレージの可用性に関する懸念から、当時はトランザクション・ダンプ・トレース・テーブルに小さいサイズを指定した場合は、ご使用の **TRTRANSZ** 値を検討し、64 ビット・ストレージが使用されるようになった現在は、より大きいトランザクション・ダンプ・トレース・テーブルを提供することを考慮してください。

トランザクション・ダンプ・トレース・テーブルは 64 ビット・ストレージにあるため、トレース・テーブルのサイズを設定する際に、z/OS パラメーター **MEMLIMIT** の現在の設定を確認してください。

補助トレース・データ・セット: ストレージの使用

補助トレース・データ・セットは、ディスク上またはテープ上の CICS 所有の BSAM データ・セットです。データ・セットは CICS の始動前に作成しておく必要があります。CICS の実行中に定義することはできません。補助トレース・データ・セットのセットアップ方法については、[補助トレース・データ・セットの設定を参照してください](#)。

CICS の始動時または CICS の実行中に補助トレースを開始すると、CICS 補助トレース・データ・セット用の 2 つの 4 KB バッファが、CICS 領域の 31 ビット (16 MB 境界より上) ストレージ内の MVS ストレージから割り振られます。MVS ストレージは、CICS DSA には含まれていません。補助トレースを停止するとバッファは解放されますが、補助トレースを一時停止したり、補助トレース・データ・セット間で切り替えたりした場合は、解放されません。

GTF データ・セット: ストレージの使用

GTF バッファは 64 ビット・ストレージから割り振られます。

JVM サーバー・トレース: ストレージの使用

SJ コンポーネントのトレース・レベルを 3、4、または 5 に設定すると、その領域のすべての JVM サーバーで処理要件が増加します。詳しくは、[「トラブルシューティング」の『JVM サーバーのトレースのアクティブ化と管理』](#)を参照してください。

CICS セキュリティー: パフォーマンスおよび調整

CICS では、3 つのタイプのセキュリティー (トランザクション、リソース、およびコマンドのセキュリティー) を確保するために、RACF など、外部セキュリティー・マネージャー (ESM) 用のインターフェースが提供されています。

効果

トランザクション・セキュリティーでは、トランザクションを実行するオペレーターの許可が検証されます。リソース・セキュリティーでは、データ・セット、トランザクション、一時データ宛先、プログラム、一時記憶域レコード、およびジャーナルへのアクセスが制限されます。コマンド・セキュリティーは、特定のコマンドへのアクセスを制限する場合に使用し、特別なシステム・プログラミング・コマンド (例えば、**EXEC CICS INQUIRE**、**SET**、**PERFORM**、**DISCARD**、**COLLECT** など) に適用されます。CMDSEC=YES を使用して定義されているトランザクションには、関連付けられているユーザーがあります。

制限

トランザクション、リソース、またはコマンドを不必要に保護すると、プロセッサ・サイクル、および実記憶と仮想記憶の要件が増加します。

推奨

トランザクション・セキュリティーは CICS で強制されるため、リソース・セキュリティーおよびコマンド・セキュリティーの使用は最小限にすることをお勧めします。オペレーターが特定のトランザクションにアクセスできる場合は、該当するリソースにもアクセスできることが前提条件となります。

実装

リソース・セキュリティーは、TRANSACTION 定義内の **RESSEC (YES)** 属性を使用して定義します。コマンド・セキュリティーは、TRANSACTION 定義内の **CMDSEC (YES)** 属性を使用して定義します。

モニター

DFHTASK では、認証要求の経過時間が使用可能です。フィールドは XSVFYPWD および XSVFYKER です。これらは問題診断のためのものです。

RMF には、全体のプロセッサ使用率が示されています。

VERIFY TOKEN および SIGNON TOKEN のための調整

最良のパフォーマンスを得るために、十分な数のオープン TCB を確保し、プログラムをスレッド・セーフとして定義してください。

VERIFY TOKEN および **SIGNON TOKEN** 要求は、可能であればオープン TCB 上で実行されます。**VERIFY TOKEN** または **SIGNON TOKEN** がオープン TCB 上で発行された場合は、その TCB 上で要求が実行されま

す。**VERIFY TOKEN** または **SIGNON TOKEN** がオープン TCB 上で発行されていない場合、使用可能であればオープン TCB に切り替えられ、そうでなければリソース所有 (RO) TCB に切り替えられます。

最良のパフォーマンスを得るために、十分な数のオープン TCB をワークロードに割り当てられるだけの大きい値に **MAXOPENTCBS** システム初期設定パラメーターを設定し、**VERIFY TOKEN** または **SIGNON TOKEN** を使用するプログラムをスレッド・セーフとして定義してください。

CICS の起動時間およびシャットダウン時間: パフォーマンスおよび調整

CICS の起動および通常シャットダウンにかかる時間を短縮する必要がある場合は、起動プロシージャおよび自動インストールを検査対象の領域に含めます。

IBM Redbooks 資料「IBM z Systems Mean Time to Recovery Best Practices」(SG24-7816) には、起動時間およびシャットダウン時間を最小化するための CICS のカスタマイズ方法に関する情報が記載されています。

以下のトピックでは、CICS の起動とシャットダウンのパフォーマンスを改善する方法について説明しています。

起動プロシージャの改善

CICS ではさまざまな構成が可能であるため、起動のさまざまな側面に注意する必要があります。

このタスクについて

起動のパフォーマンスを向上させるために、さまざまな側面を定義および調整することができます。CICS 起動プロシージャと CICS システム初期設定について詳しくは、[CICS startup](#) を参照してください。

手順

1. 以下の項目を定義します。
 - a) グローバルおよびローカル・カタログ
 - b) CICS システム定義 (CSD) データ・セット
 - c) 一時記憶域データ・セットまたは一時データ区画内データ・セット各データ・セットの定義方法については、[データ・セットの定義](#) を参照してください。
2. 端末を定義する場合は、GRPLIST 内のグループ名の位置に注意します。TYPETERM を含むグループが最後にある場合、端末定義の作成に使用されるすべてのストレージが、TYPETERM が既知になるまで保持されます。これにより、システムがストレージ不足になる可能性があります。
システム初期設定テーブル (SIT) の GRPLIST 内のグループは、順次処理されます。GRPLIST でモデル TERMINAL 定義の後にその TYPETERM が続くグループは、ユーザー・トランザクションおよびプログラムの前に配置します。この処理により、CICS が端末のインストールを処理する間に占有する仮想記憶が最小限に抑えられます。
注: MRO の代理端末管理テーブル (TCT) 項目も含め、すべての端末がインストールされます。
GRPLIST では、DFHVTAM グループは、TERMINAL または TYPETERM 定義の前に配置されるようにする必要があります。DFHVTAM グループは DFHLIST グループ・リストに含まれているため、DFHLIST を最初に GRPLIST に追加すると、確実にこの条件が満たされます。DFHLIST を追加しないと、TCT の構築に使用されるプログラムがそれぞれの端末ごとにロードされるため、初期起動およびコールド・スタートの速度が低下します。
CSD で定義されているグループ内の項目が 100 項目を超えないようにする必要があります。項目が多すぎる場合、これにより、処理中に不要なオーバーヘッドが生じたり、グループのメンテナンスがより難しくなったりする可能性があります。
3. **START** パラメーターを変更しても、自動開始させたくない機能のデフォルトが変更されないことを確認します。EXEC ステートメントの **PARM** で、指定変更する機能をコーディングするか、**START** パラメーターに ALL オプションを指定してすべての機能を指定変更することができます。

4. CICS Web サポートまたは Secure Sockets Layer を使用する予定がない場合は、SIT で TCPIP=NO が指定されていることを確認します。TCPIP=YES が指定されている 場合、ソケット・ドメインのタスク制御ブロックがアクティブになります。
5. インストールに適合するように、ローカル・カタログおよびグローバル・カタログの VSAM パラメーターを調整します。
 - a) 制御間隔 (CI) サイズは、最適なデータと DASD サイズ用に変更する必要があります (詳しくは、163 ページの『ローカル共用リソース (LSR) または非共用リソース (NSR)』を参照)。大部分の場合、2 KB の索引 CI、および 8 KB または 16 KB のデータ CI が適切なサイズです。
 - b) JCL でグローバル・カタログ・データ・セットに対して、**BUFSPACE** ではなく **AMP** パラメーターで **BUFNI** および **BUFND** パラメーターを指定できます。
 - c) ストリング数および索引内の索引セット・レコード数をコード化して、索引バッファー数を変更します。索引セット内のレコード数は、IDCAMS LISTCAT 情報から以下のように計算することができます。
 - $T = \text{索引レコードの総数 (索引 REC-TOTAL)}$ 。
 - $D = \text{データの制御間隔サイズ (データ CISIZE)}$ 。
 - $C = \text{制御域ごとのデータの制御間隔 (データ CI/CA)}$ 。
 - $H = \text{データの使用頻度の高い相対バイト・アドレス (データ HURBA)}$ 。
 - d) 索引セット・レコード数は、以下のように計算することができる。この計算では、実際には、使用される制御域の数が算出される。シーケンス・セット・レコード数は、使用される CA の数と同じです。
 - シーケンス・セット・レコード数: $S = H / (D \times C)$
 - 索引セット・レコード数: $I = T - S$

フリー・スペースを調整しても効果がないので、フリー・スペースの調整に時間を費やさないください。

索引レベル数は、CICS がシャットダウンした後に、GCD に対して IDCAMS LISTCAT コマンドを使用することによって取得できます。コールド・スタートでは主に順次処理が使用されるため、CICS によるファイルのオープン時に自動的に割り振られるバッファーのほかに、追加のバッファーを必要としません。

6. リカバリー・マネージャー・ユーティリティ・プログラム DFHRMUTL を使用するかどうかを検討します。コールド・スタートおよび初期始動時に、CICS は、通常、グローバル・カタログからすべてのリソース定義レコードを削除します。リカバリー・マネージャー・ユーティリティ・プログラム DFHRMUTL を使用すると、リソース定義レコードの削除にかかる時間を節約することができます。詳しくは、[リカバリー・マネージャー・ユーティリティ \(DFHRMUTL\)](#)を参照してください。
 - コールド・スタートの前に、入力パラメーターに SET_AUTO_START=AUTOCOLD,COLD_COPY を使用して DFHRMUTL を実行する。これにより、コールド・スタートに必要なレコードのみを含むグローバル・カタログ・データ・セットのコピーが作成されます。このジョブ・ステップからの戻りコードが通常の場合は、元のグローバル・カタログを新規コピーに置き換えることができます (必要に応じて、元のカテゴリのアーカイブを受け取ります)。JCL の例は、DFHRMUTL の説明で示されています。
 - 初期始動の前に、入力パラメーターに SET_AUTO_START=AUTOINIT,COLD_COPY を使用して DFHRMUTL を実行し、同じ手順に従って結果のカテゴリを使用する。
7. 可能な場合は、DATA および INDEX データ・セットを異なる装置に割り振ります。
8. ストレージの節約が期待されない場合でも、コールド・スタートを向上させる方法として、自動インストールされた端末の使用を考慮します。起動時にインストールされる端末が少ないほど、起動時間が短縮されます。
9. **RAPOOL** システム 初期設定パラメーターを、より速い自動インストール速度にできる値に設定します。詳しくは、144 ページの『任意受信プールのサイズの設定』を参照してください。
10. LSR プール定義で、バッファー、ストリング、およびキー長のパラメーターを指定します。このパラメーターを設定すると、LSR プールの構築にかかる時間が短縮され、プールを使用する最初のファイルのオープン時間も短縮されます。

CICS システムのパフォーマンス・グループを定義している場合、CICS ステップに先行するすべてのステップも同じパフォーマンス・グループ内にあるか、または、少なくとも、実行が遅延されないようにディスパッチング優先順位を十分に高くします。

CICS に先行するステップで使用される非 VSAM データ・セットで DISP=(...,PASS) を使用すると、次にデータ・セットが必要とされるときの割り振り時間が短縮されます。DD ステートメントで PASS を使用しないと、これらのデータ・セットのこれ以降の割り振りがカタログを最初からやり直すため、時間のかかる処理となります。

可能な場合、すべての CICS VSAM データ・セットを持つ VSAM ユーザー・カタログは 1 つのみにし、STEPCAT DD ステートメントを使用してカタログの検索時間を短縮します。

DFHRPL で定義されるライブラリー数を最小に保持します。1 つの大規模なライブラリーでは、数多くの小規模なライブラリーよりも LLACOPY の実行に時間がかかりません。始動時にインストールされる動的 LIBRARY リソースにも同様の考慮事項を適用する必要があります。リンク・パック域 (LPA) で共用モジュールを使用して、CICS 中核モジュールのロードにかかる時間を短縮できます。LPA での CICS モジュールのインストール方法については、『インストール』の『CICS モジュールを MVS リンク・パック域にインストールする』を参照してください。

CICS は、常駐プログラムの起動時にはプログラムをロードしません。ストレージ域は予約されていますが、プログラムは、このプログラムに対するプログラム制御を介した最初のアクセスでロードされます。この処理により、始動が速くなります。ストレージ内の特定プログラムまたはテーブルを見つけるには、プログラム制御 LOAD 機能を使用してプログラムまたはテーブルのアドレスを見つける方法を使用することをお勧めします。最初のアクセスのときに、LOAD 機能を物理的に使用すると、事前定義済みのストレージ・ロケーションにプログラムがロードされます。

プログラム・リスト・テーブルの初期化後処理 (PLTPI) タスクを使用してこれらのプログラムをロードする方法は、1 つの可能な技法ではありますが、PLTPI 処理が完了するまで CICS システムが操作可能にならないため、すべてのプログラムをロードすべきではないことに注意してください。必要なもののみをロードするようにしないと、起動時間が長くなります。

自動インストールのパフォーマンス

自動インストールのパフォーマンスを向上させるためにバッファ数を増やす場合があります。バッファ数を増やすと、自動インストールごとに上位索引が読み取られるのを防ぐことができます。

多数の端末が自動インストールされている場合、MXT システム初期設定パラメーターの値に達したり、CICS のストレージが不足することにより、シャットダウンが失敗する可能性があります。シャットダウン障害の考えられる原因を回避するには、CATD トランザクションをそれ自身のクラス内に置き、同時 CATD トランザクション数を制限することを考慮します。また、AIQMAX パラメーターを指定して、自動インストールのキューに入れることができる装置数を制限することができます。このパラメーターにより、他のいくつかの異常イベントの結果として生じた、自動インストールまたは削除プロセスによる仮想記憶の異常消費から保護されます。

CATD トランザクション限度に達した場合、AIQMAX システム初期設定パラメーターは、CICS による LOGON、LOGOFF、および BIND 処理に影響を与えます。CICS は、z/OS Communications Server に対して、このような要求を CICS に渡さないよう要求します。z/OS Communications Server は、CICS が新たにコマンドを受け入れることができることを示すまで、要求を保持します。

これは、キューに入れられた自動インストール要求を CICS が処理したときに起こります。

MVS 自動リスタート管理

MVS 自動リスタート管理機能 (ARM) を使用すると、シスプレックス全体にわたって統合された自動リスタート・メカニズムを実装することができます。シスプレックスは、汎用リソース・セットの数多くの端末専有領域 (TOR) にまたがる ARM および z/OS Communications Server の持続セッションを使用することができます。

自動リスタート管理 (ARM) は、シスプレックス全体にわたって統合されている再始動メカニズムであり、以下のタスクを実行します。

- ・ 異常終了した場合 (またはモニター・プログラムによって停止状態が通知された場合)、MVS サブシステムを再始動する。

- MVS の障害後に、別の MVS イメージでワークロードのすべてのエレメント (例えば、CICS TOR、アプリケーション専有領域 (AOR)、ファイル専有領域 (FOR)、および Db2) を再始動する。
- 障害の発生した MVS イメージを再始動する。

ARM および z/OS Communications Server の持続セッションでは、TOR に障害が発生した場合のリカバリー時間が改善され、ネットワークの一部分のみを再構築すれば済むため、TOR の再始動にかかる時間が短縮されます。障害の発生した TOR の再始動中、汎用リソースにログオンすることができます。

ARM では、監視および自動リスタートが提供され、再始動がより速くなります。オペレーター開始による再始動、またはその他の自動リスタート・パッケージの必要性がなくなります。MVS 自動リスタート管理について詳しくは、『インストール』の『MVS 自動リスタート管理の実施』および『z/OS MVS シスプレックスのセットアップ』を参照してください。

CICS Business Transaction Services: パフォーマンスおよび調整

Business Transaction Services (BTS) では、ビジネス・トランザクション・モデルを CICS に導入します。

効果

BTS は、数多くの個別の CICS トランザクションのフローを制御するプログラムのタイプを作成する場合に使用します。これにより、これらの個別トランザクションは単一のビジネス・トランザクションになります。

推奨

BTS トランザクションは数多くの個別の CICS トランザクションで構成される場合があります、また、著しく長期の実行時間となる場合もあります。そのため、BTS トランザクションに対する特定のパフォーマンス推奨事項はありません。ただし、いくつかの一般的な順守事項が役立つ場合があります。

実装

BTS 機能をサポートするために、CICS では、新しいタイプのデータ・セット (ローカル要求キュー (DFHLRQ) および BTS リポジトリ) にデータを保持します。ローカル要求キューのデータ・セットは、保留中の BTS 要求を保管します。各 CICS 領域には、独自のデータ・セットがあります。ローカル要求キューのデータ・セットは、リカバリー可能な VSAM キー順データ・セット (KSDS) です。VSAM KSDS と同様に最良のパフォーマンスになるよう調整してください。

1 つ以上の BTS リポジトリを使用する場合があります。BTS リポジトリは、通常、VSAM KSDS であり、プロセス、アクティビティー、コンテナ、イベント、およびタイマーの状態データを保持します。BTS リポジトリは、PROCESSTYPE 定義に基づいてプロセスに関連付けられています。BTS プロセスのアクティビティーが複数の CICS 領域にディスパッチされる場合は、その BTS リポジトリをこれらの領域間で共用する必要があります。リポジトリは、以下のいずれかのファイル・タイプです。

- ファイル専有領域が所有し、参加領域で REMOTE として定義されている VSAM KSDS ファイル
- 参加領域で共用される VSAM RLS ファイル

BTS プロセスの実行をサポートする場合、CICS は 1 つ以上のトランザクションを実行します。BTS プロセスは、1 つ以上のアクティビティーで構成されています。各アクティビティーは、一連の CICS トランザクション実行として実行されます。例えば、イベントを待機してアクティビティーが休止状態になっている場合、イベントの発生後にこのアクティビティーが再開され、これがビジネス・トランザクションの続きである場合でも、新規 CICS トランザクションが開始されます。単一 BTS トランザクションの CICS 統計内に、プロセスまたはアクティビティー定義で指定されている数多くのトランザクション ID の実行が表示されます。アクティビティーを実行するときに実行されるアプリケーション・プログラムは、必ずしもトランザクション定義内で定義されているプログラムではありません。BTS では、アプリケーション・プログラム内のプロセスまたはアクティビティー定義では、異なるプログラムを実行するように指定することができます。

実行されるトランザクションの数、および BTS リポジトリへのファイル・アクセスの数やタイプは、選択した BTS サービスの使用方法によって異なります。ご使用のアプリケーションについてこの情報を確認するには、CICS 統計レポートを調べてください。コンテナが BTS リポジトリに保管されることに注意する必要があります。リポジトリが、すべてのアクティブ BTS データを保管できる十分な大きさで

あることを確認してください。これを行うには、テスト・システムに基づいてスケーリングを使用することをお勧めします。

モニター・データ DFHCBTS を使用すると、プロセス内のアクティビティーに関する情報を収集できます。

z/OS ワークロード・マネージャーによるチューニング

z/OS ワークロード・マネージャーは、自動的かつ動的なシステム・リソース (中央処理装置およびストレージ) の平衡化をシスプレックス全体に提供します。

z/OS ワークロード・マネージャーは、以下によってシステム・リソースのバランスを取ります。

- ゴール指向アプローチの採用
- 個別のタスク・レベルのパフォーマンスを反映するサブシステムからのリアルタイム・データの収集
- z/OS (およびサブシステム) のモニター。全体のタスク実行回数に影響するレベルは遅延および待機します。
- システム・リソース管理アルゴリズムへの入力として、パフォーマンス・ゴール、およびリアルタイム・パフォーマンスと遅延データを使用した、sysplex のリソースの動的な管理

このリソース管理は、特に sysplex 環境で有効なものですが、単一 z/OS イメージで実行するサブシステムでも重要視されます。

注: CICSplex SM を使用して CICSplex の動的ルーティングを制御する場合は、z/OS ワークロード・マネージャーに定義されている、CICS トランザクションの CICS 応答時間のゴールに基づいてアクションを実行することができます。[CICSplex SM を使用した動的ルーティング](#)を参照してください。

z/OS ワークロード・マネージャーは次の利点を提供します。

- z/OS リソース管理を介したパフォーマンスの改良。改善点については、例えば以下のような多くの要因に依存する可能性があります。
 - システム・ハードウェア構成
 - システムを区画に分割する方法
 - CICS サブシステムが単一または複数領域のいずれであるか
 - 実行するアプリケーションまたはタスクを分散するタイプ、および操作のプロファイルの多様性
 - 動的に変化する sysplex ワークロードまでのエクステント
- 全体的なキャパシティの改善と作業のスループット向上による、標準的な z/OS sysplex の効率を改良
- z/OS チューニングの単純化。現行の手法では、最適なチューニングの実現や維持が難しい、または多大な時間を要してしまうような操作上の性質を持つシステムに、最も効果があります。

主な利点としては、最適のパフォーマンスを得るために、CICS を継続してモニターおよび調整する必要がないことを挙げることができます。ワークロードの目標をサービス定義内に設定すると、z/OS のワークロード・コンポーネントがリソースとワークロードを管理して目標を達成します。

z/OS ワークロード・マネージャー は、適切なパフォーマンス・ゴールの確立およびキャパシティ・プランニングに使用できるパフォーマンス報告を作成します。

z/OS ワークロード管理の CICS 機能は、CICS ストレージにほとんど影響を与えません。

CICS 開始中に、z/OS ワークロード・マネージャーの CICS サポートは、自動的に初期化されます。z/OS ワークロード管理で、z/OS イメージ上で実行されているすべての CICS 領域 (および他の z/OS サブシステム) は、ワークロード・マネージャーの影響を受けます。

このような領域内の RMI を通る CICS ベースのタスクに対してワークロード管理を正常に作動させる場合は、RMI 経由で CICS に接続された、ユーザー作成のリソース・マネージャーやその他の CICS 以外のコードは、z/OS ワークロード・マネージャー・サポートを提供するように修正されなければなりません。

IBM Redbooks 資料「[System Programmer's Guide to: Workload Manager](#)」(SG24-6472-03) では、z/OS システムのワークロード・マネージャー・コンポーネントを幅広く理解するための情報が提供されています。この資料は、z/OS 1.7 までの z/OS リリースで使用可能な新機能とともに、WLM の基本的な側面を取り上げています。この資料では、ビジネス目標や、システムで実行するトランザクションのタイプに基づいて、WLM ポリシーを作成する方法について説明しています。

z/OS ワークロード管理で使用する用語

以下の用語が z/OS ワークロード管理の説明で使用されます。

種別規則

z/OS のワークロード・マネージャー・コンポーネントがサービス・クラスを割り当てるために使用する規則。

サービス・クラス

同じサービス目標やパフォーマンス目標、リソース要件、または可用性要件を持つ作業のグループ。ワークロード管理の場合は、サービス目標と、必要に応じてリソース・グループがサービス・クラスに割り当てられます。

サービス定義

sysplex 内のすべてのワークロードと処理容量の明示的な定義。サービス・ポリシー、ワークロード、サービス・クラス、リソース・グループ、および種別規則を含む、サービス定義。

サービス・ポリシー

sysplex 内の z/OS ワークロード管理を使用したすべての z/OS イメージのパフォーマンス・ゴールのセット。sysplex、およびポリシーに対する sysplex プロセス内のゴール・モードのすべてのサブシステムに対して、アクティブ・サービス・ポリシーは 1 つのみになります。ただし、サービス・ポリシーをいくつか作成し、異なる処理期間の様々な要求に合わせてそれらを切り替えることができます。

ワークロード

サービス・クラスのグループ。

z/OS ワークロード・マネージャー操作のスパン

z/OS ワークロード・マネージャーは sysplex を介して操作します。sysplex において実行しているすべての z/OS イメージに対して、アクティブ・サービス・ポリシーは 1 つのみになります。

z/OS ワークロード管理がアクティブの状態、z/OS イメージで実行されているすべての CICS 領域（および他の z/OS サブシステム）は、ワークロード管理の影響を受けます。

CICS のワークロードで Db2 や DBCTL などの CICS 以外のリソース・マネージャーが必要になった場合、CICS は、リソース・マネージャー・インターフェース (RMI) を介して情報を受け渡して、z/OS ワークロード・マネージャーが、CICS 以外のリソース・マネージャー内のワークロードの一部を CICS 内のワークロードの一部に関連付けることができるようにします。

タスク関連ユーザー出口とリソース・マネージャーの間の通信を扱う CICS インターフェース・モジュールで、通常、リソース・マネージャー・インターフェース (RMI) またはタスク関連ユーザー出口 (TRUE) インターフェースと呼ばれます。

CICS 領域のパフォーマンス・ゴール

CICS（およびワークロードを構成する他の z/OS サブシステム）に対して、内部応答時間などのパフォーマンス・ゴールを定義できます。

以下のものに対してゴールを定義することができます。

- 個別の CICS 領域
- CICS の下で実行中のトランザクションのグループ
- CICS の下で実行中の個別のトランザクション
- 個別のユーザー ID に関連したトランザクション
- 個別のユーザー LU 名に関連したトランザクション

CICS 領域のパフォーマンス・ゴールを定義するには、各 CICS ジョブにサービス・クラスを割り振ってから、そのサービス・クラスにターゲット応答時間を指定します。通常、実動領域の応答時間はテスト領域のものよりも重要なので、実動領域とテスト領域は別のサービス・クラスに配置されます。

また、ワークロード管理は、パフォーマンスと遅延データを収集します。このデータは、リソース測定機能 (RMF)、IBM Z Decision Support、またはベンダー製品などの製品のレポートおよびモニターに使用することができます。

サービス・レベル・アドミニストレータは、インストールのパフォーマンス・ゴール、およびモニター・データを業務ニーズおよび現行のパフォーマンスに基づいて定義します。ワークロードおよびパフォーマンス・ゴールの完全な定義は、サービス定義と呼ばれます。この種類の情報は、Service Level Agreement (SLA) に記載されています。

CICS ワークロードの種別規則の定義

種別規則は着信作業をサービス・クラスと連動する方法を決定します。種別規則は、グループ・レポート・データに対して、任意で着信作業をレポート・クラスに割り当てることができます。

サービス定義ごとに種別規則のセットが 1 つあります。種別規則はサービス定義の各サービス・ポリシーに適用されるため、sysplex に 1 セットの規則があります。

種別規則は、サービス定義に定義された各サービス・クラスに使用する必要があります。詳しくは、[「z/OS MVS 計画：ワークロード管理」の『分類ルールの定義』](#)を参照してください。

種別規則は作業をサービス・クラスにカテゴリー化し、任意で作業クォリファイアに基づいて、クラスをレポートします。ワークロード管理を使用する各 z/OS サブシステム・タイプに対して、種別規則を設定できます。CICS が使用できる (および z/OS ワークロード・マネージャーに対して CICS 作業要求を識別する) 作業クォリファイアには、以下のものがあります。

CT

接続タイプ

CTG

接続タイプ・グループ

LU

LU 名

LUG

LU 名グループ

SI

サブシステム・インスタンス (総称アプリケーション ID)

SIG

サブシステム・インスタンス・グループ

TC

トランザクション・クラス

TCG

トランザクション・クラス・グループ

TN

トランザクション ID

TNG

トランザクション ID グループ

UI

ユーザー ID

UIG

ユーザー ID グループ

注:

1. 通常、作業はそれが CICS に到着する領域に分類されます。例えば、ユーザー端末が起点の作業は、通常、端末専有領域に分類されます。Web 要求は、通常、リスナー領域に分類されます。アプリケーション専用領域で発生した作業は、その領域に分類されます。作業要求が CICS 領域間にパスされているところでは、トランザクションは各領域に再分類されません。代わりに、元の分類が領域から領域にトランザクションでパスされます。
2. グループ・クォリファイアを使用して、トランザクション ID またはユーザー ID のグループを指定できます。例えば、GRPACICS は、TNG GRPACICS によって種別規則で指定できる CICS トランザクション ID のグループを指定できます。種別規則を指定する方法としては、各トランザクションを別々に分類するよりも、グループ・クォリファイアを使用するほうが優れています。

3. WLM サービス・クラス・トークンは、z/OS Communications Server の LU62 リンクではサポートされません。シスプレックスには、規則が 1 セットしかないためです。LU62 リンクは z/OS SYSPLEX の外側にある場合があり、その場合は WLM は情報にアクセスできません。

種別グループを使用して、個別の作業を同じ作業クォリファイアにグループ化できます。例えば、作業を同じサービス・クラスに割り当ててする場合です。

種別規則の階層を設定することができます。CICS がトランザクションを受信したとき、z/OS ワークロード・マネージャーは、一致するクォリファイア、およびそのサービス・クラスやレポート・クラスに対して種別規則を検索します。作業の一部には、その作業に関連した複数の作業クォリファイアがある可能性があるため、複数の種別規則と一致する可能性があります。したがって、種別規則を指定した順序によって、割り当てられるサービス・クラスが決定されます。

種別規則は、簡単にできるようにしてください。

サービス・クラスの定義

サービス・クラスは、パフォーマンス・ゴールを割り当てできるワークロード内の作業のカテゴリーです。類似の作業のグループのためにサービス・クラスを作成できます。

- パフォーマンス・ゴール

以下のパフォーマンス・ゴールをサービス・クラスに割り当てることができます。

応答時間

平均応答時間 (作業を完了するために必要な時間)、または百分位数の応答時間 (特定の時間以内に完了する仕事のパーセンテージ) を定義できます。

任意設定

特定のゴールがない作業については、ゴールは任意設定であることを指定できます。

速度

バッチ・ジョブおよび開始済みタスクなど、トランザクションに関連していない作業の場合。開始済みタスクとして開始された CICS 領域の場合は、速度ゴールは開始中にのみ適用されます。

注:

1. CICS トランザクションのサービス・クラスについては、速度パフォーマンス・ゴール、任意設定ゴール、または複数のパフォーマンス期間は設定できません。
 2. CICS 領域のサービス・クラスに対しては、複数のパフォーマンス期間を設定することができません。
- インストールにおけるビジネスの重要性
- 1 つのサービス・クラスが他のサービス・クラス・ゴールよりも重要であると認識されるように、重要性をサービス・クラスに割り当てることができます。重要性には、重要度の高い順に 1 から 5 までの番号が付けられています。

また、開始タスクおよび JES のサービス・クラスを作成し、それらのサービス・クラスにリソース・グループを割り当てることができます。このようなサービス・クラスを使用して、CICS に関連したワークロードを管理することが可能ですが、このことは、CICS トランザクション関連の作業を開始する前にのみ可能です。(このように CICS を定義する場合は、タスクまたは JES 「トランザクション」名に対して、アドレス・スペース名は TN として指定されることに注意してください。)

SYSOTHER と呼ばれるデフォルトのサービス・クラスがあります。これは、種別規則で z/OS ワークロード管理が一致するサービス・クラスを検出できない場合、CICS トランザクション用に使用されます。例えば、結合データ・セットが選択不可になる場合に使用されます。

RMF の場合、分かりやすいワークロード・アクティビティ 報告書データを提供するには、CICS トランザクションのサービス・クラスを定義する場合に、以下のガイドラインを使用することをお勧めします。同じサービス・クラスの場合:

1. CICS 提供トランザクションをユーザー・トランザクションと混合しない。
2. 経路指定されたトランザクションは、経路指定されていないものと混合しない。
3. 会話型トランザクションは、疑似会話型トランザクションと混合しない。
4. 長期実行トランザクションは、短期実行トランザクションと混合しない。

CICS パフォーマンス・パラメーターのサービス方針への適合

CICS パフォーマンス・パラメーターが、CICS ワークロードに使用されるワークロード・マネージャーのサービス・ポリシーと互換性があることを確認する必要があります。

一般的には、最初に、z/OS ワークロード・マネージャーに CICS パフォーマンス目標を定義して、CICS パフォーマンスへの影響を監視します。z/OS ワークロード・マネージャー定義が正常に作動した後に、CICS パフォーマンスをさらに拡張するために、CICS パラメーターのチューニングを検討できます。ただし、可能な限り、CICS パフォーマンス・パラメーターの使用をしないようにする必要があります。

パフォーマンス属性で使用する可能性のあるものは、以下のとおりです。

- トランザクション優先順位。動的トランザクション・ルーティングで渡されます。

各トランザクションに割り当てる優先順位を選択するときは、注意を払ってください。トランザクション優先順位には 1 から 255 を指定できますが、狭い間隔の大きな数の使用は避けてください。広い間隔の小さい数を使用した方が、利点を得られます。

CICS ディスパッチャーによって割り当てられた優先順位は、z/OS ワークロード・マネージャーに定義されたパフォーマンス・パラメーターと互換性がある必要があります。

- CICS 領域の並行処理ユーザー・タスクの最大数
- 各トランザクション・クラス内の同時タスクの最大数。
- CICS 領域間の最大セッション数。

第3章 イベント処理パフォーマンスの改善

CICS イベント処理 (EP) は 3 つの主要なコンポーネント、つまりイベントのキャプチャー、EP アダプターのディスパッチング、および EP アダプターの実行によるイベントの発行で構成されています。コンポーネントごとに、さまざまな要因がパフォーマンスに影響を及ぼします。

イベントのキャプチャー時の要素

EP の実行に関連する CICS パフォーマンスについては、次の要因を考慮してください。

- EP が開始済みで、イベント・バインディング・ファイルがインストールされていない場合、プロセッサの使用は無視できます。
- EP が開始済みで、イベント・バインディング・ファイルがインストールされているが、いずれのイベント・キャプチャー仕様のいずれの `<locationFilter>` (キャプチャー・ポイントおよびアプリケーション・コマンド述部) エレメントでもキャプチャー・ポイントが定義されていない場合、プロセッサの使用率は無視できます。
- 主述部の突き合わせの場合、**filterOperator="EQ"** (「Equals」はフィルター述部で指定) 属性を使用すると、他の **filterOperator** 値に比べてパフォーマンスが改善されることがあります。**filterOperator="EQ"** を指定すると、最適化が使用されます。

注: CICS イベント・バインディング・エディターでは、選択されたキャプチャー・ポイントのプライマリ述部を示すためにアスタリスク (*) が使われます。

- `<dataCapture>` エレメント (キャプチャー仕様の「**情報ソース (Information Sources)**」部分でマップされた「**発行済みビジネス情報**」項目) を使用して、キャプチャー・ポイントに関連付けられたデータの特定の部分をキャプチャーする場合は、`<dataCapture>` エレメントの数を道理にかなう範囲で低く抑えてください。低く抑えないと、データ・キャプチャー・コンポーネントごとに個別のコンテナーが作成されるので、パフォーマンスに悪影響が及ぶことがあります。例えば、**EXEC CICS WRITE** コマンドに関連付けられた 1 つのレコードの 0 - 5、10 - 15、20 - 25、30 - 35 バイトをキャプチャーする場合、単一の `<dataCapture>` エレメントを定義して 0 - 35 バイトをキャプチャーした方が、個別のデータ領域ごとに 4 個の `<dataCapture>` エレメントを定義するより効率的です。
- 可能な場合は、不要なフィルター述部を含めないでください。例えば、トランザクション ID と現行プログラム名の両方を基準にフィルター操作を行う必要はないことがよくあります。
- 最も多くの不要なイベントを除外するフィルター述部を、少数の不要なイベントを除外するフィルター述部より前に置いてください。
- 可能な場合は、使用できないデータを含む可能性があるゾーン 10 進数またはパック 10 進数データ・フィールドに、フィルター述部を使用しないでください。

EVENTBINDING GLOBAL STATISTICS カテゴリーの CICS 統計は、イベント・フィルター総計操作を示します。詳しくは、[EVENTBINDING 統計](#)を参照してください。

EP アダプターのディスパッチング時の要素

EP は、L8 TCB 上で実行される複数のディスパッチャー・タスクを実行して、イベントの発行を処理します。これらのディスパッチャー・タスクの数は、L8 モードおよび L9 モードのオープンな TCB 数として CICS によって設定された制限数に向かって増えていきます。CICS は、数式 $(2 * \text{MXT value}) + 32$ を使用してこの制限数を自動的に設定します。EP が L8 TCB を独占しないようにするために、EP ディスパッチャーで使われる L8 TCB の最大数はオープンな TCB 制限数の 3 分の 1 に制限されます。

EVENTPROCESS STATISTICS カテゴリーの CICS 統計には、ピーク時のイベント・キャプチャー・キューおよびピーク時のディスパッチャー・タスクが示されるので、EP による L8 TCB の使用量をモニターできます。詳しくは、[EVENTPROCESS 統計](#)を参照してください。

アダプター構成に応じて、アダプターは付加されるかリンクされるかのどちらかになります。イベントごとにタスクを付加すると、プロセッサの使用量が増えます。付加方式の使用時には、イベント・ディスパッチャー・タスクはイベントの発行を待たずにイベント・キューの処理を続行します。

CICS イベント・バインディング・エディターのアダプター・セクションにトランザクション ID またはユーザー ID が指定されている場合、アダプターは別個のタスクとして接続されます。トランザクション ID またはユーザー ID が指定されない場合、アダプターはディスパッチャー・タスクからリンクされます。ただし、HTTP EP アダプターは常に付加されます。

EP アダプターの実行によるイベント発行時の要因

EP アダプターを実行するためのコストは、使用している EP アダプターによって異なります。

TSQ

TSQ EP アダプターは、CPU の消費という点でコストは最小限で済みます。

WebSphere MQ

イベントごとに WebSphere MQ EP アダプターの MQPUT1 呼び出しがあります。

保証イベント発行：イベントに基づいて、ビジネス上重要なデータを処理するアプリケーション拡張を作成するには、保証イベント発行の機能と信頼性が必要です。ただし、イベント発行の保証 (同期発行モードを使用する) を行うと、イベント発行のコストが、非同期イベント処理タスクからアプリケーション・スレッドに移ります。このため、保証イベント発行を使用すると、アプリケーションの応答時間に関して、イベントの MQPUT をアプリケーション自体に追加する場合と同様の悪影響が及ぶ可能性があります。応答時間に影響が及ぶため、信頼性レベル向上のために同期発行モードを必要とするイベントはどれか、慎重に検討してください。イベントがときどき失われる (イベントがキャプチャーされた後、まだ発行されていない間に) ことが大きな問題にならない場合は、非同期発行モードを選択することをお勧めします。例えば、イベントが失われることがあまりないメール・アプリケーションの場合がこれに当たります。必要な場合のみ同期発行モードを指定することによって、パフォーマンスに与える影響を最小限に抑えながら、保証イベント発行をビジネスに最大限に活用できるようになります。

HTTP

それぞれのイベントで、HTTP EP アダプターに対して WEB OPEN、WEB CONVERSE、および WEB CLOSE の呼び出しを行います。デフォルトでは、イベントが発行された後に CICS はクライアント HTTP 接続を閉じ、次のイベントに対する新規の接続が開きます。イベントが発行された後、CICS はクライアント HTTP 接続を開いたままにすることができます。これにより、イベント発行後に接続を再利用でき、接続の再オープンに必要なプロセッサ・オーバーヘッドがかからずに済みます。接続を開いたままにするには、HTTP EP アダプターが接続に使用する URIMAP リソース内の SOCKETCLOSE 属性を指定します。発行の頻度に応じて、接続に適した有効期限を選択してください。

HTTP アダプターを使用して発行されたイベントごとに、CEPH タスクが付加されます。CEPH トランザクションの **DTIMEOUT** 値は 5 秒なので、イベント・サーバーとの接続を 5 秒以内に確立できない場合、HTTP EP アダプター・タスクはタイムアウトになります。付加された後、CEPH タスクは HTTP 1.1 対応のサーバーにイベントを発行します。CICS には、DFHECEPH という名前の CEPH トランザクション用プロファイルがあります。このプロファイルの **RTIMOUT** 値は 5 秒なので、イベント・サーバーが 5 秒以内に応答しない場合、HTTP EP アダプター・タスクはタイムアウトになります。このプロファイルと CEPH トランザクションを変更する必要がある場合は、これらをコピーできます。

- HTTP 1.1 準拠サーバーまたはネットワーク応答時間と比べて発行率が高い場合には、これらの CEPH タスクが CICS システムにあふれて、MXT 限界に達する可能性があります。MXT 限度に達することを防ぐために、CEPH トランザクションをコピーしてから、トランザクション・クラスにトランザクションを割り当てる必要があります。MXT 限度に達しないようにトランザクション・クラスの **MAXACTIVE** 値を十分に小さく設定し、**PURGETHRESH** 値はゼロ以外の値に設定する必要があります。
- **PURGETHRESH** 限度に達したときにパージされた CEPH タスクは、イベントを発行しなくなります。**PURGETHRESH** 限度に達してタスクがパージされると、CSMT 一時データ宛先にメッセージ DFHAC2036 Transaction CEPH has failed with abend AKCC が書き込まれます。

注：コピーしたプロファイルまたはトランザクションを使用する際には、この新しいトランザクションを使用して HTTP アダプターを実行するように、イベント・バインディング・エディター内でアダプター構成を変更する必要があります。

Transaction Start (トランザクション開始)

トランザクション開始 EP アダプターは、新しい CICS タスクを開始します。そのため、プロセッサの使用量は全体的に増えます。これは、CICS イベントの結果として実行されるトランザクションが開

始されるためです。また、各データ・キャプチャー・フィールドが、コンテナ内の開始済みトランザクションに対して使用可能になります。このような CICS タスクの数が増えると、アダプターでのプロセッサ使用量が非常に増大する可能性があります。

EVENTPROCESS STATISTICS カテゴリーの CICS 統計は、次に示すイベントの数を示します。

- WebSphere MQ EP アダプターにディスパッチされたイベント
- HTTP EP アダプターにディスパッチされたイベント
- トランザクション EP アダプターにディスパッチされたイベント
- TSQ EP アダプターにディスパッチされたイベント
- カスタム EP アダプターにディスパッチされたイベント

詳しくは、[EVENTPROCESS 統計](#)を参照してください。

イベント処理用の CICS 領域のストレージ

イベント処理のために、CICS 領域内で 64 ビット (2 GB 境界より上) のストレージが使用されます。このため、CICS 領域に適用される z/OS **MEMLIMIT** パラメーターに対して選択する値には、イベント処理の使用量が影響を及ぼします。

一時記憶域キュー (TSQ) アダプターを使用していて、主一時記憶域を選択する場合、このデータも 64 ビット・ストレージに入ります。

CICS の **MEMLIMIT** 値、および CICS 領域に現在適用されている **MEMLIMIT** の値を確認する方法については、[「パフォーマンスの改善」](#)の『**MEMLIMIT** の見積もり、確認、および設定』を参照してください。z/OS での **MEMLIMIT** について詳しくは、[「z/OS MVS P プログラミング: 拡張アドレッシング機能ガイド」](#)の『専用メモリー・オブジェクトの使用制限』を参照してください。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。この資料の他の言語版を IBM から入手できる場合があります。ただし、これを入手するには、本製品または当該言語版製品を所有している必要がある場合があります。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。IBM 製品、プログラムまたはサービスに代えて、IBM の知的所有権を侵害することのない機能的に同等のプログラムまたは製品を使用することができません。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒 103-8510

東京都中央区日本橋箱崎町 19 番 21 号

日本アイ・ビー・エム株式会社

法務・知的財産

知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様自身の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Director of Licensing

IBM Corporation

North Castle Drive, MD-NC119 Armonk,

NY 10504-1785

United States of America

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関す

る実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名前はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態を提供されるものであり、いかなる保証も提供されません。IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

プログラミング・インターフェース情報

CICS には、プログラミング・インターフェースと見なすことのできる資料と、プログラミング・インターフェースと見なすことのできない資料があります。

オンライン製品資料の以下のセクションには、CICS Transaction Server for z/OS, バージョン 5 リリース 6 のサービスを取得するプログラムをお客様が作成するためのプログラミング・インターフェースが含まれています。

- [アプリケーションの開発](#)
- [システム・プログラムの開発](#)
- [CICS TS セキュリティー](#)
- [外部インターフェースに向けた開発](#)
- [アプリケーション開発のリファレンス](#)
- [リファレンス: システム・プログラミング](#)
- [リファレンス: 接続](#)

オンライン製品資料の以下のセクションには、CICS Transaction Server for z/OS, バージョン 5 リリース 6 のプログラミング・インターフェースとして意図されていない (プログラミング・インターフェースと誤解される可能性のある) 情報が含まれています。

- [トラブルシューティングおよびサポート](#)
- [CICS TS 診断参照](#)

PDF 形式のマニュアルで CICS 資料にアクセスする場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 のサービスを取得するプログラムをお客様が作成するためのプログラミング・インターフェースが以下のマニュアルに含まれています。

- [アプリケーション・プログラミング・ガイドおよびアプリケーション・プログラミング・リファレンス](#)
- [Business Transaction Services](#)
- [Customization Guide](#)
- [C++ OO Class Libraries](#)
- [Debugging Tools Interfaces Reference](#)
- [Distributed Transaction Programming Guide](#)
- [External Interfaces Guide](#)
- [Front End Programming Interface Guide](#)

- IMS Database Control Guide
- インストール・ガイド
- セキュリティー・ガイド
- Supplied Transactions
- CICSplex SM Managing Workloads
- CICSplex SM Managing Resource Usage
- CICSplex SM アプリケーション・プログラミング・ガイドおよび CICSplex SM アプリケーション・プログラミング・リファレンス
- CICS における Java アプリケーション

PDF 形式のマニュアルで CICS 資料にアクセスする場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 のプログラミング・インターフェースとして意図されていない (プログラミング・インターフェースと誤解される可能性のある) 情報が以下のマニュアルに含まれています。

- Data Areas
- Diagnosis Reference
- Problem Determination Guide
- CICSplex SM Problem Determination Guide

商標

IBM、IBM ロゴおよび ibm.com® は、世界の多くの国で登録された International Business Machines Corporation の商標または登録商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、<http://www.ibm.com/legal/copytrade.shtml> をご覧ください。

Adobe、Adobe ロゴ、PostScript、PostScript ロゴは、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

インテル、Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Intel Centrino、Intel Centrino ロゴ、Celeron、Intel Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

Linux® は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

製品資料に関するご使用条件

これらの資料は、以下のご使用条件に同意していただける場合に限りご使用いただけます。

適用範囲

IBM Web サイトの「ご利用条件」に加えて、以下のご使用条件が適用されます。

個人使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布 (頒布、送信を含む) または表示 (上映を含む) することはできません。

商用使用

これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

権利

ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM これらの資料の内容 についていかなる保証もしません。これらの資料は、特定物として現存するままの状態 で提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。

IBM オンラインでのプライバシー・ステートメント

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (ソフトウェア・オファリング) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項をご確認ください。

CICSplex SM Web ユーザー・インターフェース (メイン・インターフェース) の場合:

このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、認証、お客様の利便性の向上、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名、およびその他の個人情報を、セッションごとの Cookie および持続的な Cookie を使用して収集する場合があります。これらの Cookie を無効にすることはできません。

CICSplex SM Web ユーザー・インターフェース (データ・インターフェース) の場合:

このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、認証、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名またはその他の個人情報を、セッションごとの Cookie を使用して収集する場合があります。これらの Cookie を無効にすることはできません。

CICSplex SM Web ユーザー・インターフェース (「Hello World」ページ) の場合:

このソフトウェア・オファリングは、展開される構成に応じて、個人情報を収集しないセッションごとの Cookie を使用する場合があります。これらの Cookie を無効にすることはできません。

CICS Explorer の場合:

このソフトウェア・オファリングは、展開される構成に応じて、セッション管理、お客様の利便性の向上、または利用の追跡または機能上の目的のために、それぞれのお客様のユーザー名、およびその他の個人情報を、セッションごとの設定および持続的な設定を使用して収集する場合があります。これらの設定を無効にすることはできませんが、ユーザー・パスワードの暗号化形式でのディスクへの保管は、サインオン中にチェック・ボックスにチェック・マークを付けることによるユーザーの明示的な操作によってのみ有効化することができます。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、『IBM オンラインでのプライバシー・ステートメント』 (<http://www.ibm.com/privacy/details/jp/ja/>) の『クッキー、ウェブ・ビー

コン、その他のテクノロジー』および『IBM Software Products and Software-as-a-Service Privacy Statement』 (<http://www.ibm.com/software/info/product-privacy>) を参照してください。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。
なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

アセンブラーHバージョン 2 [135](#)
圧縮、出力データ・ストリーム [151](#)
アドレス・スペース
 オンライン・システムの分割 [132](#)
 共用中核コード [133](#)
 プログラム・ストレージ [134](#), [135](#)
 マップの位置合わせ [133](#)
アプリケーション・プログラム
 常駐、非常駐、一時 [134](#)
 パフォーマンス分析 [18](#)
 16 MB 境界 [135](#)
異常終了
 アプリケーション 8
 大きな変更後 [141](#)
 仮想記憶の不足 [18](#)
 サブプール・ストレージの不足 [77](#)
 トランザクション [10](#)
 ページ・タスク [52](#)
 バックアウト・リカバリー [215](#)
 ONEWTE オプション [146](#)
異常条件プログラム (DFHACP) [11](#)
位置合わせなしマップ [133](#)
位置合わせマップ [133](#)
一時記憶
 データ共用 [209](#)
 パフォーマンスの向上
 複数の VSAM スtring [217](#)
 複数の VSAM バッファ [217](#)
一時記憶域
 並行入出力操作 [217](#)
一時記憶域キュー [209](#)
一時データ
 区画外 [218](#)
 区画内 [217](#)
 パフォーマンスの向上
 複数の VSAM スtring [217](#)
 複数の VSAM バッファ [217](#)
 並行入出力操作 [217](#)
一時プログラム [134](#)
違反、ストレージの [58](#)
インターバル制御値 [70](#)
インターバル・レポート
 統計 [3](#), [25](#)
インバウンド・チェーニング [142](#)
エラー率 [18](#)
応答時間
 貢献要因 [23](#)
 内部 [51](#)
オペランド
 BUFFER [147](#)
 IOAREALEN [142](#), [159](#)
 MSGINTEG [146](#)
 ONEWTE [146](#)

オペランド (続き)
 OPPRTY [66](#)
 PACING [136](#)
 PRIORITY [66](#)
 RECEIVESIZE [147](#)
 SENDSIZE [147](#)
 TERMPRIORITY [66](#)
 TIOAL [142](#)
 VPACING [136](#)
オペレーター・セキュリティー [223](#)
オペレーティング・システム
 キーポイント頻度、AKPFREQ [206](#)
 共用記憶域 [133](#)
 ログ延期インターバル、LGDFINT [207](#)
 CICS インターフェース [140](#)
オペレーティング・システムのインターフェース [140](#)
オンライン・システムの分割 [132](#)

[カ行]

カーネル記憶 [118](#)
外部アクション
 セキュリティー・インターフェース [223](#)
回復
 オプション [215](#)
 物理 [215](#)
 論理 [216](#), [218](#)
拡張機能
 共通サービス域 (ECSA) [122](#)
 システム・キュー域 (ESQA) [121](#)
 専用領域 [123](#)
 リンク・パック域 (ELPA) [133](#)
 MVS 中核 [121](#)
仮想記憶
 内部制限 [18](#)
仮想記憶間サーバー環境 (AXM) [177](#)
仮想記憶間サービス
 CSA の縮小 [155](#)
活動キーポイント頻度 (AKPFREQ) [206](#)
カップリング・ファシリティ・データ・テーブル (CFDT) [177](#)
カップリング・ファシリティ・リソース管理 (CFRM) [180](#)
完全ロード測定 [19](#), [20](#)
キーポイント頻度、AKPFREQ [206](#)
起動時間の改善 [224](#)
機能
 ストレージ保護機能 [80](#)
機能シップ [154](#)
キャパシティー・プランニング (capacity planning) [58](#)
共通サービス域 (CSA) [122](#)
共用一時記憶域 [209](#)
共用リソース
 中核コード [133](#)
 モジュール [224](#)
区画外一時データ [218](#)
区画内一時データ・レポート [67](#), [217](#)
グローバル・エンキューおよびデキュー [219](#)

言語環境 [137](#)
高性能オプション (HPO) [144](#), [146](#), [149](#)
コンテンション・モデル [178](#)

[サ行]

サービス・クラス [231](#)
最大タスク
MXT、システム初期設定パラメーター
限度に到達した回数 [12](#)
作業セット [48](#)
削除、シッパされた端末定義 DSHIPINT および DSHIPIDL の
[161](#)
サブタスキング
VSAM データ・セット制御 (VSP) [174](#)
サブタスク [72](#)
サブプール
その他 [124](#)
[229](#) [123](#), [125](#), [148](#)
[230](#) [123](#), [125](#)
CDSA [98](#), [99](#)
CICS [98](#)
ECDSA [98](#), [101](#), [116](#)
ERDSA [98](#), [116](#)
ESDSA [98](#), [115](#)
ETDSA [98](#)
EUDSA [98](#)
GCDSA [98](#), [117](#)
GSDSA [98](#), [118](#)
RDSA [98](#), [100](#)
SDSA [98](#), [100](#)
UDSA [98](#)
サブプール・ストレージのフラグメント化 [97](#)
システム・アクティビティ報告書、RMF における [20](#)
システム間通信 (ISC) [141](#)
システム管理機能 (SMF) [26](#)
システム・キュー域 (SQA) [121](#)
システム状態 [18](#)
システム初期設定パラメーター
AILDELAY [153](#)
AIQMAX [152](#)
AIRDELAY [152](#)
AKPFREQ [206](#)
BMS [134](#)
CMXT [55](#)
DSALIM [82](#), [84](#)
DSHIPINT および DSHIPIDL [161](#)
EDSALIM
デフォルト [85](#)
FEPI [174](#)
ICV [149](#)
ICVTSD [144](#), [149](#)
LGDFINT [207](#)
MROBTCH [160](#)
MROFSE [161](#)
MROLRM [161](#)
MXT [55](#), [63](#)
OPNDLIM [148](#)
PRTYAGE [66](#)
PRVMOD [133](#)
RAMAX [143](#)
RAPOOL [144](#)
SUBTSKS [174](#)
TD [217](#)

システム・タスク優先順位 [72](#)
システム・ネットワーク体系 (SNA) (Systems Network
Architecture (SNA)) [142](#)
システム・パフォーマンスの分析 [12](#)
実記憶
作業セット [48](#)
自動トランザクション開始 (ATI) [149](#)
自動トランザクション開始 (ATI) (automatic transaction
initiation (ATI)) [143](#)
自動リスタート機能 (ARM) [226](#)
自動ログオン [148](#)
ジャーナリング
構造あたりのログ・ストリーム [202](#)
ステージング・データ・セット [205](#)
AVGBUFSIZE [202](#)
HIGHOFFLOAD しきい値 [203](#)
Integrated Coupling Migration Facility (ICMF) [195](#)
LOWOFFLOAD しきい値 [203](#)
MAXBUFSIZE [202](#)
ジャーナル
いっぱいバッファ [12](#)
無効にする [199](#)
有効化 [199](#)
読み取り [199](#)
user [219](#)
シャットダウン
AIQMAX [226](#)
CATA [226](#)
CATD [226](#)
主一時記憶域 [209](#)
出力データ・ストリームの圧縮 [151](#)
種別規則 [230](#)
障害
トレース [54](#)
症状、ローパフォーマンスの [47](#), [51](#)
常駐プログラム [134](#)
シリアル機能 [49](#)
診断、問題の [12](#)
スケジューラー作業域 (SWA) [124](#)
ステージング・データ・セット [205](#)
ストリング、VSAM における数 [163](#), [166](#), [181](#)
ストレージ
違反 [58](#)
ストレス [52](#)
制限状態 [55](#)
MVS [119](#)
ストレージ・クッション [92](#)
ストレージ・ストレス [92](#)
ストレージ不足 [95](#)
ストレージ不足状態
回避 [94](#)
ストレージ保護機能
ストレージ保護 [137](#)
ストレス、ストレージ [52](#)
スレッド・セーフなファイル制御 [189](#)
制御、ストレージ・ストレスの [52](#)
制御域 (CA) [165](#), [186](#)
制御間隔 (CI) [163](#), [166](#), [181](#)
制限状態 [55](#)
制約
制限 [55](#)
ソフトウェア [49](#)
ハードウェア [48](#)
セット、作業 [48](#)

接頭部ストレージ域 (PSA) [123](#)
専用プロセッサ (zIIP および zAAP) [59](#)
専用領域 [123](#)
相互通信
 セッション [49](#)
測定
 完全ロード [20](#)
 単一トランザクション [22](#)
ソフトウェア制約 [49](#)

[タ行]

代替索引の考慮事項 [170](#)
対話式問題管理システム (IPCS) [27](#)
タスク
 最大仕様 (MXT) [63](#)
 所要時間の短縮 [140](#)
 パフォーマンス定義 [7](#)
 優先順位付け [66](#)
タスク制御ブロックの指定 [72](#)
タスクの優先順位 [72](#)
タスク優先順位 [72](#)
単一トランザクション測定
 CICS 補助トレース [22](#)
端末
 自動インストール [152](#)
 出力データ・ストリームの圧縮 [151](#)
 スキャン遅延 (ICVTSD) [149](#)
 同時ログオン/ログオフ要求 [148](#)
 入出力域 (SESSIONS IOAREALEN) [159](#)
 入出力域 (TIOA) [142, 146](#)
 入出力域 (TYPETERM IOAREALEN) [142](#)
 任意受信入力域 (RAMAX) [143](#)
 任意受信プール (RAPOOL) [144](#)
 メッセージ・ブロック・サイズ [18](#)
 HPO (高性能オプション) [146](#)
 MVS
 HPO [146](#)
 SNA チェーニングの使用 [147](#)
 SNA トランザクション・フローの最小化 [146](#)
 z/OS Communications Server での HPO [146](#)
端末入出力域 (TIOA) [142](#)
端末の自動インストール [152](#)
チェーン・アセンブリ [147](#)
中央演算処理複合システム (CPC) [59](#)
中央電子処理装置 (CEC) [59](#)
調整
 CICS PA の使用 [28](#)
 MVS 以下の CICS [140](#)
 SIGNON TOKEN [223](#)
 VERIFY TOKEN [223](#)
 VSAM [163, 224](#)
データ・セット
 レコード・ブロック・サイズ [18](#)
 DSN (データ・セット名共用) [170](#)
データ・セット名 (DSN) の共用 [170](#)
データ・テーブル
 パフォーマンス統計 [176](#)
 変更の同期 [175](#)
データベース
 設計 [49](#)
 ハードウェア競合 [48](#)
データベース・リソース・アダプター (DRA) [192](#)
デッドロック・タイムアウト [11](#)

同期点コスト [195](#)
統計
 データ・テーブル [176](#)
 モニター用 [3, 25](#)
 CICS の [24](#)
 TCB [73](#)
同時自動インストール [152](#)
動的ストレージ域
 16 MB 境界より上 [79](#)
 16 MB 境界より下 [78](#)
 2 GB 境界より上 [80](#)
動的ルーティングの制御に使用される CICSplex SM [228](#)
トランザクション
 セキュリティ [223](#)
 ループ [136](#)
 CATA [153](#)
 CATD [153](#)
 CSAC [11](#)
トランザクション・クラス
 MAXACTIVE [65](#)
 PURGETHRESH [65](#)
トランザクション・クラス DFHTCLSX および DFHTCLQ2
 効果 [159](#)
「トランザクション・グループ」レポート、CICS PA [32](#)
トランザクション分離 [82](#)
トランザクション分離および実記憶域
 トランザクション分離 [135](#)
トレース
 内部 [24](#)
 補助 [20, 22, 24](#)
 CICS 機能 [3, 25](#)
 GTF [27](#)

[ナ行]

内部アクション
 応答時間 [51](#)
 トレース [3, 24, 25](#)
名前の共用、データ・セット名 (DSN) [170](#)
入出力
 追加の物理的入出力の原因 [170](#)
 率 [18](#)
入出力比率 [18](#)
任意受信
 制御エレメント (RACE) [144](#)
 入力域 (RAIA) [143, 144](#)
 プール (RAPOOL) [49, 143, 144](#)
ネットワーク
 設計 [49](#)
 ハードウェア競合 [48](#)

[ハ行]

ハードウェア制約 [48](#)
バーを超える動的ストレージ域 [82](#)
バックアウト・リカバリー [215](#)
パフォーマンス
 改善 [61](#)
 高性能オプション (HPO) [146, 149](#)
 ゴール [231](#)
 査定 [18](#)
 制約
 症状 [47](#)

パフォーマンス (続き)

制約 (続き)

ソフトウェア [49](#)

ハードウェア [48](#)

低下の症状 [47](#)

パラメーター、サービス・ポリシーとのマッチング [232](#)

分析

概要 [12](#)

完全ロード測定 [20](#)

手法 [19](#)

症状およびソリューション [51](#)

単一トランザクション測定 [22](#)

定義 [12](#)

モニター [6](#)

NetView パフォーマンス・モニター (NPM) [143](#)

パフォーマンスおよび調整

CICS PA の使用 [28](#)

パフォーマンス管理 [58](#)

パフォーマンス・クラス・データ、CICS モニター [28](#)

パフォーマンス・コスト

カップリング・ファシリティー・データ・テーブル [186](#)

パフォーマンス測定 [4, 34](#)

パフォーマンスの考慮 [58](#)

パラメーター

キー長 [168](#)

BUFFERS [168](#)

DATABUFFERS [173](#)

INDEXBUFFERS [173](#)

MAXNUMRECS [176](#)

SHARELIMIT [168](#)

STRNO [168, 173](#)

TABLE [176](#)

VSP [174](#)

非共用リソース (NSR) [163, 166, 181](#)

非常駐プログラム [134](#)

非送信請求項目

統計 [3, 25](#)

非同期処理 [154](#)

ファイル制御

コスト [190](#)

LSR

最大キー長 [168](#)

リソース百分位数 (SHARELIMIT) [168](#)

VSAM [174](#)

複数領域操作 (MRO) [154](#)

複数領域操作バッチ要求 [71](#)

物理的入出力、追加 [170](#)

プログラム

一時 [134](#)

常駐 [134](#)

ストレージ・レイアウト [134](#)

非常駐 [134](#)

16 MB より上 [135](#)

COBOL [133](#)

PL/I [133](#)

プロセッサ・サイクル [48](#)

プロセッサ使用量 [18](#)

ブロック・サイズ [18](#)

分散トランザクション処理 (DTP) [154](#)

分散プログラム・リンク (DPL) (distributed program link (DPL)) [154](#)

平均修復時間 [224](#)

平均ブロック・サイズ [201](#)

並行操作

並行操作 (続き)

入出力操作 [217](#)

任意受信要求 [144](#)

非同期ファイル入出力 [173](#)

ログオン/ログオフ要求 [148](#)

VSAM 要求 [163, 166, 181](#)

並行モード TCB [72](#)

ページング

過大 [51, 54](#)

定義 [54](#)

問題 [54](#)

率 [18, 22](#)

ページング可能リンク・バック域 (PLPA) [122](#)

変更済みリンク・バック域 (MLPA) [122](#)

補助一時記憶域

制御間隔サイズ [213](#)

2 次エクステンント [213](#)

補助トレース [20, 22, 24](#)

[マ行]

マップの位置合わせ [133](#)

モード TCB [73](#)

モジュール

管理 [133](#)

共用 [224](#)

モニター

手法 [7](#)

汎用トレース機能 (GTF) [27](#)

リソース測定機能 (RMF) [4, 34](#)

モニター手順 [7](#)

モニターの方針 [7](#)

モニター用ツール [23](#)

問題診断 [12](#)

[ヤ行]

ユーザー・オプション

ジャーナル [219](#)

優先順位繰り上げ [72](#)

要求/応答単位 (RU) [143](#)

要求された統計 [3, 25](#)

要求されたリセット統計 [3, 25](#)

要求のバッチ処理 [71](#)

[ラ行]

リソース

共用 (LSR) [168](#)

非共用 (NSR) [163, 166, 173, 181](#)

マネージャー (SRM) [27](#)

ローカル共用 (LSR) [163, 166, 181](#)

リソース・セキュリティのレベル検査 [223](#)

リソース測定機能 (RMF) [4, 20, 34](#)

リソースの分割

オンライン・システム [132](#)

独立アドレス・スペース [132](#)

ISC の使用 [141](#)

MRO の使用 [141](#)

リソース問題の解決 [56](#)

領域

サイズ増加 [77](#)

領域を超えるフリー・ストレージ [125](#)

- リンク・パック域 (LPA)
 - CLPA (リンク・パック域作成) [122](#)
 - ELPA (拡張リンク・パック域) [133](#)
 - MLPA (変更済みリンク・パック域) [122](#)
 - PLPA (ページング可能リンク・パック域) [122](#)
- リンク・パック域作成 (CLPA) [122](#)
- 例外クラスのモニター [28](#)
- レコード・レベル共用 (RLS) [186](#)
- レポート
 - RMF における DASD アクティビティ [20](#)
 - RMF におけるシステム・アクティビティ [20](#)
- ローカル・システム・キュー域 (LSQA) [124](#)
- ロガー環境
 - ロガー環境のモニター CICS 領域の分析アクティビティ
 - ジャーナルおよびログ・ストリーム [200](#)
 - CICS システム・ログ [200](#)
 - MVS 生成の統計
 - CICS 統計 [200](#)
- ロギング
 - リカバリー後 [218](#)
- ロギングおよびジャーナリング
 - 構造あたりのログ・ストリーム [202](#)
 - ステー징・データ・セット [205](#)
 - モニター [195](#)
 - HIGHOFFLOAD しきい値 [203](#)
 - Integrated Coupling Migration Facility (ICMF) [195](#)
 - LOWOFFLOAD しきい値 [203](#)
- ロギング・マネージャー
 - 平均ブロック・サイズ [201](#)
- ログ延期インターバル (LGDFINT) [207](#)
- ログ延期インターバル、LGDFINT [207](#)
- ログオン/ログオフ要求 [148](#)
- ロック・モデル [178](#)
- 論理リカバリー [218](#)

[ワ行]

- ワークロード・マネージャー
 - z/OS
 - 操作のスパン [229](#)
 - パフォーマンス・ゴールの定義 [229](#)
 - 用語 [229](#)
 - 利点 [228](#)

[数字]

- 1 日の終わり統計 [3, 25](#)
- 16 MB 境界より上
 - 仮想記憶 [75](#)
- 16 MB 境界より下
 - 仮想記憶 [75](#)
- 2 GB 境界より上
 - 仮想記憶 [75](#)
 - 16 MB 境界 [75](#)
- 2 GB 境界より下
 - 仮想記憶 [75](#)
- 229 サブプール [148](#)
- 24 ビット・ストレージ [75](#)
- 31 ビット・アドレッシング [135](#)
- 31 ビット・ストレージ [75](#)
- 64 ビット・ストレージ [75, 82](#)

A

- ACF/Communications Server
 - 統計 [12](#)
 - プロセッサ使用量 [11](#)
 - ペーシング (pacing) [136](#)
- ACF/SNA
 - 共通システム域 (CSA および ECSA) [122](#)
 - 高性能オプション (HPO) [146](#)
 - サービス・クラス (COS) [155](#)
 - サブプール [229 77, 136](#)
 - サブプール [230 125](#)
 - ストレージ管理 [151](#)
 - 端末入出力 [142](#)
 - 調整 [140](#)
 - データ・ストリーム圧縮 [151](#)
 - 統計 [55](#)
 - トレース [27, 151, 154](#)
 - 任意受信プール (RAPOOL) [49, 144](#)
 - 複数領域操作 (MRO) [154](#)
 - ログオン/ログオフ要求 [148](#)
 - IBMTTEST [49](#)
 - ICVTSD [149](#)
 - LMPEO オプション [147](#)
- AILDELAY、システム初期設定パラメーター [153](#)
- AIQMAX、システム初期設定パラメーター [152](#)
- AIRDELAY、システム初期設定パラメーター [152](#)
- AKPFREQ
 - および MRO [207](#)
- AKPFREQ、システム初期設定パラメーター [206](#)
- AMODE(24) プログラムの Language Environment ランタイム・オプション [52](#)
- APPC
 - CICS PA レポート [30, 32](#)

B

- BMS (基本マッピング・サポート)
 - マップの位置合わせ [133](#)
- BMS、システム初期設定パラメーター [134](#)
- BTS [227](#)
- BTS レポート、CICS PA [32](#)
- BUFFER オペランド [147](#)
- BUFFER パラメーター [168](#)
- BUILDCHAIN 属性 [147](#)

C

- CA (制御域) [165, 186](#)
- CATA トランザクション [153](#)
- CATD トランザクション [153](#)
- CDSA [78](#)
- CDSA サブプール [98](#)
- CDSASZE [92](#)
- CECMCHTP [59](#)
- CECMDLID [59](#)
- CFDT サイジング [180](#)
- CFDT 利点 [177](#)
- CFDT、FILE 定義を使用する [177](#)
- CFRM、カップリング・ファシリティ・リソース管理ポリシー [180](#)
- CHANGED 戻り条件 [178](#)
- CI (制御間隔) [163, 166, 181](#)

CICS Business Transaction Services [227](#)
 CICS Performance Analyzer (CICS PA) [28](#)
 CICS トランザクション・マネージャー
 タスクの優先順位付け [63](#)
 トランザクション・クラストラランザクションの制御
 トランザクションの制御 [63](#)
 MAXACTIVE [63](#)
 トランザクション・クラス・ページしきい値トランザク
 ション・マネージャー
 PURGETHRESH [63](#)
 パフォーマンスおよび調整 [63](#)
 MXT の設定 [63](#)
 CICS トレース機能パフォーマンス・データ [3](#), [25](#)
 CICS モニター機能
 CICS PA レポート [28](#)
 CICS 領域サイズ [83](#)
 CICS 領域のストレージ保護 [80](#)
 CLPA (リンク・パック域作成) [122](#)
 COBOL
 アプリケーション・プログラム [133](#)
 CPUTONCP [59](#)
 CSA (共通サービス域)
 内容 [122](#)
 CSA (共通システム域)
 トランザクション・ループ [136](#)
 SVC 処理 [154](#)
 CSAC トランザクション [11](#)

D

DASD (直接アクセス・ストレージ・デバイス)
 使用量の検討 [11](#)
 RMF におけるアクティビティ 報告書 [20](#)
 DATABUFFERS パラメーター [173](#)
 DB2CONN、DB2ENTRY、DB2TRAN 定義 [192](#)
 DFHACP、(異常条件プログラム) [11](#)
 DL/I
 データベース [19](#)
 トランザクション [22](#)
 DLL、C++ [139](#)
 DPL (分散プログラム・リンク) [154](#)
 DRA (データベース・リソース・アダプター) [192](#)
 DSA
 サイズの設定 [92](#)
 DSA (動的ストレージ域)
 ストレージ保護機能 [80](#)
 DSALIM
 サイズの見積もり [84](#)
 DSALIM の見積もり [84](#)
 DSALIM、システム初期設定パラメーター [82](#)
 DSALIMIT
 システム初期設定パラメーター [84](#)
 DSN (データ・セット名) の共用 [170](#)
 DTIMOUT (デッドロック・タイムアウト・インターバル) [11](#)
 DTP (分散トランザクション処理) [154](#)

E

ECDSA [79](#)
 ECDSA サブプール [98](#)
 ECDSASZE [92](#)
 ECSA (拡張共通サービス域) [122](#)
 EDSA (拡張動的ストレージ域) [79](#)

EDSALIM
 サイズの見積もり [85](#)
 デフォルト [85](#)
 EDSALIM の見積もり [85](#)
 EDSALIM、システム初期設定パラメーター [82](#)
 EDSALIMIT
 システム初期設定パラメーター [85](#)
 ERDSA [79](#)
 ERDSA サブプール [98](#)
 ERDSASZE [92](#)
 ESDS ファイル
 ストリング数 [163](#), [166](#), [181](#)
 ESDSA [79](#)
 ESDSA サブプール [98](#)
 ESDSASZE [92](#)
 ESQA (拡張システム・キュー域) [121](#)
 ETDSA [79](#)
 ETDSA サブプール [98](#)
 EUDSA [79](#)
 EUDSA サブプール [98](#)
 EUDSASZE [92](#)
 EXEC CICS WRITE JOURNALNAME コマンド [199](#)

F

FEPI、システム初期設定パラメーター [174](#)
 FORCEQR [71](#)
 FORCEQR システム初期設定パラメーター [71](#)

G

GCDSA [80](#), [87](#)
 GCDSA サブプール [98](#)
 GDSA [80](#), [87](#)
 GDSA (2 GB 境界より上の動的ストレージ域)
 GCDSA [82](#)
 GRS=STAR [219](#)
 GSDSA サブプール [98](#)

H

HPO (高性能オプション) [149](#)

I

IBM Z Decision Support
 定期的レポート [10](#)
 IBMTEST コマンド [49](#)
 ICMF [195](#)
 ICV パラメーター [70](#)
 ICV、システム初期設定パラメーター [149](#)
 ICVTSD、システム初期設定パラメーター [144](#), [149](#)
 INDEXBUFFERS パラメーター [173](#)
 Integrated Coupling Migration Facility (ICMF) [195](#)
 IOAREALEN オペランド [142](#), [159](#)
 IP Interconnectivity (IPIC) [154](#)
 IPCS (対話式問題管理システム) [27](#)
 IPIC (IP 相互接続性) [154](#)
 ISC (システム間連絡)
 セッション [147](#)
 分割 [141](#)
 ミラー・トランザクション [155](#)
 2MB LPA [122](#)

K

KEYLENGTH パラメーター [168](#)

L

Large Systems Performance Reference (LSPR) 比率 [59](#)

LGDFINT、システム初期設定パラメーター [207](#)

LLA (ライブラリー・ルックアサイド機能) [135](#)

LOWOFFLOAD しきい値

HIGHOFFLOAD しきい値 [203](#)

LPA (リンク・バック域) [122](#)

LSQA (ローカル・システム・キュー域) [124](#)

LSR (ローカル共用リソース)

最大キー長 [168](#)

バッファ割り振り [163, 166, 168, 181](#)

リソース百分位数 (SHARELIMIT) [168](#)

LSRPOOL パラメーター [170](#)

VSAM ストリング設定 [168](#)

VSAM の考慮事項 [163, 166, 181](#)

M

MAXACTIVE、トランザクション・クラス [65](#)

MAXKEYLENGTH パラメーター [168](#)

MAXNUMRECS パラメーター [176](#)

MEMLIMIT

サイズの見積もり [87](#)

GDSA 用の MEMLIMIT の割り振り [82](#)

MEMLIMIT の見積もり [87](#)

MLPA (変更済みリンク・バック域) [122](#)

MRO

および XCF [154](#)

MVS シスプレックス環境 [154](#)

MRO (複数領域操作)

仮想記憶間サービス [122](#)

機能シップ [159, 161](#)

セッション [144](#)

トランザクション・ルーティング [155, 159](#)

バッチ要求 [160](#)

分割 [141](#)

2MB LPA [122](#)

CICS PA レポート [30, 32](#)

MROBTCH [71](#)

MROBTCH 要求のバッチ処理 [71](#)

MROBTCH、システム初期設定パラメーター [160](#)

MROFSE、システム初期設定パラメーター [161](#)

MROLRM、システム初期設定パラメーター [161](#)

MSGINTEG オペランド [146](#)

MVS

拡張共通サービス域 (ECSA) [122](#)

仮想記憶 [119, 132](#)

仮想記憶間サービス [155](#)

システム・チューニング [61](#)

中核および拡張中核 [121](#)

調整 [140](#)

プログラムのロード・サブタスク [55](#)

プログラム・ロードのサブタスク [52](#)

ライブラリー・ルックアサイド機能 [135](#)

リンク・バック域 (LPA) [132](#)

HPO [149](#)

Java プログラム [119](#)

QUASI タスク [48](#)

MVS ストレージ [119](#)

MVS ワークロード・マネージャー

CICS PA レポート [32](#)

MXT、システム初期設定パラメーター [63](#)

N

NetView パフォーマンス・モニター (NPM) [143, 147](#)

NPM (NetView パフォーマンス・モニター) [143, 147](#)

NSR (非共用リソース)

バッファ割り振り [163, 166, 181](#)

VSAM ストリング設定 [173](#)

VSAM の考慮事項 [163, 166, 181](#)

VSAM バッファ割り振り [173](#)

O

OFFLCPUT [59](#)

ONEWTE オペランド [146](#)

OPNDLIM、システム初期設定パラメーター [148](#)

OPPRTY オペランド [66](#)

P

PACING オペランド [136](#)

Performance Analyzer [30](#)

PL/I

アプリケーション・プログラム [133](#)

リリース 5.1 [135](#)

PLPA (ページング可能リンク・バック域) [122](#)

PRIORITY オペランド [66](#)

PRTYAGE [72](#)

PRTYAGE、システム初期設定パラメーター [66](#)

PRVMOD、システム初期設定パラメーター [133](#)

PSA (接頭部ストレージ域) [123](#)

PURGETHRESH、トランザクション・クラス [65](#)

R

RAIA (任意受信、入力域) [143](#)

RAMAX、システム初期設定パラメーター [143](#)

RAPOOL、システム初期設定パラメーター [144](#)

RDSA [78](#)

RDSA サブプール [98](#)

RDSASZE [92](#)

RECEIVSIZE 属性 [147](#)

REGION

サイズの見積もり [83](#)

REGION の見積もり [83](#)

RLS、FILE 定義を使用する [187](#)

RMF (リソース測定機能)

定期的な使用 [9](#)

RMF によって収集されたデータ [4, 34](#)

RU (要求/応答単位) [143](#)

RUWAPOL システム初期設定パラメーター [139](#)

S

S40D 異常終了 [77, 125, 141](#)

S80A 異常終了 [77, 141](#)

S822 異常終了 [77, 141](#)

SDSA [78](#)

SDSA サブプール [98](#)

SDSASZE [92](#)
SENDSIZE 属性 [147](#)
SHARELIMIT パラメーター [168](#)
SMF
 SMSVSAM、タイプ 42 レコード [188](#)
SMF88SAB [200](#)
SMF88SIB [200](#)
SMSVSAM
 SMF タイプ 42 レコード [188](#)
SNA (システム・ネットワーク体系)
 装置の TIOA [142](#)
 トランザクション・フロー [146](#)
 メッセージ・チューニング [147](#)
SNA 経由 ISC (SNA 経由のシステム間連絡) [154](#)
SNA 経由のシステム間連絡 (SNA 経由 ISC) [154](#)
SNA の COS (サービス・クラス) [155](#)
SNA のサービス・クラス (COS) [155](#)
SNT (サインオン・テーブル)
 OPPRTY [66](#)
SOS [95](#)
SOS (ストレージ不足)
 一時データ・セットの使用 [52](#)
 制限状態 [55](#)
 発生、サブプール・ストレージのフラグメント化 [97](#)
 発生の検討 [11](#)
 CICS 制約 [52](#)
 Language Environment の、AMODE(24) プログラム用の
 ランタイム・オプション [52, 139](#)
SOS 状態 [92](#)
SQA (システム・キュー域) [121](#)
SRM (システム・リソース・マネージャー)
 GTF によってトレースされたアクティビティ [27](#)
STRINGS パラメーター [168, 173](#)
SUBTSKS [72](#)
SUBTSKS、システム 初期設定パラメーター [174](#)

T

TABLE パラメーター [176](#)
TCB 統計 [73](#)
TCP/IP の管理および制御 [36](#)
TCP/IP [36](#)
TCP/IP= ソケット・ドメインを指定する [224](#)
TD、システム 初期設定パラメーター [217](#)
TERMPRIORITY オペランド [66](#)
TIOA (端末入出力域) [142](#)
Tivoli Decision Support
 定期的レポート [10](#)
TSMINLIMIT [82](#)

U

UDSA [78](#)
UDSA サブプール [98](#)
UDSASZE [92](#)
USERMOD [133](#)

V

VERIFY TOKEN
 調整 [223](#)
VPACING オペランド [136](#)
VSAM

VSAM (続き)
 カタログ [170, 214](#)
 再始動データ・セット [154](#)
 サブタスキング [174](#)
 ストリング
 ESDS ファイルの [163, 166, 181](#)
 ストリングの待機 [55](#)
 調整 [163, 224](#)
 定義パラメーター [170](#)
 データ・セット [19](#)
 トランザクション [22, 161](#)
 入出力 [174](#)
 バッファ数 [163, 166, 181](#)
 複数のストリング [217](#)
 複数のバッファ [217](#)
 呼び出し [161](#)
 16 MB 境界 [119](#)
 AIX の考慮事項 [170](#)
 DSN の共用 [170](#)
 LSR の最大キー長 [168](#)
 LSR のストリング設定 [168](#)
 LSR のバッファ割り振り [168](#)
 LSR のリソース百分位数 (SHARELIMIT) [168](#)
 NSR のストリング設定 [173](#)
 NSR のバッファ割り振り [173](#)
VSAM レコード・レベル共用 (RLS) [186](#)

X

XZCOUT1、グローバル・ユーザー出口 (SNA) [151](#)

Z

z/OS
 データ収集
 IBM Z Decision Support [6, 36](#)
z/OS グローバル・リソースの逐次化 [219](#)
z/OS の Tivoli 決断サポート [6, 36, 37](#)
z/OS ワークロード・マネージャー
 種別規則 [230](#)
 パフォーマンス・ゴール [231](#)
 用語 [229](#)
 ワークロード [231](#)
 CICS パフォーマンス・パラメーターのチューニング
 [232](#)
zAAP プロセッサ [59](#)
zIIP プロセッサ [59](#)

