

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

*CICS* システム・プログラムの開発



## 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  
Developing CICS System Programs

### 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
<b>第 1 章ユーザー出口プログラムを使用したカスタマイズ.....</b>	<b>1</b>
グローバル・ユーザー出口プログラム.....	1
グローバル・ユーザー出口プログラムの作成.....	1
出口プログラムの定義、有効化、無効化.....	10
アクティブなグローバル・ユーザー出口の表示.....	10
複数の出口プログラムを単一の出口で呼び出す.....	11
単一の出口プログラムを複数の出口で呼び出す.....	11
タスク・トークン UEPTSTOK の使用.....	11
タスク関連のユーザー出口プログラム.....	12
タスク関連ユーザー出口のメカニズム (アダプター) の概要.....	12
スタブ・プログラム.....	14
タスク関連ユーザー出口プログラムの作成.....	16
アダプターの管理.....	46
アダプター追跡サンプル・タスク関連ユーザー出口プログラム (DFH\$APDT).....	48
ユーザー出口プログラミング・インターフェース (XPI).....	48
XPI の概要.....	48
XPI 呼び出しの実行.....	49
リリースを区別する XPI 呼び出し.....	53
グローバル・ユーザー出口 XPI の例 (ストレージの使用法).....	54
XPI の構文.....	59
<b>第 2 章初期設定プログラムとシャットダウン・プログラムを使用したカスタマイズ... 63</b>	<b>63</b>
処理設定プログラムとシャットダウンプログラムの作成.....	63
初期設定プログラムの作成.....	63
シャットダウン・プログラムの作成.....	66
初期設定プログラムおよびシャットダウン・プログラムの作成時の一般的な考慮事項.....	67
<b>第 3 章ユーザー置換可能プログラムを使用したカスタマイズ.....</b>	<b>69</b>
ユーザー置き換え可能プログラムとストレージ保護機能.....	69
プログラム・エラー・プログラムの作成.....	70
サンプルのプログラム・エラー・プログラム.....	75
カスタム EP アダプターの作成.....	75
トランザクション再始動プログラムの作成.....	79
DFHREST 通信域.....	80
CICS 提供のトランザクション再始動プログラム.....	81
端末エラー・プログラムの作成.....	82
順次装置のエラー処理に関する背景情報.....	82
サンプル端末エラー・プログラム.....	83
端末エラー・プログラムの作成.....	100
ノード・エラー・プログラムの作成.....	107
CICS-z/OS Communications Server のエラー処理の背景情報.....	108
異常条件が発生した場合.....	113
サンプル・ノード・エラー・プログラム.....	121
独自のノード・エラー・プログラムの作成.....	129
持続セッションでのノード・エラー・プログラムの使用.....	134
z/OS Communications Server 総称リソースでノード・エラー・プログラムを使用する.....	136
LU の自動インストールを制御するプログラムの作成.....	136
端末の自動インストール.....	136

INSTALL 時の自動インストール制御プログラム.....	138
DELETE 時の自動インストール制御プログラム.....	145
自動インストール制御プログラムの命名、テスト、およびデバッグ.....	146
「good night」プログラムの作成.....	147
端末用のサンプルの自動インストール制御プログラム.....	150
コンソールの自動インストールを制御するプログラムの作成.....	156
コンソールの自動インストール - 事前の考慮事項.....	156
INSTALL 時の自動インストール制御プログラム.....	157
DELETE 時の自動インストール制御プログラム.....	161
コンソールの自動インストール制御プログラム・サンプル.....	161
APPC 接続の自動インストールを制御するプログラムの作成.....	162
APPC 接続の自動インストール - 事前の考慮事項.....	162
INSTALL 時の自動インストール制御プログラム.....	163
DELETE 時の自動インストール制御プログラム.....	166
APPC 接続用自動インストール制御プログラムのサンプル.....	167
IPIC 接続の自動インストールを制御するプログラムの作成.....	169
IPIC 接続の自動インストール: 事前の考慮事項.....	169
INSTALL 時の自動インストール・ユーザー・プログラム.....	171
DELETE 時の自動インストール・ユーザー・プログラム.....	172
IPIC 接続 (IPCONN) 用の自動インストール・ユーザー・プログラムのサンプル.....	173
シッパされた端末の自動インストールを制御するプログラムの作成.....	174
シッパされた端末と接続のインストール.....	175
INSTALL 時の自動インストール制御プログラム.....	176
DELETE 時の自動インストール制御プログラム.....	178
サンプル・プログラムのデフォルトのアクション.....	179
仮想端末の自動インストールを制御するプログラムの作成.....	180
クライアント仮想端末を自動インストールする方法.....	180
ブリッジ・ファシリティー仮想端末はどのように自動インストールされるか.....	182
INSTALL 時の自動インストール制御プログラム.....	183
DELETE 時の自動インストール制御プログラム.....	186
サンプル・プログラムのデフォルトのアクション.....	188
プログラムの自動インストールを制御するプログラムの作成.....	188
プログラムの自動インストール: 事前の考慮事項.....	188
プログラムの自動インストールの利点.....	190
プログラムの自動インストールの構成.....	191
INSTALL 時の自動インストール制御プログラム.....	192
プログラムのための自動インストール制御プログラムのサンプル DFHPGADX.....	195
動的ルーティング・プログラムの作成.....	197
トランザクションの動的ルーティング.....	197
DPL 要求の動的ルーティング.....	204
ブリッジ要求の動的ルーティング.....	209
アプリケーションのコンテナの変更.....	213
ユーザー ID によるルーティング.....	213
動的ルーティング・プログラムに渡されるパラメーター.....	213
動的ルーティング・プログラムの命名.....	226
動的ルーティング・プログラムのテスト.....	226
動的トランザクション・ルーティングのサンプル・プログラム.....	226
分散ルーティング・プログラムの作成.....	227
分散ルーティング・インターフェースと動的ルーティング・インターフェースの相違点.....	227
BTS アクティビティーのルーティング.....	229
非端末関連 START 要求のルーティング.....	231
インバウンド Web サービス要求のルーティング.....	235
ユーザー ID によるルーティング.....	238
ターゲット領域での異常終了の処理.....	238
分散ルーティング・プログラムのリンク・チェックおよび情報.....	238
分散ルーティング・プログラムに渡されるパラメーター.....	239
分散ルーティング・プログラムの名前指定.....	247
分散トランザクション・ルーティング・サンプル・プログラム.....	247

CICS-DBCTL インターフェース 状況プログラムの作成.....	248
サンプルの CICS-DBCTL インターフェース 状況プログラム.....	249
3270 ブリッジ出口プログラムの作成.....	249
XPLink プログラム用に Language Environment ランタイム・オプションをカスタマイズするための プログラムの作成.....	250
DFHAPXPO.....	250
アナライザー・プログラム.....	250
アナライザー・プログラムと URIMAP 定義の置き換え.....	252
アナライザー・プログラムの作成.....	253
アナライザー・プログラムとコンバーター・プログラムでのデータの共用.....	257
アナライザー・プログラムからのエスケープ・データまたはアンエスケープ・データの選択.....	258
CICS 提供のアナライザー・プログラム DFHWBAAX.....	258
CICS 提供のサンプル・アナライザー・プログラム DFHWBADX.....	259
コンバーター・プログラムの作成.....	261
コンバーター・プログラム・デコード機能の入力パラメーター.....	263
コンバーター・プログラム・デコード機能の出力パラメーター.....	264
コンバーター・プログラム・エンコード機能の入力パラメーター.....	264
コンバーター・プログラム・エンコード機能の出力パラメーター.....	265
コンバーター・プログラムからの複数のアプリケーション・プログラムの呼び出し.....	265
<b>第 4 章統計収集分析プログラムの作成.....</b>	<b>267</b>
CICS 統計を収集するプログラムの作成.....	267
CICS 統計を収集する理由.....	267
統計カウンターのリセット・オプション.....	267
CICS 統計の収集と抽出.....	268
CICS 統計レコードの形式.....	269
SMF ヘッダーおよび SMF 製品セクション.....	269
CICS 統計データ・セクション.....	271
XSTOUT グローバル・ユーザー出口プログラムを使用した統計レコードのフィルタリング.....	275
CICS 統計の出力の処理.....	275
CICS TS フォーマットのジャーナル・レコードの構造および内容.....	275
汎用ログ・ブロック・ヘッダー.....	276
汎用ログ・ジャーナル・レコード.....	277
呼び出し元データ.....	279
実行開始時レコード.....	295
COMPAT41 フォーマットのジャーナル・レコードの構造および内容.....	295
COMPAT41 ジャーナル管理ラベル・ヘッダー.....	297
ジャーナル・レコードのフォーマット.....	300
タスクおよび UOW の開始のレコードの特定.....	307
SMF に書き込まれたジャーナル・レコードの形式.....	307
SMF ブロック・ヘッダー.....	309
CICS 製品セクション.....	310
CICS データ・セクション.....	312
<b>第 5 章 CICS 互換インターフェースの開発.....</b>	<b>313</b>
動的割り振りプログラムの概要.....	313
プログラム定義とトランザクション定義のインストール.....	313
動的割り振りプログラム: 端末操作.....	314
動的割り振りプログラムのヘルプ機能の使用.....	314
動的割り振りプログラム: 値.....	314
キーワードの省略規則.....	315
システム・プログラミングに関する考慮事項.....	315
DYNALLOC 要求が出されたときの制御の流れ.....	316
<b>第 6 章ユーザー作成プログラムによるリソース定義操作のカスタマイズ.....</b>	<b>319</b>
CEDA とのプログラマブル・インターフェースの使用.....	319
DTP 環境での DFHEDAP の使用.....	320

システム定義ユーティリティー・プログラム (DFHCSDUP) 用ユーザー・プログラム .....	321
DFHCSDUP からのユーザー・プログラムの呼び出し.....	321
ユーザー・プログラムからの DFHCSDUP の呼び出し.....	330
DFHCSDUP のユーザー出口点.....	332
サンプル・プログラム DFH\$CUS1.....	336
ユーザー置換可能プログラムのアセンブルとリンク・エディット .....	337
<b>第 7 章セキュリティ処理のカスタマイズ.....</b>	<b>339</b>
ユーザー提供の ESM への制御の受け渡し.....	339
非 RACF ユーザーの場合 - ESM パラメーター・リスト.....	339
RACF ユーザーの場合 - RACF ユーザー出口パラメーター・リスト.....	340
インストール・データ・パラメーター・リストのマッピング.....	341
早期検証処理の使用.....	343
早期検証ルーチンの作成.....	344
早期検証ルーチンでの CICS API コマンドの使用.....	344
早期検証ルーチンからの戻りコードと理由コード.....	345
CICS セキュリティー制御点.....	345
非端末トランザクションの接続検査の抑止.....	348
サインオンおよびサインオフのグローバル・ユーザー出口.....	349
<b>付録 A z/OS Communications Server LOGON モード・テーブルのエントリーのコー ディング .....</b>	<b>351</b>
z/OS Communications Server LOGON モード・テーブルの概要.....	351
z/OS Communications Server MODEENT マクロ・オペランド.....	351
TYPETERM デバイス・タイプと関連 LOGON モード・データを指すポインター.....	356
LUTYPEX デバイスに関する PSERVIC 画面サイズ値.....	358
対応するモデルと LOGON モード・エントリー.....	359
CICS 提供の自動インストール・モデル用の LOGON モード定義.....	368
<b>付録 B ノード異常条件プログラムのデフォルト・アクション.....</b>	<b>371</b>
DFHZNAC: 端末エラー・コードに関するデフォルトのアクション.....	371
z/OS Communications Server エラーに関連した CICS メッセージ.....	377
DFHZNAC: システム・センス・コードに関するデフォルトのアクション.....	384
アクション・フラグの設定値および意味.....	386
<b>特記事項.....</b>	<b>389</b>
<b>索引.....</b>	<b>395</b>

## 本書について

---

本書では、システムを調整する方法として、出口プログラムのコーディング、特定の CICS 提供のデフォルト・プログラムに代わる独自に作成したバージョンへの置き換え、サンプル・プログラムの調整について説明します。さらに、「XPI 関数リファレンス」および「グローバル・ユーザー出口リファレンス」という関連する 2 つの解説書の PDF が必要になる場合もあります。CICS TS V5.4 より前は、本書とこの 2 つの解説書 PDF の情報は、「カスタマイズ・ガイド」という 1 つの PDF に収録されていました。以下にリストされているその他の PDF では、CICS の特定の領域をカスタマイズする方法について説明しており、本書に加えて参照が必要になることがあります (IBM Knowledge Center では、これらの情報はすべて「Developing system programs」と呼ばれる 1 つのセクションにまとめられています)。

CICS の領域に関するカスタマイズ情報は、以下の PDF に記載されています。

- SOAP と JSON については、「*Web サービス・ガイド*」に記載されています。
- 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・ユーザーズ・ガイド*」に記載されています。
- 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*」に記載されています。
- CICSplex SM については、「*CICSplex SM Administration*」に記載されています。
- 外部セキュリティー・マネージャーについては、「*RACF Security Guide*」に記載されています。

本書で使用されている用語と表記の詳細については、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。

### 本書の作成日

本書は、2020 年 5 月 28 日に作成されました。





# 第 1 章 ユーザー出口プログラムを使用したカスタマイズ

CICS は、作成したプログラム (ユーザー出口プログラム) に制御を転送するためのいくつかの出口点を提供しています。出口プログラムを使用して、CICS の動作方法を拡張または変更することができます。

## グローバル・ユーザー出口プログラム

CICS のグローバル・ユーザー出口点 と、ユーザーが独自に作成するグローバル・ユーザー出口プログラム という特別なタイプのプログラムを使用して、CICS 領域をカスタマイズできます。

グローバル・ユーザー出口点またはグローバル・ユーザー出口 とは、CICS がユーザー作成のグローバル・ユーザー出口プログラムに制御権を移動できる CICS モジュール内またはドメイン内の場所です。出口プログラムが処理を終了すると、CICS は制御権を取り戻します。

各グローバル・ユーザー出口点は、固有の ID を持ち、何らかの追加処理を実行することが有用なモジュール内またはドメイン内のポイントに置かれています。例えば、統計ドメイン内の出口点 XSTOUT では、各統計レコードが SMF データ・セットに書き込まれる前に出口プログラムに制御権を与え、関連する統計レコードに出口プログラムからアクセスすることができます。この出口点で出口プログラムを使用して、統計レコードを検査して不要なレコードの書き込みを抑止したりできます。

グローバル・ユーザー出口のサポートは CICS が自動的に提供します。一方、ユーザー独自の出口プログラムを作成する方法についてはいくつかの規則があります。これらの規則については、『[グローバル・ユーザー出口プログラムの作成](#)』で説明しています。グローバル・ユーザー出口プログラムは CICS モジュールまたはドメインの一部であるかのように動作するため、CICS サービスの使用にはいくつかの制限があります。ほとんどのグローバル・ユーザー出口プログラムは **EXEC CICS** コマンドを使用できませんが、既存のプログラミング・インターフェース (XPI) を使用して一部の CICS サービスを呼び出すことができます。詳しくは、『[CICS サービスの使用](#)』を参照してください。

**注:** CICS 管理モジュールのソースとオブジェクトの互換性は、異なるリリース間では保証されません。出口プログラムに影響がある変更については、アップグレード資料に記載されます。

CICS には、2 セットのサンプルとグローバル・ユーザー出口サンプル・プログラムが用意されています。詳しくは、『[サンプル](#)』を参照してください。

## グローバル・ユーザー出口プログラムの作成

グローバル・ユーザー出口プログラムをアセンブラー言語で作成し、それらを準再入可能にする必要があります。ただし、ユーザー出口プログラムが XPI を呼び出す場合は、完全再入可能にする必要があります。

**要確認:** 再入可能なプログラムは、自身のコピーをいくつかのタスクで同時に使用できるようにコーディングされています。実行中に自身に対して変更を行うことはありません。準再入可能なプログラムは、さまざまなタスクで逐次再使用可能です。制御を受け取るときと制御を解放するときの状態が同じでなければなりません。この種のプログラムは実行中に自身に変更を行うことがあります。これが、完全に再入可能ではない点です。

準再入可能なプログラムについて詳しくは、『[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。

### レジスター規則

出口プログラムへの入り口でレジスター値が渡されます。特定のレジスター値のみが保証されています。

出口プログラムへの入り口で利用できるレジスター値は次のとおりです。

- レジスター 1 には、ユーザー出口パラメーター・リスト DFHUEPAR のアドレスが入っています。

オペレーターへの書き込み (WTO) コマンドはレジスター 1 を使用します。出口プログラムで WTO コマンドを使用する場合は、先に DFHUEPAR のアドレスを保管する必要があります。

- レジスター 13 には、呼び出された後すぐに出口プログラムが自身のレジスターを保管する標準レジスター保存領域のアドレスが格納されます。このアドレスは、レジスター 1 が指すパラメーター・リストのフィールド UEPEPSA のアドレスにも入っています。

レジスター 13 を使用して保管域を指すオペレーティング・システム要求を出す場合は、レジスター 13 を別の保管域を指すように切り替える必要があります。ユーザー出口プログラムから呼び出し元に戻る前に、レジスター 13 を元の内容に戻す必要があります。

- レジスター 14 には、作業完了時に出口プログラムが分岐する戻りアドレスが格納されます。これを行うには、呼び出し元モジュールのレジスターを復元した後に BR 14 命令を使用するか、または RETURN マクロを使用します。
- レジスター 15 には、出口プログラムの入り口アドレスが格納されます。

出口プログラムは、レジスター 13 でアドレス指定された保管域を使用して、変更したレジスターの保管および復元を行う必要があります。

### 31 ビット・アドレッシング実装

次のリストに CICS のアドレス指定とアクセス・レジスターに関する考慮事項を示します。

- グローバル・ユーザー出口は 31 ビット AMODE で開始されます。
- グローバル・ユーザー出口は RMODE 24 または RMODE ANY です。
- 出口プログラムで 24 ビット AMODE に切り替える必要がある場合は、必ず、31 ビット AMODE に正しく戻してください。
- XPI 呼び出しを使用する場合は、出口プログラムは 31 ビット AMODE でなければなりません。
- DFHUEPAR で渡される一部のパラメーターは、31 ビット・ストレージ (16 MB 境界の上) のアドレスです。
- 出口プログラムのグローバル作業域は、24 ビット・ストレージにも 31 ビット・ストレージにも配置できます。出口を定義するときに、ENABLE PROGRAM コマンドの GALLOCATION オプションを使用してストレージの場所を指定します。出口プログラムのグローバル作業域のアドレスを確認するには、EXTRACT EXIT コマンドを使用できます。

### アクセス・レジスターに関する考慮事項

- グローバル・ユーザー出口は 1 次スペース変換モードで開始されます。変換モードについては、[z/Architecture](#) 解説書を参照してください。
- アクセス・レジスターの内容は予測できません。アクセス・レジスターについては、[z/Architecture](#) 解説書を参照してください。
- グローバル・ユーザー出口でアクセス・レジスターを変更した場合は、制御権を戻す前にそれらを復元する必要があります。CICS はこの目的のための保管域を提供しません。
- グローバル・ユーザー出口は 1 次アドレッシング・モードで制御権を戻す必要があります。

### CICS サービスの使用

出口プログラムでの CICS サービスの使用を制限する規則は、出口プログラムが開始される出口点によって異なります。

### このタスクについて

以下の一般規則が適用されます。

- ディスパッチャー・ドメイン内の出口点からは CICS サービスを開始できません。
- ほとんどの出口から、出口プログラミング・インターフェース (XPI) を使用して CICS サービスを開始することができます。XPI を使用する場合は、各出口および各 XPI マクロについてリストされている規則および制限に注意してください。XPI については、『[ユーザー出口プログラミング・インターフェース \(XPI\)](#)』で説明しています。
- 出口によっては、一部の CICS サービスを EXEC CICS コマンドを使用して要求できます。有効なコマンドを、各出口の詳細な説明中にリストしています。リストされているコマンドがない場合は、サポートされている EXEC CICS API または SPI コマンドがないことを意味します。XCTL が (直接または暗黙的に)

実行される EXEC CICS コマンド (**EXEC CICS XCTL** や **EXEC CICS SHUTDOWN** など) は、決して使用しないでください。

EXEC CICS コマンドの使用をサポートしていない出口で開始される出口プログラムでは、タスク関連ユーザー出口プログラム (TRUE) を呼び出してはいけません。TRUE を呼び出すことは EXEC CICS コマンドを発行することと同じです。この規則の例外は XFCFRIN 出口および XFCFROUT 出口から開始されるプログラムです。それらは TRUE を呼び出すことができます。TRUE については、『[タスク関連ユーザー出口プログラム](#)』で説明しています。

**注：** EXEC CICS ファイル制御コマンドの使用をサポートする出口では、一連のシーケンスを形成するすべてのファイル・コマンド (**EXEC CICS STARTBR**、**EXEC CICS READNEXT**、**EXEC CICS ENDBR** など) を出口プログラムの同じ呼び出しで発行する必要があります。

例えば、出口プログラムのある呼び出しで **EXEC CICS STARTBR** コマンドを発行し、その同じタスクのために出口プログラムの次の呼び出しで **EXEC CICS READNEXT** コマンドを発行した場合、READNEXT は INVREQ 状態で失敗します。

- EXEC CICS コマンドを発行するすべての出口プログラムは最初に EIB をアドレス指定する必要があります。通常の EXEC アセンブラー言語プログラムの場合のように、DFHEIENT マクロを使用してアドレス指定が自動的に行われることはありません。したがって、出口プログラムから発行される最初の EXEC コマンドは、EXEC CICS ADDRESS EIB (eib-register) である必要があります。ここで、「eib-register」はデフォルトのレジスター (R11) か、DFHEIENT マクロにパラメーターとして渡されたレジスターです。

EXEC CICS コマンドを発行して DFHEIENT マクロを使用するすべての出口プログラムは、DFHEIRET マクロを使用して、戻りコードを設定して CICS に戻す必要があります。[8 ページの『CICS に値を戻す』](#)を参照してください。

**注：**

- グローバル・ユーザー出口プログラムに EXEC CICS コマンドを含めない場合は、プログラムをアセンブルするときに CICS コマンド・レベル変換プログラムを使用しないでください。
- グローバル・ユーザー出口プログラムから CICS 以外の (RACF® や MVS™ などの) システム・サービスを呼び出さないでください。
- オペレーティング・システム要求で待機が発生すると、そのオペレーティング・システム要求が処理されるまで CICS システム全体が停止します。

### 同じ出口プログラムでの EXEC CICS 呼び出しと XPI 呼び出しの使用

いくつかの出口では EXEC CICS コマンドと XPI 呼び出しの両方を使用できますが、レジスター 13 の使用が競合しないように注意する必要があります。

このような競合を回避するために、DFHEIENT マクロで DATAREG オプションを使用してください (詳しくは、[53 ページの『XPI レジスターの使用法』](#)を参照)。

同じグローバル・ユーザー出口プログラムで EXEC CICS コマンドと XPI 呼び出しを使用する例については、[グローバル・ユーザー出口サンプル・プログラム DFH\\$XTSE](#) を参照してください。

### チャンネルおよびコンテナの使用

グローバル・ユーザー出口プログラムは、アプリケーション・プログラムによって作成されたチャンネルおよびコンテナにアクセスできます。また、独自のチャンネルを作成し、呼び出したプログラムに渡すこともできます。

チャンネルおよびコンテナについては、[チャンネルによるプログラム間データ転送](#)を参照してください。

### アセンブラー・プログラムと LEASM

LEASM オプションを指定して変換されたアセンブラー・プログラムは、グローバル・ユーザー出口プログラムとして使用できません。

LEASM は、Language Environment® に準拠したアセンブラーのメインプログラムを生成するために使用します。LEASM 変換プログラム・オプションについて詳しくは、[変換プログラム・オプション](#)を参照してください。

## EDF とグローバル・ユーザー出口

実行診断機能 (EDF) を使用してアプリケーションをデバッグする場合は、EXEC CICS コマンドを発行する出口プログラムをコンパイルするときに注意が必要です。

通常、出口プログラムが EXEC CICS コマンドを発行する場合は、EDF がアクティブであれば EDF によってそれらのコマンドが表示されます。これらのコマンドは、出口を起動したコマンドの「Start of Command (コマンドの開始)」画面と「End of Command (コマンドの終了)」画面の間に表示されます。出口プログラムから発行された EXEC CICS コマンドの表示を抑止する場合は、プログラムを変換するときに NOEDF オプションを指定する必要があります。実稼働環境のプログラムには必ず NOEDF を指定する必要があります。

リカバリー処理中に呼び出される可能性がある出口プログラムが EXEC CICS コマンドを発行する場合は、NOEDF オプションを指定して変換する必要があります。指定しないと、EDF が異常終了します。

## グローバル作業域

出口プログラムを有効にするときは、その出口プログラムのグローバル作業域を提供するように CICS に要求できます。出口プログラム専用のグローバル作業域を作成することも、別の出口プログラムが所有する作業域を共用することもできます。

出口を定義するために **ENABLE PROGRAM** コマンドを発行するときに GALENGTH オプションと GALLOCATION オプションを指定して、出口プログラム用のグローバル作業域を CICS に用意させることができます。GALENGTH はグローバル作業域の長さをバイト単位で指定し、GALLOCATION は作業域を 24 ビット・ストレージに置くか 31 ビット・ストレージに置くかを指定します。また、GAENTRYNAME オプションを指定して、現在有効にされている別のユーザー出口を指定し、その出口用に CICS が既に用意したグローバル作業域をこの出口プログラムで共用することもできます。

グローバル作業域は、出口点ではなく出口プログラムに関連付けられます。問題判別を簡単にするために、グローバル作業域を共用するのは同じ管理モジュールまたはドメインから呼び出された出口プログラムに限る必要があります。グローバル作業域のアドレスおよび長さは、DFHUEPAR パラメーター・リストの **UEPGAA** および **UEPGAL** によってアドレス指定されます (5 ページの『DFHUEPAR 標準パラメーター』を参照)。ユーザー出口プログラム専用のグローバル作業域がない場合は、UEPGAA はゼロに設定されます。

アプリケーション・プログラムは、同じグローバル作業域を使用または共用するユーザー出口プログラムと情報を交換できます。アプリケーション・プログラムでは **EXEC CICS EXTRACT EXIT** コマンドを使用してグローバル作業域のアドレスと長さを取得します。

作業域は、その作業域を使用するすべての出口プログラムが無効にされたときに初めて解放されます。グローバル作業域の使用例については、[グローバル・ユーザー出口ファウンデーション・サンプル](#) にリストされているグローバル・ユーザー出口プログラムのサンプルを参照してください。

## トレース・エントリーの作成

トレースがアクティブな場合は、出口プログラムの実行の直前および直後に CICS のトレース・テーブルにエントリーを作成するように指定できます。

## 手順

- 次のいずれかのメソッドを使用して、出口プログラムの実行前後にトレースのエントリーを作成できます。
  - CETR トランザクションの UE オプション。
  - EXEC CICS SET TRACETYPE** コマンドの UE オプション。
- グローバル・ユーザー出口がドメインにある場合は、トレース呼び出しを追加し、**EXEC CICS SET TRACETYPE** の AP オプションをレベル 1 または 2 に設定して追加の診断情報を提供することができます。
- 使用する出口点によっては、XPI DFHTRPTX TRACE\_PUT マクロを使用してユーザー出口プログラムでトレース・エントリーを作成できる場合もあります。

このマクロについては、48 ページの『ユーザー出口プログラミング・インターフェース (XPI)』に説明があります。グローバル・ユーザー出口点についての個々の説明に、XPI DFHTRPTX マクロの使用可否を記載しています。



## グローバル・ユーザー出口プログラムに渡されるパラメーター

パラメーター・リストのアドレスは、レジスター 1 でユーザー出口プログラムに渡されます。リストにはすべてのグローバル・ユーザー出口プログラムに渡される標準的なパラメーターがいくつかと、場合により、出口固有のパラメーター (出口プログラムが呼び出される出口点に固有のもの) がいくつか含まれています。

出口固有のパラメーターについては、[グローバル・ユーザー出口ルーチンのポイント](#)のセクションで出口ごとに説明しています。標準的なパラメーター・リストについては次のセクションで説明します。

パラメーター・リストは、次のマクロ命令により生成される DSECT DFHUEPAR を使用してマップできます。

```
DFHUEEXIT TYPE=EP,ID=exit_point_identifier
```

ID パラメーターにより、出口固有のパラメーターをマップするために必要な追加のデータ定義を指定します。例えば、次のマクロ命令は、

```
DFHUEEXIT TYPE=EP,ID=XTDIN
```

一時データ・プログラムで標準的なパラメーターをマップした後に出口点 XTDIN に固有のパラメーターをマップする DSECT を生成します。出口プログラムを複数の出口点で呼び出す場合は、単一の DFHUEEXIT マクロ命令で該当する出口 ID を最大 256 文字までコーディングできます。以下に例を挙げます。

```
DFHUEEXIT TYPE=EP,ID=(XMNOUT,XSTOUT,XTDIN)
```

出口プログラムをすべてのグローバル・ユーザー出口点で呼び出す場合は、次をコーディングできます。

```
DFHUEEXIT TYPE=EP,ID=ALL
```

ユーザー出口プログラムをグローバル・ユーザー出口プログラムとしてもタスク関連ユーザー出口プログラムとしても使用する場合は、次の両方をコーディングする必要があります。

```
DFHUEEXIT TYPE=EP,ID=exit_point_identifier
```

かつ

```
DFHUEEXIT TYPE=RM
```

(この順序で) コーディングすることで、両方のタイプのユーザー出口にとって適切な DSECT を生成できます。

グローバル・ユーザー出口プログラムで DFHRMCAL マクロを使用して外部 RMI を呼び出す必要がある場合は、DFHRMCAL マクロ命令を DFHUEEXIT マクロの後に実行する必要があります。

## DFHUEPAR 標準パラメーター

DFHUEPAR 標準パラメーターはすべてのグローバル・ユーザー出口プログラムに渡されます。

```
DFHUEPAR DSECT
* STANDARD PARAMETERS
UEPEXN DS A ADDRESS OF EXIT NUMBER
UEPGAA DS A ADDRESS OF GLOBAL WORK AREA
* (ZERO = NO WORK AREA)
UEPGAL DS A ADDRESS OF GLOBAL WORK AREA LENGTH
UEPCRC DS A ADDRESS OF CURRENT RETURN-CODE
UEPTCA DS A RESERVED
UEPCSA DS A RESERVED
UEPEPSA DS A ADDRESS OF REGISTER SAVE AREA
* FOR USE BY EXIT PROGRAM
UEPHMSA DS A ADDRESS OF SAVE AREA USED FOR
* HOST MODULE'S REGISTERS
UEPGIND DS A ADDRESS OF CALLER'S TASK INDICATORS
UEPSTACK DS A ADDRESS OF KERNEL STACK ENTRY
UEPXSTOR DS A ADDRESS OF STORAGE FOR XPI PARAMETERS
UEPTRACE DS A ADDRESS OF TRACE FLAG
```

## UEPEXN

出口プログラムが呼び出されたグローバル・ユーザー出口点を示す内容の 1 バイトの 2 進数フィールドを指します。複数の出口点から出口プログラムを呼び出せるようにするには、この情報が必要です。

**DFHUEXIT TYPE=EP** は、出口名 (出口 ID) を出口番号 (出口を識別するために CICS で内部的に使用されるもの) に関連付ける等式の値のリストを生成します。出口番号は CICS の今後のリリースで変更される可能性があるため、必ず出口 ID を使用してください。

## UEPGAA

出口プログラムを有効にしたときに提供されたグローバル作業域を指します。グローバル作業域が提供されなかった場合、このパラメーターはゼロに設定されます。

## UEPGAL

グローバル作業域の長さを含むハーフワードを指します。

## UEPCRSA

出口プログラムからの戻りコード値が含まれるハーフワードを指します。1 つのユーザー出口で複数のプログラムが呼び出される場合、(2 番目以降のプログラムに入るときには) このフィールドに前に呼び出されたプログラムによって設定された戻りコードが含まれています。

同じ出口点で前の出口プログラムから返された戻りコードとは違う戻りコードを出口プログラムで設定する例については、[11 ページの『複数の出口プログラムを単一の出口で呼び出す』](#)のコード・スニペットを参照してください。

## UEPTCA

フェッチ保護ストレージを指します。このフィールドを使用すると実行時に ASRD が異常終了します。

## UEPCSA

フェッチ保護ストレージを指します。このフィールドを使用すると実行時に ASRD が異常終了します。

## UEPEPSA

出口プログラムが入り口で自身のレジスターを格納する保管領域を指します。出口プログラムの開始時は、レジスター 13 もこの領域を指しています。規則により、レジスター 14、15、およびオフセット 12 (10 進数) の 0 から 12 までは順次保管されます。

## UEPHMSA

呼び出し元モジュールのレジスターが含まれている保管領域を指します。レジスター 14、15、および 0 から 13 までの値が、この順序でこの領域にオフセット 12 (10 進数) から格納されます。

出口プログラムからの戻りコード値が含まれるレジスター 15 は別として、CICS はこの保管領域の値を使用して、呼び出し元の CICS モジュールに戻るときにレジスターを再ロードします。これらの値を破損させてはいけません。

このアドレスは、CICS ドメインの出口点から呼び出されるグローバル・ユーザー出口プログラムには渡されません。

## UEPGIND

AP ドメイン・ユーザー出口で使用するための標識を含む 3 バイトのフィールドを指します。AP ドメイン・ユーザー出口でない場合、この標識は常にゼロです。

最初の標識バイトは 2 つのシンボル値、UEPGANY および UEPGCICS のいずれかです。これらの値を検査して、データ・ロケーションを 16 MB 境界の上下のどちらに置けるか、およびアプリケーション・ストレージが CICS キー・ストレージとユーザー・キー・ストレージのどちらにあるのかを調べることができます。

## UEPGANY

アプリケーションは、16 MB 境界より上のアドレスを受け入れることができます。シンボル値が UEPGANY でない場合、アプリケーションは 16 MB 境界より下のアドレスのみを受け入れることができます。

## UEPGCICS

アプリケーションの作業用ストレージおよびタスク存続時間ストレージは CICS キー・ストレージにあります (TASKDATAKEY=CICS)。シンボル値が UEPGCICS でない場合、アプリケーションの作業用ストレージとタスク存続時間ストレージはユーザー・キー・ストレージにあります (TASKDATAKEY=USER)。

2 番目と 3 番目のバイトにはグローバル・ユーザー出口プログラムの呼び出し元の TCB モードを示す値が含まれます。この値は DFHUEPAR で 2 文字のコードとシンボル値の両方として次のように表されます。

表 1. DFHUEPAR の TCB 標識		
シンボル値	2 バイト のコード	説明
UEPTQR	QR	準再入可能モード TCB
UEPTRO	RO	リソース所有モード TCB
UEPTCO	CO	並行モード TCB
UEPTSZ	SZ	FEPI モード TCB
UEPTRP	RP	ONC/RPC モード TCB
UEPTFO	FO	ファイル所有モード TCB
UEPTSL	SL	ソケット・リスナー・モード TCB
UEPTSO	SO	ソケット・モード TCB
UEPTS8	S8	セキュア・ソケット層モード TCB
UEPTD2	D2	CICS Db2 <sup>®</sup> ハウスキーピング・モード TCB
UEPTL8	L8	L8 オープン TCB。OPENAPI TRUE、または CICS キーの OPENAPI プログラムに使用。
UEPTL9	L9	L9 オープン TCB。ユーザー・キーの OPENAPI プログラムに使用。
UEPTEP	EP	イベント処理 TCB
UEPTTP	TP	Language Environment エンクレーブおよび THRD TCB プールを JVM サーバー用に所有するために使用される TP オープン TCB。
UEPTT8	T8	T8 TCB。マルチスレッド処理を実行するために JVM サーバーで使用。
UEPTX8	X8	X8 オープン TCB。CICS キーの、XPLINK オプションを指定してコンパイルされた C および C++ プログラムに使用。
UEPTX9	X9	X9 オープン TCB。ユーザー・キーの、XPLINK オプションを指定してコンパイルされた C および C++ プログラムに使用。

## UEPSTACK

カーネル・スタック・エントリーを指します。この値は XPI を呼び出す前に出口プログラムのレジスター 13 に移す必要があります。詳しくは、[48 ページの『ユーザー出口プログラミング・インターフェース\(XPI\)』](#)を参照してください。このフィールドでアドレス指定されたストレージを変更してはいけません。これが壊れた場合、出口プログラムが CICS システムに与える影響は予測できません。

## UEPXSTOR

出口プログラムが XPI を呼び出すときに使用する DFHUEH 所有の LIFO ストレージの 1024 バイト域を指します。詳しくは、[48 ページの『ユーザー出口プログラミング・インターフェース\(XPI\)』](#)を参照してください。

## UEPTRACE

呼び出し元管理モジュールまたはドメインでトレースがオンであるかどうかを示すトレース・フラグを指します。このパラメーターを使用すると、CICS モジュールまたはドメインのトレースに合わせて XPI TRACE\_PUT マクロの使用を制御することができます。XPI TRACE\_PUT 機能はトレースがオンの場合にのみ使用してください。トレース・フラグは 1 バイトです。トレースがオンに切り替わると最上位ビットがオンに設定されます。この設定をテストするには、シンボル値 UEPTRON を使用します。UEPTRACE でアドレス指定された残りのバイトは予約されており、その内容を破損させてはいけません。

## CICS に値を戻す

一部の出口点では、戻りコード値を渡して、CICS が出口プログラムから戻るときに実行する内容を操作することができます。

## このタスクについて

出口プログラムを離れる前に、戻りコード値をレジスター 15 に設定する必要があります。

## 手順

- ハードコーディングした値ではなく、文字列の値を使用します。  
有効な戻りコード値に相当する文字列を、各出口点のパラメーター・リストを使用して渡します。  
例えば、モニター・ドメイン内の出口 XMNOUT では、モニター・レコードのアドレスが与えられます。このレコードを SMF に書き込まないことを出口プログラムで決定した場合は、CICS に戻る前に戻りコード値 UERCBYP (「このレコードを無視する」という意味) を設定して、CICS にレコードを抑止させることができます。
- 1つの出口点で複数の出口プログラムを実行する場合は、DFHUEPAR のパラメーター **UEPCRCA** を使用して戻りコードを設定します。  
詳しくは、[11 ページの『複数の出口プログラムを単一の出口で呼び出す』](#)を参照してください。
- 出口プログラムで EXEC CICS コマンドを発行し、DFHEIENT マクロを使用する場合、このマクロを使用して戻りコードを設定する必要があります。  
DFHEIRET マクロで以下を実行します。
  - レジスターを復元する。
  - レジスターが復元されたら、戻りコードをレジスター 15 に置く。
  - レジスター 14 のアドレスに制御を返す。

以下に例を挙げます。

```
DFHEIRET RCREG=nn
```

ここで、*nn* は、レジスターの復元後にレジスター 15 に収容される戻りコードが入っているレジスターの番号 (13 以外) です。

## タスクの結果

特定の出口点で予期されない戻りコードを渡すと (戻りコード UERCPURG を設定した場合を除く)、正常な応答を示すデフォルトの戻りコード (一般的には UERCNORM) として受け取られます。結果が予測不能になるため、戻りコードをデフォルトの正常応答にさせないようにすることを強くお勧めします。正常な応答は、出口プログラムが呼び出されなかったかのように処理を再開することを CICS に命令します。これは、ほとんどのグローバル・ユーザー出口点で有効なオプションです。例外は、各出口の説明と一緒に記載している戻りコードのリスト中に示しています。

## プログラミング・インターフェースとしてフィールドを使用する場合の制約

CICS データ域制御ブロックのフィールド定義には、CICS アプリケーション・プログラミング・インターフェースの一部として使用してはいけないものがあります。

データ域に、CICS のプロダクト・センシティブ・プログラミング・インターフェースおよび汎用プログラミング・インターフェースの一部を形成する制御ブロック・フィールドの定義を記載しています。「[CICS Data Areas](#)」資料でプロダクト・センシティブ・プログラミング・インターフェースとも汎用プログラミング・インターフェース・フィールドとも定義されていないフィールドは、CICS プログラミング・インターフェースの一部として使用することを目的としていません。

## 出口プログラムと CICS ストレージ保護機能

ストレージ保護機能を使用して CICS を実行する場合、ユーザー出口プログラムが実行される実行キーと、出口プログラムが取得するデータ・ストレージのストレージ・キーが影響を受けます。

## グローバル・ユーザー出口プログラムの実行キー



ストレージ保護をアクティブにして実行中の場合、CICS は必ず CICS キーでグローバル・ユーザー出口プログラムを呼び出します。プログラム・リソース定義に EXECKEY(USER) を指定した場合でも、CICS は出口プログラムに制御を渡すときに CICS キーを強制します。ただし、グローバル・ユーザー出口プログラム自体が (リンク・コマンドまたは制御権移動コマンドによって) 別のプログラムに制御を渡す場合は、こうして呼び出されたプログラムは、プログラム・リソース定義に定義された実行キー (EXECKEY) に従って実行されます。

グローバル・ユーザー出口プログラムと、出口プログラムから制御が渡されるプログラムの両方を定義する場合は、EXECKEY(CICS) を指定することを強くお勧めします。

## グローバル・ユーザー出口プログラムのデータ記憶キー

グローバル・ユーザー出口プログラムによって使用されるストレージのストレージ・キーは、ストレージの取得方法に応じて異なります。

- DFHUEPAR の **UEPXSTOR** パラメーターによってアドレス指定される CICS 提供のストレージと、出口プログラムが有効にされるときに指定されるグローバル作業域は、常に CICS キーのものととなります。
- EXEC CICS コマンドを発行できるグローバル・ユーザー出口プログラムは、以下を使用してストレージを取得できます。
  - 明示的な **EXEC CICS GETMAIN** コマンド
  - SET オプションを使用した EXEC CICS コマンドの結果としてなされる暗黙的なストレージ要求。

EXEC CICS コマンドによって取得されるストレージのデフォルト・ストレージ・キーは、出口プログラムの呼び出しで使用されるトランザクションの TASKDATAKEY によって設定されます。

例として、TASKDATAKEY(USER) で定義されたトランザクションが、ファイル制御要求を発行し、その結果として XFCREQ グローバル・ユーザー出口プログラムが呼び出される場合について検討します。この場合、EXEC CICS コマンドによって出口プログラムで獲得される暗黙的または明示的なストレージは、デフォルトでユーザー・キー・ストレージのものです。ただし、EXEC CICS GETMAIN コマンドにおいて、出口プログラムは CICS DATAKEY または USER DATAKEY を指定することで TASKDATAKEY オプションをオーバーライドできます。

- 出口プログラムが XPI GETMAIN 呼び出しを使用してストレージを取得する場合、ストレージ・キーは、TASKDATAKEY の値をオーバーライドする必須の STORAGE\_CLASS オプションに指定された値に応じて異なります。

## 出口プログラムとトランザクション分離

トランザクション分離機能を使用して (TRANISO=YES) CICS を実行している場合、出口プログラムは、出口の呼び出しを引き起こしたアプリケーションのサブスペースを継承します。

GLUE が、呼び出し元タスク以外のタスクに属するストレージにアクセスする必要がある場合、その GLUE は **DFHSMRX SWITCH\_SUBSPACE XPI** コマンドを使用して基本スペースに切り替える必要があります。CICS に戻る前にサブスペース・モードに切り替えて戻す必要はありません。CICS が必要に応じてサブスペース・モードを復元するためです。

## ユーザー出口プログラムのエラー

グローバル・ユーザー出口プログラムは CICS コードの拡張であるため、呼び出されたときの CICS の実行環境の影響を受けます。

出口点でエラーが検出されると、CICS は、エラーが発生した出口プログラム、エラーが発生したプログラム内の場所、関連付けられている出口点の名前を示すメッセージを出します。エラーの検出は保証されるものではありません。エラー発生時の CICS 環境およびエラーの性質に左右されるからです。例えば、検出メカニズムがオフになっていたために、CICS がループしているユーザー出口プログラムを認識しない場合もあります。また、出口の XPCABND、XPCTA、XSRAB のいずれかで異常終了が発生すると、CICS が異常終了する可能性があります。これは、異常終了処理中に異常終了が発生すると CICS が終了するためです。

一部の出口点 (XTSREQ、XTSREQC、XICREQ、XICREQC、XTDEREQ、XTDEREQC など) で呼び出される出口プログラムは、再帰コマンド (出口点 XTSREQ での TS コマンドなど) を発行するとループに入ります。

ことがあります。最も影響を受けやすい出口には、このようなループを防ぐために使用できる再帰カウント・パラメーター UEPRECUR が用意されています。

## 重要

ユーザー出口プログラムをコーディングするときには、コードがトランザクションとしてではなく CICS コードの拡張として実行されること、また、エラーが致命的な結果をもたらす可能性があることを忘れないでください。

## 出口プログラムの定義、有効化、無効化

出口プログラムを作成したら、CICS にそれを定義する必要があります。出口プログラムを有効にするには、出口点または **EXEC CICS ENABLE** コマンドを使用します。出口プログラムを無効にするには、**EXEC CICS DISABLE** コマンドを使用します。

## 手順

1. **CEDA DEFINE PROGRAM** コマンドを使用し、RELOAD(NO) を指定して出口プログラムを定義します。
2. 出口プログラムをインストールします。
3. 次のいずれかの方法で出口プログラムを有効にします。
  - 出口プログラムがユーザー・ログ・レコード・リカバリー・プログラムまたはファイル制御リカバリー制御プログラムの出口点を使用している場合、**TBEXITS** システム初期設定パラメーターを使用できます。
  - そうでない場合は **EXEC CICS ENABLE** コマンドを使用してユーザー出口プログラムを有効にします。

インストールの前にグローバル・ユーザー出口プログラムを有効にした場合、システム初期設定パラメーターで LPA=YES が指定されていると、CICS は LPA をスキャンしてプログラムを検索します。メッセージ DFHLD0109I が出た場合は、CICS が LPA でプログラムを見つけることができず、DFHRPL または動的 LIBRARY 内のバージョンを使用することを意味します。

4. 出口プログラムの使用を終了したら、**EXEC CICS DISABLE** コマンドを使用してプログラムを無効にできます。

## 例

グローバル・ユーザー出口プログラムを有効または無効にする例については、[グローバル・ユーザー出口ファウンデーション・サンプル](#)および[特定の出口プログラムのサンプル](#)を参照してください。

## アクティブなグローバル・ユーザー出口の表示

Web ユーザー・インターフェースを使用して、CICS 領域で実行中のすべてのグローバル・ユーザー出口プログラムを表示できます。

### 始める前に

このタスクを実行するには、CICSPlex® SM をインストールして構成しておく必要があります。

### このタスクについて

#### 手順

1. Web ユーザー・インターフェースにログオンします。
2. 「**CICS 操作**」 > 「**出口操作ビュー**」 > 「**グローバル・ユーザー出口**」を選択します。  
グローバル・ユーザー出口ビューに、CICS 領域でグローバル・ユーザー出口点を使用しているすべてのプログラムの詳細が表示されます。
3. プログラムの定義の詳細を表示したいグローバル・ユーザー出口プログラムの名前を選択します。

## 次のタスク

このビューを使用して、グローバル・ユーザー出口プログラムの有効化や無効化など、追加のタスクを実行できます。

## 複数の出口プログラムを単一の出口で呼び出す

複数の出口プログラムを単一のグローバル・ユーザー出口点から呼び出すことができます。

それらのプログラムは個々に独立して動作できますが、次の点に注意する必要があります。

- **EXEC CICS ENABLE** コマンドの **START** オプションで実行可能にした出口プログラムのみが、出口で呼び出されます。複数の出口プログラムを 1 つの出口点で開始した場合、プログラムの呼び出し順は、それらのプログラムをアクティブにした順序になります (つまり、**EXEC CICS ENABLE** コマンドで出口点と関連付けた順序です)。複数のプログラムで同じデータ域を使用する場合は、呼び出し順を考慮する必要があります。例えば、端末制御出力出口で、前の出口プログラムの動作に応じて、後の出口プログラムによるメッセージの操作を変える場合があります。
- 戻りコードの管理は単一プログラムの場合よりも複雑です。各出口プログラムは戻りコードを通常どおりレジスター 15 に設定します。単一出口点から呼び出された 2 番目以降のプログラムは、前のプログラムによって設定された戻りコード値 (「現在の戻りコード」) に、DFHUEPAR のパラメーター **UEPCRC**A を使用してアクセスできます。

単一出口で呼び出される複数のユーザー出口プログラムが設定する戻りコード値については、次のルールが適用されます。

- ユーザー出口プログラムが前のプログラムの戻りコード値 (UEPCRC A でアドレス指定された値) と同じコード値を渡した場合、CICS はその値で動作します。
- ユーザー出口プログラムが前のプログラムの戻りコード値 (UEPCRC A でアドレス指定された値) とは異なるコード値を渡した場合、CICS は両方の値を無視し、「現在の戻りコード」をデフォルト値 (通常は UERCNORM) にリセットしてから、その出口点の次の出口プログラムを呼び出します。
- ユーザー出口プログラムがレジスター 15 に適格な値を設定し、それに一致するように「現在の値」 (UEPCRC A でアドレス指定された値) フィールドを変更した場合、新しい値が採用され、次のプログラム (ある場合) に渡されるか、または呼び出し元の CICS モジュールまたはドメインに戻されます。

次のコード・スニペットは、前の出口プログラムから返された「現在の戻りコード」とは異なる戻りコードを出口プログラムで設定し、CICS をその新しいコードで動作させる方法を示しています。

	LA	R15,UECTDOK	Set the contents of reg 15 to a value of 4
	L	R6,UEPCRC A	Set reg 6 to the address of the half word containing the current return code
*			
	STH	R15,0(,6)	Store the new return code at the location of the current return code.
*			
	.	.	.

## 単一の出口プログラムを複数の出口で呼び出す

単一の出口プログラムを複数の出口点から呼び出すには、各出口点に対して **ENABLE** コマンドを発行する必要があります。

**ENABLE** コマンドの発行方法に関するプログラミング情報については、[出口関連のコマンド](#)を参照してください。GALENGTH または GAENTRYNAME は、最初の **ENABLE** コマンドでのみ指定するように注意してください。そうしないと、「INVEXITREQ」が返される場合があります。

CICS サービスの使用に適用される制限に留意してください。これらの制限は、出口プログラムではなく出口点自体によって決まるからです。ある出口点から発行できるコマンドを別の出口点から発行した場合、問題が発生することがあります。

グローバル作業域は、出口点ではなく出口プログラムに関連付けられます。つまり、同じ出口プログラムを呼び出すすべての出口点で同じグローバル作業域が使用されることになります。

## タスク・トークン UEPTSTOK の使用

UEPTSTOK は、いくつかのユーザー出口点で渡される出口固有のパラメーターです。このパラメーターは、同じタスクで連続してインターバル制御要求を出す場合に要求間で情報を渡すために使用できる、4 バ

イト域のアドレスを指定します。例えば、XFCREQ 出口を連続して呼び出し、それらの呼び出しの間で情報を渡す必要がある場合は、UEPTSTOK を使用して実現できます。

UEPTSTOK は、EISEXITT フィールド (EIS のタスク存続期間トークン) をアドレス指定するために設定されます。タスクのために出口で使用するストレージ域をアドレス指定するために、このフィールドを設定する必要があります。これにより、そのタスク存続期間ストレージを、タスクの存続期間中に別の出口プログラムで共用することができます。

各出口プログラムが適切な方法で UEPTSTOK を使用しない限り、これを使用して出口固有のデータをアドレス指定すると、結果が予測不能になります。以下の方法により、このトークンを複数の出口プログラムで共用できます。

- UEPTSTOK のトークンが、アドレス・チェーンの最初です。それ以降のエディションのトークンは、チェーニングされたストレージの部分の前にあります。出口プログラムはストレージを取得し、そのストレージの複数のアドレスを UEPTSTOK からチェーニングします。
- ストレージの各部分のバイト 0 から 3 は、チェーニングされたストレージの次の部分に対するポインターです。バイト 4 から 11 には、ストレージを取得した出口プログラムの名前 (またはストレージの所有者を示す他の定数) が入ります。

UEPTSTOK を使用する出口プログラムのためにこのようなメカニズムを実装して、同じタスクから起動された複数の出口プログラムの間でフィールドを共用することができます。

**注:** 通常、出口プログラムは、タスクから初めて起動されたときに、その出口プログラム用に現在チェーニングされているストレージの部分が存在しないことを検出すると、ストレージの一部を取得し、UEPTSTOK でアドレス指定されているチェーンにそのストレージの部分を追加します。ストレージの一部をチェーンに追加するときには、チェーンの先頭に追加しても最後に追加してもかまいません。重要なことは、新しく追加されたストレージ・アドレスと出口 ID を反映するようにチェーンを更新することです。

## タスク関連のユーザー出口プログラム

タスク関連ユーザー出口 (TRUE) を使用して独自のプログラムを作成し、それ以外の方法では CICS システムで使用できないリソース (データベースなど) にアクセスします。

### タスク関連ユーザー出口のメカニズム (アダプター) の概要

タスク関連ユーザー出口 (TRUE: Task-Related User Exit) を使用すると、CICS システムで通常は使用できないデータベースなどのリソースにアクセスする独自のプログラムを作成できます。

このようなリソースのことを、非 CICS リソースと呼びます。この出口をタスク関連出口と呼ぶ理由は、この出口は、呼び出し元のタスクの一部になり、グローバル・ユーザー出口と違って出口点と関連付けられないからです。タスク関連ユーザー出口の使用は必須ではありませんが、使用すれば、独自の要件に従って、CICS システムの機能を拡張してカスタマイズすることができます。

タスク関連ユーザー出口の最も一般的な使用法は、ファイル・マネージャーやデータベース・マネージャーなどの、CICS の外部のリソース・マネージャーと通信することです。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処理する CICS インターフェース・モジュールを、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インターフェースと呼びます。

タスク関連ユーザー出口メカニズムは、非 CICS リソースにアクセスしなければならないアプリケーション・プログラムとそのリソースのマネージャーの間の接続を可能にするので、アダプター とも呼ばれます。[13 ページの図 1](#) は、アダプターの概念を表しています。

アダプターは 3 つ以上のローカルに作成されたプログラムから構成されています。「スタブ」プログラム、タスク関連ユーザー出口プログラム、1 つ以上の管理ルーチンまたはプログラムです。

スタブ・プログラムは、呼び出し元アプリケーション・プログラムから発行された要求 (外部データベース・マネージャー上で保持されているデータへのアクセス要求など) を代行受信します。スタブを使用して、ローカルに定義された高水準言語コマンドをタスク関連ユーザー出口のマクロ呼び出し DFHRMCAL に解決できます。その結果 CICS がタスク関連ユーザー出口プログラムに制御権を渡します。

タスク関連ユーザー出口プログラムは、非 CICS リソースにアクセスするコマンドを、リソース・マネージャーが受け入れられる形式に変換します。このプログラムはアセンブラ言語で作成する必要があり、16 MB 境界より下か、16 MB 境界より上で 2 GB 境界より下に常駐できます。アドレッシング・モードと

常駐モードの詳細については、33 ページの『アドレッシング・モードの影響』を参照してください。このプログラムでアクセス・レジスターのコンテンツを変更してはいけません。このプログラムは、例えば外部データベースのデータの読み取りを求める要求のような特定のアプリケーション・プログラム要求に対する応答として実行されます。その場合には、必要なレコードを取得するための検索指数などの、アプリケーション・データをプログラムに渡すことができます。リソース・マネージャーからの応答は、タスク関連ユーザー出口プログラムが呼び出し元プログラムに戻します。

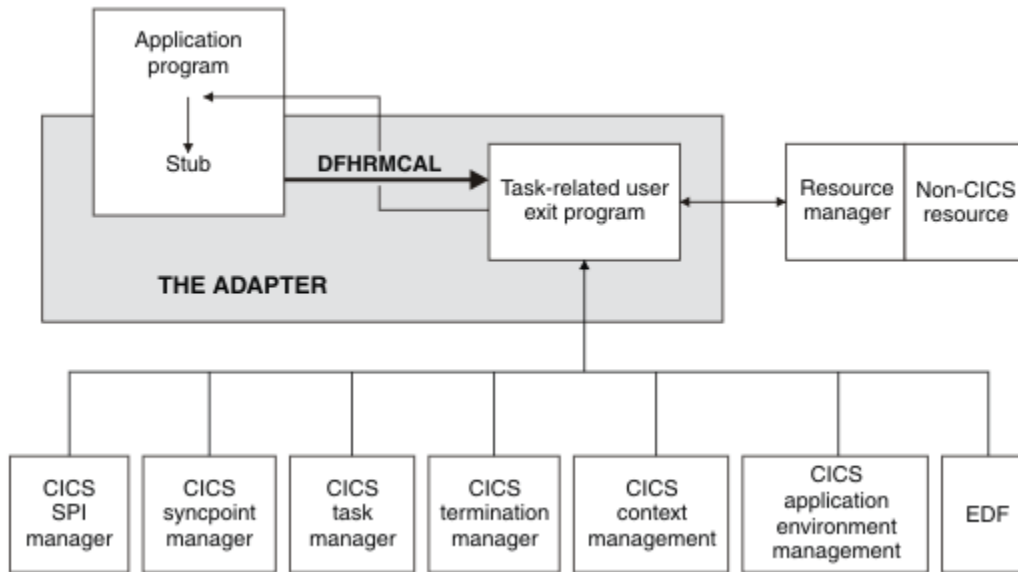


図 1. アダプターの概念

タスク関連ユーザー出口プログラムは、タスク関連ユーザー出口を処理する CICS 管理モジュールからパラメーター・リスト (DFHUEPAR) を受け取ります。このパラメーター・リストにより、タスク関連ユーザー出口は、専用の作業域のアドレスやサイズなどの情報にアクセスできます。

タスク関連ユーザー出口プログラムを呼び出せるものを次に示します。

- アプリケーション・プログラム
- CICS SPI マネージャー
- CICS 同期点マネージャー
- CICS タスク・マネージャー
- CICS 終了マネージャー
- CICS コンテキスト管理
- CICS アプリケーション環境管理
- 実行診断機能 (EDF)

パラメーター・リストにより、これらの多様な呼び出し元を区別し、呼び出し元のレジスターが含まれているレジスター保管域にアクセスすることができます。

AMODE 64 のタスク関連ユーザー出口はサポートされていません。

管理ルーチンには、EXEC CICS ENABLE コマンドと DISABLE コマンドが含まれています。これらのコマンドを使用して、タスク関連ユーザー出口プログラムをインストールしたり削除したりします。管理ルーチンには、出口プログラムの作業域の 1 つに関する情報を取得するコマンド (EXEC CICS EXTRACT EXIT コマンド) や、システム障害後に CICS と非 CICS のリソース・マネージャー間の不整合を解決するコマンド (EXEC CICS RESYNC コマンド) も含まれている場合があります。プログラミングの情報については、[ENABLE PROGRAM コマンド](#)、[DISABLE PROGRAM コマンド](#)、[EXTRACT EXIT](#)、[RESYNC ENTRYNAME](#) を参照してください。



## スタブ・プログラム

スタブ・プログラムは、アプリケーション・プログラマーが非 CICS リソース・マネージャーの構造を認識しなくても済むようにします。このプログラムはアセンブラ言語で作成されます。アセンブリー後に、スタブを使用する各アプリケーション・プログラムに、スタブをリンク・エディットします。

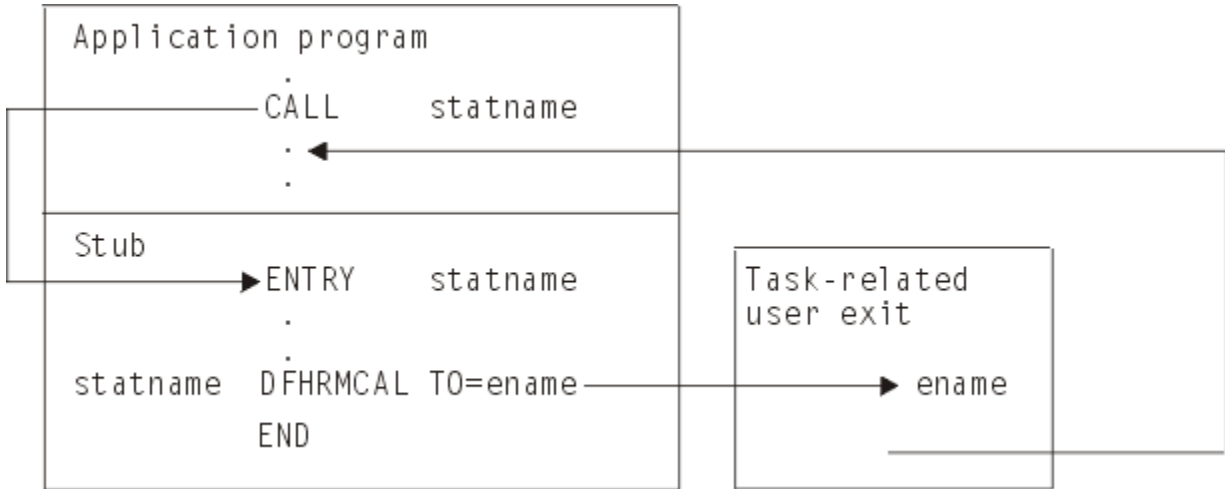


図 2. スタブの概念

### statname

外部参照できるラベル。statname は、アセンブラ言語の ENTRY ステートメントの要件に準拠していなければならない、一般的には V タイプ・アドレス定数か高水準言語 CALL のターゲットを解決します。1 つのスタブに複数のこの種のラベルを含めることができます。

### ename

リソース・マネージャー要求を処理するタスク関連ユーザー出口プログラムの入り口名 (**EXEC CICS ENABLE** コマンドで指定)。

プログラマーが非 CICS リソースにアクセスするために使用する高水準言語コマンドを定義できます。変換プログラムを使用して、ローカルに定義された高水準言語コマンドを、必要なスタブ・プログラムの入り口点に対する従来型の CALL に変換しなければなりません。あるいは、14 ページの図 2 に示すように、アプリケーション・プログラムからスタブの入り口点を指定した CALL を発行することもできます。例えば、非 CICS リソースからレコードを読み取るために、アプリケーション・プログラムで以下の COBOL ステートメントを使用することができます。

```
CALL 'XYZ' USING PARM1 PARM2...
```

XYZ はスタブ・プログラム内の入り口点 (statname) です。スタブは、このコマンドをタスク関連ユーザー出口プログラム (TO= オペランドで指定) に対するマクロ呼び出し (DFHRMCAL) に変換します。タスク関連ユーザー出口プログラムからの戻りは、スタブ・プログラムではなく呼び出し元アプリケーション・プログラムに返されます。

アプリケーションでパラメーターを使用して、リソース・マネージャーが呼び出されたかどうかを判別できます。例えば、あるパラメーターをアプリケーションでゼロに設定し、リソース・マネージャーでそのパラメーターをゼロ以外に設定する場合は、戻されたパラメーター値から、リソース・マネージャーが呼び出されたかどうかを判別できます。

### 注:

- 使用できる DFHRMCAL マクロのオペランドは TO、RTNABND、SUPPEDF だけです。その他のオペランドはすべて CICS 内部使用専用です。
- AMODE(64) のアプリケーション・プログラムで DFHRMCAL マクロを呼び出すことはできません。

## アプリケーション・プログラムへの制御の戻り

DFHRMCAL マクロで RTNABND=YES を指定すると、タスク関連のユーザー出口が (例えば、それが使用可能にされていないか、開始されていないために) 使用できない場合に制御がアプリケーション・プログラムに戻ります。

アセンブラ言語のアプリケーション・プログラムの場合、レジスター 15 の負の値は、出口が使用できないために制御が戻ったことをアプリケーション・プログラムにシグナル通知します。タスク関連のユーザー出口プログラムは、レジスター 15 で正の値 (ゼロを含む) を使用して、リソース・マネージャーの応答コードをアプリケーション・プログラムに渡すことができます。

RTNABND=YES を指定せず、タスク関連のユーザー出口が使用不可の場合には、アプリケーション・プログラムは異常終了コード 'AEY9' で異常終了します。

## タスク関連のユーザー出口と EDF

EDF によってモニターされるアプリケーションから非 CICS リソース・マネージャーを呼び出すためにタスク関連ユーザー出口 (TRUE) が呼び出される場合、EDF のデフォルトの処置は、DFHRMCAL マクロによって渡されるパラメーター・リストによってアドレッシングされるパラメーターを表示することです。TRUE を有効にする EXEC CICS ENABLE コマンドで FORMATEDF を指定することによって、パラメーター・リストをより意味のある表示に変換することができます。

TRUE は、リソース・マネージャーへの呼び出しを満たすために、呼び出しの前後に何度も呼び出され、EDF で表示されるデータを形式設定し、EDF 画面上のデータに対してユーザーが行った変更を取り扱います。

EDF のために画面を形式設定する方法については、29 ページの『CICS EDF ビルド・パラメーター』および 45 ページの『タスク関連ユーザー出口プログラムと EDF の使用』を参照してください。

タスク関連ユーザー出口プログラムに EXEC CICS コマンドが含まれている場合、EDF は TRUE 自体のデバッグする際に役立ちます。EDF で TRUE からのコマンドを表示する場合、TRUE プログラムが変換されるときに EDF オプションを指定する必要があります。このようにすると、リソース・マネージャーの呼び出しのために、CICS コマンドの標準 EDF 画面が「実行開始直前 (About to Execute)」画面から「コマンドの実行完了 (Command Execution Complete)」画面までの間、表示されます。ただし、EDF は基本的にはアプリケーション・デバッグ・ツールであり、TRUE 内の CICS コマンドは通常はアプリケーション・プログラマーには関係ないので、TRUE プログラムは通常、「NOEDF」オプションを使用して変換されます。その場合、TRUE 内の CICS コマンドの画面は表示されません。

注: DFHRMCAL マクロで SUPPEDF=YES を指定する場合、DFHRMCAL による TRUE の呼び出しに関連した「実行開始直前 (About to Execute)」画面と「コマンドの実行完了 (Command Execution Complete)」画面は表示されません。つまり、DFHRMCAL は EDF に対して「不可視」になります。SUPPEDF=YES の指定は、EDF で TRUE 内の EXEC CICS コマンドを表示するかどうかに関する決定に影響を及ぼしませんが、TRUE に渡されるパラメーターは表示されなくなります。

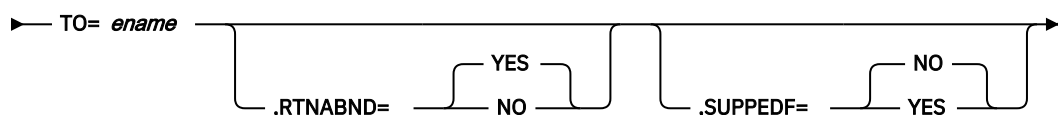
## DFHRMCAL マクロ

DFHRMCAL マクロは、リソース・マネージャー要求を処理するタスク関連ユーザー出口プログラム (TRUE) に制御権を渡します。

### 構文

#### DFHRMCAL

➡ DFHRMCAL ➡



このマクロの他のパラメーターは、CICS の内部使用専用です。

AMODE(64) のアプリケーション・プログラムでこのマクロを呼び出すことはできません。

## パラメーター

### TO=ename

ename は、リソース・マネージャー要求を処理するタスク関連ユーザー出口プログラム (TRUE) の入り口名です。

### RTNABND={YES|NO}

有効にされていない、開始されていないなどの理由で、タスク関連ユーザー出口プログラムが使用不可の場合に、制御権をアプリケーション・プログラムに戻すかどうか。有効な値は、以下のとおりです。

#### YES

タスク関連ユーザー出口プログラムが使用不可の場合に、制御権をアプリケーション・プログラムに戻します。この値はデフォルトです。

#### NO

タスク関連ユーザー出口プログラムが使用不可の場合に、制御権をアプリケーション・プログラムに戻しません。

アセンブラ言語のアプリケーション・プログラムの場合は、レジスター 15 の負の値により、出口が使用不可であるために制御権が戻されたことをアプリケーション・プログラムに通知します。タスク関連のユーザー出口プログラムは、レジスター 15 で正の値 (ゼロを含む) を使用して、リソース・マネージャーの応答コードをアプリケーション・プログラムに渡すことができます。

RTNABND=NO を指定した場合に、タスク関連ユーザー出口プログラムが使用不可であれば、アプリケーション・プログラムは異常終了コード AEY9 で異常終了します。

### SUPPEDF={YES|NO}

DFHRMCAL による TRUE の呼び出しに関係する EDF 画面を抑制するかどうか。EDF のモニター対象アプリケーションからの非 CICS リソース・マネージャーに対する呼び出しのために TRUE が呼び出された場合、EDF は「**実行開始直前 (About to Execute)**」画面と「**コマンドの実行完了 (Command Execution Complete)**」画面を表示できます。これらの画面には、DFHRMCAL マクロが渡すパラメーター・リストでアドレス指定されているパラメーターが表示されます。有効な値は、以下のとおりです。

#### YES

DFHRMCAL による TRUE の呼び出しに関係する EDF 画面を抑制します。

#### NO

DFHRMCAL による TRUE の呼び出しに関係する EDF 画面を表示します。この値はデフォルトです。

## タスク関連ユーザー出口プログラムの作成

タスク関連ユーザー出口プログラムの主な機能は、呼び出し側プログラムのパラメーターを非 CICS リソース・マネージャーで受け入れ可能な形式に変換してから、リソース・マネージャーに制御を渡すことです。

呼び出し側プログラムのパラメーターについては [24 ページの『呼び出し元のパラメーター・リスト』](#) に説明があります。

このセクションでは、ユーザー出口パラメーター・リスト、スケジュール・フラグ・ワード (これは、CICS 管理サービスによって呼び出す必要があることを登録するために出口プログラムによって使用されます)、およびタスク関連ユーザー出口プログラムでの登録処理について説明します。また、このセクションでは、CICS 同期点マネージャーと CICS タスク・マネージャーの使用についても取り上げます。さらに、CICS オープン・トランザクション環境 (OTE) によって提供される TCB の使用を計画している場合に考慮すべきいくつかの要素についても説明します。

### TRUE および使用される TCB のタイプ

タスク関連ユーザー出口 (TRUE) は、その TRUE のプログラムの属性に応じて、メイン CICS QR TCB またはオープン TCB で実行できます。

準再入可能 TRUE は、QUASIRENT オプションを指定して有効にします。準再入可能 TRUE は、必ず、オープン TCB ではなく QR TCB で起動されます。外部リソース・マネージャーを起動する準再入可能 TRUE は、独自のサブタスク TCB のセットを管理する必要があります。通常、外部リソース・マネージャーにアクセスするためにサブタスク TCB が配置されると、TRUE の QR TCB で実行されている CICS タスクは、そのサブタスクの処理が完了するまで CICS ディスパッチャー待機状態になります。CICS ディスパッチャー待機状態により、CICS ディスパッチャーは、その間、別の CICS タスクを QR TCB にディスパッチするこ



とができます。このようなアーキテクチャーになっている理由は、呼び出し元が QR TCB で外部リソース・マネージャーを直接起動することができないからです。これは、外部リソース・マネージャーからオペレーティング・システム待機が発行されると、QR TCB および CICS 全体が停止するためです。

THREADSAFE オプションを指定して有効にした TRUE は、TCB の種類にかかわらず、TRUE の起動時にタスクを実行している TCB で実行されます。これは、QR TCB である場合もオープン TCB である場合もあります。

THREADSAFE および OPENAPI の両方のオプション、または REQUIRED および OPENAPI の両方のオプションを指定して有効にした TRUE は、必ず L8 オープン TCB で実行されます。REQUIRED オプションを指定し、OPENAPI オプションは指定せずに有効にした TRUE は、必ず、キー 8 オープン TCB で実行されます。このキー 8 オープン TCB は、呼び出し元アプリケーションの言語に応じて、L8、T8、または X8 オープン TCB である場合があります。オープン TCB のモードについては、[オープン TCB の管理](#)を参照してください。

### オープン TCB で実行されるタスク関連ユーザー出口 (TRUE) の責任

TRUE をオープン TCB で実行すると、QR TCB による制約から解放されるため、独自のサブタスク TCB セットを作成して管理する必要がなくなります。外部リソース・マネージャーから発行されたオペレーティング・システム待機によってオープン TCB が停止した場合、CICS は QR TCB および他のオープン TCB で処理を続けます。それでもやはり、オープン TCB で実行される TRUE には、CICS システム全体と、使用するオープン TCB の将来のユーザーに対する責任があります。

### L8 オープン TCB で実行される TRUE の責任

L8 TCB は、それを割り振られた CICS タスクで専用されますが、CICS タスクが完了すると、クリーンな状態の L8 TCB は、ディスパッチャーが管理する L8 モード TCB のプールに返されます。ここで言う「クリーンでない TCB」とは、CICS で検出されない[スレッド・セーフの制限](#)に TRUE が違反したことを意味するのではなく、TRUE において L8 モードの TCB を使用するタスクに処理不能の異常終了が発生したことを意味します。

L8 TCB は特定の TRUE で専用されるのではなく、CICS タスクから呼び出された、L8 TCB を必要とするすべての TRUE で使用されます。L8 TCB は、そのタスクから実行されたスレッド・セーフなアプリケーション・コードでも使用されます。

### T8 および X8 オープン TCB で実行される TRUE の責任

OSGi JVM で実行される XPLINK プログラムおよび Java™ プログラムの場合は、それぞれ、X8 および T8 TCB がそのリンク・レベルの CICS タスクで専用されます。より高いリンク・レベルに制御権が戻ると、TCB は解放され、別の CICS タスクで使用できるようになります。

Liberty JVM で実行される Java プログラムの場合は、T8 TCB が、L8 TCB と同じように、CICS タスクでそのタスクの存続期間中、専用されます。T8 TCB がクリーンなままである場合、それらは後続のタスクで使用可能です。そして、呼び出し元タスクから実行されたアプリケーション・コードも、TRUE と同じ TCB で実行されます。

### スレッド・セーフに関する制約事項

オープン TCB で実行される TRUE では、以下に対して、問題の原因となるような方法でオープン TCB 環境の実行を処理してはなりません。

- 同一のタスクにより呼び出されるオープン TRUE で実行される他の TRUE
- 同一のタスクにより呼び出される OPENAPI プログラム
- オープン TCB で実行する可能性のあるアプリケーション・プログラム論理
- オープン TCB を使用する可能性のある将来のタスク
- CICS 管理コード。

特に以下の点に注意してください。

- CICS サービスを呼び出す場合、または CICS に戻る場合、オープン TCB で実行される TRUE では、TRUE への入り口での状態に MVS プログラミング環境を復元する必要があります。これには、仮想記憶間モー

ド、ASC モード、要求ブロック (RB) レベル、リンケージ・スタック・レベル、TCB ディスパッチング優先順位などとともに、追加されたすべての ESTAE の取り消しが含まれます。

- CICS タスクの終了時に、オープン TCB で実行される TRUE は、別の CICS トランザクションによる再利用に適合した状態でそのオープン TCB から抜けるようにする必要があります。特に、タスクの終了のために特別に獲得した非 CICS リソースは、必ずすべて解放してください。以下のようなリソースがその対象になります。
  - 動的に割り振られたデータ・セット
  - オープン ACB または DCB
  - STIMERM 要求
  - MVS 管理のストレージ
  - ENQ 要求
  - 接続されたサブタスク
  - ロードされたモジュール
  - 所有するデータ・スペース
  - 追加されたアクセス・リスト項目
  - 名前/トークンのペア
  - 固定ページ
  - セキュリティー設定 (TCBSENV をゼロに設定する必要があります)
- オープン TCB で実行される TRUE では、CICS 全体のオペレーションに影響する以下の MVS システム・サービスを使用してはなりません。
  - CHKPT
  - ESPIE
  - QEDIT
  - SPIE
  - STIMER
  - TTIMER
  - XCTL / XCTLX
  - すべての TSO/E サービス。
- オープン TCB で実行される TRUE では、L8 モード TCB において、MVS 言語環境の各サービスを使用している言語環境プログラムを起動してはなりません。これは、L8 モード TCB が、CICS の各サービスを使用した言語環境用に初期化されているためです。

## オープン TCB で実行されるタスク関連ユーザー出口の呼び出し

### このタスクについて

タスク関連ユーザー出口 (TRUE) を、常にオープン TCB で実行されることを示すオプションを指定して有効にした場合、CICS は、呼び出しのタイプに応じて以下の規則を使用して、TRUE を呼び出す TCB を決定します。

#### アプリケーション・プログラム呼び出し (API) – UERTAPPL

この呼び出しの場合、常に L8 で実行することを有効にした TRUE は、L8 TCB で呼び出されます。いずれかのキー 8 TCB で実行することを有効にした TRUE は、L8、T8、または X8 で起動されます。

#### CICS 同期点マネージャー呼び出し – UERTSYNC

この呼び出しの場合、常に L8 で実行することを有効にした TRUE は、L8 TCB で呼び出されます。いずれかのキー 8 TCB で実行することを有効にした TRUE は、L8、T8、または X8 で起動されます。

#### CICS タスク・マネージャー呼び出し – UERTTASK

この呼び出しの場合、CICS が TRUE を呼び出す TCB は、さらにタスク・マネージャー呼び出しのタイプに応じて以下のように決定されます。

### UERTSOTR — タスクの開始

この呼び出しの場合、Liberty スレッドとして実行されるタスクは、CICS が Liberty ThreadPool に提供した T8 TCB で TRUE を起動します。他のすべての環境では、パフォーマンス上の理由でオープン TCB は使用されず、TRUE は常に QR TCB で起動されます。

### UERTEOTR — タスクの終了

この呼び出しの場合、常に L8 で実行することを有効にした TRUE は、L8 TCB で呼び出されます。いずれかのキー 8 TCB で実行することを有効にした TRUE は、L8、T8、または X8 で起動されます。

### EDF 呼び出し — UERTFEDF

この呼び出しの場合、常に L8 で実行することを有効にした TRUE は、L8 TCB で呼び出されます。いずれかのキー 8 TCB で実行することを有効にした TRUE は、L8、T8、または X8 で起動されます。

### CICS SPI 呼び出し — UERTSPI

この呼び出しでは、パフォーマンス上の理由で、CICS が常に TRUE をスレッド・セーフな TRUE として起動するので、TRUE は、タスクを現在実行している TCB で呼び出されます。

CONNECTST または QUALIFIER オプションが指定された EXEC CICS INQUIRE EXITPROGRAM コマンドを満たすための SPI 機能は単純であり、特定の TCB で起動する必要はありません。

### CICS 終了呼び出し — UERTCTER

この呼び出しの場合、オープン TCB は使用されず、CICS は常に QR TCB で TRUE を呼び出します。

**注：**オープン TCB で起動されるタイプの呼び出しでも、オープン TCB で非同期の異常終了が発生し、それ以降の使用が不可能になり、以下の結果になる可能性があります。

- CICS が TRUE に対する API 呼び出しでオープン TCB に切り替えられない場合、CICS はトランザクションを異常終了させます。
- CICS が同期点またはタスク終了呼び出しでオープン TCB に切り替えられない場合、CICS は代わりに QR TCB で TRUE を起動します。

TRUE を呼び出す TCB モードは、**UEPTIND** シンボル名が指す 3 バイトのフィールドのアドレス・パラメーターの第 2 バイトと第 3 バイトに示されます。詳しくは、[ユーザー出口パラメーター・リスト](#)を参照してください。

### ユーザー出口パラメーター・リスト

タスク関連ユーザー出口プログラムを呼び出すときに、タスク関連ユーザー出口を処理する CICS 管理モジュールは、パラメーター・リストを出口プログラムに提供します。このパラメーター・リストのアドレスは、レジスター 1 に入れて渡されます。

リストには、以下の情報が含まれています。

- 呼び出し元の ID
- タスク関連ユーザー出口プログラムが使用できる作業域のアドレスおよびサイズ
- 呼び出し元のレジスター保管域のアドレス
- この呼び出し中にタスク関連ユーザー出口プログラムで使用するための EXEC インターフェース・ブロック (EIB) のアドレス
- 現行リカバリー単位の ID のアドレス
- スケジュール・フラグ・ワードのアドレス
- カーネル・スタック・エントリーのアドレス
- APPC 作業単位 (UOW) ID のアドレス
- ユーザー・セキュリティ・ブロック・フラグのアドレス
- ユーザー・セキュリティ・ブロックのアドレス
- リソース・マネージャー修飾子名のアドレス
- リソース・マネージャーの単一更新および読み取り専用の標識バイトのアドレス
- 呼び出し元の AMODE 標識バイトのアドレス
- アプリケーションの DATALOC および TASKDATAKEY の標識バイトのアドレス

- パフォーマンス・ブロック・トークンのアドレス
- トレース・フラグのアドレス

Liberty 環境では、後で使用するためにユーザー ID をキャッシュに入れしないでください。タスク TRUE の開始時に渡されるユーザー ID が、そのタスクに対して最終的に設定されたユーザー ID を表さなくなる可能性があります。後続の TRUE によりユーザー ID が提供されます。

出口プログラムがこのパラメーター・リストにアクセスできるように、以下のマクロを含める必要があります。

```
DFHUEEXIT TYPE=RM
```

DFHUEEXIT TYPE=RM マクロを含めると、アセンブラーがストレージ定義 (DSECT) DFHUEPAR、DFHUERTR、DFHUECON を作成します。タスク関連ユーザー出口で EDF の画面のフォーマットを調整できるようにするには、以下のマクロを含める必要があります。

```
DFHUEEXIT TYPE=RM,DSECT=EDF
```

これにより、アセンブラーが UEPEDFRM DSECT を作成します。UEPEDFRM DSECT については、[29 ページの『CICS EDF ビルド・パラメーター』](#)で説明しています。すべてのユーザー出口パラメーター・リストについて、[30 ページの『タスク関連ユーザー出口のパラメーター・リストの要約』](#)にまとめています。

以下の情報は、これらの定義の形式と目的を示しています。

## DFHUEPAR

DFHUEPAR により、アドレス・パラメーターに対する次のシンボル名が渡されます。

### UEPEXN

タスク関連ユーザー出口プログラムに、そのプログラムが呼び出されている理由を伝える機能定義のアドレス。詳細については、[23 ページの『DFHUERTR \(機能定義\)』](#)を参照してください。

### UEPGAA

EXEC CICS ENABLE コマンドで要求されたグローバル作業域のアドレス。グローバル作業域については、[36 ページの『作業域』](#)で説明しています。タスク関連ユーザー出口プログラムを有効にすると、CICS がこの作業域を X'00' に初期設定します。

### UEPGAL

グローバル作業域の長さ (バイナリー値) を含むハーフワードのアドレス。

### UEPTCA

このフィールドは、歴史的な理由で保持されています。出口プログラムで参照しないでください。

### UEPCSA

このフィールドは、歴史的な理由で保持されています。出口プログラムで参照しないでください。

### UEPHMSA

呼び出し元のレジスター保管域 (RSA) のアドレス。これは 18 ワードの保管域であり、レジスター 14 から 12 までの内容が、4 番目以降のワードに保管されます。5 番目のワードは呼び出し元プログラムのレジスター 15 を表しますが、CICS が、タスク関連ユーザー出口プログラムを起動する前にクリアします。5 番目のワードがクリアされるのは、リソース・マネージャーからの応答コードを呼び出し元プログラムに伝えるときに使用するためです。この理由のため、レジスター 15 を使用して、タスク関連ユーザー出口プログラムにデータを送信することはできません。保管域の 7 番目のワードには、呼び出し元のレジスター 1 が含まれます。レジスター 1 は、呼び出し元のパラメーター・リストのアドレスを指します。要約については、[30 ページの『タスク関連ユーザー出口のパラメーター・リストの要約』](#)を参照してください。呼び出し元がアプリケーション・プログラムである場合、レジスター 1 の内容は、アダプターの言語インターフェースのリンケージ規則によって決まります。

### UEPTAA

EXEC CICS ENABLE コマンドで要求されたローカル作業域のアドレス。ローカル作業域については、[36 ページの『作業域』](#)で説明しています。CICS は、初めて作業域を取得するとき、つまり、タスクが初めてタスク関連ユーザー出口プログラムを起動するときに、作業域をすべて X'00' に初期設定します。

### UEPTAL

ローカル作業域のバイナリーの長さを含むハーフワードのアドレス。

## UEPEIB

タスク関連ユーザー出口プログラムのために CICS が作成した EXEC インターフェース・ブロック (EIB) のアドレス。EIB は呼び出しの間のみ存在し、タスク関連ユーザー出口プログラムがコマンド・レベル・インターフェースを使用して CICS サービスを要求できるようにします。この EIB は、呼び出し元プログラムの EIB と同じものではありません。このため、呼び出し元プログラムのレジスター保管域 (RSA) のアドレスを提供する UEPHMSA を使用せずに呼び出し元プログラムの環境にアクセスすることはできません。

## UEPURID

CICS のリカバリー単位 ID のアドレス。このフィールドには、STCK の命令によって生成された 8 バイトの日時の値が含まれ、現在の作業単位を示します。

## UEPFLAGS

スケジュール・フラグ・ワードのアドレス。これは、CICS 管理プログラムのサービスに対するタスク関連ユーザー出口プログラムのニーズを登録するために、タスク関連ユーザー出口プログラムで使用するフルワードです。詳しくは、[31 ページの『スケジュール・フラグ・ワード』](#)を参照してください。

## UEPRMSTK

カーネル・スタック・エントリーのアドレス。

## UEPUOWDS

APPC 作業単位 (UOW) ID のアドレス。

## UEPSECFLG

ユーザー・セキュリティ・フラグのアドレス。ユーザー・セキュリティ・フラグは、次の値を取ることができる 1 バイトのフィールドです。

### UEPNOSEC (X'80')

この CICS システムではセキュリティはアクティブではありません。

### UEPSEC (X'20')

この CICS システムではセキュリティはアクティブです。この場合にのみ、「ユーザー・セキュリティ・ブロック」のアドレスが設定されます。

## UEPSECBLK

ユーザー・セキュリティ・ブロックをアドレス指定するフルワードのアドレスであり、ACEE です。

## UEPRMQUA

タスク関連ユーザー出口が各 API 要求でリソース・マネージャーの修飾子名を移動できる 8 バイトのフィールドのアドレス。この手法は、同じ出口プログラムを使用してリソース・マネージャーの複数のインスタンスに接続する場合に便利です。修飾子を使用して、出口が現在接続されているリソース・マネージャーのインスタンスを識別します。

1 つの UOW でさまざまな API 要求への応答としてさまざまなリソース・マネージャーの修飾子が返される場合、未確定の情報を記録するために使用される修飾子は、準備呼び出しまたはバックアウト呼び出しの直前に最後の API 要求で返されたリソース・マネージャーの修飾子です。

## UEPCALAM

呼び出し元の AMODE 標識バイトのアドレス。

### X'80'

オリジナルの呼び出し元が AMODE 31 であったことを示します。最上位ビットが設定されておらず、ビット 31 が 0 の場合、呼び出し元は AMODE 24 でした。

## UEPSYNCA

単一更新と読み取り専用の標識バイトのアドレス。出口プログラムで、このフィールド内にフラグを設定して、リソース・マネージャーが単一更新プロトコルを「理解」していることを示し、現在の作業単位 (UOW) の状況を記録することができます。[38 ページの『効率の向上: single-update and read-only プロトコル』](#)を参照してください。

### UEPSUPDR (X'80')

リソース・マネージャーは単一更新プロトコルを理解しています。つまり、適切な状況で、単一フェーズ・コミットを実行するように出口プログラムからリソース・マネージャーに命令することができます。



## UEPREADO (X'40')

リソース・マネージャーは読み取り専用プロトコルを理解しています。また、この作業単位では、これまでのところ読み取り専用モードです (このフラグが設定されていない場合は、UOW にこのリソース・マネージャーに関する更新が含まれていること、または、UOW は読み取り専用であるが、リソース・マネージャーが読み取り専用プロトコルを理解していないことを意味します)。

## UEPTIND

複数の標識を含む 3 バイトのフィールドのアドレス。

最初の標識バイトは、UEPTANY、UEPTCICS、および UEPTUTCB の 3 つのシンボル値の 1 つを取ります。これをテストして以下を決定できます。

- データ・ロケーションを 16 MB 境界の上と下のどちらに置けるか。
- アプリケーションのストレージが CICS キーとユーザー・キーのどちらのストレージ内にあるか。
- TRUE が予期しない TCB から呼び出されたかどうか。

## UEPTANY (X'80')

アプリケーションは、16 MB 境界より上のアドレスを受け入れることができます。シンボル値が UEPTANY でない場合、アプリケーションには、16 MB 境界より下のアドレスを返す必要があります。

## UEPTCICS (X'40')

アプリケーションの作業用ストレージとタスク存続期間ストレージは、CICS キー・ストレージ (TASKDATAKEY=CICS) 内にあります。シンボル値が UEPTCICS でない場合、アプリケーションの作業用ストレージとタスクの存続期間ストレージは、ユーザー・キー・ストレージ (TASKDATAKEY=USER) 内にあります。

## UEPTUTCB (X'20')

予期しない TCB を示します。この値は同期点、またはタスク終了の呼び出しでのみ設定され、タスク関連ユーザー出口で予期された TCB への切り替えに失敗したことを示します。これらの 2 つの場合、タスク関連ユーザー出口は、UEPTUTCB ビットが設定された QR TCB で呼び出されます。それ以外の呼び出しでは、CICS はタスク関連ユーザー出口を起動せずにトランザクションを異常終了させます。

2 番目と 3 番目のバイトには、呼び出し元の TCB モードを示す値が含まれます。この値は、DFHUEPAR では次のように 2 文字コードとシンボル値の両方として表されます。

表 2. DFHUEPAR の TCB 標識		
シンボル値	2 バイトのコード	説明
UEPTQR	QR	準再入可能モード TCB
UEPTRO	RO	リソース所有モード TCB
UEPTCO	CO	並行モード TCB
UEPTSZ	SZ	FEPI モード TCB
UEPTRP	RP	ONC/RPC モード TCB
UEPTFO	FO	ファイル所有モード TCB
UEPTSL	SL	ソケット・リスナー・モード TCB
UEPTSO	SO	ソケット・モード TCB
UEPTS8	S8	セキュア・ソケット層モード TCB
UEPTD2	D2	CICS Db2 ハウスキーピング・モード TCB
UEPTL8	L8	L8 オープン TCB。OPENAPI TRUE、または CICS キーの OPENAPI プログラムに使用。
UEPTL9	L9	L9 オープン TCB。ユーザー・キーの OPENAPI プログラムに使用。

表 2. DFHUEPAR の TCB 標識 (続き)		
シンボル値	2 バイト のコード	説明
UEPTEP	EP	イベント処理 TCB
UEPTTP	TP	Language Environment エンクレーブおよび THRD TCB プールを JVM サーバー用に所有するために使用される TP オープン TCB。
UEPTT8	T8	T8 TCB。マルチスレッド処理を実行するために JVM サーバーで使用。
UEPTX8	X8	X8 オープン TCB。CICS キーの、XPLINK オプションを指定してコンパイルされた C および C++ プログラムに使用。
UEPTX9	X9	X9 オープン TCB。ユーザー・キーの、XPLINK オプションを指定してコンパイルされた C および C++ プログラムに使用。

## UEPPBTOK

z/OS ワークロード・マネージャー (WLM) の Performance Block Token を含む 4 バイトのフィールドのアドレス。出口プログラムは、このトークンを次のために使用できます。

- WLM パフォーマンス・ブロックの情報 (サービス・クラス・トークンの SERVCLS など) にアクセスする。このアクセスを行うためには、出口プログラムは、Performance Block Token を MONTKN の入力パラメーターとして受け渡す WLM EXTRACT マクロの IWMMEXTR を使用する必要があります。
- 処理要求に対するリソース・マネージャーのパフォーマンス・ブロックを元の CICS パフォーマンス・ブロックに関連付ける。例えば、z/OS Workload Manager が CICS タスク全体のパフォーマンスを測定できるように、DBCTL と Db2 は、CICS の代わりに実行した処理を親の CICS タスクに関連付ける必要があります。処理を関連付けるには、WLM IWMMRELA マクロを使用する必要があります。

出口プログラムで、パフォーマンス・ブロックの内容を仮定したり、パフォーマンス・ブロックを変更したりしないでください。そうした場合、結果が予測不能になります。

## UEPTRCE

RMI のトレーシング (RI トレース・コンポーネント) がアクティブであるかどうかを示す、1 バイトのトレース・フラグのアドレス。

### UEPTRLV1 (X'80')

RMI レベル 1 トレースがアクティブです。

### UEPTRLV2 (X'40')

RMI レベル 2 トレースがアクティブです。

タスク関連ユーザー出口でこのフィールドを解決したら、例えば EXEC CICS SET TRACETYPE コマンドなどを発行して RMI トレースのレベルを再設定することができます。

## DFHUERTR (機能定義)

機能定義では、タスク関連ユーザー出口プログラムの呼び出し元を指定します。DSECT には 2 つのシンボル定義 (フィールド) があります。

## UERTFGP

X'00' に設定される 1 バイト。ゼロが設定されていることは、これがタスク関連ユーザー出口の呼び出しであること、したがってパラメーター・リストには UEPTAA、UEPTAL、UEPEIB、UEPURID、および UEPFLAGS のフィールドが含まれることを示しています。

## UERTFID

1 バイト ID。呼び出し元が、アプリケーション・プログラム、CICS SPI マネージャー、CICS 同期点マネージャー、CICS タスク・マネージャー、CICS 終了マネージャー、CICS コンテキスト管理、CICS アプリケーション環境管理、EDF のうちのどれなのかを示す ID です。以下の 7 つの設定のいずれかになります。

## UERTSPI

(X'01') CICS SPI 呼び出し。

#### **UERTAPPL**

(X'02') アプリケーション・プログラム呼び出し。

#### **UERTSYNC**

(X'04') CICS 同期点マネージャー呼び出し。

#### **UERTTASK**

(X'08') CICS タスク・マネージャー呼び出し。

#### **UERTCTER**

(X'0A') CICS 終了呼び出し。

#### **UERTFEDF**

(X'0C') EDF 呼び出し。

#### **UERTSWAE**

(X'0D') スイッチ・アプリケーション環境呼び出し。

#### **UERTFCON**

(X'0E') CICS コンテキスト管理呼び出し。

呼び出し元がどのタイプのプログラムなのかを把握しておくことは非常に重要です。タイプによって、呼び出し側プログラムのパラメーター・リストがタスク関連ユーザー出口プログラムでどう解釈されるかが違って来るからです。

### 呼び出し元のパラメーター・リスト

タスク関連ユーザー出口プログラムに DSECT の DFHUEPAR と DFHUERTR だけでなく DFHUEXIT TYPE=RM も組み込むと、タスク関連ユーザー出口の呼び出し元に関するフィールド定義が追加されます。呼び出し側プログラムのパラメーター・リストは通常、呼び出し側プログラムの RSA の R1 でアドレッシングされます。その RSA は、DFHUEPAR の UEPHMSA フィールドでアドレッシングされます。それらのパラメーターについて、以下のサブトピックで説明します。

#### アプリケーション・プログラムのパラメーター

呼び出し側がアプリケーション・プログラムである場合、そのパラメーター・リストの形式とアドレッシングはローカルに決定されます。

#### CICS SPI パラメーター

**EXEC CICS ENABLE PROGRAM** コマンドの SPI オプションを指定してタスク関連出口プログラムを有効にした場合は、**EXEC CICS INQUIRE EXITPROGRAM** コマンドが使用され、そのコマンドに **CONNECTST** または **QUALIFIER** オプションが指定されていた場合に出口プログラムを開始することができます。

**INQUIRE EXITPROGRAM** コマンドを使用して、出口プログラムがリソース・マネージャー、および入り口名の修飾子に接続されているかどうかを照会します。**INQUIRE EXITPROGRAM** コマンドについては、[INQUIRE EXITPROGRAM](#) を参照してください。

CICS SPI パラメーター・リストには、次の 2 つのエントリーが含まれます。

#### パラメーター 1

1 バイトの出力フィールドのアドレス。タスク関連出口プログラムが外部リソース・マネージャーに接続されているかどうかを示すために使用します。指定できる戻りコードの値は、次のとおりです。

#### **UERTCONN**

(X'80') 出口はそのリソース・マネージャーに接続されています。

#### **UERTNCONN**

(X'40') 出口はそのリソース・マネージャーに接続されていません。

#### パラメーター 2

タスク関連出口プログラムが外部リソース・マネージャーの修飾子を (認識している場合に) 返す 8 文字の出力フィールドのアドレス。修飾子名の詳細については、[20 ページの『DFHUEPAR』](#)の UEPHMQUA パラメーターを参照してください。

#### CICS 同期点マネージャーのパラメーター

CICS 同期点マネージャーのパラメーター・リストには、10 のエントリーが含まれていますが、ほとんどの起動では、パラメーター 1 と 10 にのみ値が含まれています。パラメーター 1 と 10 が指す操作バイトには



フラグが含まれていて、それらが組み合わさり、TRUE が起動された理由を TRUE に伝える 1 つの操作コードを形成します。

パラメーター 2 から 9 に値が含まれるのは、セッションまたはシステムに障害が発生した後に、再同期のために同期点マネージャーが TRUE に対して「無条件のコミット」または「バックアウト」呼び出しを実行した場合のみです。これらの追加のパラメーターは、タスク、タスクを開始したトランザクション、タスクが開始された端末、端末オペレーターの ID、失敗した同期点の日時、および次のトランザクション・コード (タスクに関連付けられたリカバリー単位がこれ以上ない場合) を示すフィールドを指しています。通常、これらの値は、リソースのリカバリーについての意味のあるメッセージを作成するために使用されます。システム障害の後に出口を起動したタスクは、リカバリー可能な作業を最初にスケジュールに入れたタスクではないため、これらが明示的に示されます。これらの追加パラメーターは、**元**のタスクの環境を表すものであり、直接アクセスされます。

完全なパラメーター・リストは、次のとおりです。

#### パラメーター 1

操作バイト 1 のアドレス。次のフラグが含まれます。

##### UERTPREP

(X'80') コミットするために準備します (つまり、2 フェーズ・コミットの最初のフェーズを実行します)。

##### UERTCOMM

(X'40') 無条件にコミットします (2 フェーズ・コミットの第 2 フェーズを実行します)。

##### UERTBACK

(X'20') バックアウトします。

##### UERTDGCS

(X'10') リカバリー単位は、CICS の初期始動が原因で失われました。

##### UERTDGNK

(X'08') リソース・マネージャーは、このリカバリー単位を未確定にしてはいけません。

##### UERTWAIT

(X'04') リソース・マネージャーは、このリカバリー単位の結果を待機する必要があります。この値は、CICS が UOW の結果を確定していない場合に、2 フェーズ・コミットのフェーズ 2 で設定されます。これが設定されるのは、タスク関連ユーザー出口を **INDOUBTWAIT** オプションを指定して有効にした場合のみです (47 ページの『特定の呼び出しタイプの使用可能化』を参照)。

##### UERTRSYN

(X'02') この同期点要求は、EXEC CICSRESYNC コマンドの結果として生成されました。

##### UERTLAST

(X'01') このタスクに関連付けられたリカバリー単位は、これ以上ありません。このビットが設定されていない場合、それ以上のリカバリー単位は、存在する場合もあれば存在しない場合もあります。このため、このビットに依存してタスク終了のシグナルを出すことは推奨されません。代わりに、スケジュール・フラグ・ワードのタスク・マネージャー・ビットを設定して、CICS タスク・マネージャーがタスク終了時に出口を起動するようにスケジュールする必要があります。

UERTLAST を使用してタスク終了のシグナルを出す場合は、その段階でクリーンアップ処理を実行できるのであれば、クリーンアップ処理の終了時にスケジュール・フラグ・ワードのタスク・マネージャー・ビットをオフに設定して、CICS タスク・マネージャーから不要に起動されないようにすることができます。

**ビットの有効な組み合わせ**は、UERTPREP、UERTCOMM、UERTBACK、UERTDGCS、および UERTDGNK の 1 つと、UERTLAST または UERTRSYN のいずれかあるいは両方を組み合わせて生成されたものか、または UERTWAIT と UERTLAST を組み合わせて生成されたものに限られます。

出口プログラムは、このバイトと操作バイト 2 の両方に設定されたフラグを検証して (パラメーター 10 を参照)、必要なアクションを決定する必要があります。

#### パラメーター 2

ゼロでない場合は、元のタスクを示す 4 バイトのバック 10 進数のフィールドのアドレスです。ただし、出口プログラムの起動のほとんどで、パラメーター 2 から 9 には値が含まれないことに注意してください。注 1 を参照してください。

### パラメーター 3

元のタスクを開始したトランザクションを示す 4 文字のフィールドのアドレス。注 1 を参照してください。

### パラメーター 4

元のタスクを開始した端末を示す 4 文字のフィールドのアドレス。注 1 を参照してください。

### パラメーター 5

元のタスクを開始した端末オペレーターの ID (OPID) を含む 4 文字のフィールドのアドレス。注 1 を参照してください。

### パラメーター 6

失敗した同期点の日付を含む 4 バイトのパック 10 進数フィールドのアドレス。形式は 0Cyyddds です。ここで、

- **C** は、世紀の標識です (0=1900、1=2000、2=2100 など)。
- **yy** = 年。
- **ddd** = 日。
- **s** は記号です。

注 1 を参照してください。

### パラメーター 7

失敗した同期点の時刻を含む 4 バイトのパック 10 進数フィールドのアドレスです。形式は、0hhmmss+ です。注 1 を参照してください。

### パラメーター 8

リソース・マネージャーの修飾子を含む 8 バイトのフィールドのアドレス。注 1 を参照してください。

これがリソース・マネージャーのこのインスタンスの再同期であることを確認するために、渡された修飾子が現在使用中のものであることを出口プログラムで確認する必要があります。そうでない場合、出口プログラムは、再同期を無視し、戻りコード UERFHOLD を設定して、CICS が作業単位の処理を保持する必要があることを示さなくてはなりません。

### パラメーター 9

次のトランザクション・コードを含む 4 文字フィールドのアドレス。トランザクションが次のトランザクション・コードを指定せずに EXEC CICS RETURN で終了した場合、アドレス指定されたフィールドには NULL が設定されています。そうでない場合は、アプリケーションから指定された値が設定されています。注 2 を参照してください。

### パラメーター 10

操作バイト 2 のアドレス。次のフラグが含まれます。

#### UERTONLY

(X'80') 単一フェーズのコミットを実行します (起動されたリソース・マネージャーが所有するリソース以外に、現在の UOW で更新されたリカバリー可能なリソースは存在しません)。

#### UERTELUW

(X'40') 単一フェーズのコミットを実行します (リソース・マネージャーは現在の UOW の間中、読み取り専用モードでした)。

出口プログラムは、このバイトと操作バイト 1 の両方で設定されるフラグを検証して (パラメーター 1 を参照)、予想されるアクションを決定する必要があります。

### 注:

1. パラメーター 2 から 8 に値が含まれるのは、セッションまたはシステムの障害が発生した後に、EXEC CICS RESYNC コマンドの発行により、CICS 同期点マネージャーの呼び出しが起動された場合のみです。操作バイト 1 には、ビット設定 UERTCOMM または UERTBACK が含まれます。そうでない場合は、X'00' (16 進数のゼロ) に設定されます。EXEC CICS RESYNC コマンドに関するプログラミング情報、およびシステム障害後の同期点処理手順の実行については、[RESYNC ENTRYNAME](#) を参照してください。

パラメーター 2 から 8 は、(現在 TRUE を起動したタスクではなく) 元のタスクの環境を表すことに注意してください。

2. UERTLAST ビットが操作バイト 1 に設定されていない限り、パラメーター 9 はゼロアドレスです。  
EXEC CICS RESYNC 呼び出しで起動された呼び出しの場合、UERTLAST ビットがオンに設定されますが、この場合、次のトランザクション・コードは適用されないため、パラメーター 9 は、NULL に設定されたフィールドをアドレス指定します。

#### CICS タスク・マネージャーのパラメーター

CICS タスク・マネージャーのパラメーター・リストには、TRUE の呼び出しの理由に応じて 1 つ、または 2 つの項目が入ります。タスク開始呼び出しの場合は 1 つの項目が入り、タスク終了呼び出しの場合は 2 つの項目が入ります。

各項目には 1 つのアドレスが入ります。パラメーター・リストの最後は、アドレスの先頭のビットが設定されることで示されます。

各パラメーターの意味は以下のとおりです。

#### パラメーター 1

呼び出しの理由を示すビット定義が入った 1 バイトのアドレス。

##### UERTSOTR

(X'40') CICS タスクの開始。

##### UERTEOTR

(X'80') CICS タスクの終了。

#### パラメーター 2

このパラメーターが渡されるのは、タスク終了呼び出しの場合だけです。EXEC CICS RETURN コマンドで指定されている次のトランザクション・コードが入る 4 文字フィールドのアドレスです。EXEC CICS RETURN で次のトランザクションが指定されていない状態でトランザクションが終了すると、このフィールドはヌルに設定されます。

CICS 同期点マネージャーからタスク関連ユーザー出口プログラムを無条件で呼び出す場合は、タスク開始呼び出しの時にスケジュール・フラグ・ワードを設定してください。

#### CICS 終了マネージャーのパラメーター

EXEC CICS ENABLE コマンドの SHUTDOWN オプションでタスク関連ユーザー出口プログラムを有効にし、開始した場合は、CICS の終了時にそのようなユーザー出口プログラムがすべて呼び出され、終了のタイプに応じたクリーンアップ処理が実行されます。CICS が終了すると、1 バイトの完了コードのアドレスが出口プログラムに渡されます。

そのコードに入る可能性があるビット設定を以下にまとめます。

##### UERTCORD

(X'80') CICS の正常シャットダウン

##### UERTCIMM

(X'40') CICS の即時シャットダウン

##### UERTCABY

(X'20') CICS の異常終了 (再試行可能、TCB ディスパッチ可能)

##### UERTCABN

(X'10') CICS の異常終了 (再試行不可能、TCB ディスパッチ可能)

##### UERTOPCA

(X'01') CICS の異常終了 (再試行不可能、TCB ディスパッチ不可能)

シャットダウン TRUE の詳細については、[42 ページの『CICS 終了時に呼び出されるプログラムのコーディング』](#)を参照してください。

#### CICS コンテキスト管理パラメーター

CICS コンテキスト管理パラメーターは、CICS コンテキスト管理用に CICS を呼び出すようタスク関連ユーザー出口 (TRUE) が指示したときに、CICS から渡されるパラメーターです。

スケジュール・ワード内のコンテキスト管理ビットが現行トランザクションに設定されている場合、トランザクションが非端末関連 **EXEC CICS START** コマンドを出すたびに、CICS コンテキスト管理は出口プ

ログラムを呼び出します。端末関連 **EXEC CICS START** コマンドに対しては出口プログラムが呼び出されません。

呼び出された出口プログラムに対して、以下のパラメーター・リストが渡されます (これは DFHUECON DSECT によってマップされます):

#### **UECON\_EXEC\_PLIST\_PTR**

**EXEC CICS START** コマンドに渡されるパラメーター・リストのコピーのアドレス。タスク関連ユーザー出口プログラムはこのアドレスを使用して、例えば **START** コマンドの **FROM** パラメーターで渡されるデータや、開始されるトランザクションの名前などにアクセスできます。**EXEC CICS START** パラメーター・リストは DFHICUED コピーブックによって記述されます。

#### **UECON\_CORRELATOR\_PTR**

出口プログラムが ARM ワークロード関係子を配置することのできる 512 バイト域のアドレス。

#### **UECON\_ICRX\_LEN**

将来使用するために予約済みです。

#### **UECON\_ICRX\_PTR**

将来使用するために予約済みです。

以下のアダプター・フィールドは階層順に使用されます。アダプター ID は最も一般的な情報を指定し、アダプター・データ 3 は最も具体的な情報を指定します。この順序は、タスクを識別して分離するための一貫した方法となります。

#### **UECON\_ADAPTER\_ID\_PTR**

起点データ・アダプター ID フィールドに入るデータを出口プログラムが渡すことのできる 64 文字域のアドレス。アダプターのすべてのインスタンスに対して同じ値を使用します (例えばアダプター所有者のプロダクト ID)。アダプターがこの領域で ID を指定しない場合、他のアダプター・データはまったく設定されません。

#### **UECON\_ADAPTER\_DATA1\_PTR**

起点データ・アダプターのデータ 1 フィールドに入るデータを出口プログラムが渡すことのできる 64 文字域のアドレス。このフィールドを使用して、アダプター・インスタンス (場合によっては多数のうち 1 つ) の接続先であるサーバーを識別できます。

#### **UECON\_ADAPTER\_DATA2\_PTR**

起点データ・アダプターのデータ 2 フィールドに入るデータを出口プログラムが渡すことのできる 64 文字域のアドレス。このフィールドを使用すると、**START** コマンドを使ってタスクを開始するアダプター・タスクのインスタンスを識別できます。

#### **UECON\_ADAPTER\_DATA3\_PTR**

起点データ・アダプターのデータ 3 フィールドに入るデータを出口プログラムが渡すことのできる 64 文字域のアドレス。このフィールドには、アダプター・インスタンスが **START** コマンドでこの特定のタスクを開始した理由を識別するための、詳細情報を含めることができます。

#### **UECON\_FLAGS**

1 バイトのアドレス。ここに含まれるビット定義は、**START** コマンドを発行しているタスクが起点のデータ・アダプター・データを含んでいるかどうかをアダプターに示します。

#### **UECON\_ADAPTER\_DATA\_ON**

(X'80') は、**START** コマンドを発行しているタスクが、起点のデータ・アダプター・データを含んでいるため、このタスクのセットの最初の (起点の) アダプターではないことを示しています。

リモート・トランザクションの場合、最初の接続時にミラー・トランザクションに適用されるアダプター・データは、そのミラーを使用する後続の **START** コマンドすべてで、個々の **START** コマンドの設定内容にかかわらず使用されます。

アダプター・トラッキング・サンプル **TRUE** である DFH\$APDT は、トランザクション・トラッキング用のアダプター・データ・フィールドの使用法を示します。詳しくは、『タスク関連ユーザー出口プログラムのサンプル DFH\$APDT (アダプター・トラッキング)』を参照してください。

#### CICS スイッチ・アプリケーション環境パラメーター

CICS がスイッチ・アプリケーション環境呼び出しのためにタスク関連ユーザー出口を呼び出す場合、明示的なパラメーターはありません。

CICS がオープン TCB の使用を停止する前に、その TCB で TRUE が実行されていると、その TRUE は、その TCB に関連したリソースを解放しなければなりません。場合によっては、CICS トランザクションが続行され、TRUE が別のオープン TCB で再び呼び出され、リソースがその新しいオープン TCB に関連付けられることもあります。

例えば、CONCURRENCY(REQUIRED) API(CICSAPI) として有効にされ、CICS Java アプリケーションから呼び出された TRUE は、T8 オープン TCB で実行されます。TRUE と `switch_application_environment` イベントとの関連性が設定されていると、その Java アプリケーションが完了した時に、その TRUE が呼び出され、リソースが T8 オープン TCB から解放されます。その後のタスク終了同期点では、同期点処理の実行のために、その TRUE が L8 TCB で呼び出され、リソースを L8 TCB に関連付けることができるようになります。

#### CICS EDF ビルド・パラメーター

EDF 呼び出しの場合、呼び出し側プログラムの RSA のレジスター 1 に入っているアドレスは、UEPEDFRM DSECT を指します。

DSECT には以下のフィールドがあります。

##### **UEPEDFR1**

アプリケーションの R1 パラメーター・リストのアドレス。

##### **UEPEDFFI**

入力フラグ・バイト。EDF によってタスク関連ユーザー出口が呼び出されると、UEPEDFFI に以下のビット設定が 1 つ以上入ります。

##### **UEPEDFRQ**

(X'80') 実行開始直前の呼び出しです。

##### **UEPEDFRS**

(X'40') コマンド実行完了の呼び出しです。

##### **UEPEDFRA**

(X'20') EDF にコマンドを表示する直前です。

##### **UEPEDFRC**

(X'10') EDF にコマンドが表示されました。

##### **UEPEDFSC**

(X'08') EDF ユーザーが画面を変更しました。

##### **UEPEDFWS**

(X'04') EDF ユーザーが作業用ストレージを変更しました。

##### **UEPEDFNO**

(X'01') EDF ユーザーが NOOP を要求しました。

##### **UEPEDFFO**

出力フラグ・バイト。タスク関連ユーザー出口が必要な場合は、UEPEDFFO フラグ・バイトを設定して、タスク関連ユーザー出口が EDF に実行させるアクションを EDF に指定できます。値は以下のとおりです。

##### **UEPEDDFD**

(X'80') CICS のデフォルト・アクションを実行します。(EDF 画面には、未解釈の呼び出し元の R1 パラメーター・リストが表示されます。)

##### **UEPEDFND**

(X'40') EDF にコマンドを表示しません。

##### **UEPEDFRD**

(X'20') EDF にコマンドを再表示します。

##### **UEPEDFDL**

EDF 画面の属性。通知専用です。タスク関連ユーザー出口プログラムでこのフィールドを変更することはできません。

### UEPEDFPS (ハーフワード・バイナリー)

ページ・サイズ (行数)。

### UEPEDFLS (ハーフワード・バイナリー)

行サイズ。

### UEPEDFMP (ハーフワード・バイナリー)

ページの最大数。

### UEPEDFPA

EDF 表示データ・パラメーター・リストのアドレス。タスク関連ユーザー出口で指定します。表示データ・パラメーター・リストには、属性バイト・アドレスとデータ・フィールド・アドレスを交互に並べたペアが入ります。属性バイトは、データ・フィールド・アドレスで指定されている表示データの行を参照します。データ・フィールドは、UEPEDFLS で指定されている値と同じサイズでなければなりません。表示データは、[30 ページの図 3](#)にあるような形式になります。

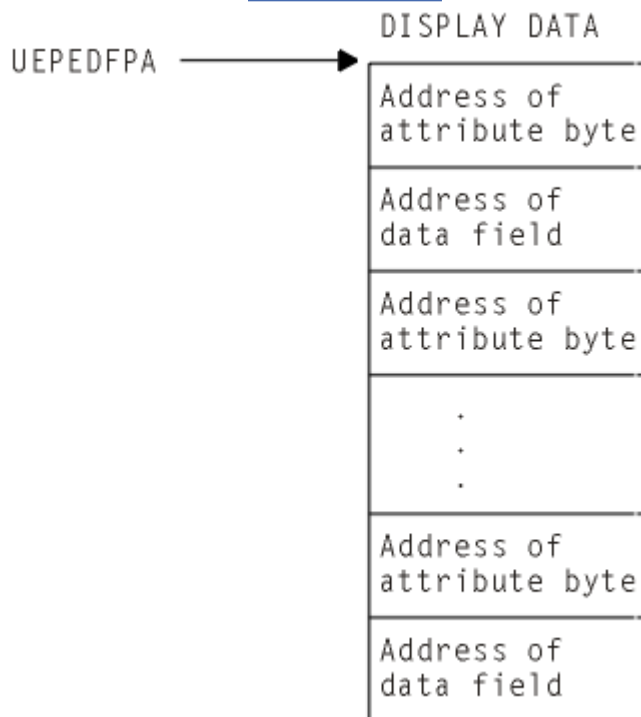


図 3. 表示データ・パラメーター・リスト

注：

1. CICS には、名前付きの標準的な属性バイトのリストが用意されています。その標準的な属性バイトが入っている DFHBMSCA をプログラムにコピーしてください。属性バイトのリストやそれぞれの意味などのプログラミング情報については、[BMS 関連の定数](#)を参照してください。
2. 最後のアドレスの上位ビットをオンに設定して、それが最後のアドレスであることを EDF に指定する必要があります。

### タスク関連ユーザー出口のパラメーター・リストの要約

[31 ページの図 4](#) は、前のセクションで説明されたパラメーター・リストの間の関係を図の形式で示しています。



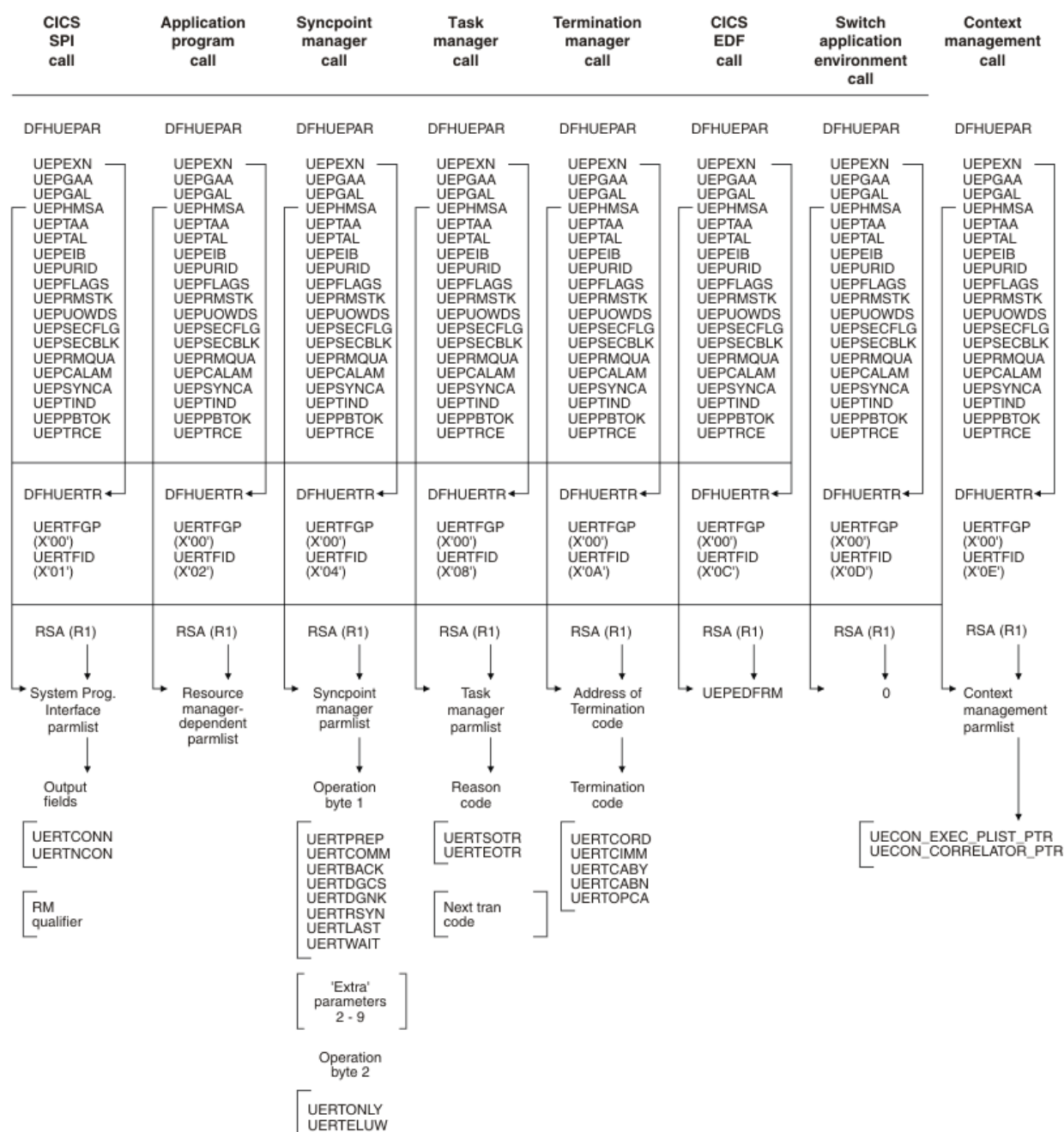


図 4. タスク関連ユーザー出口のパラメーター・リスト

### スケジュール・フラグ・ワード

スケジュール・フラグ・ワードとは、タスク関連ユーザー出口プログラムがそれ自身の呼び出しを制御するために使用するフルワード標識のことです。また、スケジュール・フラグ・ワードは、タスク関連ユーザー出口プログラムの最初の呼び出しをスケジュールするために CICS によって使用されます。

スケジュール・フラグ・ワードには、DFHUEPAR のアドレス・パラメーター UEPFLAGS でアクセスします。タスク関連ユーザー出口プログラムが使用可能になっている場合は、固有のスケジュール・フラグ・ワードが CICS タスクと指定された ENTRYNAME の間の関連ごとに存在します。

スケジュール・フラグ・ワードのデフォルト設定は、アプリケーション・プログラムの要求のためのもの（つまり、最後の 2 バイトが X'0004' に設定される）です。

表 3. スケジュール・フラグ・ワードの形式			
バイト	設定値	EXEC CICS ENABLE のオプション	コメント
0	—	—	予約済み
1	—	—	予約済み
2 UEFDCON UEFDSWAE UEFDFEDF UEFDCTER UEFDTASK	UEFMCON (X'40') UEFMSWAE (X'20') UEFMFEDF (X'10') UEFMCTER (X'04') UEFMTASK (X'01')	— — FORMATEDF SHUTDOWN TASKSTART	コンテキスト管理用のビット・マスク スイッチ・アプリケーション環境用のビット・マスク EDF 呼び出し用のビット・マスク 終了マネージャー用のビット・マスク タスク・マネージャー用のビット・マスク
3 UEFDSYNC UEFDAPPL UEFDSPI	UEFMSYNC (X'10') UEFMAPPL (X'04') UEFMSPI (X'02')	— — SPI	同期点マネージャー用のビット・マスク アプリケーション・プログラム用のビット・マスク SPI 用のビット・マスク

スケジュール・フラグ・ワードのビット設定は、どのプログラムがタスク関連ユーザー出口プログラムを呼び出すかを指定します。例えば、出口プログラムが CICS のタスク・マネージャー、CICS の同期点マネージャー、およびアプリケーション・プログラムによって呼び出される場合は、スケジュール・フラグ・ワードの最後の 2 バイトを X'0114' に設定する必要があります。出口プログラムが CICS のタスク・マネージャーおよびアプリケーション・プログラムのみによって呼び出される場合は、このフラグ・ワードの最後の 2 バイトを X'0104' に設定する必要があります。出口プログラムがタスクによって最初に呼び出される前に、CICS は API フラグ・ビットをオンに設定します。

この表の 3 番目の列は、呼び出しのタイプごとにビットを設定するために使用できる EXEC CICS ENABLE コマンドのオプション (ある場合) を示しています。(タスク関連ユーザー出口プログラムを特定のタイプの呼び出し用に呼び出すための EXEC CICS ENABLE コマンドのオプションの使用方法は、[47 ページの『特定の呼び出しタイプの使用可能化』](#)で説明されています。)

タスク関連ユーザー出口は、任意の呼び出しから戻る前に、フラグ・ワードのビット設定を変更して、異なる CICS 管理サービスによって呼び出されるその必要性を登録するか、または関連するフラグ・ビットをゼロに設定することでサービス内の必要性がないことを登録することができます。

例えば、タスク関連ユーザー出口は、CICS 以外のリカバリー可能なリソースにアクセスする必要があるアプリケーション・プログラムによって呼び出される場合があります。出口プログラムが最初に呼び出される際に、API ビットが CICS によってオンに設定されます。次に、呼び出し側プログラムがレコードを更新するための要求を出した場合は、出口プログラムはスケジュール・フラグ・ワード内の同期点マネージャーのビットをオンに設定します。その次に、呼び出し側アプリケーション・プログラムが同期点コマンドを発行するか、またはタスクの終了に達した場合は、CICS の同期点マネージャーが出口プログラムを呼び出します。

**注:** CICS は、同期点マネージャーの呼び出しが終了するごとに同期点マネージャー・ビットをオフに設定します。これは、CICS の同期点マネージャーが、出口プログラムがリカバリー中にリカバリー可能作業を実行しなかったリカバリー単位に対してタスク関連ユーザー出口プログラムを呼び出すことを避けるためのものです。このため、同期点マネージャー・ビットは、出口プログラムがリカバリー可能作業を実行するときは常にオンに設定される必要があります。

スケジュール・フラグ・ワード内のタスク・マネージャー・ビットをオンに設定した場合は、CICS はこのタスクの終了時にタスク関連出口プログラムを呼び出します。(出口プログラムをタスクの終了時だけでなく、**開始時**にも呼び出す場合は、TRUE 用の EXEC CICS ENABLE コマンド上に TASKSTART を指定する必要があることに注意してください。これにより、TRUE が**すべての**タスクの開始および終了時に呼び出されます。)



スケジュール・フラグ・ワードの最後の 2 バイトが X'1000' に設定された場合、これは、タスク関連ユーザー出口が要求を表示用にフォーマットするために EDF によって呼び出される必要があることを示しています。このスケジュール・フラグ・ビットの UEFD FDF は、EXEC CICS ENABLE FORMATEDF コマンドまたはタスク関連ユーザー出口のいずれかによってオンに設定されます。他のスケジュール・フラグ・ビットとは異なり、タスク関連ユーザー出口に EDF の必要性がないことを登録できる場合に関する制約事項 (つまり、UEFD FDF がオフに設定できる場合に関する制約事項) があります。タスク関連ユーザー出口が、EDF による「実行開始直前 (About to Execute)」または「コマンドの実行完了 (Command Execution Complete)」上への表示のために初期画面をフォーマットすると、CICS は画面作成サイクルが完了するまで、EDF ビットの UEFD FDF をオフに設定することを許可しません。

### タスク関連ユーザー出口プログラムでのレジスター処理

タスク関連ユーザー出口プログラムを作成する際、CICS レジスターや、呼び出し側プログラムのレジスター、RMI レジスターなど、さまざまなレジスター・タイプについて理解する必要があります。CICS レジスターは、タスク関連ユーザー出口を処理する CICS 管理モジュールによって使用されます。呼び出し側プログラムのレジスターは呼び出し側プログラムで使用されるレジスターであり、DFHUEPAR のパラメーター UEPHMSA によってアドレッシングされます。RMI レジスターは、リソース管理インターフェース (RMI) と呼ばれるインターフェースによって使用されます。

### CICS レジスターの保管

CICS レジスターの内容を保管することにより、タスク関連ユーザー出口プログラムを開始します。レジスター 13 は、18 ワード域をアドレッシングします。この領域の第 4 およびそれ以降のワードに、出口プログラムはレジスター 14 から 12 を保管しなければなりません。保管される値のうち、次の 3 つは重要です。

- レジスター 14 に保管される内容には、タスク関連ユーザー出口プログラムが制御を戻す CICS 内のアドレスが含まれます。
- レジスター 15 に保管される内容には、タスク関連ユーザー出口プログラムが入ったアドレスが含まれます。
- レジスター 1 に保管される内容は、タスク関連ユーザー出口プログラムのために CICS が提供するパラメーター・リスト (DFHUEPAR) をアドレッシングします。

注: 通則として、呼び出しの起点や目的を理解できない場合は、以下を行う必要があります。

1. 使用したすべてのレジスターを、コードに入ったときの状態に復元する。
2. CICS レジスター 14 に含まれているアドレスに戻る。

### 呼び出し側プログラムのレジスター

呼び出し側プログラムのレジスターは、DFHUEPAR の UEPHMSA によって指定されているアドレスに保管されます。呼び出し側プログラムが CICS 管理プログラム (例えば、同期点マネージャー) の場合、意味を持つ呼び出し側レジスターは、レジスター 1 とレジスター 15 だけです。レジスター 1 は、呼び出し側プログラムのパラメーター・リストをアドレッシングします。CICS は、タスク関連ユーザー出口プログラムが開始される前に、呼び出し側プログラムのレジスター 15 をゼロに設定します。呼び出し側のアイデンティティによっては、呼び出し側プログラムのレジスター 15 を、タスク関連ユーザー出口プログラムからの応答を呼び出し側プログラムに戻すために使用できる場合があります。呼び出し側プログラムが CICS 管理プログラムであるとき、このレジスターが戻り時にもまだゼロの場合、CICS はその呼び出しが理解されなかったとみなします。呼び出し側プログラムがアプリケーション・プログラムである場合は、戻り時のレジスター設定の意味は、リソース・マネージャーの資料に記述されているか、ローカルに定義されます。

### アドレッシング・モードの影響

タスク関連ユーザー出口 (TRUE) プログラムは、呼び出し元の AMODE で呼び出されます。ただし、出口プログラムを有効にする時に LINKEDITMODE オプションを指定した場合は除きます。

LINKEDITMODE オプションを使用すると、タスク関連ユーザー出口プログラムがリンク・エディット AMODE で有効になります。したがって、TRUE が AMODE 31 でリンク・エディットされている場合に、LINKEDITMODE オプションで有効にすると、その TRUE を 16 MB 境界より上で 2 GB 境界より下の場所に

置くことが可能になります。EXEC CICS ENABLE コマンドの LINKEDITMODE オプションのプログラミング情報については、[ENABLE PROGRAM コマンド](#)を参照してください。

**重要:** TRUE が AMODE 24 でリンク・エディットされている場合は、LINKEDITMODE オプションを使用しないでください。この組み合わせで設定すると、TRUE が常に AMODE 24 で実行されることになります。この状態は、以下の理由から望ましくありません。

- AMODE 24 TRUE は、TASKDATALOC(ANY) で実行するトランザクションからは呼び出せません。AEZB 異常終了の結果になります。
- タスク開始のために AMODE 24 TRUE プログラムを有効にすると、CICS がすべてのトランザクションを強制的に TASKDATALOC(BELOW) で実行します。
- CICS 終了呼び出しの場合は、TRUE を実行している TCA が 16 MB 境界より上にあることを CICS が検出すると、CICS は LINKEDITMODE オプションを無視して、TRUE を AMODE 31 で呼び出します。一部の終了タイプ (キャンセルなど) では、TRUE を実行する TCA をあらかじめ確定できないからです。

TRUE に関する推奨事項を以下にまとめます。

- 常に AMODE 31 で実行するように TRUE を作成します。
- TRUE AMODE 31 をリンク・エディットします。
- TRUE を LINKEDITMODE オプションで有効にします。

AMODE 64 TRUE はサポート対象外です。

タスク関連ユーザー出口プログラムを EXEC CICS ENABLE の LINKEDITMODE オプションなしで有効にした場合は、その出口プログラムが呼び出し元の AMODE で呼び出されます。例えばアプリケーション要求で、DFHRMCAL 要求の時点でアプリケーションが AMODE 24 であれば、タスク関連ユーザー出口プログラムが AMODE 24 で呼び出されます。したがって、LINKEDITMODE オプションなしで有効にしたタスク関連ユーザー出口プログラムは、16 MB 境界より下に置かなければなりません。

### 出口プログラムと CICS ストレージ保護機能

ストレージ保護機能を有効にして CICS を実行する場合は、タスク関連ユーザー出口について考慮しなければならないことが 2 点あります。タスク関連ユーザー出口プログラムの実行に使用される実行キーと、出口プログラムのために取得されるデータ・ストレージの記憶キーです。

### タスク関連ユーザー出口プログラムの実行キー

ストレージ保護をアクティブにして実行する場合、CICS はタスク関連ユーザー出口プログラムを常に CICS キーで開始します。プログラムのリソース定義で EXECKEY(USER) を指定しても、CICS は制御権を TRUE に渡す際に CICS キーを強制します。ただし、タスク関連ユーザー出口プログラム自体が (リンクや制御権移動コマンドを使用して) 制御権を他のプログラムに渡した場合、そのプログラムは、そのプログラム・リソース定義で定義された実行キー (EXECKEY) に従って開始されます。

**重要:** タスク関連ユーザー出口プログラムを定義する場合も、出口プログラムが制御権を渡すプログラムを定義する場合も、EXECKEY(CICS) を指定する必要があります。

### タスク関連ユーザー出口プログラムのデータ記憶キー

タスク関連ユーザー出口プログラムに使用されるストレージの記憶キーは、ストレージの取得方法によって異なります。

- 出口プログラムを有効にしたときに指定したグローバル作業領域またはローカル作業領域は、必ず CICS キーに置かれます。
- 出口プログラムのために取得される作業用ストレージは、出口プログラムを開始したトランザクションの TASKDATAKEY によって設定されるキーに置かれます。
- タスク関連ユーザー出口プログラムは、**EXEC CICS** コマンドを使用してストレージを取得できます。そのためには次のコマンドを発行します。
  - 明示的な **EXEC CICS GETMAIN** コマンド
  - SET オプションを使用した **EXEC CICS** コマンドの結果としてなされる暗黙的なストレージ要求。

**EXEC CICS** コマンドで取得されるストレージのデフォルトの記憶キーは、出口プログラムを開始したトランザクションの TASKDATAKEY によって設定されます。

### タスク関連ユーザー出口プログラム内の再帰

タスク関連ユーザー出口は再帰的に自己を呼び出せます。

例えば、自身の入り口名 (EXEC CICS ENABLE コマンドで指定した名前) に対する DFHRMCAL 呼び出しを発行することができます。また、SYNCPOINT 呼び出しを対象として設定した場合は、EXEC CICS SYNCPOINT を実行して、再帰的に入ることができます。

### タスクのパージ

タスク関連ユーザー出口内に制御権がある場合のタスクのパージ操作は、**ENABLE PROGRAM** コマンドの PURGEABLE オプションに応じて異なります。

PURGEABLE オプションを指定しない場合は、次のようになります。

- CICS は以下を禁止してから、タスク関連ユーザー出口プログラムに制御権を渡します。
  - タスクをパージする機能
  - ランナウェイ・タスクのモニター
- タスク関連ユーザー出口プログラム内に制御権がある間は、以下のようになります。
  - パージ要求は、タスク関連ユーザー出口プログラムから制御権が戻されるまで据え置かれます。
  - ランナウェイ・タスクのモニターは非アクティブになります。
  - 強制パージ要求は実行されます。

PURGEABLE オプションを指定した 場合、CICS はタスク関連ユーザー出口プログラムに制御権を渡す前にランナウェイ・タスクのモニターを禁止しますが、タスクをパージする機能は禁止しません。タスク関連ユーザー出口プログラム内に制御権がある間は、以下のようになります。

- パージ要求は実行されます。
- 強制パージ要求は実行されます。
- ランナウェイ・タスクのモニターは非アクティブになります。

### タスク関連ユーザー出口プログラムの待ち状態

デフォルトでは、タスク関連ユーザー出口内でアクティブになっていて CICS 待ち状態に入っているタスクはパージできません。強制パージのみ使用できます。しかし、PURGEABLE オプションを指定してタスク関連ユーザー出口を有効にした場合は、タスク関連ユーザー出口内で待ち状態のタスクを正常にパージすることができます。

このオプションを使用する場合は、タスク関連ユーザー出口プログラムを、待ち状態からパージされた応答を正しく処理できるように作成する必要があります。詳細については、[ENABLE PROGRAM コマンド](#)を参照してください。

### タスク関連ユーザー出口プログラム内での CICS サービスの使用法

出口プログラム内で CICS API コマンドを発行して、CICS サービスを呼び出すことができます。

しかし、以下のことに注意する必要があります。

- CICS の異常終了を基に呼び出されるプログラムでは、CICS サービスを使用しないでください。[42 ページの『CICS 終了時に呼び出されるプログラムのコーディング』](#)を参照してください。
- EXEC CICS XCTL や EXEC CICS SHUTDOWN などの、XCTL が (直接または暗黙的に) 実行される EXEC CICS コマンドは、決して使用しないでください。
- プログラム内に DFHEIENT および DFHEIRET を指定する必要があります。ただし、異常終了呼び出しでは DFHEIENT を使用しないという注記を [42 ページの『CICS シャットダウン時のタスク関連ユーザー出口の制限』](#)で確認してください。DFHEIENT マクロと DFHEIRET マクロの詳細については、[DFHECALL マクロ](#)を参照してください。
- 出口プログラムの入り口点の直後に、DFHEIENT マクロが CICS によって暗黙的に挿入されたかプログラムによって明示的に挿入された場合、DFHEIENT マクロの展開によって DFHEIBP と DFHEICAP に誤った

値が保管されます。その場合は、UEPEIB を DFHEIBP にコピーし、EIB 基底レジスター (DFHEIBR) を UEPEIB から再ロードし、DFHEICAP を X'80000000' に設定するようにコーディングすれば、この状況を訂正することができます。例えば、

```
TESTPROG DFHEIENT CODEREG=2,EIBREG=11,DATAREG=10
          USING DFHUEPAR,1
          MVC   DFHEIBP,UEPEIB           Get correct EIB address
          L     DFHEIBR,UEPEIB           Reload EIB base register
          MVC   DFHEICAP,=X'80000000'
```

プログラムの入り口点は、プログラムの先頭にある必要はなく、DFHEIENT マクロの後にすることができますので注意してください。

- DFHEIENT マクロは、DFHEISTG DSECT によってマップされる動的ストレージを割り振ります。動的ストレージを解放する DFHEIRET マクロを使用して CICS に戻る必要があります。
- コマンド・レベルの呼び出しは、レジスター 0、1、14、15 を使用します。
- タスクの開始、タスクの終了、または同期点呼び出しで同期点を発行しないでください。
- 同じタスクから呼び出された場合であっても、タスク関連ユーザー出口プログラムが呼び出されるたびに、新しい EXEC 環境が作成されます。したがって、リソース定義テーブルのブラウズなどの CICS 操作を、出口プログラムのある呼び出しから次の呼び出しに継続させることはできません。

### チャンネルおよびコンテナの使用

タスク関連ユーザー出口プログラムは、アプリケーション・プログラムによって作成されたチャンネルおよびコンテナにはアクセスできません。ただし、独自のチャンネルを作成して、呼び出し先プログラムにこれを渡すことができます。

チャンネルおよびコンテナについては、[チャンネルによるプログラム間データ転送](#)を参照してください。

### アセンブラー・プログラムと LEASM

LEASM オプションを指定して変換されたアセンブラー・プログラムは、タスク関連ユーザー出口プログラムとして使用できません。

LEASM は、Language Environment に準拠したアセンブラーのメインプログラムを生成するために使用します。LEASM 変換プログラム・オプションについて詳しくは、[変換プログラム・オプション](#)で LEASM を参照してください。

### 作業域

**EXEC CICS ENABLE PROGRAM** コマンドを使用して、タスク関連ユーザー出口プログラムを CICS に定義するときに、このプログラムに 1 つのローカル作業域と 1 つのグローバル作業域へのアクセス権があることを指定できます。

### グローバル作業域

グローバル・ユーザー出口プログラムと同様に、タスク関連ユーザー出口プログラムも、独自のグローバル作業域を所有すること、また、別の出口プログラムが所有する作業域を共用することができます。

出口を定義するために **ENABLE PROGRAM** コマンドを発行するときに **GALENGTH** オプションと **GALLOCATION** オプションを指定して、出口プログラム用のグローバル作業域を CICS に用意させることができます。GALENGTH はグローバル作業域の長さをバイト単位で指定し、GALLOCATION は作業域を 24 ビット・ストレージに置くか 31 ビット・ストレージに置くかを指定します。また、GAENTRYNAME オプションを指定して、現在有効にされている別のユーザー出口を指定し、その出口用に CICS が既に用意したグローバル作業域をこの出口プログラムで共用することもできます。

グローバル作業域は、出口点ではなく出口プログラムに関連付けられます。問題判別を簡単にするために、グローバル作業域を共用するのは同じ管理モジュールまたはドメインから呼び出された出口プログラムに限る必要があります。グローバル作業域のアドレスおよび長さは、DFHUEPAR パラメーター・リストの **UEPGAA** および **UEPGAL** によってアドレス指定されます (5 ページの『DFHUEPAR 標準パラメーター』を参照)。ユーザー出口プログラム専用のグローバル作業域がない場合は、UEPGAA はゼロに設定されます。

アプリケーション・プログラムは、同じグローバル作業域を使用または共用するユーザー出口プログラムと情報を交換できます。アプリケーション・プログラムでは **EXEC CICS EXTRACT EXIT** コマンドを使用してグローバル作業域のアドレスと長さを取得します。



## ローカル作業域

タスク作業域とも呼ばれるローカル作業域は、以下の特性を持ちます。

- 単一のタスクと関連付けられます。
- タスクの存続期間のみ持続します。
- 単一のタスク関連ユーザー出口プログラムで使用するためのものです。

ローカル作業域は、出口プログラムで排他的に使用するためのトランザクション作業域 (TWA、TWACOB) を論理的に拡張したものと考えられます。

**ENABLE PROGRAM** コマンドを発行して出口プログラムを定義するときに、**TALENGTH** オプションを指定すると、出口を使用するタスクごとに **CICS** にローカル作業域を提供させることができます。**CICS** は、作業域を割り振り、タスクの終了時に解放します。**ENABLE PROGRAM** コマンドで **LINKEDITMODE** オプションを指定してタスク関連ユーザー出口プログラムを有効にしている、出口プログラムを **AMODE(31)** でリンク・エディットした場合、ローカル作業域は 31 ビット・ストレージ内に配置されます。**LINKEDITMODE** オプションを指定しなかった場合、またはタスク関連ユーザー出口プログラムを **AMODE(24)** でリンク・エディットした場合は、ローカル作業域は 24 ビット・ストレージに配置されます。

ローカル作業域のアドレスおよび長さは、**DFHUEPAR** パラメーター・リストのパラメーター **UEPTAA** および **UEPTAL** でアドレス指定されます (20 ページの『**DFHUEPAR**』を参照)。

### CICS SPI により開始される出口プログラムの実行

タスク関連出口プログラムに対して **EXEC CICS ENABLE PROGRAM** コマンドで **SPI** オプションを指定した場合 (または、プログラムでスケジュール・フラグ・ワードの **SPI** ビット・マスクを設定した場合)、**EXEC CICS INQUIRE EXITPROGRAM** コマンドを使用して、出口プログラムとそのリソース・マネージャーの接続の有無、および入り口名修飾子を照会したときに、プログラムは開始されます。

### このタスクについて

**INQUIRE EXITPROGRAM** コマンドの詳細については、[INQUIRE EXITPROGRAM](#) を参照してください。

### 手順

1. **SPI** 呼び出しで開始された出口プログラムは、外部リソース・マネージャーに接続しているかどうかを、呼び出し元パラメーター・リストでアドレス指定された最初のフィールドに、対応する値を返すことで示します。

- 以下の値が返されます。

#### UERTCONN

(X'80') 出口はそのリソース・マネージャーに接続されています。

#### UERTNCONN

(X'40') 出口はそのリソース・マネージャーに接続されていません。

- 呼び出し元パラメーター・リストでアドレス指定された 2 番目のフィールド内に、リソース・マネージャー修飾子、つまり入り口名修飾子を返します。これは、API 呼び出しの **UEPRMQUA** パラメーターを使用して返され、**EXEC CICS RESYNC** コマンドで使用されます。

一般的には、両方の情報が出口プログラムのグローバル作業域内に保持されます。**SPI** 呼び出しの呼び出し元パラメーター・リストについては、24 ページの『**CICS SPI パラメーター**』で説明しています。

2. **RMI SPI** 呼び出しは、タスク関連ユーザー出口を長期実行モニター・タスクから呼び出すことを可能にします。これは、タスクが最後に呼び出された後に一度無効にされ、再び有効にされた場合であってもです。他のタイプの **RMI** 呼び出しはすべて、このような状況では失敗します。

**SPI** 呼び出しで出口プログラムを開始する場合は、タスクのローカル作業域の内容に依存する出口プログラムにしないでください。出口が一度無効にされ、再び有効にされた場合、タスクのローカル作業域のマッピングが異なる新しいバージョンがロードされている可能性があります。一方、長期実行タスクは、その最初の呼び出しで割り振られた元のタスクのローカル作業域を使用して実行されています。

## CICS 同期点マネージャーによって開始されるプログラムのコーディング

すべてのタスク関連ユーザー出口プログラムは、CICS 同期点マネージャーから開始できます。

### このタスクについて

出口プログラムは、スケジュール・フラグ・ワードの同期点マネージャーのビット・マスクを設定して、同期点マネージャーを「スケジュールする」必要があります。リカバリー可能な各作業単位の後にはビット・マスクを設定して、CICS 同期点マネージャーが同期点処理中に出口プログラムを呼び出すようにする必要があります。現在のリカバリー単位または作業単位の ID は、8 バイトのフィールド UEPURID によってアドレス指定されます。このフィールドは、アプリケーション呼び出しやそれに続く同期点マネージャー呼び出しなど、リカバリー可能なアクションが想定される出口プログラムのすべての呼び出しで使用できます。

### 効率の向上: *single-update and read-only* プロトコル

リソース・マネージャーが単一フェーズ・コミットを実行できる場合、CICS *single-update and read-only* プロトコルを使用してシステムの効率を上げることができます。

#### 単一更新プロトコル

多くの CICS トランザクションは、外部リソース・マネージャーを 1 つのみ使用します。この場合、単一フェーズ・コミットが適切です。

単一フェーズ・コミットの利点を以下に示します。

- トランザクションに必要なログの強制書き出しの数をリソース・マネージャーが 2 つから 1 つに減らせます。
- CICS が書き込むトランザクション関連のログ・レコードの数が減ります。
- 同期点での TRUE の呼び出しが 2 回ではなく 1 回のみになるため、パスの長さが短縮されます。

これらの利点を活用するには、リソース・マネージャーが単一更新プロトコルを理解していることと、このプログラム (TRUE) が同期点呼び出しを処理して単一フェーズ・コミットを実行できることを、タスク関連ユーザー出口プログラムで CICS に知らせる必要があります。このためには、出口プログラムで、DFHUEPAR パラメーター・リスト内の UEPSYNCA が指すフィールドに UEPSUPDR フラグを設定します。スケジュール・フラグ・ワードの同期点マネージャー・ビットを設定するたびに、この設定を行う必要があります。

出口プログラムで UEPSUPDR フラグを設定した場合、同期点マネージャーは次にその TRUE を呼び出すときに、そのリソース・マネージャーが、現在の UOW でリソースが更新された唯一のリソース・マネージャーであるかどうかを通知します。これは、UERTONLY ビット (同期点マネージャーのパラメーター・リストの操作バイト 2) を使用して通知します。このビットが設定されている場合は、単一フェーズ・コミットを実行するようにリソース・マネージャーに要求できます。

#### 読み取り専用プロトコル

リソース・マネージャーが現行の作業単位 (UOW) の初めから終わりまで読み取り専用モードである場合も、同様に効率を向上させることができます。

この場合も、単一フェーズ・コミットが適切です。活用するためには、UOW が読み取り専用かどうかを示すフラグを、リソース・マネージャーが TRUE に返す必要があります。そのフラグにより、これまでの UOW の「ヒストリー」(今までのところは読み取り専用であるなど)を示したり、現行の要求が読み取り専用かどうかを示したりします。これに対して、TRUE では、これまでの UOW のヒストリーを使用して、DFHUEPAR パラメーター・リスト内の UEPREADO フラグを更新する必要があります。つまり、最初は UEPREADO を設定しておき、UOW が更新を含むようになったらただちに設定を解除する必要があります (UEPREADO がいったん設定解除されると、現行の UOW の間、CICS はフラグのそれ以降の設定を無視し、UOW を更新を含むものとして扱います)。

UOW の終了時にまだ UEPREADO フラグが設定されている場合は、同期点マネージャーが、リソース・マネージャーに対する単一フェーズ・コミットの発行命令を指定して (UERTELUW ビットをオンに設定して) TRUE を呼び出します。



## 戻りコード

CICS 同期点マネージャーがタスク関連ユーザー出口プログラムを呼び出す場合、設定できる戻りコードは、呼び出された理由に応じて異なります。

39 ページの表 4 に、同期点マネージャーのパラメーター・リスト内の要求フラグと TRUE 戻りコードの関係を示しています (CICS 同期点マネージャー・パラメーターについては、24 ページの『CICS 同期点マネージャーのパラメーター』で説明しています)。

表 4. CICS 同期点マネージャーから呼び出される TRUE にとって有効な戻りコード		
要求タイプ	戻りコード	意味
UERTPREP	UERFPREP	2 フェーズ・コミットのフェーズ 1 が成功
UERTPREP	UERFBACK	2 フェーズ・コミットのフェーズ 1 が失敗
UERTWAIT	なし	適用外
UERTCOMM	UERFDONE	2 フェーズ・コミットのフェーズ 2 が成功
UERTCOMM	UERFHOLD	2 フェーズ・コミットのフェーズ 2 が失敗
UERTBACK	UERFDONE	バックアウトが成功
UERTBACK	UERFHOLD	バックアウトが失敗
UERTONLY	UERFOK	単一フェーズ・コミットが成功
UERTONLY	UERFBOUT	単一フェーズ・コミットが失敗してバックアウト
UERTELUW	なし	適用外

## リソース・マネージャーに期待されること

同期点マネージャーからのすべての要求で、リソース・マネージャーからの意味のある応答を求めるようにすれば、CICS はリカバリー可能リソース(データベースなど) への変更を同期させることができます。したがって、システム障害が発生した場合でも、すべての変更が反映されるか、すべてがバックアウトされるかのどちらかになります。

## 制限

同期点呼び出し中は、CICS 同期点マネージャーが変更を認識しないので、リカバリー可能な CICS リソースを更新しないでください。

## CICS 同期点マネージャーにより開始される TRUE のサンプル・コード

この例の疑似コードは、同期点で開始されたタスク関連ユーザー出口で検査する必要がある状況をいくつか示しています。

```
if UERTFID = UERTSYNC then          /* Caller is CICS syncpoint manager */
  select;                            /* Type of syncpoint manager request */
    when (UERTONLY)                  /* ONLY resource manager */
      invoke RM for single-phase commit /* Single-phase commit */
      if RM single-phase commit succeeded then
        give CICS syncpoint manager 'YES' vote (UERFOK)
      else
        /* Single-phase commit failed */
        /* If RM completed backout */
        if RM single-phase commit failed and backed out
          give CICS syncpoint manager 'NO' vote (UERFBOUT)
        else
          /* Don't know what happened */
          put out message and issue transaction abend
        endif
      endif
    when (UERTEL UW)                  /* RM read-only for current UOW */
      invoke RM for single-phase commit /* Single-phase commit */
    when (UERTPREP)                   /* Not ONLY resource manager, nor read-only */
      invoke RM for PREPARE /* Prepare - phase 1 of 2-phase commit */
      select (resource manager vote)
        when (YES)                   /* Phase 1 completed */
          give CICS syncpoint manager 'YES' vote (UERFPREP)
        otherwise
          give CICS syncpoint manager 'NO' vote (UERFBACK)
      endselect
    when (UERTCOMM)                   /* Commit - phase 2 of 2-phase commit */
      invoke RM for commit phase 2
      if RM commit succeeded then
        tell CICS sync manager OK (UERFDONE)
      else
        tell CICS sync manager remember could not commit (UERFHOLD)
      endif
    when (UERTBACK)                   /* Backout request */
      invoke RM for backout
      if RM backout succeeded then
        tell CICS sync manager OK (UERFDONE)
      else
        tell CICS sync manager remember could not backout (UERFHOLD)
      endif
    when (UERTWAIT)                   /* CICS indoubt about UOW */
      invoke RM to free thread
      (but maintain locks for UOW and record UOW is indoubt)
    endselect
  endif
```

図 5. CICS 同期点マネージャーにより開始されるタスク関連ユーザー出口プログラムのサンプル疑似コード

38 ページの『[効率の向上: single-update and read-only プロトコル](#)』で説明しているように、UERTONLY ビットが設定されている場合は(このリソース・マネージャーが、リソースを更新された唯一のリソース・マネージャーであることを示す)、出口プログラムにより、リソース・マネージャーに単一フェーズ・コミットを実行させる必要があります。コミットが成功した場合は、出口プログラムはレジスター 15 に「UERFOK」を返し、失敗した場合は「UERFBOUT」(コミットが失敗し、リソースがバックアウトされたことを示す)を返す必要があります。単一フェーズ・コミットの結果を出口プログラムで確認できない場合は、必要と見なす診断情報を保管または表示して、タスクを異常終了させる必要があります。

このセクションの「レジスター 15」は、同期点マネージャーのレジスター 15 (UEPHMSA でアドレス指定される区域の第 5 ワード)を指していることに注意してください。

同様に、UERTEL UW ビットが設定されている場合も(リソース・マネージャーがこの UOW の間中、読み取り専用モードであったことを示す)、出口プログラムにより、リソース・マネージャーに単一フェーズ・コミットを実行させる必要があります。UERTEL UW 呼び出しの場合は、戻りコードはありません。更新は行われていないので、データ保全性は損なわれていません。したがって、コミットが失敗しても何のアクションも実行されません。

2 フェーズ・コミットの最初のフェーズを実行する要求を受け取ったら、リソース・マネージャーは、システム障害が発生して中断された場合であっても、最後の同期点以降に加えられたリカバリー可能な変更を、要求に応じてコミットまたはバックアウトできる状態になることが期待されます。例えば、バッファの内容を不揮発性ストレージに移動する必要があります。リソース・マネージャーがこのような状態にならない場合は、出口プログラムが、バックアウトを要求する「UERFBACK」をレジスター 15 に返す必要があります。通常は、2 フェーズ・コミットのフェーズ 1 が成功したことを示すために、「UERFPREP」を返す必要があります。

待機要求を受け取ったら (CICS が UOW の結果を確定していないことを示します)、リソース・マネージャーは、スレッドなどのタスク関連リソースを解放する必要があります。しかし、UOW で保持されているロックは維持し、UOW が未確定であることを記録する必要があります。[47 ページの『特定の呼び出しタイプの使用可能化』](#)を参照してください。

2 フェーズ・コミットの 2 番目のフェーズを実行する要求、またはバックアウトする要求を受け取ったら、リソース・マネージャーは、対応する取り消し不能なステップを実行し、出口プログラムから同期点マネージャーに戻りコードを送信させる必要があります。戻りコード「UERFDONE」は、コミットまたは異常終了のプロセスが完了したことを示し、「UERFHOLD」は、コミットまたは異常終了を後で解決する必要がありますことを示します。これらの戻りコード定数は、出口プログラム内にマクロ DFHUEEXIT TYPE=RM をコーディングすると使用できるようになります。

リソース・マネージャーが呼び出しを理解できない場合は、呼び出し元に戻る前に呼び出し元のレジスター 15 の内容を変更してはいけません。呼び出し元が変更をどのように解釈するかを予測できないからです。

### 再同期

2 フェーズ・コミットのフェーズ 1 を実行した後には出口から戻ってから、それに続くフェーズ 2 またはバックアウト呼び出しが行われるまでの間に障害が発生した場合、再始動時に、リソース・マネージャーがリカバリー単位の状態を検出してそれに応じて処理を行うように準備ができていなければなりません。

**EXEC CICS RESYNC** コマンドを使用した再始動再同期に関するプログラミング情報については、[RESYNC ENTRYNAME](#) を参照してください。

CICS が作業単位について未確定の場合は、出口プログラムに待機要求 (UERTWAIT) を送信します。未確定作業単位の状況が解決されると、CICS は作業単位の結果を出口プログラムに通知するために、再同期タスクを開始します。

未確定作業単位に関する情報は、CICS のウォーム・スタートとコールド・スタートの両方を通じて保持されます。CICS 初期化およびキーポイント管理ルーチンは、リソース・マネージャーが未確定作業単位と関連付けられているすべての情報をシステム・ログから復旧し、CICS が関係するリソース・マネージャーに再接続したときにこの問題は自動的に解決されます。

### CICS タスク・マネージャーから呼び出されるプログラムのコーディング

出口プログラムでスケジュール・フラグ・ワードのタスク・マネージャー・ビットを設定した場合、出口プログラムはタスクの終了時に呼び出されます。TRUE に対して EXEC CICS ENABLE コマンドで TASKSTART を指定した場合は、タスクの開始時と、(タスク・マネージャー・ビットの設定を解除していなければ) タスクの終了時の両方で呼び出されます。

### このタスクについて

特定の呼び出しがタスクの開始時と終了時のどちらで呼び出されるかを調べるには、CICS タスク・マネージャー・パラメーター ([27 ページの『CICS タスク・マネージャーのパラメーター』](#)を参照)を確認してください。通常、プログラムにタスク・マネージャー・イベントが関係してくるのは、タスクが終了する前にパフォーマンス・データやアカウンティング・データなどのタスク関連の情報を保管する必要がある場合です。

### 出口プログラムの制限

タスク関連ユーザー出口プログラムをタスク終了 (EOT) 時に呼び出す場合は、タスク削除時の出口プログラム・アクティビティーに対する制約に注意する必要があります。

- タスク削除出口呼び出し中には、CICS リソースを一切更新しないでください。そのタスクで CICS 同期点マネージャーが呼び出されることは二度とないからです。EOT までに、タスク・ストレージ以外のリソースはすべて解放されています。

注: リソース・セキュリティまたはコマンド・セキュリティが定義されているトランザクションは、端末が解放された後は正常に実行されないことがあります。セキュリティ検査対象のリソースおよびコマンドを調べるには、[リソースおよびコマンドの検査の相互参照](#)を参照してください。これらの制限を守らないと、ABEND AEY7 - NOTAUTH 状態が発生することがあります。

- 出口プログラムから **EXEC CICS START** コマンドを使用して新しい CICS タスクをスケジュールし、データを新しいタスクに渡すことができます。しかし、**EXEC CICS START** コマンドは一時記憶域キューを使用してデータを新しいトランザクションに渡します。このキューがリカバリー可能と定義されていると、そのキューは削除タスクのためにロックされます。キューがアンロックされることはありません。タスク削除出口呼び出しが行われたときには、削除タスクのリソースは既に解放されているからです。PROTECT オプションを使用すると、別の問題が生じます。この場合、削除タスクの次の同期点まで新しいタスクをスケジュールできませんが、同期点が取られることは二度とないからです。
- タスク関連ユーザー出口プログラムではリモート・リソースにアクセスしないでください。アクセスする場合は、機能シップの会話が終了する可能性のある状況について理解しておく必要があります。

### CICS 終了時に呼び出されるプログラムのコーディング

タスク関連ユーザー出口プログラムを有効にするときに SHUTDOWN オプションを指定した場合、出口プログラムはシステム終了時に呼び出されます。

### このタスクについて

CICS システム終了マネージャーは、終了のタイプを示す 1 バイトのコードのアドレスを出口プログラムに渡します (27 ページの『[CICS 終了マネージャーのパラメーター](#)』を参照)。この呼び出しによるプログラムを使用して、終了のタイプに適した処理を実行できます。例えば、正常シャットダウンの場合に、PLT プログラムではなくタスク関連ユーザー出口プログラムを使用してアダプターをシャットダウンできます。また、CICS 異常終了の場合に、異常終了の重大度に応じて特別なアクションを実行することもできます。

### CICS シャットダウン時のタスク関連ユーザー出口の制限

シャットダウン時にタスク関連ユーザー出口 (TRUE) が呼び出されると、この出口プログラムの機能は制限されます。

CICS の異常終了およびオペレーターによるキャンセルの特性として、SHUTDOWN が指定されている場合でも、CICS がシステムの終了時にユーザー出口プログラムを呼び出せない場合があります。

ユーザー・プログラムが呼び出された場合に実行できる機能に対する制限は、終了のタイプに応じて異なります。

#### 正常シャットダウン (UERTCORD)

ユーザー出口プログラムは、CICS シャットダウンの最初の静止段階中に実行されているプログラムに関する規則に従う必要があります。この段階では、すべての CICS サービスが使用可能であるものの、プログラムは新規タスクを開始してはなりません。

#### 即時シャットダウン (UERTCIBM)

ユーザー出口プログラムは、必要最低限の処理を実行し、制御を戻すことにより、シャットダウンが進行可能になるようにしなければなりません。

#### CICS 異常終了、再試行可能、TCB ディスパッチ可能 (UERTCABY)

MVS は、「再試行に適格」として障害にフラグを立てています。ユーザー出口プログラムは、[z/OS MVS Programming: Authorized Assembler Services Guide](#) に記載されている、このタイプの障害に関する MVS 規則に従う必要があります。

領域内のサブタスク (つまり、CICS ジョブ・ステップ TCB に加えてタスク制御ブロック (TCB)) は引き続きディスパッチ可能で、ユーザー出口プログラムはそれらの下でコードを実行できます。

CICS サービスはいずれも使用できません。

#### CICS 異常終了、再試行不可、TCB ディスパッチ可能 (UERTCABN)

MVS は、「再試行に不適格」として障害にフラグを立てています。ユーザー出口プログラムは、このタイプの障害に関する MVS 規則に従う必要があります。ユーザー出口プログラムは、CICS 拡張サブタスク異常終了出口 (ESTAE) のコードから呼び出されます。MVS は、非 ESTAE コードに対するよりも多くの制限を ESTAE コードに課します。

領域内のサブタスクは引き続きディスパッチ可能で、ユーザー出口プログラムはそれらの下でコードを実行できます。

CICS サービスはいずれも使用できません。

#### **CICS 異常終了、再試行不可、TCB ディスパッチ不可 (UERTOPCA)**

領域内のサブタスクがディスパッチ不可であることを除き、UERTCABN の場合、ユーザー出口プログラムは、アタッチされている可能性があるいずれの TCB の下でもコードの実行を試みることができません。

### **重要**

異常終了での呼び出し (UERTCABY から UERTOPCA) の場合は、CICS GETMAIN を実行する DFHEIENT 呼び出しを含め、ユーザー出口プログラムはいずれの CICS サービスも使用できません。ユーザー・プログラムの呼び出しのたびに DFHEIENT 呼び出しが自動的に発行されないようにするには、NOPROLOG 変換プログラム・オプションを指定します。ただし、プログラム・ソースに、異常終了以外の呼び出しでのみ発行されるようにユーザー独自の DFHEIENT 呼び出しを組み込んでください。CICS の終了時にタスク関連ユーザー出口プログラムが呼び出されるようにコーディングする方法の例については、[44 ページの図 6](#)を参照してください。DFHEIENT 呼び出しのコーディングについて詳しくは、[DFHECALL マクロ](#)を参照してください。

#### **CICS 終了時に呼び出される TRUE のサンプル・コード**

[44 ページの図 6](#) のサンプルは多目的プログラムであり、多くのタスク関連ユーザー出口呼び出し点で呼び出すためにコーディングされていることに注意してください。しかし、すべての多目的 TRUE で CICS 異常終了をテストしなくて済むように、終了呼び出しには別のプログラムを使用することをお勧めします。





```

DFHEISTG DSECT
*
*      Local working storage for CSECT JTRUE1B
*
RSA      DS      18F      Register save area
SA      DSECT      Register save area DSECT
        DS      F
*
RSACB    DS      F      +004
RSACF    DS      F      +008
RSAR14   DS      F      +00C
RSAR15   DS      F      +010
RSAR0     DS      F      +014
RSAR1     DS      F      +018
RSAR2     DS      F
RSAR3     DS      F
RSAR4     DS      F
RSAR5     DS      F
RSAR6     DS      F
RSAR7     DS      F
RSAR8     DS      F
RSAR9     DS      F
RSAR10    DS      F
RSAR11    DS      F
RSAR12    DS      F
        DFHREGS
        DFHUEXIT TYPE=RM
        DFHEISTG
        DFHEIEND
        PRINT NOGEN
        PRINT GEN
        END

```

図 7. CICS 終了時に呼び出されるタスク関連ユーザー出口プログラムのサンプル・コード (パート 2)

### タスク関連ユーザー出口プログラムと EDF の使用

出口プログラムでスケジュール・フラグ・ワードの EDF ビットを設定し、EDF がアクティブな場合、出口プログラムは各 API 要求の前後で呼び出され、EDF の表示画面のフォーマットを調整します。

タスク関連ユーザー出口と EDF の間の通信は、タスク関連ユーザー出口インターフェースによって制御されます。EDF とタスク関連ユーザー出口の間のこのインターフェースを流れるコマンド・フローについて、[45 ページの図 8](#) にまとめています。

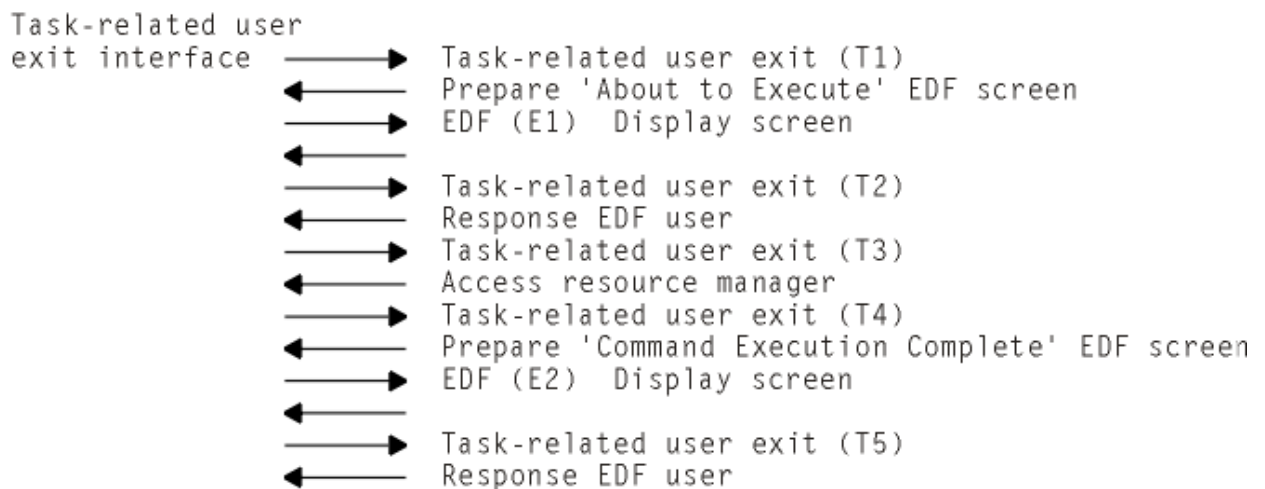


図 8. タスク関連ユーザー出口と EDF の間のインターフェース

46 ページの表 5 で、タスク関連ユーザー出口と EDF の間のインターフェースの各ステージについて説明します。それぞれの説明は、[45 ページの図 8](#) の (Tn) および (En) という表記に対応しています。

表 5. タスク関連ユーザー出口/EDF インターフェースの各ステージの説明	
呼び出し	説明
(T1)	EDF 要件を設定するためにタスク関連ユーザー出口が呼び出されます。このステージでは、タスク関連ユーザー出口がアプリケーション要求に基づいて「実行開始直前 (About to Execute)」画面を用意します。
(E1)	タスク関連ユーザー出口インターフェースが、呼び出し T1 でタスク関連ユーザー出口から戻された情報を使用し、EDF を呼び出して「実行開始直前 (About to Execute)」画面を表示します。EDF は EDF ユーザー応答を設定します (例えば、ユーザーが画面を変更した場合など)。
(T2)	「実行開始直前 (About to Execute)」画面への EDF ユーザー応答によって、タスク関連ユーザー出口が呼び出されます。
(T3)	アプリケーション要求で外部リソース・マネージャーにアクセスするためにタスク関連ユーザー出口が呼び出されます。
(T4)	アプリケーション要求の結果に基づいて「コマンドの実行完了 (Command Execution Complete)」画面を用意するためにタスク関連ユーザー出口が呼び出されます。
(E2)	タスク関連ユーザー出口インターフェースが、呼び出し T4 でタスク関連ユーザー出口から戻された情報を使用し、EDF を呼び出して「コマンドの実行完了 (Command Execution Complete)」画面を表示します。EDF は EDF ユーザー応答を設定します (例えば、ユーザーが画面を変更した場合など)。
(T5)	「コマンドの実行完了 (Command Execution Complete)」画面への EDF ユーザー応答によって、タスク関連ユーザー出口が呼び出されます。

## 重要

E1/T2 および E2/T5 サイクルは繰り返し使用される場合があります。例えば EDF ユーザーが画面を何度も変更した場合などです。

## アダプターの管理

タスク関連のユーザー出口を注意深く使用するなら、アプリケーション・プログラマーは、CICS アプリケーション・プログラムから CICS でないリソース・マネージャーを呼び出すことによる影響を回避できます。あるインストール・システムでタスク関連のユーザー出口プログラムを有効にしたり無効にしたりすることは、監視オペレーターまたはマスター端末オペレーターの責任でなければなりません。

### アダプターを使用する前に必要な作業

タスク関連ユーザー出口プログラムを実行できるようにするには、その前に有効にして開始しておく必要があります。

### 手順

1. **CEDA INSTALL PROGRAM** コマンドを使用して、タスク関連ユーザー出口プログラムをシステムに定義します。
2. **EXEC CICS ENABLE PROGRAM** コマンドを使用して、タスク関連ユーザー出口プログラムを有効にし、その作業用ストレージの要件を定義します

### 例

```
EXEC CICS ENABLE PROGRAM('EP9')
      TALENGTH(750) GALENGTH(200) SHUTDOWN

EXEC CICS ENABLE PROGRAM('EP9')
      START
```

最初のコマンドは、タスク関連ユーザー出口プログラム EP9 をロードし、200 バイトのグローバル作業域を獲得して、このプログラムに関連付けます。31 ビット・ストレージ内にグローバル作業域を配置するには、コマンドの GALLOCATION オプションに CVDA LOC31 を指定します。また最初のコマンドは、後で EP9 を開始する各タスクと、CICS 終了時の EP9 の呼び出しのための 750 バイトのローカル作業域の割り振りもスケジュールします。

2 番目のコマンドは、出口プログラムを開始します。つまり、出口プログラムの入り口点を開始できる状態にします。

### 特定の呼び出しタイプの使用可能化

**EXEC CICS ENABLE** コマンドの以下のオプションを使用すると、INDOUBTWAIT、SHUTDOWN、および SPI という特定のイベント時にユーザー出口プログラムを開始させることができます。

#### INDOUBTWAIT

フェーズ 2 同期点の時点で、CICS において UOW の結果が未確定である場合に、強制的な定義の UERTCOMM (コミット) または UERTBACK (バックアウト) ではなく、UERTWAIT verb (待機) によって出口プログラムが開始されるように指定します。UERTWAIT は、CICS で UOW の結果が既知でないことを表します。UERTWAIT 呼び出しに応じて、タスク関連ユーザー出口はリソース・マネージャーを開始して、スレッドなどのタスク関連リソースを解放する必要があります。ただしリソース・マネージャーは、UOW によって保留されているロックを維持し、UOW が未確定であることを記録する必要があります。

CICS がコーディネーターから UOW の結果を受け取ると、再同期タスクがアタッチされ、UOW の結果についてタスク関連ユーザー出口に通知されます。

CICS において UOW の結果が未確定であり、その UOW のために外部リソース・マネージャーが再同期を要求している場合 (**EXEC CICS RESYNC** コマンドの使用による)、CICS は未確定状態が解決されるまで待機してから、再同期タスクを開始します。

INDOUBT キーワードを使用してタスク関連ユーザー出口ルーチンを使用可能にしない場合、次のような影響があります。

- CICS において UOW が未確定である場合、強制的な決定が選択され、タスク関連ユーザー出口がその強制的な決定によって開始されます。
- CICS は、タスク関連ユーザー出口が INDOUBTWAIT によって有効にされていないために決定を強制される場合、UOW によって更新されるすべてのリソースに対して、強制的な決定を選択します。たとえ他のすべてのリソースが未確定状態の解決のために待機可能であっても、そのようにします。これは、ファイルなどのローカル・リソースに当てはまり、他のシステムへの LU6.1、LU6.2、または MRO 接続など、別の RMC にも当てはまります。
- CICS で未確定だった UOW の再同期を要求する、リソース・マネージャーからのインバウンド RESYNC コマンドが発行されると、CICS は強制的な決定によってタスク関連ユーザー出口を開始します。

#### SHUTDOWN

CICS のシャットダウン時に出口プログラムが開始されるように指定します。

#### SPI

CONNECTST または QUALIFIER オプションを指定した **EXEC CICS INQUIRE EXITPROGRAM** 呼び出しに対応するために出口プログラムが開始されるように指定します。このオプションは、出口プログラムがそのリソース・マネージャーに接続されているかどうか、その入り口名の修飾子が何かをユーザー・プログラムが識別できるようにするために使用します。

注：出口プログラムは、スケジュール・フラグ・ワードで UEFMSPi ビット・マスクを設定することで、このオプションを動的に設定できます。

### 管理ルーチン

タスク関連ユーザー出口プログラムは、使用する前に有効にするだけでなく、使用し終えたら無効にする必要があります。

**EXEC CICS ENABLE** コマンドおよび **DISABLE** コマンドを使用してタスク関連ユーザー出口プログラムを有効および無効にするプロシージャ (管理ルーチン) と、セッション間の再同期またはシステム障害後の再同期を行うプロシージャを準備する必要があります。有効化ルーチンは、PLT 初期設定プログラムで

もオンライン・プログラムでもかまいません。無効化ルーチンは、例えば、CICS の終了時に呼び出される TRUE によって開始することができます。

EXTRACT EXIT コマンドを使用すると、指定したタスク関連ユーザー出口プログラムが所有または共用するグローバル作業域のアドレスおよび長さを取得できます。

これらのシステム・コマンド、それらのコマンドを制限する規則、再同期に関するプログラミング情報については、[Introduction to System programming commands](#) を参照してください。

### タスク関連ユーザー出口プログラムのトレース

CICS は、制御権をタスク関連ユーザー出口に渡す直前、および出口から戻った直後に、トレース・エントリーを発行します。これらのトレース・エントリーは、CETR トレース制御トランザクションの RI オプション、または EXEC CICS SET TRACETYPE コマンドを使用して制御できます。

## アダプター追跡サンプル・タスク関連ユーザー出口プログラム (DFH\$APDT)

DFH\$APDT はサンプル・タスク関連ユーザー出口 (TRUE) プログラムであり、トランザクション・トラッキングに使用できるアダプター・データ・フィールドが含まれています。

サンプル出口プログラム DFH\$APDT は、ソース・コードとオブジェクト・コードの両方で提供されています。ソースは hlq.SDFHSAMP サンプル・ライブラリー内にあり、実行可能形式は hlq.SDFHLOAD ロード・ライブラリー内にあります。これらのサンプル・プログラムを実稼働環境で使用するには、その前に調整しなければなりません。

DFH\$APDT サンプル TRUE プログラムは、以下の 2 つの方法で使用できます。

- TASKSTART で使用できます。例えば、コマンド **EXEC CICS ENABLE PROGRAM(DFH\$APDT) TASKSTART START** を使用して、各タスクの始めと終わりに呼び出せます。この出口は各タスクの対象をコンテキスト管理で設定し、以降のタスクで START コマンドが発行されるたびに呼び出されます。
- **EXEC CICS ENABLE PROGRAM(DFH\$APDT) START** コマンドを使用して、有効にして開始できます。その後、DFHRMCAL 要求を使用して、アプリケーション・プログラムでこの出口を開始できます。出口の対象をコンテキスト管理に設定すると、それ以降、アプリケーション・プログラムから START コマンドが発行されるたびに、その出口が呼び出されます。

注：アダプター・データが既に設定されている場合、DFH\$APDT サンプルがそのデータを変更することはありません。

トランザクションのコンテキスト管理で対象を設定すると、そのトランザクションで発行される、以降のすべての START 要求で、出口が呼び出されてアダプターのデータ・フィールドが設定されます。DFH\$APDT サンプルではこれらのフィールドは定数から設定されますが、実際のアダプターの場合はこれらのフィールドの内容はコンテキストに基づきます。CICS は、これらのフィールドの内容を使用して、開始されるタスクの関連データの発信元データ・セクション内のアダプター・データ・フィールドにデータを入力します。その後、この関連データを、トランザクションの追跡に使用できます。

## ユーザー出口プログラミング・インターフェース(XPI)

ユーザー出口プログラミング・インターフェース (XPI) を使用すると、グローバル・ユーザー出口プログラムから CICS サービスを利用できます。

### XPI の概要

ユーザー出口プログラミング・インターフェース (XPI) には、一部の CICS サービスにアクセスできるグローバル・ユーザー出口プログラムが用意されています。ユーザー出口プログラムで使用できる一連のマクロ機能呼び出しも組み込まれています。

XPI を使用すれば、標準的な CICS システムに用意されている機能の範囲を超えて CICS の機能の拡張できますが、使用の際には注意が必要です。このインターフェースを使用した出口プログラムは、以下に示すガイドラインに沿って作成する必要があり、システム・エラーの原因にならないよう慎重にテストしなければなりません。

ユーザー出口プログラムは、アセンブラ言語で作成する必要があります。XPI は他の言語には対応していません。XPI 呼び出しを組み込むプログラムは、31 ビットの標準に合わせて作成しなければならず、再入可能でなければなりません。

XPI を開始する時には、1 次スペース変換モードを使用する必要があります。変換モードの詳細については、[z/Architecture 解説書](#)を参照してください。

XPI 機能については、[XPI 機能 \(ドメイン別\)](#)で一覧が示され、詳しく取り上げられています。

### 重要:

1. すべてのユーザー出口点ですべての XPI 呼び出しを使用できるわけではありません。呼び出しを使用できないタイミングについては、各機能呼び出しの説明や、[グローバル・ユーザー出口プログラム](#)にある出口点のリストを参照してください。  
  
XPI 呼び出しを使用して CICS サービスを開始する時に、間違った出口で使用すると、CICS システムで予測不能なエラーが発生します。
2. 初期化の処理の早い段階で XPI を使用することについては制限があります。  
INQUIRE\_MONITOR\_DATA、MONITOR、TRANSACTION\_DUMP、WRITE\_JOURNAL\_DATA の各 XPI 機能を使用する出口プログラムは、PLTPI の第 2 フェーズまで開始しないでください。PLTPI の詳細については、[初期設定プログラムおよびシャットダウン・プログラムの作成](#)を参照してください。
3. XPI 機能を使用すると、ユーザー出口プログラムを実行するタスクが XPI 機能の実行中に別のタスクに対する制御を失う可能性があります。CICS 機能の流れが中断すると、ロックアウトなどの問題が発生する場合があるので、XPI 機能の使用には細心の注意が必要です。
4. グローバル・ユーザー出口またはタスク関連ユーザー出口が、ある CICS リリースの CICS ライブラリーを使用してアセンブルされ、異なる CICS リリースを実行するシステムに対して XPI 呼び出しを行うことがあります。この状況で、その XPI 呼び出しを処理する正しい CICS モジュールに出口から制御が正常に移されるかどうかは、CICS リリースの組み合わせと、その XPI 呼び出しがリリースを区別する呼び出しであるかどうかによります。ユーザー出口が正常に機能するためには、XPI パラメーターがリリース間で変更されているかどうかなど、その他の要素も確認する必要があります。詳しくは、[アップグレード](#)を参照してください。

## XPI 呼び出しの実行

XPI 呼び出しには、パラメーターのセットが 2 つがあります。1 つは入力パラメーターで、XPI 機能呼び出しと、その呼び出しに渡すパラメーターが含まれます。もう 1 つは出力パラメーターで、CICS は、出力パラメーターを使用して、呼び出しが成功したかどうかを示す応答コードや理由コードなどの値を返します。

XPI マクロ呼び出しを使用するには、入力パラメーターと出力パラメーターを定義したコピーブックを組み込まなければなりません。マクロの名前は DFHxxyyX という形式でなければならない、関連するコピーブックの名前は DFHxxyyY という形式です。例えば、ストレージ管理 XPI には GETMAIN 呼び出しが含まれています。使用するマクロは DFHSMMCX、関連するコピーブックは DFHSMMCY です。

すべての XPI 呼び出しの一般的な形式は以下のとおりです (アセンブラー言語の継続文字は省略しています)。

```
macro-name [CALL],
           [CLEAR],
           [IN,
            FUNCTION(call_name),
            mandin1(value),
            mandin2(value),
            ...
            [optin1(value),]
            [optin2(value),]
            ...]
           [OUT,
            mandout1(value),
            mandout2(value),
            ...
            [optout1(value),]
            [optout2(value),]
            ...]
           RESPONSE,
           REASON]
```

XPI 呼び出しは、アセンブラー言語の以下のコーディング規則に沿って記述しなければなりません。

- macro-name は、16 桁目の前から開始する必要があります。



- 継続行は、16 桁目から開始する必要があります。
- macro-name と最初のキーワード (通常は CALL) の間の空白以外に空白を埋め込むことはできません。
- 最終行以外の行の項目は、末尾にコンマを付ける必要があります。
- 最終行以外の行は、72 桁目に継続文字を入れる必要があります。
- 入力値と出力値を括弧で囲む必要があります。入力値や出力値としてレジスター参照を使用する場合は、レジスター参照をさらに括弧で囲み、((R6)) のようにしなければなりません。
- XPI オプションの値を設定する方法の詳細については、59 ページの『XPI の構文』を参照してください。

XPI 機能には 3 種類の用途があります。以下のことを行うことができます。

- XPI 呼び出しで使用したパラメーター・リストをクリアします。
- 入力パラメーターをセットアップします。
- CICS 機能を呼び出します。

そのすべてを個別にコーディングすることもできれば (58 ページの『増分方式でパラメーター・リストを作成する例』を参照)、1 つのステートメントにすべてを組み込むこともできます。

XPI のすべての用途に共通するオプションもあります。すべての構文記述にそれらのオプションが含まれていますが、説明はここにまとめられています。CALL、CLEAR、IN、FUNCTION、OUT、RESPONSE、REASON の各オプションがそれに当たります。

## CALL

XPI 機能呼び出しを実行するコードを生成します。CALL、IN、FUNCTION、OUT を指定すると、パラメーター・リストを作成し、機能呼び出し、結果を受け取るという操作全体を実行するコードが生成されます。CALL を省略し、IN を指定してパラメーター・リストを増分方式で作成することもできます。後から CALL でそのリストを指定し、CALL、IN、FUNCTION、OUT とすべての必須指定オプションをコーディングできます。その場合は、事前設定オプションの値を示すためにアスタリスク (\*) を使用します。アスタリスクが付いている値は、すでにリストで設定されていることを示します。

**注:** 増分方式でパラメーター・リストを作成する場合は、呼び出しを最終的に実行する時に CLEAR を指定しないでください。CLEAR オプションを使用すると、パラメーター・リストがゼロに設定されるので、事前設定値が失われてしまいます。

## CLEAR

パラメーター・リスト (必須パラメーターとオプション・パラメーターの両方) の存在ビットを 2 進ゼロに設定します。各マクロには COPY コードがあります。COPY コードでは、DSECT によってパラメーター・リストを定義します。パラメーター・リストは、ヘッダー・セクションと、その後に一連の存在ビットが続き、そしてパラメーター本体という構造になっています。パフォーマンス上の理由から、ヘッダー・セクションと存在ビットだけがクリアされます。パラメーター・リストのその他の部分は変わりません。

**注:** パラメーター・リストをクリアしないと、プログラム・チェックやストレージ保護違反などの予測不能な結果になることがあります。増分方式でパラメーター・リストを作成する場合は、パラメーターを指定する前に CLEAR を指定してください。パラメーターの作成を増分方式にしない場合は、CALL を実行する時に CLEAR を指定してください。

## IN

CICS に対して、IN オプションと OUT オプションの間にあるすべてのパラメーターが入力値であることを示します。CALL を指定した場合、このオプションの指定は**必須**になります。CALL なしで機能を使用してパラメーター・リストを作成する場合は、IN といくつかのパラメーター値を指定して値をリストに保管できます。

## FUNCTION

必要なマクロの機能 (GETMAIN、FREEMAIN など) を指定します。CALL を指定した場合、このオプションの指定は**必須**になります。ただし、他のオプションとは異なり、明示的に指定しなければなりません。FUNCTION(\*) というコーディングは**無効**です。

## mandin(value)

mandin は、CALL を指定した場合に必須になるオプションです。value は、アスタリスク (\*) にすることもできます。その場合は、マクロの前回の使用時にパラメーター・リストの値がすでに設定されてい



る、という意味になります(「CALL」を参照)。value の指定方法の詳細については、59 ページの『XPI の構文』にあるそれぞれの機能呼び出しの説明を参照してください。

## OUT

CICS に対して、OUT オプションの後のすべてのパラメーターが受信側のフィールドになることを示します。CALL を指定した場合、このオプションの指定は**必須**になります。

注: CALL を指定しない場合に、以下の出力パラメーターでアスタリスク (\*) 以外の値を使用しても、その値は無効です。

## mandout(value)

mandout は、CALL を指定した場合に必須になるオプションです。アスタリスク (\*) をコーディングした場合は、出力がパラメーター・リストに組み込まれます。value で場所を指定した場合は、その場所に配置されます。RESPONSE は mandout オプションの特殊ケースです(「RESPONSE」を参照)。value の指定方法の詳細については、個々の機能呼び出しの説明を参照してください(59 ページの『XPI の構文』を参照)。

## optin1,2... optout1,2...

この項目は、どの形式のマクロでも完全に任意指定になりますが、特に CALL を指定した場合は、使用する必要がなくなります。

## RESPONSE

XPI 呼び出しから応答を受け取るために定義する必須のデータ域です。アスタリスク (\*) を使用して、CICS に対し、RESPONSE 値をパラメーター・リストに組み込むように指示できます。あるいは、RESPONSE 値を配置するフィールドの名前を指定することもできます。CALL なしでマクロを使用してパラメーター・リストを作成する場合は、RESPONSE オプションのコーディングが不要になります。

XPI 呼び出しからの応答は、OK、EXCEPTION、DISASTER、INVALID、KERNERROR、PURGED のいずれかです。CICS に用意されている応答コード値には、以下のような標準化された名前(EQU シンボル)があります。

```
xxxxy_OK, xxxx EXCEPTION, xxxx DISASTER, xxxx_INVALID,  
xxxxy_KERNERROR, and xxxx PURGED,
```

xxxxy は、関連する macro-name の DFH というストリングの後にある 4 文字から派生した接頭部です。DFHSMCMX であれば、接頭部は SMMC になり、DFHLDLDX であれば、LDLD になります。マクロ DFHxxxxY のコピーブック DFHxxxxY を組み込んだ時に、そのような名前が生成されるので、それぞれの名前から等価の値を判別してください。どのマクロ呼び出しでも、対応する RESPONSE コードの算術値が同じになるとは限りません。RESPONSE コードの意味は以下のとおりです。

## OK

この XPI 要求は正常に完了しました。

## EXCEPTION

この機能は正常に完了しませんでした。原因は予期されるもので、それがプログラムによってコーディングされることもあります(TRANSACTION\_DUMP、EXCEPTION = SUPPRESSED\_BY\_DUMPTABLE など)。REASON 値で詳細を確認できる場合もあります。

## DISASTER

この要求は完全に失敗しました。ユーザー出口プログラムでこの障害からリカバリーすることはできません。この障害が発生すると、CICS は、システム・ダンプを生成し、エラー・メッセージを発行し、DISASTER 応答を設定します。ユーザー出口プログラムは、この応答を受け取ると、その以上の処理を試みずに終了します。この応答の REASON 値はトレースだけに表示されます。REASON 値で詳細情報を確認できる場合もあります。REASON 値が呼び出し側プログラムに返されることはありません。

## INVALID

必須の値が指定されていないか、オプションの値に無効値が指定されています。ユーザー出口プログラムでこの障害からリカバリーすることはできません。この障害が発生すると、CICS は、システム・ダンプを生成し、エラー・メッセージを発行し、INVALID 応答を設定します。ユーザー出口プログラムは、この応答を受け取ると、その以上の処理を試みずに呼び出し側に戻ります。この応答の REASON 値はトレースだけに表示されます。REASON 値で詳細情報を確認できる場合もあります。その情報が出口プログラムのエラーの修正に役立つこともあります。REASON 値が呼び出し側プログラムに返されることはありません。

## KERNERROR

呼び出そうとした CICS 機能のエラーをカーネルが検出しました。要求した機能が使用不可になっていたり無効になっていたりする場合や、CICS 内にエラーがある場合があります。

## PURGED

タスクがページされたか、XPI 呼び出しで指定した間隔の期限が切れました。REASON コードを調べてください。

出口プログラムは、DFHDSSRX SUSPEND や WAIT\_MVS 以外の XPI 呼び出しでこの RESPONSE を受け取ると、戻りコードを UERCPURG に設定して呼び出し側に戻ります。

DFHDSSRX SUSPEND 呼び出しや WAIT\_MVS 呼び出しで INTERVAL を指定した時に、TIMED\_OUT という REASON でこの RESPONSE を受け取った場合は、指定した INTERVAL の時間が経過したという意味になります。次にどうするかは、ユーザー次第です。

DFHDSSRX SUSPEND 呼び出しや WAIT\_MVS 呼び出しで INTERVAL を指定した時に、TASK\_CANCELLED という REASON でこの RESPONSE を受け取った場合は、指定した INTERVAL の時間は経過していないものの、オペレーターやアプリケーションによってタスクがページされたという意味になります。その場合は、ユーザーが戻りコードを UERCPURG に設定して戻らなければなりません。

DFHDSSRX SUSPEND 呼び出しや WAIT\_MVS 呼び出しで INTERVAL を指定しなかった時に、TASK\_CANCELLED や TIMED\_OUT という REASON でこの RESPONSE を受け取った場合は、オペレーターやアプリケーションやデッドロック・タイムアウト機能によってタスクがページされたという意味になります。その場合は、ユーザーが戻りコードを UERCPURG に設定して戻らなければなりません。

その他の理由の場合に、応答コード UERCPURG を CICS に返すことは**絶対にしないでください**。そのようなことをすると、プログラムが予測不能な結果になります。

## REASON

RESPONSE 値の詳細情報を受け取るために定義する必須のデータ域です。(\*)を使用すると、CICS にとっては、REASON 値をパラメーター・リストに組み込まなければならない、という意味になります。ほとんどの XPI 呼び出しでは、EXCEPTION と PURGED という RESPONSE 値の場合に限って、標準化された理由の名前 (EQU シンボル) が用意されています。応答に伴う REASON 値は XPI 機能ごとに異なるため、詳細については、それぞれの XPI 呼び出しの説明を参照してください。

RESPONSE が OK の場合、REASON は該当しません。そのような状況で REASON フィールドをテストしないでください。

**注:** パラメーター・リストの初期化、パラメーターのセットアップ、呼び出しの実行、出力パラメーターの受け取りの方法を示した例については、[54 ページの『グローバル・ユーザー出口 XPI の例 \(ストレージの使用法\)』](#)を参照してください。そのセクションには、操作全体を示した例と個々の手順を別々に実行する例の両方があります。

## XPI 環境のセットアップ

出口プログラミング・インターフェース (XPI) には通常の CICS トランザクション環境は不要ですが、XPI 呼び出しを使用する前に、特別な出口プログラミング環境をセットアップする必要があります。

出口プログラムで XPI 機能を使用する場合は、プログラム中に次のマクロを (XPI 呼び出しを発行する前に) 指定する必要があります。

```
DFHUEXIT TYPE=XPIENV
```

このマクロが展開されると、すべての XPI 呼び出しで使用される DSECT が生成されます。また、出口プログラムで使用するレジスタの等式のリスト (R0 EQU 0、R1 EQU 1、と続く) も生成されます。このマクロで生成されるその他のフィールドは、XPI 呼び出し処理を実行するために CICS が使用します。これらのフィールドを使用しないでください。使用すると、ユーザー出口プログラムの結果が予測不能になります。

ユーザー出口プログラムは 31 ビット・アドレッシング・モードでなければなりません。

## XPI レジスタの使用法

グローバル・ユーザー出口プログラムから XPI 呼び出しを発行する前に、DFHUEPAR のパラメーター UEPSTACK (カーネル・スタック・エントリ) の内容を、出口プログラムのレジスタ 13 に移動しなければなりません。

XPI 機能展開では、レジスタ 0、1、14、15 が使用されます。したがって、出口プログラムは必要に応じて、XPI 呼び出しの前後でそれらのレジスタを保管したり復元したりしなければなりません。

同じ出口プログラムで EXEC CICS コマンドと XPI 呼び出しを使用する例については、[グローバル・ユーザー出口サンプル・プログラム DFH\\$XTSE](#) を参照してください。

## XPI コピーブック

XPI 機能ごとに、その機能に関連する DSECT を提供するコピーブックがあります。これらの DSECT を使用して、XPI 呼び出しのパラメーター、応答コード、理由コードをマップできます。

使用する各 XPI 機能の COPY ステートメントを出口プログラムに含める必要があります。コピーブック名は、最後の文字「X」が文字「Y」になる点を除き、マクロ名と同じです。

例えば、XPI 機能 DFHSMCMX のコピーブックを含めるには、以下のステートメントを含める必要があります。

```
COPY DFHSMCMY
```

使用する機能に対してトレースをオンにした場合は、XPI 呼び出しのトレース・エントリにこれらのパラメーター・リストが示されます。

## XPI 呼び出しに関係する再入可能性についての考慮事項

XPI の呼び出しでは、CICS が、XPI 呼び出しの処理中に別のタスクに制御権を与えることがあります。この 2 番目のタスクが、同じ出口プログラムを呼び出して、同じ XPI 呼び出しを (大抵は別のパラメーター値を使用して) 行うことがあります。このような状況でロックアウト状態が発生しないようにする必要があります。

XPI 呼び出しの処理中に、CICS が、同じユーザー出口プログラムを使用する別のユーザー出口点を検出することがあります。したがって、XPI パラメーター・リストは、出口プログラムの単一の呼び出しに関連付けられたストレージ内に作成しなければなりません。

出口プログラムがグローバル・ユーザー出口である場合、CICS はそのプログラムに 1024 バイトの LIFO ストレージ (出口プログラムの単一の呼び出しのために排他的に使用されます) を提供します。出口プログラムは、DFHUEPAR パラメーター・リストのパラメーター UEPXSTOR を使用して、このストレージにアクセスできます。XPI パラメーター・リストを作成する際には、DFHxyyY コピーブックで提供される DSECT に基づいてこのストレージを使用してください。このようにすると、出口プログラムに再入しても、パラメーターは壊れません。

このようにして提供された XPI サービス用のパラメーター・リストは、256 バイト以下です。UEPXSTOR ストレージの残りの 768 バイトは、出口プログラムのためだけに使用できます。この 768 バイトの予備のストレージがあるため、ほとんどの場合は、出口プログラムで追加のストレージを取得する必要はありません。提供される予備の 768 バイトよりも多くのストレージが必要な場合は、DFHSMCMX FUNCTION (GETMAIN) マクロか MVS GETMAIN 要求のいずれかを使用して取得してください。

出口プログラムの呼び出しをまたいで保持する情報は、出口プログラム (または出口プログラムのグループ) 用に定義できるグローバル作業域に保管できます。1024 バイトの LIFO ストレージは動的であるため、この目的には使用できません。

## リリースを区別する XPI 呼び出し

リリース依存の XPI 呼び出しを使用して、現在サポートされているすべての CICS リリースで XPI 呼び出しを正常に実行できるようにすることができます。これを行うには、**CALL XPI** パラメーターを **RELENSCALL** XPI パラメーターに置き換え、プログラムをアセンブルします。リリース依存の XPI 呼び出しをすべての XPI コマンドの代替として使用することができます。

**RELENSCALL** パラメーターは、その機能をサポートする CICS XPI モジュールの呼び出しが成功することのみを保証します。XPI 呼び出しに指定したパラメーターが、その呼び出しが行われる CICS リリースのすべてで有効であることを確認する必要があります。異なる呼び出しパラメーターを使用すること以外は、

他のすべての XPI 構文規則が適用されます。構文規則について詳しくは、[59 ページの『XPI の構文』](#)を参照してください。

以下の例は、**RELENSCALL** パラメーターを使用する XPI GETMAIN 呼び出しです。

```
DFHSMCXX RELENSCALL,      -
    CLEAR,                  -
    IN,                      -
    FUNCTION(GETMAIN),      -
    GET_LENGTH((r6)),        -
    SUSPEND(YES),           -
    STORAGE_CLASS(USER),    -
    OUT,                     -
    ADDRESS((r5)),           -
    RESPONSE(*),            -
    REASON(*)
```

以下の例は、**CALL** パラメーターを使用する同じ XPI GETMAIN 呼び出しです。

```
DFHSMCXX CALL,            -
    CLEAR,                  -
    IN,                      -
    FUNCTION(GETMAIN),      -
    GET_LENGTH((r6)),        -
    SUSPEND(YES),           -
    STORAGE_CLASS(USER),    -
    OUT,                     -
    ADDRESS((r5)),           -
    RESPONSE(*),            -
    REASON(*)
```

さまざまな CICS リリースとリリース依存呼び出しがユーザー出口に与える影響について詳しくは、[XPI の変更点](#)を参照してください。

## グローバル・ユーザー出口 XPI の例 (ストレージの使用法)

グローバル・ユーザー出口プログラムで XPI とストレージを使用する例を以下に示します。グローバル・ユーザー出口プログラムの入り口コード/出口コードと XPI 機能の使用例にすぎず、完全なプログラムではありません。

この例で使用している DFHSMCXX マクロのオプションの説明については、[ストレージ管理 XPI 関数](#)を参照してください。

この例では、XPI GETMAIN 呼び出しを使用してこのプログラム呼び出しのためのストレージを取得してから、UEPXSTOR でアドレッシングする LIFO ストレージの最初の 4 バイトにそのストレージのアドレスを保管する、という手法を使用しています。この例では、CLEAR オプションによるパラメーター・リストの初期化、パラメーター・リストの作成、GETMAIN の呼び出しを 1 つのマクロで実行します。増分方式でパラメーター・リストを作成する方法の詳細や、CLEAR 呼び出しと GETMAIN 呼び出しを分離する方法の詳細については、[58 ページの『増分方式でパラメーター・リストを作成する例』](#)を参照してください。

```

TITLE 'GUEXPI - GLOBAL USER EXIT PROGRAM WITH XPI'
*****
* The first three instructions set up the global user exit *
* environment, identify the user exit point, prepare for the use of *
* the exit programming interface, and copy in the definitions that *
* are to be used by the XPI function. *
*****
*
*       DFHUEXIT TYPE=EP,ID=XFCREQ       PROVIDE DFHUEPAR PARAMETER
*                                         LIST FOR XFCREQ IN THE FILE
*                                         CONTROL PROGRAM AND LIST
*                                         OF EXITID EQUATES
*
*       DFHUEXIT TYPE=XPIENV             SET UP ENVIRONMENT FOR
*                                         EXIT PROGRAMMING INTERFACE --
*                                         MUST BE ISSUED BEFORE ANY
*                                         XPI MACROS ARE ISSUED
*
*       COPY  DFHSMCMCY                  DEFINE PARAMETER LIST FOR
*                                         USE BY DFHSMCMCX MACRO
*
*****
* The following DSECT maps a storage area you can use to make the *
* exit program reentrant by storing the address of the storage you *
* acquire in the first four bytes of the 260-byte area provided by *
* the user exit handler (DFHUEH) and addressed by UEPXSTOR. *
*****
*
TRANSTOR DSECT                          DSECT FOR STORAGE OBTAINED BY
*                                         GETMAIN
*
.
.
.
storage declarations
.
.
.
*
*****
* The next seven instructions form the normal start of a global user *
* exit program, setting the program addressing mode to 31-bit, saving *
* the calling program's registers, establishing base addressing, and *
* establishing the addressing of the user exit parameter list. *
*****
*
GXPI      CSECT
GXPI      AMODE 31                      SET TO 31-BIT ADDRESSING
*
*       SAVE (14,12)                    SAVE CALLING PROGRAM'S REGISTERS
*
*       LR    R11,R15                   SET UP USER EXIT PROGRAM'S
*       USING GXPI,R11                  BASE REGISTER
*
*       LR    R2,R1                     SET UP ADDRESSING FOR USER
*       USING DFHUEPAR,R2              EXIT PARAMETER LIST -- USE
*                                     REGISTER 2 AS XPI CALLS USE
*                                     REGISTER 1
*

```

図 9. XPI を使用したグローバル・ユーザー出口プログラム (パート 1)

```

*****
* Before issuing an XPI function call, set up addressing to XPI      *
* parameter list.                                                    *
*****
*
*       L      R5,UEPXSTOR      SET UP ADDRESSING FOR XPI
*                                     PARAMETER LIST
*
*       USING DFHSMC_ARG,R5      MAP PARAMETER LIST
*
*****
* Before issuing an XPI function call, you must ensure that register *
* 13 addresses the kernel stack.                                     *
*****
*
*       L      R13,UEPSTACK      ADDRESS KERNEL STACK
*
*****
* Issue the DFHSMCX macro call, specifying:                          *
*
* CALL --      the macro is to be called immediately                *
*
* CLEAR --     initialize the parameter list before inserting values. *
*
* IN --        input values follow.                                  *
*
*              FUNCTION(GETMAIN) -- acquire storage                 *
*              GET_LENGTH(120) -- 120 bytes of it                   *
*              SUSPEND(NO) -- don't suspend if storage not available *
*              INITIAL_IMAGE(X'00') -- clear acquired storage        *
*                                     to hex zero throughout.        *
*              STORAGE_CLASS(USER) -- class of storage to be        *
*                                     acquired is user storage        *
*                                     above the 16MB line.             *
*
* OUT --       output values follow                                  *
*
*              ADDRESS((R6)) -- put address of acquired storage in   *
*                                     register 6.                     *
*              RESPONSE(*) -- put response at SMMC_RESPONSE in      *
*                                     macro parameter list.           *
*              REASON(*) -- put reason at SMMC_REASON in macro      *
*                                     parameter list.                 *
*****
*
*       DFHSMCX CALL,
*           CLEAR,
*           IN,
*           FUNCTION(GETMAIN),
*           GET_LENGTH(120),
*           SUSPEND(NO),
*           INITIAL_IMAGE(X'00'),
*           STORAGE_CLASS(USER),
*           OUT,
*           ADDRESS((R6)),
*           RESPONSE(*),
*           REASON(*)
*

```

図 10. XPI を使用したグローバル・ユーザー出口プログラム (パート 2)



```

*****
* Test SMMC_RESPONSE -- if OK, then branch round error handling.      *
*****
*
*      CLI    SMMC_RESPONSE,SMMC_OK    CHECK RESPONSE AND...
*      BE     STOK                     ...IF OK, BYPASS ERROR ROUTINES
*
*      .
*      .
*      error-handling routines
*      .
*      .
*****
* The next section maps TRANSTOR on the acquired storage.              *
*****
STOK    DS     0H
        USING TRANSTOR,R6          MAP ACQUIRED STORAGE
        ST     R6,0(R5)            SAVE STORAGE ADDRESS IN FIRST
*                                     4 BYTES OF STORAGE ADDRESSED
*                                     BY UEPXSTOR
*
*      LA     R5,4(R5)             ADDRESS 4-BYTE OFFSET
*      DROP   R5                   REUSE REGISTER 5 TO BASE ALL
        USING DFHxxyy_ARG,R5      FOLLOWING XPI PARAMETER LISTS
*                                     AT 4-BYTE OFFSET INTO STORAGE
*                                     ADDRESSED BY UEPXSTOR
*
*      .
*      .
rest of user exit program
*      .
*      .
*
*****
* When the rest of the exit program is completed, free the storage
* and return.
*****
*
*      DROP   R5                   REUSE REGISTER 5 TO MAP DFHSMCX
*      USING  DFHSMCX_ARG,R5      XPI PARAMETER LIST
*
*      L      R13,UEPSTACK        ADDRESS KERNEL STACK
*
*****
* Issue the DFHSMCX macro call, specifying:
*
*      CALL --      the macro is to be called immediately.
*
*      CLEAR --     initialize the parameter list before inserting values.
*
*      IN  --       input values follow.
*
*      FUNCTION(FREEMAIN) -- release storage
*      ADDRESS((R6)) -- address of storage is in register 6.
*      STORAGE_CLASS(USER) -- class of acquired storage was
*                           31-bit user storage.
*

```

図 11. XPI を使用したグローバル・ユーザー出口プログラム (パート 3)

```

*   OUT --          output values follow                                *
*                                                           *
*           RESPONSE(*) -- put response at SMMC_RESPONSE in      *
*                           macro parameter list.                *
*           REASON(*) -- put reason at SMMC_REASON in macro      *
*                           parameter list.                        *
*                                                           *
*****
*
*           DFHSMMCX CALL,                                         +
*                           CLEAR,                                 +
*                           IN,                                   +
*                           FUNCTION(FREEMAIN),                   +
*                           ADDRESS((R6)),                        +
*                           STORAGE_CLASS(USER),                 +
*                           OUT,                                  +
*                           RESPONSE(*),                          +
*                           REASON(*)                             +
*                                                           *
*****
* Test SMMC_RESPONSE -- if OK, then branch round error handling. *
*****
*
*           CLI   SMMC_RESPONSE,SMMC_OK   CHECK RESPONSE AND...
*           BE    STEND                   ...IF OK, BYPASS ERROR ROUTINES
*                                                           *
*
*           .
*           .
*           .
*           error-handling routines
*           .
*           .
*           .
*
*****
* Restore registers, set return code, and return to user exit handler *
*****
*
STEND    DS    0H
        L      R13,UEPEPSA
        RETURN (14,12),RC=UERCNORM
        LTORG
        END    GXPI

```

図 12. XPI を使用したグローバル・ユーザー出口プログラム (パート 4)

### 増分方式でパラメーター・リストを作成する例

増分方式でパラメーター・リストを作成する例を以下に示します。CLEAR オプションによるパラメーター・リストの初期化、パラメーター・リストの作成、GETMAIN の呼び出しを別々のステップで実行します。

```

:           DFHSMMCX CLEAR
:
:           DFHSMMCX GET_LENGTH(100)
:
:           DFHSMMCX CALL,                                         *
:                           IN,                                   *
:                           FUNCTION(GETMAIN),                   *
:                           GET_LENGTH(*),                       *
:                           SUSPEND(NO),                         *
:                           INITIAL_IMAGE(X'00'),               *
:                           STORAGE_CLASS(USER),                 *
:                           OUT,                                  *
:                           ADDRESS((R6)),                       *
:                           RESPONSE(*),                         *
:                           REASON(*)                             *

```

### 重要

XPI 機能のみを使用してパラメーターを設定する必要があります。

## XPI の構文

XPI 機能では、特殊な構文を使用します。各機能の説明では、その機能の呼び出しに固有のオプションだけを定義しています。

すべての機能呼び出しに当てはまるオプションについては、[49 ページの『XPI 呼び出しの実行』](#)で説明されています。以下のタイプの引数を使用します。

### **name1, name2,...**

それぞれ、所定のサイズ (バイト数) のフィールドの名前を指しています。name1 は、1 バイトのフィールドの名前を指定する、という意味です。

### **literalconst**

リテラル形式の数値です。例えば、B'00000000'、X'FF'、X'FCF4'、"0" などがあります。また、そのような値に等価のシンボルを付けた形式も可能です。

### **expression**

アセンブラー言語の有効な式。10 進整数か、アセンブラー言語で有効な演算式 (シンボル値も含む)。以下に例を示します。

```
20; L'AREA; L'AREA+10; L'AREA+X'22'; SYMB/3+20 .
```

### **(Rn)**

レジスター参照。引数を囲む括弧のほかに、さらに括弧が必要です。例えば、OPTION((R5)) のようになります。

### **block-descriptor**

データ・アドレスとデータ長の両方のフィールドのソースです。ブロック記述子では、シングル形式の値かダブル形式の値を使用できます。以下に示すのは、シングル形式の値です。

```
OPTION(blkdbname)
```

### **blkdbname**

ブロック記述子の名前。連続するフルワードのペアで、最初のワードにはデータのアドレスが入り、2 番目のワードにはデータの長さ (バイト数) が 1 つのフルワード・バイナリー値として入ります。このシングル形式の値では、レジスター表記を使用できません。

以下に示すのは、ダブル形式の値です。

```
OPTION(addr,len)
```

### **addr**

データ・アドレス。{namea | (Ra) | aliteral} という形式になります。

#### **namea**

データ・アドレスが含まれている場所の名前。

#### **(Ra)**

データ・アドレスが含まれているレジスター。

#### **aliteral**

アドレス定数リテラル。例えば、A(data) のようになります。

### **len**

データ長。{namel | (Rn) | expression} という形式になります。

#### **namel**

データ長 (バイト数) を指定したバイナリー・フルワードが含まれている場所の名前。

#### **(Rn)**

レジスター。このレジスターの内容として、フルワード・バイナリーでデータのバイト数を指定します。

## expression

10 進整数か、アセンブラー言語で有効な演算式 (シンボル値も含む)。以下に例を示します。

```
L'AREA ; L'AREA+10 ; L'AREA+X'22' ; SYMB/3+20 .
```

## buffer-descriptor

データ・アドレスと最大データ長の両方のフィールドのソースです。バッファー記述子の各部分は、出力情報の受信フィールドとしても予約されています。バッファー記述子では、シングル形式の値か複数形式の値を使用できます。以下に示すのは、シングル形式の値です。

```
OPTION(bufdname)
```

### bufdname

バッファー記述子の名前。最大で 4 つの連続したフルワードを 1 つのグループとしてまとめた名前であり、各フルワードがバッファー記述子のさまざまな構成要素に対応しています。各フィールドの解釈は以下のとおりです。

- 1 番目のワードには、データのアドレスが入ります (入力)。
- 2 番目のワードは、データの現在の長さ (バイト数) を 1 つのフルワード・バイナリー値として受け取るために予約されています (出力)。3 つの構成要素を指定する場合は、3 番目の構成要素がこのワードに対応します。バッファー記述子で構成要素を 2 つだけ指定する場合は、2 番目の構成要素がこのワードに対応します。
- 3 番目のワードには、データの最大長 (バイト数) が 1 つのフルワード・バイナリー値として入ります (入力)。3 つの構成要素を指定する場合は、2 番目の構成要素がこのワードに対応します。バッファー記述子で構成要素を 2 つだけ指定する場合は、このフィールドを使用しません。
- 4 番目のワードは、XPI 用に予約されています。

このシングル形式の値では、レジスター表記を使用できません。

以下に示すのは、複数形式の値です。

```
OPTION(addr,maxlen,*)
```

### addr

データ・アドレス。{namea | (Ra) | aliteral} という形式になります。

#### namea

データ・アドレスが含まれている場所の名前。

#### (Ra)

データ・アドレスが含まれているレジスター。

#### aliteral

アドレス定数リテラル。例えば、A(data) のようになります。

### maxlen

最大データ長。{namel | (Rn) | expression} という形式になります。

#### namel

最大データ長 (バイト数) を指定したバイナリー・フルワードが含まれている場所の名前。

#### (Rn)

レジスター。このレジスターの内容として、フルワード・バイナリーでデータの最大バイト数を指定します。

## expression

10 進整数か、アセンブラー言語で有効な演算式 (シンボル値も含む)。以下に例を示します。

```
L'AREA ; L'AREA+10 ; L'AREA+X'22' ; SYMB/3+20 .
```

★

パラメーター・リストを予約フィールドとして使用することを示す必須パラメーター。このパラメーターをコーディングした場合は、バッファー記述子で返される `_N` 構成要素から必須の値を取り込む必要があります。





## 第2章 初期設定プログラムとシャットダウン・プログラムを使用したカスタマイズ

CICS 処理の初期化フェーズおよびシャットダウン・フェーズ時に実行するプログラムを作成できます。

### 処理設定プログラムとシャットダウンプログラムの作成

CICS 処理の初期化フェーズおよびシャットダウン・フェーズ時に実行するプログラムを作成できます。これらの時に実行するすべてのプログラムは、プログラム・リスト・テーブル (PLT) の CICS に定義する必要があります。

PLT のコーディング方法については、[プログラム・リスト・テーブル \(PLT\)](#)を参照してください。

### 初期設定プログラムの作成

CICS の初期設定時に実行するすべてのプログラムをプログラム・リスト・テーブル (PLT) で指定し、その PLT の接尾部をプログラム・リスト・テーブル初期化後処理 (PLTPI) システム初期設定パラメーターで指定しなければなりません。

プログラム・リスト・テーブル (PLT) の実行には2つの段階があり、PLT 内の DFHDELIM ステートメントで各段階が区分されています。

#### 第1フェーズの PLT プログラム

CICS 初期設定処理の初期段階で実行できる PLT プログラムは、グローバル・ユーザー出口プログラムおよびタスク関連ユーザー出口プログラムを有効化するコマンドを含むプログラムのみです。このようなプログラムは、PLTPI リストの最初の部分 (DFHDELIM ステートメントの前) に指定されるため、リカバリー中に必要になる出口プログラムを有効にすることができます。

第1フェーズの PLT プログラムは、CICS 初期設定の第2段階で実行されるものであり、**EXEC CICS INQUIRE SYSTEM** コマンドまたは **XPI INQUIRE\_SYSTEM** 呼び出しで SECONDINIT として返されるフェーズです。

第1段階の PLT プログラムを実行してから第2段階の PLT プログラムを実行するまでの間に、動的 LIBRARY リソースがインストールされるか、復元されて再活動化されます。第1段階の PLT プログラムは DFHRPL のデータ・セットに含める必要がありますが、第2段階の PLT プログラムは動的 LIBRARY リソースに含めて、そこからロードすることができます。

以下の点がすべての第1フェーズの PLTPI プログラムに適用されます。

- プログラムは、アセンブラ言語で作成する必要があります。
- プログラムは、AMODE 31 または AMODE 64 を実行する必要があります。
- 含めることができる EXEC CICS コマンドは以下に限られます。
  - **ASSIGN APPLID**
  - **ASSIGN INITPARM**
  - **ENABLE**
  - **EXTRACT EXIT**

この段階は、リカバリー前の、初期設定が不完全なときに実行されるため、他の CICS サービスを呼び出すことはできません。

- 第1フェーズの PLTPI プログラムで、XPI 呼び出し INQUIRE\_APP\_CONTEXT、INQUIRE\_MONITORING\_DATA、MONITOR、TRANSACTION\_DUMP、または WRITE\_JOURNAL\_DATA のいずれかを発行する出口プログラムを有効にする場合、**EXEC CICS ENABLE COMMAND** に START オプションを指定してはいけません。
- 第1段階 PLTPI プログラムでは、TASKSTART オプションを使用して任意のタスク関連ユーザー出口プログラムを使用可能にしないでください。

- 第1段階 PLT プログラムは、CICS 初期設定の早い段階で実行されるため、リソース定義はすべて使用できません。インストール済みの PROGRAM 定義 (または、プログラム自動インストール・ユーザー・プログラム) を使用して、第1フェーズの PLT プログラムを CICS に定義したり、第1フェーズ PLT プログラムで有効にするユーザー出口プログラムを定義したりすることはできません。代わりに、CICS はデフォルトの定義を自動的にインストールします。プログラムの自動インストールを **PGAIPGM** システム初期設定パラメーターでアクティブとして指定しても、定義の変更を可能にするために自動インストール・ユーザー・プログラムが呼び出されることはありません。

CICS によるこのタイプの自動インストールのことをシステムによる自動インストールと呼びます。

CICS は、第1段階 PLT プログラム、および第1段階 PLT プログラムが以下の属性で使用可能にするユーザー出口プログラムを定義します。

```
LANGUAGE(Assembler)
RELOAD(No)
STATUS(Enabled)
CEDF(No)
DATALOCATION(Any)
EXECKEY(CICS)
EXECUTIONSET(Fullapi)
CONCURRENCY(Quasirent)
```

必ず、スレッド・セーフなグローバル・ユーザー出口プログラムを作成してください。ただし、システムにより自動インストールされたプログラムの定義には、**CONCURRENCY(Quasirent)** が指定されます。つまり、出口プログラムが準再入可能と定義されます。第1段階の PLT グローバル・ユーザー出口プログラムをスレッド・セーフとして定義するには、**EXEC CICS ENABLE** コマンドで **THREADSAFE** キーワードを指定します。この値は、システムが自動インストールするプログラム定義の **CONCURRENCY(QUASIRENT)** 設定をオーバーライドします。

- 第1フェーズの PLT プログラムをデバッグするために、Debug Tool を使用することはできません。

## 第2フェーズの PLT プログラム

CICS の初期設定の最後の段階では、ほとんどの CICS サービスを PLT プログラムで使うことができます。これらのプログラムは、PLTPI リストの2番目の部分 (DFHDELIM エントリーの後) で指定されます。

第2フェーズの PLT プログラムは、CICS 初期設定の第3段階で実行されるものであり、**EXEC CICS INQUIRE SYSTEM** コマンドまたは **XPI INQUIRE\_SYSTEM** 呼び出しで **THIRDDINIT** として返されるフェーズです。

第2フェーズ PLTPI プログラムで使えるサービスには、次のような制限があります。

- 領域間通信 (IRC) とシステム間連絡 (ISC) の機能には疑似端末エントリーが関連付けられているため、PLTPI の処理中に IRC または ISC 機能 (ISC over SNA や IP 相互接続 (IPIC) を含む) を実行することはできません。第2フェーズの PLT プログラムでは、リモート・リソースにアクセスしようとする、トランザクション・ルーティングが関係する (したがって議事端末が関係するものも含む) **EXEC CICS** コマンドを実行してはいけません。INQUIRE コマンドさえ実行してはいけません。

この制限は、リモート・リソースが使用できない場合に発生します。リモート・リソースは、以下のいずれかの理由で使えない可能性があります。

- **AUTOCONNECT=NO** が接続定義で指定されている。
- リモート領域が実行されていない。
- リモート領域のリモート・リソースが使用できない。
- ネットワークの問題などでリンクが破損している。

ただし、リモート領域との接続があり、リモート領域のリソースも使用できる場合、この制限は適用されません。

注: 疑似端末:

- AOR にのみ存在する代理 TCTTE でなければなりません。
- トランザクション・ルーティング環境でのみ使用できます。
- 分散プログラム・リンク (DPL) 要求と共存できません。

- 機能シッブ要求のタイプと共存できません。
- 分散トランザクション内に存在できません。
- PLTPI プログラムは、呼び出し元タスクを中断するサービスを要求できますが、タスクを中断すると、制御権が CICS に渡される時間に影響する可能性があります。再開することを別のタスクが判断しなければならないような中断は行わないでください。PLTPI プログラムは、**EXEC CICS RUN TRANSID**、**EXEC CICS FETCH CHILD**、および **EXEC CICS FETCH ANY** コマンドを発行できます。子タスクからの応答がまだない場合、呼び出し元タスクが中断され、制御権が CICS に渡される時間に影響する可能性があります。
- PLTPI プログラムはインターバル制御の **START** コマンドを発行できますが、ATTACH オプションを指定しない限り、要求されたトランザクションは初期設定段階が完了するまで接続されません。**START ATTACH** を使用すると、初期設定が完了する前に、PLTPI プログラムで発行される **START** コマンドを有効にすることができます。ATTACH オプションなしで **START** を使用する場合、起動されたトランザクションは、PLTPI プログラムが完了するまで開始されません。
- PLTPI プログラムでメモリー・ダンプ要求を出してはいけません。
- PLTPI プログラムで **EXEC CICS PERFORM SHUTDOWN** コマンドを使用してはいけません。PLTPI でこのコマンドを使用すると、DFHMDM で重大エラーが発生します。**EXEC CICS PERFORM SHUTDOWN IMMEDIATE** コマンドは使用できます。
- JVM サーバーは PLTPI プログラムとは非同期に開始されるため、PLTPI プログラムを JVM サーバーで実行される Java プログラムにしてはいけません。
- 第 2 段階の静止の PLT プログラムには、プログラム・リソース定義は必要ありません。これらが定義されていないければ、(プログラムの自動インストールのシステム 初期設定パラメーターに関係なく) システムに自動インストールされます。定義の変更を可能にするために自動インストール出口が呼び出されることはありません。プログラムは次の属性で定義されます。

```
LANGUAGE(ASSEMBLER)
STATUS(ENABLED)
CEDF(NO)
DATALOCATION(BELOW)
EXECKEY(CICS)
EXECUTIONSET(FULLAPI)
```

その結果、システムで自動インストールされたプログラムは、デフォルトの CONCURRENCY 設定 QUASIRENT、およびデフォルトの API 設定 CICSAPI になります。別の CONCURRENCY 設定または API 設定で PLT プログラムを実行する場合、または XPLINK コンパイラー・オプションを使用してコンパイルした C または C++ プログラムを使用する場合は、適切なリソース定義を用意してください。また、Language Environment 準拠プログラムを使用する場合は、CICSVAR ランタイム・オプションを使用して、適切な CONCURRENCY 値および API 値を設定してください。[言語環境プログラムのランタイム・オプションの定義](#)を参照してください。

- PLTPI プログラムで **EXEC CICS INVOKE SERVICE** コマンドを使用してはいけません。CICS 初期設定のこの段階では、パイプライン・スキャンは開始されていません。パイプライン・スキャンの結果として作成される WEBSERVICE 定義がまだないため、Web サービス要求は失敗します。
- Debug Tool を使用して、第 2 フェーズの PLT プログラムをデバッグすることはできません。

### PLTPI 処理時の遅延リカバリーの効果

リカバリー処理は PLTPI 処理が完了するまで実行されないため、PLT プログラムは、緊急時再始動中に保存ロックで保護されているリソースにアクセスしようとした場合、失敗することがあります。LOCKED 例外条件を処理するように PLT プログラムが作成されていない場合は、AEX8 異常終了コードで異常終了します。

CICS アプリケーションの開始を許可する前に PLTPI 処理を正常に完了することが不可欠な場合、必要な PLT 処理を完了する代替方法を検討してください。緊急時再始動のリカバリー処理が終了するのを許容し、ロックが解放されたときに、失敗した PLTPI 処理を完了することが必要な場合があります。

## シャットダウン・プログラムの作成

CICS のシャットダウン時に実行するすべてのプログラムをプログラム・リスト・テーブル (PLT) で定義し、その PLT の名前をプログラム・リスト・テーブル・シャットダウン (PLTSD) システム 初期設定パラメーターで指定しなければなりません。

その PLTSD 値をオーバーライドすることもできます。その場合は、CICS Explorer® の「領域」操作ビューの「シャットダウン」オプションを使用するか、**CEMT PERFORM SHUTDOWN** コマンドまたは **EXEC CICS PERFORM SHUTDOWN** コマンドで PLT 名を指定します。PLTSD プログラムが異常終了すると、同期点ロールバックが発生します。

### 第 1 静止フェーズの PLT プログラム

CICS シャットダウンの第 1 静止段階中に実行されるプログラムは、PLT の前半 (DFHDELIM ステートメントの前) で指定されます。

第 1 段階の PLTSD プログラムを CICS に定義する必要があります。プログラムを静的に定義することも、プログラムの自動インストールを使用することもできます。JVM サーバーで実行される Java プログラムを定義することはできません。

第 1 静止段階ではまだ端末を使用できますが、端末の入力によって開始されたタスクは拒否されます。ただし、シャットダウン・トランザクション・リスト・テーブル (XLT) で指定されているタスク、または CEMT、CSAC、CSTE、CSNE などの CICS 提供のトランザクションであるタスク (用意されている定義で SHUTDOWN(ENABLED) として定義されているもの) は拒否されません。

第 1 静止段階は、第 1 段階のすべての PLT プログラムの実行が完了し、システムにユーザー・タスクが存在しなくなったときに完了します。

第 1 静止段階では、Debug Tool を使用して PLT プログラムをデバッグすることはできません。

### 第 2 静止段階の PLT プログラム

CICS シャットダウンの第 2 静止段階中に実行されるプログラムは、PLT の後半 (DFHDELIM ステートメントの後) で指定されます。

第 2 段階の初期設定と第 2 段階の静止の PLT プログラムに、プログラムのリソース定義は必要ありません。これらが定義されていなければ、(プログラムの自動インストールのシステム 初期設定パラメーターに関係なく) システムにより自動インストールされます。つまり、定義の変更を可能にするために自動インストール出口が呼び出されることはありません。プログラムは次の属性で定義されます。

```
LANGUAGE(ASSEMBLER) STATUS(ENABLED) CEDF(NO)
DATALOCATION(ANY) EXECKEY(CICS)
EXECUTIONSET(FULLAPI)
```

その結果、システムで自動インストールされたプログラムは、デフォルトの CONCURRENCY 設定 QUASIRENT、およびデフォルトの API 設定 CICSAPI になります。

- API 属性に OPENAPI 値を指定して定義したスレッド・セーフな PLT プログラム、または XPLINK コンパイラー・オプションを使用してコンパイルした C または C++ プログラムを使用する場合は、適切なリソース定義を用意してください。また、Language Environment 準拠プログラムを使用する場合は、CICSVAR ランタイム・オプションを使用して、適切な CONCURRENCY 値および API 値を設定してください。[言語環境プログラムのランタイム・オプションの定義を参照してください。](#)

第 2 静止段階では、新しいタスクは開始できず、端末も使用できません。このため、第 2 フェーズの PLT プログラムでは、他のタスクを開始してはいけませんし、端末と通信することもできません。また、第 2 フェーズの PLT プログラムで、リソース・セキュリティ検査または Db2 呼び出しを実行させてはいけません。PLT プログラムを、JVM サーバーで実行される Java プログラムにすることはできません。

PLTSD プログラムの実行中にトランザクションの異常終了が発生した場合、CICS は永続的な待機状態になります。これを防止するには、PLTSD プログラムで**すべての**異常終了状況を処理するようにしてください。

第 2 静止段階は、第 2 フェーズのすべての PLT プログラムの実行が完了したときに、完了します。

第 2 静止段階では、Debug Tool を使用して PLT プログラムをデバッグすることはできません。



## シャットダウン補助ユーティリティー・プログラム、DFHCESD

CICS には、シャットダウンの第 1 静止段階で実行できるシャットダウン補助トランザクションが用意されています。これは通常のシャットダウンでも即時シャットダウンでも実行できます。

シャットダウン・トランザクションの名前は、システム初期設定パラメーター **SDTRAN**、または **PERFORM SHUTDOWN** および **PERFORM SHUTDOWN IMMEDIATE** コマンドの **SDTRAN** オプションで指定します。また、シャットダウン補助トランザクションを実行しないように指定することもできます。シャットダウン補助トランザクションを実行しないように指定した場合は、次の処理が行われます。

- 通常シャットダウンでは、実行中のすべてのタスクが終了するのを待ってから、CICS は静止の第 2 段階に移行します。長期実行トランザクションや会話型トランザクションでは、許容できない遅延が発生したり、オペレーターによる介入が必要になったりする場合があります。
- 即時シャットダウンの場合、CICS では実行中のタスクを終了することができません。また、緊急時再始動が行われるまで、バックアウトは実行されません。これによって、許容できない数の作業単位が放置され、複数のロックが不要に保持されることになります。

シャットダウン補助トランザクションの目的は、このような問題を解決することです。すなわち、できる限り多くのタスクを妥当な時間内に正常にコミットまたはバックアウトします。

デフォルトのシャットダウン補助トランザクションは **CESD** です。これは、CICS 提供のプログラム **DFHCESD** を起動します。**DFHCESD** は、徐々に強力を増していく手法を使用して、長期実行タスクのページおよびバックアウトを試行します。できる限り多くのタスクを正常にコミットまたはバックアウトして、制御された方法で CICS がシャットダウンできるようにします。**DFHCESD** の詳細およびシャットダウン補助トランザクションを独自で作成する方法については、[シャットダウン補助プログラム \(DFHCESD\)](#) を参照してください。

## 初期設定プログラムおよびシャットダウン・プログラムの作成時の一般的な考慮事項

初期設定プログラムおよびシャットダウン・プログラムを作成する場合は、**PLT**、**PLTPI**、**PLTSD** プログラムへの影響を考慮してください。

以下の情報は、初期設定プログラムとシャットダウン・プログラムの両方に適用されます。

- **EXEC CICS RETURN** コマンドを使用して、**PLT** プログラムをすべて終了してください。
- **PLT** プログラムは、1 次スペース変換モードで制御権を受け取ります。変換モードの詳細については、[z/Architecture](#) 解説書を参照してください。**PLT** プログラムは同じモードで制御権を CICS に戻さなければならず、使用した汎用レジスターまたはアクセス・レジスターを復元しなければなりません。
- すべての **PLTPI** プログラムは、CICS 内部トランザクション名 **CPLT** で実行されます。したがって、CICS 内部トランザクションは未確定待機属性を **YES** に設定して定義されているので、**PLTPI** プログラムの実行時に未確定の障害が発生すると、関係する **UOW** は放置されます。**PLTPI** プログラムは **ASP1** を異常終了させ、CICS は **PLTPI** テーブル内に定義されている次のプログラム (ある場合) を実行します。
- **PLTSD** プログラムは、**PERFORM SHUTDOWN** コマンドを発行したトランザクションで実行されます。**CEMT** トランザクションは **WAIT(YES)** を指定して定義されています。したがって、**CEMT PERFORM SHUTDOWN** コマンドまたは CICS Explorer の「領域」操作ビューの「シャットダウン」オプションによってシャットダウンされる場合、**PLTSD** プログラムの実行中に未確定の障害が発生すると、**UOW** が放置されます。一方、ユーザー・トランザクションから発行された **EXEC CICS PERFORM SHUTDOWN** コマンドによってシャットダウンされる場合、未確定の障害により **UOW** が放置されるか強制的に決定されるかは、そのユーザー・トランザクションの未確定属性に応じて異なります。**CEDA DEFINE TRANSACTION** コマンドの未確定オプションの詳細については、[TRANSACTION 属性](#) を参照してください。
- **CEMT** の **TRANSACTION** リソース定義は、**TASKDATALOC(ANY)** を指定します。このため **CEMT** トランザクションは、16 MB 境界より上の 31 ビット・ストレージを使用します。**CEMT** を使用して CICS をシャットダウンする際に、**PLTSD** プログラムが **AMODE(24)** である場合は、**AEZC** 異常終了が発生します。この状況を回避するには、シャットダウン・プログラムを **AMODE(31)** に変更し、該当するプログラム定義を更新します。

## PLT プログラムの記憶キー

以下の点を考慮する必要があります (ストレージ保護機能を有効にして CICS を実行するかどうかは関係ありません)。

- PLT プログラムの呼び出しに使用される実行キー
- PLT プログラムのために取得されるデータ・ストレージの記憶キー。

### PLT プログラムの実行キー

CICS は、常に CICS キーで PLT プログラムに制御権を渡します。

プログラムのリソース定義で EXECKEY(USER) を指定しても、CICS は、初期設定中またはシャットダウン中に起動された PLT プログラムに制御権を渡すときに CICS キーを強制します。ただし、PLT 定義のシャットダウン・プログラム自体が、(リンクまたは制御権移動コマンドによって) 別のプログラムに制御権を渡した場合、起動されたプログラムは、そのプログラムのリソース定義で定義された実行キー (EXECKEY) に従って実行されます。

**重要:** PLT プログラム、および PLT プログラムから制御権を渡すプログラムの両方を定義するときには、EXECKEY(CICS) を指定することを強くお勧めします。

### PLT プログラム用のデータ記憶キー

PLT プログラムで使用するデータ記憶キーの内容は、ストレージを取得した方法によって異なります。

ストレージは以下の方法で取得できます。

- PLT プログラムによって要求される作業用ストレージは、PLT プログラムを開始するトランザクションの TASKDATAKEY 値によって設定されているキーで指定されます。初期化中に実行される PLT プログラム (PLTPI プログラム) の場合、トランザクションは常に内部 CICS トランザクションであり、TASKDATAKEY 値は常に CICS になります。シャットダウン中に実行されるプログラム (PLTSD プログラム) の場合、設定はシャットダウン・コマンドの発行にどのトランザクションを使用するかによって異なります。CICS Explorer の「領域」操作ビューから「シャットダウン」オプションを選択するか、**CEMT PERFORM SHUTDOWN** コマンドを発行すると、TASKDATAKEY 値は常に CICS になります。ユーザー定義のトランザクションを実行して、**EXEC CICS PERFORM SHUTDOWN** コマンドを発行するプログラムを開始すると、TASKDATAKEY は USER または CICS のいずれかになります。
- PLT プログラムは、**EXEC CICS** コマンドを使用して、ストレージを取得するために以下を発行することができます。

– 明示的な **EXEC CICS GETMAIN** コマンド

– SET オプションを使用した EXEC CICS コマンドの結果としてなされる暗黙的なストレージ要求

**EXEC CICS** コマンドによって取得されるストレージのデフォルト・ストレージ・キーは、PLT プログラムを開始したトランザクションの TASKDATAKEY 値によって、作業用ストレージについて記述されているとおりに設定されます。

一例として、あるトランザクションに、PLT シャットダウン・プログラムを開始する TASKDATAKEY(USER) が定義されている場合を考えてみましょう。この場合、この PLT プログラムが **EXEC CICS** コマンドを使用して獲得する暗黙的または明示的なストレージは、デフォルトでは、ユーザー・キー・ストレージ内にあります。ただし、EXEC CICS GETMAIN コマンドにおいて、PLT プログラムは CICS DATAKEY または USER DATAKEY を指定することで TASKDATAKEY オプションをオーバーライドできます。

## 第3章 ユーザー置換可能プログラムを使用したカスタマイズ

ユーザー置換可能プログラムは、CICS 処理の特定のポイントで CICS コードの一部であるかのように常に呼び出される CICS 提供プログラムです。提供されているプログラムに独自の論理を組み込んでそれを変更したり、自分で作成したバージョンでそのプログラムを置き換えたりできます。

独自版のユーザー置換可能プログラムを作成する場合の注意点を以下にまとめます。

- CICS でサポートされているどの言語でも (つまり、アセンブラー言語でも、COBOL、PL/I、C のいずれかでも) ユーザー置換可能プログラムをコーディングできます。ほとんどのプログラムについては、CICSTS56.CICS.SDFHSAMP ライブラリーにアセンブラー言語版がソース形式で用意されています。一部のプログラムについては、COBOL 版、PL/I 版、C 版も用意されています。各プログラムの説明では、用意されているサンプル・プログラムやコピーブックやマクロを取り上げています。
- ユーザー置換可能プログラムから **EXEC CICS HANDLE ABEND** コマンドを実行することによって、異常終了をトラップできます。ただし、**HANDLE ABEND** を実行しないと、CICS は、タスクを異常終了させずに、そのプログラムを呼び出した CICS モジュールに制御を戻します。その CICS モジュールが実行するアクションは、該当するユーザー置換可能プログラムによって異なります。
- ユーザー置換可能プログラムから制御を戻される時に、CICS は、常に 1 次スペース変換モードで制御を受け取り、すべてのアクセス・レジスターを元の内容に復元し、すべての汎用レジスターも (戻りコードやリンケージ情報を提供するレジスターを除いて) 復元します。

変換モードについては、[z/Architecture 解説書](#)を参照してください。

- z/OS では、どの許可モード (監視プログラム状態、システム PSW キー、または APF 許可) でも、呼び出し元に制御を返す SVC または PC ルーチンをインストールしないでください。そのようにすることは、[z/OS Statement of Integrity](#) に反しています。こうしたサービスを CICS から起動する場合、システム保全性が損なわれることがあり、結果として発生するどんな問題も IBM サービス技術員によって解決されません。
- ユーザー置換可能プログラムや、ユーザー置換可能プログラムから呼び出すすべてのプログラムも、RMODE は ANY でかまいませんが、AMODE は必ず 31 にしてください。
- ユーザー置換可能プログラムは、ローカル・プログラムとして定義する必要があります。ユーザー置換可能プログラムをリモート領域で実行することはできません。この規則は、自動インストール制御プログラムや動的ルーティング・プログラムも含め、すべてのユーザー置換可能プログラムに当てはまります。
- ユーザー置換可能プログラムがプログラム・チェック時に生成するのは、システム・ダンプだけです。トランザクション・ダンプは生成しません。
- CICS 実行診断機能 (EDF) を使用してユーザー置換可能プログラムを検査できます。ただし、初期トランザクションが CICS 提供トランザクションになっていると、EDF は機能しません。

### ユーザー置き換え可能プログラムとストレージ保護機能

ストレージ保護機能を使用して CICS を実行する場合、プログラムが実行される実行キーと、プログラムによって取得されるデータ・ストレージのストレージ・キーを決める必要があります。

#### ユーザー置き換え可能プログラムの実行キー

ストレージ保護をアクティブにして実行中の場合、CICS は CICS キーでユーザー置き換え可能プログラムに制御を渡します。

PROGRAM リソースに **EXECKEY(USER)** を指定した場合でも、CICS はプログラムを呼び出すときに CICS キーを強制します。ただし、ユーザー置き換え可能プログラム自体が別のプログラムに制御を渡す場合は、呼び出されたプログラムは、PROGRAM リソースに定義された実行キー (**EXECKEY**) に従って実行されます。

**重要:** ユーザー置き換え可能プログラムと、ユーザー置き換え可能プログラムから制御が渡されるプログラムの両方を定義する場合は、**EXECKEY(CICS)** を指定してください。

## ユーザー置き換え可能プログラムのデータ記憶キー

ユーザー置き換え可能プログラムによって使用されるストレージのストレージ・キーは、ストレージの取得方法に応じて異なります。

- 呼び出し元によってユーザー置き換え可能プログラムに渡される通信域は、常に CICS キーのものです。
- ユーザー置き換え可能プログラムのために取得される作業用ストレージは、そのプログラムの開始に使用されるトランザクションの TASKDATAKEY によって設定されたキー・セットのものです。
- ユーザー置き換え可能プログラムは、以下を発行することで、**EXEC CICS** コマンドを使用してストレージを取得できます。
  - 明示的な **EXEC CICS GETMAIN** コマンド
  - SET オプションを使用した **EXEC CICS** コマンドの結果としてなされる暗黙的なストレージ要求。

**EXEC CICS** コマンドで取得されるストレージのデフォルトの記憶キーは、ユーザー・プログラムを開始したトランザクションの TASKDATAKEY によって設定されます。

例として、ユーザー置き換え可能プログラムの呼び出しを引き起こす、TASKDATAKEY(USER) で定義されたトランザクションについて検討します。この場合、**EXEC CICS** コマンドによってユーザー・プログラムで獲得される暗黙的または明示的なストレージは、デフォルトでユーザー・キー・ストレージのものです。ただし、**EXEC CICS GETMAIN** コマンドにおいて、ユーザー・プログラムは CICS DATAKEY または USER DATAKEY を指定することで TASKDATAKEY オプションをオーバーライドできます。

## プログラム・エラー・プログラムの作成

提供されているデフォルト・プログラム DFHPEP に基づいて、プログラム・エラー・プログラムを作成できます。

CICS 提供のデフォルトのプログラム・エラー・プログラム (DFHPEP) には、プログラムのアドレス可能性を取得したり、通信域にアクセスしたり、**EXEC CICS RETURN** コマンドによって CICS に制御を戻したりするためのコードが含まれています。

DFHPEP のソースは、アセンブラ言語および C 言語のバージョンで提供されます。いずれかのバージョンに変更を加えて、独自のロジックを組み込むことができます。あるいは、CICS でサポートされているいずれかの言語で、独自のプログラム・エラー・プログラムを作成することも可能です。プログラム・エラー・プログラムには、固有の制約事項が適用されます。

- プログラムの名前は DFHPEP でなければなりません。
- プログラムで、MRO 機能または ISC 機能 (分散トランザクション処理や機能シップなど) を使用する **EXEC CICS** コマンドを発行することはできません。
- プログラムで、リカバリー可能リソースにアクセスするコマンドを発行することはできません。
- プログラムで、トランザクション・ダンプを取得するかどうかに影響を与えることはできません。

デフォルトの DFHPEP モジュールは、ダミー・モジュールです。カスタマイズするには、ソースを自分でコーディングする必要があります。DFHPEP のリストは、[72 ページの図 13](#) に記載されています。プログラム・エラー・プログラムを作成したら、そのプログラムを変換してアセンブルし、提供されているダミー・プログラムと置き換えて使用します。ユーザー置換可能プログラムのアセンブルとリンク・エディットに必要なジョブ制御ステートメントに関する情報は、[337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』](#)を参照してください。

通信域で DFHPEP に対して使用できる情報には、以下のものが含まれます。

- PEP\_COM\_CURRENT\_ABEND\_CODE にある、現行の異常終了コード。
- PEP\_COM\_ORIGINAL\_ABEND\_CODE にある、オリジナルの異常終了コード。トランザクションが複数回異常終了した場合、オリジナルの異常終了コードと現行の異常終了コードは異なります。例えば、障害が起きたプログラムが、その前の異常終了の処理中に異常終了する場合などが該当します。この場合、オリジナルの異常終了とは、トランザクションで最初に発生した異常終了のことです。
- PEP\_COM\_USERS\_EIB にある、最後の **EXEC CICS** コマンドの時点の EIB。



- PEP\_COM\_ABPROGRAM にある、(現行の) 異常終了が発生したプログラムの名前。  
PEP\_COM\_ABPROGRAM は、以下のようにプログラムを識別します。
  - リモート・システムで実行している分散プログラム・リンク (DPL) サーバー・プログラムで異常終了が発生した場合、サーバー・プログラムを識別します。
  - 異常終了がローカルの ASRA、ASRB、または ASRD である場合、プログラム・チェックまたはオペレーティング・システムの異常終了が発生したプログラムを識別します。
  - 他のすべての場合、現行のプログラムを識別します。
- PEP\_COM\_PSW、または 16 バイト PSW の場合は PEP\_COM\_PSW16 にある、(現行の) 異常終了時のプログラム状況ワード (PSW)。PSW の完全な内容は、ASRA、ASRB、および ASRD の異常終了コードの場合にのみ意味があります。PEP\_COM\_PSW の最後の 4 バイトおよび PEP\_COM\_PSW16 の最後の 8 バイト (PSW アドレス) は、AICA コードにも適用されます。
- PEP\_COM\_REGISTERS にある、(現行の) 異常終了時の GP レジスター (0-15)。
- PEP\_COM\_KEY にある、(現行の) 異常終了が発生した時点のプログラムの実行キー。PEP\_COM\_KEY の値は、ASRA および ASRB の異常終了コードの場合にのみ意味があります。
- PEP\_COM\_STORAGE\_HIT にある、ストレージ保護例外の結果として (現行の) 異常終了が発生したかどうかを示す情報。PEP\_COM\_STORAGE\_HIT の値は、ASRA 異常終了コードの場合にのみ意味があります。この値は、問題のプログラムが上書きしようとした保護動的ストレージ域 (CDSA、RDSA、ECDSA、ERDSA、ETDSA、GCDSA、または GUDSA) を示します (該当する場合)。
- 追加のレジスター情報を使用できる場合があります。例えば、64 ビット GP レジスター、アクセス・レジスター、浮動小数点レジスター、ベクトル・レジスターなどです。どのレジスター値が使用可能であるかを示す標識が通信域で設定されます。
- ブレーク・イベント・アドレスが使用可能な場合は、通信域に保管されます。ブレーク・イベント・アドレスが使用可能でない場合は、ゼロになります。
- PEP\_COM\_INT にある、プログラム状況ワード中断情報。

PSW、レジスター、実行キー、およびアプリケーションが上書きを試みた保護ストレージの種類に関する情報は、ローカル・システムで異常終了が発生した場合にのみ意味があります。リモート・システムで実行している DPL サーバー・プログラムで異常終了が発生した場合、これらのフィールドはゼロに設定されます。

トランザクションを使用不可にする場合は、値 PEP\_COM\_RETURN\_DISABLE を PEP\_COM\_RETURN\_CODE フィールドに割り当てます。そうでない場合は、このフィールドをデフォルトのゼロにするか、または値 PEP\_COM\_RETURN\_OK に設定します。CICS では CICS 提供のトランザクションを使用不可にできません。したがって、ID が文字 C で始まるトランザクションを使用不可にしないでください。

72 ページの図 13 に、デフォルトのプログラム・エラー・プログラムのアセンブラ言語ソース・コードを記載します。73 ページの図 14 から 75 ページの図 16 は、通信域のソース・コードを示しています。



```

DFHEISTG DSECT ,
*
*      Insert your own storage definitions here
*
      DFHPCOM TYPE=DSECT
*****
* * * * *      P R O G R A M   E R R O R      * * * * *
* * * * *      P R O G R A M      * * * * *
*****
DFHPEP      CSECT      PROGRAM ERROR PROGRAM CSECT
DFHPEP      RMODE ANY
DFHPEP      DFHREGS ,      EQUATE REGISTERS
      XR      R1,R1
      ICM      R1,B'0011',EIBCALEN Get Commarea length
      BZ      RETURNX      ...no Commarea; exit
      EXEC CICS ADDRESS COMMAREA(R2) ,
      USING DFHPEP_COMMAREA,R2
*
*      Insert your own code here
*
      LA      R1,PEP_COM_RETURN_OK
      B      RETURN
DFHEJECT
*
RETURNER DS      0H      Return for error cases
      LA      R1,PEP_COM_RETURN_DISABLE
RETURN    DS      0H
      ST      R1,PEP_COM_RETURN_CODE
RETURNX   DS      0H
      EXEC CICS RETURN ,
      END      DFHPEP

```

図 13. デフォルトのプログラム・エラー・プログラム (DFHPEP) のソース・コード

73 ページの図 14 から 75 ページの図 16 は、デフォルトのプログラム・エラー・プログラムの通信域のアセンブラー言語ソース・コードを示しています。

```

DFHPEP_COMMAREA DSECT
*
*                               Standard header section
*
PEP_COM_STANDARD          DS      0F
PEP_COM_FUNCTION          DS      CL1      Always '1'
PEP_COM_COMPONENT        DS      CL2      Always 'PC'
PEP_COM_RESERVED         DS      C        Reserved
*
*                               Abend codes and EIB
*
PEP_COM_CURRENT_ABEND_CODE DS      CL4      Current abend code
PEP_COM_ORIGINAL_ABEND_CODE DS      CL4      Original abend code
PEP_COM_USERS_EIB        DS      CL(EIBRLDBK-EIBTIME+L'EIBRLDBK)
*                               EIB at last EXEC CICS command

*
* Debugging information (program, PSW, registers and execution key at
* time of abend, hit storage indicator). If the abend occurred in a
* DPL server program running remotely, only program is meaningful.
*
PEP_COM_DEBUG            DS      0F
PEP_COM_ABPROGRAM        DS      CL8      Program causing abend
PEP_COM_PSW              DS      CL8      PSW at abend
*                               (codes ASRA, ASRB, AICA, ASRD)
PEP_COM_REGISTERS        DS      CL64     GP registers at abend
*                               (registers 0-15)
PEP_COM_KEY              DS      X        Execution key at abend
*                               (ASRA and ASRB only)
PEP_COM_USER_KEY         EQU      9      User key
PEP_COM_CICS_KEY         EQU      8      CICS key
*
PEP_COM_STORAGE_HIT      DS      X        Storage type hit by 0C4
*                               (ASRA only)
PEP_COM_NO_HIT           EQU      0      No hit, or not 0C4
PEP_COM_CDOSA_HIT        EQU      1      CDOSA hit
PEP_COM_ECDSA_HIT        EQU      2      ECDSA hit
PEP_COM_ERDSA_HIT        EQU      3      ERDSA hit
PEP_COM_RDOSA_HIT        EQU      4      RDOSA hit
PEP_COM_EUDSA_HIT        EQU      5      EUDSA hit
PEP_COM_UDSA_HIT         EQU      6      UDSA hit

PEP_COM_ETDSA_HIT        EQU      7      ETDSA hit
PEP_COM_GCDSA_HIT        EQU      8      GCDSA hit
PEP_COM_GUDSA_HIT        EQU      9      GUDSA hit
*

```

図 14. DFHPEP 通信域のソース (アセンブラ言語)

```

PEP_COM_SPACE          DS      X      Subspace/basespace
PEP_COM_NOSPACE        EQU      0
PEP_COM_SUBSPACE       EQU      10    Abending task was in
*                               subspace
PEP_COM_BASESPACE      EQU      11    Abending task was in
*                               basespace
PEP_COM_PADDING        DS      CL2    Reserved
*
*                               Return code
*
PEP_COM_RETURN_CODE     DS      F
PEP_COM_RETURN_OK       EQU      0
PEP_COM_RETURN_DISABLE EQU      4    Disable transaction
*
*                               Additional Program status word information
*
PEP_COM_INT            DS      CL8    PSW interrupt codes
*

*                               Breaking Event Address
*
PEP_COM_BEAR           DS      AD     Breaking Event Addr
*

*                               Additional register information
*
PEP_COM_FLAG1          DS      0D     Force alignment
PEP_COM_GP64_REGS_AVAIL EQU      X'80' 64 bit register values
*                               available in
*                               PEP_COM_G64_REGISTERS
PEP_COM_ACCESS_REGS_AVAIL EQU      X'40' 64 bit register values
*                               available in
*                               PEP_COM_ACCESS_REGISTERS
PEP_COM_ORIGINAL_FPR_AVAIL EQU      X'20' FPR 0, 2, 4 & 6 values
*                               available in
*                               PEP_COM_FP_REGISTERS
PEP_COM_ADDITIONAL_FPR_AVAIL EQU      X'10' All FPR available in
*                               PEP_COM_FP_REGISTERS &
*                               FPCR in
*                               PEP_COM_FPC_REGISTER
*                               Reserved
PEP_COM_GP64_REGISTERS DS      CL7
PEP_COM_FP_REGISTERS   DS      CL128 64 bit GP registers
PEP_COM_FP_REGISTER0   DS      FD     FP register 0
PEP_COM_FP_REGISTER1   DS      FD     FP register 1
PEP_COM_FP_REGISTER2   DS      FD     FP register 2
PEP_COM_FP_REGISTER3   DS      FD     FP register 3
PEP_COM_FP_REGISTER4   DS      FD     FP register 4
PEP_COM_FP_REGISTER5   DS      FD     FP register 5
PEP_COM_FP_REGISTER6   DS      FD     FP register 6
PEP_COM_FP_REGISTER7   DS      FD     FP register 7
PEP_COM_FP_REGISTER8   DS      FD     FP register 8
PEP_COM_FP_REGISTER9   DS      FD     FP register 9
PEP_COM_FP_REGISTER10  DS      FD     FP register 10
PEP_COM_FP_REGISTER11  DS      FD     FP register 11
PEP_COM_FP_REGISTER12  DS      FD     FP register 12
PEP_COM_FP_REGISTER13  DS      FD     FP register 13
PEP_COM_FP_REGISTER14  DS      FD     FP register 14
PEP_COM_FP_REGISTER15  DS      FD     FP register 15
PEP_COM_FPC_REGISTER   DS      F      FPC register
PEP_COM_ACCESS_REGISTERS DS      CL64  Access registers
*

```

図 15. DFHPEP 通信域のソース (アセンブラー言語) (続き)

```

*
*                               16 byte PSW at time of abend
*
PEP_COM_PSW16                DS      CL16      16 byte PSW
*
*                               Vector Register Information
*
PEP_COM_VR_REGISTERS         DS      0CL512    VR registers
PEP_COM_VR_REGISTER0         DS      CL16      VR Register 0
PEP_COM_VR_REGISTER1         DS      CL16      VR Register 1
PEP_COM_VR_REGISTER2         DS      CL16      VR Register 2
PEP_COM_VR_REGISTER3         DS      CL16      VR Register 3
PEP_COM_VR_REGISTER4         DS      CL16      VR Register 4
PEP_COM_VR_REGISTER5         DS      CL16      VR Register 5
PEP_COM_VR_REGISTER6         DS      CL16      VR Register 6
PEP_COM_VR_REGISTER7         DS      CL16      VR Register 7
PEP_COM_VR_REGISTER8         DS      CL16      VR Register 8
PEP_COM_VR_REGISTER9         DS      CL16      VR Register 9
PEP_COM_VR_REGISTER10        DS      CL16      VR Register 10
PEP_COM_VR_REGISTER11        DS      CL16      VR Register 11
PEP_COM_VR_REGISTER12        DS      CL16      VR Register 12
PEP_COM_VR_REGISTER13        DS      CL16      VR Register 13
PEP_COM_VR_REGISTER14        DS      CL16      VR Register 14
PEP_COM_VR_REGISTER15        DS      CL16      VR Register 15
PEP_COM_VR_REGISTER16        DS      CL16      VR Register 16
PEP_COM_VR_REGISTER17        DS      CL16      VR Register 17
PEP_COM_VR_REGISTER18        DS      CL16      VR Register 18
PEP_COM_VR_REGISTER19        DS      CL16      VR Register 19
PEP_COM_VR_REGISTER20        DS      CL16      VR Register 20
PEP_COM_VR_REGISTER21        DS      CL16      VR Register 21
PEP_COM_VR_REGISTER22        DS      CL16      VR Register 22
PEP_COM_VR_REGISTER23        DS      CL16      VR Register 23
PEP_COM_VR_REGISTER24        DS      CL16      VR Register 24
PEP_COM_VR_REGISTER25        DS      CL16      VR Register 25
PEP_COM_VR_REGISTER26        DS      CL16      VR Register 26
PEP_COM_VR_REGISTER27        DS      CL16      VR Register 27
PEP_COM_VR_REGISTER28        DS      CL16      VR Register 28
PEP_COM_VR_REGISTER29        DS      CL16      VR Register 29
PEP_COM_VR_REGISTER30        DS      CL16      VR Register 30
PEP_COM_VR_REGISTER31        DS      CL16      VR Register 31

*                               length of DFHPEP_COMMAREA
*
PEP_COM_LEN EQU *-PEP_COM_STANDARD

```

図 16. DFHPEP 通信域のソース (アセンブラー言語) (続き)

## サンプルのプログラム・エラー・プログラム

デフォルト・プログラムのソース・レベルのバージョン (アセンブラー言語でコーディングされた DFHPEP と C でコーディングされた DFHPEPD) が用意されています。どちらも CICSSTS56.CICS.SDFHSAMP ライブラリーにあります。

アセンブラー言語マクロ DFHPCOM および対応する C コピーブック DFHPCOMD を使用して、通信域を定義できます。これらはそれぞれ、CICSSTS56.CICS.SDFHMAC ライブラリーおよび CICSSTS56.CICS.SDFHC370 ライブラリーにあります。

CICS でサポートされているどの言語でもプログラム・エラー・プログラムをコーディングできますが、プログラムの名前は必ず DFHPEP にする必要があります。

## カスタム EP アダプターの作成

カスタム EP アダプターは、イベント・バインディングによって生成されたイベントのフォーマットを設定した後、イベントを発行する、イベント・バインディングに関連付けられた CICS プログラムです。

### このタスクについて

CICS 提供 EP アダプターではイベント処理の要件を満たせない場合、イベント・データを処理する独自のカスタム EP アダプターを自分で作成することができます。

CICS は、発行されるイベントごとに EP アダプターを呼び出します。カスタム EP アダプターへの入力 は 現行チャネルであり、このチャネルには CICS イベント・オブジェクトがコンテナの集合として格納されています。コンテナは、DFHEP.CONTEXT、DFHEP.DESSCRIPTOR、DFHEP.ADAPTER、

DFHEP.ADAPTPARM、DFHEP.CHAR.nnnnnn、DFHEP.DATA.nnnnnn、およびDFHEP.ERR.nnnnnnです。DFHEP.CONTEXT、DFHEP.DESSCRIPTOR、およびDFHEP.ADAPTPARMの各コンテナに対して、コピーブックが提供されています。これらのコピーブックは、CICS リリース間で変更される可能性があります。このため、CICS のリリースが新しくなるたびに、カスタム EP アダプターを再コンパイルする必要があります。

発行されるイベントのほかに、カスタム EP アダプターは成功または失敗の標識を生成する必要があります。

## 手順

1. 使用するプログラミング言語用の、イベント・コンテキスト・データ・コピーブックをインクルードします。  
このコピーブックには、EP アダプターで処理するイベントのコンテキスト・データのDFHEP.CONTEXT コンテナが記述されています。
  - アセンブラー言語用の DFHEPCXD
  - COBOL 用の DFHEPCX0
  - PL/I 用の DFHEPCXL
  - C 用の DFHEPCXH
2. 使用するプログラミング言語用の、イベント記述子コピーブックをインクルードします。  
このコピーブックには、EP アダプターで処理するイベントのキャプチャー・ビジネス・データを記述する、DFHEP.DESSCRIPTOR コンテナが記述されています。
  - アセンブラー言語用の DFHEPDED
  - COBOL 用の DFHEPDE0
  - PL/I 用の DFHEPDEL
  - C 用の DFHEPDEH
3. 使用するプログラミング言語用の EP アダプター・パラメーター・コピーブックをインクルードします。  
このコピーブックは、DFHEP.ADAPTPARM コンテナを記述しています。
  - アセンブラー言語用の DFHEPAPD
  - COBOL 用の DFHEPAPO
  - PL/I 用の DFHEPAPL
  - C 用の DFHEPAPH
4. イベント・コンテキスト・データ・コピーブックを使用して、DFHEP.CONTEXT コンテナからコンテキスト情報を取得します。
5. EP アダプター・カスタム・データを取得します。DFHEP.ADAPTER コンテナには、「**カスタム・アダプターに渡されたデータ (Data passed to the Custom Adapter)**」フィールドに指定されたデータが格納されます。このフィールドは、イベント・バインディングを設定する、イベント・バインディング・エディターの「**アダプター (Adapter)**」タブにあります。  
**注:** このコンテナのデータは、EBCDIC 文字に限定されます。
6. EP アダプター・パラメーター構成コピーブックを使用して、DFHEP.ADAPTPARM コンテナから EP アダプター・パラメーターを取得します。  
DFHEP.ADAPTPARM コンテナ内で重要な情報項目は、発行のリカバリー可能性を示す標識 EPAP-RECOVER です。このコンテナには、EPAP-ADAPTER-NAME というアダプター名も含まれています。
7. カスタム・アダプターは、カスタム EP アダプター内の DFHEP.ADAPTPARM コンテナの EPAP-RECOVER 設定値を検査する必要があります。  
EP アダプターは、リカバリー可能またはリカバリー不能な方式でトランスポートを使用して、イベントを発行する必要があります。設定値が EPAP-ANY-RECOVERABLE でなければ、EPAP-RECOVER の設定値を適用する必要があります。同期発行をサポートするには、コンテキスト・コンテナ内の



EPAP-RECOVER 設定値によって示されているトランスポートのリカバリー可能性要件に応じて、EP アダプターは次のような処理を行う必要があります。

- フィールドが EPAP-RECOVERABLE に設定されている場合、EP アダプターはリカバリー可能な方式でトランスポートへの書き込みを行う必要があります。
- フィールドが EPAP-NON-RECOVERABLE に設定されている場合、EP アダプターはリカバリー可能な方式でトランスポートへの書き込みを行ってはなりません。

リカバリー可能性の設定を順守できない場合は、アダプターを終了させる必要があります。そうしなければ、イベントのトランザクション要件が正しく実装されません。

非同期発行の場合、フィールドは EPAP-ANY-RECOVERABLE に設定されます。

8. イベント記述子コピーブックを使用して、DFHEP.DESSCRIPTOR コンテナからイベント記述子を取得します。

イベント記述子は、同記述子内のデータ項目の数を記述する接頭部、および各データ項目の記述子で構成されます。データ項目記述子には、ビジネス・データ項目の名前およびそのタイプが含まれます。データ項目自体は、DFHEP.CHAR.nnnnnn および DFHEP.DATA.nnnnnn 形式の名前のコンテナ内にあります。ここで、nnnnnn は、キャプチャーしたデータの順序を示す 5 桁の数値から成る連番であり、00001 から始まります。DFHEP.CHAR.nnnnnn コンテナには、イベント指定で要求されたフォーマットが設定された、印刷可能 (文字) 形式のキャプチャー・データが含まれます。

DFHEP.DATA.nnnnnn コンテナには、フォーマットが設定されていないキャプチャー・データが含まれます。

9. DFHEP.DATA.nnnnnn コンテナから必須データ項目を取得します。または、データを自分でフォーマット設定しない場合には、イベント仕様で要求されているとおりにフォーマット設定されているデータを、対応する DFHEP.CHAR.nnnnnn コンテナから取得することができます。

注: データ・キャプチャー対象のデータ項目が存在しなかった場合は、対応する DFHEP.DATA.nnnnnn コンテナが CICS イベント・オブジェクトに含まれません。例えば、キャプチャー仕様で指定しているキャプチャー・データ項目が、イベントを引き起こした API コマンドには存在しないオプション・パラメーターに関連付けられている場合に、この状況が生じる可能性があります。代わりに、DFHEP.ERR.nnnnnn コンテナが CICS イベント・オブジェクト内にあり、それにはキャプチャー・データ項目用に指定されたキャプチャー長が含まれています。対応する DFHEP.CHAR.nnnnnn コンテナが CICS イベント・オブジェクト内に含まれますが、それには 1 つ以上のアスタリスクが含まれることになります。

10. データのフォーマットを設定し、イベントを発行します。

DESCRIPTOR 配列内の各項目は、キャプチャーされるソース・データのタイプ、およびデータがフォーマット設定される場合にそのデータに必要な長さとタイプを定義します。

データは、キャプチャー・データ項目仕様に指定された長さまでキャプチャーされます。キャプチャーされるデータは、ソース・データ域にあるものとまったく同じです。キャプチャー・データが取得できない場合、対応する DFHEP.DATA コンテナは作成されません。

フォーマット長「**Automatic (自動)**」(0) は、すべてのデータ型でサポートされます。フォーマット長「**Automatic (自動)**」を使用すると、等価な EPDE フィールドが epde\_formatLen\_auto に設定されます。

フォーマット精度「**Automatic (自動)**」を指定すると、長さは指定されたデータ型と精度を使用するために必要な長さに設定され、この精度はすべての数値データ型でサポートされます。フォーマット精度「**Automatic (自動)**」を使用すると、等価な EPDE フィールドが epde\_formatPrec\_auto に設定されます。

11. EXEC CICS RETURN コマンド、または EXEC CICS ABEND コマンドを発行して、処理を完了します。

発行されるイベントのほかに、カスタム EP アダプターは成功または失敗の標識を生成する必要があります。イベントを発行できないカスタム EP アダプターは、異常終了コードを出して終了する必要があります。これにより、CICS は必要なリカバリー・アクションを開始でき、該当する統計値を増やすことができます。

## COBOL 言語によるコード・フラグメントの例

カスタム EP アダプター DFH0EPAC サンプルから取得されたこのコード・フラグメントは、この手順で説明されている一連のステップを示しています。ここには、EP アダプター 情報またはデータ項目の処理を含めていません。サンプル全体は、CICS TS 56 .CICS.SDFHSAAMP ライブラリーにあります。

```
*****
Linkage section.
*****
01 EPContext.
   copy dfhepcxo.
01 EPDescriptor.
   copy dfhepdeo.
01 EPAdapter      pic x(16).
01 EPAdaptparm
   copy dfhepapo.
01 EPData         pic x(32000).
*****
Main-program section.
*****
*
*   perform Initial-processing.
*
*   Process the data items
*   perform Process-data-item
*       varying ItemNum from 1 by 1
*       until ItemNum > epde-itemcount.
*
*****
* Any final EVENT PROCESSING code to go here
*****
*
*   Return to caller
*   EXEC CICS RETURN END-EXEC.
*
*   Main-program-exit.
*   exit.
*
*****
Initial-processing section.
*****
*
*   Obtain the DFHEP.CONTEXT container
*   EXEC CICS GET CONTAINER('DFHEP.CONTEXT')
*       SET(address of EPContext)
*       FLENGTH(EPContextLength)
*
*   END-EXEC.
*
*   Obtain the DFHEP.DEScriptor container
*   EXEC CICS GET CONTAINER('DFHEP.DEScriptor')
*       SET(address of EPDescriptor)
*       FLENGTH(EPDescriptorLength)
*
*   END-EXEC.
*
*   Obtain the DFHEP.ADAPTER container
*   EXEC CICS GET CONTAINER('DFHEP.ADAPTER')
*       SET(address of EPAdapter)
*       FLENGTH(EPAdapterLength)
*
*   END-EXEC.
*
*   Obtain the DFHEP.ADAPTPARM container
*   EXEC CICS GET CONTAINER('DFHEP.ADAPTPARM')
*       SET(address of EPAdaptparm)
*       FLENGTH(EPAdaptparmLength)
*
*   END-EXEC.
*
*
*   Check the recoverability of the transport is right for the event
*   if not epap-any-recoverable
*       perform Check-recoverability.
*
*   Initial-processing-exit.
*   exit.
*
*****
Process-data-item section.
*****
*****
*
```

```

* Process a data descriptor item
*
* Build the data container name: DFHEP.DATA.nnnnn
string 'DFHEP.DATA.' delimited by size
      ItemNum      delimited by size
      into ContainerName
end-string.
*
* Obtain the DFHEP.DATA.nnnnn container - if present
EXEC CICS GET CONTAINER(ContainerName)
      SET(address of EPData)
      FLNGTH(EPDataLength)
      RESP(Resp) RESP2(Resp2)

END-EXEC.
*****
* Any final code to process DATA ITEM to go here
*****
*
* Convert the data according to epde-datatype
perform Convert-data.
*
* Calculate the target field length
perform Calculate-length.
*
* Format the data according to epde-formattype
perform Format-data.
*
* Move over the data item ready for the next one
add TSQFieldLength to TSQFieldIndex.
*
Process-data-item-exit.
exit.
*
*****

```

## トランザクション再始動プログラムの作成

トランザクション再始動ユーザー置換可能プログラム (DFHREST) を作成して、トランザクションの再始動に関する決定に加わらせることができます。

トランザクションのリソース定義で RESTART(YES) が指定されていると (デフォルトは RESTART(NO))、CICS は、トランザクションの異常終了時に DFHREST を呼び出します。

デフォルトのプログラムは、一定の条件のもとで再始動を要求します。例えば、プログラム分離デッドロックの場合 (つまり、2 つのタスクがあり、どちらも相手側が特定の DL/I データベース・セグメントやファイル・レコードを解放するのを待っている状況) は、いずれかのタスクがバックアウトされ、自動的に再始動されることで、もう一方のタスクが更新を完了できるようになります。

トランザクションの再始動に関する一般情報については、[リカバリー処理のためのトラブルシューティング](#)を参照してください。

注:

1. トランザクション再始動プログラムがトランザクションを再始動することにした場合は、そのトランザクションの初期プログラムを呼び出すための新しいタスクが追加されます。これは、そのタスクが 2 つ目以降の UOW で異常終了して、DFHREST が再始動を要求した場合も該当します。
2. 再始動の総数の統計はトランザクションごとに保持されます。
3. 緊急時再始動の場合は、タスクの再始動は発生しません。
4. 場合によっては、トランザクション再始動の代わりに SYNCPOINT ROLLBACK コマンドを使用して、同様のメリットを実現することもできます。ROLLBACK コマンドの使用は通常は推奨されませんが、アプリケーション・プログラムのすべての実行可能コードを保持できるという利点があります。

デフォルトの DFHREST の置換を計画している場合は、再始動に適さないロジックのトランザクションがないかどうかを確認してください。

- 1 つの作業単位として実行するトランザクションは安全です。ループを実行するトランザクションも、それぞれのパスでリカバリー可能な宛先から 1 つのレコードを読み取り、その他のリカバリー可能リソースを更新し、同期点で閉じるのであれば、安全です。

- ・ 異常終了の前兆となる、作業単位における誤った処理の反復が発生する状況为了避免するために変更しなければならないトランザクションには、2つのタイプがあります。
  1. 最初の作業単位とそれ以降の作業単位で、それぞれ別のリソースを変更するトランザクション。
  2. 入力データ域の内容を複数の作業単位で使用するトランザクション。
- ・ 基本機能が APPC リンクになっている分散トランザクションは、トランザクション再始動の候補にしないでください。会話の相手側がまだアクティブな状況でバックエンド・トランザクションまたはフロントエンド・トランザクションを再始動すると、正しいエラー処理や会話状態のリカバリーに関して問題が発生します。

CICS がトランザクション再始動プログラムを呼び出すには、以下のすべての条件が満たされている必要があります。

- ・ トランザクションが異常終了すること。
- ・ トランザクション終了時の暗黙的な同期点のコミット・ポイントに達する前に、トランザクションの強制終了が検出されること。
- ・ トランザクションが再始動可能であることがトランザクション定義で定義されていること。
- ・ トランザクションが基本機能に関連付けられていること。

上記の条件が満たされていると、CICS はトランザクション再始動プログラムを呼び出し、そのトランザクション再始動プログラムがトランザクションの再始動を要求するかどうかを決定します。CICS がその決定をオーバーライドすることもあります (動的バックアウトが失敗した場合など)。トランザクション再始動プログラムが異常終了した場合も、トランザクションは再始動されません。

上記の条件が満たされていないと、CICS はトランザクション再始動プログラムを呼び出さないため、トランザクションは再始動されません。

## DFHREST 通信域

CICS 提供のデフォルトのトランザクション再始動プログラムはアセンブラーで書かれており、以下の処理のためのロジックを含んでいます。

- ・ CICS から渡される通信域をアドレッシングする
- ・ トランザクション再始動を要求するかどうかを決定する
- ・ 再始動が要求される場合はメッセージを CSMT に送信する
- ・ EXEC CICS RETURN コマンドを使用して制御を CICS に戻す

通信域は XMRS\_COMMAREA DSECT によってマップされます。これは、DFHXRSD コピーブックで提供されています。C、COBOL、および PL/I の場合の相当する構造体が、コピーブック DFHXRSH、DFHXRSD、および DFHXRSP にそれぞれ収められています。

通信域で渡される情報は、以下のとおりです。

### XMRS\_FUNCTION

この再始動プログラム呼び出しの機能コードを 1 バイト・フィールドで示します。これは常に 1 に設定されていて、XMRS\_TRANSACTION\_RESTART と等価になります。つまり、トランザクション再始動を処理するために DFHREST が呼び出されるということです。

### XMRS\_COMPONENT\_CODE

呼び出し元のコンポーネント・コードを 2 バイト・フィールドで示します。これは常に XM に設定されていて、XMRS\_TRANSACTION\_MANAGER と等価になります。トランザクション・マネージャーは、トランザクションを再始動するかどうかの決定を調整する CICS コンポーネントです。

### XMRS\_READ

初期入力 of 目的以外の端末読み取り要求をトランザクションが発行したかどうかを 1 バイト・フィールドで示します。

このパラメーターの同等値は、以下のとおりです。

**XMRS\_READ\_YES**

トランザクションによって端末読み取りが行われたことを意味します。

**XMRS\_READ\_NO**

端末読み取りが行われなかったことを意味します。

**XMRS\_WRITE**

端末書き込み要求をトランザクションが発行したかどうかを 1 バイト・フィールドで示します。

このパラメーターの同等値は、以下のとおりです。

**XMRS\_WRITE\_YES**

トランザクションによって端末書き込みが行われたことを意味します。

**XMRS\_WRITE\_NO**

トランザクションによって端末書き込みが行われなかったことを意味します。

**XMRS\_SYNCPOINT**

トランザクションが同期点を実行したかどうかを 1 バイト・フィールドで示します。

このパラメーターの同等値は、以下のとおりです。

**XMRS\_SYNCPOINT\_YES**

1 つ以上の同期点が行われたことを意味します。

**XMRS\_SYNCPOINT\_NO**

同期点が行われなかったことを意味します。

**XMRS\_RESTART\_COUNT**

トランザクションが再始動された回数を符号なしハーフワード・バイナリー値として示します。

トランザクションが再始動されていない場合は、ゼロになります。 トランザクション定義を対象とした再始動の累計では**ありません**。正しくは、例えば、単一のオペレーター入力を処理しようとしているトランザクションを対象とした再始動の累計です。

**XMRS\_ORIGINAL\_ABEND\_CODE**

トランザクションによって記録された最初の異常終了コードを提供します。

**XMRS\_CURRENT\_ABEND\_CODE**

現在の異常終了コードを提供します。元の異常終了コードと現在の異常終了コードで値が異なることがあります。例えば、トランザクションが異常終了を処理し、その後異常終了した場合などです。

**XMRS\_RESTART**

トランザクションの再始動を CICS に要求するかどうかを示すためにトランザクション再始動プログラムが設定する 1 バイト出力フィールドです。

このフィールドの同等値は、以下のとおりです。

**XMRS\_RESTART\_YES**

再始動を要求します。

**XMRS\_RESTART\_NO**

再始動を要求しません。

## CICS 提供のトランザクション再始動プログラム

CICS 提供のデフォルトのトランザクション再始動プログラムは、以下の場合にトランザクションの再始動を要求します。

1. トランザクションが端末読み取り (初期入力データの読み取り以外)、端末書き込み、または同期点を実行しなかった。 **かつ**、
2. 再始動回数が 20 (再始動回数の限度) より少ない。 **かつ**、
3. 現在の異常終了コードが次のいずれかである。
  - ADCD: DBCTL デッドロックのためにトランザクションが異常終了したことを示す
  - AFCE: ファイル制御で検出されたデッドロックのためにトランザクションが異常終了したことを示す

- AFCW: VSAM で検出されたデッドロック (RLS のみ) のためにトランザクションが異常終了したことを示す

注: RETURN TRANSID CHANNEL() で始まる疑似会話型トランザクションは、再始動できません。

CICS 提供のデフォルトのトランザクション再始動プログラム DFHREST のソースが、アセンブラー言語でのみ CICSTS56.CICS.SDFHSAMP ライブラリーに用意されています。

通信域をマップするためのアセンブラー・コピーブックが、CICSTS56.CICS.SDFHMAC ライブラリーにあります。

## 端末エラー・プログラムの作成

CICS 端末エラー・プログラム (TEP) は、順次アクセス方式の装置のエラー状態を処理します。z/OS Communications Server に対応した装置では、端末エラー・プログラムを使用できません。z/OS Communications Server の場合は、ノード・エラー・プログラムを使用してください。

CICS には、独自のプログラムの基礎として使用できるサンプル端末エラー・プログラムが用意されています。

## 順次装置のエラー処理に関する背景情報

CICS 端末エラー処理によって、端末エラーへの対応として CICS 操作を変更することが可能になります。CICS でありとあらゆる処理を想定することはできないので、エラー処理機能は、ユーザーが端末ネットワークで発生するエラーに合わせた独自の解決方法を自由に実施するための柔軟性を最大限に持たせた設計になっています。

順次装置の使用時に発生したエラーの検出と修正のために、以下の CICS コンポーネントが呼び出されます。

- 端末管理プログラム (DFHTCP)
- 端末異常条件プログラム (DFHTACP)
- 端末エラー・プログラム (DFHTEP)

これらのコンポーネントについては、以後のセクションで説明します。論理装置に対応する CICS コンポーネントについては、[107 ページの『ノード・エラー・プログラムの作成』](#)を参照してください。

## 異常条件が発生した場合

特定の端末や回線に関連した異常条件が発生すると、端末管理プログラムがその端末をサービス休止状態にして、端末異常条件プログラム (DFHTACP) に制御を渡します。それを受けて端末異常条件プログラムは、特定バージョンの端末エラー・プログラム (DFHTEP、CICS 提供のプログラムかユーザー作成のプログラム) に制御を渡し、そのプログラムが適切な処理を実行します。

## 端末管理プログラム

エラーが検出された端末がサービス休止状態になると、端末管理プログラムが端末異常条件回線項目 (TACLE) を作成します。その項目は、実際の項目 (つまり、エラーが発生した回線の端末管理テーブル回線項目 (TCTLE)) からチェーニングされた項目です。TACLE には、エラーに関する情報が含まれています。

## 端末異常条件プログラム

TACLE が確立されると、DFHTACP を実行するタスクが端末管理プログラムによって追加され、エラーが発生した実際の回線項目 (TCTLE) へのポインターが渡されます。基本的なエラー分析を実行してデフォルトの処理を確立すると、DFHTACP は DFHTEP に制御を渡し、通信域 (DFHTEPCA) を渡します。その結果、DFHTEP は、エラーを調べて代替処理を実行できるようになります。



通信域からは、エラーを正しく評価するために必要なすべてのエラー情報にアクセスできます。通信域には、DFHTACP によってすでに設定されているデフォルトの処理を変更するために操作できる特殊なアクション・フラグも入っています。

DFHTEP は、必要な機能を実行すると、EXEC CICS RETURN コマンドを実行して制御を DFHTACP に戻します。その後 DFHTACP が通信域にあるアクション・フラグで指定されている処理を実行すると、エラー処理タスクは終了します。

注：DFHTACP が処理を実行する前に 1 つの回線に存在するエラーが 8 つを超えると、その回線はサービス休止状態になります。これはシステム・パフォーマンスの低下を避けるためです。

## 端末エラー・プログラム

端末エラー・プログラムは、端末管理プログラムによって検出された端末エラーまたは回線エラーの原因を分析します。CICS 提供バージョン (サンプル端末エラー・プログラム、DFHXTEP) は、汎用の基本的なリカバリー処理を実行する設計になっています。アプリケーションに合わせた具体的なリカバリー処理を実行するために、ユーザー作成バージョンのプログラムを用意することもできます。

端末異常条件プログラムがユーザー作成の端末エラー・プログラムをリンクする方法は、CICS 提供バージョンの場合と同じです。エラーに関連する情報が、通信域と TACLE に取り込まれます。

サンプル端末エラー・プログラムの生成のために用意されているマクロについては、以下の各セクションで取り上げます。主なステップは、サンプル DFHTEP モジュールとテーブルの生成です。それぞれ、DFHTEPM マクロと DFHTEPT マクロを使用します。このサンプル・プログラムで該当するオプションを選択することも、サンプル・プログラムを元に独自のバージョンを作成することも可能です。

CICS 提供のサンプル端末エラー・プログラム (DFHXTEP) の説明や、ユーザー作成バージョンを生成するためのアドバイスについては、以下のサブセクションを参照してください。

## 通信域

通信域は、サンプル DFHTEP が使用する基本的なインターフェースであり、ユーザー作成の DFHTEP でも使用できます。その目的を以下にまとめます。

- TACLE のアドレッシング。
- DFHTACP に戻る時に実行する処理の指定。

DFHTACP は、DFHTEP に制御を渡す前に、実行するデフォルトの処理を確立します。実行するデフォルトの処理は、検出されたエラー状態によって異なります。デフォルトの処理は、1 バイトの通信域フィールド TEPCAACT のビット設定で指定します。通信域のフィールドやデフォルトの処理やビット設定の詳細については、[100 ページの『端末エラー・プログラムの作成』](#)を参照してください。

## 端末異常条件回線項目 (TACLE)

TACLE には、エラーのタイプやエラー状態の端末のタイプに関する詳細情報が入ります。

検出されたエラー状態を示すコードが TACLE の TCTLEPFL というラベルの 1 バイト・フィールドで DFHTEP に渡されます。

端末異常条件回線項目 (TACLE) DSECT の形式の説明については、[100 ページの『端末エラー・プログラムの作成』](#)を参照してください。

## サンプル端末エラー・プログラム

CICS には、サンプル端末エラー・プログラム (TEP) が用意されており、端末エラーを処理するための汎用的なプログラム構造としてそのサンプルを利用できます。

ソース・コード形式のサンプル TEP (DFHXTEP) は、アセンブラ言語のみの提供ですが、独自の端末エラー・プログラムは、CICS でサポートされているどの言語でも作成できます。

DFHXTEP は、アセンブル後に DFHTEP としてリンク・エディットされます。ユーザー置換可能プログラムのアセンブルとリンク・エディットに必要なジョブ制御ステートメントに関する情報は、[337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』](#)を参照してください。

用意されているデフォルト・オプションでサンプル 端末エラー・プログラムを生成して使用することもできますが、オペレーティング環境のニーズに合わせて該当する生成オプションや変数を選択して、端末エラー・サポートをカスタマイズすることもできます。それぞれのエラー状態は別々のルーチンで処理されるので、サンプル TEP の生成時に CICS 提供のルーチンユーザー作成のルーチンに置き換えることも可能です。

### サンプル端末エラー・プログラムのコンポーネント

サンプル 端末エラー・プログラムには、端末エラー・プログラム自体と 2 つの端末エラー・プログラム・テーブルが入っています。

- TEP エラー・テーブル
- TEP デフォルト・テーブル

どちらのテーブルでも、サンプル DFHTEP で制御したり対応したりするさまざまなエラー状態のしきい値が定義されています。このしきい値は、サンプル DFHTEP が DFHTACP のデフォルトの処理を受け入れる前に、特定の端末で特定タイプのエラーが何回まで発生してよいか、という値です。オプションとして、一定の時間間隔に基づいて発生回数を制御して処理を実行することもできます (例えば、1 時間以内に特定タイプのエラーの発生回数が 3 回を超えると、端末をサービス休止状態にする、といった具合です)。

### TEP エラー・テーブル

端末エラー・プログラム (TEP) のエラー・テーブルには、端末で発生したエラーに関する情報が保持されます。

このテーブルは次の 2 つの部分から構成されます ([84 ページの図 17](#) を参照)。

- TEP エラー・テーブル・ヘッダー (TETH)。TEP エラー・テーブル・コンポーネントのロケーションとサイズに関連するアドレスと定数が含まれます。
- 端末エラー・ブロック (TEB)。次のいずれかです。
  - 永続 (P-TEB)。それぞれが特定の端末に関連付けられます。
  - 再使用可能 (R-TEB)。特定の端末に永続的に関連付けられることはありません。

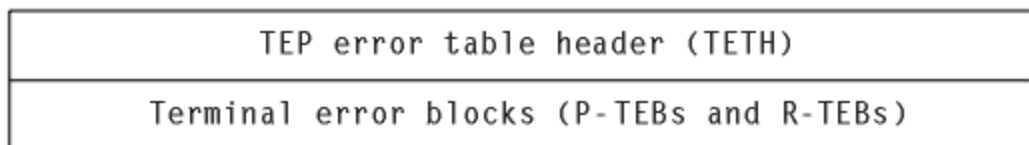


図 17. TEP エラー・テーブル

TEB には、端末に関連付けられたエラー情報が保持されます。生成される TEB の総数を指定する必要があります。必要な最大数は、端末ごとに 1 つです。その場合、TEB は永続的です。

特定の端末に永続的に関連付けられない再使用可能な TEB のプールを割り振ることで、TEB として使用されるストレージの総量を削減できます。再使用可能な TEB は、端末に関連付けられたエラーが初めて発生したときに動的に割り当てられ、対応するエラー・プロセッサがその端末をサービス休止にしたときに、再利用のために解放されます。

**注：**プールの大きさは、あらゆる時点において、未解決のエラーが予期される端末の最大数に対応できる十分な大きさにしてください。プールの制限を超えると、端末エラーの処理が断続的になります。この状況についての警告は発行されません。

ネットワークにとって重要な端末には、永続的な TEB を割り当てる必要があります。ネットワークの残りについては、再使用可能な TEB のプールを生成できます。

現在使用中の各 TEB、または永続的に割り当てられた各 TEB には、[85 ページの図 18](#) に示すように、端末のシンボル端末 ID、および 1 つ以上のエラー状況エレメント (ESE) が含まれています。

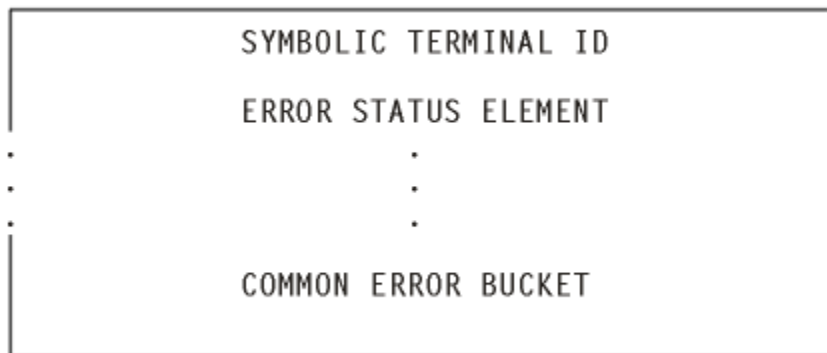


図 18. 端末エラー・ブロック (TEB)

ESE は、端末に関連付けられた特定のタイプのエラーの発生を記録します。エラー状況エレメントの内容は、TEPCD の DSECT (DFHTEPM TYPE=INITIAL マクロによって生成されます) のコメント「ERROR STATUS ELEMENT FORMAT」の下に表されています。TEB 1 つあたりの ESE の数は、すべての TEB で同じです。この数は、TEP テーブルを生成するときに指定します。DFHTACP (25) で認識されるエラー・タイプの最大数より少ない数を指定した場合は、「共通エラー・バケット」という追加の ESE が TEB ごとに 1 つ生成されます。

特別なエラー・タイプのために、各 TEB で ESE スペースを永続的に予約することができます。永続的に予約されなかったものは再使用可能と見なされ、端末に関連付けられた特定のエラー・タイプが初めて発生したときに動的に割り当てられます。ESE で現在表されていないエラー・タイプが発生した場合、および、すべての再使用可能な ESE が他のエラー・タイプに割り当てられた場合、そのエラーの発生は、共通エラー・バケットに記録されます。DFHTACP は、標準的な端末ネットワークで発生する可能性があるエラー・タイプよりもはるかに多くのエラー・タイプを認識できます。最大値より小さい値を指定して、サンプル DFHTEP で ESE を動的に割り当てるようにすれば、テーブルのサイズを最小化しながらも、ネットワークに関連するエラー・タイプを制御して記録することができます。

### TEP デフォルト・テーブル

端末エラー・プログラム (TEP) のデフォルト・テーブルには、制御および記録するエラーのタイプごとに「数と時間」のしきい値が含まれています。

デフォルト・テーブルの最初にあるインデックス配列には 2 つの機能があります。インデックスの値が正の場合、そのエラー・コードには各 TEB に永続的に定義された ESE があります。また、インデックスの値は、その予約された ESE に対する変位を表しています。インデックスの値が負の場合は、過去の発生によってまだ ESE が作成されていなければ、再使用可能な ESE から動的に ESE を割り当てる必要があります。負のインデックスの値の補数は、TEP のデフォルト・テーブルで保持されるエラー・タイプのしきい値に対する変位です。

### サンプル端末エラー・プログラムの構造

サンプル端末エラー・プログラム (DFHXTEP) の構造は、以下の 6 つの主要領域に分割できます。

1. 入り口と初期設定
2. 端末の識別とエラー・コードの検索
3. エラー・プロセッサの選択
4. エラー処理の実行
5. 一般的な出口
6. 共通のサブルーチン

これらの領域について、以下の各セクションで詳しく説明します。

サンプル端末エラー・プログラムの構造の概要を [88 ページの図 19](#) に示します。

## 入り口と初期設定

入り口で、サンプル TEP は DFHEIENT を使用して、基底レジスターを設定し、EXEC インターフェース・コンポーネントに対するアドレッシング機能を取得します。

このサンプルは、EXEC CICS ADDRESS COMMAREA を使用して、DFHTACP から渡された通信域に対するアドレッシング機能を取得します。また、EXEC CICS ADDRESS EIB コマンドを使用して、EXEC インターフェース・ブロックに対するアドレッシング機能を取得します。通信域から TACLE のアドレスを取得し、EXEC CICS LOAD を使用して TEP へのアクセスを確立します。時間のサポートを生成した場合は、それ以降の処理のエラーにタイム・スタンプが設定されます (現在時刻が通信域に渡されます)。システムが初期設定された後、初めてサンプル TEP の入り口が開始されると、TEP テーブルが初期設定されます。

## 端末 ID とエラー・コードの検索

一般的な入り口処理の後には、エラーと関連付けられた端末のための端末エラー・ブロック (TEB) がないか、TEP エラー・テーブルがスキャンされます。

一致するエントリが見つからない場合は、新しい TEB が作成されます。すべての TEB が現在使用されている場合 (再使用可能な TEB が存在しない場合)、デフォルトのアクションでは、処理が終了され、RETURN 要求が発行され、制御権が DFHTACP に戻されます。

端末の TEB が検出または作成されると、その TEB のエラー状況エレメント (ESE) で同じようなスキャンが実行され、現在処理中のエラーのタイプが過去に発生したかどうか、また、そのエラーのタイプのために永続的に予約された ESE スペースがあるかどうかを特定します。関連付けられた ESE が検出されない場合、再使用可能な ESE からそのエラー・タイプのために ESE が割り当てられます。再使用可能な ESE が存在しない場合、そのエラーは端末の共通エラー・バケットに記録されます。適切な制御域 (TEB および ESE) のアドレスが、適切なエラー・プロセッサでできるようにレジスターに配置されます。

## エラー・プロセッサの選択

ユーザー指定のメッセージの選択肢の中からメッセージが選択され、それらのメッセージが、指定された一時データ宛先に書き込まれます。エラー・コードのタイプをテーブルのインデックスとして使用して、そのタイプのエラーを処理するエラー・プロセッサのアドレスが決定されます。

エラー・コードが無効な場合、または、そのエラーのタイプを処理するようにサンプル TEP が生成されなかった場合、デフォルト・アクションでは、オプションでエラー・メッセージを生成して DFHTACP に制御権を返すルーチンのアドレスが指定されます。有効なエラー・プロセッサのアドレスがテーブルから取得された場合は、そのルーチンに制御権が渡されます。

## エラー処理の実行

各エラー・プロセッサの機能は、特定のエラーに対して DFHTACP で設定されたデフォルトのアクションを実行するか、それともエラー・プロセッサで設定されたアクションを実行するかを決定します。

共通エラー・バケットは、特定のエラー・プロセッサによって処理されます。ただし、共通エラー・バケットのしきい値を使用して、制限に達したかどうか判断されます。エラー・プロセッサが判断できるように、サンプル TEP には、特定の端末に関連付けられたエラーごとにカウントと時間のしきい値の合計を維持するサブルーチンが用意されています。また、エラーの状況、およびエラー・プロセッサによって実行されたリカバリー・アクションの状況をログに記録するサブルーチンも用意されています。

サンプル TEP に用意されているエラー・プロセッサは、ユーザー作成のエラー・プロセッサに置換できます。レジスターのリンケージ規則、エラー状況、DFHTACP のデフォルトのアクション、およびサンプル TEP のエラー・プロセッサのアクションについて、サンプル DFHXTEP のソース・リスト中にコメントで説明しています。ただし、多くの場合、TEP テーブルを生成するときにしきい値を変更して、サンプル DFHXTEP のアクションを変更することができます。

## 汎用出口ルーチン

各エラー・プロセッサは、端末のサービスを継続するかどうかを決定する汎用出口ルーチンに制御権を渡します。

端末をサービス休止にすると、端末エラー・ブロックおよびその端末のすべてのエラー状況エレメントが TEP エラー・テーブルから削除されます (その端末が永続エントリとして定義されている場合を除く)。端末のサービスを再開し、その後にエラーが発生した場合、新しい端末エラー・ブロックが割り当てられます。

## 共通のサブルーチン

サンプル DFHXTEP には、エラー・プロセッサで使用するためのいくつかのサブルーチンが用意されています。各サブルーチンの入り口には、「TEPxxxx」 という形式のラベルが付いています。「xxxx」 はサブルーチンの名前です。

. サブルーチン内のすべてのラベルは TEPx で始まります。「x」 はサブルーチンの名前の先頭文字です。すべてのサブルーチンは、サブルーチン・セクションのモジュール内にアルファベット順に配置されています。レジスター規則とサブルーチンの使用方法を、ソース・リストの各サブルーチンの最初にコメントとして記載しています。

次のサブルーチンを、独自のエラー・プロセッサを作成するために使用できます。

### TEPACT

プログラムの生成時に適切な PRINT オプションを選択した場合は、これを使用して、DFHTACP およびサンプル DFHTEP によって通信域のフィールド **TEPCA**ACT に設定されたアクション・ビットの名前を出力できます。

### TEPDEL

エラー・プロセッサからの出口で、TEP エラー・テーブルから、特定の端末の端末エラー・ブロックとエラー状況エレメントを削除できます。

### TEPHEXCN

これを TEPPUTTD で使用して、4 ビットの 16 進値を 8 ビットの印刷可能な同等の値に変換することができます。

### TEPINCR

特定の端末のエラー状況エレメントで維持されているカウントと時間のしきい値の合計を更新およびテストできます。

### TEPLOC

特定の端末 ID 用の端末エラー・ブロックとエラー状況エレメントを検索または割り当てることができます。

### TEPPUTTD

ユーザー定義の一時データ宛先に文字または 16 進データを出力できます。

### TEPTMCHK

これを TEPINCR で使用して、時間のしきい値が経過したかどうかを判断することができます。

### TEPWGHT

特定の端末のエラー状況エレメントで維持されている重み/時間のしきい値を更新できます。



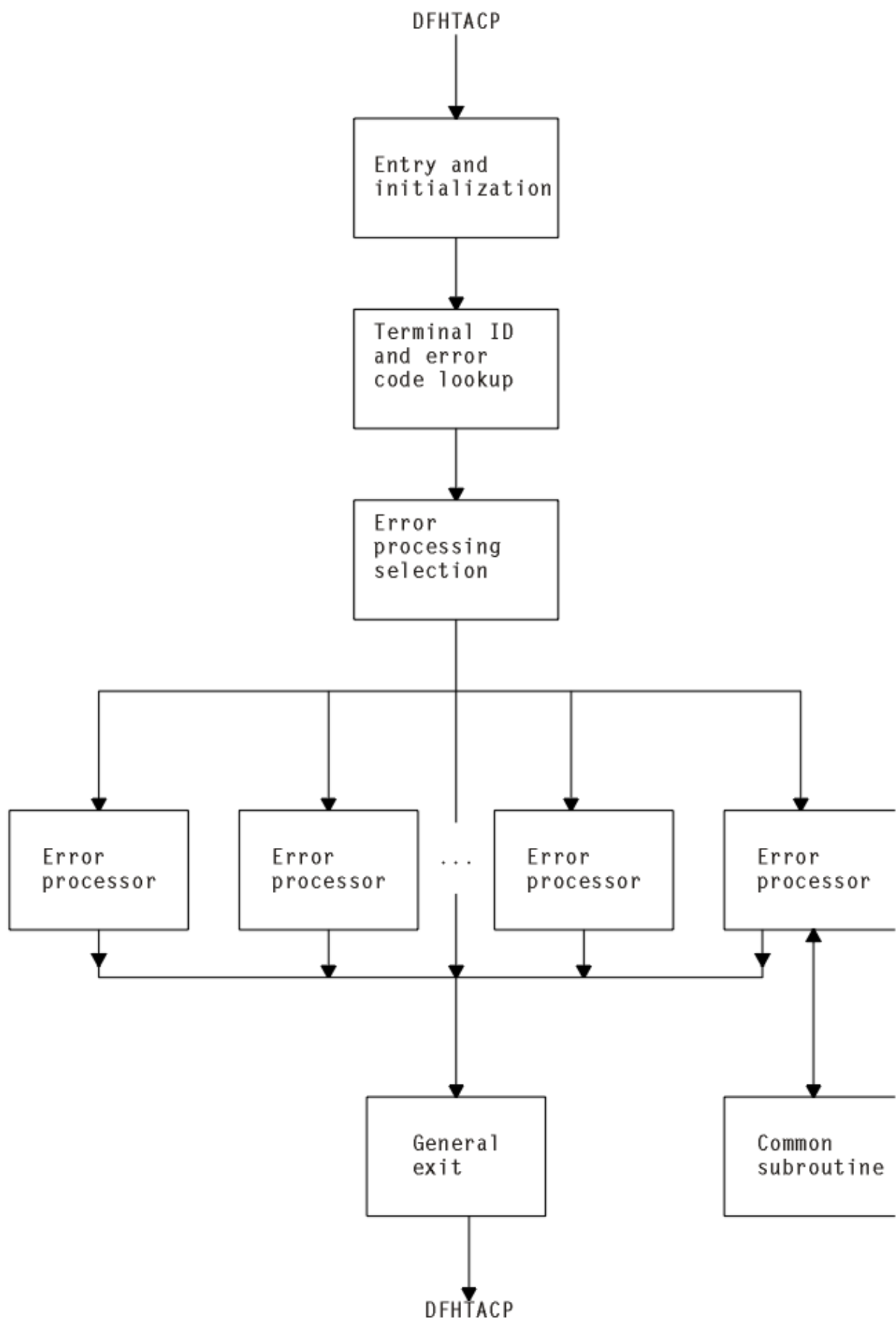


図 19. サンプルの端末エラー・プログラム (DFHXTEP) の概要



## サンプル端末エラー・プログラムのメッセージ

一時データ宛先 CSMT (または DFHTEPM TYPE=INITIAL の OPTIONS オペランドで指定されている宛先) のログに記録されるメッセージには 6 つのタイプがあり、それぞれ固有のメッセージ接頭語によってタイプを識別できます。

DFHTEPM TYPE=INITIAL の PRINT オペランドで指定するパラメーターによって、各タイプのメッセージの選択を制御できます。

以下のメッセージがあります。

### DFHTEP、ERROR – エラー・テキスト

DFHTEP モジュールの生成時に、PRINT パラメーターで ERRORS が指定されました。NOERRORS オプションを使用すれば、このメッセージを抑止できます。エラー・テキストは、以下のいずれかです。

#### Unsupported error code, “xx”

DFHTACP によって DFHTEP に表示されたエラー・コードは、DFHTEP にとって不明です。

#### “DFHTEPT” not defined in system

DFHTEP テーブルをストレージにロードできませんでした。

#### Unknown error status message, “xxxx”

3270 タイプのリモート装置から表示されたエラー状況メッセージをデコードできませんでした。

これらは、発生するべきではないエラーです。

### DFHTEP、ACTION – アクション・フラグ名

DFHTEP モジュールの生成時に、PRINT パラメーターで TACPACTION と TEPACTION のいずれかまたは両方が指定されました。両方が指定された場合は、DFHTEP の呼び出しごとに、このメッセージがログに 2 回記録されます。最初のメッセージでは、DFHTEP への入り口で DFHTACP によって設定されたアクション・フラグが表示されます。2 つ目のメッセージでは、エラー処理後に DFHTEP によって DFHTACP に返されるアクション・フラグが表示されます。NOTACPACTION オプションと NOTEPACTION オプションを使用すれば、これらのメッセージを抑止できます。

アクション・フラグの名前と説明を以下にまとめます。DFHTACP によって実行される処理の詳細については、100 ページの『通信域の内容のアドレッシング』にある TEPCAACT フィールドの説明を参照してください。

#### フラグ名 説明

##### LINEOS

回線をサービス休止状態にします。

##### NONPRGT

端末にページ不能タスクが存在します。

##### TERMOS

端末をサービス休止状態にします。

##### ABENDT

端末でタスクを異常終了させます。

##### ABORTWR

書き込みを打ち切り、端末のストレージを解放します。

##### RELTIOA

着信メッセージを解放します。

##### SIGNOFF

端末をサインオフします。

### DFHTEP、TID - tid

DFHTEP モジュールの生成時に、PRINT パラメーターで TID が指定されました。このメッセージには、エラーに関連した装置のシンボリック端末 ID が入ります。NOTID オプションを使用すれば、このメッセージを抑止できます。

### DFHTEP、DECB - DECB 情報

DFHTEP モジュールの生成時に、PRINT パラメーターで DECB が指定されました。この 2 行のメッセージには、エラーの原因になった端末の DECB が入ります (16 進形式で出力されます)。DECB は、TACLE

に入っています (変位 +16 [10 進数])。100 ページの『[端末エラー・プログラムの作成](#)』にある TACLE DSECT の項を参照してください。NODECB オプションを使用すれば、このメッセージを抑止できます。

#### DFHTEP、TACLE - TACLE 情報

DFHTEP モジュールの生成時に、PRINT パラメーターで TACLE が指定されました。このメッセージは、16 進形式で出力されます。DFHTACP によって DFHTEP に渡される TACLE の最初の 16 バイトが入ります。100 ページの『[端末エラー・プログラムの作成](#)』にある TACLE DSECT の項を参照してください。NOTACLE オプションを使用すれば、このメッセージを抑止できます。

#### DFHTEP、ESE - ESE 情報

DFHTEP モジュールの生成時に、PRINT パラメーターで ESE が指定されました。このメッセージには、エラー状況エレメントが入ります。NOESE オプションを使用すれば、このメッセージを抑止できます。

ESE の長さは、TEP テーブルの生成時に TIME オプションを指定したかどうかに応じて、6 バイトか 12 バイトのいずれかになります。形式は以下のとおりです。

表 6. DFHTEP、ESE メッセージのエラー状況エレメントの形式 (NOTIME を指定した場合)		
表示	長さ (バイト)	意味 - NOTIME を指定した場合
0	2	エラーしきい値のカウンターまたは重みの値 (2 進形式)
2	2	現在のエラー件数または重みの値 (2 進数)
4	1	エラー・コード
5	1	使用されません。

表 7. DFHTEP、ESE メッセージのエラー状況エレメントの形式 (TIME を指定した場合)		
表示	長さ (バイト)	意味 - TIME を指定した場合
0	5	エラーしきい値のカウンターまたは重みの値 (2 進形式)
5	3	時刻のしきい値 (100 分の 1 秒単位)
8	4	このエラーが最初に発生した時刻。時刻の表記は 2 進整数になります (100 分の 1 秒単位)。

#### サンプル端末エラー・プログラムの生成

サンプル端末エラー・プログラムとサンプル端末エラー・テーブルの生成方法については、[337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』](#)を参照してください。

サンプル・プログラムとサンプル・テーブルには、端末エラーに対するデフォルトのエラー処理が用意されています。用意されているエラー・プロセッサをユーザー作成のエラー・プロセッサに置き換える場合は、DFHTEPM マクロと DFHTEPT マクロを使用して、ユーザー作成のルーチンを組み込んだサンプル・エラー・プログラムとサンプル・エラー・テーブルを生成する必要があります。DFHTEPM マクロと DFHTEPT マクロで指定するいくつかのパラメーターは、相互に関連しているので、互換性を確保するために慎重に使用してください。

サンプル端末エラー・プログラム (DFHXTEP) を使用する場合は、CEDA INSTALL GROUP (DFHSTAND) コマンドを使用して、必要なプログラム定義とトランザクション定義を生成できます。

#### サンプルの端末エラー・プログラムを生成するためのジョブ制御

サンプルの端末エラー・プログラム (TEP) の生成は、2 つの個別のアセンブリーとリンク・エディットの手順から構成されます。1 つはサンプル TEP モジュール自体を作成するための手順で、もう 1 つは TEP テーブルを作成するための手順です。

コンポーネントのリンク・エディットに使用する必要がある名前を次に示します。

##### DFHTEP

DFHXTEP からアセンブルするサンプル TEP モジュール。

##### DFHTEPT

DFHXTEPT からアセンブルするサンプル TEPT テーブル。

ユーザー置換可能プログラムのアセンブルとリンク・エディットに必要なジョブ制御ステートメントに関する情報は、337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』を参照してください。

### DFHTEPM – サンプル DFHTEP モジュールの生成

サンプル DFHTEP モジュールは、次のマクロによって生成されます。

- DFHTEPM TYPE=USTOR – ユーザー・ストレージ定義の始まりを示します。
- DFHTEPM TYPE=USTOREND – ユーザー・ストレージ定義の終わりを示します。
- DFHTEPM TYPE=INITIAL – CICS DSECT の出力の制御、オプション・ルーチンの提供、およびエラーが発生したときにログに記録する情報のタイプの指定を行います。
- DFHTEPM TYPE=ENTRY – ユーザーの「入り口」ルーチンをコーディングします。
- DFHTEPM TYPE=EXIT – ユーザーの「出口」ルーチンをコーディングします。
- DFHTEPM TYPE=ERRPROC – これを使用すると、サンプルの端末エラー・プログラムに用意されているエラー・プロセッサを、ユーザー作成のバージョンに置換できます。
- DFHTEPM TYPE=FINAL – サンプル DFHTEP モジュールの終わりを示します。

注：これらのマクロを使用する場合は、エラー・プロセッサに変換プログラムのオプション NOPROLOG と NOEPILOG をコーディングする必要があります。

```
DFHTEPM TYPE=USTOR
```

このマクロは、ユーザー・ストレージ定義の始まりを示します。この後にストレージ定義を指定し、その後に DFHTEPM TYPE=USTOREND を指定する必要があります。DFHTEPM TYPE=USTOR を使用してストレージを定義する場合は、これと DFHTEPM TYPE=USTOREND の両方を DFHTEPM TYPE=INITIAL の前にコーディングする必要があります。

```
DFHTEPM TYPE=USTOREND
```

このマクロは、ユーザー・ストレージ定義の終わりを示します。DFHTEPM TYPE=USTOR をコーディングした場合は、これを使用する必要があります。DFHTEPM TYPE=USTOR を使用してストレージを定義する場合は、これと DFHTEPM TYPE=USTOREND の両方を DFHTEPM TYPE=INITIAL の前にコーディングする必要があります。

```
DFHTEPM TYPE=INITIAL
[,DSECTPR={YES|NO}]
[,OPTIONS=( [TD| (TD,destid)|NOTD]
            [,EXITS|,NOEXITS]
            [,TIME|,NOTIME]
            [,PRINT=( [ERRORS|NOERRORS]
                    [,TACPACTION|,NOTACPACTION]
                    [,TEPACTION|,NOTEPACTION]
                    [,TID|,NOTID]
                    [,DECB|,NODECB]
                    [,TACLE|,NOTACLE]
                    [,ESE|,NOESE]))]
```

#### TYPE=INITIAL

サンプル DFHTEP モジュール自体の生成の開始を設定します。

#### DSECTPR={YES|NO}

サンプル DFHTEP アセンブリー・リストの CICS DSECT の出力を制御します。この目的は、リストのサイズを減らすことです。デフォルトは DSECTPR=YES です。

##### YES

DSECT の出力が許可されます。

##### NO

選択した CICS DSECT の出力が抑止されます。このパラメーターを、アセンブラー F で使用してはいけません。

**OPTIONS=optional-routines**

DFHTEP モジュールのオプション・ルーチンを指定または除外します。オプションを 1 つしか指定しない場合でも、括弧が必要です。このオペランドを省略すると、すべてのデフォルト・オプションが生成されます。

**TD|(TD, destid)|NOTD**

エラーに関する情報を一時データ・キューに書き込むかどうかを指定します。

**TD**

一時データ出力ルーチンが生成されます。暗黙の一時データ・キューは CSMT です。

**(TD, destid)**

一時データ出力ルーチンが生成されます。メッセージは、「destid」で指定した一時データ・キュー (これは、TDQUEUE リソース定義を使用して CICS に定義されていなければなりません) に送信されます。

**NOTD**

一時データ・キューにメッセージは書き込まれません。

**EXITS|NOEXITS**

「入り口」および「出口」ユーザー・ルーチンのサポートを含めるかどうかを指定します。

**EXITS**

エラー処理の前後に入り口ルーチンおよび出口ルーチンへの分岐が設定されます。ユーザー・ルーチンを使用しない場合は、ダミー・ルーチンが提供されます。

**NOEXITS**

ユーザー・ルーチンへの分岐が設定されます。

**TIME|NOTIME**

指示した時間間隔でしきい値のテストを制御するかどうかを指定します。例えば、特定のタイプのエラー・インスタンスが 1 時間に 3 回より多く発生した場合に、端末をサービス休止にしたりできます。このパラメーターは、DFHTEPT TYPE=INITIAL マクロの OPTIONS オペランドと同じでなければなりません。

**TIME**

このタイプのしきい値のテストがサポートされます。

**NOTIME**

このタイプのしきい値のテストはサポートされません。

**PRINT=print-information**

エラーが発生するたびに一時データ・キューに記録する情報のタイプを指定します。OPTIONS オペランドで NOTD を指定した場合は、すべての PRINT パラメーターがデフォルトで NO に設定されます。すべての PRINT パラメーターに一時データ出力ルーチンが必要です。パラメーターを 1 つしか指定しない場合でも、括弧が必要です。

**ERRORS|NOERRORS**

サンプル DFHTEP で検出された処理不可能な状況を一時データ・キューに記録するかどうかを指定します。

**ERRORS**

エラー・メッセージがログに記録されます。

**NOERRORS**

エラー・メッセージはログに記録されません。

**TACPACTION|NOTACPACTION**

DFHTACP のデフォルトのアクションを一時データ・キューに記録するかどうかを指定します。

**TACPACTION**

デフォルトのアクションがログに記録されます。

**NOTACPACTION**

デフォルトのアクションはログに記録されません。

**TEPACTION|NOTEPACTION**

サンプル DFHTEP 処理の結果として選択されたアクションを一時データ・キューに記録するかどうかを指定します。

**TEPACTION**

最終アクションがログに記録されます。

**NOTEPACTION**

最終アクションはログに記録されません。

**TID|NOTID**

エラーに関連付けられた端末のシンボル端末 ID を一時データ・キューに記録するかどうかを指定します。

**TID**

端末 ID がログに記録されます。

**NOTID**

端末 ID はログに記録されません。

**DECB|NODECB**

エラーに関連付けられた行の DECB を一時データ・キューに記録するかどうかを指定します。

**DECB**

DECB がログに記録されます。DECB の 16 進表記が 2 つの 24 バイト・メッセージとしてログに記録されます。

**NODECB**

DECB のロギングは実行されません。

**TACLE|NOTACLE**

TACLE 接頭部を一時データ・キューに記録するかどうかを指定します。

**TACLE**

DFHTACP から受け取った 16 バイトの TACLE 接頭部がログに記録されます。

**NOTACLE**

TACLE 接頭部のロギングは実行されません。

**ESE|NOESE**

エラーに関連付けられた ESE を一時データ・キューに記録するかどうかを指定します。

**ESE**

ESE が更新後、および削除前 (アクションが端末をサービス休止にする場合) にログに記録されます。

**NOESE**

ESE のロギングは実行されません。

**ユーザーの入り口ルーチンと出口ルーチンのための DFHTEPM TYPE=ENTRY および EXIT**

サンプルの DFHTEP は、エラー・プロセッサ・ルーチンの作成方法、特にレジスターとサブルーチン・リンケージ規則についてのガイドとして機能します。

ルーチンは、次の制限も遵守する必要があります。

- エラー・プロセッサは、アセンブラ言語でコーディングする必要があります。
- ルーチンの最初の実行可能ステートメントには TEPCDxx というラベルを付ける必要があります。ここで、「xx」は「DFHTEPM TYPE=ERRPROC, CODE=errcode」マクロで指定するエラー・コードです。
- レジスターの使用法の規則と制限については、サンプルの DFHTEP ソース内に記載されています。
- エラー・プロセッサは、サンプルの DFHTEP シンボル・ラベルへの出口でなければなりません。

ユーザーの「入り口」ルーチンに必要なマクロは、次のとおりです。

```
DFHTEPM  TYPE=ENTRY
```

このマクロの直後に、ユーザーの「入り口」ルーチン・コードを指定します。そのコードはラベル「TEPENTRY」で始まり、BR 14 の命令で終わります。

ユーザーの「出口」ルーチンに必要なマクロは、次のとおりです。

```
DFHTEPM  TYPE=EXIT
```

このマクロの直後に、ユーザーの「出口」ルーチン・コードを指定します。そのコードはラベル「TEPEXIT」で始まり、BR 14 の命令で終わります。

### DFHTEPM TYPE=ERRPROC – エラー・プロセッサの置き換え

サンプル DFHTEP に用意されているエラー・プロセッサをユーザー作成のエラー・プロセッサに置換するために必要なマクロは、次のとおりです。

```
DFHTEPM  TYPE=ERRPROC
          ,CODE=errcode
          (followed by the appropriate error
           processor source statements)
```

#### TYPE=ERRPROC

CICS 提供のエラー・プロセッサ・ルーチンを、このマクロの直後に指定されたユーザー作成のエラー・プロセッサに置換することを示します。このマクロはオプションです。使用する場合は、DFHTEPM TYPE=INITIAL マクロの後に指定する必要があります。ユーザー作成の各エラー・プロセッサのソース・ルーチンの前に DFHTEPM TYPE=ERRPROC マクロを 1 つ指定する必要があります。

#### CODE=errcode

対応するエラー状況に割り当てられているエラー・コードを指定するために使用します。これらのコードについては、[104 ページの図 23](#) に記載しています。

### DFHTEPM TYPE=FINAL – サンプル DFHTEP モジュールの終了

サンプル DFHTEP モジュールを終了するマクロは、次のとおりです。

```
DFHTEPM  TYPE=FINAL
```

この後に END DFHTEPNA ステートメントを指定します。

#### DFHTEPM マクロの例

この例では、CICS 提供のエラー・プロセッサとすべてのデフォルト・オプションを使用してサンプル DFHTEP モジュールを生成します。これは、CICS 提供のサンプル端末エラー・プログラムに相当します。

1. 以下の例は、サンプル DFHTEP モジュールを生成するために必要な最小数のステートメントを示しています。

```
DFHTEPM  TYPE=INITIAL
DFHTEPM  TYPE=FINAL
END DFHTEPNA
```

2. 95 ページの図 20 は、さらに調整されたサンプル DFHTEP モジュールの例です。この例では、3270 のサポートは生成されません。TACP および TEP のアクションを除くすべてのデフォルト・タイプの情報が、一時データ宛先 TEPQ に記録されます。CICS DSECT は、サンプル DFHTEP のアセンブラ言語リストに出力されません。2 つのエラー・プロセッサ・ルーチン (コード「87」と「9F」) があります。



```

* GENERATE USER STORAGE

    DFHTEPM      TYPE=USTOR
USORFLD      DS  F
    DFHTEPM      TYPE=USTOREND

* MODULE SPECIFICATIONS

    DFHTEPM      TYPE=INITIAL,
                  OPTIONS=((TD,TEPQ),N03270,EXITS),
                  PRINT=(NOTEPACTION,NOTACPACTION),
                  DSECTPR=NO
*
*
* USER-SUPPLIED ERROR PROCESSORS

    DFHTEPM      TYPE=ERRPROC,CODE=87
TEPCD81      DS  0H
    -
    -      error processor "87" source statements
    -
    B      TEPRET

    DFHTEPM      TYPE=ERRPROC,CODE=9F
TEPCD9C      DS  0H
    -
    -      error processor "9F" source statements
    -
    B      TEPRET

* USER "EXIT" EXIT CODE

    DFHTEPM      TYPE=EXIT
TEPEXIT      DS  0H
    -
    -
Additional user source statements to be executed after
error processing:
    -
    -
    BR      R14

* CONCLUDE MODULE GENERATION

    DFHTEPM      TYPE=FINAL
END DFHTEPNA

```

図 20. サンプル *DFHTEP* モジュールの生成に使用される *DFHTEPM* マクロの例

### DFHTEPT – サンプル *DFHTEP* テーブルの生成

端末エラー・プログラムのテーブルを生成するためには、次のマクロが必要です。

- *DFHTEPT* *TYPE=INITIAL* – 制御セクションを設定します。
- *DFHTEPT* *TYPE=PERMTID* – 特定の端末のために永続的に予約される端末エラー・ブロック (TEB) を定義します。
- *DFHTEPT* *TYPE=PERMCODE|ERRCODE* – 永続的に予約されるエラー状況エレメント (ESE) を定義します。
- *DFHTEPT* *TYPE=BUCKET* – 共通エラー・バケットに記録する特定のエラー状況を定義します。
- *DFHTEPT* *TYPE=FINAL* – *DFHTEPT* マクロのセットを終了します。

### DFHTEPT *TYPE=INITIAL* – 制御セクションの設定

TEP テーブルの制御セクションの設定に必要な *DFHTEPT* *TYPE=INITIAL* マクロは、次のとおりです。

```

DFHTEPT      TYPE=INITIAL
              ,MAXTIDS=number
              [,MAXERRS={25|number}]
              [,OPTIONS={TIME|NOTIME}]

```

#### **TYPE=INITIAL**

TEP テーブルの生成の開始を設定します。

## MAXTIDS=number

TEP エラー・テーブルで生成する永続的な端末エラー・ブロックおよび再使用可能な端末エラー・ブロックの総数を指定します。永続的なエンタリーは、このセクションで後述する DFHTEPT TYPE=PERMTID マクロを使用して定義します。永続的なものとして定義されていないエンタリーは、端末がサービス休止になるか、エラー・プロセッサの要求で削除される場合に再利用されます。エラーが発生し、利用できる TEB スペースがない場合、エラーは処理されず、DFHTACP のデフォルトのアクションが実行されます。ブロックの最小数は 1 です。最大数はチェックされませんが、ネットワークの端末の数より多くしないでください。

## MAXERRS=25|number

端末ごとに記録するエラーの数を指定します。この値によって、各 TEB に含まれる永続的なエラー状況エレメントおよび再使用可能なエラー状況エレメントの数が決まります。指定できる最大数は 25 (デフォルト値) です。これより多い数を要求しても、この最大数しか生成されません。より少ない数を要求した場合は、TEB ごとに追加の ESE が 1 つ生成されます。この追加の ESE が共通エラー・バケットです。永続的に予約される ESE は、このセクションで後述する DFHTEPT TYPE=PERMCODE マクロを使用して定義します。永続的なものとして定義されていない ESE は、非永続的なエラー・タイプが端末に初めて関連付けられたときに、動的に割り当てられます。最大値より小さい数を定義して、サンプル DFHTEP で ESE を動的に割り当てるようにすれば、テーブルのサイズを最小化しながらも、ネットワークに関連するエラー・タイプを制御して記録することができます。指定できる最小数はゼロです。この場合は、1 つの共通エラー・バケットのみが生成されます。

## OPTIONS={TIME|NOTIME}

DFHTEPM TYPE=INITIAL マクロで指定する TIME オプションをサポートするために、時間のしきい値のスペースを予約するかどうかを指定します。デフォルトは OPTIONS=TIME です。

### TIME

時間のしきい値のスペースが予約されます。

### NOTIME

時間のしきい値のスペースは予約されません。

## DFHTEPT TYPE=PERMTID – 永続端末エラー・ブロックの割り当て

DFHTEPT TYPE=PERMTID マクロを使用して、特定の端末のために永続的に予約される端末エラー・ブロックを定義できます。

## 構文

```
DFHTEPT TYPE=PERMTID  
        ,TRMIDNT=name
```

## TYPE=PERMTID

特定の端末のために永続的に予約される端末エラー・ブロックを定義します。永続的な TEB は、システムの稼働にとって重要な端末のために定義します。これにより、その端末に関連付けられたエラーが発生した場合に、必ずエラー・プロセッサが実行されるようにします。永続的な TEB を定義しない場合、このマクロは必要ありません。永続的に予約する TEB ごとに、マクロを発行する必要があります。永続的な TEB の最大数は、DFHTEPT TYPE=INITIAL マクロの MAXTIDS オペランドに指定した数です。

## TRMIDNT=name

永続的に定義された TEB に対するシンボル端末 ID (1 文字から 4 文字) を指定します。マクロ 1 つにつき端末を 1 つのみ指定できます。

## DFHTEPT TYPE=PERMCODE|ERRCODE – エラー状況エレメントの定義

DFHTEPT TYPE=PERMCODE|ERRCODE マクロを使用して、サンプル DFHTEP のデフォルトのしきい値定数の変更、および永続的に予約されるエラー状況エレメントの定義を行えます。

```
DFHTEPT TYPE={PERMCODE|ERRCODE}  
        ,CODE={errcode|BUCKET}  
        [,COUNT=number]  
        [,TIME=(number[,SEC|,MIN|,HRS])]
```

## TYPE={PERMCODE|ERRCODE}

このマクロで指定するエラー・コードの ESE が、永続的に予約されるか、それとも動的に割り当てられるかを示します。これらのマクロは、永続的に予約される ESE を定義する場合、またはサンプル DFHTEP のデフォルトのしきい値定数をオーバーライドする場合にのみ必要です。これらのリストを 98 ページの表 8 に示しています。

### PERMCODE

このエラー・コードを、ESE が永続的に予約されるものとして指定することを示します。永続的に予約されるすべての ESE を、ESE ごとに DFHTEPT TYPE=PERMCODE マクロを個々に使用して指定する必要があります。すべての DFHTEPT TYPE=PERMCODE マクロを、すべての DFHTEPT TYPE=ERRCODE マクロの前に指定する必要があります。

### ERRCODE

指定するエラー・コードに永続的に予約される ESE が不要であること、ただし、サンプル DFHTEP のデフォルトのしきい値定数を変更することを示します。永続的に予約されるものとして定義されたエラー・コードを除き、しきい値定数の変更が必要なすべてのエラー・コードを、エラー・コードごとに DFHTEPT TYPE=ERRCODE マクロを個々に使用して指定する必要があります。すべての DFHTEPT TYPE=ERRCODE マクロを、すべての DFHTEPT TYPE=PERMCODE マクロの後に指定する必要があります。

## CODE={errcode|BUCKET}

TYPE=PERMCODE|ERRCODE パラメーターが表すエラー・コードを指定します。これらのコードについては、104 ページの図 23 に記載しています。CODE=BUCKET は、DFHTEPT TYPE=ERRCODE マクロにのみ適用できます。共通エラー・バケットに対して設定されたデフォルトのしきい値定数をオーバーライドするために使用します。

## COUNT=number

DFHTEPT TYPE=PERMCODE または TYPE=ERRCODE マクロのいずれかで使用して、サンプル DFHTEP のデフォルトのカウントしきい値をオーバーライドできます (98 ページの表 8 を参照)。指定したエラー・タイプの発生回数がしきい値に達すると、エラー・プロセッサは、通常、DFHTACP のデフォルトのアクションを実行するロジック・パスを取ります。発生回数がしきい値よりも小さい場合、エラー・プロセッサは、通常、DFHTACP のデフォルトのアクションをオーバーライドするロジック・パスを取ります。現在のしきい値カウントの更新とテストは、DFHTEP サブルーチンによって実行されます。このサブルーチンは、制限に達したかどうかを判断するためにエラー・プロセッサがテストできる状況コードを設定します。COUNT オペランドの数値として 0 を指定した場合は、しきい値に達しても通知されません。

## TIME=(number[,SEC|,MIN|,HRS])

DFHTEPT TYPE=PERMCODE または TYPE=ERRCODE マクロのいずれかで使用して、サンプル DFHTEP のデフォルトの時間しきい値をオーバーライドできます (98 ページの表 8 を参照)。このオペランドは、DFHTEPM および DFHTEPT TYPE=INITIAL マクロの両方で OPTIONS=TIME を指定した場合にのみ適用できます。このパラメーターで指定した間隔内に (前述のように) 発生回数が COUNT オペランドで指定されているしきい値に達した場合、エラー・プロセッサは通常、DFHTACP のデフォルトのアクションが実行されるロジック・パスを取ります。間隔内の発生回数がしきい値よりも小さい場合、エラー・プロセッサは通常、DFHTACP のデフォルトのアクションをオーバーライドするロジック・パスを取ります。時間間隔が経過すると、通常は現在のしきい値のカウントを更新してテストするサンプル DFHTEP のサブルーチンが、発生回数をリセットして、新しい有効期限を設定します。この場合、サブルーチンによって設定される状況コードは、しきい値に達していないことを示します。

サンプル DFHTEP の時間制御は、エラー・タイプの最初の発生によって開始されます。同じエラー・タイプの 2 回目以降の発生では、新しい開始時間は設定されませんが、最初の発生で開始された間隔内に発生したことが記録されます。これは、最初の発生によって開始された間隔内でエラー・カウントがしきい値に達するまで、または間隔が経過するまで続けられます。後者の場合では、処理中のエラーが最初の発生になり、新しい間隔が開始されます。0 の時間間隔は、時間間隔に関係なく発生数が記録され、制御されることを意味します。ゼロは、COUNT オペランドの値が 0 または 1 の場合の暗黙的な時間間隔です。また、時間オプションが生成されない場合の暗黙的な時間間隔でもあります。

時間間隔は、4 つの単位 (時、分、秒、または 100 分の 1 秒) のいずれかで表現できます。最大間隔は、24 時間未満に相当する間隔でなければなりません。現実的な最小値は、1 分から 2 分です。これにより、アクセス・メソッドの再試行、および各エラーを処理するタスクの作成に必要な時間を確保できます。しきい値の時間間隔を表す 4 つの方法を以下に示します。

**number**

100 分の 1 秒の間隔。この方法を使用する場合、括弧は必要ありません。最大値は 8,640,000 (24 時間) 未満でなければなりません。

**(number,SEC)**

秒単位の間隔。括弧で囲む必要があります。最大数は 86,400 (24 時間) 未満でなければなりません。

**(number,MIN)**

分単位の間隔。括弧で囲む必要があります。最大値は 1,440 (24 時間) 未満でなければなりません。

**(number,HRS)**

時間単位の間隔。括弧で囲む必要があります。最大値は 24 未満にする必要があります。

98 ページの表 8 は、DFHTEPT TYPE=PERMCODE|ERRCODE マクロの TYPE、COUNT、および TIME オペランドの説明で言及したサンプルの端末エラー・プログラムのデフォルトのしきい値を示しています。

表 8. サンプル TEP のデフォルトのしきい値		
CODE=	COUNT=	TIME=
81	3	(7,MIN)
84	1	0
85	1	0
87	50 (注 98 ページの『2』)	0
88	1	0
8C	1	0
8D	1	0
8E	1	0
8F	1	0
90	0	0
91	0	0
94	7	(10,MIN)
95 (注 98 ページの『1』)	0	0
96	2	(1,MIN)
97 (注 98 ページの『1』)	0	0
99	1	0
9F (注 98 ページの『1』)	0	0
BUCKET	5	(5,MIN)

**注：**

1. エラー・プロセッサは、エラー・カウントのみを保持します。DFHTACP のデフォルトのアクションは、しきい値に関係なく必ず実行されます。
2. エラー・プロセッサは、しきい値のカウントではなく、しきい値の「重み」を使用します (サンプル DFHTEP のソース・コードを参照)。

**DFHTEPT TYPE=BUCKET – 特定のエラーに対してエラー・バケットを使用する**

DFHTEPT TYPE=BUCKET マクロは、特定のエラー状況を常に共通エラー・バケットに記録するために使用します。

```
DFHTEPT TYPE=BUCKET
        ,CODE=errcode
```

## TYPE=BUCKET

共通エラー・バケットに特定のエラー状況を記録するマクロを生成します。DFHTEPT TYPE=INITIAL マクロで MAXERR=25 を指定した場合は、このマクロは使用できません。共通エラー・バケットに特別に記録するエラー・コードが存在しない場合は、このマクロは不要です。すべてのエラー・コードを、エラー・コードごとに DFHTEPT TYPE=BUCKET マクロを個々に使用して指定する必要があります。

## CODE=errcode

共通エラー・バケットに特別に記録するエラー・コードを指定します。同じエラー・コードを、DFHTEPT TYPE=PERMCODE または TYPE=ERRCODE マクロで指定しないでください。

## DFHTEPT TYPE=FINAL – DFHTEPT エントリーの終了

DFHTEPT TYPE=FINAL マクロは、DFHTEP テーブルの生成を終了します。

## 構文

```
DFHTEPT TYPE=FINAL
```

## DFHTEPT - マクロの使用例

この例では、再使用可能な端末エラー・ブロックを 10 個生成します。どの端末エラー・ブロックも最大数のエラー・タイプを記録できます。時間しきい値の制御がサポートされ、すべてのしきい値は、サンプル DFHTEP でサポートされるデフォルトです。これは、CICS 提供のサンプル端末エラー・プログラムに相当します。

1. 以下の例は、TEP テーブルを生成するために必要な最小数のステートメントを示しています。

```
DFHTEPT TYPE=INITIAL,MAXTIDS=10
DFHTEPT TYPE=FINAL
END
```

2. 99 ページの図 21 は、カスタマイズされた TEP テーブルの例です (継続文字は省略しています)。

```
* TABLE SPECIFICATIONS
      DFHTEPT      TYPE=INITIAL,MAXTIDS=10,
                   MAXERRS=5
* PERMANENT TERMINAL DEFINITIONS
      DFHTEPT      TYPE=PERMTID,TRMIDNT=TM02
* PERMANENT ERROR CODE DEFINITIONS
      DFHTEPT      TYPE=PERMCODE,CODE=81
      DFHTEPT      TYPE=PERMCODE,CODE=87,
                   COUNT=2,TIME=(1,MIN)
* OTHER THRESHOLD OVERRIDES
      DFHTEPT      TYPE=ERRCODE,CODE=BUCKET,
                   COUNT=3,TIME=(3,MIN)
* CONCLUDE TABLE GENERATION
      DFHTEPT      TYPE=FINAL
      END
```

図 21. DFHTEP テーブルを生成するための DFHTEPT マクロの使用例

この例は 10 個の端末エラー・ブロックを生成しますが、そのうちの 1 個は、シンボリック ID が TM02 の端末のために予約され、他の 9 個は再使用可能です。各 TEB には、5 つのエラー状況エレメントと 1 つの共通エラー・バケットのためのスペースがあります。5 つの ESE のうち、2 つがエラー・コード「81」と「87」のために予約されています。残りの ESE は動的に割り当てることができます。エラー・コード「87」と共通エラー・バケットのしきい値が変更されています。共通エラー・バケットに記録される特別なエラー・コードはありません。



## 端末エラー・プログラムの作成

CICS でサポートされているすべての言語で、独自の端末エラー・プログラム (TEP) を作成できます。

用意されている CICS 提供のコードは、アセンブラー言語のみです。用意されているソース・ファイルとマクロの名前や、それらのリソースが入っているライブラリーについては、[100 ページの表 9](#) を参照してください。

表 9. 用意されているソース・ファイルとマクロ			
名前	タイプ	説明	ライブラリー
DFHXTEP	ソース	サンプル端末エラー・プログラム (アセンブラー言語)	CICSTS56.CICS.SDFHSAMP
DFHXTEPT	CSECT	サンプル端末エラー・テーブル (アセンブラー言語)	CICSTS56.CICS.SDFHSAMP
DFHTEPM	マクロ	サンプル TEP プログラム生成プログラム (アセンブラー言語)	CICSTS56.CICS.SDFHMAC
DFHTEPT	マクロ	TEP テーブル生成プログラム (アセンブラー言語)	CICSTS56.CICS.SDFHMAC
DFHTEPCA	マクロ	アセンブラー言語通信域	CICSTS56.CICS.SDFHMAC

ユーザー作成の DFHTEP は、CICS 提供のサンプル DFHTEP と同じ要領で制御を受け取ります ([82 ページの『順次装置のエラー処理に関する背景情報』](#)を参照)。したがって、DFHTACP の基本的なインターフェースには通信域を使用してください。

### 独自の端末エラー・プログラムを作成する理由

- CICS が入力専用の端末にメッセージを送信しようとする状況があります。例えば、無効なトランザクション ID のメッセージや、アプリケーション・プログラムから間違って送信されるメッセージなどがそれに当たります。一時データ機能やインターバル制御機能によってそのようなメッセージをシステムの宛先 (CSMT や CSTL など) に転送するための、端末エラー・プログラムを用意する必要があります。
- 端末エラーの発生時には、アプリケーション関連のアクティビティーを実行しなければならない場合があります。例えば、エラー状態が原因でメッセージが端末に送信されなかった場合は、メッセージの転送が必要である旨をアプリケーションに通知する必要があるかもしれません。
- すべてのエラーが通信システム障害に相当するわけではありません。例えば、SAM のデータ終了条件などもあります。

### EXEC CICS コマンドの使用に関する制限

端末エラー・プログラム (TEP) が実行できるコマンドには制限があります。特に、基本機能を必要とするコマンドは避けてください。そのようなコマンドを使用した場合の結果は予測不能です。

以下の機能呼び出すコマンドは、使用しないでください。

- 端末管理 (CEMT タイプのコマンド (**EXEC CICS INQUIRE TERMINAL** など) は使用できます)
- BMS (ルーティング以外)
- ISC 通信 (機能シップを含む)

### 通信域の内容のアドレッシング

端末エラー・プログラムは、DFHTACP から制御を受け取った後に、EXEC CICS ADDRESS COMMAREA コマンドによって通信域のアドレスを取得しなければなりません。

### このタスクについて

通信域 DSECT を生成するには、プログラムで DFHTEPCA TYPE=DSECT をコーディングします。通信域のレイアウトを [101 ページの図 22](#) に示します。



				IN/OUT PARM	
TEPCALDS	DS	0XL4		Standard Header	
TEPCAGDS	DS	XL1	I	Function Code	Always '1'
	DS	XL2	I	Component Code	Always 'TC'
	DS	XL1		Reserved	
TEPCATCA	DS	A	I	Address of TACLE being processed	
TEPCECIA	DS	A	I	Address of TCTUA	
TEPCECIL	DS	H	I	Length of TCTUA	
TEPCAACT	DS	XL1	I/O	User action byte	
TEPCATID	DS	CL4	I	Terminal identity	
TEPCATDB	DS	F	I	Current time of day binary	

図 22. DFHTACP/DFHTEP 通信域

パラメーター・リストには、次の情報が含まれています。

#### TEPCALDS

機能コード。機能コードは、DFHTEP を呼び出した TCP にあるタスクの ID に相当する印刷可能文字です。値は常に 1 になります。

#### TEPCAGDS

コンポーネント・コード。値は常に TC になります。TCP のコンポーネントを表します。

#### TEPCATCA

処理対象の TACLE のアドレスが入ります。

#### TEPCECIA

端末管理テーブル・ユーザー域 (TCTUA) のアドレスが入ります。

#### TEPCECIL

TCTUA の長さが入ります。

#### TEPCAACT

ユーザー・アクション・バイト。通信域の主な用途の 1 つは、端末に対して実行するアクションを送信することです。TEPCAACT には以下のフラグが入ります。いずれも、DFHTEP 内でリセットできるフラグです。

##### LINEOS (X'80')

回線をサービス休止状態にします。

##### NONPRGT (X'40')

端末にパージ不能タスクが存在します。

##### TERMOS (X'20')

端末をサービス休止状態にします。

##### ABENDT (X'10')

端末でタスクを異常終了させます。

##### ABORTWR (X'08')

書き込みを異常終了させ、端末のストレージを解放します。

##### RELTIOA (X'04')

TCAM 着信メッセージを解放します。(TCAM はサポートされなくなりました。)

##### SIGNOFF (X'02')

サインオフ・プログラムを呼び出します。

DFHTEP への入り口では、これらのフラグは、DFHTACP で設定されているデフォルト・アクションになります。回線サービス休止ビット (X'80') が設定されると、書き込み異常終了ビット (通信域フィールド **ABORTWR**) とタスク異常終了ビット (通信域フィールド **ABENDT**) が常に設定されますが、どちらのビットも、ダミー端末標識を指定すると抑止されます ([102 ページの『ユーザー・アクション・バイト TEPCAACT のフラグのリセット』](#)を参照)。

DFHTACP への戻りでは、これらのフラグは、DFHTEP によって変更されたアクションになります。

#### TEPCATID

エラー状態の端末の ID が入ります。

## TEPCATDB

エラー発生時刻が入ります (2 進形式)。

## ユーザー・アクション・バイト TEPCAACT のフラグのリセット

### このタスクについて

TEPCAACT のアクション・ビットを変更する場合の注意点を以下にまとめます。

- データ・セキュリティを維持する方法を検討してください。例えば、障害の原因が取り除かれるまでの間、端末がしばらくサービス休止状態になると、端末がサービス状態に戻った際に、元のオペレーターが存在しないにも関わらず、サインオン情報が TCTTE に引き続き存在していることがあります。そのようなセキュリティ違反の可能性を排除するために、**SIGNOFF** ビットを設定して端末をサインオフすることができます。
- TCTLEPF2 のダミー端末標識は、端末を特定できない場合にエラーに設定されます。したがって、ダミー端末標識があると、タスク異常終了や書き込み異常終了は設定されません。ダミー端末は、回線を識別する目的でのみ使用されます。
- タスク異常終了ビット (TEPCAACT の X'10') には、TACP のトランザクション異常終了処理の一部として他の 2 つのビットが常に関連付けられます。その 2 つのビットとは、ページ不能タスクと書き込み異常終了です (どちらも TEPCAACT にあり、それぞれ X'40' と X'08' です)。
- 書き込み異常終了は、常にタスク異常終了と同時にオンに設定されます。この設定には、処理対象のエラーが TC WRITE で発生した場合に、TCTTE から元の書き込み要求標識を消去する効果があります。
- ページ不能タスクがオンに設定されるのは、トランザクションに端末に関連付けられていて、トランザクション ID で TPURGE=NO が指定されている場合です。

ダミー端末標識がオンになっていると、通常は DFHTACP によってタスク異常終了が処理対象エラーのデフォルトとして設定される場合でも、タスク異常終了ビットや書き込み異常終了ビットやページ不能タスク・ビットは設定されません。したがって、以下の言及は、実際の端末に関連したエラーだけに当てはまります。

- 端末にトランザクションが関連付けられていない場合は、タスク異常終了は無効です (ただし、端末に疑似会話型タスクが関連付けられている場合は、次の transid がクリアされます)。そうでない場合は、ページ不能タスク標識があると、トランザクションは (通常は SUSPEND 状態で) 端末に接続されたままになり、DFHTACP が DFHTC2522 INTERCEPT REQUIRED メッセージを CSMT に書き込みます。ページ不能マークが付いていないタスクは、コード AEXY で異常終了します。まれに AEXZ で異常終了することもあります。
- TCTTE に READ 要求が関連付けられている場合は、書き込み異常終了は無効です。その場合は通常、回線と端末がサービス状態のままになっていれば、読み取りが再試行されます。

### TACLE の内容のアドレッシング

エラーが発生すると、端末管理プログラムによって TACLE が作成されます。TACLE には、BSAM によって提供されるすべての入出力エラー情報が入ります。

### このタスクについて

TACLE の内容をアドレッシングするには、ユーザー作成端末エラー・プログラムに COPY DFHTACLE ステートメントと COPY DFHTCTLE ステートメントが、その順序で組み込まれている必要があります。それらのステートメントで完全な DFHTCTLE DSECT を定義します。TACLE のフィールドのアドレッシングと、エラーに関連した実際の回線項目のフィールドのアドレッシングの両方で、その DSECT にあるシンボル名が使用されます。

TACLE には、16 バイトの接頭部 (COPY DFHTACLE で定義) と 48 バイトのセクションがあります。そのセクションは、TACLE 作成時の実際の回線項目の DECB を変更したコピーです。

したがって、TACLE をアドレッシングするために、ユーザー作成端末エラー・プログラムに以下のようなステートメントを組み込む必要があります。

```
COPY DFHTACLE  
COPY DFHTCTLE
```

```
L TCTLEAR,TEPCATCA      POINT TO TACLE
  USING DFHTCTLE,TCTLEAR
```

実際の回線項目 DECB に通常含まれているフィールドの場合は、TACLE でオフセットが 16 ずつ増えます。  
TACLE 内の DECB コピーの以下のフィールドは、実際の回線項目のデータ・コピーに対応していません。

```
TCTLEDCB                (Offset 24 in TACLE,
                          8 in real TCTLE)
```

このフィールドは、TACLE 内では実際の回線項目を指していますが、実際の回線項目内では回線グループの BSAM DCB を指しています。

```
TCTLECSW                (Offsets 46, 48 in TACLE,
  TCTLEALP                30, 32 in real TCTLE)
```

TACLE 内では、これらのフィールドを SAM エラー情報のために使用します。

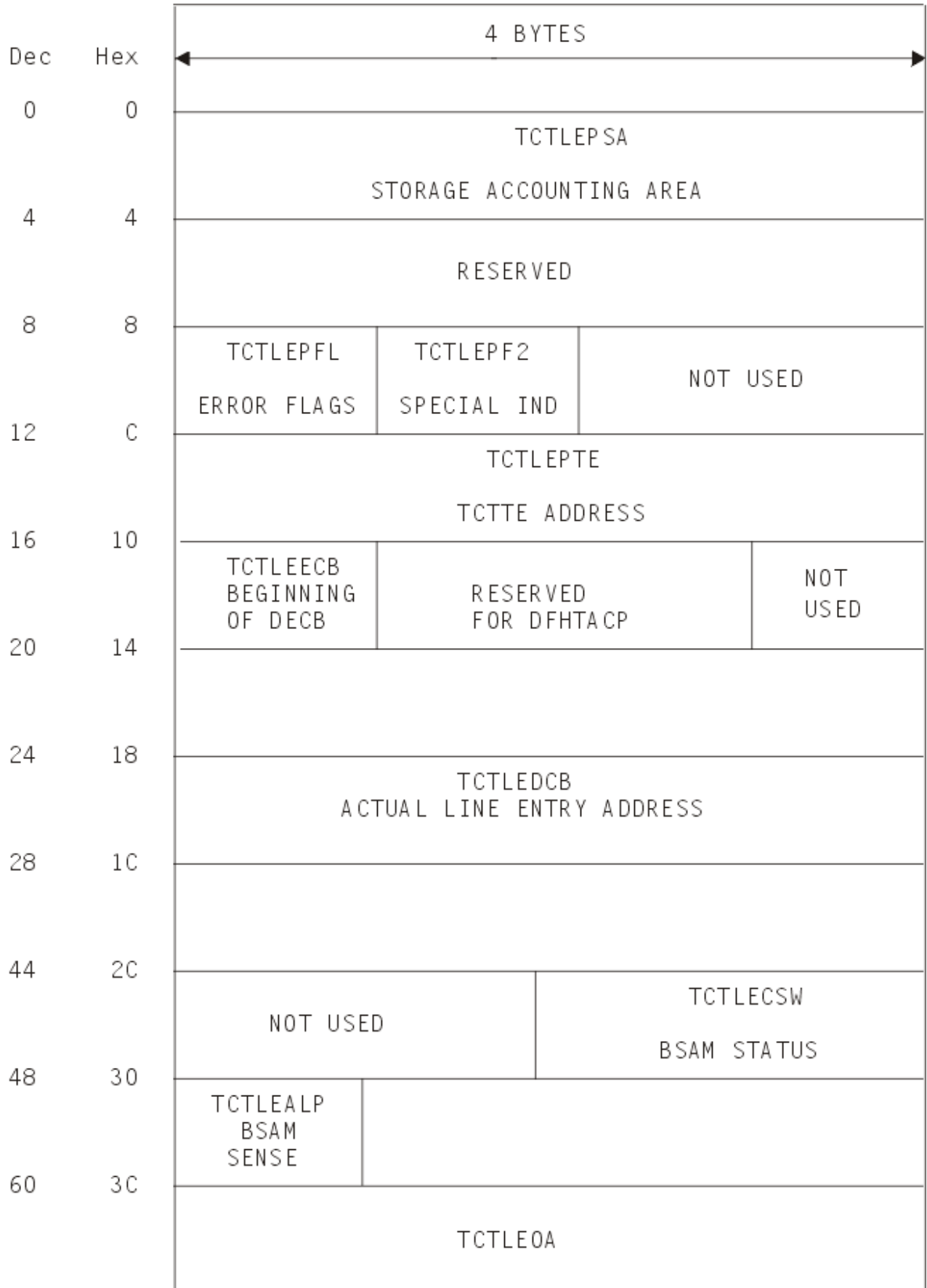
以下のステートメントによって、**実際の**回線項目に直接的なアドレッシング機能を設定します。

```
COPY DFHTCTLE
COPY DFHTCTTE

L      TCTLEAR,TEPCATCA      POINT TO TACLE
USING  DFHTCTLE,TCTLEAR
L      TCTTEAR,TCTLEPTE      POINT TO ERROR TCTTE
USING  DFHTCTTE,TCTTEAR
DROP   TCTLEAR
L      TCTLEAR,TCTTELEA      POINT TO TCTLE
USING  DFHTCTLE,TCTLEAR
```

必要な機能を実行し、場合によっては、DFHTACP によってスケジュールが設定されていたデフォルト・アクションを変更したら、EXEC CICS RETURN コマンドを実行して、ユーザー作成の DFHTEP から DFHTACP に制御を戻す必要があります。DFHTACP は、TACLE で指定されている処理を実行して、エラー処理タスクを終了します。

TACLE DSECT の形式を [104 ページの図 23](#) に示します。



Displacement					
Dec	Hex	Code	Bytes	Label	Meaning
0	0		4	TCTLEPSA	Storage accounting
				RESERVED	
8	8	81	1	TCTLEPFL	Error flags
					Message too long
		84			TCT search error
		85			Write not valid
		87			Unsolicited input
		88			Input event rejected
		8C			Output event rejected
		8D			Output length of zero
		8E			No output area
		8F			Output area exceeded
		94			Unit check
		95			Unit check
					(should not occur)
		96			Unit exception
		97			Unit exception
					(should not occur)
		99			Undetermined I/O error
		9F			Invalid destination
					(TCAM: no longer supported)
		.			
		.			(All codes not listed are reserved and are
		.			not intended for use by DFHTEP)
		.			
9	9	01	1	TCTLEPF2	Special indicator
					dummy terminal
12	C		4	TCTLEPTE	Address of terminal
					entry for terminal
					in error
16	10		4	TCTLEECB	DECB/copy of line
					when error occurred
60	3C		4	TCTLEOA	For TCAM lines only.
					(No longer supported)

図 24. TACLE DSECT の形式の説明 (パート 2)

## ユーザー作成端末エラー・プログラムの例

端末エラー・プログラムの一部を設計するのに必要なロジックの各ステップの例を 106 ページの『DFHTEP 再帰的再試行ルーチン』に示します。106 ページの図 25 では、端末ごとに 10 回の再試行を記述していますが、どんな回数でも同じロジックを使用できます。前提事項を以下にまとめます。

### USER FIELD A (PCISAVE)

TCTTE のプロセス制御情報 (PCI) 域にある 6 バイト・フィールドに対応します。このフィールドを使用して、最初のエラーの発生時に TCTTE の入力と出力のカウントを保存します。そのカウントは、TCTTE 内の TCTTENI と TCTTEN0 にある 3 バイト・フィールドに入っています。

### USER FIELD B (PICNT)

再帰的エラーのカウントを集計するためのユーザー定義フィールドに対応します。そのフィールドは、TCTTE のプロセス制御情報 (PCI) 域に存在します。

### SYSTEM COUNT (TCTTENI)

TCTTE 内の 6 バイト・フィールドに対応します。端末の入力と出力のカウント (TCTTENI+TCTTEN0) が入ります。この例では、その 2 つの隣接しているフィールドを 1 つの 6 バイト・フィールドと見なしています。

この例では、TCT 端末入力 (TCTTE) にアクセスして、SYSTEM COUNT を調べたりプロセス制御情報 (PCI) 域を見つけたりする必要があるので、DFHTCTTE シンボリック・ストレージ定義を組み込んで、シンボルによって各フィールドを参照できるようにしています。

## DFHTEP 再帰的再試行ルーチン

```
*ASM      XOPTS(NOPROLOG NOEPILOG SP)
*****
*
*                      DFHTEP RECURSIVE RETRY ROUTINE
*
*****
      DFHEISTG
      DFHEIEND
      DFHTEPCA TYPE=DSECT      COMMAREA passed by TACP
      COPY  DFHA06DS          Statistics DSECT
      USING DFHA06DS,STATBAR
PCIAREA DSECT
PCISAVE DS    XL6      User Field A
PCICNT  DS    PL2      User Field B
*
TCTLEAR EQU    2      Pointer to TACLE
STATBAR EQU    4      Pointer to statistics DSECT
TCTUABAR EQU    5      Pointer to TCTUA
COMMABAR EQU   12      Pointer to COMMAREA passed by TACP
      EJECT
DFHTEP  CSECT
*****
*      Establish addressability
*
*****
      DFHEIENT
*
      EXEC CICS ADDRESS EIB(11)
*
      EXEC CICS ADDRESS COMMAREA(COMMABAR)
*
      USING DFHTEPCA,COMMABAR
      L      TCTLEAR,TEPCATCA    Load TACLE address
*
      USING PCIAREA,TCTUABAR
      L      TCTUABAR,TEPCECIA   Load TCTUA address
*
*****
*      Start processing
*
*****
      TM      PCICNT+1,X'0C'      Has User Field B been initialized
                                  to a packed decimal number?
*
      BO      CKCOUNT           YES .... so compare the system count
                                  with the existing count in Field B
*
      RESET   DS      0H
      MVC     PCICNT,=PL2'+0'    NO .... so initialize field B to
                                  packed zero.
*
*
```

図 25. DFHTEP 再帰的再試行ルーチン (パート 1)



```

EXEC CICS COLLECT STATISTICS TERMINAL(TEPCATID) SET(STATBAR)
*                               Get statistics for this terminal
*                               using TERMID passed in Commarea
*
MVC    PCISAVE,A06TENI        Save the current system counts. これにより、
*                               is a new error, or first time
*                               through.
INCR    DS    0H
AP      PCICNT,=P'1'          Increment the number of times this
*                               error has occurred (recursive count)
*
CP      PCICNT,=P'10'         Has the maximum recursive error
*                               limit been reached?
BNE     RETRY                 NO .... set action
*
ZAP     PCICNT,=P'0'          Clear and reset user fields for next
*                               error set
EXEC CICS COLLECT STATISTICS TERMINAL(TEPCATID) SET(STATBAR)
*                               Get statistics for this terminal
*                               using TERMID passed in COMMAREA
*
MVC     PCISAVE,A06TENI        Get current system counts
B       NORETRY               Action indicators for no retry
*
CKCOUNT DS    0H
EXEC CICS COLLECT STATISTICS TERMINAL(TEPCATID) SET(STATBAR)
*                               Get statistics for this terminal
*                               using TERMID passed in COMMAREA
*
CLC     PCISAVE,A06TENI        Has system count changed since last
*                               entry to TEP?
BNE     RESET                 YES .... this is a new error since
*                               some I/O activity has occurred on
*                               terminal.
B       INCR                  NO .... this is a recursive error,
*                               so increment the recursive count and
*                               check for retry.
RETRY   DS    0H
*                               The user would include here the code
*                               necessary to alter the flags in the
*                               COMMAREA so that a retry can be
*                               performed on the terminal.
NORETRY DS    0H
*                               The user would include here the code
*                               necessary to allow DFHTACP to take
*                               final actions on the terminal; that
*                               is, abend task, put line out of
*                               service, and others.
*
LTORG ,
END

```

図 26. DFHTEP 再帰的再試行ルーチン (パート 2)

106 ページの図 25 のコードは、再帰的エラー処理の技法を示したり、該当する制御ブロックへのアドレッシング機能を確認するために必要なステップを示したりするための例にすぎません。

## ノード・エラー・プログラムの作成

ACF/SNA インターフェースを使用してサポートされる端末および論理装置に対するノード・エラー・プログラム (NEP) を作成することができます。

CICS には、独自のプログラムの基礎として使用できるサンプル・ノード・エラー・プログラムが用意されています。z/OS Communications Server 以外の装置の端末エラー・プログラムの場合と同じく、SNA 接続端末のノード・エラー・プログラムにも、3 つの形式があります。

1. デフォルトのノード・エラー・プログラム
2. CICS 提供のサンプル・ノード・エラー・プログラム
3. ユーザー作成のバージョン

アプリケーション・プログラムで **EXEC CICS HANDLE CONDITION TERMERR** コマンドをコーディングする場合、アプリケーション・プログラムがノード・エラー・プログラムを使用せずに例外ケースを処理できることが時々あります。TERMERR 状態は、ノード異常条件プログラム (DFHZNAC) が ABTASK (「ATNI」

異常終了)を実行した場合に発生します。TERMERR 状態は、アプリケーション関連であり、セッション関連の問題に使用しなければならないノード・エラー・プログラムの代替ではありません。アプリケーション・プログラムでエラーを処理することは、システム間通信 (ISC) 環境では特に有用です。

## CICS-z/OS Communications Server のエラー処理の背景情報

CICS-z/OS Communications Server LU 制御によって検出されたエラーはキューに入れられ、CICS ノード・エラー・ハンドラー (トランザクション CSNE) という特殊なタスクによって処理されます。

CICS は、いくつかのハウスキーピング処理のためにその同じタスクを使用します。例えば、「good morning」メッセージを送信したり、セッションの開始や終了をログに記録したりする処理です。そのような状態はエラーではありません。

まれに、z/OS Communications Server によって CICS にシグナル通知される例外がエラーとして処理されず、ノード・エラー・ハンドラーに渡されない、というケースもあります。例えば、CICS は、自動トランザクション開始処理の一部として z/OS Communications Server の BID コマンドを送信することがあります。例外コード 0813 (待機) で BID が拒否されるのは標準的な応答であり、CICS は、その状態をエラーと見なせずに端末管理で再試行を処理します。この後の説明では、エラーだけを取り上げます。

CSNE タスクは、バックグラウンド・タスクとして実行されるので、いずれか 1 つの CICS 端末に関連付けられるわけではありません。いつでもそのようなタスクが最大で 1 つ存在し、1 つのノード・エラー・キューで処理を実行します。

そのキューに入っているすべてのノード・エラーは、テーブルをベースにした DFHZNAC (ノード異常条件プログラム) という CICS 提供プログラムによって分析されます。そのプログラムは、絶対に変更しないでください。

DFHZNAC は、ほとんどのノード・エラーを処理する時に、DFHZNAP というモジュールが CICS システムに存在すればそのモジュールにリンクします。(特定のノードに関連していないエラーの場合は、DFHZNAP にリンクしません。例えば、z/OS Communications Server のシャットダウンが原因になっているエラーなどがそれに当たります。) このリンクのインターフェースについては、[113 ページの『異常条件が発生した場合』](#)を参照してください。DFHZNAC と DFHZNAP の間で機能するこのインターフェースを使用すれば、エラー状態を分析する独自のコードを用意したり、さまざまなアクション・フラグを設定してデフォルト・アクションを変更したり、アプリケーションに適した追加のアクションを実行したりできます。

CICS には、事前に生成されたデフォルトの DFHZNAP が用意されています。z/OS Communications Server のストレージの問題が検出されると、このプログラムによって「TCTTE を出力する」アクションのフラグが設定され、DFHZNAC に制御が戻されます。他のすべてのアクション・フラグは変更されずにそのまま残るので、DFHZNAC のデフォルト・アクションは影響されません。(さまざまなエラー状態に対する DFHZNAC のデフォルト・アクションについては、[ノード異常条件プログラムのデフォルト・アクション](#)を参照してください。)

### NEP を使用して CICS のデフォルトのアクションを補完する理由

さまざまな理由で、独自のノード・エラー・プログラムを作成しなければならない場合があります。

独自のノード・エラー・プログラムを作成して、CICS および z/OS Communications Server に用意されているデフォルトのアクションを拡張する理由を、いくつか以下に示します。

- すべてのエラーが通信システム障害を表すわけではありません (ゼロの長さのデータを書き込もうとするなどの) 一部のエラーはアプリケーションの特殊な状態を表していて、特殊なアクションが必要な場合があります。
- DFHZNAC から送信されるエラー・メッセージに加えて、追加のデータを出力したい場合があります (ノード・エラー・プログラムでは、DFHZNAC からのメッセージを抑止できないことに注意してください)。DFHZNAC と DFHZNAP からのすべての出力データは、一時データ・キュー CSNE に書き込まれます。
- その他のケースで、CICS で生成される診断情報の量を変更したい場合があります。デフォルトはエラー・タイプに応じて異なります。例えば、エラーに関連付けられた z/OS Communications Server の RPL が、不要なのに出力されたり、必要なのに出力されなかったりします。
- ノード・エラーが発生したときに実行すべきアプリケーション関連のアクティビティがある場合があります。例えば、メッセージを端末に送信できなかった場合は、通常、別の端末にリダイレクトする必要があります。例外応答のみを指定して送信されたメッセージの場合、CICS にはメッセージを再送信するために使用可能なデータが存在しなくても、要求元アプリケーションでメッセージを再作成できる可能性

があります。例えば、プリンターにドキュメントを送信中にエラーがシグナル通知された場合に、最初から再開することも、特定のページから再開することもできます。

- 3650 Retail Store System などの一部のデバイスは、「ユーザー・センス・データ」フィールドにアプリケーション・タイプのデータを返します。これは NEP でのみ取得できます。NEP で、これ以降のアプリケーション・プログラムのデータをキャッチして保管する必要があります。

## NEP 作成の概要

DFHZNEP モジュールは、定義されているインターフェースに準拠していなければなりません。つまり、定義されている通信域フィールドを使用してエラーを分析してから DFHZNAC に戻るリンク先のプログラムでなければなりません。1 つの NEP を作成するためのスケルトンとして、CICS に用意されているデフォルト NEP のソース・コードを使用できます。

CICS には、複雑なサンプル NEP を生成するためのマクロも用意されています。サンプル NEP は、独自の NEP を開発するための補助ツールであり、そのまま使用する必要はありません。

独自のエラー処理ロジックをいくつかのモジュールに分けてコーディングすることもできますが、DFHZNAC から制御を受け取るモジュールは DFHZNEP という名前ではなりません。

DFHZNEP コードでは、他のユーザー・モジュールを呼び出すために CICS の標準機能 (LINK、XCTL) を使用できます。そのようにして呼び出す各モジュールには、インストール済みの CSD プログラム定義か自動インストール済みのプログラム定義が必要です。DFHZNAC と DFHZNEP のプログラム・リソース定義は、IBM 提供の CSD グループ DFHVTAM に用意されています。

DFHZNAC と DFHZNEP の間に入るインターフェースの主要機能を以下にまとめます。

- DFHZNEP は、CICS でサポートされているどの言語でも作成できます。

注：CICS 提供の NEP コードは、アセンブラ言語のみで用意されています。通信域パラメーター・リストには、アセンブラ言語版と C 言語版があります。

- DFHZNEP は、キューに入っているノード関連エラーごとに別々にリンクされます。(エラーにはいくつかのセンス・コードが常に関連付けられているため、DFHZNEP はセンス・コードごとに別々にリンクされるわけではない点にご注意ください。)
- 2 つのモジュールの間の通信には、通信域 (DFHNEPCA) を使用します。

通信域の構造については、[114 ページの『通信域』](#)を参照してください。

DFHZNEP の呼び出しごとに、通信域の 1 つのフィールドに、DFHZNAC によって割り当てられる 1 バイトの内部エラー・コードが入ります。そのエラー・コードでエラーのタイプを特定できます。その他のフィールドによって、エラーに関連する CICS TCTTE (LU) や SNA センス・コードを確認できます。さらに、DFHZNEP から DFHZNAC にユーザー・メッセージを渡すためのフィールドもあります。そのようなメッセージは、DFHZNAC でログに記録されます。

アクション・フラグが入るフィールドもあります。各フラグによって、DFHZNEP が DFHZNAC に制御を戻した時に DFHZNAC が実行できるアクションを指定します。アクションには、以下のようなタイプがあります。

- レポート (制御ブロックやアクションのダンプ)
- 状況の変更 (TCTTE などの状況)
- クリーンアップ処理 (関連するトランザクションのキャンセル、z/OS Communications Server セッションの終了)

アクション・フラグは、DFHZNEP 内で設定したりリセットしたりできます。

特定のエラー・コードやセンス・コードに対して DFHZNAC で設定されるアクション・フラグについては、[ノード異常条件プログラムのデフォルト・アクション](#)を参照してください。

## デフォルトの NEP

CICS 提供のデフォルトの NEP である DFHZNEP は、z/OS Communications Server のストレージの問題が検出された場合に、「TCTTE を出力する」アクションのフラグを設定します (ユーザー・オプション・バイト TWAOPT1 の TWAOTCTE。[117 ページの『ユーザー・オプション・バイト \(TWAOPTL\)』](#)を参照してください)。そうでない場合は、何の処理も実行せず、DFHZNAC によって設定されたアクション・フラグを変更せず、制御権を DFHZNAC に返します。

## サンプル NEP

サンプルのノード・エラー・プログラム (NEP) は、論理装置から検出されたエラーを処理するための汎用プログラム構造です。

サンプル NEP のコンポーネントはいずれも、標準 CICS 生成プロセスの一環としては生成されませんが、その代わりにこのセクションと [121 ページの『サンプル・ノード・エラー・プログラム』](#) に説明されているようにオプションで生成することができます。

CICS 提供のサンプル NEP は、以下の 2 つの主要な特色を備えた設計となっています。

- エラー・タイプのさまざまな「グループ」を処理するために、別々のユーザー提供エラー・プロセッサを呼び出すことを前提としています。ある 1 つのルーチンで処理する対象となる、同じ「グループ」と見なす DFHZNAC 内部エラー・コードを指定し、続いてそのルーチン用のコードを準備します。CICS には、いくつかの役立つ標準的なケースが用意されています。
- 提供されているエラー・プロセッサは、ノード・エラー・テーブルという別個に生成されたモジュールと連携して機能することがあります。これを使用すると、NEP が処理するエラー・グループごとに統計を作成できます。このテーブルは、サンプル端末エラー・プログラムで使用する端末エラー・テーブル DFHTEPT と類似しています。

CICS 提供のエラー・プロセッサのいくつかは、ノード・エラー・テーブルを使用します。例えば、3270 LU に影響を与えるエラー (GROUP=1) 用のものがあります ([125 ページの『DFHSNEP TYPE=DEF3270 で組み込む 3270 論理装置用のエラー・プロセッサ』](#)を参照)。

## ノード・エラー・テーブル

サンプル NEP を理解するために、まず、ノード・エラー・テーブルの構造について詳しく説明します。

ノード・エラー・テーブルは、多くの場合、NET と省略されます。この頭字語を、「ネットワーク」の「ネット」や NETNAME と混同しないようにしてください。

ノード・エラー・テーブルは、CICS マクロの DFHSNET を使用して生成できます。[122 ページの『ノード・エラー・テーブル』](#) および [127 ページの『DFHSNET で生成されるノード・エラー・テーブル』](#) を参照してください。このテーブルをどれくらい複雑にするか選択します。

ノード・エラー・テーブルは、RESIDENT プログラムとして定義する必要があります。そうすると、(CICS LOAD 要求を使用して) NEP がこのテーブルを簡単に検出できます。また、再ロードによってカウンターがリセットされることを防止できます。テーブルは任意の名前にすることができます。デフォルトは DFHNET です。

テーブルは、エラーの記録域のセットから構成されます。各セットはノード・エラー・ブロック (NEB) と呼ばれ、単一の LU に関連するノード・エラーをカウントするために使用されます。CICS 実行の全体を通して、特定の NEB を特定の LU の専用にすることができます。そして、その他の再使用可能な NEB を一般使用のための NEB にすることができます。10 個の LU に関するエラー統計を集める場合は、10 個から 12 個の NEB が必要です。

各 NEB には複数の記録域が含まれていて、区別するエラーのグループごとに 1 つの記憶域を使用します。このエラー・グループは、NEP 内のエラー・グループに対応します。つまり、個別の処理ロジックを必要とするエラー・タイプのグループです。

各記録域を、エラー状況ブロック (ESB) と呼びます。各 ESB のために予約するスペースを指定します。通常は、エラーをカウントするスペース、または現在のシリーズの 1 つ目がいつ発生したかを記録するスペースが含まれます。1 つの NEB では、1 つの LU についてのみカウントすることに注意してください。

最後に、しきい値カウントおよび時間制限をテーブルに指定できます。これらは、NEP のコードで ESB をテストするために使用できる定数です。それにより、特定のタイプのエラーが、指定された間隔内にしきい値の回数よりも多く発生したかどうかを確認できます。この時間制限は、1 つの汎用 NEB を、ある LU のために使用してから別の LU のために再利用するまでの間隔にも影響を与えます。

最小の NET は、対象のすべてのエラー・タイプをグループ化した ESB を 1 つのみ含む少数の NEB で構成されます。

## サンプル NEP のコーディング

サンプル NEP は、DFHSNEP マクロを使用してコーディングされています。

基本的な形式は、以下のとおりです。



```
DFHSNEP TYPE=INITIAL
```

Specific error handling code. For example:

```
DFHSNEP TYPE=DEF3270
```

```
DFHSNEP TYPE=FINAL  
END      DFHNEPNA
```

デフォルトでは、これは DFHZNEP というモジュールを生成し、DFHNET というノード・エラー・テーブルを使用します。別のテーブルを使用する場合は、TYPE=INITIAL の後に NETNAME=MYTABLE とコーディングします。DFHSNEP マクロの詳細については、[124 ページの『サンプル・ノード・エラー・プログラムの生成』](#)で説明しています。

サンプル・コードを理解するには、TYPE=DEF3270 を指定して標準的な NEP を生成し ([125 ページの『DFHSNEP TYPE=DEF3270 で組み込む 3270 論理装置用のエラー・プロセッサ』](#)を参照)、生成されたアセンブラ言語のリストを確認してください。コードについて以下に説明します。

INITIAL および FINAL マクロは、NEP の基本的なスケルトンを生成します。これは、いくつかの初期設定コードといくつかの共通ルーチンから構成されます。すべてのコードは、前述のノード・エラー・テーブルが既に存在するという想定に立っています。

初期コードは、まず、DFHZNAC から渡された内部エラー・コードをテストして、NEP が処理する必要があるグループに属しているかどうかを判別します (これらのグループは、DFHSNEP INITIAL および FINAL マクロの間に指定するコードで指定します。これについては、[124 ページの『サンプル・ノード・エラー・プログラムの生成』](#)で説明しています)。特定のエラー・コードが NEP の対象でない場合、制御権は一度 DFHZNAC に返され、デフォルトのアクションが実行されます。

そうでない場合は、関連するノード・エラー・テーブルを CICS LOAD 要求によって検出します (前述のように、このテーブルが仮想ストレージ内に存在している必要があります)。そして、NEP コードで、選択された NEB 内の適切な ESB を検出します。NEB は、エラーが発生した LU の専用として永続的に確保された NEB (名前付き NEB) である場合もあれば、汎用プールから取得された NEB である場合もあります。

初期コードで、そのエラー・グループに対応する適切なユーザー・ロジックを起動します。また、初期コードで、通信域、NEB、および ESB へのポインターも設定します。詳しくは、[124 ページの『サンプル・ノード・エラー・プログラムの生成』](#)を参照してください。

NEP の共通ルーチンは、ユーザー作成ロジックのために一般的なサービスを備えています。これらは ESB 内にエラーのカウントおよびタイム・スタンプを記録し、エラーしきい値を超過していないかどうかをテストします。これらについての説明は、サンプル・リスト内にしかありません。必要に応じて、これらを使用せずに NEP を生成することもできます。

ユーザー作成のコードを、DFHSNEP TYPE=INITIAL マクロと TYPE=FINAL マクロの間に挿入します。

**注：**DFHSNEP マクロの間に挿入したユーザー・コードに EXEC CICS コマンドが含まれている場合は、コマンドを変換し、変換後のコードを DFHSNEP マクロの間に入力する必要があります。

特定のグループのエラー・タイプを処理するためのユーザー・ロジックを含む各セクションの前に、次のタイプのマクロを指定します。

```
DFHSNEP TYPE=ERRPROC, CODE=(ab, cd, ...), GROUP=n
```

ここで、X'ab'、X'cd' などは、処理する DFHZNAC 内部エラー・コードです。n はエラー・グループの数であり、したがって、ノード・エラー・テーブルの NEB 内に含まれる、対応する ESB の数でもあります。連続する DFHSNEP TYPE=ERRPROC マクロでは、グループ 1、2、3 などを使用する必要があります。

DFHSNEP TYPE=ERRPROC マクロには、いくつかの役割があります。それらを以下に示します。

- NEP の生成に、エラー・グループの数を通知する
- 各グループに含まれるエラー・タイプを示す
- 各グループのコードを導入する

1 つの DFHZNAC エラー・コードは、1 つのエラー・グループにしか属しません。また、指定されていないコードは NEP によって無視されます。各 DFHSNEP TYPE=ERRPROC マクロの後に、ユーザー作成ロジック

クを指定します。ロジックは、レジスターを保管する標準的なコードで開始するか、アドレッシング機能をセットアップする必要があります。これはサンプル NEP リストからコピーすることをお勧めします。

CICS には、2 種類の LU の特定のエラーを処理する標準的なエラー・プロセッサがいくつか用意されています。非 SNA 3270 (CICS と z/OS Communications Server に接続された BSC 3270) 用のものと、3767 などの対話式 SNA 論理装置用のものです。これについては、[113 ページの『異常条件が発生した場合』](#)で詳しく説明しています。

非 SNA 3270 用のコードは、次のようにコーディングして生成できます。

```
DFHSNEP TYPE=DEF3270
```

その他の場合は、DFHSNEP TYPE=ERRPROC マクロとユーザー作成ロジックをコーディングします。実際には、TYPE=DEF3270 は 2 つのエラー・グループを定義し、それぞれにエラー・プロセッサを関連付けます。最初のグループは、4 つの DFHZNAC エラー・コード X'D9'、X'DC'、X'DD'、および X'F2' から構成されます。2 番目のグループには、「プリンターを使用できません」という状況に対応するエラー・コード X'42' のみが含まれます。これは、CICS が 3270 の印刷要求に応答してプリンターを割り当てることができない場合にシグナル通知される特別な例外条件です。

3270 サンプル・コードは、すべてのエラー状況に対応するものではありません。このコードは SNA 3270 (LU セッション・タイプ 2) には適していないことに注意してください。これらで発生するエラー状況によって、さまざまな DFHZNAC エラー・コードが生成され、さまざまな処理が必要になる可能性があります。

CICS 提供のコードがその他のアプリケーション関連の理由により、十分でないことがあります。例えば、一定の時間間隔の後に、失われたセッションを再獲得する必要がある場合があります。3767 用に用意されているコードは、競合プロトコルで発生する可能性がある DFHZNAC エラー・コード X'DC' を 1 つ含む 1 つのエラー・グループのみに対応しています。

これらの CICS 提供のエラー・プロセッサを使用して、チュートリアルのもので、ユーザー・コードを作成せずに、有効な DFHSNEP リストを生成できます。

この NEP の設計には次の制限があるので注意してください。

- 想定していなかったエラー・タイプは NEP によって無視され、エラー・バケットに累積されます。
- ある特定の状況を処理する場合は、発生状況にかかわらず常に処理することをお勧めします。DFHZNAC が状況に応じて異なるエラー・コードを割り当てる場合であってもです。例えば、SNA 3270 では、TEST 状態との切り替えによって、X'082B' の状況が生成されます (表示スペースの完全性が失われます)。これは、複数の DFHZNAC エラー・コードのうちのいずれかを生成します。

サンプル NEP の構造では、この最後のケースを別のエラー・プロセッサでテストするか、すべての DFHZNAC エラー・コードをグループ化する必要があります。独自の NEP コードを最初から作成した場合は、NEP への入り口で、この状況を含む通信域のフィールドをテストします。

### 複数の NEP

特定のプロファイルやセッションや端末タイプを使用するすべてのトランザクションに適用する NEP トランザクション・クラスを定義できます。

そのためには、PROFILE、SESSIONS、TYPETERM の各リソースの NEPCCLASS 属性を使用します。(PROFILE リソースで指定する NEPCCLASS の値は、SESSIONS リソースや TYPETERM リソースで指定する値より優先されます。) NEPCCLASS は、1 バイトの 2 進数フィールドであり、0 から 255 の範囲の値が入ります。NEPCCLASS の目的は、LU でトランザクションを実行している間、そのトランザクションに適した特殊バージョンのノード・エラー処理を実行できるようにする、ということです。(そのような処理のことをトランザクション・クラス・エラー・ルーチンと呼ぶこともあります。) デフォルト値の NEPCCLASS(0) は、有効な NEPCCLASS がないという意味です。

DFHZNAC から制御を取得する DFHSNEP は、エラーに関連した LU について、制御取得時点で有効になっていた NEPCCLASS を検査しなければなりません。その後、適切なモジュール (実際の NEP) に制御を移すか、自身の中の特定ビットのコードに分岐します。

DFHSNEPI マクロ ([132 ページの『DFHSNEPI マクロ』](#)を参照) によって DFHSNEP モジュールが生成されます。このモジュールは、ルーティング専用のモジュールです。DFHZNAC から渡されたノード・エラーについて、有効な NEPCCLASS を検査し、マクロによって作成された NEPCCLASS/名のルーティング・テーブルに基づいて、別のモジュール (実際の NEP) に制御を移します (リンクします)。



有効な NEPCCLASS がない場合 (CEDA DEFINE PROFILE NEPCCLASS(0) の場合に相当) や該当する NEPCCLASS がルーティング・テーブルにない場合は、デフォルトのモジュールが呼び出されます。その名前を DFHZNPEI TYPE=INITIAL マクロで指定しなければなりません。(133 ページの『DFHZNPEI TYPE=INITIAL: デフォルト・ルーチンの指定』を参照してください。) その名前を指定しなければ、モジュールの呼び出しは行われません。

さまざまな NEP トランザクション・クラスに対応したサブ NEP も指定しなければなりません。もちろん、その中にデフォルトの NEPCCLASS(0) に対応したサブ NEP も含めます。それぞれのサブ NEP には別々のプログラム定義が必要です。それぞれのサブ NEP をコーディングする時にも、NEP が 1 つだけの場合と同じ選択肢があります。つまり、独自の NEP を使用するか、CICS サンプル・マクロ DFHSNEP を使用するか、という選択肢です。DFHSNEP を使用する場合は、DFHSNEP TYPE=INITIAL マクロに NAME= というもう 1 つのオペランドがあります。このオペランドを使用して、生成するモジュールに DFHZNPEI のルーティングに合わせた名前を設定できます。サブ NEP ごとに別々のノード・エラー・テーブルを使用することも可能です。

NEP のルーティングを開始する前に考えておくべく注意点を以下にまとめます。

- LU (TCTTE) とトランザクション NEPCCLASS との関連付けが有効なのは、CICS タスクが存在している間にほぼ限られます。出力メッセージの遅延などの問題で CICS タスクの終了後に検出されるエラーは、作成側のトランザクションの NEPCCLASS に関連付けられない可能性があります。

別の CICS タスクからの内部要求の結果として (例えば、EXEC CICS START 要求などによって)、CICS が LU で新しいタスクを開始しようとする場合にも、問題が発生する可能性があります。そのような処理のことを自動トランザクション開始と呼ぶこともあります。そのような場合は、CICS がタスクの開始前に、もしセッションが存在しなければ、z/OS Communications Server SIMLOGON 要求を実行して新しいセッションを開いてから、前述のとおり、BIND コマンドを送信しなければなりません。そのすべてが正常に完了するまで、対象のタスクは追加されません。その時まで、NEPCCLASS はトランザクション定義から選択されません。したがって、ATI プロセスで発生するエラー (通常は BIND または BID のエラー) は、NEPCCLASS が正しく設定される前に発生するので、NEPCCLASS の NEP ではなくデフォルトの NEP にルーティングされる可能性があります。そうすると、アプリケーションの全体的なノード・エラー処理が複雑になってしまいます。

例えば、夜間に無人の状態ではプログラマブル・コントローラーにアクセスしてその日の入力を読み取るアプリケーションがあるとします。そのようなアプリケーションのリカバリー設計はごく基本的なものであり、ATI とファイル送信の両方のエラーに対応すればそれで済みます。その 2 つを 2 種類の NEP に分離すると、処理が複雑になりすぎる可能性があります。

- NEPCCLASS レベルで NEP を分離すると、開発のための労力がかかりすぎ、効率的ではないかもしれません。一般に、ロジックを分離しなければならない場合は、LU レベルで分離してください (プログラマブル・コントローラーが 3270 以外のアプリケーションを実行する場合もあります)。

この概要の締めくくりとして、以下の点を頭に入れておいてください。CICS サンプル NEP は、独自の NEP を作成するためのアイデアの宝庫ではありますが、それぞれのニーズに最適なフレームワークであるとは限りません。まずは、それぞれにニーズに合わせてシンプルな NEP を作成することをお勧めします。

## 異常条件が発生した場合

論理装置で異常条件が検出されると、以下の CICS コンポーネントが呼び出されます。

- 端末管理プログラムの z/OS Communications Server for SNA セクション: DFHZCA、DFHZCB、DFHZCC、DFHZCP、DFHZCQ、DFHZCW、DFHZCX、DFHZCY、DFHZCZ。
- ノード異常条件プログラム DFHZNAC。
- CICS 提供のデフォルト・ノード・エラー・プログラム DFHZNPEI または独自バージョンのプログラム。

論理装置の場合は、端末の処理状態に関するすべての情報が TCTTE と要求パラメーター・リスト (RPL) に入ります。したがって、論理装置の端末エラーを処理しなければならない時には、TCTTE 自体がシステム・エラー・キューに入れられます。

DFHZNAC は、論理装置から例外応答を受け取る時点でシステム・センス・コードが存在する、という前提で処理を進めます。したがって、応答の理由を確認するために分析を実行します。受け取ったシステム・センス・コードに基づいて、どのアクション・フラグを設定するか、どの要求が必要か、などを決定しま

す。センス情報がなければ、デフォルト・アクション・フラグが設定され、DFHZEMW から否定応答を端末に送信するスケジュールが設定されます。応答が未解決であれば、否定応答と一緒にエラー・メッセージも端末に送信します。

特定のインバウンド・システム・センス・コードを受け取った時に DFHZNAC で設定されるアクション・フラグについては、[ノード異常条件プログラムのデフォルト・アクション](#)を参照してください。

DFHZNAC は、指定のルーチンを実行する前に DFHZNEP にリンクします。DFHZNEP を使用すれば、DFHZNAC の対応範囲を超えた追加のエラー処理を実行したり、DFHZNAC が設定したデフォルト・アクションを変更したりできます。ノード・エラー・プログラムをコーディングする必要があるのは、そのどちらかを実行する場合に限られます。

DFHZNAC がノード・エラー・プログラムを支援するために設定するアクション・フラグは、通信域の TWAOPTL フィールドに入ります。

異常状態後の DFHZNAC のアクションを変更しようとしている場合は、DFHZNEP から TWAOPTL フィールドの内容を確認してビット設定を変更できます。DFHZNAC の推奨アクションでよければ、TWAOPTL フィールドを変更せずにそのままにします。

ほとんどの場合は、DFHZNEP によって DFHZNAC の推奨アクションを変更できます。DFHZNEP による TWAOPTL フィールドの変更を DFHZNAC がオーバーライドするのは、論理装置を CICS から切断する場合に限られます。つまり、異常状態の結果、CICS から論理装置に対して ACF/SNA CLSDST マクロを実行する必要があると DFHZNAC が判断した場合に限られます。そのような場合は、DFHZNAC が端末を切断し、タスクを異常終了させます。DFHZNEP がそのようなアクションをブロックしようとしたとしても、DFHZNAC のアクションが優先されます。

否定応答が論理装置に送信された場合や、DFHZEMW が論理装置にエラー・メッセージを書き込まなければならない場合も、ノード・エラー・プログラムによるタスク終了フラグのリセットが無視されます。

ノード・エラー・プログラムは、機能を実行すると、**EXEC CICS RETURN** コマンドによって DFHZNAC に制御を戻します。

DFHZNEP から制御が戻されると、DFHZNAC は、TWAOPTL フィールドで指定されているアクションを実行します(前述のとおり、論理装置を切断する場合は別です)。さらに、必要に応じて、メッセージを送信したりエラー・コードを設定したりします。

## 通信域

DFHZNEP は、DFHZNAC から制御を受け取ると、**ADDRESS COMMAREA** API コマンドによって通信域のアドレスを取得します。

通信域の全般的な構造を [114 ページの図 27](#) に示します。

Header
Error_being_processed
User option bytes
VTAM information
Additional information for NEP
Additional system parameters
XRF parameters

図 27. 通信域の全般的な構造

通信域の各セクションの意味を以下にまとめます。

## Header

すべてのユーザー置換可能プログラムに共通する 4 バイトのヘッダー。

## Error\_being\_processed

エラー・コードとエラーに関連する端末の ID。

## User option bytes

DFHZNAC で設定されているデフォルト・アクションを示すフラグ (デフォルト・アクションは DFHZNEP でリセットされる可能性があります)。

## z/OS Communications Server information

センス・コードと RPL コード。

## Additional info. for NEP

NEP に関する他の役立つ情報。

## Additional system parameters

間接的なパラメーター (TCTTE など) や他のシステム情報の場所。

## XRF parameters

リカバリー通知データ。TWAXRNOT 内のフィールドは、NEP でリセットされる可能性があります。

通信域の詳細なリストについては、[115 ページの図 28](#) を参照してください。

```
*****
**                               Header                               **
**                               These fields are READ ONLY          **
*****
NEPCAHDR DS    0XL4              Standard Header
NEPCAFNC DS    XL1              Function Code      Always '1'
NEPCACMP DS    XL2              Component Code    Always 'ZC'
          DS    XL1              Reserved
*****
**                               Error_being_processed              **
**                               Identity of terminal and the error code associated with it **
**                               These fields are READ ONLY          **
*****
TWAEC     DS    XL1              Error Code
          DS    CL3              Reserved
TWANID    DS    CL4              Terminal identity
TWANETN   DS    CL8              Netname
*****
**                               User option bytes                  **
**                               Initially set to the default actions. **
**                               DFHZNEP can change the defaults.      **
*****
TWAOPTL   DS    0XL3              User option bytes
TWAOPT1   DS    XL1              User option byte 1
TWAOPT2   DS    XL1              User option byte 2
TWAOPT3   DS    XL1              User option byte 3
          DS    XL1              Reserved
```

図 28. DFHZNAC/DFHZNEP 通信域 (パート 1)

```

*****
**      z/OS Communications Sever information - Any sense and RPL codes  **
**      These fields are READ ONLY                                     **
*****
TWAUTAM DS    0XL12      z/OS Communications Sever information
TWARPLCD DS    H         z/OS Communications Sever RPL feedback codes
DS        H         Reserved
TWASNSS DS    0F         Sense codes to be sent
TWASS1 DS    XL1        System sense byte No 1
TWASS2 DS    XL1        System sense byte No 2
TWAUS1 DS    XL1        User sense byte No 1
TWAUS2 DS    XL1        User sense byte No 2
*
TWASNSR DS    0F         Sense codes received
TWASR1 DS    X          System sense byte No 1
TWASR2 DS    X          System sense byte No 2
TWAUR1 DS    X          User sense byte No 1
TWAUR2 DS    X          User sense byte No 2
*
*****
**      Additional information for the NEP                               **
**Except for TWANPFW, TWANLD, and TWANLDL these fields are READ ONLY  **
*****
TWAADINF DS    0XL22
DS        F          Reserved
TWACTLB DS    X          General use control byte
*        EQU    X'80'    Reserved
*        EQU    X'40'    Reserved
TWACSC EQU    X'20'    Clear sense code indicator
TWAPSC EQU    X'10'    Print z/OS Communications Sever sense codes
TWATIOA EQU    X'08'    Print portion of I/O area
*        EQU    X'04'    Reserved
TWAUTRTC EQU    X'02'    z/OS Communications Sever return code available
TWANEP R DS    XL1      NEP return code byte
TWANPFW EQU    X'80'    Retry write with FORCE=YES
TWAREASN DS    XL1      z/OS Communications Sever reason code
TWASTAT DS    XL1      z/OS Communications Sever status code
TWATRSN DS    XL1      CICS terminal control
*        terminal error code
TWAXRSN DS          Exception response seq number recd
TWAR EQU    *
TWAPFLG DS    XL1      CLSDST pass flag
TWAPIP EQU    X'80'    CLSDST pass in progress
TWANEP C DS    XL1      NEP class flag
TWAEISAB DS    XL1      Stand-alone begin bracket indicator
TWAESAB EQU    X'04'    Stand-alone begin bracket
DS        XL3          Reserved
TWANLD DS    A          Address of data to be logged
TWANLDL DS    H          Length of data to be logged

```

図 29. DFHZNAC/DFHZNEP 通信域 (パート 2)

```

*****
**          Additional system parameters          **
** Except for TWAPNETN, TWAPNTID, TWAUPRRRC these fields are READ ONLY **
*****
TWASYSMP DS    0XL68
TWATCTA  DS    AL4          Address of TCTTE being processed
TWARPL   DS    AL4          Address of z/OS Communications Sever RPL
TWATIOAA DS    AL4          Address of data portion of TIOA
TWATIOAL DS    H           Length of data portion of TIOA
TWACOMML DS    H           Length of commarea data for TCTTE
TWACOMMA DS    CL4          Address of commarea data for TCTTE
TWATECIA DS    AL4          Address of TCTTE user area
TWATECIL DS    H           Length of TCTTE user area
TWAPPNTN DS    CL8          Primary 3270 printer netname
TWAPPTID DS    CL4          Primary 3270 printer termid
TWAPPELG DS    X           Primary printer eligible indicator
TWAPPELY EQU  X'01'        Primary printer is eligible flag
TWASPNTN DS    CL8          Secondary 3270 printer netname
TWASPTID DS    CL4          Secondary 3270 printer termid
TWASPELG DS    X           Secondary printer eligible indicator
TWASPELY EQU  X'01'        Secondary printer is eligible flag
TWAPNETN DS    CL8          Selected 3270 printer netname
TWAPNTID DS    CL4          Selected 3270 printer termid
TWAUPRRRC DS    B           Unavailable Printer return code
TWAUPRNP EQU  X'00'        No printer selected
TWAUPRPS EQU  X'01'        Printer selected
TWAUPRDD EQU  X'FF'        Data disposal complete
TWAUPRPE EQU  X'FE'        Error on Put request
TWAERRF1 DS    B           Error flag byte 1
TWALXS   EQU  X'80'        Logon crossed simlogon
          DS    XL2          Reserved
*****
**          XRF parameters          **
**          XRF recovery notification data          **
**          DFHZNEP can change these default actions          **
*****
TWAXRNOT DS    X           Recovery notification options
TWAXRNON EQU  X'80'        Recov notification = none
TWAXRMSG EQU  X'40'        Recov notification = message
TWAXRTRN EQU  X'20'        Recov notification = transact.
          DS    XL3          Reserved
TWAXMSTN DS    CL8          Recovery mapset name
TWAXMAPN DS    CL8          Recovery map name
TWAXTRAN DS    CL4          Recovery transaction ID
*

```

図 30. DFHZNAC/DFHZNEP 通信域 (パート 3)

以下の各セクションでは、DFHZNEP でリセットされる可能性のあるパラメーター・リスト内のフィールドを取り上げます。「unavailable printer return code」フィールドのフラグを使用法を取り上げた [130 ページ](#) の『3270 「unavailable printer」状態に対応するためのコーディング』も参照してください。

### ユーザー・オプション・バイト (TWAOPTL)

TWAOPTL には、ユーザー・オプション・バイト TWAOPT1、TWAOPT2、TWAOPT3 が含まれていて、それぞれにアクション・フラグが入ります。DFHZNEP への入り口では、これらのフラグは、DFHZNAC で設定されているデフォルト・アクションになります。デフォルト・アクションは、DFHZNEP でリセットできます。

#### TWAOPT1

ユーザー・オプション・バイト 1。TWAOPT1 には、主にデバッグに役立つフラグが入ります。最初の 5 つのフラグで該当するビットを設定すると、DFHZNAC が該当する情報を CSNE ログに書き込みます。6 番目のフラグ (TWAODNTA) をオンに設定し、TWAOPT2 の TWAOTAT フラグもオンに設定すると、エラー検出の時点で端末にタスクが接続されていなければ、CICS がシステム・ダンプを生成します。TWAONQN フラグを設定すると、そのアクション・フラグが入っているメッセージの後にネットワーク修飾名が出力されます。同じように、TWAOTNA フラグを設定すると、TNADDR 情報が出力されます。

以下のフラグがあります。

#### TWAOAF (X'80')

アクション・フラグを出力します。

**TWAORPL (X'40')**

z/OS Communications Server の RPL を出力します。

**TWAOTCTE (X'20')**

TCTTE を出力します。

**TWAOTIOA (X'10')**

TIOA を出力します。

**TWAOBIND (X'08')**

BIND 域を出力します。

**TWAODNTA (X'04')**

タスクが接続されていない場合にシステム・ダンプを生成します。

**TWAONQN (X'02')**

NQNAME を出力します。

**TWAOTNA (X'01')**

TNADDR (TCP/IP クライアント・アドレス、ポート、さらにオプションとしてホスト名) を出力します。

注:

1. DFHZC2411 は、特定のノードに関連していません。つまり、TCTTE はまだ作成されておらず、メッセージはダミー TCTTE に出力されます。その場合、ノード・エラー・プログラムは呼び出されないの、デフォルト設定をオーバーライドすることはできません。したがって、DFHZC2411 メッセージでは、NQNAME と TNADDR 情報が常に出力されます。
2. DFHZC2410 がダミー TCTTE に対して生成される場合、NQNAME と TNADDR は出力されません。

**TWAOPT2**

ユーザー・オプション・バイト 2。TWAOPT2 には、タスク関連のフラグが入ります。

NEP は、TWAOTAT を設定してタスクを異常終了させることも、TWAOTCT を設定してタスクを取り消すこともできます。タスク異常終了は、タスクが端末管理操作を要求するか完了するまで有効になりませんが、タスク取り消しは、システムやデータの保全性を維持できる状態になったらすぐに有効になります。通常は、TWAOTAT を設定してタスクを異常終了させるだけで十分ですが、タスクが端末要求の間に長い処理 (データベース参照など) を実行する場合は別です。TWAOTAT と TWAOTCT を両方とも設定した場合は、TWAOTCT (タスク取り消し) のほうが優先されます。

タスクを異常終了させると、送信と受信がパージされます。TWAOGMM を設定すると、次の transid がクリアされ、端末に関連する通信域が解放されますが、TERMINAL 定義で TRANSACTION(name) として永続 transid が指定されている場合は、通信域が解放されません。端末の TYPETERM で「good morning」メッセージがサポートされていて (LOGONMSG(YES))、TWAONINT がオフになっていて、端末が BMS ページング・セッションに参加していない場合は、「good morning」メッセージ・トランザクション (システム初期設定パラメーター GMTRAN で指定されているトランザクション) が開始されます。

以下のフラグがあります。

**TWAOAS (X'80')**

この端末のすべての SEND を放棄します。

**TWAOAR (X'40')**

この端末のすべての RECEIVE を放棄します。

**TWAOTAT (X'20')**

TCTTE に接続されているすべてのタスクを異常終了させます。

**TWAOTCT (X'10')**

TCTTE に接続されているすべてのタスクを取り消します。

**TWAOGMM (X'08')**

「good morning」メッセージを送信します。

**TWAOPBP (X'04')**

このセッションのすべての BMS ページをパージします。



## **TWAOASM (X'02')**

SIMLOGON が必要です。

注:

1. 確定応答の SEND が実行された場合は、CICS がその応答を取得するために RECEIVE を実行しなければなりません。否定応答の場合は、DFHZNAC が実行され、TWAOAS フラグ (SEND の放棄) と TWAOAR フラグ (RECEIVE の放棄) が設定されます。応答の RECEIVE を放棄するために、TWAOAR を常にオンにしておく必要があります。
2. 要求を再試行する時に、接続切断アクション・フラグをオフにした場合 (つまり、TWAOPT3 の TWAOCN をオフにした場合) は、TWAOAS、TWAOAR、TWAONEGR のいずれか 1 つ以上と TWAOAT をオフにする必要があります。
3. TWAOCT を設定した結果として返される異常終了コードは予測不能です。
4. TWAOGMM がノード・エラー・プログラムによってオンに設定されると、TWAOAT が強制的に設定されます。
5. TWAOPBP を設定すると、TWAOAT が強制的にオンに設定されます。
6. 非パイプライン端末では、タスクの初回のディスパッチがまだ実行されていない場合に、TWAOAT が取り消し要求 (TWAOCT) として機能します。

## **TWAOPT3**

ユーザー・オプション・バイト 3。TWAOPT3 には、ノード関連のフラグが入ります。

以下のフラグがあります。

### **TWAOINT (X'80')**

内部生成ログオン (INTLOG) を実行できます。

### **TWAOINT (X'40')**

内部生成ログオンを実行できません。エラー・コード X'49' (TCZCLSIN) を処理する場合は、このフラグを設定しないでください。

### **TWAONCN (X'10')**

通常の CLSDST です (リセット不能)。

### **TWAOSCN (X'08')**

通常の CLSDST です (リセット可能)。

### **TWAONEGR (X'04')**

否定応答を送信します。

### **TWAOOS (X'02')**

ノードをサービス休止状態のままにします。

### **TWAOCN (X'01')**

CLSDST ノード。エラー・コード X'49' (TCZCLSIN) を処理する場合は、このフラグを設定しないでください。

TWAOINT を設定すると、TWAOCN が強制的に設定されます。

TWAONEGR を設定すると、TWAOAR と TWAOAT が強制的に設定されます。

TWAOOS を設定すると、TWAOCN が強制的に設定されます。

TWAOCN を設定すると、TWAOAR、TWAOAS、TWAOAT が強制的に設定されます。

TWAOOS は、このノードでは処理がそれ以上実行されない、という意味です。そのノードは、論理的にサービス休止状態になります。

LU6.1 のシステム間連絡セッションでは、TWAOOS や TWAONINT を設定した場合に、指定のアクションの結果として割り振り可能なセッションが残っていない状態になると、システム・エントリーがサービス休止状態になります。(APPC セッションをバインドしようとしている時に不明なモードネームが z/OS Communications Server に渡された場合や、z/OS Communications Server の 3270 タイプの端末のログモード名が無効だった場合にも、セッションがサービス休止状態になることがあります。ただ

---

<sup>1</sup> エラー・コード X'49' (TCZCLSIN) を処理する場合は、このフラグを設定しないでください。

し、その状態の結果として発生する CICS のデフォルト・アクションを NEP でオーバーライドすることはできません。)

TWAOCN を設定すると、タスクが異常終了して、ノードとの通信が失われます。そのフラグを NEP でリセットすることはできません。

TWAOSCN と TWAONCN の機能は同じですが、TWAOSCN の場合は、セッションを閉じない限り NEP によるリセットが可能です。

例外応答を受け取った結果として DFHZNAC のスケジュールが設定された場合は、DFHZNAC と DFHZNAP が TCTTE 内のセンス情報を使用して、必要なアクションを判別できます。

CICS と論理装置の間の接続が失われた結果として DFHZNAC のスケジュールが設定された場合は、その障害の発生時点で進行中になっていたトランザクションを DFHZNAC が異常終了させます。DFHZNAP とトランザクション・クラス・エラー・ルーチンによる分析と処理は可能ですが、メッセージ送信を再試行するべきではありません。

ただし、アプリケーション・プログラムが TERMERR 状態を処理する場合は、トランザクションの異常終了は発生しません。制御がプログラムに戻されます。その場合、障害が発生したセッションをそれ以上使用することはできません。

### NEP (TWAADINF) の追加情報

フィールド TWANPFW、TWANLD、および TWANLDL は、NEP でリセットできます。

TWANPFW の使用方法については、用意されているサンプル・ノード・エラー・プログラム、および [124 ページ](#)の『対話式論理装置で使用するオプション・エラー・プロセッサ』を参照してください。

#### TWANLD および TWANLDL – DFHZNAC ログ機能の利用

DFHZNAC のログ機能を利用して、TWANLD フィールドと TWANLDL フィールドから情報を取得できます。

通信域の TWANLD フィールドに、検査するデータのアドレスを指定し、TWANLDL フィールドにそのデータの長さを指定します。今後の検査のために、そのデータは CSNE 一時データ・キューに記録されます。

注：220 バイトを超過するデータは記録されません。

また、一時データ機能を使用して、ユーザー作成のメッセージを CSNE ログに送信することもできます。メッセージを作成するには、EXEC CICS WRITEQ TD 命令をノード・エラー・プログラムに直接コーディングする必要があります。

#### TWAPIP とアプリケーション・ルーティング障害

EXEC CICS ISSUE PASS コマンドを使用して、CICS から名前付きの別の z/OS Communications Server アプリケーションに制御を渡せます。ISSUE PASS コマンドによって、z/OS Communications Server のマクロ CLSDST を OPTCD=PASS を指定して呼び出して、CICS に CLSDST 要求の結果を通知することも可能です。

EXEC CICS ISSUE PASS コマンドのプログラミング情報については、ISSUE PASS を参照してください。ISSUE PASS コマンドは、z/OS Communications Server のマクロ CLSDST を OPTCD=PASS を指定して呼び出します。さらに、CLSDSTP システム初期設定パラメーターで NOTIFY が指定されている場合は、PARMS=(THRDPTY=NOTIFY) も付けて呼び出します。そうすると、CLSDST 要求の結果が CICS に通知されます。

この通知の結果、通知メッセージが生成され、CLSDST 要求が失敗したか成功したかに関わらず、DFHZNAC が NEP を呼び出します。NEP は、TWAPFLG フィールドで受け渡し進行標識 TWAPIP を調べて、CLSDST OPTCD=PASS 要求が進行中かどうかを確認できます。TWAEC のエラー・コードを調べれば、CLSDST OPTCD=PASS 要求が成功したか失敗したかも判別できます。

受け渡し操作が失敗すると、DFHZNAC は一連のデフォルト・リカバリー・アクションをセットアップします。これは、NEP で変更することもできます。例えば、ターゲット・アプリケーション・プログラムがアクティブではない場合のリカバリーとして考えられるのは、SIMLOGON 要求を使用して初期アプリケーションとのセッションを再確立し、CICS から端末に「good morning」メッセージを送信する、という処理です。デフォルト・アクションは、セッションを切断状態のままにして NOCREATE にする、という処理です。

CLSDSTP=NONOTIFY が指定されている状態で自動インストールを使用すると、ISSUE PASS が失敗した場合でも、CICS はアクションを実行しません。

持続セッション・サポートがアクティブになっていると、AIRDELAY後に自動インストール端末が削除されます。したがって、CLSDSTP=NOTIFYのコーディングの結果として実行されるはずのNEP処理は発生しません。

### 追加のシステム・パラメーター (TWASYSPPM)

NEPが駆動されるときに、パラメーター・リストのこのセクションで参照されているデータ・エレメント(例えば、TIOA)が存在しない場合、そのアドレスおよび長さフィールドはゼロに設定されます。

フィールド TWAPNETN、TWAPNTID、および TWAUPRRRC は、NEPによりリセットできます。

## サンプル・ノード・エラー・プログラム

サンプル・ノード・エラー・プログラムには、エラー処理ルーチン(エラー・プロセッサー)を実行するための全般的な環境が用意されています。それぞれのプログラムは、ノード異常条件プログラムによって生成される特定のエラー・コードに対応しています。

対話式論理装置ネットワークの通常の操作に対応したオプションのエラー・プロセッサーが十分に用意されています。ユーザー作成のエラー・プロセッサーで簡単に補完したり置換したりすることもできます。

SNA ネットワークで発生するエラーには3つのタイプがあります。

- ホスト・システムのエラー
- 通信エラー(セッション障害など)
- 端末の異常条件(介入の必要な状態や無効な要求など)

CICS には、サンプル・ノード・エラー・プログラムが用意されていて、それを独自に作成するノード・エラー・プログラムの基盤として使用できます。以下の機能があります。

- エラー処理プログラムを追加するための全般的な環境
- いくつかのノード・エラー・プログラムのあるシステムのデフォルト・ノード・エラー・プログラム

CICS 提供のサンプル・ノード・エラー・プログラムの詳細については、以下の各トピックを参照してください。

### サンプルの端末エラー・プログラムとの互換性

センス・コードまたは状況コードの受け取りは、エラー・コード X'D9'、X'DC'、X'DD'、および X'F2'に対応しています。これらのメッセージの重み付きカウントは、数値および時間のしきい値と比較して維持されます。数値しきい値を超過した場合は、デフォルトのアクションが実行されます。時間のしきい値に達した場合は、カウントがリセットされます。これはサンプル TEP の機能と同じです。ただし、COPY コマンドの「from」デバイスで生成されたセンスまたは状況が、「to」デバイス上のエラーとしてノード・エラー・プログラムに示されるようになります。これにより、しきい値を超過して、端末がまだサービス中であるのに要求が終了します。3270 ディスプレイ 装置で発生するエラーの重みのいくつかは改訂されましたが、それ以外の重み値およびしきい値は、サンプル TEP で使用されるデフォルトと同じです。サンプル NEP の時間しきい値の保守は必須であり、サンプル TEP のようにオプションではありません。

時間およびしきい値のカウント制限について詳しくは、[82 ページの『端末エラー・プログラムの作成』](#)のサンプル端末エラー・プログラムについての情報を参照してください。

3270 のメッセージ「プリンターを使用できません」は、エラー・コード X'42' (インターバル制御 PUT 要求は失敗しました)に対応しています。プリンターの選択に使用されるアルゴリズムは、z/OS Communications Server サポートでは異なります。サンプル・ノード・エラー・プログラムの再試行アルゴリズムは、この新しい選択アルゴリズムに似ています。

### サンプル・ノード・エラー・プログラムのコンポーネント

サンプル・ノード・エラー・プログラムは、以下のコンポーネントで構成されています。

- 入り口セクション。
- ルーティング・メカニズム。
- ノード・エラー・テーブル。
- オプションの共通サブルーチン。

- 3270 または対話式論理装置のオプションのエラー・プロセッサ。ノード・エラー・プログラムを、3270 と対話式論理装置の両方のエラー・プロセッサを使用して生成することはできません。

各コンポーネントについて、以下のセクションで説明します。

### 入り口セクション

入り口で、サンプル NEP は DFHEIENT を使用して、基底レジスターを設定し、EXEC インターフェースに対するアドレッシング機能を取得します。**EXEC CICS LOAD PROGRAM** コマンドを使用して、ノード・エラー・テーブル (NET) および共通サブルーチン・ベクトル・テーブル (CSV T) (含まれている場合) に対するアドレッシング機能を取得します。

**EXEC CICS ADDRESS COMMAREA** コマンドを使用して、DFHZNAC から渡された通信域に対するアドレッシング機能を取得し、**EXEC CICS ADDRESS EIB** コマンドを使用して、EXEC インターフェース・ブロックに対するアドレッシング機能を取得します。時間のサポートを生成した場合は、それ以降の処理のエラーにタイム・スタンプが設定されます

### ルーティング・メカニズム

ルーティング・メカニズムは、ノード異常条件プログラムから渡されたエラー・コードに応じて、適切なエラー・プロセッサを呼び出します。

DFHSNEP マクロには、1 つ以上のエラー・コードのグループが定義されています。各グループには、インデックス (X'01' から X'FF' までの範囲内) およびエラー・プロセッサが関連付けられています。変換テーブルが生成されると、グループ・インデックスがエラー・コードごとに適切なオフセットに置かれます。グループに定義されていないエラー・コードの場合、テーブルの値はゼロになります。エラー・プロセッサ・ベクトル・テーブル (EPVT) には、インデックスに応じた位置にエラー・グループ・プロセッサのアドレスが含まれています。このベクトル・テーブルは、定義された最大のインデックスまで拡張されます。未定義の中間値はゼロのアドレスで表されます。

エラー・コードを変換して、エラー・グループ・インデックスを取得します。値がゼロの場合、ノード・エラー・プログラムはそれ以上のアクションを実行しません。そうでない場合は、そのインデックスを使用して、EPVT から適切なエラー・プロセッサのアドレスを取得します。アドレスがゼロの場合、ノード・エラー・プログラムはそれ以上のアクションを実行しません。そうでない場合、エラー・プロセッサへの呼び出しを行います。これは、NET 域および CSV T 域への直接アドレッシング機能を使用して入力されます。エラー・プロセッサが実行されたら、ノード・エラー・プログラムは、ノード異常条件プログラムに制御権を返します。

### ノード・エラー・テーブル

ノード・エラー・プログラムはノード・エラー・テーブル (NET) を使用することがあり、そのテーブルは個々のノードのエラー状況情報を保持するために使用するノード・エラー・ブロック (NEB) で構成されます。

NEB の一部またはすべては、特定のノード用に永続的に予約されている場合があります。それ以外の部分は、エラーが発生したときにノードに動的に割り当てられます。動的に割り当てられた NEB は、明示的に解放されるまで、割り当てられたノードで排他的に使用されます。どの NEB もエラー状況ブロック (ESB) の構造は同じです。各 ESB は 1 つのエラー・プロセッサ用に予約され、該当するエラー・グループ・インデックスを使ってそのエラー・プロセッサに関連付けられます。ESB の長さや形式は、関連付けられている特定のエラー・プロセッサに合わせてカスタマイズできます。

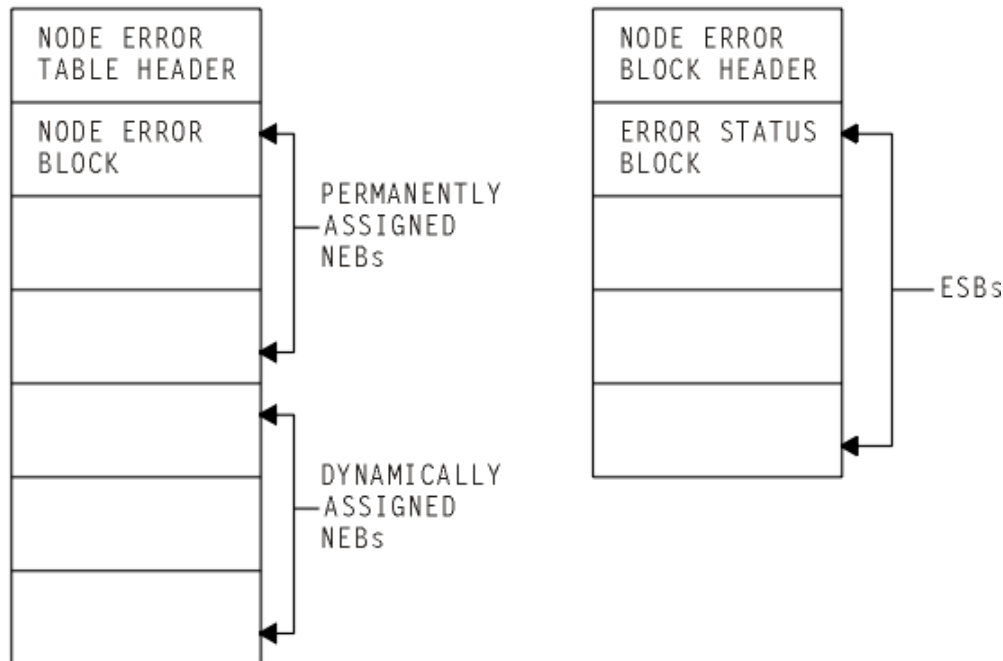


図 31. ノード・エラー・テーブルとノード・エラー・ブロックの形式

### オプションの共通サブルーチン

共通サブルーチンは、CSV T を介してアドレス指定され、エラー・プロセッサに以下の機能を提供します。

- ノード ID およびエラー・グループ・インデックスに基づいて NEB および ESB を配置または割り当てます。
- エラーにタイム・スタンプを付け、エラー・カウントを更新し、数および時間のしきい値と比較してエラー・カウントをテストします。
- 動的に割り振られた NEB を特定のノードから解放します。

### 3270 論理装置用のオプションのエラー・プロセッサ

以下のように 2 つのエラー・プロセッサが 3270 LU 用に用意されています。

1. グループ・インデックス 1、エラー・コード X'D9'、X'DC'、X'DD'、および X'F2'。

これらのエラー・コードは、RPL のユーザー・センス・フィールドのセンス・バイトまたは状況バイトの受け取りに対応しています。エラー・プロセッサは、標準形式の ESB を検出し、重み付きのエラー・カウントを更新します。重み、しきい値、およびタイマーの値は、前のセクションに記載したものを除き、サンプル端末エラー・プログラム 3270 で使用されるものに基づいています。しきい値を超過していない場合、SEND の異常終了フラグ、RECEIVE の異常終了フラグ、トランザクションの異常終了フラグ、および印刷アクションのすべてのフラグはオフです。そうでない場合は、デフォルトのアクションが実行され、再使用可能な NEB が解放されます。

2. グループ・インデックス 2、エラー・コード X'42'。

このコードは、3270 画面で生成された印刷要求を満たすために使用できる 3270 プリンターがなかったことを意味します。エラー・プロセッサは、この画面に定義されたプリンターを検査し、それらが使用できない理由を判別します。前の PRINT 要求または COPY 要求で使用中である場合 (つまり、タスクに CSPP または CSCY のトランザクション ID が接続されている)、または使用不可になっている場合は、そのプリンターのアドレスが、IC PUT コマンドを使用して印刷要求を再試行しているノード異常条件プログラムに返されます。そうでない場合には、デフォルトのアクションが実行されます (詳しくは、[130 ページの『3270 「unavailable printer」状態に対応するためのコーディング』のセクションを参照してください](#))。



## 対話式論理装置で使用するオプション・エラー・プロセッサ

1つのエラー・プロセッサが対話式論理装置: グループ・インデックス 1 用に提供され、エラー・コードは X'DC' です。

このエラー・コードとユーザー・センス値 X'081B' の組み合わせは、「受信側は送信モードです」状態を表します。TWANPFW のアクション・フラグは操作され、失敗した SEND 要求の再試行が許可されます。

## サンプル・ノード・エラー・プログラムの生成

ルーティング・メカニズム、共通サブルーチン、CICS 提供エラー・プロセッサ、およびユーザー提供エラー・プロセッサが、DFHSNEP マクロによって生成されます。

サンプル・ノード・エラー・プログラムとテーブルは、変換、アセンブル、およびリンク・エディットを行う必要があります。ユーザー置換可能プログラムのアセンブルとリンク・エディットに必要なジョブ制御ステートメントに関する情報は、[337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』](#)を参照してください。

変換プログラム・オプション NOPROLOG および NOEPILOG をノード・エラー・プログラム内にコーディングする必要があることに注意してください。

共通サブルーチン・レジスター保管域用に 24 バイト余分に必要で、またエラー・プロセッサ保管域用にさらにスペースが必要になることにも注意してください。CICS サンプル・プロセッサは、この領域の 4 バイトを使用します。

サンプル・ノード・エラー・プログラムを生成する DFHSNEP マクロには、以下の 7 種類があります。

### TYPE=USTOR

ユーザー・ストレージ定義の開始を指定します。

### TYPE=USTOREND

ユーザー・ストレージ定義の終了を指定します。

### TYPE=INITIAL

ルーティング・メカニズムおよび、必要に応じて、共通サブルーチンを生成します。

### TYPE=DEF3270

3270 デバイス用のデフォルトの CICS 提供エラー・プロセッサを生成します。

### TYPE=DEFILU

コンテンション・モードで動作する対話式論理装置用のデフォルトの CICS 提供エラー・プロセッサを生成します。

### TYPE=ERRPROC

ユーザー提供エラー・プロセッサの開始を指定します。

### TYPE=FINAL

サンプル・ノード・エラー・プログラムの終了を指定します。

## DFHSNEP TYPE=USTOR および USTOREND - ユーザー・ストレージの定義

DFHSNEP TYPE=USTOR および DFHSNEP TYPE=USTOREND マクロは、それぞれユーザー・ストレージ定義の始まりと終わりを示します。

## DFHSNEP TYPE=USTOR

DFHSNEP TYPE=USTOR マクロの形式は以下のとおりです。

```
DFHSNEP TYPE=USTOR
```

このマクロは、ユーザー・ストレージ定義の始まりを示します。この後にストレージ定義を指定し、その後 DFHSNEP TYPE=USTOREND を指定する必要があります。DFHSNEP TYPE=USTOR を使用してストレージを定義する場合には、DFHSNEP TYPE=INITIAL の前に、これと DFHSNEP TYPE=USTOREND の両方をコーディングする必要があります。



## DFHSNEP TYPE=USTOREND

DFHSNEP TYPE=USTOREND マクロの形式は以下のとおりです。

```
DFHSNEP TYPE=USTOREND
```

このマクロは、ユーザー・ストレージ定義の終わりを示します。DFHSNEP TYPE=USTOR をコーディングした場合は、これを使用する必要があります。DFHSNEP TYPE=USTOR を使用してストレージを定義する場合には、DFHSNEP TYPE=INITIAL の前に、これと DFHSNEP TYPE=USTOREND の両方をコーディングする必要があります。

### DFHSNEP TYPE=INITIAL - ルーティング・メカニズムの生成

DFHSNEP TYPE=INITIAL マクロは、サンプル・ノード・エラー・プログラムの開始を示し、ルーティング・メカニズムを生成させます。

1 つの DFHSNEP TYPE=INITIAL マクロを、DFHSNEP TYPE=USTOR および DFHSNEP TYPE=USTOREND (これらがコーディングされている場合) の直後かつ他のマクロの直前に指定する必要があります。

```
DFHSNEP TYPE=INITIAL  
[,CS=NO]  
[,NAME=name]  
[,NETNAME=netname]
```

#### TYPE=INITIAL

サンプル・ノード・エラー・プログラムの開始を示し、ルーティング・メカニズムを生成させます。

#### CS=NO

共通サブルーチンの生成を抑止するように指定します。

#### NAME=name

ノード・エラー・プログラム・モジュール ID の名前を指定します。名前は 1 文字から 8 文字のストリングである必要があります。このオペランドはオプションであり、デフォルトは DFHZNEP0 です。NAME オペランドのデフォルトを指定する場合は、DFHTEP と DFHZNEP のリンク・エディット・ステートメントの例を使用して、リンク・エディットのステートメントを作成できます。ただし、異なる NAME を指定するには、それに応じてリンク・エディットのステートメントを変更する必要があります。インターフェース・モジュール DFHZNEP (DFHZNEPI マクロによって生成されます) を使用する場合は、このオペランドを (DFHZNEP 以外の名前を使用して) 指定する必要があります。

#### NETNAME=netname

初期設定時にロードするノード・エラー・テーブルの名前を指定します。名前は 1 文字から 8 文字のストリングである必要があります。このオペランドはオプションであり、デフォルトは DFHNET です。

### DFHSNEP TYPE=DEF3270 で組み込む 3270 論理装置用のエラー・プロセッサ

DFHSNEP TYPE=DEF3270 マクロの形式は次のとおりです。

```
DFHSNEP TYPE=DEF3270
```

#### TYPE=DEF3270

3270 論理装置用の CICS 提供エラー・プロセッサをノード・エラー・プログラムに組み込むことを指定します。このマクロによって、次のソース・コードが生成されます。

```
DFHSNEP TYPE=ERRPROC,GROUP=1,CODE=(D9,DC,DD,F2)  
Sense/status error processor code.  
  
DFHSNEP TYPE=ERRPROC,GROUP=2,CODE=42  
Unavailable printer error processor code.
```

### DFHSNEP TYPE=DEFILU - INTLU のエラー・プロセッサを含む

DFHSNEP TYPE=DEFILU マクロの形式は以下のとおりです。

```
DFHSNEP TYPE=DEFILU
```

## TYPE=DEFILU

対話式論理装置用の CICS 提供のエラー・プロセッサをノード・エラー・プログラムに含めるように指定します。このマクロによって、次のソース・コードが生成されます。

```
DFHSNEP TYPE=ERRPROC,GROUP=1,CODE=DC
(receiver in transmit mode error processor code)
```

## DFHSNEP TYPE=FINAL - DFHSNEP エントリーの終わり

ノード・エラー・プログラムの終わりを示すために、他のすべての DFHSNEP マクロの後に DFHSNEP TYPE=FINAL マクロを 1 つ指定する必要があります。

形式は次のとおりです。

```
DFHSNEP TYPE=FINAL
```

## TYPE=FINAL

ノード・エラー・プログラムの終わりを示し、エラー・プロセッサ・ベクトル・テーブル (EPVT) を生成させます。EPVT は、ノード・エラー・プログラムのルーティング・メカニズムで起動されるエラー・グループ・プロセッサのアドレスが収められているテーブルです。

## DFHSNEP TYPE=ERRPROC で指定するユーザー・エラー・プロセッサ

DFHSNEP TYPE=ERRPROC マクロは、ユーザー提供エラー・プロセッサの開始を指定するために使用します。このマクロの直後に、実際のエラー・プロセッサ・コードが続きます。アセンブリの終了には、ステートメント END DFHNEPNA を使用する必要があります。

次のオペランドを DFHSNEP TYPE=ERRPROC マクロで使用できます。

```
DFHSNEP TYPE=ERRPROC
        ,CODE=(error-code,...)
        ,GROUP=error-group-index
```

## TYPE=ERRPROC

ユーザー提供エラー・プロセッサの開始を指定します。

### CODE=(error-code,...)

エラー・グループを構成するエラー・コードを指定します。つまり、提供されるエラー・プロセッサはこれらのエラー・コードを処理します。オペランドは、1 バイト 16 進数コードの 2 文字表記によるサブリストとしてコーディングされます。(単一のコードの場合は、括弧を省略できます。) 指定する各コードにおいて、エラー・グループ・インデックスが変換テーブルの等価なオフセットに配置されます。したがって、このコードが出現すると、該当するエラー・プロセッサを特定できます。

### GROUP=error-group-index

エラー・プロセッサのエラー・グループ・インデックスを指定します。このインデックスは、エラー・プロセッサを指定し、エラー・プロセッサ・ベクトル・テーブル (EPVT) でのエラー・プロセッサのアドレスを指定し、必要に応じて、エラー・プロセッサと各 NEB 内の ESB とを関連付けるために使用します。指定するインデックスは、X'01' から X'FF' までの範囲の 1 バイト 16 進数の 2 文字表記でなければなりません(先行ゼロは省略可能)。エラー・プロセッサ名の形式は NEPROCxx で、「xx」はエラー・グループ・インデックスです。この名前の CSECT ステートメントが生成され、それによりエラー・プロセッサ・コードはアセンブルされてノード・エラー・プログラム・モジュールの末尾に置かれ、独自のアドレス指定が可能になります。

独自のエラー・プロセッサをサンプル・ノード・エラー・プログラムに追加する場合は、以下の要素を考慮してください。

- 通信域のレイアウト。通信域については、[115 ページの図 28](#)で詳しく説明されています。
- 特定の関数を DFHSNEP 内で使用できないこと。(130 ページの『EXEC CICS コマンドの使用に関する制限』を参照。)
- サンプル・ノード・エラー・プログラムが使用するレジスター規則。これについては、[127 ページの表 10](#)で説明します。

表 10. レジスター割り当て	
レジスター	効果
0	作業レジスター
1	EXEC パラメーター・リストのアドレス
2	NEB 基底レジスター (DFHSNEP のみ)
3	ESB 基底レジスター (DFHSNEP のみ) NEP エラー・クラス・レジスター (DFHZNEPI のみ)
4	NEP 名ポインター・レジスター (DFHZNEPI のみ)
5	NEP インターフェース 基底レジスター (DFHZNEPI のみ)
6	作業レジスター
7	作業レジスター
8	作業レジスター
9	作業レジスター
10	コード基底レジスター
11	EIB のアドレス
12	通信域のアドレス
13	DFHEISTG ストレージのアドレス
14	CSVT 基底およびエラー・プロセッサ・リンク・レジスター 共通サブルーチン・リンク・レジスター
15	エラー・プロセッサ・ブランチ・レジスター 共通サブルーチン・ブランチ・レジスター

注:

1. レジスター 14 は、エラー・プロセッサからの戻りのために保管する必要があります。共通サブルーチン・ベクトル・テーブル (CSVT) はエラー・プロセッサへの BALR の後にコーディングされるため、このレジスターも CSVT 基底です。
2. レジスター 1、10、12、13、14、および 15 はエラー・プロセッサへの入力時に設定されます。
3. レジスター 14 から 11 までは、エラー・プロセッサを使用して、EXEC インターフェース・ストレージ内の予約済み領域のラベル NEPEPRS に保管できます。レジスター 15 から 11 までは、エラー・プロセッサからの戻りの前に復元する必要はありません。
4. レジスター 4 から 9 までは、共通サブルーチンを使用して、EXEC インターフェース・ストレージ内の予約済み領域のラベル NEPCSRs に保管できます。これらは、サブルーチンからの戻りより前に復元する必要があります。

#### DFHSNET で生成されるノード・エラー・テーブル

DFHSNET マクロは、ノード・エラー・テーブルの生成に使用します。生成するノード・エラー・テーブルは、それぞれが CICS に定義されていなければなりません。

```
DFHSNET [NAME=DFHNET|name]
        [,COUNT=100|threshold]
        [,ESBS=1|(index,length,...)]
        [,NEBNAME=(name,...)]
        [,NEBS=10|number]
        [,TIME=(7,MIN)|(interval,units)]
```

#### NAME=DFHNET|name

NET ヘッダーに組み込む ID を指定します。1 から 8 文字のストリングでなければなりません。このオペランドはオプションであり、デフォルトは DFHNET です。

### COUNT=100|threshold

エラー件数しきい値を指定します。この値は、NET ヘッダーに保管され、共通サブルーチンが標準 ESB を更新するときに使用します。しきい値を超過している場合は、サブルーチンを起動したエラー・プロセッサに対して戻りコードで通知されます。最大値は 32 767 です。このオペランドはオプションであり、デフォルトは 100 です。

### ESBS=1|(index,length,...)

各 NEB の ESB 構造を指定します。このオペランドはサブリストとしてコーディングされます。サブリストの各要素は、次の 2 つの値で構成されます。つまり、「index」は NEB に組み込む ESB のエラー・グループ・インデックスを指定します。「length」はその ESB の状況域の長さをバイト数で指定します。要素が 1 つである場合は、括弧を省略できます。「index」は、X'01' から X'FF' までの範囲の 1 バイト 16 進数の 2 文字表記で指定しなければなりません (先行 0 は省略可能)。「length」は制約されています。なぜなら、8 バイトの NEB ヘッダーと ESB ごとに 4 バイトのヘッダーを NEB の最大長である 32 767 バイトに入れる必要があるためです。ヌル値を指定すると、状況域の長さが 10 バイトの標準 ESB が想定されます。共通サブルーチンがタイム・スタンプ付きエラー件数を保持するために使用する場合に、この設定が適しています。

このオペランドはオプションであり、デフォルトは 1 です。そのため、各 NEB はエラー・グループ 1 の ESB 1 つ (状況域の長さは 6 バイト) が設定された状態で生成されます。

### NEBNAME=(name,...)

永続的に割り当てられる NEB が配置されるノードの名前を指定します。指定した名前は、指定した順に、NEBS オペランドで要求した NEB のセットに割り当てられます。残りの NEB は、エラー発生時に他のノードへの動的割り振りに使用できます。名前は、1 から 4 文字のストリングでなければなりません。名前が 1 つである場合は、括弧を省略できます。このオペランドはオプションであり、デフォルトはありません。

### NEBS=10|number

NET に必要な NEB の数を指定します。有効な最大数は 32 767 であり、デフォルトは 10 です。

### TIME=(7,MIN)|(interval,units)

時間間隔を指定します。この値は、NET ヘッダーに保管され、共通サブルーチンがエラー件数を標準 ESB で維持するために使用します。この時間間隔が経過するまでに COUNT オペランドに指定されているしきい値を超過しない場合、エラー件数は 0 にリセットされます。「units」には、SEC、MIN、または HRS を指定します。「interval」の最大値は、それぞれ (86400,SEC)、(1440,MIN)、および (24,HRS) です。このオペランドはオプションであり、デフォルトは (7,MIN) に設定されます。

## ノード・エラー・プログラム DSECT

CICS には、ノード・エラー・プログラム (NEP) で使用するための複数の DSECT が用意されています。

以下の DSECT が用意されています。

### ノード・エラー・テーブル・ヘッダー

これには、テーブル名およびテーブル内のすべてのノード・エラー・ブロック (NEB) に関する一般情報が含まれます。

DFHNETH	DSECT		
NETHNAM	DS	CL8	Table name
NETHNB	DS	H	Number of NEBs in table
NETHNB	DS	H	Length of NEBs in table
NETHTIM	DS	PL8	Error count time interval
NETHECT	DS	H	Error count threshold
NETHFLG	DS	X	Flag byte
NETHINI	EQU	X'01'	Table initialized
	DS	X	Reserved
NETHFNB	DS	0F	First NEB

### ノード・エラー・ブロック

このテーブルには、各ノードのエラー情報を記録するために使用するノード・エラー・ブロックが含まれます。これらは、特定のノードに永続的に割り当てることも、エラー・プロセッサの要求時に動的に割り当てることもできます。

DFHNETB	DSECT		
NEBNAM	DS	CL4	Node name
NEBFLG	DS	X	Flag byte
NEBPERM	EQU	X'01'	Permanently assigned NEB

NEBFESB	DS	XL3	Reserved
	DS	0X	First NEB

## エラー状況ブロック

NEB にはエラー状況ブロックを含めることができます。これらは、特定のエラー・プロセッサのために予約されていて、対応するエラー・グループ・インデックスによって識別されます。ESB の形式は、ユーザーが定義することも、一定の時間間隔でエラーをカウントするのに適した標準形式にすることもできます。

DFHNETE	DSECT		
ESBEGI	DS	X	Error group index
ESBFLG	DS	X	Flag byte
ESBSTAN	EQU	X'01'	Standard format ESB
ESBTTE	EQU	X'02'	Time threshold exceeded
ESBCTE	EQU	X'04'	Count threshold exceeded
ESBSLEN	DS	XL2	Status area length
ESBHLEN	EQU	*-DFHNETE	ESB header length
ESBSTAT	DS	0X	Status area

以下のフィールドが、標準形式に適用されます。

ESBTIM	DS	PL8	Time stamp
ESBEC	DS	XL2	Error count

## 共通サブルーチン・ベクトル・テーブル

CSVТに基づき、エラー・プロセッサは共通サブルーチンのアドレスを指定することができます。エラー・プロセッサのリンク・レジスターに基づき、CSVТ のアドレスを指定することができます。そのため、この DSECT の最初のセクションは、実際のテーブルで分岐を生成するために必要なコードになっています。

DFHNEPC	DSECT		
	DS	F	Load instruction
	DS	F	Branch instruction
CSVТNEP	DS	A	Node error program base address
CSVТESBL	DS	A	NEPESBL - ESB locate routine
CSVТNEBD	DS	A	NEPNEBD - NEB delete routine
CSVTECUP	DS	A	NEPECUP - error count update routine

## 独自のノード・エラー・プログラムの作成

CICS でサポートされているどの言語でも、独自のノード・エラー・プログラム (NEP) を作成できます。

CICS 提供の NEP コードは、アセンブラー言語で用意されています。通信域パラメーター・リストは、アセンブラー言語版と C 言語版があります。用意されているソース・ファイルとコピーブックとマクロの名前や、そのようなリソースが入っているライブラリーについては、[129 ページの表 11](#) を参照してください。

表 11. 用意されているソース・ファイル、コピーブック、マクロ			
名前	タイプ	説明	ライブラリー
DFHZNEP0	プログラ ムが	デフォルト・ノード・エラー・プログラム (アセンブラー言語)	CICSTS56.CICS.SDFHSAMP
DFHZNEPX	ソース	デフォルト NEP (DFHZNEP0 が COPY ステートメントによって埋め込む NEP)	CICSTS56.CICS.SDFHSAMP
DFHSNEP	マクロ	サンプル NEP プログラム生成プログラム (アセンブラー言語)	CICSTS56.CICS.SDFHMAC
DFHZNEPI	マクロ	NEP インターフェース生成プログラム (複数の NEP の場合)	CICSTS56.CICS.SDFHMAC

表 11. 用意されているソース・ファイル、コピーブック、マクロ (続き)			
名前	タイプ	説明	ライブラリー
DFHNEPCA	マクロ	アセンブラー言語通信域	CICSTS56.CICS.SDFHMAC
DFHNEPCA	コピーブック	C 言語通信域	CICSTS56.CICS.SDFHC370

アセンブラー言語でコーディングする場合は、独自のノード・エラー・プログラムを作成するための枠組みとしてサンプル NEP を使用できます。

### EXEC CICS コマンドの使用に関する制限

ノード・エラー・プログラム (NEP) が実行できるコマンドには制限があります。特に、基本機能を必要とするコマンドは使用しないでください。そのようなコマンドを使用した場合の結果は予測不能です。

以下の機能を開始するコマンドは、使用しないでください。

- 端末管理。例えば、**EXEC CICS DELAY** コマンドを実行すると、CSNE タスクが一時停止したまま再開しなくなることがあります。そうすると、領域のシャットダウンがハングする可能性があります。ただし、CEMT タイプのコマンド (**EXEC CICS INQUIRE TERMINAL** など) は使用できます。
- BMS (ルーティング以外)。
- ISC 通信 (機能シップも含む)。リモート・トランザクションの **START** 要求もそれに当たります。そのような要求はお勧めできません。ALLOCATE コマンドをリモート・システムに対して実行している間に、CSNE (ノード異常条件タスク) が一時停止する可能性があるからです。

リモート・トランザクションを開始する場合は、まずローカル・トランザクションを開始して、ローカル・トランザクションからリモート・トランザクションを開始するようにしてください。

- リカバリー可能リソースの更新。そのようなリソースが別のタスクによってロックされていると、CSNE 作業単位が一時停止したり中断したりすることがあります。

NEP を使用して DFHZNAC メッセージを抑止することはできません。

### 入り口およびアドレッシング機能

入り口で、NEP は次のコマンドを発行する必要があります。

```
EXEC CICS ADDRESS COMMAREA
EXEC CICS ADDRESS EIB
```

これらのコマンドにより、DFHZNAC から渡された通信域に対するアドレッシング機能、EXEC インターフェース・ブロックに対するアドレッシング機能をそれぞれ取得できます。

ノード・エラー・プログラムをアセンブラー言語で作成する場合は、以下をコーディングして通信域 DSECT を生成します。

```
DFHNEPCA TYPE=DSECT
```

プログラムを C 言語で作成する場合は、以下をコーディングして通信域の定義を含めます。

```
#include <dfhnepca>
```

## 3270 「unavailable printer」状態に対応するためのコーディング

### このタスクについて

3270 印刷要求機能を使用して印刷要求を送信した時に、制御装置にプリンターがなかったり、プリンターが以下のいずれかの状態になっていたりすると、「unavailable printer」状態になります。

- サービス休止状態になっている場合
- 自動トランザクション開始で TRANSCEIVE や RECEIVE 以外の状況になっている場合



- タスクに接続されている場合
- 前の操作でビジー状態になっている場合
- 介入が必要な場合

CEDA DEFINE TERMINAL コマンドで PRINTER オペランドや ALTPRINTER オペランドを使用する場合は、以下の手順が 3270 論理装置や 3270 互換モード論理装置に当てはまります。

端末管理プログラムは、この状態を検出すると、READ BUFFER 要求を実行して、データを端末入出力域に収集します。端末入出力域 (TIOA) は、アプリケーション・プログラムが端末管理バッファ読み取り要求を実行した場合と同じ形式になります。

その後、端末管理プログラムの z/OS Communications Server セクション (DFHZCP) が TCTTE をノード異常条件プログラムのキューに入れて、エラー・コード X'42' (TCZCUNPRT) を生成します。ノード異常条件プログラム (DFHZNAC) は、CSNE 一時データ・キューに以下の内容を書き込みます。

- DFHZC2497 UNAVAILABLE PRINTER (装置タイプ 3270P と LUTYPE3)
- DFHZC3493 INVALID DEVICE TYPE FOR A PRINT REQUEST (他のすべてのプリンター・タイプ)

DFHZNAC は、ノード・エラー・プログラムにリンクする前に、1 次プリンターと 2 次プリンターのネット名と端末 ID を通信域に挿入し、いずれかのプリンターが印刷要求に対応できるかどうかを通知します。DFHZNAC は、デフォルト・アクションが設定されていない状態でノード・エラー・プログラムにリンクします。

ノード・エラー・プログラムから制御が戻されると、DFHZNAC は、通信域で追加のシステム・パラメーター TWAUPRRRC をチェックし ([115 ページの図 28](#) を参照)、その内容に基づいて以下のいずれかのアクションを実行します。

- NEP が TWAUPRRRC を X'FF' (-1) に設定していた場合、DFHZNAC は、ノード・エラー・プログラムがすでに印刷対象データを処理したと想定し、何もアクションを実行しません。
- NEP が TWAUPRRRC をゼロに設定していた場合、DFHZNAC は、プリンターを使用できないと想定し、何もアクションを実行しません。
- NEP が TWAUPRRRC をゼロにも -1 にも設定していなかった場合、DFHZNAC は、TWAPNETN フィールドと TWAPNTID フィールドのいずれかが設定されていると想定して処理を進めます。(両方とも設定されている場合は、TWAPNTID(terminid) を優先します。) 指定の端末に対してインターバル制御機能の PUT を実行します。開始するトランザクションが CSPP (印刷プログラム) の場合は、時間間隔がゼロになります。
  - インターバル制御機能の PUT でエラーが発生すると、DFHZNAC は、DFHZC2496 IC FAILURE メッセージを宛先の CSNE に書き込みます。その後、DFHZNAC は、ノード・エラー・プログラムに再びリンクして、TWAUPRRRC フィールドを -2 に設定します。ノード・エラー・プログラムに、データを処理する最後の機会を与えるためです。ノード・エラー・プログラムから 2 度目に制御が戻されると、DFHZNAC は TWAUPRRRC を再びチェックします。TWAUPRRRC が -1 であれば、ノード・エラー・プログラムがすでにデータを処理したという意味になります。
  - インターバル制御機能の PUT でエラーが発生しなければ、DFHZNAC は、プリンターが以下の状態になっているかどうかを確認します。
    - サービス休止状態
    - 「要介入」
    - RECEIVE 状況や TRANSCEIVE 状況以外の状態

そのいずれかの状態になっている場合、DFHZNAC は、DFH2495 PRINTER OUTSERV/IR/INELIGIBLE-REQ QUEUED メッセージを宛先の CSNE に送信します。

最後に DFHZNAC は、要求元の端末で印刷要求を終了し、その端末でアクション・フラグに基づく通常の処理を実行します。

## セッション障害のためのコーディング

論理装置またはバス障害に関連するいくつかのカテゴリのエラーが発生した後に、CICS と論理装置の間のセッションが失われる場合があります。DFHZNAC によって実行されるデフォルトのアクションにより、TCTTE のサービスが休止する可能性があります。

### このタスクについて

セッションを自動的に再獲得する方法として、ノード・エラー・プログラムを使用して、デフォルトの DFHZNAC アクションを変更し、TCTTE をサービス状態に維持する方法があります。そして、ノード・エラー・プログラムから、CICS の「good morning」サインオン・メッセージ (CSGM) と同じように作成されたトランザクションに対する EXEC CICS START TERMID(name) コマンドをその TCTTE を使用して発行します。トランザクションが自動トランザクション開始 (ATI) により開始された場合、CICS はセッションの再獲得を試行します。セッションが再び失敗した場合は、DFHZNAC が再度呼び出され、このプロセスが繰り返されます。

EXEC CICS START コマンドで指定する時間は、インストール環境に依存し、予期される平均値によって決定します。

このように使用すると、開始されたトランザクションは、セッションが取得されたときに、適切なサインオン・メッセージを書き込むことができます。ただし、CEDA DEFINE TYPETERM コマンドで LOGONMSG=YES を指定した場合は、セッションが開かれるときに、CICS の「good morning」メッセージも開始されることに注意してください。[130 ページの『EXEC CICS コマンドの使用に関する制限』](#)を参照してください。

## 特定の SNA センス・コードのコーディング

### このタスクについて

[132 ページの図 32](#) は、NEP エラー・プロセッサで特定の SNA センス・コードの存在を検査する方法を示しています。

```
TEST1  EQU  *
        CLC  TWASENSR(2),SNS1          SENSE CODE EQUAL TO NNNN
        BNE  TEST2                     NO, THEN NEXT TEST
        NI   TWAOPT1,TWAOAF             PRINT ACTION MESSAGES ONLY
        OI   TWAOPT2,TWAOAS+TWAOAR+TWAOT ABANDON SEND,RECEIVE AND TASK
        NI   TWAOPT2,255-TWAOASM        SET SIMLOGON OFF
        OI   TWAOPT3,TWAOINT            SET INTLOG NOW ALLOWED
        NI   TWAOPT3,255-TWAOINT        OR RESET NOINTLOG
        B    END
        .
        .
        .
SNS1    DC   X'NNNN'
```

図 32. ノード・エラー・プログラムで特定の SNA センス・コードを検査する方法を示すサンプル・コード

### 複数の NEP の作成

[112 ページの『複数の NEP』](#)で説明しているように、複数のノード・エラー・プログラムを作成することができます。エラーが発生すると、ノード異常条件プログラムがインターフェース・モジュール DFHZNEPI に制御権を渡し、このインターフェース・モジュールがトランザクション・クラスを判別して、適切なノード・エラー・プログラムに制御権を渡します。

ノード・エラー・プログラムを 1 つしか使用しない場合、インターフェース・モジュール (DFHZNEPI) は不要です。ノード・エラー・プログラムの名前が DFHZNEP の場合、ノード異常条件プログラムはそれに直接分岐します。複数のノード・エラー・プログラムを使用する場合は、インターフェース・モジュール (DFHZNEPI) が必要です。その場合、ノード・エラー・プログラムの名前は DFHZNEP 以外にする必要があります。生成したすべてのノード・エラー・プログラムのプログラム定義をインストールしておく必要があります。

### DFHZNEPI マクロ

ノード・エラー・プログラム・インターフェース・モジュール (DFHZNEPI) を生成するには、以下のマクロが必要です。

- DFHZONEPI TYPE=INITIAL: デフォルト・トランザクション・クラス・ルーチンの名前を指定します。
- DFHZONEPI TYPE=ENTRY: トランザクション・クラスにトランザクション・クラス・エラー処理ルーチンを関連付けます。
- DFHZONEPI TYPE=FINAL: DFHZONEPI 項目を終了します。

ノード異常条件プログラムが対象のユーザー作成ノード・エラー・プログラムに制御を渡してエラーを解決する必要がある場合は、DFHZONEPI インターフェース・モジュールを生成しなければなりません。

#### DFHZONEPI TYPE=INITIAL: デフォルト・ルーチンの指定

DFHZONEPI TYPE=INITIAL マクロでは、DFHZONEPI モジュールで使用するデフォルト・トランザクション・クラス・ルーチンの名前を指定します。

#### 構文

```
DFHZONEPI TYPE=INITIAL
           [,DEFAULT=name]
```

#### DEFAULT=name

使用するデフォルト・トランザクション・クラス・ルーチンの名前を指定します。CEDA DEFINE PROFILE、CEDA DEFINE SESSIONS、CEDA DEFINE TYPETERM のいずれかのコマンドを使用して、トランザクションの NEPCCLASS 値として 0 (デフォルト) を指定するか、255 より大きい値を指定した場合や、NEPCCLASS オペランドで指定したクラスで DFHZONEPI TYPE=ENTRY マクロを使用してトランザクション・クラス・ルーチンを指定しなかった場合は、そのデフォルト・ルーチンへのリンクが確立されます。

上記のいずれかが当てはまる状態で DEFAULT オペランドをコーディングしなかった場合は、ルーチンが呼び出されません。

DFHZONEPI TYPE=INITIAL マクロは、どんな場合でも指定する必要があります。他の形式の DFHZONEPI マクロの前に配置することも必要です。指定できる TYPE=INITIAL マクロは 1 つだけです。

#### DFHZONEPI TYPE=ENTRY - トランザクション・クラス・ルーチンの指定

DFHZONEPI TYPE=ENTRY マクロを使用して、トランザクション・クラス (NEPCCLASS) をトランザクション・クラス・エラー処理ルーチンに関連付けます。

このマクロの形式は以下のとおりです。

```
DFHZONEPI TYPE=ENTRY
           ,NEPCLAS=integer
           ,NEPNAME=name
```

#### NEPCLAS=integer

トランザクション・クラスを指定します。1 から 255 までの範囲内でなければなりません。前の DFHZONEPI TYPE=ENTRY の命令で指定した値は、指定しないでください。

#### NEPNAME=name

指定したトランザクション・クラスに関連付けるトランザクション・クラス・ルーチンの名前を指定します。DFHZONEPI を名前として指定した場合、また、8 文字より長い名前を指定した場合は、エラー状態になります。

注: インターフェース・モジュール DFHZONEPI のトランザクション・クラス・ルーチンとして、サンプルのノード・エラー・プログラム (DFHZONEPI 以外の名前のもの) を使用できます。

#### DFHZONEPI TYPE=FINAL - DFHZONEPI エントリーの終了

```
DFHZONEPI TYPE=FINAL
```

#### TYPE=FINAL

モジュール DFHZONEPI の定義を完成させたら、最後に指定する必要があります。アセンブリーは、入り口名が指定されていない END ステートメントによって、またはステートメント END DFHZONEPIA によって終了する必要があります。

## ノード・エラー・プログラムでのシャットダウン・ハング状態の端末の処理

エラー・コード X'6F' の場合、DFHZNAC は **TCSACTN** システム 初期設定パラメーターおよび DFHZC2351 理由コードを DFHZNEP に渡します。DFHZNEP は、現行の端末に対する強制クローズ・アクションを変更できます。

### エラー・コード

X'6F'

### シンボル名

TCZSDAS

### メッセージ番号

DFHZC2351

TCSACTN の設定は、TWAOSCN を設定することで DFHZNEP に渡されます。

- TWAOSCN オフ (B'0') は、TCSACTN=NONE を示します。
- TWAOSCN オン (B'1') は、TCSACTN=UNBIND を示します。

DFHZC2351 理由コードは、NEP 通信域 (NEPCA) フィールド TWATRSN を介して DFHZNEP に渡されます。TWATRSN は 1 バイトのコード・フィールドです。現在、TWATRSN は TWAREASN (同じく 1 バイトのフィールド) をオーバーレイしています。各コードとそれらの意味は、以下のとおりです。

### 01

要求が進行中

### 02

タスクはまだアクティブです

### 03

SHUTC を待機中

### 04

BIS を待機中

### 05

UNBIND を待機中

### 06

RTR を待機中

### 07

BID が進行中

### 08

他の TC 処理を保留中

### 99

(X'63') 不明

詳細については、端末管理メッセージ DFHZC2351 を参照してください。

DFHZNEP は、TWAOSCN を以下のように設定して、現行の端末に対する強制クローズ・アクションを変更できます。

- DFHZNEP が TWAOSCN をオフ (B'0') に設定した場合、DFHZNAC は端末の強制クローズを試行しません (TCSACTN=NONE)。
- DFHZNEP が TWAOSCN をオン (B'1') に設定した場合、DFHZNAC は端末の強制クローズを試行します (TCSACTN=UNBIND)。内部的には、DFHZNAC は、TWAOSCN の正常クローズを TWAOCN の強制クローズに変換してこれを実行します。

DFHZNEP は、TCSWAIT または TCSACTN システム 初期設定パラメーターを変更できません (TCSWAIT システム 初期設定パラメーターおよび TCSACTN システム 初期設定パラメーターを参照してください)。

## 持続セッションでのノード・エラー・プログラムの使用

このセクションでは、持続セッション環境で NEP を使用する場合に役立つ情報を取り上げます。

## 持続セッションがサポートされる場合のノード・エラー・プログラム

持続セッション・サポートについては、「[CICS Recovery and Restart Guide](#)」で説明されています。

会話再開プロセスの中で行われるステップの 1 つは、エラー・コード X'FD' でノード・エラー・プログラムにリンクすることです。システム全体のリカバリー通知に関するオプション (端末ユーザーにリカバリーを通知するかどうか、リカバリー・メッセージ、リカバリー・トランザクション) を、いくつかの端末で変更できるようにするには、コード X'FD' を処理する独自のエラー・プロセッサを作成する必要があります。

持続セッションを使用する場合は、次の点に注意してください。

- 持続セッションを使用する場合は、セッションがリカバリーされたときにコード X'48' が渡されないため、セッションがリカバリーされたときに、通常の「セッション開始」(コード X'48') 処理に相当する NEP 処理を実行することをお勧めします。
- 障害が発生した後もセッションが持続しているのに、再始動時にバインドされなかった場合 (CICS の障害の後に COLD スタートが実行された場合など)、NEP は起動されません (持続セッションをサポートしないシステムでは、z/OS Communications Server のセッションが終了したときに、必ず NEP がコード X'49' (セッションが終了しました) で起動されます)。次のメッセージが発行される状態では、NEP は起動されません。メッセージはシステム・コンソールに表示されます。

DFHZC0120	DFHZC0124
DFHZC0121	DFHZC0129
DFHZC0122	DFHZC0130
DFHZC0123	

メッセージ DFHZC0125 および DFHZC0131 が発行される状態では、それぞれコード X'FB' および X'FC' で NEP が起動されます。これらの状態では、通常の「セッションが終了しました」の NEP 処理に相当する NEP 処理を実行することをお勧めします。

- AIRDELAY システム 初期設定パラメーターをゼロに指定した場合、自動インストールされた端末は再始動後にリカバリーされません。同様に、自動インストールされた端末が再始動後に使用される前に、AIRDELAY で指定した遅延期間が経過した場合、その端末定義は削除されます。このような状況では、CLSDSTP=NOTIFY をコーディングした結果として予期される NEP 処理は実行されません。

### リカバリー通知の変更

端末に対するリカバリー通知の方法は、TYPETERM 定義の RECOVNOTIFY オプションを使用して定義されます。これについては、[TYPETERM 属性](#)で説明しています。CICS はテークオーバーの早い段階で通知手順を開始するため、これが、ネットワーク全体のリカバリー通知方法を最も効率的に指定する方法です。

ノード・エラー・プログラムを使用して、切り替えられたいくつかの端末に対するリカバリー通知方法を変更することができます。例えば、特定のタイプのほとんどの端末については、テークオーバー時にリカバリー・メッセージを受信させたいが、残りについてはサービスの復元完了を通知しないようにしたいという場合があります。そのためには、TYPETERM 定義で RECOVNOTIFY(MESSAGE) をコーディングしておき、ノード・エラー・プログラムを使用して、比較的少ない数の通知しない端末に対してリカバリー通知を NONE に変更します。

### リカバリー・メッセージの変更

TYPETERM 定義で RECOVNOTIFY(MESSAGE) を指定して端末を定義した場合は、テークオーバー後にその端末にリカバリー・メッセージが送信されます。

デフォルトでは、XRF テークオーバーのメッセージは、マップ・セット DFHXMSG の BMS マップ DFHXRC1 に定義されている以下の CICS 提供のメッセージになります。

```
CICS has recovered after a system failure.  
Execute recovery procedures.
```

持続セッションのリカバリーでは、BMS マップ DFHXRC3 が使用されます。このマップでは、上記のメッセージの前に CICS のメッセージ番号 DFHZC0199 が付きます。切り替えられたいくつかの端末に対するリカバリー・メッセージを変更したい場合は、ノード・エラー・プログラムで独自のマップ・セットを指定できます。例えば、サインオン・セキュリティを適用していて、端末ユーザーに再度サインオンするように伝えたい場合などに便利です。指定するマップ・セットには、インストールされたプログラム定義



が必要です。すべての端末のリカバリー・メッセージを変更する場合は、CICS 提供のマップをユーザー独自のマップに置き換えるほうが効率的です。

## リカバリー・トランザクションの変更

テークオーバー後に端末で開始されるリカバリー・トランザクションは、RMTRAN システム初期設定パラメーターを使用して指定します。これが、ネットワークに対してリカバリー・トランザクションを最も効率的に指定する方法です。切り替えられたいくつかの端末に対して、通信域のフィールド TWAXTRAN を上書きして別のトランザクションを指定することができます。指定するトランザクションには、インストールされたトランザクション定義が必要です。また、オプション ATI(YES) を指定して端末を定義しておく必要もあります。

TWAXTRAN で指定したトランザクションが存在しない場合、CICS は CSGM トランザクションの開始を試行します。これも失敗した場合、CICS はセッションを終了します。

## z/OS Communications Server 総称リソースでノード・エラー・プログラムを使用する

(アプリケーション・プログラムまたは CECI から) **EXEC CICS ISSUE PASS** コマンドを使用することにより、CICS から LU を切断し、LUNAME オプションで指定された z/OS Communications Server アプリケーションにその LU を転送することができます。例えば、この CICS から別の LU 専有領域に LU を転送する場合は、次のコマンドを発行します。

```
CECI ISSUE PASS LUNAME(applid)
```

ここで、applid は、LU の転送先の TOR の APPLID です。

TOR が z/OS Communications Server 総称リソース・グループのメンバーである場合は、総称リソース名として LUNAME を指定することにより、グループのいずれかのメンバーに LU を転送できます。例を以下に示します。

```
CECI ISSUE PASS LUNAME(grname)
```

ここで、grname は総称リソース名です。z/OS Communications Server は、LU の転送先として最適なグループ・メンバーを選択します。CICS 総称リソース・グループ内の特定の TOR に LU を転送する必要がある場合は、メンバー名として LUNAME (つまり、最初の例のように CICS APPLID) を指定する必要があります。

ISSUE PASS LUNAME(grname) コマンドを発行するシステムが、その総称リソース名のもとに現在登録されている唯一の CICS である場合 (例えば、他の領域がすべてシャットダウンされている場合) でも、その ISSUE PASS コマンドが INVREQ で失敗することはありません。その代わりに、LU がログオフされ、メッセージ DFHZC3490 が CSNE ログに書き込まれます。

メッセージ DFHZC3490 (DFHZNAC エラー・コード X'C3') が出された状況に対処するためのノード・エラー・プログラムをコーディングすることもできます。

## LU の自動インストールを制御するプログラムの作成

ローカル接続の SNA LU (APPC 単一セッション装置も含む) の自動インストールを制御するプログラムを作成できます。

BIND 要求によって開始するローカル APPC 接続の自動インストールの制御については、[162 ページの『APPC 接続の自動インストールを制御するプログラムの作成』](#)を参照してください。シップされた LU と接続のインストールの制御について詳しくは、[174 ページの『シップされた端末の自動インストールを制御するプログラムの作成』](#)を参照してください。CICS クライアント製品と 3270 ブリッジで使用する仮想 LU のインストールの制御については、[180 ページの『仮想端末の自動インストールを制御するプログラムの作成』](#)を参照してください。

## 端末の自動インストール

**DEFINE TERMINAL** コマンドおよび **DEFINE TYPETERM** コマンドを使用して、z/OS Communications Server デバイスを CICS に定義することができます。これらのコマンドは、CICS システム定義 (CSD) ファイルにリソース定義を追加します。定義情報に、それらの定義を自動インストールのモデルとして使用可能であることを指定できます。



端末制御に使用するモデル・リソース定義を定義およびインストールすることにより、不明なデバイスが CICS への接続を要求してきたときに、CICS が端末制御テーブル (TCT) の必須項目を自動的に作成できるようになります。自動インストールの利点の 1 つは、リソースが TCT 内のストレージを占有するのは、リソースが CICS に接続している間、最後に使用した後の指定遅延期間のみである、という点です。

自動インストール制御プログラムを使用して、端末を自動インストールするために必要ないくつかのデータ (特に CICS 端末名と各インスタンス内で使用するモデル名) を選択します。CICS 提供の自動インストール・プログラムを使用するか、またはそのプログラムを独自に拡張することができます。

自動インストールの概要については、[自動インストール](#)を参照してください。また、同じマニュアル内の、制御プログラムが動作可能な環境を作成する CEDA コマンドについて説明しているセクションも参照してください。

ご使用の端末の一部またはすべてにおいて自動インストールを選択する場合は、以下の作業を行う必要があります。

1. モデルの TERMINAL 定義を作成する。
2. z/OS Communications Server に端末を定義する。このようにして、z/OS Communications Server における端末の定義が CICS のモデルの TERMINAL 定義と一致するようにします。
3. モデル端末サポート (MTS) を使用する場合、MTS テーブルを z/OS Communications Server に定義する。
4. デフォルトの端末用自動インストール制御プログラム (DFHZATDX) を使用するか、またはデフォルト・プログラムのソース・コードと関連トピックにあるカスタマイズ例を参考にして独自のプログラムを作成する。(デフォルト・プログラムでは要件に合わない場合、まったく新しいプログラムを作成することができますが、まずはデフォルト・プログラムをベースにしたプログラムを試してみることをお勧めします。) プログラムの作成には CICS でサポートされているどの言語でも使用できます。デフォルト・プログラムのソースは、アセンブラ言語、COBOL、PL/I、および C 言語で提供されています。ユーザー作成のプログラムの名前を変更することができます。

**注:**

- a. 自動インストール制御プログラム (または提供されている DFHZCTDX) は Language Environment 対応のコンパイラを使用してコンパイルし、Language Environment 対応でプログラムを実行する必要があります。
- b. アクティブ自動インストール制御プログラムは 1 つだけ使用することができ、それで端末と APPC 接続の両方を処理することが可能です。アクティブ・プログラムの名前を **AIEXIT** システム初期設定パラメーターに指定します。[162 ページの『APPC 接続の自動インストールを制御するプログラムの作成』](#)で説明されている DFHZATDY プログラムには、DFHZATDX と同じ端末自動インストール機能に加えて、BIND 要求が開始した APPC 接続の自動インストール機能もあります。DFHZATDX と DFHZATDY はともに、提供されている端末と接続をインストールする機能を備えています。したがって、例えば、APPC 接続に加えて SNA LU も自動インストールするには、DFHZATDX ではなく DFHZATDY をカスタマイズしたバージョンを使用する必要があります。

## **z/OS Communications Server LOGON モード・テーブルのエントリーのコーディング**

CICS は、自動インストール要求を処理する際に、z/OS Communications Server LOGON モード・テーブルのログモード・データを使用します。自動インストールは、z/OS Communications Server に定義したログモード・エントリーが、CICS に指定したモデル TERMINAL 定義と一致する場合にのみ、正常に機能します。

[Coding entries in the VTAM LOGON mode table](#) の表に、さまざまな LU デバイスについて、自動インストールする LU をログモード・テーブルに定義する z/OS Communications Server MODEENT マクロで何をコーディングする必要があるかを示しています。それらの表の間に、MODEENT マクロの各オペランドで指定する必要がある値についての表も示しています。

付録の例のいくつかは、ISTINCLM という IBM 提供のログオン・モード・テーブルのエントリーに正確に対応しています。これに該当する場合、表では ISTINCLM でのエントリーの名前を記載しています。

## モデル端末サポート (MTS) の使用

MTS を使用して、SNA テーブルの各 LU ごとに、モデル名、プリンター (PRINTER)、および代替プリンター (ALTPRINTER) を定義できます。

この情報は、拡張 CINIT RU の z/OS Communications Server によって送信されます。CICS は、ログオン時の自動インストール処理の一部としてそれを取り込み、それを使用して LU 用の TCTTE を作成します。

## MTS のエントリーのコーディング

モデル名 (MDLTAB、MDLENT、および MDLPLU マクロ)、プリンター、および関連付けられたプリンター名 (ASLTAB、ASLENT、および ASLPLU マクロ) を z/OS Communications Server に定義する必要があります。

## 端末の自動インストール制御プログラム

リソース定義の管理に加えて、自動インストール制御プログラムは、この時点で必要なその他のプロセスも実行できます。コマンド・レベル・インターフェースへのアクセスは、通常の非端末ユーザー・タスクのものであります。

利用できる使用法のいくつかは、[150 ページの『端末用のサンプルの自動インストール制御プログラム』](#)にリストされています。

この制御プログラムは、以下の場合に呼び出されます。

1. 自動インストールの INSTALL 要求が処理されるとき
2. 自動インストール DELETE 要求がちょうど完了したとき
3. 自動インストール要求がユーザー・プログラムで既に受け入れられているものの、それに続く INSTALL プロセスが失敗したとき。

自動インストール制御プログラムが呼び出されるたびに、(通信域を使用して) パラメーター・リストが渡されます。これにより、実行される機能 (INSTALL または DELETE) を示し、特定のイベントに関連したデータを提供します。(ケース 3 では、この制御プログラムは DELETE の場合のように呼び出されます)。

INSTALL および DELETE イベントが詳細に記述されます。

## INSTALL 時の自動インストール制御プログラム

自動インストールが有効になっていると、以下の INSTALL 時に自動インストール制御プログラムが呼び出されます。

- ローカル SNA LU
- MVS コンソール
- CINIT によって開始されたローカル APPC 単一セッション接続
- BIND によって開始されたローカル APPC 並列セッション接続
- BIND によって開始されたローカル APPC 単一セッション接続
- クライアント 仮想端末
- シップされた端末と接続

CICS は、呼び出しごとに、DFHEICAP によってアドレッシングされる通信域を経由して制御プログラムにパラメーター・リストを渡します。MVS コンソールの INSTALL 時に渡されるパラメーター・リストについては、[157 ページの『INSTALL 時の自動インストール制御プログラム』](#)を参照してください。BIND 要求によって開始するローカル APPC 接続の INSTALL 時に渡されるパラメーター・リストについては、[164 ページの『APPC 接続の INSTALL 時の通信域』](#)を参照してください。シップされた端末と接続の INSTALL 時に渡されるパラメーター・リストについては、[177 ページの『シップされた端末の INSTALL 時の通信域』](#)を参照してください。クライアント 仮想端末の INSTALL 時に渡されるパラメーター・リストについては、[183 ページの『クライアント 仮想端末の INSTALL 時の通信域』](#)を参照してください。MVS コンソールの INSTALL 時に渡されるパラメーター・リストについては、[156 ページの『コンソールの自動インストールを制御するプログラムの作成』](#)を参照してください。このセクションでは、ローカル端末 (CINIT によって開始される APPC 単一セッション接続も含む) の INSTALL についてのみ取り上げます。

端末の INSTALL 時に制御プログラムが呼び出されるのは、以下の両方に該当する場合です。

- 自動インストール可能で NETNAME が TCT に存在しないリソースから z/OS Communications Server for SNA のログオン要求を受け取った場合。
- 制御プログラムからの情報 (端末 ID と自動インストール・モデル名) がないと処理を続けられない時点まで自動インストール処理が完了した場合。

### INSTALL 実行時の端末の通信域

通信域のレイアウトを 139 ページの図 33 に示します。

Fullword 1	Standard Header	
Byte 1	Function Code	(X'F0' for INSTALL)
Bytes 2 - 3	Component Code	Always 'ZC'
Byte 4	Reserved	Always X'00'
Fullword 2	Pointer to NETNAME_FIELD	
Fullword 3	Pointer to MODELNAME_LIST	
Fullword 4	Pointer to SELECTED_PARMS	
Fullword 5	Pointer to CINIT_RU	

図 33. INSTALL 実行時の自動インストール制御プログラムの通信域

パラメーター・リストには、次の情報が含まれています。

1. 標準ヘッダー。バイト 1 は要求タイプを示します (これは INSTALL を表す 16 進数文字 X'F0' です)。
2. 2 バイト長フィールドを指すポインター。これに LOGON を要求しているリソースの NETNAME が続きます。
3. 適格な自動インストール・モデルの名前の配列を指すポインター。配列の前には、配列に含まれる 8 バイトの名前要素の数を記述する 2 バイトのフィールドが配置されます。配列に要素が何もない場合、数値フィールドはゼロに設定されます。
4. 情報を CICS に返すときに使用する記憶域を指すポインター。z/OS Communications Server の CINIT からの MTS 情報はその領域に格納します。
5. z/OS Communications Server の LOGON データ (CINIT 要求単位) を指すポインター。このデータには 3 文字の NS ヘッダーが含まれ、このデータのの前には CINIT 要求単位の長さを示す 2 バイト長のフィールドが置かれます。

CICS は、パラメーター・リストのフルワード 3 でアドレッシングされた領域内に、対象となる自動インストール・モデルのリストを渡します。

モデル名が MTS が提供していない場合、制御プログラムはこのリストからログオン中のデバイスに適したモデルを選択し、そのモデル名をパラメーター・リストのフルワード 4 でアドレス指定する領域の最初の 8 バイトに移動する必要があります。

例えば、3270 プリンターが自動インストールを試行する場合、一致するモデルのサブセットは、z/OS Communications Server のカテゴリ 2 にモデルとして定義したすべてのタイプとなります。このサブセットには、以下のモデルが含まれます。

- DEVICE(3270) TERMMODEL(2)
- DEVICE(3270) TERMMODEL(1)
- DEVICE(3270P) TERMMODEL(2)
- DEVICE(3270P) TERMMODEL(1)
- DEVICE(3275) TERMMODEL(2)
- DEVICE(3275) TERMMODEL(1)

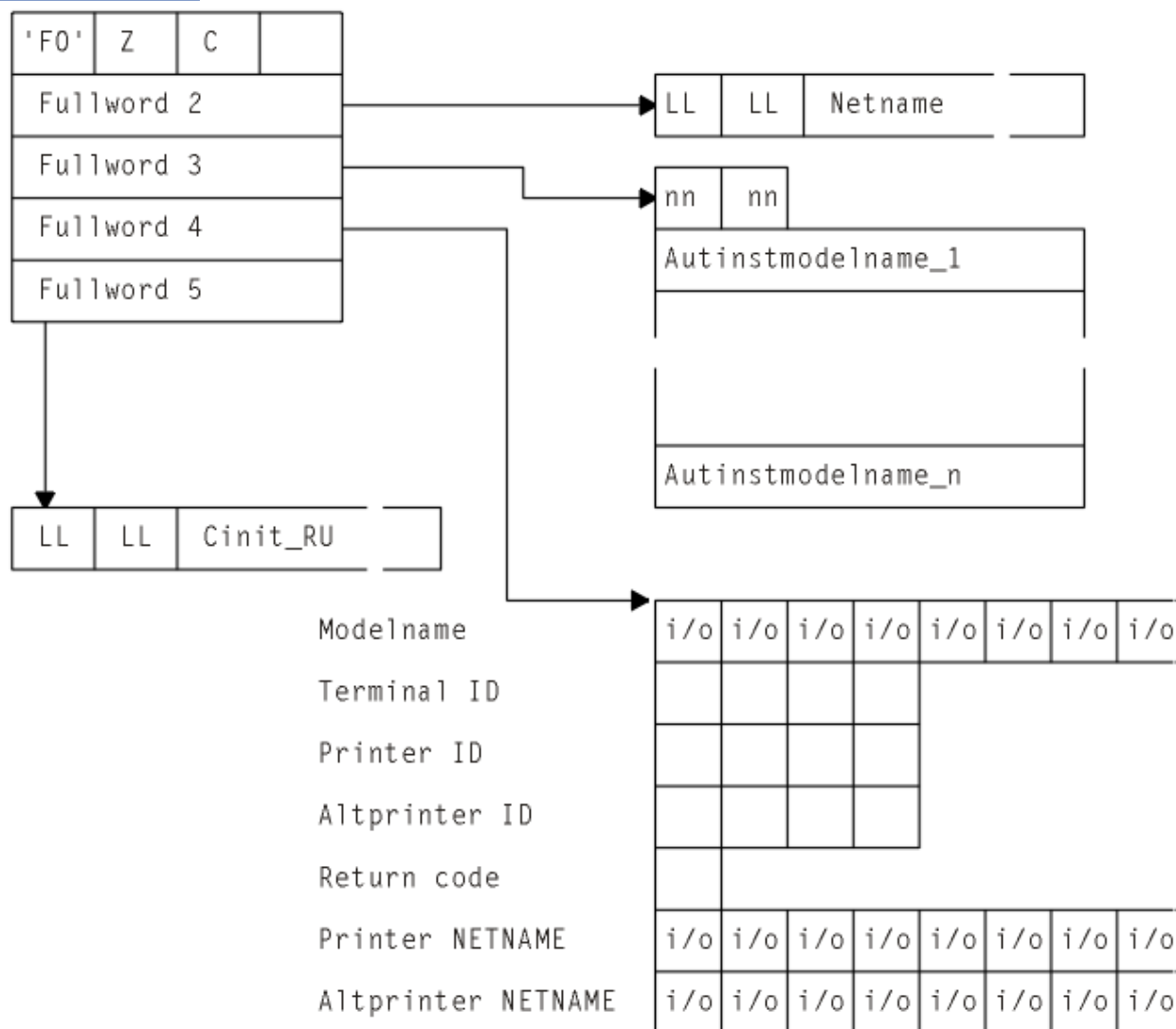
制御プログラムはこのリストから 1 つのモデルを選択し、CICS はそのモデルを使用してデバイスの TCTTE をビルドします。デフォルトの自動インストール制御プログラム DFHZATDX は、常にリストの最初のモデル名を選択します。

MTS を使用していないものの、この端末に関連付けられているプリンター ID や NETNAME (または代替プリンターの ID や NETNAME) が必要である場合、制御プログラムはこの情報をフルワード 4 でアドレス指定されている領域で指定することができます。

MTS を使用している場合は、CICS が z/OS Communications Server の ASLTAB マクロで指定されたプリンターおよび代替プリンターの NETNAME を制御プログラムに渡します。

CICS に戻る前に、制御プログラムはログオン中のデバイスに CICS 端末名を提供する必要があり、自動インストール要求が許可されている場合は戻りコード・フィールドを X'00' に設定する必要があります。

140 ページの図 34 は、これらのフィールドのすべてを、必要とされる順序で示しています。



Note: i/o designates an input/output field.  
The other fields in SELECTED\_PARMS are output only.  
Input may be supplied by MTS from the MTS CINIT.

図 34. INSTALL 実行時の自動インストール制御プログラムのパラメーター・リスト

### CICS による自動インストール・モデルのリストの作成方法

CICS が MTS モデル名を検出すると (かつ、このモデルが CICS に定義されており、リソースを記述する z/OS Communications Server 情報と互換性がある場合)、CICS はモデル名をモデル名リスト (Autinstmodelname\_1) に書き出し、パラメーター・リストのフルワード 4 によってアドレス指定される選択リストのモデル名フィールド (Modelname) にも書き出します。

CICS が MTS 制御ベクトルから MTS モデル名を検出できないか、または指定されたモデルが存在しないか無効である場合、CICS はそのリソースを記述した z/OS Communications Server 情報と互換性のあるモデルを端末モデル全体のリストから選択して、自動インストール・モデルのリストを作成します。CICS が常時使用可能な自動インストール・モデル全体のリストは、CICS 初期スタートまたはコールド・スタート時の GRPLIST と、CEDA が発行した INSTALL GROUP コマンドの両方によってインストールされた、



AUTINSTMODEL(YES) および AUTINSTMODEL(ONLY) が指定されているすべての定義によって構成されます。自動インストール・モデルでは、モデルの定義について説明しています。

TYPETERM デバイス・タイプと関連 LOGON モード・データを指すポインターには、特定の端末がインストールを試行したときに、どのモデル・タイプが自動インストール制御プログラムに渡されるモデルのサブセットに含まれるかを確認するのに役立つ情報があります。このサブセットは、ログオンを試行するデバイスの z/OS Communications Server 特性によって決まります。図の右側の列にある数字は、全体リストからのサブセットの選択内容を示します。特定の組み合わせの DEVICE 値、SESSIONTYPE 値、および TERMMODEL 値を持つ端末がログオンを試行すると、表の右側の列にある z/OS Communications Server カテゴリー番号に対応する DEVICE 値、SESSIONTYPE 値、および TERMMODEL 値を持つすべてのモデルが、制御プログラムに渡される一致するモデルのサブセットに組み込まれます。

BIND が完全に一致するモデルを CICS が検出できず、パラメーター・リストのフルワード 4 がアドレス指定する領域の戻りコードが非ゼロである場合、CICS はエラー・メッセージ DFHZC6987 を発行します。このメッセージには「最大の障害」モデル名が含まれますが、これは診断用にのみ提供されます。これについては、144 ページの『制御プログラムから戻ったときの CICS アクション』で詳しく説明しています。

### CICS への情報の戻し

INSTALL イベント時には、自動インストール制御プログラムが新規端末リソースによる CICS システムへの接続を許可するか拒否するかを決定します。この決定は、セキュリティや接続済み端末の合計数など、インストール済み環境に依存するいくつかの要素に基づいて行われます。この検査に CICS はまったく関与しません。このような検査を行うかどうか、またそれをどのように行うかは、ユーザーが決定します。

INSTALL 要求を続行する場合、制御プログラムは以下を実行する必要があります。

- 自動インストール・モデル名がまだ MTS サポートによって設定されていない場合は、パラメーター・リストのフルワード 4 がアドレス指定する領域の最初の 8 バイトにあるモデル名を返します。

制御プログラムから返されるモデル名が CICS から渡されたサブセットに含まれるモデルではない場合、CICS はそれ以降の処理を行った場合に何が生じるかを保証できません。ストレージ内の「モデル」オブジェクトとのインターフェースは提供されていないため、特定のログオン要求を特定のモデル名に関連付けることによる影響はユーザーの責任において判断する必要があります。

- 戻り領域の次の 4 バイトに CICS 端末名 (TERMID) を指定します。

DFHZATDX は NETNAME (パラメーター・リストのフルワード 2 によってアドレス指定される) の最後の 4 つの非ブランク文字を端末名として使用するため、この名前がご利用のインストール済み環境の命名規則と一致しない場合は、独自の自動インストール・プログラムをコーディングする必要があります。このことについては、142 ページの『TERMINAL 名の設定』を参照してください。

LU6.2 単一セッション端末の AUTOINSTALL 要求を処理するときには、ユーザー・プログラムから返される 4 バイトの端末 ID を使用して CONNECTION に名前が付けられます。そのため、端末 ID は (TERMINAL ではなく) CONNECTION の命名標準 (CONNECTION 属性の定義を参照) に準拠している必要があります。ユーザー・プログラムは、次のいずれかの方法で LU6.2 AUTOINSTALL 要求を識別することができます。

- MODEL 命名規則を使用し、フルワード 3 によって指し示されるモデル名を調べる。
- X'0602' (LU6.2) のフルワード 5 によって指し示されている CINIT BIND のバイト 14 および 15 をテストする。

- 戻りコードを X'00' に設定する。

自動インストール制御プログラムへの入り口では、戻りコードには常にゼロ以外の値が含まれます。これを変更しない場合、自動インストール要求は拒否されます。

MTS を使用している場合、z/OS Communications Server が PRINTER NETNAME と ALTPRINTER NETNAME を提供します (指定されている場合)。

この段階でプリンターをインストールする必要はありませんが、「印刷キー」サポートを使用する前にはインストールする必要があります。PRINTER ID と ALTPRINTER ID は PRINTER NETNAME と ALTPRINTER NETNAME を指定変更します。

TERMID、PRINTER、および ALTPRINTER のみが、自動インストール制御プログラムによって設定可能な TERMINAL 定義の属性です。それ以外のすべての属性は、次のソースのいずれかから提供する必要があります。

- z/OS Communications Server の LOGMODE 項目 (MODEENT)
- 自動インストール・モデルの TERMINAL 定義
- 属性が参照する TYPETERM 定義
- QUERY 関数
- z/OS Communications Server の MDLTAB MDLENT から提供されるモデル名、および z/OS Communications Server の ASLTAB ASLENT から提供されるプリンターの NETNAME (MTS を使用している場合)。

注:

1. QUERY 関数は TYPETERM 定義に指定されている拡張属性を指定変更します。
2. LOGMODE 項目の情報をモデル TERMINAL および TYPETERM で指定変更することはできません。両方の情報が一致している必要があります。

制御プログラムが INSTALL 要求を拒否することを決定した場合、そのプログラムから CICS に戻りコードで非ゼロ値を返す必要があります。

処理が完了したら、制御プログラムは EXEC CICS RETURN コマンドを発行して CICS に戻る必要があります。

### 自動インストール・モデルの選択

モデル端末サポートを使用してモデル名を指定した場合、その名前のモデルが存在していて有効になっていれば、CICS がそのモデル名を自動インストール制御プログラムに渡します。ユーザーが何かの選択をする必要はありません。

一般に、プログラムに渡されるリストに含まれているすべてのモデルが LU の SNA データに対応しています。つまり、有効な TCT 項目は通常、いずれかのモデルの使用によって生成されます。(この規則の例外は、z/OS Communications Server for SNA の RUSIZE です。この値に互換性がないと、CICS がエラー・メッセージを生成します。) デフォルトの自動インストール制御プログラムは、単純にリスト内の先頭のモデルを選択します。しかし、必要な属性が必ずしもそのモデルに用意されているとは限りません。例えば、3270 プリンターに 3270 ディスプレイ 装置定義は不要です。その端末に必要な特性 (セキュリティー特性など) が用意されているモデルを制御プログラムによって選択できる必要があります。

ストレージの節約のために、制御プログラムで使用する異なるモデルの数と、それらのモデルから参照する異なる TYPETERM 定義の数を最小限にしてください。DFHTCT マクロから定義を移行する場合は、それらの定義をよく吟味し、別々にする必要のないモデルを除去してください。QUERY 機能を使用すれば、定義に対応したすべての装置を確認できます。QUERY に対応していない 2 進同期装置の場合は、定義をできるだけシンプルにして特殊な機能を組み込まないようにするのも一案です。

特殊なケースに対応した特殊なモデルが必要な場合は、シンプルなマッピングを使用できます。例えば、NETNAME (汎用または特化) と AUTINSTNAME のマッピングなどです。制御プログラムによって特殊なケースの NETNAME のテーブルをチェックし、個別に指定のモデルを選択できます。テーブルに含まれていない端末では、デフォルト・モデルを使用します。制御プログラムに渡すモデルのリストは、アルファベット順になっています。ただし、1 つだけ例外があります。その例外については、[z/OS Communications Server MODEENT マクロ・オペランド](#)の注記を参照してください。

### TERMINAL 名の設定

TERMINAL 名は、固有の名前でなければなりません。長さは 1 文字から 4 文字の間です。CICS は、端末の ID として TERMINAL 名を使用します。z/OS Communications Server は、端末の ID として NETNAME を使用します。

使用できる文字のリストについては、[TERMINAL 属性](#)を参照してください。

トランザクションが開始元の端末や接続先の端末に依存していて、その端末に TERMINAL 名が設定されている、という状況があります。特定の端末だけに制限されているトランザクションもあれば、端末によって動作の異なるトランザクションもあります。トランザクションが端末の使用状況に関する統計を収集する時に、参照先として TERMINAL 名を使用する、というケースもあります。TERMINAL 名がネットワークの管理や使用や保守の担当者に役立つ場合もあります。例えば、地理的位置や部署や役職に関する情報が組み込まれていることもあるからです。



そのような目的のためには、**TERMINAL** 名より **NETNAME** のほうが適しています。**NETNAME** は長さが 8 文字だからです。**NETNAME** を使用できる場合は、自動インストール制御プログラムによって **TERMINAL** 名をランダムに割り当てることもできます。ユーザーがログオンするたびに端末の **TERMINAL** 名が変わっても問題はありません。その場合に制御プログラムが必要になるのは、端末の自動インストール先のシステムで **TERMINAL** 名を固有の名前にする、という目的のためだけです。制御プログラムが、すでに TCT 項目のある **TERMINAL** 名の TCT 項目をインストールしようとする、たとえその端末がインストールの条件を満たしていて、適切なモデルが見つかった場合でも、インストールは拒否されます。(一方、**NETNAME** にすでに TCT 項目がある場合は、端末がその項目を使用するので、自動インストールが呼び出されることはありません。)

デフォルトの自動インストール制御プログラムは、**NETNAME** の最後の 4 つの非ブランク文字から **TERMINAL** 名を作成します。その結果、固有性の条件を満たせなくなる場合もあります。そのような問題を解決する 1 つの方法は、制御プログラムから **EXEC CICS INQUIRE** コマンドを使用して、その **TERMINAL** 名がすでに使用されているかどうかを確認することです。すでに使用されている場合は、最後の文字を変更して、再度確認します。

一方、端末の名前として固有の予測可能な **TERMINAL** 名を引き続き使用しなければならない状況もあります。その場合は、ユーザーがログオンするたびに、制御プログラムによって正しい **TERMINAL** 名を各端末に割り当てることができる必要があります。この問題を解決するには、2 つの方法が考えられます。

- **NETNAME** から予測可能な **TERMINAL** 名を生成する別のアルゴリズムを作成します。
- テーブルやファイルを使用して **TERMINAL** 名を **NETNAME** に対応付けます。

アルゴリズムを作成すれば、テーブルやファイルを使用する場合の欠点を回避できますが、固有性と予測可能性の両方を確保するのが難しくなる可能性があります。**NETNAME** に含まれている一部の情報が **CICS** にとって不要であれば、**TERMINAL** 名でその情報を省略できます。その場合は、アルゴリズムを作成するほうがよいかもしれません。

**テーブルの使用**には 2 つの欠点があります。つまり、ストレージの使用量が増えることと、保守が必要になることです。どちらの場合も、自動インストールのメリットがいくらか失われてしまいます。テーブルを作成する場合は、メインの一時ストレージで作成し、16MB 境界より上の拡張ストレージに配置する、という方法があります。ストレージの問題を避けるために、テーブルの代わりに **VSAM ファイル**を使用することも可能です。ただしその場合は、ファイルに関連付けられた入出力が原因で処理が遅くなる可能性があります。テーブルやファイルに **PRINTER** や **ALTPRINTER** などの情報を組み込み、特定の自動インストール・モデルの必要な装置については **AUTINSTNAME** などの情報を追加できます。(142 ページの『自動インストール・モデルの選択』を参照してください。)

#### SNA 動的別名に関する考慮事項

**CICS** 領域が動的 **LU** 別名を使用している (つまり、**SNA APPL** 定義において **LUAPFX=xx** が指定されている) 場合、固有の **TERMINAL** 名を選択すると、選択しない場合よりも複雑になる可能性があります。以下の要因を考慮に入れる必要があります。

- デフォルト・プログラムは **NETNAME** の末尾の 4 文字を使用します。そのため、動的 **LU** 別名を割り当てられた **LU** に対して反復可能 **TERMID** を生成することはありません。ログオンを繰り返しても端末 **ID** を同じにできることが重要な場合は、**CINIT** または **BIND** でネットワーク修飾名を使用することを考慮してください。
- **NETNAME** の末尾の 4 文字を使用する場合、動的 **LU** 別名から 0001、0002、というように端末 **ID** が生成されます。**RDO** 定義済み端末にそのような名前がないことを確認し、必要に応じて自動インストール制御プログラムのロジックを変更してください。例えば、**NETID** の末尾の文字を実際のネットワーク名の末尾の 3 文字と連結して使用することができます。
- **DFHZATDX** および **DFHZATDY** に、いくつかの新しいサンプル・コードがあります。これは、**CINIT** または **BIND** から **NQNAME** として参照されるネットワーク修飾名を抽出し、**NETID** の末尾の文字と実際のネットワーク名の末尾の 3 文字を使用して代替の **TERMID** を提供するものです。

なんらかの理由でこのロジックで端末 **ID** を作成できない場合、この方法は失敗して通常通りネットワーク名から端末 **ID** を作成します。このコードはコメントで囲まれており、**CINIT** および **BIND 'OE'** 制御ベクトルから必要な情報を抽出する方法を示す目的でのみ提供されていることに注意してください。

- サンプル・コードも、**DFHZATDX** の **C**、**COBOL**、および **PL/I** バージョンにコメントの形式で追加されています。これらを使用する場合、以下のことに注意してください。

- PL/I サンプル DFHZPTDX は、PL/I コンパイラー・オプション LANGLVL(SPROG) でコンパイルする必要があります。
- COBOL サンプル DFHZCTDX は、コンパイラー・オプション TRUNC(OPT) でコンパイルする必要があります。

### 制御プログラムから戻ったときの CICS アクション

自動インストール制御プログラムから戻るときに、CICS は戻りコードを調べ、その情報を使用して、ログオン要求を完了すべきかどうかを判別します。

戻りコードがゼロであり、他の必須情報が十分に提供されている場合、CICS はログオン要求を完了するために OPNDST で使用する新規リソースをスケジュールします。インストール・プロセスが失敗すると、DELETE が発生した場合と同じように、再び制御プログラムが呼び出されます。(詳細については、[145 ページの『DELETE 時の自動インストール制御プログラム』のセクション](#)を参照。) この INSTALL 要求が成功するという想定で行われたすべての割り振り (端末 ID など) をプログラムが解放できるようにするために、この処理は必要です。

戻りコードがゼロではないとき、CICS は接続要求を拒否します。それは、自動インストールが有効でない場合に、CICS が不明な端末によるログオン試行を拒否するのと同じです。

すべての自動インストール・アクティビティーで、一時データ宛先 CADL にメッセージが書き込まれます。INSTALL が失敗する場合、理由コードと一緒にメッセージが CADL に送信されます。したがって、CADL からの出力を確認することで、自動インストール要求が失敗した理由を突き止めることができます。

完全に一致しないことが原因で自動インストールの試行が失敗する場合は、モデルと BIND イメージの「最大の障害」一致の詳細が CADL 一時データ宛先に書き込まれます。

メッセージは以下のような形式になります。

```
DFHZC6987 BEST FAILURE FOR NETNAME: nnnnnnnn,
        WAS MODEL_NAME: mmmmmmmm,
        CINIT BIND: cccccccc...,
        MODEL BIND: bbbbbbbb...,
        MISMATCH BITS: xxxxxxxx...
```

ここで、

- 'nnnnnnnn' は、ログオンに失敗した LU のネット名です。
- 'mmmmmmmm' は、「最大の障害」となったモデル名です。(つまり、z/OS Communications Server から提供された BIND イメージとビットの相違が最も少ないモデル名です。)
- 'ccccccc...' は、CINIT BIND イメージです。
- 'bbbbbbb...' は、モデル BIND イメージです。
- 'xxxxxxx...' は、16 進数字のストリングです。ここで、'xx' は 1 バイトを表し、各バイト位置は BIND イメージでの対応するバイト位置を表します。ビットが '1' に設定されている場合は、z/OS Communications Server から提供された BIND イメージと、モデルに関連付けられている BIND イメージとの間で位置に不一致があることを示します。

推奨されている一連の作業は、以下のとおりです。

1. 'mmmmmmmm' などのモデルが適当であるかどうかを判別します。同一の BIND イメージを持つ複数のモデルがある場合、エンド・ユーザー・オプションが異なっているだけなら、それらの最初のモデルの名前だけが上記のメッセージに示されます。ログモード・テーブル項目を訂正するときどのモデルを選択するかは、制御プログラムが決定します。
2. 使用中の z/OS Communications Server ログモード・テーブル項目を識別します。
3. このログモード・テーブル項目を使用している他のアプリケーションがないかを確認してください。その項目を問題なく使用している他のアプリケーションがある場合にこの項目を変更すると、そのアプリケーションが失敗する可能性があります。
4. ログモード・テーブル項目の、不一致ストリング内の 1 のビットに対応するビットを切り替えて、修正してください。つまり、上記の 'xxxxxxx...' で「1」に設定されているビット位置に対応する z/OS Communications Server BIND イメージのビットが「1」に設定されている場合は「0」に設定し、「0」である場合は「1」に設定します。

## DELETE 時の自動インストール制御プログラム

自動インストール・プロセスの制御に関する対称性という意味で、以下の場合に自動インストール制御プログラムが呼び出されます。

- 以前に自動インストールされたリソースとのセッションが終了した場合。
- ユーザー・プログラムが自動インストール要求を受け取ったのに、何かの理由でその後の INSTALL プロセスが失敗した場合。

制御プログラムの作成を簡略化するために、この 2 つのイベントを同一と見なしてもかまいません。(存在する環境にも、実行するアクションにも、違いはありません。)

DELETE 時に制御プログラムを呼び出すことによって、INSTALL イベントで実行されたプロセスを元に戻すことができます。例えば、INSTALL 時に制御プログラムで自動インストール・リソースの総数を増やした場合は、DELETE 時に制御プログラムでその数を減らすことができます。

### 端末の DELETE 時の通信域

プログラムに対する入力、DFHEICAP でアドレス指定された通信域に指定されます。

通信域のレイアウトを [145 ページの図 35](#) に示します。

Fullword 1	Standard Header
Byte 1	Function Code (X'F1')
Bytes 2 - 3	Component Code Always 'ZC'
Byte 4	Reserved Always X'00'
Fullword 2	Terminal ID of terminal to be deleted
Fullword 3	NETNAME of terminal to be deleted
Bytes 1-2	Delete netname length
Bytes 3-4	Start of Delete netname ID
Next 15 bytes	Remainder of Delete netname ID

図 35. DELETE 時の自動インストール制御プログラムの通信域

パラメーター・リストには、次の情報が含まれています。

1. 標準ヘッダー。バイト 1 は要求のタイプを示します。ローカル端末 (CINIT 要求によってインストールされた APPC 単一セッション・デバイスを含む) の削除の場合、値は X'F1' です。

注：X'F5' または X'F6' の値は、BIND 要求によってインストールされたローカル APPC 接続の削除を表します ([166 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照)。X'FA' または X'FB' の値は、シップされた端末または接続の削除を表します ([178 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照)。X'FC' の値は、クライアント 仮想端末の削除を表します ([186 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照)。

2. 削除されたリソースの端末 ID ([145 ページの表 12](#) を参照)。

表 12. 表 1. DELETE 時の自動インストール制御プログラムのパラメーター・リスト				
	最初のバイト	2 番目のバイト	3 番目のバイト	4 番目のバイト
最初のフルワード	"F1"	"Z"	"C"	予約済み
2 番目のフルワード	削除対象の端末の ID	削除対象の端末の ID	削除対象の端末の ID	削除対象の端末の ID
3 番目のフルワード	削除対象のネット名の長さ	削除対象のネット名の長さ	ネット名の最初の 2 つのバイト	ネット名の最初の 2 つのバイト
次の 15 バイト	ネット名の残り	ネット名の残り	ネット名の残り	ネット名の残り

制御プログラムが呼び出されるまでに指定されたリソースは削除されているため、TC LOCATE タイプ関数によって検出されないことに注意してください。

## 自動インストール制御プログラムの命名、テスト、およびデバッグ

### 自動インストール制御プログラムの命名

CINIT は、用意されている自動インストール制御プログラム DFHZATDX を開始します。これは、端末および APPC 単一セッション接続用であり、ユーザーが置換可能なプログラムです。独自のバージョンの制御プログラムを作成する場合は、別の名前にすることができます。

システムがロードされた後に、現在 CICS に指定されている自動インストール制御プログラムの名前を調べるには、CICS Explorer の「領域」操作ビュー、**EXEC CICS INQUIRE AUTOINSTALL** コマンド、または **CEMT INQUIRE AUTOINSTALL** コマンドを使用します。

デフォルト名は DFHZATDX です。

現行のプログラムを変更するには、以下のいずれかのオプションを使用します。

- CICS Explorer の「領域」操作ビューの「属性」タブ。
- AIEXIT システム 初期設定パラメーター。AIEXIT の使用法について詳しくは、[AIEXIT システム 初期設定パラメーター](#)を参照してください。
- **EXEC CICS SET AUTOINSTALL** コマンドまたは **CEMT SET AUTOINSTALL** コマンド。こうしたコマンドについて詳しくは、[SET AUTOINSTALL](#) および [CEMT SET AUTOINSTALL](#) を参照してください。

### テストおよびデバッグ

自動インストール制御プログラムの動作をテストするために、そのプログラムを通常の端末関連アプリケーションとして実行することができます。

そのためには、プログラムを定義し、端末から開始します。

プログラムに渡されるパラメーター・リストについては、[138 ページの『INSTALL 時の自動インストール制御プログラム』](#)で説明しています。テスト・プログラムでダミーのパラメーター・リストを構成し、そのリストを基に処理を実行することができます。プログラムを使用する前に端末で実行することは、すなわち、EDF トランザクションを使用してプログラムをデバッグできることを意味します。プログラムを対話式にして、端末からデータの送受信を行うこともできます。

プログラムに対して CICS から自動インストール・モデルが渡されなかった場合は、使用したいモデル名 (AUTINSTNAME) を強制する自動インストール・テスト・プログラムを作成できます。z/OS Communications Server のバッファー・トレースを実行した状態で、デバイスを CICS にログオンさせてみます。CICS が BIND を送信しようとしなかった場合は、以下を確認してください。

- モデル TERMINAL が正しい TYPETERM を参照していること (あるいは、問題の TYPETERM が正しい TERMINAL 定義によって参照されていること)。
- TERMINAL 定義が AUTINSTMODEL(YES または ONLY) であること。
- 自動インストール・モデル (TERMINAL および TYPETERM 定義) を含むグループをインストールしていること。

CICS が BIND を試行した場合は、デバイスの CINIT RU と CICS BIND を比較し、適宜修正してください。

z/OS Communications Server の LOGMODE テーブルの端末に対するエントリーが正しいことを確認することが非常に重要です。誤ってコーディングされたエントリーに合わせて新しい自動インストール・モデルを定義することはやめてください。テスト中は、LOGMODE エントリーが誤ってコーディングされていると CICS 自動インストールが機能しないことを常に念頭に置いてください。

TYPETERM 定義にデバイス属性を指定してもデバイス属性を強制できないことに注意してください。自動インストールの場合、LOGMODE エントリーに定義されている属性が、モデルに定義されている属性と一致する必要があります。そうでない場合、モデルは選択されません。z/OS Communications Server に対して端末を定義した方法とは別の方法で、CICS に対して端末を定義することはできません。

制御プログラムが異常終了した場合、デフォルトでは、CICS はトランザクション・ダンプを書き込みません。異常終了した場合にダンプを取らせるには、プログラムで **EXEC CICS HANDLE ABEND** コマンドを発行する必要があります。



## 「good night」プログラムの作成

**GNTRAN** システム初期設定パラメーターを使用して、端末がタイムアウトしたときに CICS から呼び出す「good night」トランザクションを指定できます。

**GNTRAN** のデフォルト値は NO です。CICS は「good night」トランザクションをスケジュールせず、代わりに端末ユーザーをサインオフさせようとします。サインオフが成功するかどうかは、端末の TYPETERM 定義の SIGNOFF 属性の値によって決まります。

**GNTRAN** パラメーターに指定するトランザクションは、端末のタイムアウトが発生した場合に渡されるタイプの通信域を処理できる必要があります。CICS のサインオフ・トランザクション CESF はこれを行えますが、CESN やその他の付属のトランザクションはこれを行えません。詳しくは、[GNTRAN システム初期設定パラメーター](#)を参照してください。

独自の「good night」プログラムを作成して、サインオフに機能を追加したり、サインオフを置換したりできます。例えば、プログラムで端末ユーザーにパスワードを入力するプロンプトを表示し、正しい応答を受け取ったらセッションの続行を許可するように設定できます。CICS には、これを実行するサンプルの「good night」プログラム DFH0GNIT、および DFH0GNIT を指すサンプルの TRANSACTION リソース GNIT が用意されています。

CICS は 147 ページの図 36 に示すパラメーター・リストを通信域を介して「good night」プログラムに渡します。疑似会話型トランザクション中に端末がタイムアウトした場合に、プログラムでパラメーター・リストの情報を使用して次のアクションを実行できます。

- ユーザーに応答を要求し、その応答を確認する
- タイムアウト・トランザクションによって残された画面をリストアする
- カーソルの位置をリストアする
- タイムアウトしたトランザクションの通信域 (入力メッセージとして「good night」プログラムに渡されます) を受け取る
- 対話の次のトランザクションの TRANSID を使用して戻る。

### 「good night」プログラムの通信域

147 ページの図 36 は、「good night」プログラムに渡される通信域を示しています。

DFHSNGS			
DFHSNGS_FIXED	DS	0CL64	Fixed part of parameter list
GNTRAN_START_TRANSID	DS	CL4	TRANSID that invoked GNTRAN
GNTRAN_PSEUDO_CONV_FLAG	DS	CL1	Pseudoconversational flag
GNTRAN_SCREEN_TRUNCATED	DS	CL1	Screen buffer truncation flag
GNTRAN_TRANSLATE_TIOA	DS	CL1	Uppercase translation flag
	DS	CL9	Reserved
GNTRAN_TIMEOUT_TIME	DS	CL8	Time of terminal timeout
GNTRAN_TIMEOUT_REASON	DS	CL1	Reason for timeout
	DS	CL11	Reserved
GNTRAN_PSEUDO_CONV_TRANSID	DS	CL4	Next transaction ID
GNTRAN_SCREEN_LENGTH	DS	FL2	Length of screen buffer
GNTRAN_CURSOR_POSITION	DS	FL2	Cursor position
GNTRAN_SCREEN_WIDTH	DS	FL2	Width of screen
GNTRAN_SCREEN_HEIGHT	DS	FL2	Height of screen
GNTRAN_USER_FIELD	DS	CL16	Available to user program
DFHSNGS_VARIABLE	DS	0X	Variable part of parameter list
GNTRAN_SCREEN_BUFFER	DS	0X	Contents of screen buffer

図 36. 「good night」プログラムに渡される通信域 (アセンブラー)

#### GNTRAN\_START\_TRANSID

「good night」トランザクションを開始したトランザクションの ID。端末のタイムアウトが理由で CICS が開始した場合、GNTRAN\_START\_TRANSID は「CEGN」に設定されます。プログラムでこのフィールドを検査して、タイムアウト処理に該当すること (つまり、「good night」トランザクションが開始された理由が端末のタイムアウトであり、その他の理由ではないこと) を確認する必要があります。

#### GNTRAN\_PSEUDO\_CONV\_FLAG

端末が疑似会話型トランザクションの中でタイムアウトになったかどうかを示すフラグ。

**Y**

端末は、疑似会話型アプリケーションの一部を形成するトランザクション間でタイムアウトになりました。

**N**

端末は、疑似会話型アプリケーションの一部を形成するトランザクション間でタイムアウトになりませんでした。

#### **GNTRAN\_SCREEN\_TRUNCATED**

3270 画面バッファを切り捨てる必要があったかどうかを示すフラグ。

**Y**

画面バッファは切り捨てられました。

**N**

画面バッファは切り捨てられませんでした。

#### **GNTRAN\_TRANSLATE\_TIOA**

TYPETERM または PROFILE 設定で要求された場合に、DFHZSUP で TIOA を大文字に変換するかどうかを示す内部フラグ。

**Y**

TIOA は変換されます。

**N**

大文字変換は迂回されます。

#### **GNTRAN\_TIMEOUT\_TIME**

端末がタイムアウトになった時刻を CICS ABSTIME 形式で示します。

#### **GNTRAN\_TIMEOUT\_REASON**

タイムアウトになった理由:

**T**

端末からの入力がない

**X**

XRF テークオーバー。

#### **GNTRAN\_PSEUDO\_CONV\_TRANSID**

端末が疑似会話型シーケンスの中でタイムアウトになった場合は、次のトランザクションの ID。(端末が疑似会話型シーケンスの中でタイムアウトにならなかった場合、このフィールドの値に意味はありません。)

#### **GNTRAN\_SCREEN\_LENGTH**

画面バッファの長さ。

#### **GNTRAN\_CURSOR\_POSITION**

カーソル位置。

#### **GNTRAN\_SCREEN\_WIDTH**

端末がタイムアウトになったときに使用されていた画面の幅。

#### **GNTRAN\_SCREEN\_HEIGHT**

端末がタイムアウトになったときに使用されていた画面の高さ。

GNTRAN\_SCREEN\_WIDTH および GNTRAN\_SCREEN\_HEIGHT を使用して、ユーザーの画面を復元するときに ERASE DEFAULT または ERASE ALTERNATE オプションを使用するかどうかを決めます。

#### **GNTRAN\_USER\_FIELD**

このフィールドは、「good night」ユーザー・プログラムで使用できます。これは 2 進ゼロに初期設定され、CICS が変更することはありません。これを使用すると、疑似会話型の「good night」トランザクションの開発に有用です。

#### **GNTRAN\_SCREEN\_BUFFER**

画面バッファの内容を含む可変長フィールド。



## サンプル「good night」プログラム DFH0GNIT

サンプル「good night」プログラムは、DFH0GNIT という名前の疑似会話型 COBOL プログラムです。

「good night」プログラムに渡される通信域のコピーブックは、アセンブラー言語、COBOL、PL/I、および C で提供されています。提供されているプログラム、コピーブック、およびマップ・セットの名前と、それらが含まれる CICSTS56.CICS ライブラリーは、[149 ページの表 13](#) に要約されています。

表 13. サンプル「good night」プログラム、コピーブック、およびマップ・セット		
言語	メンバー名	ライブラリー
プログラム・ソース: COBOL のみ	DFH0GNIT	SDFHSAMP
コピーブック: アセンブラー COBOL PL/I C	DFHSNGSD DFHSNGSO DFHSNGSL DFHSNGSH	SDFHMAC SDFHCOB SDFHPL1 SDFHC370
マップ・セット:	DFH\$GMAP	SDFHSAMP

## サンプル・プログラムの実行内容

DFH0GNIT サンプル・プログラムは以下を実行します。

- CICS によって渡される通信域の GNTRAN\_START\_TRANSID フィールドを検査することで、端末のタイムアウトのために呼び出されたのか検査します。'CEGN' 以外が入っている場合は、終了します。
- GNTRAN\_USER\_FIELD 内のフラグが、このタイムアウトに対する最初の呼び出しであることを示している場合は、以下ようになります。
  - GNTRAN\_PSEUDO\_CONV\_FLAG が、疑似会話中に端末がタイムアウトしたことを示しているなら、EXEC CICS RECEIVE を発行して通信域を取得します。
  - GNTRAN\_USER\_FIELD 内の別のフィールドに、通信域の長さを保管します。
  - 通信域があれば、それを一時記憶域キューに書き込みます。
  - パスワードの入力をユーザーに求める画面を表示し、それがなされたことを示すフラグを設定します。
  - TRANSID GNIT および COMMAREA オプションを指定した EXEC CICS RETURN を発行することで、疑似会話としてタイムアウト・プロセスを続行します。
- これがタイムアウトに対する最初の呼び出しでない場合は、以下ようになります。
  - 元の通信域があれば、それを一時記憶域キューからリカバリーします。
  - ユーザーから受け取ったパスワードを検査し、間違っている場合は、エラー・メッセージを出してタイムアウト画面を再表示します。
- 間違った応答の回数が、外部セキュリティ・マネージャーに指定された最大数を超えた場合、DFH0GNIT は即時に TRANSID CESF で戻ります。これにより、当該ユーザー ID のサインオフが試みられます。
- 正しいパスワードが入力された場合、DFH0GNIT の処理は以下のとおりです。

- 画面内容を復元します。
- カーソル位置を復元します。

疑似会話型トランザクション中に端末がタイムアウトした場合、DFH0GNIT の処理には以下も含まれます。

- タイムアウトしたトランザクションの通信域を復元します。
- 中断された会話における次のトランザクションの TRANSID が設定されて戻ります。

## サンプル「good night」プログラムのカスタマイズ

CICS アプリケーションおよびシステム・プログラミング・インターフェースへの全アクセス権限を利用して、CICS でサポートされる任意の言語で独自の「good night」プログラムを作成することができます。

提供されているプログラムをカスタマイズするか、または独自の「good night」プログラムを作成する場合は、以下の点に留意してください。

- サンプルのように、対象のプログラムを疑似会話型にする必要があります。これは、多数のユーザーのために同時に呼び出される可能性があるためです (例えば、多数の端末が昼休みにタイムアウトになる場合など)。プログラムが会話型である場合、CICS 最大タスク数 (MXT) にすぐに達してしまう可能性があります。

タイムアウト・プログラムの疑似会話を続行しているときは、対象の「good night」トランザクションの名前 (例えば GNIT) を次の TRANSID として必ず指定してください。そうしないと、CICS はタイムアウトの処理が続いていることを認識できず、予測不能の結果が生じる可能性があります。

- 対象のプログラムは必ず、サンプル・プログラムのように、CICS によって渡される通信域の GNTRAN\_START\_TRANSID フィールドを検査することで開始されなければなりません。端末のタイムアウト以外の理由で (例えば、EXEC CICS START 要求によって) 「good night」トランザクションが開始されたことを検出した場合、タイムアウト処理は適切でない可能性があります。
- 疑似会話においてタイムアウトしたトランザクションの通信域を取得するには、プログラムが EXEC CICS RECEIVE コマンドを発行する必要があります。 (呼び出し時に渡される通信域は、タイムアウトしたトランザクションのものとは異なりますが、タイムアウトしたトランザクションに関する情報が含まれます。)
- プログラムが端末ユーザーのサインオフを試みる場合、その結果は端末の TYPETERM 定義の SIGNOFF オプションの指定内容に応じて、以下のように異なります。

#### YES

端末はサインオフされますが、ログオフされません。

#### NO

端末はログオンもサインオンも維持されます。

#### LOGOFF

端末はサインオフとログオフの両方がなされます。

- GNTRAN システム初期設定パラメーターで、対象の「good night」トランザクションの ID (TRANSID) を指定します。

サンプル・プログラム DFHOGNIT をカスタマイズした場合は、提供されているサンプル・トランザクション定義 GNIT を指定します。

DFHOGNIT 以外の名前を付けた、独自の「good night」プログラムを作成した場合は、そのプログラムを指すトランザクション定義をインストールし、GNTRAN SIT パラメーターでその定義を指定する必要があります。

## 端末用のサンプルの自動インストール制御プログラム

CICS 提供のデフォルト自動インストール・プログラムは、DFHZATDX という名前のアセンブラー言語コマンド・レベル・プログラムです。デフォルト・プログラムのソースは、COBOL、PL/I、C、およびアセンブラー言語で提供されています。

提供されているプログラムの名前とその関連コピーブック、およびプログラムで使用されている CICSTS56.CICS ライブラリーについては、[150 ページの表 14](#) にまとめられています。COBOL、PL/I、および C コピーブックには、それぞれ DFHTCUDS の別名があります。

表 14. 自動インストール・プログラムおよびコピーブック			
言語	メンバー名	別名	ライブラリー
プログラム:			
アセンブラー	DFHZATDX	なし	SDFHSAMP
COBOL	DFHZCTDX	なし	SDFHSAMP
PL/I	DFHZPTDX	なし	SDFHSAMP
C	DFHZDTEX	なし	SDFHSAMP

表 14. 自動インストール・プログラムおよびコピーブック (続き)

言語	メンバー名	別名	ライブラリー
コピーブック:			
アセンブラー	DFHTCUDS	なし	SDFHMAC
COBOL	DFHTCUD	DFHTCUDS	SDFHCOB
PL/I	DFHTCUDP	DFHTCUDS	SDFHPL1
C	DFHTCUD	DFHTCUDS	SDFHC370

アセンブラー言語のソース・プログラムから生成されるモジュールは、CICSTS56.CICS.SDFHLOAD でシッ  
プされている事前生成済みライブラリーの一部です。このソース・コードは変更せずに使用することもで  
きますし、要件に合わせてカスタマイズすることもできます。サンプル・プログラムのコードを変更する  
場合は、サンプルをコピーして、それに変更を加えてください。変更後に、正しい手順でモジュールを変  
換、アセンブル、およびリンク・エディットしてください。その後、ロード・モジュールを、DFHRPL ス  
テートメントで CICSTS56.CICS.SDFHLOAD の前に連結されているユーザー・ライブラリーに書き込みま  
す。(この方法は新規作成したモジュールと変更したサンプル・モジュールに適用されます。) 提供されてい  
るプロシージャに関する指針については、[アプリケーション・プログラムをインストールするための  
CICS 提供プロシージャの使用](#)を参照してください。カスタマイズしたモジュールでサンプルを上書き  
しないでください。後続のサービスによってモジュールが上書きされる可能性があるためです。カスタマ  
イズしたユーザー・プログラムの新しいリソース定義をインストールする必要があります。

INSTALL 実行時の、サンプル・プログラムのデフォルト・アクションは、リストの最初のモデルを選択し、  
NETNAME の最後の非ブランクの 4 文字から端末 ID を取り出し、状況バイトを設定して、CICS に戻りま  
す。リストにモデルがない場合は、アクションを行わずに戻ります。

DELETE 実行時のデフォルト・アクションは、渡されたパラメーター・リストをアドレス指定し、アクショ  
ンを行わずに CICS に戻ります。

ご利用のインストール済み環境に適した処理を実行するように、サンプル・プログラムをカスタマイズで  
きます。カスタマイズの例が 151 ページの『[サンプル・プログラムのカスタマイズ](#)』に示されています。  
一般に、ユーザー・プログラムを使用して以下の処理を行うことができます。

- ・ログオン端末の総数をカウントおよび制限する。
- ・自動的にインストールされる端末の数をカウントおよび制限する。
- ・特定の端末に関する使用状況情報を保持する。
- ・TERMINAL 名と NETNAME をマップする。
- ・自動的にインストールされる端末の TNADDR (TCP/IP クライアント・アドレス、IP ポート、およびオプ  
ションでホスト名) をマップする。
- ・汎用ロギングを実行する。
- ・特殊なケースを処理する (特定の端末またはユーザーのログオンを常に許可するなど)。
- ・オペレーターにメッセージを送信する。
- ・自動インストールに対するネットワーク全体の制御を行う。ネットワーク全体のグローバルな自動イン  
ストール制御プログラムは、1 つの CICS システムに常駐させることができます。リモート CICS システ  
ムの制御プログラムからの自動インストール要求を受信すると、このグローバル制御プログラムが呼び出  
され、データが制御プログラム間で転送されます。

### サンプル・プログラムのカスタマイズ

実稼働環境でサンプル・プログラムを使用する前に、インストール環境に合わせてカスタマイズする必要  
があります。

#### アセンブラー言語

152 ページの図 37 では、アセンブラー言語でネット名 L77A および L77B へのログオンを制限しています。  
使用されているモデル名は、事前に認識されています。他の端末からのログオン要求、または見つからな  
いモデルに対する要求は、拒否されます。

```

* REGISTER CONVENTIONS =
* R0 free
* R1 free
* R2 Base for Input parameters
* R3 Base for code addressability
* R4 Base for model name list
* R5 Base for output parameter list
* R6 Work register
* R7 -----
* R8 -----
* R9 free
* R10 Internal subroutine linkage return
* R11 Base for EIB
* R12 free
* R13 Base for dynamic storage
* R14 free
* R15 free
*
* SELECT MODEL
*
*      LH R6, TABLEN      Number of valid netnames
*      LA R7, TABLE      Address the table
*
* LOOP1 CLC NETNAME(4), 0(R7) Is this netname in table?
*      BE VALIDT
*
*      LA R7, 16(R7)      Next table entry
*      BCT R6, LOOP1
*
*      Now we know its not a valid netname
*      return and the logon is rejected
*
*      B RETURN
*

```

図 37. DFHZATDX サンプル・プログラムをカスタマイズする方法の例 (パート 1)

```

*
VALIDT  CLI  SELECTED_MODELNAME,C' '      R7 now points to model name
        BNE  VALIDM1                      MTS model name supplied?
        LH   R6,MODELNAME_COUNT           Yes
        LTR  R6,R6                        Count of models
        BZ   RETURN                       Were any presented?
        LA   R8,MODELNAME                 No
                                           First model name
*
LOOP2   CLC  8(8,R7),0(R8)                Is this model name here?
        BE   VALIDM
*
        LA   R8,L'MODELNAME(R8)           Next model name
        BCT  R6,LOOP2
*
*                                           Now we know the required model name was not presented
*                                           to this exit by CICS, a return rejects the logon
*
        B    RETURN
*
*                                           At this point the model name was found in those presented
*                                           It is given to CICS and the new termid is
*                                           the netname
*
VALIDM  MVC  SELECTED_MODELNAME,0(R8)      R8 was left pointing at
*                                           model name
VALIDM1 DS   0H
        MVC  SELECTED_TERM_ID,NETNAME     Use netname for termid
                                           (4 chars)
*
*
* SELECTIONS COMPLETE, RETURN
*
        MVI  SELECTED_RETURN_CODE,X'00'   Indicate all OK
        B    RETURN                       Exit program
*
*                                           Table of netnames allowed to log on and the model name
*                                           necessary for the logon to be successful
*
*                                           Format of table :
*                                           Bytes 1 to 8   Netname allowed to log on
*                                           9 to 16   Model required for netname
*
        DS   0D
TABLE   DC   CL8'L77A',CL8'3270064'
        DC   CL8'L77B',CL8'3270065'
TABLEN DC   Y((*-TABLE)/16)
*

```

図 38. DFHZATDX サンプル・プログラムをカスタマイズする方法の例 (パート 2)

## COBOL

154 ページの図 39 では、COBOL で、最後の 4 文字を使用して、サンプル制御プログラムで選択されたものより適したモデルを選択できるように NETNAME を再定義します。

```

*
*
* Redefine the netname so that the last 4 characters (of 7)
* can be used to select the autoinstall model to be used.
*
* The netnames to be supplied are known to be of the form:
*
*      HVMXNNN
*
* HVM is the prefix
* X   is the system name
* NNN is the address of the terminal
*
01 NETNAME-BITS.
02 FIRST-CHRS PIC X(3).
02 NEXT-CHRS.
03 NODE-LETTER PIC X(1).
03 NODE-ADDRESS PIC X(3).
02 LAST-CHR PIC X(1).
*
*
PROCEDURE DIVISION.
*
*
*
* Select the autoinstall model to be used according to the
* node letter (see above). The models to be used are user
* defined.
*
* (It is assumed that the netname supplied in the commarea by CICS
* has been moved to NETNAME-BITS).
*
* If the node letter is C then use model AUT02
* If the terminal netname is HVMC289 (a special case) then use
* model AUT01.
* Otherwise (node letters A,B,D...) use model AUT03.
*
      IF NODE-LETTER = 'C' THEN MOVE 'AUT02' TO SELECTED-MODELNAME.
      IF NEXT-CHRS = 'C289' THEN MOVE 'AUT01' TO SELECTED-MODELNAME.
      IF NODE-LETTER = 'A' THEN MOVE 'AUT03' TO SELECTED-MODELNAME.
      IF NODE-LETTER = 'B' THEN MOVE 'AUT03' TO SELECTED-MODELNAME.
      IF NODE-LETTER = 'D' THEN MOVE 'AUT03' TO SELECTED-MODELNAME.
*
*

```

図 39. DFHZCTDX サンプル・プログラムをカスタマイズする方法の例

## PL/I

155 ページの図 40 では、PL/I で、z/OS Communications Server の CINIT RU から情報を抽出しています。この RU には、BIND イメージが含まれています。この情報には、デフォルトの画面サイズや代替画面サイズなどの画面表示サービスの情報が含まれています。代替画面サイズを使用して、ログオンを要求している端末のモデルを判別しています。提示されたモデルの中で一致を検索し、一致するものがなかった場合は、それらのモデルのうち最初のモデルを使用します。



```

DCL 1 CINIT                                BASED(INSTALL_CINIT_PTR),
      2 CINIT_LEN                          FIXED BIN(15),
      2 CINIT_RU                          CHAR(256);
DCL  SAVE_CINIT                          CHAR(256);
/* Temp save area for CINIT RU */
DCL 1 SCRNSZ                              BASED(ADDR(SAVE_CINIT)),
      2 SPARE                              CHAR(31),
/* Bypass first part of CINIT and reach */
/* into BIND image carried in CINIT */
      2 DHGT                              BIT(8),
/* Screen default height in BIND PS area */
      2 DWID                              BIT(8),
/* Screen default width in BIND PS area */
      2 AHGT                              BIT(8),
/* Screen alternate height in BIND PS area */
      2 AWID                              BIT(8);
/* Screen alternate width in BIND PS area */
DCL  NAME                                CHAR(2);
/* Used to work up a screen model type */
DCL  TERMID                              PIC'9999' INIT(1) STATIC;
/* Used to work up a unique termid */
DCL  ENQ                                CHAR(8) INIT('AUTOPRG');
/* Used to prevent multiple access to termid */
/* If model name supplied by MTS, bypass model name selection */
IF SELECTED_MODELNAME ^= ' '
  THEN GO TO MODEL_EXIT;
/* Clear the CINIT save area and move in the */
/* z/OS Communications Server CINIT RU.*/
/* This is useful if you fail to recognize the model */
/* of terminal; provide a dump and analyze this data */
SAVE_CINIT = LOW(256);
SUBSTR(SAVE_CINIT,1,CINIT_LEN) = SUBSTR(CINIT_RU,1,CINIT_LEN);

```

図 40. DFHZPTDX サンプル・プログラムをカスタマイズする方法の例 (パート 1)

```

/* Now access the screen PS area in the portion of the BIND
image presented in the CINIT RU */
/* using the screen alternate height as a guide to the model
of terminal attempting logon. If this cannot be determined
then default to the first model in the table */
SELECT (AHGT); /* NOW GET SCRNR ALTERNATE HEIGHT */
WHEN (12) NAME = 'M1'; /* MODEL 1 */
WHEN (32) NAME = 'M3'; /* 3 */
WHEN (43) NAME = 'M4'; /* 4 */
WHEN (27) NAME = 'M5'; /* 5 */
OTHERWISE NAME = 'M2'; /* 2 */
END;
/* Search the model entries for a matching entry. */
/* The criterion here is that a model definition should */
/* contain the chars M2 for a model 2, and so on. */
/* For example, L3270M2, L3270M5 */
/* TERMM2, TERMM5 */
IF MODELNAME_COUNT = 0
THEN GO TO EXIT;
DO I = 1 TO MODELNAME_COUNT;
IF INDEX(MODELNAME(I),NAME)
THEN GO TO FOUND_MODEL;
END;
NO_MODEL: /* Matching entry was not found, default to first model */
SELECTED_MODELNAME = MODELNAME(1);
GO TO MODEL_EXIT;
FOUND_MODEL: /* Move the selected model name to the return area */
SELECTED_MODELNAME = MODELNAME(I);
MODEL_EXIT: /* ENQ to stop multiple updates of counter. */
/* A simple counter is used to generate unique */
/* terminal identities, so concurrent access to */
/* this counter is denied to ensure no two get */
/* the same identifier or update the counter. */

/* To use this method the program must be defined as resident.*/
EXEC CICS ENQ RESOURCE(ENQ);
SELECTED_TERMID = TERMID; /* Set SELECTED_TERMID to
count value */
TERMID = TERMID + 1; /* Increase the count value by 1 */
IF TERMID = 9999 THEN TERMID = 1; /* Reset if too large */
EXEC CICS DEQ RESOURCE(ENQ);
NAME_EXIT:
INSTALL_RETURN_CODE = LOW(1);
/* Set stat field to X'00' to allow
logon to be processed */
GO TO EXIT;
END INSTALL;

```

図 41. DFHZPTDX サンプル・プログラムをカスタマイズする方法の例 (パート 2)

## コンソールの自動インストールを制御するプログラムの作成

MVS コンソール・デバイス (TSO コンソールを含む) の自動インストールを制御するプログラムを作成できます。

ローカル接続 SNA LU の自動インストールの制御については、[136 ページの『LU の自動インストールを制御するプログラムの作成』](#)を参照してください。

## コンソールの自動インストール - 事前の考慮事項

MVS コンソールの自動インストールを使用する理由は、SNA デバイス用 z/OS Communications Server の自動インストールに当てはまる理由と同じです。つまり、各デバイスを明示的に定義する必要がなく、ストレージに保管します ([136 ページの『端末の自動インストール』](#)を参照)。

### CICS でコンソールを自動インストールする方法

通常の自動インストール・サポートに加えて、CICS では、自動インストール・プログラムを呼び出さずにコンソールを自動インストールすることも可能です。

自動インストール・プログラムを呼び出さないコンソールの自動インストールを指定した場合、CICS は以下のオプションのいずれかを使用します。

- MVS コンソール名と一致する AUTINSTNAME (モデル名) を含むモデル・コンソール定義
- 英数字順で最初に見つかった適切なコンソール・モデル

自動インストール制御プログラムを呼び出さない場合、CICS は ¬ (論理否定) 記号で始まる 4 文字の端末 ID を生成します。

CICS でコンソールを自動インストールする場合は、以下の操作を実行する必要があります。

- AICONS=AUTO を指定するか、自動インストールのための以下のいずれかのオプションを使用します。
  - CICS Explorer の「領域」操作ビューの「属性」タブ
  - **CEMT SET AUTOINSTALL FULLAUTO** コマンド
  - **EXEC CICS SET AUTOINSTALL FULLAUTO** コマンド
  - **EXEC CICS SET AUTOINSTALL CONSOLES(1073)** コマンド
- DEVICE(CONSOLE) を指定した TYPETERM 定義を参照するモデル TERMINAL 定義を 1 つ以上作成します。要件に合えば、グループ DFHTERM に含まれている IBM 提供の定義を使用できます。
- モデル TERMINAL および TYPETERM 定義をインストールします。

### 自動インストール・プログラムの使用

デフォルトの自動インストール制御プログラムを使用することも、デフォルトのプログラムのソース・コードおよびカスタマイズ例をベースとして使用して、独自のプログラムを作成することもできます。

### このタスクについて

自動インストール制御プログラムをコンソールに対して開始する場合は、以下の手順を実行します。

#### 手順

1. 端末用のデフォルトの自動インストール制御プログラム (DFHZATDX または DFHZATDY) を使用するか、デフォルトのプログラムのソース・コードおよびカスタマイズ例をベースとして使用して独自のプログラムを作成します。  
端末、コンソール、および APPC 接続を処理する自動インストール制御プログラムは 1 つしかアクティブにできません。AIEXIT システム 初期設定パラメーターでアクティブ・プログラムの名前を指定します。自動インストール・プログラムには、コンソールの **INSTALL** および **DELETE** のパラメーター・リストを認識し、モデル名、端末 ID、および戻りコードを返す機能がなければなりません。
2. システム 初期設定パラメーターとして **AICONS=YES** を指定するか、CICS Explorer の「領域」操作ビューの「属性」タブを使用するか、**SET AUTOINSTALL CONSOLES(PROGAUTO)** コマンドを発行して、CICS のコンソールの自動インストール機能を有効にします。
3. AIEXIT システム 初期設定パラメーターを指定して、CICS に自動インストール制御プログラムを定義します。

### タスクの結果

自動インストール機能が有効になっている場合、CICS は、以下の条件が満たされたときに、自動インストール制御プログラムを開始します。

- 自動インストールの **INSTALL** 要求の処理が行われている。
- 自動インストール要求が自動インストール制御プログラムに受け入れられたが、その後の **INSTALL** 処理が失敗した。
- コンソールが最後に使用された後、一定の遅延期間が経過した。

## INSTALL 時の自動インストール制御プログラム

自動インストールが有効な場合は、138 ページの『[INSTALL 時の自動インストール制御プログラム](#)』にリストされているデバイスに加えて、MVS コンソールについても、CICS が自動インストール制御プログラムを呼び出すことを指定できます。コンソール用自動インストール制御プログラムを CICS が呼び出せるようにするには、システム 初期設定パラメーターとして **AICONS=YES** を指定するか、**SET AUTOINSTALL CONSOLES(PROGAUTO)** コマンドを発行します。

自動インストール制御プログラムの呼び出しごとに、CICS は DFHEICAP でアドレッシングされた通信域を使用して、パラメーター・リストを制御プログラムに渡します。この情報は、コンソール定義のうちのインストール機能のみ示しています。

制御プログラムは、以下の時点でコンソールの `INSTALL` で呼び出されます。

- CICS に対してコンポーネント名が定義されていない MVS コンソールから、CICS が `MODIFY` コマンドを受け取った。
- MVS `MODIFY` コマンドで渡された CICS トランザクションを処理するために自動インストール制御プログラムからの端末 ID と自動インストール・モデル名を必要とするポイントまで、CICS が自動インストール処理を完了した。

## コンソールの `INSTALL` での通信域

通信域のレイアウトを [158 ページの図 42](#) に示します。

Fullword 1	Standard Header	
Byte 1	Function Code	(X'FD' for INSTALL)
Bytes 2 - 3	Component Code	Always 'ZC'
Byte 4	Reserved	Always X'00'
Fullword 2	Pointer to CONSOLENAME_FIELD	
Fullword 3	Pointer to MODELNAME_LIST	
Fullword 4	Pointer to SELECTED_PARS	
Fullword 5	Reserved	

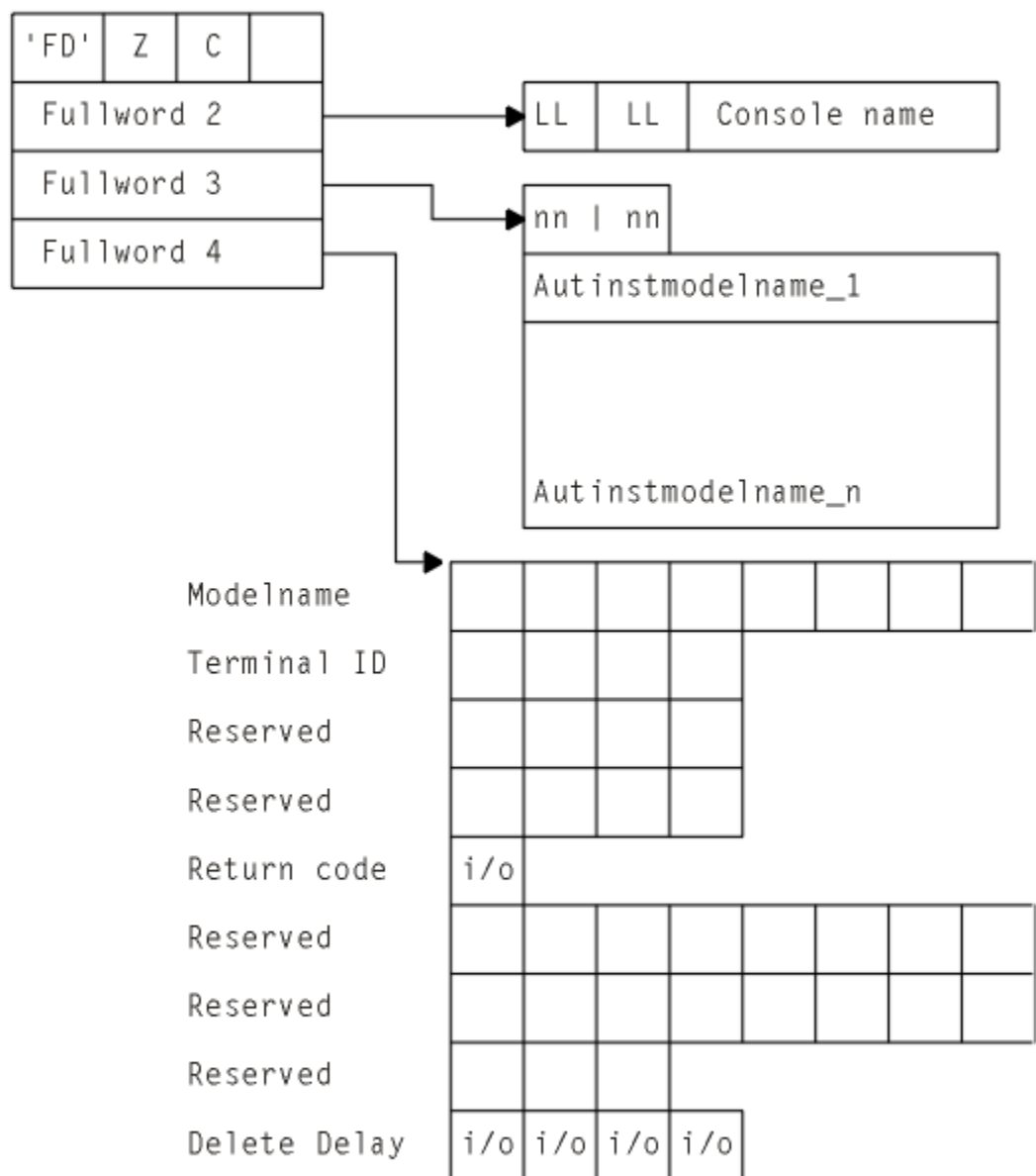
図 42. コンソールの `INSTALL` での自動インストール制御プログラムの通信域

パラメーター・リストには、次の情報が含まれています。

1. 標準ヘッダー。バイト 1 は要求タイプを示します (`INSTALL` の場合は 16 進文字 `X'FD'`)。バイト 2 から 3 にはコンポーネント・コードが含まれ、常にコンソールの場合の `ZC` です。(バイト 4 は予約済みです。)
2. 2 バイト長フィールドへのポインター。この後に、メッセージを送信したコンソールのコンソール名が続きます。
3. 対象となる自動インストール・モデルの名前の配列へのポインター。配列の前に、配列内の 8 バイトの名前エレメントの数を含む 2 バイト・フィールドがあります。配列に要素が何もない場合、数値フィールドはゼロに設定されます。
4. CICS に情報を戻すために使用するストレージ域へのポインター。

CICS は、パラメーター・リストのフルワード 3 でアドレッシングされた領域内に、対象となる自動インストール・モデルのリストを渡します。制御プログラムは、このリストからコンソール・デバイスに適したモデルを選択し、パラメーター・リストのフルワード 4 でアドレッシングされた領域の最初の 8 バイトにモデル名を移さなければなりません。CICS に戻る前に、制御プログラムはログオンされているコンソールの CICS 端末 ID (4 文字) を指定し、自動インストール要求が許可される場合は戻りコード・フィールドを `X'00'` に設定します。コンソールが最後に使用されてから CICS によって自動的に削除されるまでの遅延期間をプログラムで設定することもできます。この値は、自動インストール制御プログラムに入るときに、デフォルト値の 60 分に設定されます。この値をオーバーライドするには、独自の遅延期間 (分単位) をフルワード・バイナリー値として保管します。このフィールドをゼロ (0) に設定すると、CICS はコンソールを削除しないことになります。

[159 ページの図 43](#) は、これらのフィールドのすべてを、必要とされる順序で示しています。



Note: i/o designates an input/output field.  
The other fields in SELECTED\_PARMS are output only.

図 43. INSTALL での自動インストール制御プログラムのパラメーター・リスト

### CICS による自動インストール・モデルのリストの作成方法

CICS は、端末モデルの完全なリストからコンソール・デバイスを定義するモデルを選択して、自動インストール・モデルのリストを作成します。

CICS がいつでも使用できる 自動インストール・モデルの完全なリストには、インストール済みの AUTINSTMODEL(YES) および AUTINSTMODEL(ONLY) が含まれるすべての定義と、DEVICE(CONSOLE) を指定する TYPETERM 定義を参照する定義が含まれます。

CICS は、コンソール用のモデルを検出できないと、メッセージ DFHZC6902 を発行します。パラメーター・リストのフルワード 4 によってアドレス指定された領域にある戻りコードがゼロ以外の値の場合、CICS はエラー・メッセージ DFHZC6987 を発行します。

CICS Explorer の「領域」操作ビュー、CEMT、または **EXEC CICS INQUIRE AUTINSTMODEL** コマンドを使用して、自動インストール・モデル定義のリストを取得できます。

## CICS への情報の戻し

INSTALL イベントでは、自動インストール制御プログラムが CICS 領域への新しいコンソール・リソースのインストールを許可するか拒否するかを決定します。この決定は、セキュリティーなどのいくつかのインストール依存の要因や、接続されている端末の合計数などに基づいて行われます。この検査に CICS はまったく関与しません。このような検査を行うかどうか、またそれをどのように行うかは、ユーザーが決定します。

## このタスクについて

INSTALL 要求を処理するには、ご使用の自動インストール制御プログラムで以下の手順を実行します。

- 自動インストール・モデル名を、パラメーター・リストのフルワード 4 によってアドレス指定された領域の最初の 8 バイトで戻します。
- 戻り領域の次の 4 バイトに CICS 端末名 (TERMIN) を指定します。DFHZATDX および DFHZATDY は、コンソール名の最後の 4 つの非空白文字 (パラメーター・リストのフルワード 2 によってアドレス指定されたもの) を取って、端末名として使用します。これがご使用のインストール済み環境の命名規則に沿わない場合、独自の自動インストール・プログラムをコーディングします。
- 戻りコードを X'00' に設定します。自動インストール制御プログラムへの入り口では、戻りコードには常にゼロ以外の値が含まれます。これを変更しない場合、自動インストール要求は拒否されます。
- 削除の遅延期間を指定するか、デフォルト値の 60 分のままにします。

自動インストール制御プログラムによって設定できる TERMINAL 定義の属性は、これらのみであることにご注意ください。他のすべての属性は、以下のいずれかのソースから取得する必要があります。

- MVS コンソール・インターフェース・ブロック (CIB)
- 自動インストール・モデルの TERMINAL 定義
- 参照先の TYPETERM 定義

制御プログラムが INSTALL 要求を拒否すると判断した場合、戻りコードにゼロ以外の値を指定して CICS に戻ります。処理が完了したら、制御プログラムは EXEC CICS RETURN コマンドを発行して CICS に戻る必要があります。

## 自動インストール・モデルの選択

プログラムに渡されるリスト内のすべてのモデルは、コンソール用です。つまり、実行可能な TCT 項目は通常、それらのいずれかを使用することで生成されます。デフォルトの自動インストール制御プログラムは、リストの最初のモデルを選択します。しかし、必要な属性が必ずしもそのモデルに用意されているとは限りません。制御プログラムは、コンソールで必要な性質を持つ (例えば、必要なセキュリティー特性を持っているなど) モデルを選択する必要があります。

## TERMINAL 値の設定

TERMINAL 値は固有である必要があります、長さは 1 文字から 4 文字である必要があります。TERMINAL 値は、CICS がコンソールで使用する 名前、または ID です。CONSNAME 値は、MVS がコンソールで使用する ID です。

制御プログラムが、既に TCT 項目を持つ TERMINAL 値の TCT 項目をインストールしようとする、端末が適格で、適したモデルが見つかった場合でも、インストールは拒否されます。ただし、一致する CONSNAME 値を持つ項目が既に TCT にある CICS の MVS コンソールから MODIFY コマンドを受信した場合は、CICS はその項目を使用し、自動インストール制御プログラムを開始しません。

デフォルトの自動インストール制御プログラムは、CONSNAME 値の最後の非空白の 4 文字から TERMINAL 値を作成します。これは、端末名が固有ではない可能性があることを意味します。この問題を回避する 1 つの方法として、CICS Explorer の「端末」操作ビューを使用するか、または制御プログラムから EXEC CICS INQUIRE コマンドを使用して、TERMINAL 値が既に使用されているかを判別します。すでに使用されている場合は、最後の文字を変更して、再度確認します。

## 制御プログラムから戻ったときの CICS アクション

CICS は、自動インストール制御プログラムから制御を戻されると、以下のように戻りコード・フィールドを検査します。



- 戻りコードがゼロで、提供されたその他の必要な情報に問題がない場合、CICS は MODIFY コマンドに指定されたトランザクションをスケジュールに入れて、要求を完了させます。インストール・プロセスが失敗すると、DELETE 機能が処理されるのと同じように、自動インストール制御プログラムが再び駆動されます。
- 戻りコードがゼロ以外の場合、CICS は、自動インストールが有効ではない場合に CICS に変更要求を送信する不明なコンソールによる試行を拒否するのと同じように、接続要求を拒否します。

すべての自動インストール・アクティビティーで、一時データ宛先 CADL にメッセージが書き込まれます。INSTALL が失敗する場合、理由コードと一緒にメッセージが CADL に送信されます。CADL からの出力を確認すると、自動インストール要求が失敗した理由を見つけることができます。

## DELETE 時の自動インストール制御プログラム

コンソール自動インストール要求が失敗したとき、または削除遅延期間が満了したときに、自動インストール制御プログラムを開始できます。

自動インストール・プロセスのもう一方の制御として、以下の状況が発生したときも、自動インストール制御プログラムが開始されます。

- コンソール自動インストール要求がユーザー・プログラムに受け入れられたが、INSTALL プロセスが失敗した。
- コンソールが最後に使用されてから削除遅延期間が経過し、CICS は AICONS=YES が有効な状態で実行中である。コンソールの自動インストールのこの状況は、CICS Explorer 「領域」操作ビューを使用するか、**CEMT INQUIRE AUTOINSTALL** コマンドを発行することによって照会できます。AICONS=YES が指定されていれば、CONSOLES オプションの値は PROGAUTO になります。

プログラムへの入力は、DFHEICAP でアドレッシングされた通信域を経由します。通信域のレイアウトを次の図に示します。

Fullword 1	Standard Header
Byte 1	Function Code (X'FE')
Bytes 2 - 3	Component Code Always 'ZC'
Byte 4	Reserved Always X'00'
Fullword 2	Terminal ID of console being deleted
Fullword 3	Consolename of console being deleted
Bytes 1-2	Deleted consolename length
Bytes 3-4	Start of deleted consolename ID
Next 6 bytes	Remainder of deleted consolename ID

図 44. コンソールの DELETE での自動インストール制御プログラム通信域

パラメーター・リストには、次の情報が含まれています。

1. 標準ヘッダー。バイト 1 は要求タイプを示します (16 進文字 X'FE' は DELETE を表します)。バイト 2 から 3 にはコンポーネント・コードが含まれ、コンソールの場合は常に ZC です。
2. 2 番目のフルワードには、削除されるコンソールの端末 ID が含まれます。
3. 3 番目のフルワードの最初の 2 バイトには、削除されるコンソール名の長さ、最後の 2 バイトにはコンソール名の最初の文字と 2 番目の文字が含まれます。
4. 通信域の最後の 6 バイトには、コンソール名の残りの部分 (3 番目から 8 番目の文字まで) が含まれます。

## コンソールの自動インストール制御プログラム・サンプル

CICS はデフォルトの自動インストール制御プログラムを提供します。サポートされる各プログラミング言語で書かれ、そのすべてにコンソールを扱うために必要なサポートが含まれています。

プログラムについて詳しくは、[150 ページの『端末用のサンプルの自動インストール制御プログラム』](#)を参照してください。

## APPC 接続の自動インストールを制御するプログラムの作成

ローカル APPC 接続の自動インストールを制御するプログラムを作成できます。

ローカル SNA LU の自動インストールの制御について詳しくは、136 ページの『LU の自動インストールを制御するプログラムの作成』を参照してください。シップされた端末と接続のインストールの制御について詳しくは、174 ページの『シップされた端末の自動インストールを制御するプログラムの作成』を参照してください。CICS クライアントで使用する仮想 LU のインストールの制御について詳しくは、180 ページの『仮想端末の自動インストールを制御するプログラムの作成』を参照してください。

### APPC 接続の自動インストール - 事前の考慮事項

ローカル APPC 接続の自動インストールを検討するうえで、次の違いを区別する必要があります。

1. CINIT 要求によって開始されたローカル APPC 単一セッション接続
2. 着信したバインド要求によって開始されたローカル APPC 並列セッション接続および単一セッション接続(「着信」とは、要求がパートナー・システムによって開始されたことを意味します。)

#### CINIT によって開始されたローカル APPC 単一セッション接続

CINIT 要求によって開始されたローカル APPC 単一セッション接続の自動インストールは、端末の自動インストールと同じように機能します。TERMINAL-TYPETERM モデルのペア、および、用意されている自動インストール制御プログラム DFHZATDX または DFHZATDY のいずれかのカスタマイズ・バージョンを用意する必要があります。

136 ページの『LU の自動インストールを制御するプログラムの作成』を参照してください。

#### BIND によって開始されたローカルの APPC 並列セッション接続および単一セッション接続

自動インストールを有効にすると、APPC サービス・マネージャー (SNASVCMG) のセッション (または単一セッション接続の唯一のセッション) で着信 APPC BIND 要求を受信した場合に、一致する CICS CONNECTION 定義がなければ、新しい接続が自動的に作成され、インストールされます。

APPC 接続用の自動インストールでは、他のリソース用の自動インストールと同様にモデル定義を必要とします。しかし、端末の自動インストールで使用するモデル定義と違い、APPC リンクの自動インストールに使用するモデル定義は、モデルとして明示的に定義する必要はありません。CICS は、インストール済みの接続定義を新しい定義の「テンプレート」として使うことができます。自動インストールが機能するためには、自動インストールを適用する接続の種類ごとにテンプレートが必要です。

#### APPC 接続用の自動インストール・テンプレート

テンプレートの目的は、同じ特性をもつすべての接続に使用できる定義を CICS に提供することです。z/OS Communications Server for SNA から受け取る情報に応じて、新しい接続ごとに適切なテンプレートを選択するように、用意されている自動インストール制御プログラム、DFHZATDY をカスタマイズします。

テンプレートは、CONNECTION 定義とそれに対応する SESSIONS 定義からなります。必要になるセッション特性群ごとに 1 つの定義をインストールする必要があります。

インストール済みの任意の接続定義をテンプレートとして使用できますが、パフォーマンスを考慮すると、使用しないインストール済みの接続定義をテンプレートとすることが推奨されます。CICS が定義をコピーしている間は定義はロックされるため、多数のセッションを自動インストールする場合には、それによる遅れが顕著になることがあります。

#### 自動インストールの利点

同じ特性をもつ APPC 並列セッション・デバイスが多数ある場合は、自動インストールのサポートが役立つと期待されます。

例えば、すべて同じ特性を持つパーソナル・コンピュータ (PC) が 1000 台ある場合、テンプレートを 1 つ設定すれば、そのすべてを自動インストールすることができます。また、ある特性の PC が 500 台、別の特性の PC が 500 台ある場合には、2 つのテンプレートを使ってそれらを自動インストールすることができます。

あらゆる再始動が大幅に速くなります。端末数が多い場合は特に顕著です。

また、システム管理のオーバーヘッドやストレージの節約にもつながります。これは、自動インストールされたリソースは、使用されるまでスペースを占有することがないためです。

## 自動インストールの要件

APPC 接続の自動インストールは、サポート対象のすべてのリリースの ACF/SNA で使用できます。

端末および接続でアクティブにできる自動インストール制御プログラムは、1 つだけです。アクティブ・プログラムの名前を AIEXIT システム初期設定パラメーターで指定する必要があります。BIND 要求によって開始された自動インストール APPC 接続に機能を提供することに加え、サンプル・プログラム、DFHZATDY は、デフォルトの制御プログラム、DFHZATDX の端末の自動インストールと同じ機能を提供します (136 ページの『[LU の自動インストールを制御するプログラムの作成](#)』で説明されています)。そのため、カスタマイズしたバージョンの DFHZATDY を使用して、端末と APPC 接続の両方を自動インストールできます。

**注：**DFHZATDX および DFHZATDY の両方が、シップされた端末や接続、およびクライアントの仮想端末をインストールする機能を提供します。

提供されているバージョンの DFHZATDY で、使用目的には十分である場合があります。そうでない場合は、提供されたプログラムのカスタマイズ・バージョンを作成でき、また拡張機能を提供するために独自のプログラムを作成することもできます。

## APPC 接続用の自動インストール制御プログラム

自動インストール制御プログラムの目的は、CICS にその自動インストール要求の完了に必要なすべての特別な情報を提供することです。APPC 接続の場合、制御プログラムは、使用するテンプレートを選択し、新規の接続に名前を提供します。

自動インストールが使用可能な場合、CICS が SNASVCMG セッション (または単一セッション接続の唯一のセッション) の APPC BIND 要求を受信したときに、一致する CONNECTION 定義がないなら、CICS はパートナーの z/OS Communications Server NETNAME を自動インストール制御プログラムに渡します。この制御プログラムは、通信域に渡される BIND からの情報を使用して、新規接続のベースとなる最適なテンプレートを選択します。

制御プログラムは、最適なテンプレートの名前を戻すために、すべてのテンプレートの NETNAME または SYSID を認識している必要があります。不適切なテンプレートを使用しようとすると、メッセージ DFHZC6922 が発行されて、そのテンプレートを使用できない理由が示されます。

テンプレートが使用可能な場合、CICS はそれに含まれる定義のコピーを作成し、新規 CONNECTION 定義のインストールを試みます。このインストールが失敗した場合は、メッセージ DFHZC6903 が発行されます。

## リカバリーと再始動

自動インストールされた接続は CICS によってカタログ作成されないので、緊急リスタートまたはウォーム・リスタートの際にリカバリーされません。

## INSTALL 時の自動インストール制御プログラム

自動インストール制御プログラムは、次の INSTALL 時に呼び出されます。

- ローカル SNA LU
- MVS コンソール
- CINIT によって開始されたローカル APPC 単一セッション接続
- BIND によって開始されたローカル APPC 並列セッション接続
- BIND によって開始されたローカル APPC 単一セッション接続
- シップされた端末と接続
- クライアント仮想端末。

CICS は、呼び出しごとに、DFHEICAP によってアドレッシングされる通信域を経由して制御プログラムにパラメーター・リストを渡します。CINIT によって開始されたローカル端末と APPC 単一セッション接続の INSTALL 時に渡されるパラメーター・リストについては、[139 ページの『INSTALL 実行時の端末の通信域』](#)で説明します。MVS コンソールの INSTALL 時に渡されるパラメーター・リストについては、[157 ページの『INSTALL 時の自動インストール制御プログラム』](#)を参照してください。シップされた端末と接続の INSTALL 時に渡されるパラメーター・リストについては、[177 ページの『シップされた端末の INSTALL 時の通信域』](#)を参照してください。クライアント仮想端末の INSTALL 時に渡されるパラメーター・リストに

については、183 ページの『クライアント仮想端末の INSTALL 時の通信域』を参照してください。このセクションでは、BIND 要求によって開始されたローカル APPC 接続の INSTALL についてのみ説明します。

### APPC 接続の INSTALL 時の通信域

通信域は、アセンブラー・バージョンの DFHZATDY の DSECT によってマップされます。この DFHZATDY は、CICSTS56.CICS.SDFHMAC にあります。

```

*-----*
* APPC Install parameter list - Functions 2, 3, and 4 *
*-----*
INSTALL_APPC_COMMAREA      DSECT      Install Parameter List
*
INSTALL_APPC_STANDARD      DS  F        Standard field
                                ORG INSTALL_APPC_STANDARD
INSTALL_APPC_EXIT_FUNCTION DS  XL1      Install request type
INSTALL_APPC_PS_CINIT      EQU X'F2'    Install PS via CINIT
INSTALL_APPC_PS_BIND       EQU X'F3'    Install PS via BIND
INSTALL_APPC_SS_BIND       EQU X'F4'    Install SS via BIND
INSTALL_APPC_EXIT_COMPONENT DS  CL2      Component ID 'ZC'
                                DS  XL1      Reserved
*
                                ORG ,
INSTALL_APPC_NETNAME_PTR    DS  A        -> NETNAME          Input
INSTALL_APPC_CINIT_PTR     DS  0A        -> CINIT_RU         Input
INSTALL_APPC_BIND_PTR      DS  A        -> BIND              Input
INSTALL_APPC_SELECTED_PTR  DS  A        -> Return fields     Output
INSTALL_APPC_SYNCLEVEL_PTR DS  A        -> Sync level        Input
*
INSTALL_APPC_TEMPLATE_NETNAME_PTR DS A -> Template NETNAME I/O
INSTALL_APPC_TEMPLATE_SYSID_PTR DS A -> Template SYSID   Output
INSTALL_APPC_SYSID_PTR     DS  A        -> New SYSID        Output
INSTALL_APPC_NETNAME2_PTR  DS  A        -> Generic or      Input
                                member NETNAME
*
INSTALL_APPC_NETID_PTR     DS  A        -> Network ID of    Input
                                incoming bind
*
INSTALL_APPC_TYPE_PTR      DS  A        -> Generic          Input
                                resource type
*
TEMPLATE_NETNAME           DS  CL8      Put netname of template here
TEMPLATE_SYSID             DS  CL4      Put sysid of template here
SYSID                     DS  CL4      Put name of new connection here
SYNCLEVEL                 DS  XL2      Synclevel of new connection
APPC_NETID                DS  CL8      NETID of incoming bind
APPC_GR_TYPE              DS  CL1      G = NETNAME is generic resource name
                                M = NETNAME is member name
                                blank = This CICS is not a generic
                                resource or the partner is not a
                                generic resource.
*
APPC_NETNAME2_FIELD        DSECT
APPC_NETNAME2_LENGTH      DS  XL2      Length of NETNAME
APPC_NETNAME2             DS  0X      Generic or member NETNAME

```

図 45. INSTALL 時の自動インストール制御プログラムの通信域

### INSTALL\_APPC\_STANDARD ヘッダー

フルワード入力フィールドは、次の情報で構成されます。

### INSTALL\_APPC\_EXIT\_FUNCTION

インストール要求タイプを定義する 1 バイトのフィールド。同等の値は以下のとおりです。

### INSTALL\_APPC\_PS\_CINIT

X'F2' は、CINIT 要求による 2 次ノードからの APPC 並列セッション接続のインストール要求を表します。

注：これらの要求は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 では受信できません。

### INSTALL\_APPC\_PS\_BIND

X'F3' は、BIND による APPC 並列セッション接続のインストール要求を表します。

### INSTALL\_APPC\_SS\_BIND

X'F4' は、BIND による APPC 単一セッション接続のインストール要求を表します。



注: 値 X'F0' と X'F1' はそれぞれ、端末のインストール要求と削除要求を表します (APPC 単一セッション・デバイスを含む)。136 ページの『[LU の自動インストールを制御するプログラムの作成](#)』を参照してください。

#### **INSTALL\_APPC\_EXIT\_COMPONENT**

2 バイトのコンポーネント・コード。「ZC」に設定されます。

#### **INSTALL\_APPC\_NETNAME\_PTR**

2 バイトの長さフィールドを指すフルワード・ポインタ (入力フィールド)。インストールされる NETNAME が続きます。

パートナーが汎用リソースである CICS TOR への接続では、APPC\_GR\_TYPE の設定に応じて、NETNAME はパートナーの汎用リソース名、またはメンバー名になります (汎用リソースの概要情報については、『[Configuring](#)』の『[Configuring z/OS Communications Server generic resources](#)』を参照してください)。

#### **INSTALL\_APPC\_CINIT\_PTR**

着信セッションが 2 次の場合の着信 CINIT を含む入力フィールドを指すフルワード・ポインタ。

注: CICS Transaction Server for z/OS, バージョン 5 リリース 6 には適用されません。

#### **INSTALL\_APPC\_BIND\_PTR**

2 バイトの長さフィールドを指すフルワード・ポインタ。着信 BIND を含む入力フィールドが続きます。

#### **INSTALL\_APPC\_SELECTED\_PTR**

戻りフィールドを指すフルワード・ポインタ。これらは端末の自動インストールと同じ形式です。

APPC 自動インストール (X'F3' および X'F4' の機能) では、戻りコードのみが使用されることに注意してください。APPC のその他の情報は、通信域で定義されたその他のフィールドに入れて返されます。

#### **INSTALL\_APPC\_SYNCLEVEL\_PTR**

接続の同期点レベルを示す 2 バイトの入力フィールドを指すフルワード・ポインタ。BIND から抽出されます。有効値は以下のとおりです。

**X'0000'**

同期レベル 0

**X'0001'**

同期レベル 1

**X'0002'**

同期レベル 2。

#### **INSTALL\_APPC\_TEMPLATE\_NETNAME\_PTR**

8 バイトの入出力域 (TEMPLATE\_NETNAME) を指すフルワード・ポインタ。起動時は、通常 TEMPLATE\_NETNAME にはブランクが含まれています。ただし、パートナーとローカルの CICS の両方が汎用リソースとして登録されている場合は、汎用リソース名接続の NETNAME が (存在していれば) 含まれています (汎用リソース名接続については、『[総称リソース環境における接続の定義](#)』を参照してください)。

制御プログラムは TEMPLATE\_NETNAME フィールドを使用して、テンプレートの NETNAME を指定できます。汎用リソース間の接続では、プログラムは CICS から渡されたテンプレートの候補を受け入れたり、このフィールドに別の値を指定したり、テンプレートの候補をブランクで上書きして TEMPLATE\_SYSID フィールドに値を設定したりできます。

指定した名前が 8 バイトに満たない場合は、末尾ブランクを埋め込む必要があります。テンプレートの NETNAME を指定する代わりに、プログラムで TEMPLATE\_SYSID に CONNECTION 名を指定する場合、TEMPLATE\_NETNAME はブランクで埋める必要があります。

#### **INSTALL\_APPC\_TEMPLATE\_SYSID\_PTR**

制御プログラムがテンプレートの SYSID (接続名) を指定するために使用できる 4 バイトの出力域 (TEMPLATE\_SYSID) を指すフルワード・ポインタ。名前が 4 バイトに満たない場合は、末尾ブランクを埋め込む必要があります。テンプレートの SYSID を指定する代わりに、プログラムで TEMPLATE\_NETNAME に NETNAME を指定する場合は、TEMPLATE\_SYSID をゼロで埋める必要があります。

### INSTALL\_APPC\_SYSID\_PTR

新しく自動インストールされた接続の SYSID をプログラムで入力する必要がある 4 バイトの出力域を指すフルワード・ポインター。指定する名前は固有の名前でなければなりません。端末 ID の作成に使用するものと同じロジックまたは類似したロジックを使用して作成できます。名前が 4 バイトに満たない場合は、末尾ブランクを埋め込む必要があります。

リカバリー可能なリソースを使用する場合は、再始動後の接続のために選択する SYSID は、その前の CICS の実行で選択されたものと同じでなければなりません。

### INSTALL\_APPC\_NETNAME2\_PTR

2 バイトの長さフィールドを指すフルワード・ポインター。8 バイトの入力フィールド (APPC\_NETNAME2) が続きます。

パートナーとローカルの CICS が両方とも汎用リソースである場合は、APPC\_GR\_TYPE の設定に応じて、APPC\_NETNAME2 はパートナーの汎用リソース名またはメンバー名です。

パートナーが汎用リソースでない場合は、APPC\_NETNAME2 には NETNAME と同じ値が含まれています。

ローカル CICS が汎用リソースでない場合、APPC\_NETNAME2 の値は意味がありません。

### INSTALL\_APPC\_NETID\_PTR

パートナーのネットワーク ID を含む 8 バイトの入力フィールドを指すフルワード・ポインター。このフィールドは、ローカル CICS が汎用リソースとして登録されている場合に、必ず設定されます。その他の場合はすべて、0 の値になります。

### INSTALL\_APPC\_GR\_TYPE\_PTR

1 バイトの入力フィールドを指すフルワード・ポインター。これが汎用リソース間の接続かどうかを示し、そうである場合は、BIND に渡された NETNAME がパートナーの汎用リソース名とメンバー名のどちらであるかを示します。同等の値は以下のとおりです。

#### G

NETNAME はパートナーの汎用リソース名であり、APPC\_NETNAME2 はメンバー名 (アプリケーション ID) です。

#### M

NETNAME はパートナーのメンバー名 (アプリケーション ID) であり、APPC\_NETNAME2 は汎用リソース名です。

#### ブランク

この CICS が汎用リソースとして登録されていないか、パートナーが登録されていません。

## DELETE 時の自動インストール制御プログラム

自動インストール・プロセスを対称に制御できるように、自動インストールされた APPC 接続が削除されるときにも、自動インストール制御プログラムが呼び出されます。

DELETE 時に制御プログラムを呼び出すことによって、INSTALL イベントで実行されたプロセスを元に戻すことができます。例えば、INSTALL 時に制御プログラムで自動インストール・リソースの総数を増やした場合は、DELETE 時に制御プログラムでその数を減らすことができます。

プログラムに対する入力は、DFHEICAP でアドレス指定された通信域を介して行われます。通信域のレイアウトを 166 ページの図 46 に示します。

Fullword 1	Standard Header	
Byte 1	Function Code	(X'F5' or X'F6')
Bytes 2 - 3	Component Code	Always "ZC"
Byte 4	Reserved	Always X'00'
Fullword 2	SYSID of deleted connection	
Fullword 3	NETNAME of deleted connection	
Bytes 1-2	NETNAME length	
Bytes 3-10	NETNAME	

図 46. DELETE 時の自動インストール制御プログラムの通信域



ファンクション・コードのバイト (フルワード 1 のバイト 1) は、ユーザー・プログラムが呼び出された理由を示します。

#### **X'F5'**

BIND により開始された並列セッションの APPC 接続が削除された後。

#### **X'F6'**

BIND により開始された単一セッションの APPC 接続が削除された後。

注：値 X'F1' は、CINIT 要求を介して自動インストールされたローカル端末または APPC 単一セッション・デバイスの削除を表します。詳しくは、[145 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照してください。値 X'FA' または X'FB' はシップされた端末または接続の削除を表します。詳しくは、[178 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照してください。値 X'FC' は、クライアント仮想端末の削除を表します。詳しくは、[186 ページの『DELETE 時の自動インストール制御プログラム』](#)を参照してください。

### **自動インストールされた APPC 接続が削除される場合**

自動インストールされた APPC 接続項目は、接続が破棄されるとすべて削除されます。また接続項目は、端末またはシステムがログオフされたり、CICS から切断されたりした場合も削除されます。

このような暗黙的な削除は、次のタイプの自動インストールされた APPC 接続で発生します。

- CINIT によってインストールされた単一セッション接続。

これらの接続は、AILDELAY システム初期設定値の有効期限が切れた後に、端末ユーザーがログオフすると削除されます ([AILDELAY システム初期設定パラメーター](#)を参照)。

- BIND によってインストールされた Synclevel 1 接続。

BIND 要求によって自動インストールされたほとんどの同期レベル 1 のみの APPC 接続は、次のような場合に暗黙的に削除されます。

- 接続が解放された場合
- SNA が異常終了した場合
- CICS が SNA ACB をクローズする場合
- ウォーム・スタートまたは緊急始動に続いて AIRDELAY インターバルの期限が終了した後 (AIRDELAY システム初期設定パラメーターの値がゼロより大きい場合)。[AIRDELAY システム初期設定パラメーター](#)を参照してください。

ただしこれは、CICS 総称リソース・メンバーにインストールされた限定リソース接続には適用されません。BIND によってインストールされた同期レベル 2 接続を参照してください。

- BIND によってインストールされた同期レベル 2 接続。

BIND 要求によってインストールされた同期レベル 2 対応の APPC 接続は、それらが CICS 総称リソース・メンバーにインストールされており、親和性が終了した場合にのみ、暗黙的に削除されます。それ以外の場合は、暗黙的に削除されることはありません。

これは、同期レベル 1 専用の、CICS 総称リソース・メンバーにインストールされた限定リソース接続にも当てはまります。

## **APPC 接続用自動インストール制御プログラムのサンプル**

APPC 接続の自動インストール用のサンプル制御プログラムは DFHZATDY です。ソース・コードはアセンブラ言語のみで、ライブラリー CICSSTS56.CICS.SDFHSAMP にあります。

DFHZATDY は、BIND 要求により開始された APPC 接続を自動インストールする機能のほかに、端末を自動インストールするための DFHZATDX プログラムの機能 ([136 ページの『LU の自動インストールを制御するプログラムの作成』](#)を参照) と同じ機能を備えています。カスタマイズ・バージョンの DFHZATDY を使用して、端末と APPC 接続の両方を自動インストールすることができます。

### **サンプル・プログラムのデフォルトのアクション**

APPC 接続のインストールにおける DFHZATDY の役割は、使用するテンプレートを選択すること (その NETNAME または SYSID を指定)、および新規接続の名前 (SYSID) を指定することです。

提供されたバージョンのプログラムによるアクションは、次のとおりです。

1. INSTALL\_APPC\_EXIT\_FUNCTION フィールドで渡される、次の要求タイプを調べます。

#### **X'F0'**

端末または APPC 単一セッション・デバイスの着信 CINIT。DFHZATDX に対して開始されます。  
[136 ページの『LU の自動インストールを制御するプログラムの作成』](#)を参照してください。

#### **X'F1'**

端末または APPC 単一セッション・デバイスの削除要求。DFHZATDX に対して開始されます。[136 ページの『LU の自動インストールを制御するプログラムの作成』](#)を参照してください。

#### **INSTALL\_APPC\_PS\_CINIT (X'F2')**

APPC 並列セッション接続の着信 CINIT。INSTALL\_APPC\_TEMPLATE\_SYSID が指すフィールドを 'CCPS' に設定して、テンプレートを指定します。

注：このタイプの要求は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 では受信できません。

#### **INSTALL\_APPC\_PS\_BIND (X'F3')**

APPC 並列セッション接続の着信 BIND。テンプレートを指定します。これは、次の 2 つの方法のいずれかの方法で行われます。

- 2 つの総称リソース間の接続の場合、NETNAME が TEMPLATE\_NETNAME に渡される、提案されたテンプレート (総称リソース名接続) を受け入れます。総称リソース名接続がない場合、TEMPLATE\_SYSID を 'CBPS' に設定します。
- それ以外のすべての場合は、TEMPLATE\_SYSID を 'CBPS' に設定します。

#### **INSTALL\_APPC\_SS\_BIND (X'F4')**

APPC 単一セッション接続の着信 BIND。INSTALL\_APPC\_TEMPLATE\_SYSID が指すフィールドを 'CBSS' に設定して、テンプレートを指定します。

#### **X'F5'**

BIND によってインストールされた APPC 並列セッション接続の削除要求。COMMAREA に対してアドレッシング可能にし、戻ります。

#### **X'F6'**

BIND によってインストールされた APPC 単一セッション接続の削除要求。COMMAREA に対してアドレッシング可能にし、戻ります。

2. INSTALL\_APPC\_NETNAME\_PTR が指す入力 NETNAME の末尾の 4 文字の非ブランク文字を  
INSTALL\_APPC\_SYSID\_PTR が指すフィールドにコピーすることで、新規接続の名前を指定します。
3. 戻りコードを RETURN\_OK に設定して、選択を行ったことを示します。

### **リソース定義**

CICS は、DFHZATDY を定義し、CCPS、CBPS、および CBSS 用の CONNECTION 定義が含まれる、DFHAI62 と呼ばれるリソース定義グループを提供します。

提供されたバージョンの DFHZATDY を使用する場合、DFHAI62 を CICS 始動グループ・リストに追加する必要があります。ただし、DFHZATDY をカスタマイズする場合は通常、独自の定義を作成する必要があります。

DFHZATDY は、DFHAI62 で次のように定義されています。

```
DEFINE PROGRAM(DFHZATDY)
  DESCRIPTION(Assembler definition for sessions autoinstall control program)
  GROUP(DFHAI62)
  LANGUAGE(ASSEMBLER) RELOAD(NO) RESIDENT(NO)
  USAGE(NORMAL) STATUS(ENABLED) CEDF(NO)
  DATALOCATION(ANY) EXECCKEY(CICS) EXECUTIONSET(FULLAPI)
```

## IPIC 接続の自動インストールを制御するプログラムの作成

IPIC 接続の自動インストールを制御するプログラムを作成できます。

IPIC を含むさまざまなタイプの CICS 相互通信リンクの概要については、[相互通信方式](#)を参照してください。

### IPIC 接続の自動インストール: 事前の考慮事項

IPCONN 自動インストール・ユーザー・プログラムは、APPC 自動インストール・ユーザー・プログラムに似ています。IPCONN 自動インストール・ユーザー・プログラムは、APPC 自動インストール・ユーザー・プログラムと同様に、インストール済みの接続を選択して新規接続用のテンプレートとして使用します。大きく異なる点は、テンプレートが CONNECTION 定義ではなく IPCONN 定義である点と、テンプレートの使用はオプションである点です。

IPCONN 自動インストールがアクティブの場合は、CICS は以下の情報を使用して新規の IPIC 接続をインストールします。

- 接続フロー内の情報
- IPCONN 自動インストール・ユーザー・プログラムによってオプションで選択された IPCONN テンプレート
- ユーザー・プログラムによって返された通信域内の各値
- CICS 提供値

### IPCONN リソース用の自動インストール・テンプレート

IPCONN リソース用の自動インストールでは、他のリソース用の自動インストールと違って、モデル定義を必要としませんが、推奨はされています。ただし、端末の自動インストールに使用されるモデル定義とは異なり、IPCONN リソースの自動インストールに使用されるモデル定義では、モデルとして明示的に定義される必要はありません。代わりに、CICS は、前にインストール済みの任意の IPCONN 定義を新規の定義用のテンプレートとして使用することができます。

テンプレートの目的は、CICS にすべての接続に同じプロパティを用いて使用できる定義を提供することです。提供された自動インストール・ユーザー・プログラムが CICS から受信する情報に応じて、新しい接続ごとに適切なテンプレートを選択するために、このプログラムをカスタマイズします。

すべてのインストール済み IPCONN 定義がテンプレートとして使用できますが、パフォーマンス上の理由により、適切なテンプレートとして、使用しないインストール済みの定義を使用してください。定義は、CICS がそれをコピー中はロックされ、同時に自動インストールされる IPCONN が多数存在する場合は、著しい遅延が発生するおそれがあります。

### 自動インストールの要件

以下の条件が満たされる場合は、IPCONN 自動インストールはアクティブになります。

1. 受信側領域には、PROTOCOL(IPIC) を指定した TCIPSERVICE を少なくとも 1 つインストールしている必要があります。
2. IPCONN 自動インストール・ユーザー・プログラムの名前が、インストール済みの TCIPSERVICE 定義の URM オプションに指定されている必要があります。

**注:** この要件は、自動インストール・ユーザー・プログラムの名前が AIEXIT システム初期設定パラメーターに指定される APPC 接続の自動インストールと異なる点です。IPCONN 自動インストールには、同等のシステム初期設定パラメーターはありません。代わりに、自動インストール・ユーザー・プログラムの名前は、TCIPSERVICE 定義上に指定されます。

APPC と同様に、テンプレートの各 IPCONN をサービス休止にすると、自動インストール機能が使用不可になります。

### ユーザー・プログラムが呼び出された場合

ユーザー・プログラムは、以下の条件の両方が満たされた場合に呼び出されます。

1. PROTOCOL(IPIC) で定義された TCIPSERVICE リソースが、インストール済みの IPCONN 定義が存在していない NETWORKID と APPLID の組み合わせを含む接続フロー、または APPLID がヌルの接続フローのいずれかを受信した。特定の IPv6 アドレスの代わりに HOST(ANY) が受信側 CICS システムの TCIPSERVICE 定義に指定されている場合は、IPCONN はデフォルトの IPv4 アドレスを使用して通信を保証できるようにします。
2. 受信側 TCIPSERVICE リソースの URM 属性に自動インストール・ユーザー・プログラムの名前が指定されている。URM 属性に NO が含まれている場合に、自動インストール要求が拒否された。

### IPCONN リソース用自動インストール・ユーザー・プログラム

自動インストール・ユーザー・プログラムの目的は、CICS にその自動インストール要求の完了に必要なすべての特別な情報を提供することです。IPIC 接続の場合、ユーザー・プログラムは、新規の接続に名前を提供します。ユーザー・プログラムは、オプションでサービス中の IPCONN 定義を選択してテンプレートとして使用することができ、また新規の接続の APPLID、HOST、および PORT の属性の値を接続フロー上で提供された各属性の値から変更することができます。

自動インストールされる IPCONN リソース上の RECEIVECOUNT 属性は、接続フロー上でクライアントから要求された値に設定されるか、またはテンプレートが指定されている場合は、この値とテンプレートからの RECEIVECOUNT 値のうち小さい方の値に設定されます。

自動インストールされた IPCONN リソースの SENDCOUNT 属性は、テンプレートが指定されている場合は、テンプレートの RECEIVECOUNT 値または SENDCOUNT 値と同じ値に設定されます。

新規の IPCONN 定義の他のすべての属性は、テンプレートから取得されるか、テンプレートが指定されていない場合は CICS 提供値から取得され、ユーザー・プログラムが変更することはできません。

選択されたテンプレートが使用可能な場合は、CICS は、その内部でこの定義のコピーを作成して、新規の IPCONN 定義のインストールを試みます。このインストールが失敗した場合は、メッセージが発行されます。

デフォルトの自動インストール・ユーザー・プログラムの DFHISAIP は、アセンブラ言語プログラムです。APPC と IPIC 自動インストールの間の主な相違点は、プロトコルとして IPIC が指定された TCIPSERVICE の URM オプションのデフォルト値が DFHISAIP であることです。したがって、IPIC 接続がデフォルトで自動インストールされます。自動インストールを使用不可にするには、URM=NO を TCIPSERVICE リソース定義で指定します。DFHISAIP は 8 文字の IPCONN ID を作成するので、CICS 間の通信に IPIC 接続を使用する場合は、4 文字の IPCONN 名と 4 つの末尾スペースを必ず指定してください。その理由は、端末専有領域の REMOTESYSTEM 属性が IPCONN の先頭 4 文字だけを読み取るためです。

デフォルトのユーザー・プログラムが目的に適していない場合は、デフォルトのプログラムのカスタマイズ・バージョンを作成でき、また拡張機能を提供するために独自のプログラムを作成することもできます。CICS では、デフォルトのプログラムのソース・コードを数種類のプログラミング言語で提供しています (173 ページの『IPIC 接続 (IPCONN) 用の自動インストール・ユーザー・プログラムのサンプル』を参照)。

### リカバリーと再始動

自動インストールされる IPCONN リソースは、緊急時再始動時のリカバリー専用で、CICS によってカタログ作成されます。ウォーム・リスタート時にはリカバリーされません。

## INSTALL 時の自動インストール・ユーザー・プログラム

IPCONN リソースをインストールするために自動インストール・ユーザー・プログラムが呼び出されると、CICS は DFHEICAP でアドレス指定された通信域でパラメーター・リストをそのプログラムに渡します。

### IPCONN の INSTALL 時の通信域

通信域はアセンブラー DSECT DFHISAIC によりマップされます。このアセンブラーは、CICSTS56.CICS.SDFHSAMP 内に用意されています。

ISAIC_FUNCTION	DS CL1	Function code (X'F0' for Install)
ISAIC_RESPONSE	DS CL1	Response code
	DS CL2	Reserved
ISAIC_IPCONN	DS CL8	Name for the autoinstalled IPCONN
ISAIC_APPLID	DS CL8	The applid of remote system
ISAIC_SUGGESTED_APPLID	DS CL8	Suggested applid, if isaic_applid is blank
*		
ISAIC_NETWORKID	DS CL8	Network ID of remote system
ISAIC_TCPIPSERVICE	DS CL8	Name of the TCPIP SERVICE on which this connect flow arrived
*		
ISAIC_TEMPLATE	DS CL8	Name of the template IPCONN
ISAIC_HOST	DS CL116	Host name of remote system
ISAIC_PORT	DS F	Call back port number of remote system
*		
ISAIC_RECEIVECOUNT	DS F	Number of receive sessions wanted by remote system
*		

図 47. INSTALL 時の自動インストール・ユーザー・プログラムの通信域

#### isaic\_applid (入出力)

接続フローで送信された、接続を試行しているリモート・システムのアプリケーション ID を含む 8 文字のフィールド。ユーザー・プログラムでこの値を変更できるのは、入力時にこの値がブランクであった場合のみです (通常は接続システムが Java クライアントであることを示しています)。出力時にこの値がブランクである場合、CICS は isaic\_suggested\_applid が指す「提示したアプリケーション ID」を使用します。

#### isaic\_function (入力)

自動インストール・ユーザー・プログラムが呼び出された機能を示す 1 文字のコード。インストールの場合は X'F0' が含まれます。

#### isaic\_host (入出力)

接続フローで渡された、リモート・システムのホスト名を含む 116 文字のフィールド。これは自動インストール・ユーザー・プログラムで変更可能です。接続してきたシステムをローカルで識別するための名前は、そのシステムよりも自動インストール・ユーザー・プログラムのほうが正確に認識しているからです。

#### isaic\_ipconn (出力)

自動インストールする IPCONN 接続で使用する名前を含む 8 文字のフィールド。この名前はユーザー・プログラムが提供しなければなりません。

#### isaic\_networkid (入力)

接続フローで送信された、接続を試行しているシステムのネットワーク ID を含む 8 文字のフィールド。

#### isaic\_port (入出力)

クライアントのコールバック・ポート番号。-1 は、コールバックできないことを意味します。自動インストール・ユーザー・プログラムは、ホスト名を変更できる理由と同じ理由でこの値を変更できません。ただし、-1 の場合は例外で、変更できません。自動インストール・ユーザー・プログラムはこの値を -1 に変更することもできません。

#### isaic\_receivecount (入力)

この IPCONN によりリモート・システムでサポートする受信セッションの数を含む 4 バイトの 2 進数フィールド (リモート・システムの終了時には送信セッションになります)。

#### isaic\_response (出力)

応答コード: ゼロは OK を意味します。



### isaic\_suggested\_applid (入力)

CICS が「提示」したリモート・システム・アプリケーション ID を含む 8 文字のフィールド。isaic\_applid が指すリモート・システムのアプリケーション ID がブランクの場合、CICS はカウンターを使用して「00000027」形式の 8 文字の 10 進数の名前を生成します。

### isaic\_tcpipservice (入力)

この接続フローが到着した TCPIPService の名前を含む 8 文字のフィールド。

### isaic\_template (入出力)

新しい IPCONN リソースのテンプレートとして使用するインストール済みの IPCONN の名前を含む 8 文字のフィールド。

デフォルトでは、このフィールドはブランクです。CICS が、IPCONN リソースの作成に必要な情報を提供します。

自動インストール・ユーザー・プログラムでこのフィールドを変更して、テンプレート IPCONN リソースの名前を指定することができます。テンプレート IPCONN リソースがサービス休止の場合、自動インストール要求は拒否されます。

## DELETE 時の自動インストール・ユーザー・プログラム

自動インストール・プロセスを対称に制御できるように、自動インストールされた IPCONN リソースが削除されるときにも、自動インストール・ユーザー・プログラムが呼び出されます。

DELETE 時にユーザー・プログラムを呼び出すことで、INSTALL イベント時に実行したプロセスを取り消すことができます。例えば、自動インストールされたリソースの総数カウントを INSTALL 時にユーザー・プログラムで増やした場合、DELETE 時にユーザー・プログラムでそのカウントを減らします。

プログラムに対する入力は、DFHEICAP でアドレス指定された通信域を介して行われます。通信域のレイアウトを [172 ページの図 48](#) に示します。

ISAIC_FUNCTION	DS CL1	Function code (X'F1' for Delete)
	DS CL3	Reserved
ISAIC_IPCONN	DS CL8	Name of the autoinstalled IPCONN
ISAIC_APPLID	DS CL8	Applid of the autoinstalled IPCONN
	DS CL8	Reserved
ISAIC_NETWORKID	DS CL8	Network ID of the autoinstalled IPCONN
ISAIC_TCPIPService	DS CL8	Name of the TCPIPService on which
*		this connect flow arrived

図 48. DELETE 時の自動インストール・ユーザー・プログラムの通信域

### isaic\_function (入力)

ユーザー・プログラムが呼び出された機能。

X'F1'

IPCONN の削除の後。

### isaic\_ipconn (入力)

自動インストールされた IPCONN の名前。

### isaic\_applid (入力)

自動インストールされた IPCONN に対して指定されている (リモート・システムの) アプリケーション ID。

### isaic\_networkid (入力)

自動インストールされた IPCONN に対して指定されている (リモート・システムの) ネットワーク ID。

### isaic\_tcpipservice (入力)

接続フローが到着した TCPIPService の名前。

DELETE 通信域内のフィールドはすべて入力専用です。

## 自動インストールされた IPCONN の削除時

自動インストールされた IPCONN は、RELEASED 状態に移行すると必ず削除されます。RELEASED 状態の IPCONN が存在し続けるのは、緊急時再始動で CICS 間の IPCONN が復元される場合だけです。この場合は、再獲得でリカバリー処理できるようになるのを待機します。



## IPIC 接続 (IPCONN) 用の自動インストール・ユーザー・プログラムのサンプル

IPCONN の自動インストールのためのデフォルトのユーザー・プログラムは、DFHISAIP というアセンブラー言語プログラムです。その通信域を定義する、対応するコピーブックは DFHISAIC です。

デフォルトのプログラムのソース・コードとその通信域のコピーブックは、アセンブラー言語、COBOL、PL/I、および C のバージョンのものが用意されています。用意されているプログラムとコピーブック、およびそれらが含まれている CICSTS56.CICS ライブラリーについて、[173 ページの表 15](#) に要約しています。

表 15. IPCONN 自動インストール・ユーザー・プログラムおよびコピーブック			
	言語	メンバー名	ライブラリー
Programs (プログラム)	アセンブラー	DFHISAIP	SDFHSAMP
Programs (プログラム)	C	DFHISDIP	SDFHSAMP
Programs (プログラム)	COBOL	DFHISCIP	SDFHSAMP
Programs (プログラム)	PL/I	DFHISPIP	SDFHSAMP
コピーブック	アセンブラー	DFHISAIC	SDFHSAMP
コピーブック	C	DFHISAIC	SDFHC370
コピーブック	COBOL	DFHISAIC	SDFHCOB
コピーブック	PL/I	DFHISAIC	SDFHPL1

独自の IPCONN 自動インストール・ユーザー・プログラムを COBOL、PL/I、C、またはアセンブラー言語で作成できます。

### サンプル・プログラムのデフォルトのアクション

IPCONN のインストールにおける自動インストール・ユーザー・プログラムの役割は、使用する IPCONN テンプレートを選択し、新規接続の名前を渡すことです。オプションで、新しい接続の APPLID、HOST、および PORT 属性の値を、接続フローで渡された値から変更することもできます。新規 IPCONN の他のすべての属性は、テンプレートから取得されるか、CICS 提供のデフォルト値から取得され、ユーザー・プログラムで変更することはできません。

### INSTALL 時

IPCONN 定義をインストールするために呼び出されると、用意されているバージョンのユーザー・プログラムは以下のアクションを実行します。

1. isaic\_ipconn フィールドに、isaic\_applid フィールド内にある接続システムのアプリケーション ID の最後の 4 文字 (ブランクを除く) と同じ値を設定して、新規接続の名前を指定します。  
  
isaic\_applid フィールドがブランクの場合は、その値と接続名に、isaic\_suggested\_applid フィールド内の「提示したアプリケーション ID」を設定します。
2. その他のすべての接続属性はデフォルト値になるようにして、戻ります。

### DELETE 時

IPCONN 定義を削除するために呼び出されると、指定されたバージョンのユーザー・プログラムは一切の操作を行わずに、即時に戻ります。

### リソース定義

CICS には、DFHISCIP というリソース定義グループが用意されています。DFHISCIP は、用意されているデフォルトの自動インストール・プログラム DFHISAIP を定義します。

これは、デフォルトの CICS 始動グループ・リスト DFHLIST に含まれています。異なる CICS 始動グループ・リストを使用する場合、DFHISCIP グループをそれに追加していることを確認してください。

DFHISAIP をカスタマイズした場合には、独自のリソース定義の作成が必要になることがあります。作成した場合は、必ず、そのリソース定義グループを CICS 始動グループ・リストに追加してください。

### 事前定義された接続テンプレートをサポートするためのサンプル自動インストール・ユーザー・プログラム

事前にインストールされたテンプレート IPCONN に応じて IPCONN 名および APPLID が生成されるように IPCONN の自動インストールをカスタマイズする手法を説明するサンプルとして、ユーザーが置き換え可能なプログラムのソースが提供されています。

ユーザーが置き換え可能な追加の IPCONN 自動インストール・プログラムのソースは、SDFHSAMP ライブラリーに提供されています。このコードは、アセンブラーではモジュール DFH\$ISAI として、COBOL ではモジュール DFH0ISAI として提供されています。実行可能ロード・モジュールは、CICSTS SDFHLOAD ライブラリーに提供されています。

このサンプルは、説明のために提供されており、特定の要件に合わせて調整することができます。このプログラムは、同じシステムから複数のクライアントに接続する場合に適しています。例えば、CICS Transaction Gateway でローカル・モードで稼働する WebSphere® Application Server for z/OS を使用する場合などです。

ユーザー置き換え可能プログラムがデプロイされると、すべての IPIC インストール要求は、テンプレート IPCONN に基づくものとなります。それは、パートナーのネットワーク ID の名前 (CICS Transaction Gateway クライアントの場合は APPLID 修飾子) に一致していなければなりません。接続要求が受け入れられるのは、パートナーの APPLID が、テンプレート IPCONN の中で指定されている APPLID 値と一致する場合のみです。

## INSTALL

ユーザー置き換え可能プログラムは、INSTALL 実行時に以下の機能を提供します。

以下の場合、接続要求が拒否されます。

- パートナーのネットワーク ID が、インストール済み IPCONN テンプレートの名前と一致しない。
- パートナーの APPLID が、IPCONN テンプレートに指定された APPLID 値と異なる長さになっている。
- パートナーの APPLID が、IPCONN テンプレートに指定された APPLID 文字と一致しない。
- パートナーの IP アドレスが、HOST 値と一致しない (IPCONN テンプレートにこの値が定義されている場合)。
- 自動インストールされる IPCONN 名 (パートナーの APPLID とその後ろに続くカウント接尾部で構成される) の最大数を超過している。

接続要求が受け入れられる場合、パートナーの APPLID とそれに続く固有の整数接尾部を使用することによって、自動インストールされる IPCONN の値と、それにより指定される APPLID とが動的に生成されます。接尾部は、ユーザー置き換え可能サンプル・モジュールがテンプレート IPCONN ごとに保守するカウントの値から生成されます。カウントの現行値は、CICS 一時ストレージ・エレメントに記録されます。例えば、指定された APPLID の長さが 7 文字の場合 (例えば CTGSYST)、完全な 8 文字の IPCONN 名にするために、1 文字のカウント番号が追加されます (CTGSYST1)。このようにして、同じパートナーから最大 9 個のクライアントに接続できます。

## DELETE

ユーザー置き換え可能プログラムは、DELETE 実行時に以下の機能を提供します。

IPCONN 定義を削除するために呼び出されると、指定されたバージョンのユーザー・プログラムは一切の操作を行わずに、即時に戻ります。

## シッパされた端末の自動インストールを制御するプログラムの作成

シッパされた端末および接続のインストールを制御するプログラムを作成できます。付属の自動インストール制御プログラム DFHZATDX と DFHZATDY はどちらも、リモート端末およびリモート接続のシッパされた定義をインストールする機能を提供します。DFHZATDX または DFHZATDY のいずれかを基に、カスタマイズした制御プログラムを作成できます。

端末専有領域 (TOR) で自動インストール・ユーザー・プログラムを使用してローカルの端末および接続の自動インストールを制御できるのとまったく同じように、アプリケーション所有領域 (AOR) でも同様のプログラムを使用して、シップされた端末および接続のインストールを制御することができます。

## シップされた端末と接続のインストール

自動インストール制御プログラムは、シップされた端末と接続に対して起動されるため、これを使用して、シップされた定義の **TERMINAL** (または **CONNECTION**) 属性を**別名**として再設定することができます。これにより、アプリケーション占有領域 (AOR) に既にインストールされているリモート端末、ローカル端末、セッション、および接続の名前との競合を避けることができます。

**REMOTENAME** 属性を再設定する必要はありません。これは、TOR での端末の識別名が設定されたままにします。自動インストール・モデル名は、シップされた定義には適用されません。

自動インストール制御プログラムが、ローカルの端末名と競合する端末名を選択した場合、要求は拒否されます。自動インストール制御プログラムがもう一度起動されることはありません。

リモート定義に対して別名を使用する方法については、[端末のローカル名とリモート名を参照してください](#)。

**注：**自動インストール制御プログラムは、CICS クライアントで使用されるシップされた仮想端末定義を含め、すべてのシップされた端末と接続に対して起動されます。

### CICS が生成する別名

自動インストール制御プログラムは、シップされた端末または接続の定義をインストールするたびに呼び出されます。CICS は、シップされた定義の名前が、既にアプリケーション所有領域 (AOR) にインストールされているリモート端末、ローカル端末、セッション、または接続と競合することを検出すると、別名 **TERMINID** を生成し、それを通信域のフィールド **SELECTED\_SHIPPED\_TERMINID** で制御プログラムに渡します。

名前が競合しないことを検出した場合は、CICS は、TOR での端末または接続の識別名、つまり、シップされた定義の **TERMINAL** または **CONNECTION** 属性の値を **SELECTED\_SHIPPED\_TERMINID** で渡します。

制御プログラムは渡された **TERMINID** を受け入れることも変更することもできます。また、シップされた定義のインストールを拒否することもできます。

CICS が生成する別名は 1 文字の接頭部と 3 文字の接尾部で構成されます。接頭部は常に「{」です。接尾部には、値「AAA」から「999」を指定できます。つまり、接尾部の各文字は値「A」から「Z」または「0」から「9」になります。CICS によって生成される最初の接尾部の値は「AAA」です。その後「AAB」、「AAC」..「AAZ」、「AA0」、「AA1」というように続いて、最後が「999」になります。

別名を作成する必要があるたびに、CICS は使用中として記録されていない 3 文字の接尾部を生成します。CICS が生成した **TERMINID** を自動インストール制御プログラムでオーバーライドした場合、CICS はその接尾部を使用中として記録せずに、次の別名にその同じ接尾部を渡します。

### 端末 ID の再設定

自動インストール・プログラムを作成する場合は、制御プログラムで別名の **TERMINID** を割り振るために使用するアルゴリズムを検討する必要があります。

CICS のタイムアウト削除メカニズムによって削除された定義が、再びシップされてインストールされた場合の結果を検討する必要があります。自動インストール・プログラムで前と同じ **TERMINID** を割り振る必要があるか (これは、TOR での端末の識別名を AOR で割り振られた別名にファイルでマッピングすること意味します)、それとも別の **TERMINID** を割り振ることが許容されるかを判断する必要があります。後者の場合は、CICS で生成されるデフォルトの別名を使用できます。この判断は、いくつかの要因に基づいて行います。以下に例を挙げます。

- アプリケーション・プログラムによる一時記憶域キュー名の割り振り方法。一時記憶域キュー名を (特定のエンド・ユーザーとキューを関連付けるために) **TERMINID** から派生させている場合は、トランザクション終了で (障害などが原因で) キューが空にされず、**TERMINID** が同じ端末に一貫して割り振られなかった場合に、データの不一致の問題が発生する可能性があります。

最良の解決策は、アプリケーション・プログラムで一時記憶域キューを作成する前に、同じ名前のキューが存在するかどうかを必ず確認して、存在する場合は削除することです。これにより、自動インストール・プログラムで **TERMINID** を一貫して割り振る必要がなくなります。

ただし、アプリケーション・プログラムにまだこのチェックを実装していない場合は、すべてを修正できない可能性があります。その場合は、前述のように自動インストール・プログラムでマッピング・ファイルを使用する必要がある可能性があります。

- アプリケーション・プログラムで TERMID を後で使用するために記録しているどうか。例えば、アプリケーションで EXEC CICS START TERMID コマンドを発行してから、時間間隔を空けて、指定した端末に対してトランザクションを開始する場合があります。遅延期間中に端末定義が削除され、再シップされて別のローカル TERMID で再インストールされた場合、該当する TERMID が存在しないため、開始されたトランザクションは失敗します。

アプリケーション・プログラムがこのように TERMID を記録する場合は、自動インストール・プログラムでマッピング・ファイルを使用する必要がある可能性があります。

#### 例

この例では、AOR が、同じ端末 ID のセットを使用する 2 つの端末専有領域から端末 ID を解決する方法を示します。

2 つの端末専有領域の TORA と TORB があり、それらが同じ端末 ID のセット、T001 から T500 を使用していると想定します。TORA と TORB は、トランザクションを同じアプリケーション専有領域である AOR1 にルーティングします。端末が AOR1 にシップされる際に命名の競合を避けるために、AOR1 の制御プログラムで次のようにすることができます。

- TORA によって割り振られた TERMID を受け入れる。つまり、リモート定義の TERMINAL 属性を REMOTENAME 属性と同じままにします。
- TORB によって割り振られた TERMID の別名を作成する。つまり、説明にあるとおりに、マッピング・ファイルを使用して、リモート定義の TERMINAL 属性をリセットします。例えば、T001 から T500 の TERMID は、A001 から A500 の別名にマッピングできます。

この方法により、同じ TERMID のセットを使用する 2 つの TOR が同じ AOR にアクセスできるようになります。ただし、AOR で作成された別名は TOR の TERMID に一貫してマッピングされますが、この方法によって、端末が再シップされたときにデータの不一致の問題が起こらないことが保証されるわけではありません。これは、この方法が TERMID が TOR 内で一貫して割り振られること、つまり特定の TERMID が常に同じ物理デバイスに割り当てられることを利用しているためです。

注：ご使用の制御プログラムでは、各端末に含まれている関連 ID と接続定義を使用して、定義が TOR に再インストールされているか確認できます。INSTALL\_SHIPPED\_CORRID\_PTR パラメーターの説明については、177 ページの『シップされた端末の INSTALL 時の通信域』を参照してください。

AOR の端末別名を端末のネット名にマップするほうが、解決法として適している場合があります。この方法では少なくとも、特定の別名が常に同じ物理デバイスに関連付けられることが保証されます。ただし、別名が作成されていない TERMID については、TOR で一貫して割り振る必要があります。

## INSTALL 時の自動インストール制御プログラム

自動インストール制御プログラムは、次の INSTALL 時に呼び出されます。

- ローカル SNA LU
- MVS コンソール
- CINIT によって開始されたローカル APPC 単一セッション接続
- BIND によって開始されたローカル APPC 並列セッション接続
- BIND によって開始されたローカル APPC 単一セッション接続
- クライアント仮想端末
- リモートにシップされた端末と接続。シップされたクライアント仮想端末の定義も含まれます。

CICS は、呼び出しごとに、DFHEICAP によってアドレッシングされる通信域を経由して制御プログラムにパラメーター・リストを渡します。CINIT によって開始されたローカル端末と APPC 単一セッション接続の INSTALL 時に渡されるパラメーター・リストについては、139 ページの『INSTALL 実行時の端末の通信域』で説明します。MVS コンソールの INSTALL 時に渡されるパラメーター・リストについては、157 ページの『INSTALL 時の自動インストール制御プログラム』を参照してください。BIND 要求によって開始するローカル APPC 接続の INSTALL 時に渡されるパラメーター・リストについては、164 ページの『APPC 接続の

『[INSTALL 時の通信域](#)』を参照してください。クライアント 仮想端末の INSTALL 時に渡されるパラメーター・リストについては、183 ページの『[クライアント 仮想端末の INSTALL 時の通信域](#)』を参照してください。このセクションでは、シップされた端末と接続の INSTALL についてのみ説明します。

## シップされた端末の INSTALL 時の通信域

通信域は、CICSTS56.CICS.SDFHMAC に用意されているアセンブラー・バージョンの DFHZATDX の DSECT によってマップされます。

```

*-----*
* Remote install parameter list - Shipped definition functions 7 & 8      *
*-----*
INSTALL_SHIPPED_COMMAREA          DSECT          Install Parameter List
*
INSTALL_SHIPPED_STANDARD          DS  F           Standard field
                                ORG INSTALL_SHIPPED_STANDARD
INSTALL_SHIPPED_EXIT_FUNCTION     DS  XL1         Install type
INSTALL_SHIPPED_TERM              EQU X'F7'       Install terminal
INSTALL_SHIPPED_RSE               EQU X'F8'       Install remote system entry
INSTALL_SHIPPED_EXIT_COMPONENT    DS  CL2         Component ID 'ZC'
INSTALL_SHIPPED_CLASH             DS  CL1         Install clash Y/N
                                ORG ,
INSTALL_SHIPPED_NETNAME_PTR       DS  A           Pointer to netname
INSTALL_SHIPPED_SELECTED_PTR      DS  A           Pointer to return fields
INSTALL_SHIPPED_TERMID_PTR        DS  A           Pointer to incoming TERMID
INSTALL_SHIPPED_APPLID_PTR        DS  A           Pointer to applid of TOR
INSTALL_SHIPPED_SYSID_PTR         DS  A           Pointer to sysid
INSTALL_SHIPPED_CORRID_PTR        DS  A           Pointer to correlation ID
INSTALL_SHIPPED_SELECTED_PARMS    DSECT ,
                                DS  CL8           Reserved
SELECTED_SHIPPED_TERMID           DS  CL4         Selected TERMID
SELECTED_SHIPPED_RETURN_CODE      DS  CL1         Selected return code
RETURN_OK                         EQU X'00'       Accept request
REJECT                           EQU X'01'       Reject request
*

```

図 49. INSTALL 時の自動インストール制御プログラムの通信域

## INSTALL\_SHIPPED\_STANDARD

次の情報が含まれたフルワード入力フィールド。

### INSTALL\_SHIPPED\_EXIT\_FUNCTION

インストールされるリソースのタイプを示す 1 バイトのフィールド。リモートの端末および接続のインストールの場合、指定される値は次のとおりです。

#### INSTALL\_SHIPPED\_TERM (X'F7')

シップされた端末

#### INSTALL\_SHIPPED\_RSE (X'F8')

シップされた接続 (リモート・システム・エンタリー)。

### INSTALL\_SHIPPED\_EXIT\_COMPONENT

2 バイトのコンポーネント・コード。「ZC」に設定されます。

## INSTALL\_SHIPPED\_CLASH

シップされた定義の TERMID が AOR で既に使用されているかどうかを示す 1 文字の入力フィールド。

**Y**

TOR での端末または接続の識別名 (シップされた定義の TERMINAL または CONNECTION 属性の値) は、インストール済みのリモートの端末または接続を識別するために既に AOR で使用されています。

**N**

TOR での端末または接続の識別名は、リモートの端末または接続を識別するために AOR で使用されていません。

## INSTALLED\_SHIPPED\_NETNAME\_PTR

インストールする端末または接続のネット名が含まれている 8 文字の入力フィールドを指すフルワード・ポインター。



## **INSTALL\_SHIPPED\_SELECTED\_PTR**

戻りフィールドを指すフルワード・ポインター。プログラムで使用する出力フィールドは次のとおりです。

## **SELECTED\_SHIPPED\_TERMID**

このシステムでリモートの端末または接続を識別するために使用する名前を指定する 4 文字のフィールド。名前が 4 文字より短い場合は、末尾ブランクを埋め込む必要があります。端末名に使用できる文字のリストについては、[TERMINAL 属性](#)を参照してください。

呼び出しのときに INSTALL\_SHIPPED\_CLASH が「N」(端末名の競合がないことを示す)に設定されていた場合、SELECTED\_SHIPPED\_TERMID には、INSTALL\_SHIPPED\_TERMID\_PTR が指すフィールドと同じ値 (シッパされた定義の TERMINAL または CONNECTION 属性の値) が入っています。INSTALL\_SHIPPED\_CLASH が「Y」に設定されている場合、SELECTED\_SHIPPED\_TERMID には CICS が生成した別名が入っています。

CICS が生成した別名は、ユーザー・プログラムでこのフィールドを使用してオーバーライドできます。端末名と接続名の選択についてのアドバイスを、[175 ページの『端末 ID の再設定』](#)に記載しています。

## **SELECTED\_SHIPPED\_RETURN\_CODE**

1 文字の戻りコード・フィールド。同等の値は以下のとおりです。

### **RETURN\_OK (X'00')**

リモートの端末または接続をインストールします。リソースを自動インストールする場合、ユーザー・プログラムからこの値が返される必要があります。

### **REJECT (X'01')**

リモートの端末または接続をインストールしません。これはデフォルト値です。

## **INSTALL\_SHIPPED\_TERMID\_PTR**

TOR での端末または接続の識別名が含まれている 4 文字の入力フィールドを指すフルワード・ポインター (これはシッパされた定義の TERMINAL または CONNECTION 属性の値です)。

## **INSTALL\_SHIPPED\_APPLID\_PTR**

TOR のネット名 (applid) が含まれている 8 文字の入力フィールドを指すフルワード・ポインター。

## **INSTALL\_SHIPPED\_SYSID\_PTR**

TOR への接続の名前 (sysid) が含まれている 4 文字の入力フィールドを指すフルワード・ポインター。

## **INSTALL\_SHIPPED\_CORRID\_PTR**

シッパされた定義の相関識別子が含まれている 8 文字の入力フィールドを指すフルワード・ポインター。相関識別子は、端末または接続の定義がインストールされるときに作成される「インスタンス・トークン」であり、定義内に保管されます。したがって、その定義が別の領域にシッパされると、そのトークンの値も一緒にシッパされます。CICS は接続処理中にこの相関 ID を使用して、AOR にシッパされた既存の定義が最新かどうか、または、端末が TOR に再インストールされたために、シッパされた既存の定義を削除して再シッパする必要があるかどうかを検査します。インスタンス・トークンについて詳しくは、[シッパされた端末定義の効率的な削除](#)を参照してください。

制御プログラムで相関 ID を記録して、TOR で割り当てられた TERMID を AOR で割り当てる別名にマップすると、端末が TOR に再インストールされたかどうかを検査できます。端末が再インストールされていた場合は、TOR で割り当てられた TERMID が、その TERMID で最後にインストールされた物理デバイスとは異なるデバイスに関連付けられている可能性があります。

## **DELETE 時の自動インストール制御プログラム**

自動インストール制御プログラムは、自動インストールされたリソースが削除されるときに再度呼び出されます。DELETE 時にユーザー・プログラムを呼び出すことで、INSTALL 時に実行したプロセスを取り消すことができます。

自動インストールできるリソースのリストを [176 ページの『INSTALL 時の自動インストール制御プログラム』](#)に示します。

ローカル端末の DELETE 時にユーザー・プログラムに渡されるパラメーター・リストについては、[145 ページの『DELETE 時の自動インストール制御プログラム』](#)で説明します。ローカル APPC 接続の DELETE 時に渡されるパラメーター・リストについては、[166 ページの『DELETE 時の自動インストール制御プログラム』](#)で説明します。クライアント仮想端末の DELETE 時に渡されるパラメーター・リストについては、[186](#)



ページの『DELETE 時の自動インストール制御プログラム』で説明します。このセクションでは、シップされた端末と接続の DELETE についてのみ説明します。

シップされた端末および接続の定義は、タイムアウト削除メカニズムによって削除されます。タイムアウト削除メカニズムについて詳しくは、[シップされた端末定義の効率的な削除](#)を参照してください。

179 ページの図 50 は、DELETE 時に自動インストール・ユーザー・プログラムに渡される通信域を示しています。

DELETE_SHIPPED_COMMAREA	DSECT ,	Delete parameter list
DELETE_SHIPPED_STANDARD	DS F	Standard field
DELETE_SHIPPED_EXIT_FUNCTION	DS XL1	Delete type
DELETE_SHIPPED_TERM	EQU X'FA'	Delete terminal
DELETE_SHIPPED_RSE	EQU X'FB'	Delete remote system entry
DELETE_SHIPPED_EXIT_COMPONENT	DS CL2	Component ID 'ZC'
	DS CL1	Reserved
DELETE_SHIPPED_TERMID	DS CL4	TERMID in TOR
DELETE_SHIPPED_APPLID	DS CL8	Applid of TOR
DELETE_SHIPPED_LTERMID	DS CL4	TERMID in AOR
DELETE_SHIPPED_NETNAME	DS CL8	Netname of terminal

図 50. DELETE 時の自動インストール制御プログラムの通信域

DELETE 時は、通信域のすべてのフィールドが入力専用です。リストしていないフィールドについては、INSTALL で説明しています。

#### DELETE\_SHIPPED\_EXIT\_FUNCTION

削除されるリソースのタイプを示す 1 バイトのフィールド。同等の値は以下のとおりです。

##### DELETE\_SHIPPED\_TERM (X'FA')

シップされた端末

##### DELETE\_SHIPPED\_RSE (X'FB')

シップされた接続 (リモート・システム・エントリー)。

#### DELETE\_SHIPPED\_TERMID

TOR での端末または接続の識別子 (TERMID) が含まれている 4 文字のフィールド。

#### DELETE\_SHIPPED\_APPLID

TOR のネット名 (applid) が含まれている 8 文字のフィールド。

#### DELETE\_SHIPPED\_LTERMID

AOR での端末または接続の識別名が含まれている 4 文字のフィールド。これは、AOR で別名が使用されたかどうかに応じて、DELETE\_SHIPPED\_TERMID と同じである場合も、異なる場合があります。

#### DELETE\_SHIPPED\_NETNAME

削除される端末のネット名が含まれている 8 文字のフィールド。

## サンプル・プログラムのデフォルトのアクション

シップされた端末または接続の INSTALL 時に DFHZATDX または DFHZATDY が呼び出されると、以下を実行します。

1. 必要に応じて SELECTED\_SHIPPED\_TERMID フィールドを更新し、TOR での端末または接続の識別名を指定します。

注：

- a. CICS が現在インストール済みのリモート TERMID との競合を検出した場合、サンプル・プログラムを呼び出すときの SELECTED\_SHIPPED\_TERMID には、CICS が生成した別名が入っています。このサンプル・プログラムは、この値を上書きします。
- b. CICS が現在インストール済みのリモート TERMID との競合を検出なかった場合、サンプル・プログラムを呼び出すときの SELECTED\_SHIPPED\_TERMID には、シップされた定義の TERMINAL 属性の値 (INSTALL\_SHIPPED\_TERMID\_PTR が指す値) が入っています。このサンプル・プログラムは、この値を受け入れます。

2. 戻りコード・フィールドに RETURN\_OK を設定して戻り、リモート定義のインストールを許可します。

シップされた端末または接続の DELETE 時に DFHZATDX または DFHZATDY が呼び出された場合は、アクションを実行せずに戻ります。

## 仮想端末の自動インストールを制御するプログラムの作成

仮想端末のインストールを制御するプログラムを作成できます。仮想端末は、外部表示インターフェース (EPI) および CICS クライアントと CICS Link3270 ブリッジの端末エミュレーター機能で 사용됩니다。付属の自動インストール制御プログラム DFHZATDX と DFHZATDY はどちらも、仮想端末の定義をインストールする機能を備えています。DFHZATDX または DFHZATDY のいずれかを基に、カスタマイズした制御プログラムを作成できます。

ブリッジ環境では、仮想端末 (別名: ブリッジ・ファシリティ) は、3270 トランザクションの基本ファシリティである実際の 3270 に置き換わるものです。

## クライアント仮想端末を自動インストールする方法

クライアント仮想端末は、リモートの 3270 データ・ストリーム・デバイスとして CICS に定義されます。

### 自動インストール・モデル

自動インストール制御プログラムは、別の自動インストール・モデルを選択することはできません。

仮想端末のインストールに使用される自動インストール・モデルは、以下の手順を使用して判別できます。

1. EPI プログラムの場合: CICS\_EpiAddTerminal 機能の **DevType** パラメーターから (クライアント EPI プログラムによって指定されている場合)。

クライアント 端末エミュレーターの場合: エミュレーターの開始に使用する **cicsterm** コマンドの **/m** パラメーターから (ワークステーション・ユーザーによって指定されている場合)。

注: クライアントによって指定された自動インストール・モデルはすべて、CICS で定義されている必要があります。ただし、z/OS Communications Server 定義はクライアントの仮想端末には必要ないため、z/OS Communications Server LOGMODE テーブルに、対応する項目を作成する必要はありません。

2. CICS 提供の自動インストール・モデル、DFHLU2。

### 端末 ID

仮想端末のインストール時に CICS 自動インストール機能に渡される端末 ID (TERMID) は、以下の手順で判別できます。

1. **EPI プログラムの場合: CICS\_EpiAddTerminal** 機能の **NetName** パラメーターから (Client EPI プログラムによって指定されている場合)。

クライアント 端末エミュレーターの場合: エミュレーターの開始に使用する **cicsterm** コマンドの **/n** パラメーターから (ワークステーション・ユーザーによって指定されている場合)。

2. CICS によって自動生成された名前。

CICS によって生成されたクライアント 端末用の TERMID は、1 文字の接頭部と 3 文字の接尾部で構成されます。デフォルトの接頭部は '¥' ですが、VTPREFIX システム 初期設定パラメーターを使用して、別の接頭部を指定することができます。接尾部には、値「AAA」から「999」を指定できます。つまり、接尾部の各文字には、値「A」から「Z」または「0」から「9」を指定できます。CICS によって生成される最初の接尾部の値は「AAA」です。その後に「AAB」、「AAC」..「AAZ」、「AA0」、「AA1」というように続いて、最後が「999」になります。

クライアントの仮想端末が自動インストールされるたびに、CICS は使用中であることが記録されていない 3 文字の接尾部を生成します。

注: 接頭部を指定することで、このシステムに自動インストールされたクライアント 端末の TERMID が、トランザクション・ルーティング・ネットワーク上で固有であることを保証できます。これにより、2 つ以上の領域が仮想端末の定義を同じアプリケーション占有領域 (AOR) にシップする場合に発生する可能性のある競合を防ぐことができます。

簡単に言うと、クライアントに指定された名前または生成された **VTPREFIX** 名が、提供名です。クライアントは、仮想端末を常に提供名で認識します。ただし、自動インストール制御プログラムが別名を割り振ることができ、仮想端末は別名によって CICS に認識されます。

CICS 自動インストール機能は、この領域にインストール済みのリモート端末または接続の名前と提供名が競合することを検出した場合、別名 **TERMID** を生成します。CICS は、仮想端末の別名 **TERMID** を、シップされた端末の別名を生成するのと同じ方法で生成します。詳しくは [175 ページの『CICS が生成する別名』](#) を参照してください。

**注：**提供名が、ローカルの端末または接続の名前と競合する場合、仮想端末のインストールは拒否され、自動インストール制御プログラムは起動されません。

自動インストール制御プログラムは、仮想端末定義がインストールされるたびに 1 回ずつ起動されます。起動されると、通信域のフィールド **INSTALL\_SHIPPED\_TERMID\_PTR** が提供された **TERMID** を指します。フィールド **SELECTED\_SHIPPED\_TERMID** には、名前の競合が検出されたかどうかによって、提供された **TERMID** あるいは生成された別名が含まれます。

制御プログラムは、**SELECTED\_SHIPPED\_TERMID** で渡される **TERMID** を受け入れることも、変更することも、仮想端末のインストールを拒否することもできます。

### TERMID をオーバーライドする理由

提供された **TERMID** の別名を作成 (または、名前が競合する場合は CICS によって生成される別名をオーバーライド) する必要があるのはなぜでしょうか。その必要がない場合もあり、必要があるかどうかは、サーバー・プログラムの作成方法によって決まります。「サーバー・プログラム」とは、クライアント **EPI** プログラムによって開始されるトランザクション・プログラムと、クライアント端末エミュレーターから開始されるトランザクション・プログラムの両方を意味します。

### CICS が生成した TERMID のオーバーライド

CICS が生成した **TERMID** を使用していて、クライアント端末をインストールできる各領域に、仮想端末用に予約済みの別の接頭部を指定した場合、仮想端末がインストールされている領域にも、異なる領域が同じ **AOR** にクライアント定義をシップした場合にも、名前の競合は起こらないはずですが。

ただし、CICS が生成した **TERMID** を使用している場合、サーバー・プログラムが、**TERMID** が特定のクライアント端末に一貫して割り振られることを利用してはなりません。

クライアント端末は、**CICS\_EpiDelTerminal** 要求を送信するクライアントによって、またはクライアント端末エミュレーターまたはクライアント自体をシャットダウンするユーザーによって、あるいは接続の失敗が発生した場合に削除される場合があります。再インストールされた場合、CICS は必ずしも前と同じ **TERMID** を生成するとはかぎりません。このため、次のような場合に問題が発生する可能性があります。

- サーバー・プログラムが、(各キューを特定のユーザーと関連付けるために) **TERMID** から一時記憶域キュー名を派生させる場合。(通常は失敗が原因で) キューがトランザクションの終了時に削除されない場合、データ不一致の問題が発生する可能性があります。

最良の解決策は、アプリケーション・プログラムで一時記憶域キューを作成する前に、同じ名前のキューが存在するかどうかを必ず確認して、存在する場合は削除することです。ただし、サーバー・アプリケーションの数が多く、それらすべてを確認して変更することは不可能である場合もあります。

- サーバー・プログラムが、将来使用するために **TERMID** を記録する場合。例えば、アプリケーションで **EXEC CICS START TERMID** コマンドを発行してから、時間間隔を空けて、指定した端末に対してトランザクションを開始する場合があります。この遅延間隔中に仮想端末が削除され、別の **TERMID** で再インストールされた場合、開始されたトランザクションは **TERMID** が存在しないため失敗する可能性があります。

サーバー・プログラムを再作成できない場合、自動インストール制御プログラムで、CICS によって生成された **TERMID** の別名を作成する必要が生じることがあります。例えば、マッピング・ファイルを使用して特定の別名を特定のクライアント・ワークステーション (接続名で識別される) に関連付けることができます。

サーバー・プログラムがバックエンド **AOR** にある場合、仮想端末がシップされると、シップされる他の定義の場合と同じように、自動インストール制御プログラムが **AOR** 内で呼び出されます。必要な場合は、別名 **TERMID** をシップされた定義に割り振ることもできます。(シップされた定義をインストールするための制御プログラムを作成する方法について詳しくは、[174 ページの『シップされた端末の自動インストールを制御するプログラムの作成』](#) を参照してください。)

### クライアントによって指定された **TERMID** のオーバーライド

TERMID がクライアント EPI プログラムによって常に一貫性のある方法で指定される場合、TERMID を記録するサーバー・プログラムによる データ不一致の問題は発生しないはずです。

ただし、クライアントによって指定される TERMID は、クライアント以外のリモートの TERMID と競合する可能性があります。あるいは、複数のクライアントが同じ CICS システムに、また相互に接続されている場合も同様です。これが CTIN トランザクションが実行される領域で起こる場合、一貫性を保つために、自動インストール制御プログラムは CICS によって提供される別名に依存するのではなく、別名 TERMID を割り振ることが必要になる場合があります。(つまり、前述のように、特定の TERMID を 特定のクライアント・ワークステーションに関連付けることが必要な場合があります。)

名前の競合が AOR で発生する場合、自動インストール制御プログラムが AOR で呼び出されます。別名端末 ID をシッパされた定義に割り振ることで、競合を解決することができます。

## ブリッジ・ファシリティー仮想端末はどのように自動インストールされるか

ブリッジ・ファシリティー仮想端末は LU2 デバイスとして定義されます。これらのデバイスは、ブリッジ環境での CICS 3270 アプリケーションの実行をサポートするために 3270 ブリッジ・メカニズムによって作成されます。ブリッジ環境では、すべての端末相互作用がインターセプトされて 3270 ブリッジ・メカニズムに渡されます。

3270 ブリッジ・メカニズムはモデル 3270 端末定義 (*facilitylike*) を使用してブリッジ・ファシリティーを構築し、それを識別するための 8 バイトのトークンと、TERMID と NETNAME の両方に使用される 4 文字の端末 ID を作成します。

START BREXIT コマンドで作成されたブリッジ・ファシリティーの場合、トークンと名前はブリッジ・ファシリティーが作成される領域内で固有であり、TERMID は }AAA の形式になります (AAA は逐次昇順アルファベット)。

link3270 ブリッジによって作成されたブリッジ・ファシリティーの場合、トークンと名前は CICSplex 全体で固有であり、TERMID は AAA} の形式になります。共用ファイルを使用して名前の割り振りを管理することによって、固有性が確保されます。

ブリッジ・ファシリティーの端末名とネット名は通常、ブリッジ・メカニズムによって動的に割り振られますが、**AIBRIDGE** システム初期設定パラメーターが YES に設定されている場合は、端末自動インストール制御プログラムが呼び出されるので、これを使用してインストール固有の名前を割り当てることができます。

### 端末の自動インストール制御プログラムをブリッジ・ファシリティーで使用する

AIBRIDGE (YES) を指定する場合、ブリッジ・ファシリティーが割り振られるか削除されると、自動インストール制御プログラムが呼び出されます。

自動インストール制御プログラムには、パラメーター・リスト (通信域) が渡されます (185 ページの『ブリッジ・ファシリティー仮想端末の INSTALL での通信域』および 187 ページの『DELETE 時のブリッジ・ファシリティー仮想端末の通信域』で説明)。これは、プログラムが Link3270 で呼び出されたか、START ブリッジ・ファシリティーで呼び出されたかを示します。

インストール済み環境に固有の端末名は、次の 2 つの方法のいずれかで割り振ることができます。

- 名前は、クライアント・プログラムによって定義でき、最初の Link3270 要求に渡されます。その後、自動インストール制御プログラムは、これらの名前を許可、変更、または拒否できます。
- 名前は、自動インストール制御プログラムによって定義できます。

クライアント・プログラムによって提案された名前は、通信域に渡されます。また自動インストール制御プログラムは、EXEC CICS ASSIGN USERID を使用して USERID を取得し、これを使用して提案された TERMID および NETNAME を検証することができます。

クライアントによって定義された TERMID フィールドおよび NETNAME フィールドは、インストール済み環境に固有のデータを自動インストール制御プログラムに渡して、必要な名前を生成するために使用することもできます。

### START ブリッジ・ファシリティーの自動インストール

通信域に含まれる情報とは別に、以下の情報を取得できます。



- 疑似会話型の最初のトランザクションの USERID を **EXEC CICS ASSIGN USERID** を使用して取得できます。
- 疑似会話型の最初のトランザクションの TRANSID を EIBTRNID から取得できます。

自動インストール制御プログラムは、USERID および TRANSID の値を使用して、新しい TERMID および NETNAME 値を派生させ、それらを通信域に戻すことができます。

### Link3270 ブリッジ・ファシリティの自動インストール

#### ブリッジ・ファシリティの名前の固有性

アプリケーションの中には、端末 ID を使用して、固有のリソースを割り振るものがあります。これは、名前が CICSplex 内で固有であることを利用しています。ブリッジ・ファシリティ名は、端末 ID と同じ名前空間を持ちます。ただし、CICS は、自動インストール制御プログラムによって戻されるブリッジ・ファシリティ名が CICSplex 内の任意の場所にある端末 ID と同じではないことを確認することはできません。自動インストール制御プログラムによって戻される端末 ID もネット名も検証されません。

## INSTALL 時の自動インストール制御プログラム

自動インストール制御プログラムは、次の INSTALL 時に呼び出されます。

- ローカル SNA LU
- MVS コンソール
- CINIT によって開始されたローカル APPC 単一セッション接続
- BIND によって開始されたローカル APPC 並列セッション接続
- BIND によって開始されたローカル APPC 単一セッション接続
- クライアント 仮想端末
- ブリッジ・ファシリティ 仮想端末
- リモートにシップされた端末と接続 (シップされたクライアント 仮想端末の定義も含む)。

CICS は、呼び出しごとに、DFHEICAP によってアドレッシングされる通信域を経由して制御プログラムにパラメーター・リストを渡します。CINIT によって開始されたローカル端末と APPC 単一セッション接続の INSTALL 時に渡されるパラメーター・リストについては、139 ページの『[INSTALL 実行時の端末の通信域](#)』で説明します。BIND 要求によって開始するローカル APPC 接続の INSTALL 時に渡されるパラメーター・リストについては、164 ページの『[APPC 接続の INSTALL 時の通信域](#)』を参照してください。MVS コンソールの INSTALL 時に渡されるパラメーター・リストについては、157 ページの『[INSTALL 時の自動インストール制御プログラム](#)』を参照してください。シップされた端末と接続の INSTALL 時に渡されるパラメーター・リストについては、177 ページの『[シップされた端末の INSTALL 時の通信域](#)』を参照してください。このセクションでは、クライアント 仮想端末の INSTALL 時に渡されるパラメーター (183 ページの『[クライアント 仮想端末の INSTALL 時の通信域](#)』) およびブリッジ・ファシリティの INSTALL 時に渡されるパラメーター (185 ページの『[ブリッジ・ファシリティ 仮想端末の INSTALL 時の通信域](#)』) についてのみ説明します。

#### クライアント 仮想端末の INSTALL 時の通信域

通信域は、CICSTS56.CICS.SDFHMAC に用意されているアセンブラー・バージョンの DFHZATDX または DFHZATDY の DSECT によってマップされます。

注：仮想端末の INSTALL のための通信域は、シップされた端末および接続の INSTALL のための通信域と同じです。したがって、フィールド名には「SHIPPED」という語が含まれています。

```

*-----*
* Remote install parameter list - Client virtual terminal function 9 *
*-----*
INSTALL_SHIPPED_COMMAREA          DSECT          Install Parameter List
*
INSTALL_SHIPPED_STANDARD          DS  F            Standard field
                                ORG INSTALL_SHIPPED_STANDARD
INSTALL_SHIPPED_EXIT_FUNCTION     DS  XL1           Install type
INSTALL_SHIPPED_TERM              EQU X'F9'         Install virtual terminal
INSTALL_SHIPPED_EXIT_COMPONENT    DS  CL2           Component ID 'ZC'
INSTALL_SHIPPED_CLASH             DS  CL1           Install clash Y/N
                                ORG ,
INSTALL_SHIPPED_NETNAME_PTR       DS  A            Pointer to netname of Client
INSTALL_SHIPPED_SELECTED_PTR      DS  A            Pointer to return fields
INSTALL_SHIPPED_TERMID_PTR        DS  A            Pointer to incoming TERMID
INSTALL_SHIPPED_APPLID_PTR        DS  A            Pointer to applid of Client
INSTALL_SHIPPED_SYSID_PTR         DS  A            Pointer to sysid of Client
INSTALL_SHIPPED_CORRID_PTR        DS  A            Pointer to correlation ID
INSTALL_SHIPPED_SELECTED_PARMs    DSECT ,
                                DS  CL8            Reserved
SELECTED_SHIPPED_TERMID           DS  CL4           Selected TERMID
                                DS  CL4            Reserved
                                DS  CL4            Reserved
                                DS  CL4            Reserved
SELECTED_SHIPPED_RETURN_CODE      DS  CL1           Selected return code
RETURN_OK                         EQU X'00'         Accept request
REJECT                           EQU X'01'         Reject request
*

```

図 51. INSTALL 時の自動インストール制御プログラムの通信域

## INSTALL\_SHIPPED\_STANDARD

次の情報が含まれたフルワード入力フィールド。

### INSTALL\_SHIPPED\_EXIT\_FUNCTION

インストールされるリソースのタイプを示す 1 バイトのフィールド。クライアント仮想端末のインストールの場合、指定される値は INSTALL\_SHIPPED\_TERM (X'F9') です。

### INSTALL\_SHIPPED\_EXIT\_COMPONENT

2 バイトのコンポーネント・コード。「ZC」に設定されます。

## INSTALL\_SHIPPED\_CLASH

指定された TERMID がこの領域で既に使用されているかどうかを示す 1 文字の入力フィールド。

**Y**

CICS 自動インストール機能に渡された名前は、インストールされているリモートの端末または接続を識別するために既にこの領域で使用されています。

**N**

CICS 自動インストール機能に渡された名前は、リモートの端末または接続を識別するために、この領域で使用されていません。

## INSTALL\_SHIPPED\_NETNAME\_PTR

クライアント・ワークステーションのネット名を含む 8 文字のフィールドに対するフルワード・ポインター。このフィールドには、INSTALL\_SHIPPED\_APPLID\_PTR が指すフィールドと同じ値が含まれています。

## INSTALL\_SHIPPED\_SELECTED\_PTR

戻りフィールドを指すフルワード・ポインター。プログラムで使用する出力フィールドは次のとおりです。

### SELECTED\_SHIPPED\_TERMID

CICS がこの仮想端末を識別するために使用する名前を指定する 4 文字のフィールド。名前が 4 文字より短い場合は、末尾ブランクを埋め込む必要があります。端末名に使用できる文字のリストについては、[TERMINAL 属性](#)を参照してください。

呼び出しのときに INSTALL\_SHIPPED\_CLASH が「N」(端末名の競合がないことを示す)に設定されていた場合、SELECTED\_SHIPPED\_TERMID には、INSTALL\_SHIPPED\_TERMID\_PTR が指すフィールドと同じ値(渡された名前)が入っています。INSTALL\_SHIPPED\_CLASH が「Y」に設定されている場合、SELECTED\_SHIPPED\_TERMID には CICS が生成した別名が入っています。

この提示された名前は、ユーザー・プログラムでオーバーライドすることができます。



## SELECTED\_SHIPPED\_RETURN\_CODE

1 文字の戻りコード・フィールド。同等の値は以下のとおりです。

### RETURN\_OK (X'00')

仮想端末をインストールします。これはデフォルト値です。リソースを自動インストールする場合、ユーザー・プログラムからこの値が返される必要があります。

### REJECT (X'01')

仮想端末をインストールしません。

## INSTALL\_SHIPPED\_TERMID\_PTR

CICS 自動インストール機能に渡された TERMID (つまり、提供された名前) が含まれている 4 文字の入力フィールドを指すフルワード・ポインター。

## INSTALL\_SHIPPED\_APPLID\_PTR

クライアント・ワークステーションのネット名 (applid) が含まれている 8 文字の入力フィールドを指すフルワード・ポインター。

## INSTALL\_SHIPPED\_SYSID\_PTR

クライアント・ワークステーションへの接続の名前 (sysid) が含まれている 4 文字の入力フィールドを指すフルワード・ポインター。

## INSTALL\_SHIPPED\_CORRID\_PTR

仮想端末のインストールに使用されない 8 文字の入力フィールドを指すフルワード・ポインター。

## ブリッジ・ファシリティ仮想端末の INSTALL での通信域

通信域は、CICSTS56.CICS.SDFHMAC に用意されているアセンブラー・バージョンの DFHZATDX または DFHZATDY の DSECT によってマップされます。

```
-----
* Install Bridge Facility                                - Function 15 & 17
*-----*
INSTALL_BRFAC_COMMAREA      DSECT      Install Parameter List
INSTALL_BRFAC_STANDARD      DS   F      Standard field
                              ORG INSTALL_BRFAC_STANDARD
INSTALL_BRFAC_EXIT_FUNCTION DS   XL1     Install type
INSTALL_LINK_BRFAC          EQU X'0F'   Install Link Brfacility
INSTALL_START_BRFAC         EQU X'11'   Install Start Brfacility
INSTALL_BRFAC_EXIT_COMPONENT DS   CL2    Component ID 'BR'
                              DS   CL1    Reserved
                              ORG ,
INSTALL_BRFAC_NETNAME_PTR   DS   A      Pointer to input netname
INSTALL_BRFAC_SELECTED_PTR  DS   A      Pointer to return fields
INSTALL_BRFAC_TERMID_PTR   DS   A      Pointer to input termid
                              DS   A      Reserved
                              DS   A      Reserved
                              DS   A      Reserved
INSTALL_BRFAC_SELECTED_PARMS DSECT ,
                              DS   CL8    Reserved
SELECTED_BRFAC_TERMID      DS   CL4     Selected termid
SELECTED_BRFAC_RETURN_CODE DS   B       Selected return
SELECTED_BRFAC_NETNAME     DS   CL8     Selected netname
*
*-----*
```

図 52. INSTALL 時の自動インストール制御プログラムの通信域

## INSTALL\_BRFAC\_STANDARD

次の情報が含まれたフルワード入力フィールド。

### INSTALL\_BRFAC\_EXIT\_FUNCTION

インストールされるリソースのタイプを示す 1 バイトのフィールド。ブリッジ・ファシリティ仮想端末のインストール用です。同等の値は以下のとおりです。

### INSTALL\_LINK\_BRFAC (X'0F')

link3270 ブリッジによって使用されるブリッジ・ファシリティのインストール時に自動インストール・プログラムが呼び出されました。

### INSTALL\_START\_BRFAC (X'11')

START ブリッジによって使用されるブリッジ・ファシリティのインストール時に自動インストール・プログラムが呼び出されました。

## **INSTALL\_BRFAC\_EXIT\_COMPONENT**

2 バイトのコンポーネント・コード。「BR」に設定されます。

## **INSTALL\_BRFAC\_NETNAME\_PTR**

ブリッジ・ファシリティのネット名を含む 8 文字フィールドへのフルワード・ポインター。これは、クライアントによって指定された値か、クライアントが BRIHNN-DEFAULT (デフォルト値) を指定した場合に CICS によって生成された値のどちらかです。

## **INSTALL\_BRFAC\_SELECTED\_PTR**

戻りフィールドを指すフルワード・ポインター。プログラムで使用する出力フィールドは次のとおりです。

### **SELECTED\_BRFAC\_TERMID**

CICS がこの仮想端末を識別するために使用する名前を指定する 4 文字のフィールド。名前が 4 文字より短い場合は、末尾ブランクを埋め込む必要があります。端末名に使用できる文字のリストについては、[TERMINAL](#) 属性を参照してください。INSTALL\_BRFAC\_TERMID\_PTR での名前をコピーするか、新しい値を設定することができます。

### **SELECTED\_BRFAC\_RETURN\_CODE**

1 文字の戻りコード・フィールド。同等の値は以下のとおりです。

#### **RETURN\_OK (X'00')**

仮想端末をインストールします。これはデフォルト値です。リソースを自動インストールする場合、ユーザー・プログラムからこの値が返される必要があります。

#### **REJECT (X'01')**

仮想端末をインストールしません。

## **SELECTED\_BRFAC\_NETNAME**

ブリッジ・ファシリティのネット名を指定するために使用される 8 文字フィールド。名前が 8 文字より短い場合は、末尾ブランクを埋め込む必要があります。INSTALL\_BRFAC\_NETNAME\_PTR での名前をコピーするか、新しい値を設定することができます。

## **INSTALL\_BRFAC\_TERMID\_PTR**

CICS 自動インストール機能に渡された TERMID (つまり、提供された名前) が含まれている 4 文字の入力フィールドを指すフルワード・ポインター。

## **DELETE 時の自動インストール制御プログラム**

自動インストール制御プログラムは、自動インストールされたリソースが削除されるときに再度呼び出されます。DELETE 時にユーザー・プログラムを呼び出すことで、INSTALL 時に実行したプロセスを取り消すことができます。

自動インストールできるリソースのリストを [183 ページの『INSTALL 時の自動インストール制御プログラム』](#) に示します。

ローカル端末の DELETE 時にユーザー・プログラムに渡されるパラメーター・リストについては、[145 ページの『DELETE 時の自動インストール制御プログラム』](#) で説明します。ローカル APPC 接続の DELETE 時に渡されるパラメーター・リストについては、[166 ページの『DELETE 時の自動インストール制御プログラム』](#) で説明します。シップされた定義の DELETE 時に渡されるパラメーター・リストについては、[178 ページの『DELETE 時の自動インストール制御プログラム』](#) で説明します。このセクションでは、クライアント仮想端末の DELETE ([187 ページの『クライアント仮想端末の DELETE 時の通信域』](#)) およびブリッジ・ファシリティ仮想端末の DELETE ([187 ページの『DELETE 時のブリッジ・ファシリティ仮想端末の通信域』](#)) について説明します。

シップされた端末および接続の定義は、CICS タイムアウト削除メカニズムによって削除されます。タイムアウト削除メカニズムについて詳しくは、[シップされた端末定義の効率的な削除](#)を参照してください。

## クライアント仮想端末の DELETE 時の通信域

DELETE 時に自動インストール・ユーザー・プログラムに渡される通信域。

DELETE_SHIPPED_COMMAREA	DSECT ,	Delete parameter list
DELETE_SHIPPED_STANDARD	DS F	Standard field
DELETE_SHIPPED_EXIT_FUNCTION	DS XL1	Delete type
DELETE_SHIPPED_TERM	EQU X'FC'	Delete virtual terminal
DELETE_SHIPPED_EXIT_COMPONENT	DS CL2	Component ID 'ZC'
	DS CL1	Reserved
DELETE_SHIPPED_TERMID	DS CL4	TERMID
DELETE_SHIPPED_APPLID	DS CL8	Applid of Client workstation
DELETE_SHIPPED_LTERMID	DS CL4	TERMID in this region
DELETE_SHIPPED_NETNAME	DS CL8	Netname of Client workstation

図 53. DELETE 時の自動インストール制御プログラムの通信域

DELETE 時は、通信域のすべてのフィールドが入力専用です。以下にリストしていないフィールドについては、INSTALL で説明しています。

### DELETE\_SHIPPED\_EXIT\_FUNCTION

削除されるリソースのタイプを示す 1 バイトのフィールド。クライアント仮想端末の場合、指定される値は DELETE\_SHIPPED\_TERM (X'FC') です。

注：値 X'F1' は、CINIT 要求を使用して自動インストールされたローカル端末または APPC 単一セッション・デバイスの削除を表します (145 ページの『DELETE 時の自動インストール制御プログラム』を参照)。値 X'F5' または X'F6' は、BIND 要求によりインストールされた APPC 接続の削除を表します (166 ページの『DELETE 時の自動インストール制御プログラム』を参照)。値 X'FA' または X'FB' は、シブされた端末または接続の削除を表します (179 ページの図 50 を参照)。

### DELETE\_SHIPPED\_TERMID

クライアントでの仮想端末の識別名が含まれている 4 文字のフィールド。

### DELETE\_SHIPPED\_APPLID

クライアント・ワークステーションのネット名 (applid) が含まれる 8 文字のフィールド。

### DELETE\_SHIPPED\_LTERMID

この領域での仮想端末の識別名が含まれている 4 文字のフィールド。これは、インストール時に別名が使用されたかどうかに応じて、DELETE\_SHIPPED\_TERMID の値と同じである場合も、異なる場合もあります。

### DELETE\_SHIPPED\_NETNAME

クライアント・ワークステーションのネット名が含まれている 8 文字のフィールド。このフィールドには DELETE\_SHIPPED\_APPLID と同じ値が含まれています。

## DELETE 時のブリッジ・ファシリティ仮想端末の通信域

DELETE 時に自動インストール・ユーザー・プログラムに渡される通信域。

```
*
*-----*
* Delete Bridge Facility - Function 16 *
*-----*
DELETE_BRFAC_COMMAREA DSECT , Delete parameter list
DELETE_BRFAC_STANDARD DS F Standard field
ORG DELETE_BRFAC_STANDARD
DELETE_BRFAC_EXIT_FUNCTION DS XL1 Delete type
DELETE_LINK_BRFAC EQU X'10' Delete Link Brfacility
DELETE_START_BRFAC EQU X'12' Delete Start Brfacility
DELETE_BRFAC_EXIT_COMPONENT DS CL2 Component ID 'BR'
DS CL1 Reserved
DELETE_BRFAC_TERMID DS CL4 Termid
DS CL8 Reserved
DS CL4 Reserved
DELETE_BRFAC_NETNAME DS CL8 Netname of terminal
```

図 54. DELETE 時の自動インストール制御プログラムの通信域

DELETE 時は、通信域のすべてのフィールドが入力専用です。以下のリストに含まれていないフィールドは、INSTALL で説明しています。

#### **DELETE\_BRFAC\_EXIT\_FUNCTION**

削除されるリソースのタイプを示す 1 バイトのフィールド。ブリッジ・ファシリティ 仮想端末の削除用。同等の値は以下のとおりです。

#### **INSTALL\_LINK\_BRFAC (X'10')**

link3270 ブリッジによって使用されるブリッジ・ファシリティのインストール時に自動インストール・プログラムが呼び出されました。

#### **INSTALL\_START\_BRFAC (X'12')**

START ブリッジによって使用されるブリッジ・ファシリティのインストール時に自動インストール・プログラムが呼び出されました。

#### **DELETE\_BRFAC\_EXIT COMPONENT**

2 バイトのコンポーネント・コード。「BR」に設定されます。

#### **DELETE\_BRFAC\_TERMID**

ブリッジ・ファシリティの名前を含む 4 文字のフィールド。

#### **DELETE\_BRFAC\_NETNAME**

ブリッジ・ファシリティのネット名を含む 8 文字のフィールド。

## **サンプル・プログラムのデフォルトのアクション**

クライアント 仮想端末の INSTALL 時に DFHZATDX または DFHZATDY が呼び出されると、以下を実行します。

1. CICS により SELECTED\_SHIPPED\_TERMID 内に入れられた端末名を受け入れます。

CICS が現在インストール済みのリモート TERMID との競合を検出しなかった場合、SELECTED\_SHIPPED\_TERMID には、INSTALL\_SHIPPED\_TERMID\_PTR が指す値 (クライアントから指定された名前、または CICS が生成した「VTPREFIX」名) が入っています。

CICS が現在インストール済みのリモート TERMID との競合を検出した場合、SELECTED\_SHIPPED\_TERMID には CICS が生成した別名が入っています。

2. 戻りコード・フィールドの設定をデフォルト値 RETURN\_OK のままにして戻り、リモート定義のインストールを許可します。

クライアント 仮想端末の DELETE 時に DFHZATDX または DFHZATDY が呼び出された場合は、アクションを実行せずに戻ります。

## **プログラムの自動インストールを制御するプログラムの作成**

PROGRAM、MAPSET、および PARTITIONSET の自動インストールを制御するプログラムを作成できます。プログラムの自動インストールは、特に明示されない限りこれら 3 つのプログラム・タイプを自動でインストールすることを意味します。

### **プログラムの自動インストール: 事前の考慮事項**

端末接続、IPCONN リソース接続、および APPC 接続と同様に、プログラム、マップ・セット、区画セットを自動インストールできます。プログラムの自動インストール機能を有効にすると、事前に定義されていないプログラム、マップ・セット、または区画セットに対する暗黙的または明示的なロード要求が発行された場合に、CICS が、適宜、定義を動的に作成し、インストールしてカタログします。

暗黙的または明示的なロードが発生するのは以下の場合です。

- CICS がトランザクションを開始するとき。
- アプリケーション・プログラムが、以下のコマンドのいずれか 1 つを実行するとき。
  - **EXEC CICS LINK** - [189 ページの『EXEC CICS LINK コマンドによって開始される自動インストール・プログラム』](#)を参照
  - **EXEC CICS XCTL**
  - **EXEC CICS LOAD**

- **EXEC CICS ENABLE** (グローバル・ユーザー出口プログラム、またはタスク関連のユーザー出口プログラムの場合)
- **EXEC CICS RECEIVE** または **SEND MAP**
- **EXEC CICS SEND PARTNSET**
- **EXEC CICS RECEIVE PARTN**
- 動的 COBOL 呼び出し
- プログラム異常終了が発生すると、CICS は **EXEC CICS HANDLE ABEND** コマンドで指定されたプログラムに制御権を移動します。
- CICS は、プログラムまたは端末を自動インストールするプログラムを除く、任意のユーザー置換可能プログラムを呼び出します。
- プログラムは PLTPI または PLTSD リストに指定します。

### 自動インストール・モデル定義

プログラムの自動インストールは、モデル定義 をユーザー置換可能な制御プログラムと一緒に使用して、自動インストールする必要のあるリソースの明示的な定義を作成します。

モデルの目的は、同じ特性をもつすべてのプログラムに使用できる定義を CICS に提供することです。CICS は、プログラム・タイプ (プログラム、マップ・セット、または区画セット) に適した CICS 提供のデフォルトのモデル定義の名前が含まれるパラメーター・リストと一緒に自動インストール制御プログラムを呼び出します。自動インストール制御プログラムは、デフォルト・モデルを受け入れることも、別のモデル (インストール済みのプログラム定義はすべてモデルとして使用可能) を選択することもできます。また、プログラムに固有のプロパティを明示的に指定して、モデル定義に指定されたプロパティをオーバーライドすることもできます。ローカル定義またはリモート定義をインストールするように指定できます。

制御プログラムからの戻りを受けて、CICS はパラメーター・リスト に戻されたモデルやプロパティからリソース定義を作成します。

CICS のプログラム、マップ・セット、または区画セット (つまり "DFH" の文字から始まるすべてのオブジェクト) について、CICS はデフォルトの モデル定義を使用しますが、ユーザー置換可能な自動インストール制御プログラムは呼び出しません。独自の自動インストール制御プログラムがある場合、それを使用して文字 "DFH" で始まるオブジェクトのリソース定義を変更することはできません。

### EXEC CICS LINK コマンドによって開始される自動インストール・プログラム

自動インストール・プログラムは動的ルーティング・プログラムと関連します。SYSID が指定されていない **EXEC CICS LINK** コマンドで指定されたプログラムの定義がインストールされていないために自動インストール制御プログラムが開始されると、自動インストール制御プログラムは状況に応じて別の定義をインストールすることができます。

自動インストール制御プログラムは、以下の定義をインストールできます。

#### サーバー・プログラムのローカル定義

CICS は、ローカル領域でサーバー・プログラムを実行します。

#### REMOTESYSTEM(remote\_region) と DYNAMIC(NO) を指定する定義

CICS は、LINK 要求をリモート領域にシップします。

#### DYNAMIC(YES) を指定する定義

CICS は、LINK 要求をルーティングするための動的ルーティング・プログラムを開始します。

注: DYNAMIC 属性は、REMOTESYSTEM 属性に優先します。したがって、REMOTESYSTEM(remote\_region) と DYNAMIC(YES) の両方を指定した定義では、プログラムは特定のリモート領域に置かれるのではなく、動的なものとして定義されます。この場合、動的ルーティング・プログラムに渡されるデフォルトのサーバー領域を REMOTESYSTEM 属性で指定します。

#### サーバー・プログラムの定義がない

CICS は、LINK 要求をルーティングするための動的ルーティング・プログラムを開始します。

注：この状況では、自動インストール制御プログラムは定義をインストールしないと見なされます。自動インストールが失敗したために定義がインストールされていない場合、動的ルーティング・プログラムは開始されません。

### 自動インストールによるマップ・セットの処理

190 ページの表 16 に、プログラムの自動インストールがアクティブになっている CICS 領域とアクティブになっていない CICS 領域におけるマップ・セットの処理の違いを示します。

表 16. 自動インストールと非自動インストールにおけるマップ・セットの処理の違い	
プログラムの自動インストール INACTIVE	プログラムの自動インストール ACTIVE
CSD 定義が必要。CICS は、接尾部を持つ参照されたマップ・セットのロードを試行します。失敗すると、CICS は接尾部のないバージョンを試行します。これが失敗する場合、APCT の異常終了が発行されます。	CSD 定義は不要です。自動インストールを使用して、CICS は参照された接尾部のあるマップ・セットまたは区画セットのロードを試行し、その後、接尾部のないバージョンを試行します。(それぞれのケースで、定義は自動インストールされます。) リソースを要求するトランザクションが異常終了するのは、ライブラリーに接尾部のあるものとないもの、いずれのバージョンのリソースも存在しない場合のみです。  接尾部のあるマップ・セットがライブラリーにない場合、定義は「ロード不可能」としてマークされます。

### システムによる自動インストール

一部のプログラム (静的に定義されていない場合) は、CICS のシステムによる自動インストール機能によって自動インストールされます。その際、モデル定義や自動インストール制御プログラムのサポートは必要ありません。

このカテゴリーに含まれるプログラムは、次のとおりです。

- ・言語環境プログラムに必要なプログラム。
- ・最初のフェーズのプログラム・リスト・テーブル事後初期化 (PLTPI) プログラム (つまり、PLT テーブルの区切り文字 DFHDELIM の前に定義されている PLTPI プログラム)。
- ・2 番目のフェーズのプログラム・リスト・テーブル・シャットダウン (PLTSD) プログラム (つまり、PLT テーブルの区切り文字 DFHDELIM の後に定義されている PLTSD プログラム)。

注：DFHDELIM の後に定義されている PLTPI プログラムと、DFHDELIM の前に定義されている PLTSD プログラムは、他のユーザー・プログラムと同様に扱われます。これらは、プログラムの自動インストールの対象です。

## プログラムの自動インストールの利点

プログラムの自動インストールを使用すると、システム管理を軽減し、仮想ストレージの使用量を削減できます。場合によっては再起動時間も短縮できます。

### システム管理コストの削減

自動インストールを使用しない場合、新規のプログラム、マップ・セット、および区画セットを使用するには、それらをすべて CICS で定義する必要があります。自動インストールを使用すると、その必要がなくなり、これらのリソースは事前の定義をせずに使用できます。さらに、事前定義の定義を保守する必要もなくなるため、システム管理にかかる労力を大幅に削減することができます。

### 仮想記憶域の節約

CICS アドレス・スペース内の仮想ストレージが節約されます。これは、自動インストールされたリソースの定義は、生成されるまでテーブル・スペースを占有しないためです。

### 起動時間の短縮



### ウォーム・スタートと緊急時始動

プログラムの自動インストールを使用する場合、再始動時間はプログラムの自動インストールでカタログ作成を行うかどうかによって異なります。

カタログ作成を行うプログラムの自動インストールを使用する場合、再始動の時間はプログラムの自動インストールを使用していない CICS 領域の再始動と同程度です。これは、これらどちらのケースでも再始動時にリソース定義がカタログから再インストールされるためです。再始動の後の定義は、システム終了前に存在していた定義です。

カタログ作成を行わない自動インストールを使用する場合、CICS の再始動時間が短縮されます。これは、CICS が定義を CICS グローバル・カタログからインストールしないためです。その代わり、定義はプログラム、マップ・セット、区画セットが再始動後に参照されて必要になったときに自動インストールされます。

カタログ作成について詳しくは、[リカバリー処理のためのトラブルシューティング](#)を参照してください。

### 初期始動とコールド・スタート

プログラムの自動インストールを使用しない領域よりも、起動時間が短縮されます。これは、プログラム定義が起動時にすべてインストールされるのではなく、必要になったときに個別にインストールされるためです。

## プログラムの自動インストールの構成

プログラム、マップ・セット、区画セットを自動的にインストールするには、自動インストール・プログラムを使用して必要なリソースをインストールするように CICS を構成する必要があります。

### このタスクについて

CICS には DFHPGADX という自動インストール・プログラムが付属していますが、必要場合はカスタマイズ・バージョンを独自に作成できます。

### 手順

1. 用意されているバージョンの自動インストール制御プログラム DFHPGADX で目的が完全に満たされる場合を除き、カスタマイズ・バージョンを作成してください。
2. **PGAEXIT** システム初期設定パラメーターまたは **SET SYSTEM PROGAUTOEXIT** コマンドで、作成した制御プログラムの名前を指定します。  
デフォルト名は DFHPGADX です。このパラメーターについて詳しくは、[PGAEXIT システム初期設定パラメーター](#)を参照してください。
3. **PGAIPGM** システム初期設定パラメーターで「ACTIVE」と指定するか、または **SET SYSTEM PROGAUTOINST(AUTOACTIVE)** コマンドを発行して、プログラムの自動インストールをアクティブにします。  
このパラメーターについて詳しくは、[PGAIPGM システム初期設定パラメーター](#)を参照してください。
4. 自動インストールされたプログラム定義を CICS グローバル・カタログに記録するかどうかを指定します。  
**PGAICTLG** システム初期設定パラメーターまたは **SET SYSTEM PROGAUTOCTLG** コマンドを使用できます。このパラメーターについて詳しくは、[PGAICTLG システム初期設定パラメーター](#)を参照してください。
5. DFHPGAIP リソース定義グループを、CICS 始動グループ・リストに含めます。  
DFHPGAIP は、用意されている始動リスト DFHLIST に既に含まれています。リストには、自動インストール制御プログラムに渡されるデフォルトのプログラム、マップ・セット、および区画セットのモデル定義と、DFHPGADX の定義が含まれています。DFHPGADX の定義を変更しなければならない場合があります。
6. 必要な追加のプログラム、マップ・セット、区画セットのモデル定義を作成し、始動グループ・リストにそのグループを追加します。
7. プログラムの自動インストールに関連したメッセージを記録する場合は、CSPL 一時データ (TD) キューを定義します。

## タスクの結果

これで、プログラム、マップ・セット、区画セットの自動インストールを正常に構成できました。

## INSTALL 時の自動インストール制御プログラム

呼び出しのときに、CICS は、DFHEICAP によりアドレス指定された通信域を使用してパラメーター・リストを自動インストール制御プログラムに渡します。通信域はコピーブックによってマップされます。コピーブックは、CICS のすべてのサポート対象言語のものが用意されています。

アセンブラ形式のパラメーター・リストを次に示します。

### PGAC\_PROGRAM

自動インストールするオブジェクトの 8 バイトの名前を渡します。これは入力専用フィールドであり、ユーザー置換可能プログラムで変更してはいけません。

### PGAC\_MODULE\_TYPE

インストールするオブジェクトのタイプを示す 1 バイトの標識を渡します。同等の値は以下のとおりです。

#### PGAC\_TYPE\_PROGRAM

プログラム

#### PGAC\_TYPE\_MAPSET

マップ・セット

#### PGAC\_TYPE\_PARTITIONSET

区画セット

これは入力専用フィールドであり、ユーザー置換可能プログラムで変更してはいけません。

### PGAC\_MODEL\_NAME

使用する自動インストールのモデル名 (8 バイト) を制御プログラムで指定できます。このフィールドを設定しない場合、CICS はオブジェクトのタイプに応じたデフォルトのモデル名を使用します。

#### DFHPGAPG

プログラムの場合

#### DFHPGAMP

マップ・セットの場合

#### DFHPGAPT

区画セットの場合。

### PGAC\_LANGUAGE

自動インストールするプログラムの言語 (1 バイトのフィールド) を制御プログラムで指定できます。同等の値は以下のとおりです。

#### PGAC\_ASSEMBLER

アセンブラ

#### PGAC\_COBOL

COBOL

#### PGAC\_C370

C

#### PGAC\_LE370

Language Environment

#### PGAC\_PLI

PL/I。

このフィールドを設定しない場合、自動インストール・ルーチンはモデルに定義されている言語 (指定されている場合) を使用します。ただし、制御権がプログラムに渡されるときに CICS はプログラム自体から言語を特定し、指定された値をオーバーライドします。

コンパイルの前に EXEC CICS 変換プログラムを使用して変換された実行可能プログラムの言語を指定する必要はありません。

**PGAC\_CEDF\_STATUS**

自動インストールするプログラムの実行診断機能 (EDF) のステータスを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_CEDF\_YES**

EDF はこのプログラムで使用できます。

**PGAC\_CEDF\_NO**

EDF はこのプログラムで使用できません。

**PGAC\_DATA\_LOCATION**

タスク存続時間ストレージのデータ・ロケーションを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_LOCATION\_BELOW**

タスク存続時間ストレージは 16 MB 境界の下でなければなりません。

**PGAC\_LOCATION\_ANY**

タスク存続時間ストレージは 16 MB 境界の上下どちらにも割り振れます。

**PGAC\_EXECUTION\_KEY**

プログラムの実行キーを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_CICS\_KEY**

プログラムを CICS キーで実行します。

**PGAC\_USER\_KEY**

プログラムをユーザー・キーで実行します。

**PGAC\_LOAD\_ATTRIBUTE**

オブジェクトのロード属性を 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_RELOAD**

CICS は要求ごとにオブジェクトの最新コピーをロードします。

**PGAC\_RESIDENT**

CICS はオブジェクトを永続的に常駐させます。

**PGAC\_TRANSIENT**

このオブジェクトのストレージは、使用回数がゼロに達するとすぐに開放されます。

**PGAC\_REUSABLE**

CICS は、現在ストレージ内にあるオブジェクトの任意のコピーを使用できます。

**PGAC\_USE\_LPA\_COPY**

CICS でプログラムの LPA 常駐コピーを使用するかどうかを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_LPA\_YES**

CICS は LPA にあるコピーを使用します。

**PGAC\_LPA\_NO**

CICS は独自の DFHRPL または動的 LIBRARY 連結からプライベート・コピーをロードします。

**PGAC\_EXECUTION\_SET**

CICS API の分散プログラム・リンク (DPL) のサブセットを使用するようにプログラムを制限するかどうかを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

**PGAC\_DPLSUBSET**

プログラムは EXEC CICS API の DPL サブセットを使用するように制限されます。

**PGAC\_FULLAPI**

プログラムは、全 API を使用できます。

**PGAC\_REMOTE\_SYSID**

プログラムを実行するリモート・システムの名前を 4 バイトのフィールドで指定できます。CICS 機能は、このプログラムに対するすべての要求を、指定されたリモート CICS にシップします。

#### **PGAC\_REMOTE\_PROGID**

リモートの CICS 領域でのプログラムの識別名を 8 バイトのフィールドで指定できます。リモート・プログラムの場合にこのフィールドに空白を設定すると、リモート名はデフォルトのローカル名になります。

#### **PGAC\_REMOTE\_TRANSID**

リモートの場合に、このプログラムを実行する CICS ミラー・トランザクションの名前を 4 バイトのフィールドで指定できます。デフォルトでは、これには CICS ミラー・トランザクションの名前である CSMI が設定されます。

#### **PGAC\_DYNAMIC\_STATUS**

プログラムがプログラム・リンク要求の対象である場合に、その要求を動的にルーティングできるかどうかを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

#### **PGAC\_DYNAMIC\_NO**

プログラムがプログラム・リンク要求の対象である場合、動的ルーティング・プログラムが呼び出されません。

分散プログラム・リンク (DPL) 要求の場合は、PROGRAM 定義の REMOTESYSTEM オプションまたは **EXEC CICS LINK** コマンドの SYSID オプションで、プログラムを実行するサーバー領域を明示的に指定する必要があります。指定しないと、デフォルトのローカル領域になります。

#### **PGAC\_DYNAMIC\_YES**

プログラムがプログラム・リンク要求の対象である場合、動的ルーティング・プログラムが呼び出されます。リモート・サーバー領域が **EXEC CICS LINK** コマンドの SYSID オプションで明示的に指名されていない場合、ルーティング・プログラムはその要求をプログラムを実行する領域にルーティングできます。

#### **PGAC\_CONCURRENCY**

プログラムがスレッド・セーフ標準に従って作成されているかどうかを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

#### **PGAC\_QUASIRENT**

このプログラムは、準再入可能専用であり、共用リソースにアクセスするときには CICS により提供されるシリアライゼーションに依存します。

プログラムは、CICS で許可されているプログラミング・インターフェースに制限され、CICS の準再入可能規則に準拠している必要があります。

#### **PGAC\_THREADSAFE**

プログラムはスレッド・セーフ標準に従って作成されています。また、共用リソースにアクセスするときには、同時に実行されている他のプログラムが同じリソースを変更しようとしている可能性を考慮します。

#### **PGAC\_JVM**

プログラムが JVM で実行されるものかどうかを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

#### **PGAC\_JVM\_YES**

プログラムは Java バイトコード・プログラムであり、JVM の制御下で実行する必要があります。

#### **PGAC\_JVM\_NO**

プログラムの実行に JVM は不要です。

#### **PGAC\_JVM\_CLASS\_LEN**

PGAC\_JVM\_CLASS\_DATA で指定する Java クラス名の長さを 2 バイトのバイナリー値で指定できます。

#### **PGAC\_JVM\_CLASS\_DATA**

呼び出す Java クラスの名前を 256 バイトのフィールドで指定できます。

#### **PGAC\_JVM\_PROFID**

プログラムを実行する JVM で使用する JVM プロファイルの名前を、ユーザーが 8 バイトのフィールドで指定できます。

## PGAC\_RETURN\_CODE

自動インストール制御プログラムから CICS への戻りコードを 1 バイトのフィールドで指定できます。同等の値は以下のとおりです。

## PGAC\_RETURN\_OK

通信域パラメーター・リストに戻された値を使用して、プログラム定義をインストールします。

## PGAC\_RETURN\_DONT\_DEFINE\_PROGRAM

プログラムを定義しません。

## プログラムのための自動インストール制御プログラムのサンプル DFHPGADX

CICS 提供のデフォルトの自動インストール・プログラムは、アセンブラー言語によるコマンド・レベル・プログラムであり、名前は DFHPGADX です。デフォルト・プログラムのソースは、COBOL、PL/I、C、およびアセンブラー言語で提供されています。

CICS 提供のプログラムおよび関連するコピーブックの名前、およびそれらが入っている CICSTS56.CICS ライブラリーについては、[195 ページの表 17](#) にまとめています。

表 17. プログラム自動インストール用のプログラムおよびコピーブックのサンプル		
言語	メンバー名	ライブラリー
<b>実行可能ファイル:</b>		
アセンブラー (のみ)	DFHPGADX	SDFHLOAD
<b>プログラムのソース:</b>		
アセンブラー	DFHPGADX	SDFHSAMP
COBOL	DFHPGAOX	SDFHSAMP
PL/I	DFHPGALX	SDFHSAMP
C	DFHPGAHX	SDFHSAMP
<b>コピーブック:</b>		
アセンブラー	DFHPGACD	SDFHMAC
COBOL	DFHPGACO	SDFHCOB
PL/I	DFHPGACL	SDFHPL1
C	DFHPGACH	SDFHC370

## サンプル・プログラムのカスタマイズ

自動インストール制御プログラムは、CICS がサポートするすべての言語で作成できます。制御プログラムは、CICS アプリケーションおよびシステム・プログラミング・インターフェースへのフルアクセスを持っています。

提供された制御プログラムをカスタマイズする場合や、独自の制御プログラムを作成する場合は、以下の点にご注意ください。

- **入力:** パラメーター・リストの最初の 2 つのフィールドは入力専用のフィールドで、プログラムによって変更することはできません。
- **出力:** パラメーター・リストの残りのフィールドは、入出力用、または出力専用のフィールドで、モデル定義のフィールドをオーバーライドする属性を指定するために使用できます。
- パラメーター・リストの一部の出力フィールドは、マップ・セットおよび区画セットには適用されません。インストール対象のオブジェクトのタイプに適用されないパラメーターが指定された場合、CICS はこれを無視します。
- CICS のパラメーター・リストに戻す属性は、モデル定義を変更するために使用され、CICS はその変更された定義をインストールします。インストールしたら、定義は CICS Explorer の「プログラム」操作ビ

ユー、**EXEC CICS SET PROGRAM** コマンド、または **CEMT SET PROGRAM** コマンドを使用して通常どおり変更できます。

- 制御プログラムを変更した場合、NEWCOPY オプションまたは属性をビューまたはコマンドで使用して、新しいバージョンを使用可能にすることができます。
- 定義は、インストール後に破棄することができます。それらは、次の参照時に再度インストールされます。
- CICS に戻すパラメーターが有効で、CICS 領域内の他のシステム属性と整合性があることを確認する必要があります。以下に例を挙げます。
  - CICS がシステム 初期設定パラメーター LPA=NO で実行されている場合は、PGAC\_USE\_LPA\_COPY パラメーターに対して PGAC\_LPA\_YES を戻さないでください。
  - 制御プログラムで呼び出されるタスクが CICS キーのタスク存続期間ストレージで実行されている場合、PGAC\_EXECUTION\_KEY パラメーターに対して PGAC\_USER\_KEY (デフォルト) を戻さないでください。

以下の **EXEC CICS** コマンドを使用して、トランザクション定義の TASKDATAKEY オプションをテストすることで、タスクのストレージ・キーを判別できます。

- **EXEC CICS ADDRESS EIB**

- **EXEC CICS INQUIRE TRANSACTION(EIBTRANS) TASKDATAKEY(...)**

## 重要

自動インストールされたプログラム定義を作成する場合、CICS はモデル・プログラム定義に指定されているプログラム言語を無視します。CICS は、自動インストールされたプログラムが開始されると、ロード・モジュール自体から言語を判別します。

ただし、CICS はロード・モジュールから言語以外の特性は推測できません。他のプログラムの特性は、自動インストール制御プログラムまたは RDO によって、明示的に定義する必要があります。変化する特性がプログラムに含まれる場合 (例えば、変化する AMODE 要件や DATALOCATION 要件)、自動インストールを使用する際に、各タイプを区別できる必要があります。デフォルトの特性への例外のリストを保持し、自動インストール制御プログラムがこのリストを参照するようにコーディングします。あるいは、例外の明示的な RDO 定義をインストールすることもできます。

## リソース定義

自動インストール制御プログラムは、それ自体を自動インストールしたり、参照先のプログラムを自動インストールしたりすることはできません。制御プログラムやその他の参照先のプログラム用のプログラム・リソース定義を作成する必要があります。

これらの定義が含まれるグループを始動グループ・リストに含めることで、始動時に定義が CICS 領域にインストールされるようにする必要があります。制御プログラムに無効な名前を指定した場合、CICS はプログラムを無効にするため、プログラムの自動インストール機能は無効になります。

以下のプログラム・リソース定義は、自動インストール制御プログラム用に CICS によって提供されています。デフォルトはアセンブラ・バージョンの DFHPGADX です。これらの定義が各自の使用に適していない場合、RDO または DFHCSDUP ユーティリティを使用して、独自の定義を作成することができます。

- DFHPGADX のデフォルトの自動インストール制御プログラム定義。これはアセンブラ・バージョンを定義し、状況は ENABLED に設定されます。

GROUP(DFHPGAIP)	PROGRAM(DFHPGADX)
DESCRIPTION(Assembler	definition for program autoinstall exit)
LANGUAGE(ASSEMBLER)	EXECKEY(CICS)
RELOAD(NO)	EXECUTIONSET(FULLAPI)
STATUS(ENABLED)	RESIDENT(NO)
CONCURRENCY(THREADSAFE)	USAGE(NORMAL)
	CEDF(NO)
	DATALOCATION(ANY)

- DFHPGAOX の自動インストール制御プログラム定義。これは CICS 提供の COBOL バージョンを定義し、状況は DISABLED に設定されます。

GROUP(DFHPGAIP)	PROGRAM(DFHPGAOX)
DESCRIPTION(COBOL definition for program autoinstall exit)	



LANGUAGE(COBOL)	EXECKEY(CICS)	EXECUTIONSET(FULLAPI)
RELOAD(NO)	RESIDENT(NO)	USAGE(NORMAL)
STATUS(DISABLED)	CEDF(NO)	DATALOCATION(ANY)
CONCURRENCY(THREADSAFE)		

- DFHPGAHX の自動インストール制御プログラム定義。これは CICS 提供の C バージョンを定義し、状況は DISABLED に設定されます。

GROUP(DFHPGAIP)	PROGRAM(DFHPGAHX)	
DESCRIPTION(C definition for program autoinstall exit)		
LANGUAGE(C)	EXECKEY(CICS)	EXECUTIONSET(FULLAPI)
RELOAD(NO)	RESIDENT(NO)	USAGE(NORMAL)
STATUS(DISABLED)	CEDF(NO)	DATALOCATION(ANY)
CONCURRENCY(THREADSAFE)		

- DFHPGALX の自動インストール制御プログラム定義。これは CICS 提供の PL/I バージョンを定義し、状況は DISABLED に設定されます。

GROUP(DFHPGAIP)	PROGRAM(DFHPGALX)	
DESCRIPTION(PL/I definition for program autoinstall exit)		
LANGUAGE(PLI)	EXECKEY(CICS)	EXECUTIONSET(FULLAPI)
RELOAD(NO)	RESIDENT(NO)	USAGE(NORMAL)
STATUS(DISABLED)	CEDF(NO)	DATALOCATION(ANY)
CONCURRENCY(THREADSAFE)		

### プログラムのテストとデバッグ

CICS 実行診断機能 (EDF) を使用して、自動インストール制御プログラムをテストすることができます。ただし、EDF は、DFH の文字で始まる名前のプログラムには使用できません。そのため、EDF を使用する場合は、プログラム名を デフォルト名以外のものにする必要があります。

### このタスクについて

## 動的ルーティング・プログラムの作成

CICS には、端末や CICS コマンドのサブセットによって開始されたトランザクションや、プログラム・リンク要求のルーティングのために、動的ルーティング・プログラムが用意されています。CICSplex SM には、ワークロード・ルーティングを実行するための動的ルーティング・プログラムがあります。それらのプログラムで要件に対応できない場合は、独自の動的ルーティング・プログラムを作成できます。

動的ルーティング・プログラムを作成するには、CICS のトランザクション・ルーティングや分散プログラム・リンクや動的ルーティングの原則をしっかりと理解しておく必要があります。START コマンドで開始されるトランザクションや、動的ルーティングに適したプログラム・リンク要求の詳細については、[START コマンドで呼び出されたトランザクションのルーティング](#)を参照してください。

### 制約事項

以下のルーティングのために動的ルーティング・プログラムを使用することはできません。

- CICS ビジネス・トランザクション・サービスのアクティビティーおよびプロセス。
- 非端末関連の **EXEC CICS START** 要求
- インバウンド Web サービス要求。
- **EXEC CICS RUN TRANSID** 要求。

これらのタイプの要求のルーティングを実行するには、分散ルーティング・プログラムを使用する必要があります。分散ルーティング・プログラムの作成方法の詳細は、[227 ページの『分散ルーティング・プログラムの作成』](#)を参照してください。

## トランザクションの動的ルーティング

トランザクションの動的ルーティングは、ユーザー端末から開始するか、適格な端末関連 **EXEC CICS START** コマンドで開始できます。

CICS にトランザクションを定義する際には、リモート またはローカル として記述できます。ローカル・トランザクションは常に端末専有領域で実行されます。リモート・トランザクションは、IPIC、MRO、ま

たは APPC (LUTYPE6.2) ISC リンクにより端末専有領域に接続されているその他の領域にルーティングできます。IPIC は、CICS TS 4.1 以降の領域間の 3270 端末のトランザクション・ルーティングをサポートします。この端末では、端末専有領域 (TOR) は APPLID によって固有に識別されます。

動的ルーティング・プログラムを使用して、トランザクションのルーティング先のシステムとトランザクションのリモート名の両方を、CICS にトランザクションを定義するときではなく、動的に選択することができます。CICS 提供のデフォルトのルーティング・プログラムは DFHDYP と呼ばれます。ソース・レベルのコードは、アセンブラ言語、COBOL、PL/I、C バージョンのものが用意されています。これらの言語のいずれかで独自のプログラムを作成する場合は、デフォルトのプログラムをモデルとして使用できます。

### 動的トランザクション

トランザクションを動的にルーティングするには、それらのトランザクションを値 DYNAMIC(YES) で定義し、リモートとローカルの両方のオプション用の値を指定する必要があります。

このような方法でトランザクションを定義することで、トランザクションがルーティングされる際に CICS が適切な値を選択し、不要な値は無視するようにすることができます。

トランザクションを動的トランザクション・ルーティング用に定義する方法については、[トランザクション・ルーティングのトランザクションの定義](#)を参照してください。

### 動的ルーティング・プログラムを呼び出す場合

ユーザー端末から開始されたトランザクション、または端末関連の適格な **EXEC CICS START** コマンドにより開始されたトランザクションの場合、CICS は次のように動的ルーティング・プログラムを呼び出します。

- DYNAMIC(YES) として定義されたトランザクションが開始されたとき。

注：

1. トランザクション定義が見つからない場合は、CICS は [DTRTRAN](#) システム初期設定パラメーターで指定された共通のトランザクション定義を使用します。
  2. DYNAMIC(YES) として定義され、端末関連の **EXEC CICS START** コマンドによって開始されたトランザクションが動的ルーティングに不適格な場合、ルーティング・プログラムは通知のためにのみ起動されます。つまり、トランザクションをルーティングすることはできません。
- ルート選択でエラーが発生した場合、例えば、最初の (ルート選択) 呼び出しでルーティング・プログラムから返されたターゲット領域が使用できない場合。この場合、ルーティング・プログラムは代替ターゲットを指定できます。このプロセスは、ルーティング・プログラムが使用可能なターゲットを選択するか、ゼロでない戻りコードを設定するまで繰り返されます。
  - ルーティングされたトランザクションが完了した後 (ルーティング・プログラムで終了時に再起動することを要求していた場合)
  - ルーティングされたトランザクションが異常終了した場合 (ルーティング・プログラムで終了時に再起動することを要求していた場合)

[199 ページの図 55](#) に、動的ルーティング・プログラムが起動されるポイントを示します。

Route selection
Notification
Route selection error
Transaction/link request termination
Transaction abend

図 55. 動的ルーティング・プログラムを呼び出す場合

#### 動的ルーティング・プログラムに渡される情報

CICS の中継プログラム DFHAPRT は、通信域を使用して情報を動的ルーティング・プログラムに渡します。通信域には、DSECT DFHDYPDS によってマップされたフィールドが含まれています。

DFHDYPDS DSECT については、213 ページの『動的ルーティング・プログラムに渡されるパラメーター』で詳しく説明します。トランザクションのルーティングで、通信域を介して動的ルーティング・プログラムに渡されるデータを次に示します。

- ・ トランザクションがインストールされたときに指定されたリモート CICS 領域の SYSID
- ・ リモート CICS 領域のネット名
- ・ リモート・トランザクションの名前
- ・ 中継トランザクション・タスクの優先順位 (MRO および IPIC 接続のみ)
- ・ リモート CICS 領域にすぐに使用できるセッションがない場合に、要求をキューに入れるかどうか
- ・ リモート・トランザクションの通信域のアドレス
- ・ トランザクションの端末入出力域 (TIOA) のコピーのアドレス
- ・ タスク・ローカル・ユーザー・データ域。
- ・ アプリケーション・コンテキスト・データ。

通信域 DSECT には、渡される情報を説明するコメントが含まれます。

動的ルーティング・プログラムはこれらの値を受け入れることも変更することもできます。また、CICS に対してトランザクションのルーティングを続行しないように命令することもできます。使用される値は、実行する機能によって異なります。つまり、一部の値は無視される場合があります。

動的ルーティング・プログラムに渡される情報は、トランザクションが動的にルーティングされるか静的にルーティングされるかを示します。トランザクションが動的にルーティングされる場合、動的ルーティング・プログラムは SYSID またはネット名を変更して、トランザクションの実行場所を決定できます。

静的にルーティングされるトランザクションに対して動的ルーティング・プログラムが呼び出される場合もあります。例えば、インターバル制御開始要求の期間満了などによって、DYNAMIC(YES) と定義されたトランザクションを自動トランザクション開始 (ATI) が開始した場合に呼び出されますが、このトランザクションを動的ルーティングすることはできません。この場合、動的ルーティング・プログラムは呼び出されますが、トランザクションの実行場所を自身に通知するだけです。動的ルーティング・プログラムはリモート・システム名を変更できないので、通信域の SYSID または NETNAME フィールドに変更を加えても無視されます。

リモートとして定義されたか、またはリモート CICS 領域に動的にルーティングされたためにリモートで実行されるトランザクションの場合、CICS モニタリングにはリモート CICS 領域の SYSID が通知されます。動的ルーティング・プログラムがローカルにルーティングしたトランザクションの場合、モニタリング・フィールドはヌルに設定されます。

## ターゲット CICS 領域の変更

トランザクションのルーティング先のデフォルトの CICS 領域のシステム ID (sysid) とネット名を変更することで、動的ルーティング・プログラムはターゲット CICS 領域を変更することができます。

初期設定では、動的ルーティング・プログラムに渡される通信域には、トランザクションのルーティング先のデフォルトの CICS 領域のシステム ID (sysid) とネット名が含まれています。これらは、インストールされたトランザクション定義の REMOTESYSTEM オプションの値から派生します。トランザクション定義が REMOTESYSTEM 値を指定しない場合、渡される sysid とネット名は、ローカルの CICS 領域のもので

す。

動的ルーティング・プログラムは、SYSID とネット名を変更できます。経路選択のために呼び出された時に変更する場合、トランザクションのルーティング先の領域は、次のように決定されます。

- NETNAME および SYSID が変更されない。

CICS は、通信域で当初指定されていたとおりの SYSID へのルーティングを試行します。

- NETNAME は変更されないが、SYSID が変更される。

CICS は新しい SYSID に対応する NETNAME で通信域を更新し、新しい SYSID へのルーティングを試行します。

- NETNAME は変更されるが、SYSID は変更されない。

CICS は新しい NETNAME に対応する SYSID で通信域を更新し、新しい SYSID へのルーティングを試行します。

- NETNAME は変更され、さらに SYSID も変更される。

CICS は新しい NETNAME に対応する SYSID で通信域を上書きし、その新しい SYSID へのルーティングを試行します。

指定された NETNAME が無効であるか、見つからない場合、動的ルーティング・プログラムに SYSIDERR が返されます。別の SYSID または NETNAME を返すことによって、このエラーに対応することができます。[201 ページの『システムが使用不可または不明の場合』](#)を参照してください。

ルーティング・プログラムが通知のために呼び出された場合、ルーティング・プログラムが SYSID または NETNAME を変更しても、その変更は無効です。

## TOR での共通トランザクション定義の使用

動的にルーティングされるすべてのリモート・トランザクションには、単一の共通定義を使用することをお勧めします。

共通定義の名前は、DTRTRAN システム初期設定パラメーターに指定されています。共通定義の REMOTESYSTEM オプションを使用して、トランザクションのルーティング先のデフォルトの AOR を指定できます。リモート・トランザクションを動的トランザクション・ルーティング用に定義する方法については、[リモート・リソースの定義](#)を参照してください。

**重要:** DTRTRAN 定義によって定義されたトランザクションをルーティングするには、動的ルーティング・プログラムで通信域の DYRDTRRJ フィールドを 'N' (デフォルトは 'Y') に設定する必要があります。DYTDTRRJ が 'Y' に設定されている場合、トランザクションは拒否されます。

DYRDTRXN フィールドをテストして、ルーティング・プログラムに渡されたトランザクションが DTRTRAN 定義で定義済みかどうか確認できます。[201 ページの図 56](#)には、DTRTRAN で定義されたルーティング・トランザクションのスケルトン・コードが含まれます。

```

if DYRDTRXN='Y' then          /* Is DYP invoked because of DTRTRAN */
do                             /* .. Yes                               */
  Call Find_AOR(sysid)        /* Select the SYSID of the AOR         */
  if rc=0 then                /* Is AOR available?                   */
  do                           /* .. Yes                               */
    DYRRETC=RETCOD0           /* Set OK Return Code                  */
    DYRSYSID=sysid            /* Set the sysid                       */
    DYRDTRRJ='N'              /* Don't reject DTRTRAN defs          */
    ...                       /* Set other commarea fields          */
  end                           /* ..                                     */
  else                         /* .. No                               */
    ...                       /* AOR unavailable logic               */
  end                           /* ..                                     */
end                             /* ..                                     */

```

図 56. DTRTRAN で定義されたトランザクションをルーティングする疑似コードの例

## プログラム名の変更

DYNAMIC として定義されたトランザクションでは、ルーティング・プログラムを呼び出すと、通信域の DYRLPROG フィールドにはルーティングするトランザクションに関連付けられた初期プログラムの名前が入っています。トランザクションをローカルにルーティングすることを決定した場合、このフィールドを使用して、実行される代替プログラムを指定することができます。

例えば、すべてのリモート CICS 領域が使用不能なため、トランザクションをルーティングできない場合、ローカル CICS 端末専有領域でプログラムを実行して、適切なメッセージをユーザーに送信したい場合があります。

## トランザクションをルーティングするか終了するかについての CICS への指示

ルーティングのためにルーティング・プログラムが呼び出された場合、プログラムではトランザクションをルーティングするか終了するかを選択することができます。

値を変更したかどうかにかかわらずトランザクションをルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された SYSID と自身のローカル SYSID を次のように比較します。

- SYSID が同じ (または、戻された SYSID がブランク) である場合、CICS はトランザクションをローカルで実行します。
- 2 つの SYSID が異なる場合、CICS はリモート・トランザクション名を使用して、トランザクションをリモートの CICS 領域にルーティングします。

トランザクションをメッセージを出して終了させるか異常終了で終了させるには、戻りコード X'8' (または、X'4' 以外のその他の非ゼロの戻りコード) を設定します。

トランザクションをメッセージを出さずに終了させるか異常終了させるには、戻りコード X'4' を設定します。

**Warning :** APPC トランザクション・ルーティングで戻りコード X'4' を設定すると、予期しない結果につながるため、避けてください。

ルーティング・プログラムが通知のため、またはトランザクションの終了時に呼び出された場合は、DYRRETC に値を戻しても無効です。

## システムが使用不可または不明の場合

ルート選択呼び出しで指定したリモート・システム名が不明または使用不可の場合は、動的ルーティング・プログラムが再度呼び出されます。

プログラムが再呼び出しされたら、以下のアクションのいずれかを選択できます。

- DYRRETC で戻りコード「8」を発行することで、トランザクションのルーティングの試行を続けないよう CICS に通知できます。システムが使用不可であることがエラーの原因である場合、CICS は「DFHAC2014」または「DFHAC2029」メッセージを端末ユーザーに発行します。システムが不明であることがエラーの原因である場合、DFHAPRT はトランザクションを異常終了します。
- DYRRETC に戻りコード「4」を設定することで、メッセージを出したり異常終了したりせずにトランザクションを終了するよう CICS に通知できます。ただし、戻りコード「4」を設定することに対する警告に注意してください。

- ・リモート・システムが直ちに使用できるセッションがないことがエラーの原因である場合は、フィールド **DYRQUEUE** を「Y」に再設定し (このエラーが発生するということは、これまでは「N」 (要求はキューに入れられない) に設定されていたはず)、**DYRRETC** で戻りコード「0」を発行して、トランザクションのルーティングを再試行します。

**DYRQUEUE** を「Y」に再設定せずに (なおかつシステム ID を変更せずに) トランザクションのルーティングを再試行した場合、システムがまだ使用不可であれば、**DFHDYP** が再呼び出しされます。一方、戻りコード「8」を設定することを選択すると、**CICS** はメッセージ「**DFHAC2030**」でトランザクションを終了します。

- ・システム ID を変更し、**DYRRETC** で戻りコード「0」を発行することで、トランザクションのルーティングを再試行できます。なお、システム ID を変更する場合は、異なるリモート・トランザクション ID も指定しなければならないことがあります。例えば、トランザクションのリモート・トランザクション名がシステムごとに異なる場合は、このような処理が必要です。

このトランザクションのルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド **DYRCOUNT** で渡されます。このカウントを使用すると、トランザクションのルーティングの試行をいつ中止するか決定に役立ちます。

### ルーティングされたトランザクションの終了時の動的ルーティング・プログラムの呼び出し

ルーティングされたトランザクションが完了したときに動的ルーティング・プログラムが再度呼び出されるようにするには、**CICS** に制御を戻す前に、通信域の **DYROPTER** フィールドを 'Y' に設定する必要があります。

これは、例えば、特定の **CICS** 領域で現在実行中のトランザクションの数をカウントし続けている場合に必要になります。ただし、この再呼び出し時には、動的ルーティング・プログラムは独自のリソースしか更新できません。なぜなら、この段階では **AOR** 内のアプリケーション・プログラムから端末への最終コマンドが保留中の可能性があり、動的ルーティング・プログラムが終了しようとしているからです。

### 異常終了時の動的ルーティング・プログラムの呼び出し

**DYROPTER** を 'Y' に設定しており、ルーティングされたトランザクションが異常終了した場合、動的ルーティング・プログラムが再度呼び出されて異常終了について通知されます。トランザクションの異常終了に応じてユーザー定義プログラムを開始するために、この呼び出しを使用することができます。

ルーティングされたトランザクションが異常終了すると、**TOR** 内の **APRT** プログラムは以下を実行します。

1. トランザクションの異常終了が発生したことを示す応答を **CICS** トランザクション・マネージャーに戻します。
2. トランザクションの終了時に動的ルーティング・プログラムを再度呼び出すよう要求されていた場合 (ルーティングのための呼び出し時に **DYROPTER** を 'Y' に設定することにより)、動的ルーティング・プログラムを再度呼び出します。
3. **CICS** トランザクション・マネージャーに制御を戻します。

### 初期端末データの変更

ルーティングされたトランザクションが初期端末データを取得できなくなるため、動的ルーティング・プログラムで **EXEC CICS RECEIVE** および **EXEC CICS GDS RECEIVE** コマンドを実行してはいけません。

**CICS** 中継プログラムの **DFHAPRT** は、ユーザーの初期端末入力のコピーを、独立したバッファに置きます。この情報には、**APPC** のマップ式会話および非マップ式会話の **SNA** プレゼンテーション・サービス・ヘッダーが含まれています。このバッファのポインタ (**DYRBPNTNTR**) および長さ (**DYRBLGTH**) は、**DFHAPRT** から動的ルーティング・プログラムに渡される通信域を介して提供されます。

次の点に注意してください。

- ・**DYRBPNTNTR** が指すバッファには、メッセージの最初の要求単位 (RU) で届いたデータが含まれています。RU サイズがメッセージ全体を収容できる十分な大きさである場合、バッファには完全なメッセージが含まれています。一方、RU サイズがメッセージ長より短い場合、(バッファ自体の長さがメッセージ全体を収容できる十分な大きさであっても) バッファには最初の RU のデータのみが含まれています。
- ・**DYRBLGTH** フィールドが指す長さは、メッセージの長さであり、バッファのデータの長さではありません。メッセージ全体が単一の RU で届いた場合にのみ、**DYRBLGTH** の長さは、バッファのデータの長さになります。



- 次のすべてが真の場合、初期端末入力データはルーティング・プログラムに渡されません。

1. ルーティング・プログラムが AOR で実行されている。
2. 元の要求が、TOR からルーティングされたトランザクションである。
3. 元の機能が APPC 並列セッションである。

この時点ではトランザクション・プロファイルは照会されていないため、入力データの大文字変換は実行されていません。ただし、TYPETERM 定義で UCTRAN(YES) が指定されている場合は別です。

ユーザーによる初期データ入力を変更する必要がある場合があります (例えば、通信域の DYRTRAN フィールドを使用して、リモート・トランザクションの ID を変更する場合などに、変更が必要になることがあります)。入力データを変更するには、ルート選択で呼び出されたときに、ルーティング・プログラムで以下を実行する必要があります。

1. DYRBPNTN が指す入力データを、長さが DYRBLGTH の名前付き変数にコピーする
2. その名前付き変数のデータを変更する
3. EXEC CICS RETURN コマンドの INPUTMSG オプションを使用して、変更したデータをアプリケーション・プログラムに提供する。

EXEC CICS RETURN コマンドの INPUTMSG の使用方法については、[INPUTMSG](#) で説明するその他の方法を参照してください。INPUTMSG オプションのプログラミング情報については、[RETURN](#) を参照してください。

**注：**入力データを変更した後に、選択したトランザクションへのルーティングでエラーが発生し、動的ルーティング・プログラムが再び呼び出された場合は、元のユーザー入力を変更したことを、動的ルーティング・プログラムが「思い出す」必要があります。

### アプリケーションの通信域の変更

ルーティングされたアプリケーションの通信域の変更が必要な場合があります。例えば、ルーティング・プログラムがリモート・トランザクション ID を変更した場合に、ルーティングされたアプリケーションに渡される入力通信域の変更も必要になることがあります。

これは、ルーティング・プログラムの通信域のフィールド DYRACMAA で行うことができます。これは、アプリケーションの通信域のポインターです。

[213 ページの『アプリケーションのコンテナの変更』](#)も参照してください。

### ルーティングされたトランザクションからの情報の受信

ルーティングされたトランザクションの終わりに動的ルーティング・プログラムを再起動することを選択した場合は、出力通信域と出力 TIOA をモニタリングして、トランザクションに関する情報を動的ルーティング・プログラムで取得することができます。

### 出力通信域のモニター

ルーティングされたトランザクションは、その出力通信域の動的トランザクション・ルーティング・プログラムに情報を渡して戻すことができます。トランザクションの終了時に呼び出されると、ルーティング・プログラムは出力通信域 (DYRACMAA によってポイントされる) を調べることができます。

この機能の使用例を以下に示します。

- 機能的に同等の TOR および AOR のセットで構成される CICSplex を持ち、トランザクション・ルーティングに影響を及ぼす可能性のあるすべてのトランザクション間の親和性を識別する必要があります。CICS Interdependency Analyzer を使用してこれを識別できますが、ユーティリティでは検出できない親和性もあります (例えば、CICS 以外の機能によって作成されたもの)。また、トランザクションによっては、親和性を作成する場合と、作成しない場合があります。

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

ただし、ルーティングされたトランザクション自体は、いつ親和性が作成されたかを認識でき、動的トランザクション・ルーティング・プログラムにこれを通知できます。その後、ルーティング・プログラムはそのようなトランザクションを適宜ルーティングできます。

[213 ページの『アプリケーションのコンテナの変更』](#)も参照してください。

## 出力 TIOA のモニター

トランザクションの終了時に呼び出された場合、ルーティング・プログラムは DYRBPNTNTR によってポイントされた、ルーティングされたトランザクションの出力 TIOA のコピーを調べることができます。

これは、例えば CICSplex 内の 1 つの AOR がソフトウェア問題を発生させる状況から守る場合に役立ちます。これらは、ユーザーに対しては、トランザクション異常終了ではなく、メッセージによって報告されます。これが発生した場合、ルーティング・プログラムは失敗を認識できず、問題のある AOR を回避することはできません。出力 TIOA を読み取ると、ルーティング・プログラムは失敗の具体的な種類を示すメッセージを確認して、影響のある AOR を回避することができます。

## 処理に関する考慮事項

- ルーティング・プログラムから EXEC CICS RECEIVE 以外のすべての EXEC CICS コマンドを実行できます (202 ページの『初期端末データの変更』を参照)。トランザクションをルーティングする前にリンクが使用可能かどうかを確認したい場合は、EXEC CICS の INQUIRE CONNECTION コマンドと INQUIRE IRC コマンドが特に便利です。EXEC CICS の INQUIRE コマンドと SET コマンドについては、を参照してください。
- ルーティング・プログラムからあらゆる EXEC CICS コマンドを実行できるとはいっても、保護リソースを変更するコマンドについては、その影響を慎重に検討しなければなりません。ルーティングしたトランザクションのロジックに基づいて、そのようなリソースの変更が間違っコミットされたりバックアウトされたりすることがあるからです。APPC トランザクション・ルーティングに対する、EXEC CICS の SYNCPOINT コマンドと ABEND コマンドの影響も慎重に検討してください。
- トランザクションのルーティングに関する情報を保存したい場合は、ユーザー・ルーティング・プログラムでその処理を実行する必要があります。そのために、対象の端末に関連した一時記憶域キューに情報を書き込む、という方法が考えられます。
- いくつかのトランザクションをまとめて、ユーザーとの 1 つの会話を形成することもできます。会話の開始時点で、会話の状態を記録するためのリソースを割り振ります。そのようなリソースは、会話に含まれている最初のトランザクションのルーティング先のシステムのローカル・リソースなので、その会話が終了するまで、ルーティング・プログラムによってそのシステムへのルーティングを続けられる必要があります。
- 複雑なデジジー・チェーンが作成されないようにすることは、とても重要です。動的にルーティングするどのトランザクションについても、ルーティング元のノードに再びルーティングして輪を作るような処理を避けなければなりません。
- 動的ルーティング・プログラムの場合、RMODE は ANY でかまいませんが、AMODE は 31 でなければなりません。

## 作業単位に関する考慮事項

端末タイプに応じて、CICS 中継プログラム、動的ルーティング・プログラム、およびルーティングされたトランザクションは、単一の作業単位を構成することがあります。そのため、動的ルーティング・プログラムが所有する保護リソースは、ルーティングされたトランザクションの同期点アクティビティによって影響を受けることがあります。これは、これらのリソースが、ルーティングされたトランザクションによって意図せずにコミットまたはバックアウトされることがあることを意味します。これを防ぐには、ルーティング・プログラムのリソースを、保護された状態ではなく、無保護で定義する必要があります。

## DPL 要求の動的ルーティング

プログラム・リンク要求が動的ルーティングに対して適格である場合、リモート・プログラムをローカル・システムに対して DYNAMIC(YES) と定義するか、リモート・プログラムをローカル・システムに対して定義しないようにします。

注: SYSID なしの EXEC CICS LINK コマンドで指定されたプログラムが現時点で定義されていない場合、次に行われることは、プログラム自動インストールがアクティブであるかどうかによって異なります。

- プログラム自動インストールが非アクティブの場合には、動的ルーティング・プログラムが呼び出されません。
- プログラム自動インストールがアクティブの場合には、自動インストール・ユーザー・プログラムが呼び出されます。そして、動的ルーティング・プログラムは、次のような場合のみ呼び出されます。

- 自動インストール・プログラムにより、DYNAMIC(YES) を指定するプログラム定義がインストールされた場合。
- 自動インストール・プログラムにより、プログラム定義がインストールされなかった場合。

189 ページの『EXEC CICS LINK コマンドによって開始される自動インストール・プログラム』を参照してください。

**EXEC CICS LINK PROGRAM** コマンドによって実行される CICS-CICS 間 DPL 呼び出しと同様に、CICS の外部から受け取ったプログラム・リンク要求も動的にルーティングすることができます。例えば、次のタイプのプログラム・リンク要求はすべて動的にルーティングすることができます。

- 外部 CICS インターフェース (EXCI) クライアント・プログラムからの呼び出し
- CICS クライアント・ワークステーション製品からの外部呼び出しインターフェース (ECI) 呼び出し
- ONC/RPC 呼び出し

CICS の外側から受け取ったプログラム・リンク要求を動的にルーティングするには、次のようにします。

- CICS TS に対してプログラムを DYNAMIC(YES) と定義する
- 要求をルーティングするための動的ルーティング・プログラムをコーディングする

### 動的ルーティング・プログラムを呼び出す場合

CICS は、適格なプログラム・リンク要求の 動的ルーティング・プログラムを呼び出すことができます。

CICS は、次のような状況で動的ルーティング・プログラムを呼び出します。

- リンク先プログラムが実行される前は、次のいずれかが行われます。
  - リンクのルーティング先の領域の SYSID を取得します。

注：呼び出し側の連絡域 (COMMAREA) のアドレスがルーティング・プログラムに渡されるので、COMMAREA の内容 (適切な場合) によって要求をルーティングすることができます。

- ルーティング・プログラムに要求が静的ルーティングであることを通知します。この処理が行われるのは、プログラムが DYNAMIC(YES) と定義されている場合、またはプログラムが定義されていない一方で、呼び出し側が LINK コマンドの SYSID オプションでリモート領域の名前を指定している場合です。

この場合には、ターゲット領域の明示的な指定が、動的ルーティング・プログラムから戻された SYSID に優先します。

- 代替 SYSID を指定する際に、ルーティング選択でエラーが発生した場合。例えば、動的ルーティング・プログラムから返された SYSID を利用できないか、それが認識されない場合。あるいは指定されたターゲット領域でリンクに失敗した場合です。このプロセスは、プログラム・リンクが成功するまで、または動的ルーティング・プログラムからの戻りコードがゼロ以外になるまで、繰り返されます。戻りコードがゼロではない場合、CICS はプログラムをルーティング領域で実行しようとします。

これは特殊なケースです。注意が必要です。：

次のすべての条件に当てはまる場合、経路選択呼び出しは失敗しますが、経路選択のエラーではルーティング・プログラムは再呼び出しされません。

1. プログラムがローカル領域で定義されていない。
2. プログラムの自動インストールが、ローカル領域でアクティブでない。
3. ルート選択呼び出しで、ルーティング・プログラムがリンク要求をローカル領域にルーティングする。

したがって、ルーティング・プログラムがローカルでルーティングする可能性のある プログラム・リンク要求を動的にルーティングするには、以下のいずれかを実行する必要があります。

1. プログラム定義をローカル領域にインストールして、DYNAMIC(YES) を指定する。
  2. プログラムの自動インストールをアクティブに設定し、これを使用して DYNAMIC(YES) を指定する定義をインストールする。
- リンク要求が完了した後で、ルーティング・プログラムにより再呼び出しが要求された場合。

- リンク要求を指定のリモート・システムにシップした後で異常終了が検出された場合、ルーティング・プログラムにより再呼び出しが要求された場合。
- ルーティング・プログラムにより再呼び出しが要求された場合に、作業単位の終わりに、作業単位が完了したことを知らせる通知を発行する場合。CICSplex SM ワークロード管理では、これらの通知を使用して UOW 親和性を管理します。

199 ページの図 55 に、動的ルーティング・プログラムが起動されるポイントを示します。

## ターゲット CICS 領域の変更

初期設定では、動的ルーティング・プログラムに渡される通信域には、リンク要求のルーティング先のデフォルトの CICS 領域のシステム ID (sysid) とネット名が含まれています。これらは、インストールされたプログラム定義の REMOTESYSTEM オプションの値から派生します。REMOTESYSTEM が指定されていない場合、あるいはプログラム定義がない場合、渡される sysid とネット名は、ローカル CICS 領域のもので

す。

動的ルーティング・プログラムは、SYSID とネット名を変更できます。経路選択のために呼び出された時に変更する場合、リンク要求のルーティング先の領域は、次のように決定されます。

- NETNAME および SYSID が変更されない。

CICS は、通信域で当初指定されていたとおりの SYSID へのルーティングを試行します。

- NETNAME は変更されないが、SYSID が変更される。

CICS は新しい SYSID に対応する NETNAME で通信域を更新し、新しい SYSID への要求のルーティングを試行します。

- NETNAME は変更されるが、SYSID は変更されない。

CICS は新しい NETNAME に対応する SYSID で通信域を更新し、新しい SYSID への要求のルーティングを試行します。

- NETNAME は変更され、さらに SYSID も変更される。

CICS は新しい NETNAME に対応する SYSID で通信域を上書きし、その新しい SYSID への要求のルーティングを試行します。

プログラム定義の REMOTESYSTEM オプションでリモート領域が指名されると、ルーティング・プログラムは要求をローカルでルーティングすることができません。

DPL 要求は、IPIC 接続および ISC over SNA 接続の両方でルーティングできます。選択したターゲット領域他に対する IPIC 接続と ISC over SNA 接続の両方が存在し、それらが同じ名前である場合、IPIC 接続が優先されます。つまり、リモート SYSID "CICB" が IPCONN 定義と CONNECTION 定義の両方で定義されている場合、CICS は IPCONN 接続を使用します。

指定された NETNAME が無効であるか、見つからない場合、動的ルーティング・プログラムに SYSIDERR が返されます。別の SYSID または NETNAME を返すことによって、このエラーに対応することができます。

207 ページの『ルート選択でエラーが発生した場合』を参照してください。

ルーティング・プログラムが通知のために呼び出された場合、ルーティング・プログラムが SYSID または NETNAME を変更しても、その変更は無効です。

## プログラム名の変更

ルーティング・プログラムが経路選択のため、またはプログラム・リンク要求の通知のために呼び出された場合、通信域の DYRLPROG フィールドには、以下の手順で取得した、リンクされるプログラムの名前が含まれます。

1. インストールされたプログラム定義の REMOTENAME オプションから
2. REMOTENAME が指定されていない場合、あるいはプログラム定義がない場合は、EXEC CICS LINK コマンドの PROGRAM オプションから

ルーティングのために呼び出された場合<sup>2</sup>(静的にルーティングされた要求の通知目的ではない場合)、ルーティング・プログラムは、DYRLPROG フィールドを上書きすることによって、代替プログラムがリンクされるように指定することができます。要求のルーティング先の領域に応じて、ローカル・プログラムまたはリモート・プログラムのいずれかを指定できます。



## トランザクション ID の変更

ルーティングのために呼び出された場合 (静的にルーティングされた要求の通知目的ではない場合)、ルーティング・プログラムは、通信域の DYRTRAN フィールドを上書きすることによって、リモート・トランザクション ID を変更することができます。

トランザクション ID は、常に各動的プログラム・リンク要求に関連付けられます。CICS は、次の順序でトランザクション ID を取得します。

1. LINK オプションの TRANSID オプションから
2. プログラム定義の TRANSID オプションから
3. 「CSMI」、一般ミラー・トランザクション。TRANSID オプションのいずれも指定されていない場合は、これがデフォルトです。

**注：**CICSPlex System Manager を使用してプログラム・リンク要求をルーティングする場合には、トランザクション ID はさらに重要となります。これは、CICSPlex System Manager のルーティング論理はトランザクション・ベースだからです。CICSPlex System Manager は、対応付けられたトランザクションに対して指定されたルールに従って、各 DPL 要求をルーティングします。

CICSPlex System Manager システム・プログラマーは、EYU9WRAM というユーザーが置き換え可能なモジュールを使用して、DPL 要求に関連付けられたトランザクション ID を変更することができます。

## DPL 要求をルーティングするか終了するかについての CICS への指示

ルーティングのためにルーティング・プログラムが呼び出された場合、プログラムではリンク要求をルーティングするか拒否するかを選択することができます。値を変更したかどうかにかかわらず要求をルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。

戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された SYSID と自身のローカル SYSID を次のように比較します。

- SYSID が同じ (または、戻された SYSID がブランク) である場合、CICS はリンク要求をローカルで実行します。
- 2 つの SYSID が異なる場合、CICS は戻されたプログラム名とトランザクション名を使用して、要求をリモートの CICS 領域にルーティングします。

CICS がリンク要求を拒否するようにするには、ゼロ以外の値を返します。EXEC CICS LINK コマンドを発行したプログラムは、RESP2 値が 25 の PGMIDERR 条件を受け取ります。

ルーティング・プログラムが通知のため、または要求の終了時に呼び出された場合は、DYRRETC に値を返しても無効です。

## ルート選択でエラーが発生した場合

ルート選択でエラーが発生した場合、例えば動的ルーティング・プログラムから戻された SYSID が使用不可または不明の場合、あるいは指定されたターゲット領域でリンクが失敗した場合、動的ルーティング・プログラムが再度呼び出されます。

プログラムが再呼び出しされたら、以下のアクションのいずれかを選択できます。

- DYRRETC でゼロ以外の戻りコードを発行することで、要求のルーティングの試行を続行しないよう CICS に指示できます。
- リモート・システムが直ちに使用できるセッションがないことがエラーの原因である場合は、フィールド DYRQUEUE を「Y」に再設定し (このエラーが発生するということは、これまでは「N」 (要求はキューに入れられない) に設定されていたはず)、DYRRETC で戻りコード「0」を発行して、要求のルーティングを再試行します。
- システム ID を変更し、DYRRETC で戻りコード「0」を発行することで、要求のルーティングを再試行できます。システム ID を変更する場合、異なるリモート・プログラム名またはトランザクション ID も指定しなければならないことがあります。

---

<sup>2</sup> 「ルーティングのために呼び出された」とは、「経路選択目的の呼び出し」と「経路選択でエラーが発生したことによる呼び出し」の両方を意味します。

この要求のルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド DYRCOUNT で渡されます。このカウントを使用すると、トランザクションのルーティングの試行をいつ中止するか決定に役立ちます。

### 特殊ケースに関する注意

以下のすべてが該当する場合、ルート選択呼び出しは失敗しますが、ルート選択エラーに対してルーティング・プログラムは再呼び出しされません。

1. プログラムがローカル領域で定義されていない。
2. プログラムの自動インストールが、ローカル領域でアクティブでない。
3. ルート選択呼び出しで、ルーティング・プログラムがリンク要求をローカル領域にルーティングする。

したがって、ルーティング・プログラムがローカルでルーティングする可能性のあるプログラム・リンク要求を動的にルーティングするには、以下のいずれかを実行する必要があります。

1. DYNAMIC(YES) を指定して、ローカル・システムにプログラム定義をインストールする。
2. プログラムの自動インストールをアクティブに設定し、これを使用して DYNAMIC(YES) を指定する定義をインストールする。

### XPCERES 出口を使用したターゲット領域でのリソースの使用可否の検査

XPCERES グローバル・ユーザー出口プログラムを使用して、リンク先プログラムに必要なすべてのリソースがターゲット領域で使用可能であることを確認できます。

XPCERES 出口は (有効な場合)、動的にルーティングされるプログラム・リンク要求を CICS が処理する前に、ターゲット領域で起動されます。

例えば、ターゲット領域でリンク先プログラムが無効な場合や、必要なファイルがない場合などに、出口プログラムにより、動的ルーティング・プログラムが要求を別の領域にルーティングする機会を与えることができます。このためには、出口プログラムで戻りコード UERCRESU を設定する必要があります。こうすることにより、CICS は次のことを実行します。

1. ミラーによりターゲット領域に対して実行された EXEC CICS LINK コマンドに RESUNAVAIL 状態を戻します。(この状態は、アプリケーション・プログラムには戻されません。)
2. ルーティング・プログラムの通信域の DYRERROR フィールドを「F」(リソース使用不可) に設定します。
3. ルート選択エラーのために、ルーティング領域でルーティング・プログラムを再起動します ([207 ページの『ルート選択でエラーが発生した場合』](#)を参照)。

XPCERES グローバル・ユーザー出口プログラムの作成方法については、[プログラム制御出口 XPCREQ、XPCERES、XPCREQC](#) を参照してください。

必要なリソースがターゲット領域になく、XPCERES 出口が使用不可または無効になっている場合 (または有効であるが UERCRESU 戻りコードが設定されていない場合) には、クライアント・プログラムはエラー応答を受け取ります。

### ルーティングされた要求の終了時の動的ルーティング・プログラムの呼び出し

ルーティングされた要求の完了時に動的ルーティング・プログラムが再度呼び出されるようにするには、CICS に制御を戻す前に、通信域の DYROPTER フィールドを 'Y' に設定する必要があります。

これは、例えば、特定の CICS 領域上で現在実行されているリンク要求の数を数えている場合に必要になります。

DYROPTER を 'Y' に設定していて、リンクされたプログラムが異常終了した場合、異常終了を知らせる動的ルーティング・プログラムが呼び出されます。



## アプリケーションの入力通信域の変更

ルーティングされたアプリケーションの通信域の変更が必要な場合があります。例えば、ルーティング・プログラムがリモート・プログラムの名前を変更した場合、プログラムに渡される入力通信域も変更する必要がある場合があります。

これは、ルーティング・プログラムの通信域のフィールド `DYRACMAA` で行うことができます。これは、アプリケーションの通信域のポインターです (`LINK` コマンドで通信域が指定されていない場合は `NULL`)。

213 ページの『[アプリケーションのコンテナの変更](#)』も参照してください。

## アプリケーションの出力通信域のモニタリング

ルーティングされたアプリケーションは、その出力通信域で、動的トランザクション・ルーティング・プログラムに情報を戻すことができます。ルーティングされた DPL 要求の最後に動的ルーティング・プログラムを再起動するようにした場合は、そのプログラムで、`DYRACMAA` が指す出力通信域 (存在する場合) を検査することができます。

213 ページの『[アプリケーションのコンテナの変更](#)』も参照してください。

## 処理に関する考慮事項

動的ルーティング・プログラムでは、次のような処理に関する考慮事項があります。

- プログラム・リンク要求で呼び出された場合、動的ルーティング・プログラムは `EXEC CICS` コマンドの使用を、DPL サブセット内に限定する必要があります。どのコマンドが DPL サブセットを構成するかについては、[LINK コマンドの例外条件](#)を参照してください。
- ルーティング・プログラムは、DPL サブセットのどの `EXEC CICS` コマンドも発行できますが、保護されているリソースを変更するコマンドの影響については注意が必要です。これは、これらのリソースへの変更は、ルーティングされたプログラムのロジックの結果として意図せずコミットまたはバックアウトされる場合があるためです。
- リンク要求のルーティングに関する情報を保持するには、情報を一時記憶域キューに書き込むなどして、これをユーザー・ルーティング・プログラムで行う必要があります。
- "複雑なデ이지ー・チェーン"を作成することは避けてください。動的にルーティングされたプログラム・リンク要求では、以前のルーティング元であったノードにルーティングすることは避けてください。詳しくは、[DPL 要求のデ이지ー・チェーン](#)を参照してください。
- 動的ルーティング・プログラムは `RMODE ANY` である場合がありますが、`AMODE 31` である必要があります。

## 作業単位に関する考慮事項

クライアント・プログラム、動的ルーティング・プログラム、および場合によってはサーバー・プログラムは、単一の作業単位を構成します。そのため、動的ルーティング・プログラムが所有するリカバリー可能リソースは、クライアント・プログラムの同期点アクティビティーによって影響を受けることがあります。これは、これらのリソースが、クライアント・プログラムによって意図せずにコミットまたはバックアウトされることがあることを意味します。これを防ぐには、ルーティング・プログラムのリソースを、リカバリー不能として定義する必要があります。

DPL クライアントおよびサーバー・プログラムの同期点アクティビティーについては、[サーバー・プログラム](#)を参照してください。

## ブリッジ要求の動的ルーティング

ブリッジの制御下で 3270 ユーザー・トランザクションを実行するためには、クライアント・プログラムはまず、ブリッジ・ルーター領域で実行中の `DFHL3270` に `LINK`、`ECI`、または `EXCI` 呼び出しを発行して、ブリッジ・インバウンド・メッセージ・ヘッダー (`BRIH`) を含む `COMMAREA` を渡す必要があります。

`BRIH` には、ターゲット・ユーザー・トランザクションの名前が含まれます。`DFHL3270` (ブリッジ・プログラム) はそれを受けて、`CICS` ドライバー・プログラムにリンクして `COMMAREA` を渡します。ユーザー・トランザクションが動的ルーティングの対象となる場合、`DFHL3270` は、ドライバー・プログラムが実行されるターゲット・システムを決定するための動的ルーティング・プログラムを呼び出します。

ユーザー・トランザクションは常に、ドライバー・プログラムと同じ領域で実行されます。ユーザー・トランザクションではなく、ユーザー・トランザクションを実行するためのクライアント要求が動的にルーティングされます。

ドライバー・プログラムに対するブリッジ要求が動的ルーティングの対象となるかどうかは、ルーター領域でのターゲット・トランザクションのリソース定義を使用して判別されます。ルーター領域でターゲット・ユーザー・トランザクションが定義されていなければ、DTRTRAN システム初期設定パラメーターで指定された共通トランザクション定義を使用して、要求が動的ルーティングの対象となるかどうか判別されます。

セッション・モードでは、最初のユーザー・トランザクション要求のターゲット・システムが、セッション内の後続のすべてのユーザー・トランザクション要求をどこにルーティングするかを決定します。リモート要求は、MRO リンクでルーター領域に接続された他の領域にルーティングすることも、APPC (LUTYPE6.2) ISC リンクで接続された他のシステムにルーティングすることもできます。

**注:** ローカル・システムとは、動的ルーティング・プログラムが実行されている CICS ルーター領域のことです。

動的ルーティング・プログラムは、以下の場合に呼び出されます。

- 単一トランザクション・モードで、トランザクションが DYNAMIC(YES) と定義されているか、またはトランザクションが定義されておらず DTRTRAN トランザクションが DYNAMIC(YES) と定義されている場合。
- セッション・モードで、最初のユーザー・トランザクションが DYNAMIC(YES) と定義されているか、またはトランザクションが定義されておらず DTRTRAN トランザクションが DYNAMIC(YES) と定義されている場合。
- セッション・モードで、後続のユーザー・トランザクションが DYNAMIC(YES) と定義されているか、またはトランザクションが定義されておらず DTRTRAN トランザクションが DYNAMIC(YES) と定義されている場合。この場合、セッションの最初のユーザー・トランザクションによってターゲット・システムが既に決まっているので、通知のためにのみルーティング・プログラムが呼び出されます。ルーティング・プログラムが要求のターゲット・システムを変更することはできません。
- ルート選択でエラーが発生した場合、例えば、初期(ルート選択)呼び出しでルーティング・プログラムから戻されたターゲット領域が使用不可の場合。この場合、ルーティング・プログラムは代替ターゲットを指定できます。このプロセスは、ルーティング・プログラムが使用可能ターゲットを選択するか、ゼロ以外の戻りコードを設定するまで繰り返されます。
- ユーザー・トランザクションが完了した後、終了時に再呼び出しするようルーティング・プログラムが要求した場合。

### ブリッジ要求パラメーターの変更

初期設定では、動的ルーティング・プログラムに渡される通信域には、一部変更可能なパラメーターやポインターが含まれます。

初期設定では、動的ルーティング・プログラムに渡される通信域には、検査可能なパラメーターやポインターが含まれます。これらはすべて、213 ページの『動的ルーティング・プログラムに渡されるパラメーター』で説明されています。Link3270 ブリッジ要求で変更できるのは、次のパラメーターのみです。

- 要求のルーティング先の CICS 領域のシステム ID (SYSID) および ネット名
- Link3270 ブリッジの制御下で実行されるターゲット・ユーザー・アプリケーションのトランザクション ID (TRANSID)
- AOR のユーザー・トランザクションのディスパッチャーの優先順位
- タスク・ローカル・ユーザー・データ域。
- DTRTRAN 標識
- 終了オプション

### Link3270 ブリッジ要求 SYSID の変更

要求のルーティング先のデフォルトの CICS 領域の SYSID およびネット名の初期値は、インストールされたユーザー・プログラム定義の REMOTESYSTEM オプションの値から派生します。REMOTESYSTEM が指

定されていない場合、あるいはプログラム定義がない場合、渡される sysid とネット名は、ローカル CICS 領域のものです。

要求のルーティング先の領域は、次のように決定されます。

- NETNAME および SYSID が変更されない。

CICS は、通信域で当初指定されていたとおりの SYSID へのルーティングを試行します。

- NETNAME は変更されないが、SYSID が変更される。

CICS は新しい SYSID に対応する NETNAME で通信域を更新し、新しい SYSID への要求のルーティングを試行します。

- NETNAME は変更されるが、SYSID は変更されない。

CICS は新しい NETNAME に対応する SYSID で通信域を更新し、新しい SYSID への要求のルーティングを試行します。

- NETNAME は変更され、さらに SYSID も変更される。

CICS は新しい NETNAME に対応する SYSID で通信域を上書きし、その新しい SYSID への要求のルーティングを試行します。

指定された NETNAME が無効であるか、見つからない場合、動的ルーティング・プログラムに SYSIDERR が返されます。別の SYSID または NETNAME を返すことによって、このエラーに対応することができます。[211 ページの『Link3270 ブリッジ要求のルート選択エラーの処理』](#)を参照してください。

戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された SYSID と自身のローカル SYSID を次のように比較します。

- SYSID が同じ (または、戻された SYSID がブランク) である場合、CICS は リンク要求をローカルで実行します。
- 2 つの SYSID が異なる場合、CICS は 戻されたトランザクション名を使用して、要求をリモートの CICS 領域にルーティングします。

### ブリッジ要求 TRANSID の変更

ターゲット・ユーザー・トランザクションの TRANSID は、DYRTRAN の動的ルーティング・プログラムに渡されます。通信域の DYRTRAN フィールドを上書きすることで、これを変更できます。

### Link3270 ブリッジ要求のトランザクション優先順位の変更

ユーザー・トランザクションのディスパッチング 優先順位は、DYRPRTY で優先順位を指定して DYRRTPRI で“Y”を指定することで変更できます。この優先順位は、AOR の TRANSACTION リソース定義で指定された優先順位を指定変更します。

### Link3270 ブリッジ要求の拒否

ルーティングのためにルーティング・プログラムが呼び出された場合、プログラムではリンク要求をルーティングするか拒否するかを選択することができます。値を変更したかどうかにかかわらず要求をルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。

ルーティング・プログラムは、フィールド DYRRETC で値 4 または 8 を戻すことで、要求を拒否できます。

クライアントに戻された BRIH には、ルーティング・プログラムが要求を拒否したことを示す戻りコードの値が含まれます。BRIH 完了コードに、ルーティング・プログラムによる要求のルーティングの最後の試行に関する情報が示されています。ルーティング・プログラムでフィールド DYRRETC に戻りコード値 8 が配置された場合、最後に試行された要求のルーティングの詳細が含まれるメッセージが発行されます。

ルーティング・プログラムが要求の終了時に呼び出された場合、または通知が呼び出された場合は、DYRRETC に値を戻しても無効です。

### Link3270 ブリッジ要求のルート選択エラーの処理

ルート選択でエラーが発生した場合、例えば動的ルーティング・プログラムから戻された SYSID が使用不可または不明の場合、あるいは指定されたターゲット領域でリンクが失敗した場合、動的ルーティング・プログラムが再度呼び出されます。

この場合は、以下のアクションのいずれかを選択できます。

- DYRRETC でゼロ以外の戻りコードを発行することで、要求のルーティングの試行を続行しないよう CICS に指示できます。
- システム ID を変更し、DYRRETC で戻りコード「0」を発行することで、要求のルーティングを再試行できます。なお、システム ID を変更する場合は、異なるトランザクション ID も指定しなければならないことがあります。

この要求のルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド DYRCOUNT で渡されます。このカウントを使用すると、トランザクションのルーティングの試行をいつ中止するか決定に役立ちます。

### **XPCERES 出口を使用したターゲット領域でのリソースの使用可否の検査**

XPCERES グローバル・ユーザー出口プログラムを使用して、3270 ユーザー・トランザクションに必要なすべてのリソースがターゲット領域で使用可能であることを確認できます。

この出口は (有効な場合)、動的にルーティングされる Link3270 ブリッジ要求を CICS が処理する前に、ターゲット領域で起動されます。

例えば、ターゲット領域で 3270 ユーザー・トランザクションが無効な場合や、必要なファイルがない場合などに、出口プログラムにより、動的ルーティング・プログラムが要求を別の領域にルーティングする機会を与えることができます。このためには、出口プログラムで戻りコード UERCRESU を設定する必要があります。こうすることにより、CICS は次のことを実行します。

1. ミラーによりターゲット領域に対して実行された DFHL3270 への EXEC CICS LINK 呼び出しに RESUNAVAIL 状態を戻します。
2. ルーティング・プログラムの通信域の DYRERROR フィールドを「F」(リソース使用不可) に設定します。
3. ルート選択エラーのために、ルーティング領域でルーティング・プログラムを再起動します ([211 ページの『Link3270 ブリッジ要求のルート選択エラーの処理』](#)を参照)。

XPCERES グローバル・ユーザー出口プログラムの作成方法については、[プログラム制御出口 XPCREQ、XPCERES、XPCREQC](#) を参照してください。

必要なリソースがターゲット領域になく、XPCERES 出口が使用不可または無効になっている場合 (または有効であるが UERCRESU 戻りコードが設定されていない場合) には、クライアント・プログラムはエラー応答を受け取ります。

### **Link3270 ブリッジ要求の後の動的ルーティング・プログラムの再呼び出し**

ルーティングされた要求の完了時に動的ルーティング・プログラムが再度呼び出されるようにするには、CICS に制御を戻す前に、通信域の DYROPTER フィールドを 'Y' に設定する必要があります。

これは、例えば、特定の CICS 領域上で現在実行されているリンク要求の数を数えている場合に必要になります。

DYROPTER を 'Y' に設定していて、リンクされたプログラムが異常終了した場合、異常終了を知らせる動的ルーティング・プログラムが呼び出されます。

### **Link3270 ブリッジの動的ルーティングに関する考慮事項**

動的ルーティング・プログラムには、次のような Link3270 ブリッジに関する考慮事項があります。

- DTRTRAN 定義を使用して Link3270 要求をルーティングする場合、ルーティング・プログラムは通信域の DYRTTRRJ フィールドを N (デフォルトは Y) に設定する必要があります。DYRTTRRJ を Y の設定のままにすると、要求は拒否されます。DYRDTRXN フィールドをテストして、ルーティング・プログラムに渡されたトランザクションが DTRTRAN 定義によって定義済みかどうか確認できます。
- Link3270 ブリッジ要求で呼び出された場合、動的ルーティング・プログラムは EXEC CICS コマンドの使用を、DPL サブセット内に限定する必要があります。どのコマンドが DPL サブセットを構成するかについては、[LINK コマンドの例外条件](#)を参照してください。
- ルーティング・プログラムは、DPL サブセットのどの EXEC CICS コマンドも発行できますが、保護されているリソースを変更するコマンドの影響については注意が必要です。これは、これらのリソースへの変更は、ルーティングされたプログラムのロジックの結果として意図せずコミットまたはバックアウトされる場合があるためです。



- リンク要求のルーティングに関する情報を保持するには、情報を一時記憶域キューに書き込むなどして、これをユーザー・ルーティング・プログラムで行う必要があります。
- 動的ルーティング・プログラムは RMODE ANY である場合がありますが、AMODE 31 である必要があります。

## アプリケーションのコンテナの変更

このセクションは、次のルーティングに適用されます。

- 端末関連の START 要求によって開始されたトランザクション ([197 ページの『トランザクションの動的ルーティング』](#)で説明しています)
- プログラム・リンク (DPL) の要求 ([204 ページの『DPL 要求の動的ルーティング』](#)で説明しています)
- 非端末関連の START 要求 ([231 ページの『非端末関連 START 要求のルーティング』](#)で説明しています)

ユーザー・アプリケーションが通信域ではなくチャンネルを使用する場合、ルーティング・プログラムはフィールド DYRCHANL でそのチャンネルの名前を受け取ります。ルーティング・プログラムにはチャンネルのアドレスではなく、名前が渡されるので、DYRCHANL の内容を使用してチャンネルのコンテナの内容を検査または変更することはできません。

ただし、チャンネルを使用するアプリケーションは、チャンネル内に DFHROUTE という名前の特殊なコンテナを作成できます。アプリケーションが LINK 要求、または動的にルーティングされる 端末関連の START 要求 (しかし、非端末関連の START 要求ではない) 場合、動的ルーティング・プログラムに、DFHDYPDS の DYRACMAA フィールドで DFHROUTE コンテナのアドレスが指定され、内容を検査および変更できます。

## ユーザー ID によるルーティング

オブションで、ルーティング・プログラムは、要求に関連付けられた CICS ユーザー ID (userid) によって要求をルーティングすることができます。通信域の DYRUSERID フィールドには、ユーザー ID が含まれています。ルーティングのため、またはルート選択エラーのために起動された場合、ルーティング・プログラムはこのフィールドの内容に基づいてルーティングを決定できます。

さまざまなタイプの要求でユーザー ID が設定される方法について詳しくは、[213 ページの『動的ルーティング・プログラムに渡されるパラメーター』](#)で DYRUSERID フィールドの説明を参照してください。

## 動的ルーティング・プログラムに渡されるパラメーター

パラメーターは、通信域 (COMMAREA) またはコンテナを使用して、CICS 中継プログラムから動的ルーティング・プログラムに渡されます。コピーブック DFHDYPDS によって COMMAREA またはコンテナがマップされます。これは、サポートされるすべてのプログラミング言語を対象に、該当する CICS ライブラリーにあります。

動的ルーティング・プログラムと分散ルーティング・プログラムの両方に、同じ情報が渡されます。一方のルーティング・プログラムにとっては意味があっても、もう一方にとっては意味がないパラメーターもあります。一方のルーティング・プログラムには渡されるものの、もう一方には渡されることがないパラメーター値もあります。以下のリストでは、動的ルーティング・プログラムで有効なパラメーターに限り詳しく説明しています。動的ルーティング・プログラムに渡されないパラメーター値はリストされていません。例えば、DYRFUNC パラメーターでは、値 X'5' がリストされていません。X'5' は、分散ルーティング・プログラムに対するルート開始呼び出し時にのみ使用されるため、動的ルーティング・プログラムに渡されることはありません。

動的ルーティング・プログラムと分散ルーティング・プログラムの両方として同じプログラムを使用する場合、分散ルーティングの呼び出しを使用するときに有効なパラメーターと値の説明については、[239 ページの『分散ルーティング・プログラムに渡されるパラメーター』](#)を参照してください。

### DYRABCDE

ルーティングされたトランザクションまたはプログラム・リンク要求が異常終了するか、または Link3270 ユーザー・トランザクションが異常終了したときに戻される、異常終了コード。

### DYRABNLC

異常イベント・コード、またはヌル。

このパラメーターは、ルーティングされた要求を停止するために動的ルーティング・プログラムが呼び出されるときに有効です。ヌル以外の値は、異常イベント (トランザクションの異常終了を除く) が、

要求のルーティング先である領域で発生したことを表します。エラーの原因が調べられて修正されるまで、ルーティング・プログラムは同じ領域にさらに要求をルーティングしてはなりません。

このフィールドは CICSplex System Manager で使用するためのものです。現時点では、リソース・マネージャーとしての Db2、IMS、IBM MQ、または VSAM RLS への接続が使用できないことの結果として設定されます。詳しくは、[ストーム・ドレーン作用の回避](#)を参照してください。

## DYRACMAA

このフィールドは、以下の項目のルーティングに適用されます。

- 端末で開始されたトランザクション
- 端末関連の START コマンドで開始されるトランザクション
- プログラム・リンク (DPL) 要求

これらのタイプの要求のルーティングでは、以下のエントリーのいずれかが DYRACMAA に含められます。

- アプリケーションの通信域 (COMMAREA) の 31 ビット・アドレス。ユーザー・アプリケーションが COMMAREA を使用する場合、またはトランザクション・ルーティングを使用する場合で、最初のトランザクションによって **EXEC CICS RETURN** コマンドでチャンネルまたは COMMAREA が指定されているときに含められます。
- DFHROUTE コンテナの 31 ビット・アドレス。ユーザー・アプリケーションがチャンネルを使用し、そのチャンネルに DFHROUTE というコンテナを作成した場合に含められます。
- ヌル文字。ユーザー・アプリケーションに COMMAREA も DFHROUTE コンテナもない場合に含められます。

その他すべてのタイプの要求のルーティングでは、DYRACMAA にヌル文字が含まれます。

リストされた 3 タイプの適格な要求のルーティングにおいて、ユーザー・アプリケーションが COMMAREA を使用する場合は、動的ルーティング・プログラムの呼び出し方法に応じてアドレスが異なります。

- ルーティング (DYRFUNC=0) のために動的ルーティング・プログラムが呼び出された場合、入力通信域のアドレス (使用可能なものがある場合)。同様に、ルート選択エラー (DYRFUNC=1) または通知 (DYRFUNC=3) のためにルーティング・プログラムが呼び出された場合も、アドレスは入力通信域のアドレスになります。
- 前にルーティングされたトランザクションまたはリンク 要求が正常に終了した (DYRFUNC=2) のためにルーティング・プログラムが呼び出された場合、出力通信域のアドレス (使用可能なものがある場合)。ルーティングされたアプリケーションは、出力通信域を使用して、情報を動的ルーティング・プログラムに渡すことができます。

トランザクションをルーティングしており、ユーザー・アプリケーションがチャンネルを使用する場合は、チャンネルのアドレスではなく名前がルーティング・プログラムに渡されます。そのため、DYRCHANL パラメーターを使用してコンテナの内容を検査および変更することはできません。

ルーティングされたトランザクションの異常終了 (DYRFUNC=4) のためにルーティング・プログラムが呼び出された場合、通信域または DFHROUTE コンテナのいずれの情報も有効ではありません。

ルーティング・プログラムは、DYRACMAA によってアドレス指定された、いずれかのアプリケーションの通信域または DFHROUTE コンテナに含まれるデータを変更できます。

## DYRACMAL

以下の項目のルーティングに適用されます。

- 端末で開始されたトランザクション
- 端末関連の START コマンドで開始されるトランザクション
- プログラム・リンク (DPL) 要求

これらのタイプの要求のルーティングでは、以下の数値のいずれかが DYRACMAL に含められます。

- アプリケーション COMMAREA の長さ (バイト単位)。ユーザー・アプリケーションが COMMAREA を使用する場合に含められます。



- DFHROUTE コンテナ内のデータの長さ (バイト単位)。ユーザー・アプリケーションがチャネルを使用し、そのチャネルに DFHROUTE というコンテナを作成した場合に含められます。
- ゼロ。ユーザー・アプリケーションに COMMAREA も DFHROUTE コンテナもない場合に含められます。

その他すべてのタイプの要求のルーティングでは、DYRACMAL にゼロが含まれます。

#### **DYRACTCMP**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRACTID**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRACTN**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRAPPLICATION**

アプリケーション・コンテキストのアプリケーション名。アプリケーション・コンテキストを使用できない場合、またはアプリケーション・コンテキストを使用できても、アプリケーション名が設定されていない場合は、ヌルに設定されます。

#### **DYRAPPLMAJOR**

アプリケーション・コンテキストのアプリケーション・メジャー・バージョン。アプリケーション・コンテキストを使用できない場合は 0 に設定され、アプリケーション・コンテキストを使用できても、アプリケーション・メジャー・バージョンが設定されていない場合は -1 に設定されます。

#### **DYRAPPLMICRO**

アプリケーション・コンテキストのアプリケーション・マイクロ・バージョン。アプリケーション・コンテキストを使用できない場合は 0 に設定され、アプリケーション・コンテキストを使用できても、アプリケーション・マイクロ・バージョンが設定されていない場合は -1 に設定されます。

#### **DYRAPPLMINOR**

アプリケーション・コンテキストのアプリケーション・マイナー・バージョン。アプリケーション・コンテキストを使用できない場合は 0 に設定され、アプリケーション・コンテキストを使用できても、アプリケーション・マイナー・バージョンが設定されていない場合は -1 に設定されます。

#### **DYRAPPLVER**

アプリケーション・コンテキストのアプリケーション・バージョン。すべてのバージョン番号が、アプリケーション・コンテキストを使用できない場合は 0 に設定され、アプリケーション・コンテキストを使用できても、アプリケーション・バージョンが設定されていない場合は -1 に設定されます。

#### **DYRBLGTH**

TIOA DFHLUC バッファのコピーの長さ。

このフィールドは、動的トランザクション・ルーティングまたは Link3270 要求にのみ適用されます (プログラム・リンク要求のルーティングは適用外)。

#### **DYRBPNTNTR**

TIOA LUC バッファのコピーの 31 ビット・アドレス。

このフィールドは、動的トランザクション・ルーティングにのみ適用され、プログラム・リンク要求のルーティングには適用されません。

ルーティング、ルート選択エラー (DYRFUNC=0)、または通知 (DYRFUNC=3) のために動的ルーティング・プログラムが呼び出された場合、入力 TIOA のコピーが渡されます。ルーティング・プログラムは、ルーティングされたトランザクションに渡された端末入力データを変更できます。[202 ページの『初期端末データの変更』](#)を参照してください。

前にルーティングされたトランザクションが正常に終了した (DYRFUNC=2) ためにルーティング・プログラムが呼び出された場合、出力 TIOA のコピーが渡されます。ルーティング・プログラムは、出力 TIOA をモニターして、AOR 内の問題を検出することができます。[203 ページの『ルーティングされたトランザクションからの情報の受信』](#)を参照してください。

Link3270 ブリッジ要求 (DYRTYPE=8) のためにルーティング・プログラムが呼び出された場合、TIOA LUC バッファのコピーのアドレスは DYRBPNTN で渡されません。

#### DYRBRNK

Link3270 ブリッジ要求に関連付けられた、8 バイトのブリッジ・ファシリティー・トークン。このフィールドは、DYRTYPE=8 の場合にのみ有効です。

#### DYRCABP

CICS に標準の異常終了処理を続行させるかどうかを示します。

このフィールドは、動的トランザクション・ルーティングにのみ適用され、プログラム・リンク要求および Link3270 要求のルーティングには適用されません。リンク先プログラムがリモート領域で異常終了した場合、その異常終了はローカル領域に反映されます。つまり、**EXEC CICS LINK** コマンドを発行したプログラムにそれが渡されます。

#### Y

CICS 異常終了処理を続行します。

#### N

トランザクションを停止し、CICS 異常終了処理を続行せず、DYRLPROG で指定されたプログラムに制御を渡します。

制御された方法でこの状態を処理し、端末ユーザーに適切なメッセージを発行できるローカル・プログラムに制御を渡すには、このオプションを使用します。

N を入力する場合は、DYRLPROG がローカル・システム上の有効なプログラムの名前を指定していることを確認してください。

DYRCABP に適用されるデフォルト値はありません。

#### DYRCHANL

プログラム・リンクまたは START コマンドにチャンネルが関連付けられていれば、そのチャンネルの名前です。このフィールドは、DPL 要求、非端末関連 START 要求、および端末関連 START 要求によって開始されたトランザクションのルーティングにのみ適用されます。他のタイプの要求の場合、またはコマンドにチャンネルが関連付けられていない場合、このフィールドにはブランクが入ります。

ルーティング・プログラムにはチャンネルのアドレスではなく名前が渡されるため、このフィールドの内容を使用してコンテナの内容を検査または変更することはできないことに留意してください。ルーティング・プログラムがアプリケーション・コンテナの内容を検査または変更する方法については、213 ページの『[アプリケーションのコンテナの変更](#)』と DYRACMAA フィールドの説明を参照してください。

#### DYRCLOUD

アプリケーション・コンテキストのクラウド・ルーティング・データ。これは、他のすべてのアプリケーション・コンテキスト・フィールドをカプセル化するコンテナです。デフォルトで、またアプリケーション・コンテキストを使用できない場合に、ヌルに設定されます。

#### DYRCOMP

CICS コンポーネント・コード。動的ルーティング・プログラムの呼び出しの場合は、常に RT に設定されます。

#### DYRCOUNT

DYRFUNC が 0、1、または 3 に設定された、このトランザクションまたはリンク要求で動的ルーティング・プログラムが呼び出された回数。このフィールドを使用して、プログラムが要求のルーティングを試行する回数を制限します。

#### DYRDTRR

**DTRTRAN** システム初期設定パラメーターで指定された共通トランザクション定義で定義されたトランザクションの処理を拒否するか受け入れるかを示します。

このフィールドは、動的トランザクション・ルーティングおよび Link3270 要求のルーティングにのみ適用され(プログラム・リンク要求のルーティングは適用外)、DYRTRXN が Y に設定されている場合にのみ該当します。

以下の値が有効です。

**Y**

トランザクションは拒否されます。Yがデフォルトです。

**N**

トランザクションは拒否されません。

動的ルーティング・プログラムが呼び出されると、このパラメーターは必ず拒否状態に設定されます。**DTRTRAN** システム初期設定パラメーターによって定義されたトランザクションを動的にルーティングするには、この標識を受け入れ状態に変更する必要があります。

トランザクションを拒否した場合、動的トランザクション・ルーティングでは、メッセージ DFHAC2001、「トランザクション *tranid* を認識できません。(Transaction *tranid* is unrecognized.)」がユーザーの端末に送信されます。Link3270 要求では、クライアントに戻された BRIH に、トランザクションが見つからなかったことを示す戻りコードと、**DTRTRAN** システム初期設定パラメーターで指定されたトランザクションをルーティング・プログラムが拒否したことを示す完了コードが含まれます。

#### **DYRDTRXN**

ルーティングするトランザクションが、**DTRTRAN** システム初期設定パラメーターで指定された共通トランザクション定義によって定義されたものか、それとも特定のトランザクション定義によって定義されたものかを示します。

このフィールドは、動的トランザクション・ルーティングおよび Link3270 要求にのみ適用され、プログラム・リンク要求のルーティングには適用されません。

以下の値が有効です。

**Y**

トランザクションは、**DTRTRAN** システム初期設定パラメーターで指定された定義によって定義されます。つまり、入力トランザクション ID に対するリソース定義はありません。

動的トランザクション・ルーティングの場合、トランザクションは、**DTRTRAN** システム初期設定パラメーターで指定されたトランザクション ID を使用する端末専有領域で開始されます。

動的トランザクション・ルーティングの場合、入力トランザクション ID は、DYRTRAN フィールドで動的ルーティング・プログラムに渡されます。Link3270 要求の場合、共通トランザクション定義を使用して、要求のルーティング特性が決定されます。要求には、共通トランザクション ID ではなく、元のトランザクション ID がまだ入っています。要求がローカルで実行される場合、要求は正常にドライバに渡されますが、ユーザー・トランザクションが定義されていないためドライバはユーザー・トランザクションの開始に失敗します。

**N**

トランザクションは、**DTRTRAN** システム初期設定パラメーターで指定された定義によって定義されません。入力トランザクション ID のためにインストールされたリソース定義が存在します。

動的トランザクション・ルーティングの場合、トランザクションは、入力トランザクション ID を使用する端末専有領域で開始されます。DYRTRAN フィールドで動的ルーティング・プログラムに渡されるトランザクション ID は、トランザクション・リソース定義からのリモート・トランザクション ID です (この ID が入力トランザクション ID と異なる場合)。

Link3270 要求の場合、DYRTRAN フィールドでルーティング・プログラムに渡されるトランザクション ID は、TRANSACTION リソース定義に定義されているリモート・トランザクション ID です。

#### **DYRERROR**

DYRFUNC が 1 に設定されている場合にのみ、値があります。DYRERROR は、ルート選択を最後に試行していたときに発生したエラーのタイプを示します。IPIC 接続でのルーティングの試行に失敗し、同じ名前の接続を使用したその後の試行も失敗した場合 (SYSID 不検出以外の理由で)、その接続でのルーティングの試行で発生したエラーのタイプが戻されます。以下の値が有効です。

**0**

選択された SYSID が不明です。

**1**

選択されたシステムはサービス中ではありません。

2

選択されたシステムはサービス中ですが、使用可能なセッションがありません。

3

割り振り要求が拒否され、SYSIDERR がアプリケーション・プログラムに戻されました。このエラーは、以下のいずれかの理由で発生します。

- XZIQUE グローバル・ユーザー出口プログラムが、割り振りの拒否を要求した。
- CONNECTION リソース定義で指定された QUEUELIMIT 値に達したため、CICS が割り振り要求を自動的に拒否した。

4

割り振り要求のキューがパージされており、SYSIDERR が待機中のすべてのアプリケーション・プログラムに戻されました。このエラーは、以下のいずれかの理由で発生します。

- XZIQUE グローバル・ユーザー出口プログラムが、キューのパージを要求した。
- CONNECTION リソース定義で指定された MAXQTIME 限度に達したため、CICS がキューを自動的にパージした。

5

選択されたシステムはこの機能をサポートしていません。この値は、ルーティング・プログラムが以下のいずれかのアクションを実行しようとした場合に設定されます。

- **EXEC CICS START** コマンドによって開始されたトランザクションを、MRO または APPC 並列セッション・リンクで接続されていない領域にルーティングする。
- LU6.1 接続でトランザクションか、プログラム・リンク要求または Link3270 要求をルーティングする。
- Link3270 要求を、サポートされないリリースの CICS の領域にルーティングする。
- IPIC 接続でトランザクションを CICS TS for z/OS バージョン 4.1 より前の領域にルーティングする。
- IPIC 接続で APPC デバイスをルーティングする。

値 6 から B はすべて、プログラム・リンク要求のルーティングの試行に適用されます。これらのエラー状態の意味については、[LINK](#) を参照してください。

6

**EXEC CICS LINK** コマンドが LENGERR を戻しました。

7

**EXEC CICS LINK** コマンドが PGMIDERR を戻しました。

8

**EXEC CICS LINK** コマンドが INVREQ を戻しました。

9

**EXEC CICS LINK** コマンドが NOTAUTH を戻しました。

A

**EXEC CICS LINK** コマンドが TERMERR を戻しました。

B

**EXEC CICS LINK** コマンドが ROLLEDBACK を戻しました。

F

ターゲット領域にある XPCERES グローバル・ユーザー出口プログラムは、UERCRESU の戻りコードを設定しました。これは、必要なリソースがターゲット領域で使用できないことを意味します。このエラー・コードは、プログラム・リンク、Link3270 ブリッジ、および非端末関連の START 要求の場合に設定されます。

## DYRFUNC

動的ルーティング・プログラムの今回の呼び出しの理由を示します。以下の値が有効です。

0

ルート選択のために呼び出されました。

1

ルート選択でエラーが発生したために呼び出されました。

2

前にルーティングされたトランザクションまたはプログラム・リンク要求が正常に終了したために呼び出されたか、あるいはユーザー・トランザクションが正常に終了した要求のために呼び出されました。

3

静的にルーティングされた要求の宛先を通知するために呼び出されました。この通知が該当するケースは以下のとおりです。

#### ATI 要求

DYNAMIC(YES) として定義されたトランザクションが、端末関連の自動トランザクション開始 (ATI) 要求によって (例えば、インターバル制御機能の開始要求の有効期限によって) 開始されたものの、このトランザクションが動的ルーティングに不適格な場合。

端末関連の **EXEC CICS START** コマンドによって開始される、動的ルーティングに適格なトランザクションについては、START コマンドで呼び出されたトランザクションのルーティングを参照してください。

#### プログラム・リンク要求

プログラムが DYNAMIC(YES) と定義されているか、またはプログラムが定義されていない一方で、呼び出し側が **EXEC CICS LINK** コマンドの SYSID オプションでリモート領域の名前を指定した場合。

この場合には、ターゲット領域の明示的な指定が、動的ルーティング・プログラムから戻された SYSID に優先します。

#### ブリッジ要求

セッション・モードで、要求されたトランザクションが最初のユーザー・トランザクションではなく、DYNAMIC(YES) と定義されている場合。

4

ルーティングされたトランザクション、または要求されたユーザー・トランザクションが異常終了したために呼び出されました。

7

作業単位の終了を処理するための呼び出しを示すために呼び出されました。

DYRTYPE フィールドは、ルーティング要求または通知要求のタイプを通知します。

#### DYRLEVEL

ルーティングされた要求をターゲット AOR が正常に処理するために必要な、CICS のレベル。以下の値が有効です。

##### X'00'

現在サポートされているすべてのバージョンの CICS が要求を処理できます。

##### X'03'

CICS TS は CICS TS for z/OS バージョン 3.1 以上でなければなりません。この値は、以下の要求の場合に設定されます。

- チャンネルが関連付けられている DPL 要求。
- チャンネルが関連付けられている START 要求。
- (分散ルーティング・プログラムによって処理される) インバウンド Web サービス要求。

このパラメーターは、複数領域の論理サーバーの「ローリング・アップグレード」を実行するときに役立ちます。これにより、サーバーをダウンさせずに、一度に 1 つの領域が CICS のあるリリースから次のリリースにアップグレードされます。特定のレベルの CICS を必要とする要求を適切な AOR に経路指定することができます。

同じ論理サーバー内の個々の CICS 領域がそれぞれ CICS の異なるレベルとなる、こうした混合レベルの運用は、ローリング・アップグレードでのみ利用されることを目的としています。この運用は、一部のインターオペラビリティ・シナリオで障害の発生リスクを高めるため、恒久的に利用するための



ものではありません。推奨される通常の運用方式では、論理サーバー内の全領域を同レベルの CICS および Java にします。

## DYRLPROG

ルーティングされるトランザクションの最初のプログラムの名前か、またはルーティングされるリンク・コマンドで指定されたプログラムの名前。

### トランザクション・ルーティング

トランザクションをローカルでルーティングする場合には、このフィールドを使用して、代わりに実行するプログラムの名前を指定できます。例えば、すべてのリモート CICS 領域が使用不能なため、トランザクションをルーティングできない場合、ローカル端末専有領域でプログラムを実行して、適切なメッセージをユーザーに送信したい場合があります。

DYRCABP=N を指定した場合は、DYRLPROG をブランクに設定しないでください。DYRCABP=N を指定した場合は、DYRLPROG で有効なプログラム名を必ず指定してください。

### プログラム・リンク要求

DYRFUNC が 0 または 3 に設定されている場合、DYRLPROG には、以下の順序で取得されたリンク対象のプログラムの名前が含まれます。

1. インストールされたプログラム定義の REMOTENAME オプションから。
2. REMOTENAME が指定されていない場合、またはプログラム定義がない場合、EXEC CICS LINK コマンドの PROGRAM オプションから。

このフィールドを使用すると、プログラム・リンク要求で指定されたものとは異なる、代替プログラムをリンクするように指定できます。要求のルーティング先の領域に応じて、ローカル・プログラムまたはリモート・プログラムのいずれかを指定できます。

DYRLPROG の値を変更した場合、選択した代替プログラムが DYNAMIC(YES) と定義されているなら、ルート選択のために動的ルーティング・プログラムが再度呼び出されることに注意してください。

### ブリッジ要求

DYRTYPE=8 の場合、このフィールドを変更しないでください。加えられた変更はどれも CICS によって無視されます。

動的ルーティング・プログラムのすべての呼び出しで DYRLPROG を変更できますが、これが有効になるのは DYRFUNC が 0 または 1 に設定されている場合のみです。

## DYRLUOW

8 バイトのローカル作業単位 ID。このトークンは、LOCKED 親和性タイプのキーの一部を形成します。

このフィールドは、DYRTYPE=4 または 9 (DPL) あるいは DYRFUNC=7 (作業単位の終了) の場合にのみ有効です。DYRTYPE 4 は CHANNEL なしの DPL で、DYRTYPE 9 は CHANNEL ありの DPL です。

## DYRNETNM

DYRSYSID で示された CICS 領域のネット名。

DYRNETNM 値が動的ルーティング・プログラムの初回呼び出しによって変更された場合、CICS は、新規ネット名の CICS 領域への要求のルーティングを試みます。

## DYRNUOW

27 バイトのネットワーク作業単位 ID。このトークンは、LOCKED 親和性タイプのキーの一部を形成します。

このフィールドは、DYRTYPE=4 または 9 (DPL) あるいは DYRFUNC=7 (作業単位の終了) の場合にのみ有効です。

## DYROPERATION

アプリケーション・エントリー・ポイントのアプリケーション・コンテキスト・オペレーション名。アプリケーション・コンテキストを使用できない場合、ヌルに設定されます。

## DYROPTER

ルーティングされたトランザクションまたはリンク要求が終了 (成功または失敗して終了) したときに動的ルーティング・プログラムを再度呼び出すかどうかを指定します。以下の値が有効です。

**N**

動的ルーティング・プログラムを再度呼び出しません。N がデフォルトです。

**Y**

動的ルーティング・プログラムを再度呼び出します。

リモート CICS 領域にルーティングされるか、またはローカルで実行される、トランザクション、リンク要求、またはブリッジ要求の場合に、このオプションを指定することができます。

#### **DYRPLATFORM**

アプリケーションがデプロイされているプラットフォームのアプリケーション・コンテキスト・プラットフォーム名。アプリケーション・コンテキストを使用できない場合、またはアプリケーション・コンテキストを使用できても、プラットフォーム名が設定されていない場合は、ヌルに設定されます。

#### **DYRPROCCMP**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRPROCID**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRPROCN**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRPROCT**

動的ルーティング・プログラムでは使用されません。呼び出し時はヌルに設定されています。

#### **DYRPRTY**

端末専有領域とアプリケーション専有領域の間の接続が MRO または IPIC である場合、あるいはブリッジ要求を処理しているときに、アプリケーション専有領域でタスクのディスパッチ優先順位を設定するために使用できます。

##### **トランザクション・ルーティング**

動的ルーティング・プログラムを呼び出す前に、CICS はこの値を中継トランザクション・タスクの優先順位に設定します。

##### **プログラム・リンク要求**

動的ルーティング・プログラムを呼び出す前に、CICS はこの値を、プログラム・リンク要求を発行したタスクの優先順位に設定します。

##### **ブリッジ要求**

動的ルーティング・プログラムを呼び出す前に、CICS はこの値を、ルーター領域におけるユーザー・トランザクションの TRANSACTION リソース定義に定義された値に設定します。

動的ルーティング・プログラムの初回呼び出しからの戻り時に、DYRRTPRI 値が Y であり、端末専有領域とアプリケーション専有領域の間に MRO または IPIC 接続が存在する場合、CICS は DYPPRTY 値をアプリケーション専有領域に渡します。

#### **DYRQUEUE**

DYRSYSID で指定されたリモート・システムですぐに使用できるセッションがない場合に、要求をキューに入れるかどうかを指定します。以下の値が有効です。

**Y**

必要な場合に要求をキューに入れます。Y がデフォルトです。

**N**

要求をキューに入れません。

ブリッジ要求の場合、DYRQUEUE が Y に設定されてから、動的ルーティング・プログラムが呼び出されます。ユーザー置換可能プログラムによってこの値に加えられた変更は、CICS で無視されます。

#### **DYRRET**

進め方を CICS に通知する戻りコードが入ります。

##### **トランザクション・ルーティング**

以下の値が有効です。

**0**

トランザクションの処理を続行します。

**4**

メッセージまたはアベンドなしでトランザクションを停止します。

**8**

メッセージまたはアベンドを出してトランザクションを終了します。

ルーティング・プログラムが呼び出されるときはいつでも、DYRRETC が 0 に設定されます。ルート選択のため、またはルート選択でエラーが発生したために呼び出されるときに、CICS にトランザクションの処理を続行させる場合は、0 に設定したままにする必要があります。

CICS にトランザクションを停止させ、メッセージまたはアベンドを発行させるには、値 8 を戻します。

メッセージまたはアベンドを発行せずに CICS にトランザクションを停止させる (DFHDYP が必要とするすべての処理を完了したことを示す) には、値 4 を戻します。

APPC トランザクションのルーティングで戻りコード 4 を設定することは、予測不能な結果をもたらすため、避ける必要があります。

4 を除くゼロ以外の戻りコードを設定することは、8 を設定することと等価です。

### プログラム・リンク要求

以下の値が有効です。

**0**

リンク要求の処理を続行します。

#### ゼロ以外の値

エラー状態をプログラムに戻します。

ルーティング・プログラムが呼び出されるときはいつでも、DYRRETC が 0 に設定されます。ルート選択のため、またはルート選択でエラーが発生したために呼び出されるときに、CICS にリンク要求の処理を続行させる場合は、0 に設定したままにする必要があります。

CICS にリンク要求を拒否させるには、ゼロ以外の値を戻します。EXEC CICS LINK コマンドを発行したプログラムは、RESP2 値が 27 で PGMIDERR 状態を受け取ります。

### ブリッジ要求

以下の値が有効です。

**0**

要求の処理を続行します。

**4**

エラー・メッセージを発行せずに要求の処理を停止します。

**8**

エラー・メッセージを出して要求の処理を停止します。

ルーティング・プログラムが呼び出されるときはいつでも、DYRRETC が 0 に設定されます。ルート選択のため、またはルート選択でエラーが発生したために呼び出されるときに、CICS にリンク要求の処理を続行させる場合は、0 に設定したままにする必要があります。

メッセージを発行せずに CICS に要求を停止させるには、値 4 を戻します。クライアントに戻された BRIH メッセージ・ヘッダーには、動的ルーティング・プログラムが要求を拒否したことをクライアントに通知する戻りコードと、要求をルーティングする最後の試みが失敗した理由の詳細を示す完了コードが含まれます。

CICS に要求を停止させ、メッセージを発行させるには、値 8 を戻します。クライアントに戻された BRIH には、動的ルーティング・プログラムが要求を拒否したことをクライアントに通知する戻りコードと、要求をルーティングする最後の試みが失敗した理由の詳細を示す完了コードが含まれます。

ルーティング・プログラムが通知のために呼び出される場合、またはトランザクションの終了時に呼び出される場合は、戻りコードを設定しません。どのようなコードが設定されても CICS は無視します。

## DYRRTPRI

端末専有領域とアプリケーション専有領域の間の接続が MRO または IPIC である場合に、トランザクション、リンク要求、または要求のディスパッチ優先順位をアプリケーション専有領域に渡すかどうかを示します。以下の値が有効です。

### N

ディスパッチ優先順位は渡されません。N がデフォルトです。

### Y

ディスパッチ優先順位は渡されます。

## DYRSRCTK

ルーティングされたトランザクションに使用されている MVS ワークロード管理サービスおよびレポートのクラス・トークンです。ルーティング・プログラムはこの値を変更してはなりません。値は CICS によって設定され、CICSplex SM によって使用されます。

## DYRSYSID

CICS 領域のシステム ID (SYSID)。このパラメーターの厳密な意味は、以下に示すように DYRFUNC および DYRTYPE の値に応じて異なります。

- DYRFUNC が 0 (ルート選択) に設定されている場合

- DYRTYPE が 0、2、3、または 8 (いずれかのタイプのトランザクション・ルーティング) に設定されている場合、DYRSYSID には以下のいずれかの名前が含まれます。
    - インストールされているトランザクション定義の REMOTESYSTEM オプションに指定されている CICS 領域名
    - REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。
  - DYRTYPE が 4 または 9 (DPL ルーティング) に設定されている場合、DYRSYSID には以下のいずれかの名前が含まれます。
    - インストールされているプログラム定義の REMOTESYSTEM オプションに指定されている CICS 領域名。
- 注: REMOTESYSTEM オプションでリモート・リージョンが指名されると、ルーティング・プログラムは要求をローカルでルーティングすることができません。
- REMOTESYSTEM が指定されていない場合、またはプログラム定義がない場合、ローカル CICS 領域のシステム名。

動的ルーティング・プログラムは、DYRSYSID の値をそのまま受け取るか、またはこの値を変更してから CICS に戻すことができます。

CICS に戻す SYSID がローカル SYSID と同じである場合、CICS はローカル領域でトランザクションまたはプログラムを実行します。

- DYRFUNC が 1 (ルート選択エラー) に設定されている場合、DYRSYSID には、動的ルーティング・プログラムによって前回呼び出し時に CICS に戻された CICS 領域名が含まれます。

DYRFUNC=1 の場合に動的ルーティング・プログラムが実行できる処理は、DYRERROR パラメーターの設定に応じて異なります。

- DYRERROR が 0 (不明な SYSID) または 1 (CICS 領域がサービス中でない) に設定されている場合に、CICS にルーティングを再試行させるには、DYRSYSID を変更してから CICS に戻す必要があります。
- DYRERROR が 2 (使用可能なセッションがない) に設定されている場合に、CICS にルーティングを再試行させるには、DYRSYSID を変更するか、または DYRQUEUE の値を Y (セッションが使用可能になるまで要求をキューに入れる) に変更する必要があります。
- DYRFUNC が 2 (ルーティングされた要求の終了) に設定されている場合、DYRSYSID には、完了したトランザクションまたはリンク要求が実行された CICS 領域の名前が含まれます。
- DYRFUNC が 3 (通知) に設定されている場合:
  - ATI 要求の場合、DYRSYSID には以下のいずれかの名前が含まれます。
    - **EXEC CICS START** コマンドの SYSID オプションに指定されているリモート CICS 領域名

- SYSID が指定されていない場合は、インストールされているトランザクション定義の REMOTESYSTEM オプションに指定されているリモート CICS 領域名
- REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。
- プログラム・リンク要求の場合、DYRSYSID には、EXEC CICS LINK コマンドの SYSID オプションに指定されているリモート CICS 領域名が含まれます。
- ブリッジ要求の場合、要求がルーティングされ、ユーザー・トランザクションが実行された CICS 領域の SYSID が DYRSYSID に含まれます。

DYRSYSID または DYRNETNAME の値に加えられた変更は無視されます。

- DYRFUNC が 4 (異常終了) に設定されている場合、DYRSYSID には、トランザクションが異常終了した CICS 領域の名前が含まれます。

## DYRTRAN

リモート・トランザクション ID が含まれます。

### トランザクション・ルーティング

DYRFUNC が 0 または 3 に設定されている場合、DYRTRAN には、インストールされている TRANSACTION リソースの REMOTENAME オプションに指定されたリモート・トランザクション ID が含まれます。

### ブリッジ要求

DYRTYPE=8 の場合、DYRTRAN にはターゲット・ユーザー・トランザクションのトランザクション ID が含まれます。この ID は現在の領域で既知であるためです。これは現行トランザクション ID とは異なる点に注意してください。

### プログラム・リンク要求

DYRFUNC が 0 または 3 に設定されている場合、DYRTRAN には、以下の順序で取得されたリモート・ミラー・トランザクションのトランザクション ID が含まれます。

1. LINK コマンドの TRANSID オプションから。

注: LINK コマンドの TRANSID オプションに指定された値は、ルーティング・プログラムで指定変更できません。

2. プログラム定義の TRANSID オプションから。

3. CSMI、一般ミラー・トランザクション。TRANSID オプションのいずれも指定されていない場合は、CSMI がデフォルトです。

動的ルーティング・プログラムは、このリモート・トランザクション ID をそのまま受け取るか、または別のトランザクション名を指定してリモート CICS 領域に転送することができます。指定した名前が 4 文字を超える場合、CICS によって切り捨てが実行されます。

動的ルーティング・プログラムのすべての呼び出しで DYRTRAN を変更できますが、この変更が有効になるのは以下の状況に限られます。

- DYRFUNC が 0 または 1 に設定されている場合。
- 元の値が、**EXEC CICS LINK** コマンドの TRANSID オプションから取得されたのではない場合。  
LINK コマンドの TRANSID オプションに指定された値は、ルーティング・プログラムで指定変更できません。

## DYRTYPE

呼び出されているプログラムを必要としているルーティング要求のタイプです。トランザクション・ルーティングの場合、このフィールドは、DYRFUNC が 0 (ルート選択) または 3 (通知) に設定されているときにのみ有効です。以下の値を動的ルーティング・プログラムに渡すことができます。

**0**

端末から開始されたトランザクション。

**1**

静的にルーティングされる ATI 要求。

**2**

START にデータもチャネルも関連付けられていない、端末関連の **EXEC CICS START** コマンドで開始されたトランザクション。



3

START にデータが関連付けられているものの、チャンネルが関連付けられていない、端末関連の **EXEC CICS START** コマンドで開始されたトランザクション。

4

チャンネルなしのプログラム・リンク要求。

8

ブリッジ要求。

9

チャンネルありのプログラム・リンク要求。

A

START にチャンネルが関連付けられている、端末関連の **EXEC CICS START** コマンドで開始されたトランザクション。

#### **DYRUAPTR**

DYRVER が 7 以上の場合、このフィールドには新しいユーザー域 DYRUSERN のアドレスが含まれます。新しいユーザー域のメカニズムにより、ルーティング・プログラムのソースが、通信域を作成した CICS リリースから独立します。これまでのユーザー域フィールド DYRUSER は、互換性の目的でのみ保持されています。

ユーザー域は、DYRUAREA DSECT によってマップできます。

DYRUAPTR が 7 より小さいシステムの場合、DYRUAPTR の内容は予測不能です。

#### **DYRUOWAF**

このフィールドは、現行のネットワーク作業単位が完了するときに DYRTPE=4 または 9 (DPL) の要求に対して FUNC=7 コールバックが必要であることを CICS に通知するために、呼び出し先ユーザー出口アプリケーションによって使用されます。

CICS からの呼び出しを受け取った後は、このフィールドに該当のデータは含まれません。DYRUOWAF は、DPL 要求において CICS への応答を提供するためにのみ使用されます。以下の値が有効です。

N

コールバックは必要ありません。

Y

コールバックが必要です。

ある UOW に対して複数の DPL 呼び出しがある場合、それらの呼び出しのいずれかで Y が戻されたら、この UOW の最後にコールバックが行われます。

#### **DYRUSER**

1024 バイトのユーザー域。

このフィールドは互換性の目的でのみ保持されています。DYRUAPTR および DYRUSERN フィールドの説明を参照してください。

#### **DYRUSERID**

要求に関連付けられた CICS ユーザー ID。

トランザクション・ルーティング、プログラム・リンク要求、およびブリッジ要求の場合、DYRUSERID には現在のトランザクションの実行に使用されているユーザー ID が含まれます。

ルーティングまたはルート選択エラー (それぞれ DYRFUNC=0 または 1) のために呼び出された場合、このフィールドを検査することで、ルーティング・プログラムは、要求に関連付けられたユーザー ID に基づいて要求をルーティングできます。

#### **DYRUSERN**

1024 バイトのユーザー域。

CICS は、任意のタスクに応じて動的ルーティング・プログラムを呼び出す前に、このユーザー域をゼロに初期化します。このユーザー域は、動的ルーティング・プログラムで変更できます。変更されたユーザー域は、同じ要求におけるその後の動的ルーティング・プログラムの呼び出しに渡されます。

## DYRVER

動的ルーティング・プログラム・インターフェースのバージョン 番号。 CICS Transaction Server for z/OS, バージョン 5 リリース 6 の場合、この番号は 11 です。

## 動的ルーティング・プログラムの命名

用意されている動的ルーティング・プログラムの名前は DFHDYP です。これはユーザーが置換可能です。独自のバージョンを作成する場合は、別の名前にすることができます。

### 手順

1. 動的ルーティング・プログラムの名前を調べるには、EXEC CICS INQUIRE SYSTEM コマンドを使用します。DTRPROGRAM フィールドに、現行プログラムの名前が含まれています。
2. 以下のいずれかの方式を使用して、動的ルーティング・プログラムの名前を変更します。
  - DTRPGM システム 初期設定パラメーターの値を変更します。
  - SET SYSTEM DTRPROGRAM コマンドを使用します。
3. カスタマイズした動的ルーティング・プログラム用の新しい PROGRAM リソースを作成します。

### タスクの結果

用意されているプログラム DFHDYP の代わりに、カスタマイズした動的ルーティング・プログラムが CICS で使用されます。

## 動的ルーティング・プログラムのテスト

CICS の実行診断機能 (EDF) を使用して、動的ルーティング・プログラムを テストできます。そのためには、DFHDYP 以外の名前をプログラムに付ける必要があります。「DFH」で始まる名前のプログラムには EDF を使用できないからです。EDF の使用法の詳細については、IBM Knowledge Center の[実行診断機能 \(EDF\)](#)を参照してください。

EDF は単一または二重端末モードで使用できます。単一端末モードを選択すると、EDF は、動的ルーティング・プログラムの画面と、ルーティングされたトランザクションから呼び出されたアプリケーション・プログラムの画面を両方とも表示します。これらの画面は、次のものに関係しています。

- ルート選択または通知のための動的ルーティング・プログラムの初期呼び出し (DYRFUNC=0 または DYRFUNC=3)
- ルート選択でエラーが発生した場合の動的ルーティング・プログラムの呼び出し (DYRFUNC=1)
- アプリケーション・プログラムの呼び出し
- タスクの終了
- DYROPTER=Y を指定している場合、ルーティングされたトランザクションまたはリンク 要求の終了時の動的ルーティング・プログラムの呼び出し (DYRFUNC=2)
- DYROPTER=Y を指定している場合、ルーティングされたトランザクションが異常終了した場合の動的ルーティング・プログラムの呼び出し (DYRFUNC=4)

EDF で動的ルーティング・プログラムの実行のみを表示するには、二重端末モードを選択するか、[実行診断機能 \(EDF\)](#)で説明しているその他の方式のいずれかを使用してください。

## 動的トランザクション・ルーティングのサンプル・プログラム

CICS 提供の動的ルーティングのサンプル・プログラム DFHDYP を使用するか、独自のプログラムを COBOL、PL/I、C、またはアセンブラー言語で作成できます。プログラムの名前を変更することもできます。

CICS 提供の動的ルーティングのサンプル・プログラムは DFHDYP という名前です。通信域を定義する、対応するコピーブック は DFHDYPDS です。アセンブラー言語、COBOL、PL/I、C のソース・レベルのサンプルとコピーブックがあります。提供されているプログラムとコピーブック、およびそれらが入っているライブラリーについては、[227 ページの表 18](#) にまとめられています。

表 18. 動的トランザクション・ルーティング・プログラムとコピーブック		
言語	メンバー名	ライブラリー
プログラム: アセンブラー COBOL PL/I C	DFHDYP DFHDYP DFHDYP DFHDYP	SDFHSAMP SDFHCOB SDFHPL1 SDFHC370
コピーブック: アセンブラー COBOL PL/I C	DFHDYPDS DFHDYPDS DFHDYPDS DFHDYPDS	SDFHMAC SDFHCOB SDFHPL1 SDFHC370

DYRFUNC を「0」に設定して起動した場合、サンプル・プログラムは、通信域の DYRSYSID と DYRTRAN フィールドで渡された sysid とリモート・トランザクション名を受け入れ、DYRRETC を「0」に設定してから、CICS に戻ります。DYRFUNC を「2」または「3」に設定して起動した場合、戻りコードは「0」に設定されます。DYRFUNC を「1」または「4」に設定して起動した場合、戻りコードは「8」に設定されます。

トランザクションまたは DPL 要求を動的にルーティングするには、DFHDYP をカスタマイズするか、独自のルーティング・プログラムに完全に置き換えるか、または CICSplex System Manager を使用する必要があります。

## 分散ルーティング・プログラムの作成

分散ルーティング・プログラムを使用して、CICS 内のさまざまなタイプの要求 (インバウンド Web サービスや非端末の START 要求など) をルーティングできます。

分散ルーティング・プログラムは、ルーティング CICS 領域とターゲット CICS 領域の **DSRTPGM** システム初期設定パラメーターで指定されます。分散ルーティング・プログラムを使用して、以下の要求をルーティングできます。

- CICS ビジネス・トランザクション・サービス (BTS) プロセスおよびアクティビティー
- 非端末関連の **EXEC CICS START** 要求

分散ルーティングの対象となる非端末関連 START 要求については、[START コマンドで呼び出されたトランザクションのルーティング](#)を参照してください。

- インバウンド Web サービス要求

以下の要求は、分散ルーティング・プログラムを使用してルーティングすることはできません。

- ユーザー端末から開始されたトランザクション
- 端末関連 **EXEC CICS START** コマンドによって開始されるトランザクション
- プログラム・リンク要求

これらの要求をルーティングするには、**DTRPGM** システム初期設定パラメーターに指定した動的ルーティング・プログラムを使用する必要があります。動的ルーティング・プログラムの作成方法は、[197 ページの『動的ルーティング・プログラムの作成』](#)に説明されています。動的ルーティング・プログラムと分散ルーティング・プログラムが同じプログラムであってもかまいません。

CICSplex System Manager (CICSplex SM) を使用して CICSplex を管理する場合は、ルーティング・プログラム EYU9XLOP を使用できます。このプログラムは、ワークロード・バランシングとワークロード分離をサポートしています。CICSplex 内のどの領域がワークロードに参加できるかを定義でき、特定の要求のルーティング先となる領域を決定するためのトランザクション・アフィニティーを定義できます。CICSplex SM のワークロード管理について詳しくは、[ワークロード管理の構成](#)を参照してください。

## 分散ルーティング・インターフェースと動的ルーティング・インターフェースの相違点

分散ルーティング・インターフェースは、動的ルーティング・インターフェースとはいくつかの重要な点で異なります。

動的ルーティング・プログラムを以前作成したことがあり、分散ルーティング・プログラムをこれから作成しようとしている場合は、以下の点に留意してください。

1. 動的ルーティング・プログラムと分散ルーティング・プログラムは、リソース (トランザクションまたはプログラム) が DYNAMIC(YES) と定義されている場合に呼び出されます。非同期で実行される BTS アクティビティーの場合は例外で、関連トランザクションが DYNAMIC(NO) と定義されていても、分散ルーティング・プログラムが呼び出されます。この場合、分散ルーティング・プログラムは要求を経路指定できませんが、ワークロードに対する要求の影響をモニターするか、他のアクティビティーを実行することができます。つまり、分散ルーティング・プログラムは、静的ルーティング要求の影響をターゲット領域の相対ワークロードでモニターすることができます。
2. 動的ルーティング・プログラムは階層型の「ハブ」ルーティング・モデルを使用する (1 つのルーティング・プログラムで、複数のターゲット領域上のリソースへのアクセスを制御する) ので、ルーティング要求の終了時に呼び出されるルーティング・プログラムは、経路選択の際に呼び出したものと同じプログラムになります。

一方、分散ルーティング・プログラムは、ピアツーピア・システムである、分散モデルを使用し、ルーティング・プログラムそのものが分散されます。ルーティングされたトランザクションの開始時、終了時、または異常終了時に呼び出されるルーティング・プログラムは、経路選択のために呼び出されたプログラムと同じではありません。呼び出されるのは、ターゲット領域のルーティング・プログラムです。

動的ルーティング・プログラムはルーティング領域でのみ呼び出されるため、動的ルーティング・プログラムの呼び出しの順序は、以下のように厳密に定義されています。

- a. ルート選択または通知
  - b. ルート選択エラー (該当する場合。繰り返される可能性がある。)
  - c. トランザクション終了または異常終了 (要求された場合)
3. 単一要求の場合、動的ルーティング・プログラムの各呼び出しに渡されるユーザー域は、ストレージの同じ部分です。ある呼び出しでユーザー域に加えられた変更は保持され、次の呼び出しに渡されます。  
一方、分散ルーティング・プログラムは、ルーティング領域だけでなく、ターゲット領域でも呼び出される場合があります。そのため、分散ルーティング・プログラムの呼び出しの順序は、それほど厳密に定義されていません。例えば、ルーティング領域での最終呼び出し (「ルーティング試行完了」の際) が、ターゲット領域での最初の呼び出し (「トランザクション開始」の際) の前に行われる場合もあれば、後で行われる場合もあります。この不確実性に対処するために、ターゲット領域での最初の呼び出しで分散ルーティング・プログラムに渡されるユーザー域は、ルーティング領域でのユーザー域のコピーになっています。これは、ターゲット領域でユーザー域に加えられた変更は、ルーティング領域内のユーザー域に影響を及ぼさないことを意味します。詳しくは、[239 ページの『分散ルーティング・プログラムに渡されるパラメーター』](#)の DYRUSER フィールドの説明を参照してください。
  4. 分散ルーティング・プログラムが呼び出されるポイントは、動的ルーティング・プログラムよりも多くなります。分散ルーティング・プログラムが呼び出されるポイント、および各呼び出しが行われる領域について、[232 ページの『分散ルーティング・プログラムを呼び出す場合』](#)で説明しています。
  5. 動的ルーティング・プログラムとは異なり、分散ルーティング・プログラムは以下の処理を行いません。

- ネット名を指定してターゲット領域を選択する (通信域の DYRNETNM フィールドに設定されている値は無視されます)。CICS システム ID (sysid) でターゲットを指定する必要があります。
- ターゲット領域に渡されるリモート・トランザクション名を変更する。 (通信域の DYRTRAN フィールドに設定されている値は無視されます。)
- ルーティングされた要求に関連付けられた初期プログラムを変更する。 (通信域の DYRLPROG フィールドに設定されている値は無視されます。)
- ターゲット領域への MRO セッションがない場合に要求をキューに入れなかったことを選択する。 (通信域の DYRQUEUE フィールドは常に「Y」に設定されています。)
- ルーティングされたアプリケーションの通信域を変更する。 (フィールド DYRACMAA 内のルーティングされたアプリケーションの通信域のアドレスは、ルーティング・プログラムに渡されません。)
- トランザクションのディスパッチ優先順位をターゲット領域に渡す。 (通信域の DYRRTPRI フィールドは常に「N」に設定されています。)

これらの制約事項については、DFHDYPDS 通信域の関連フィールドの説明に、より詳しく記載されています。

## BTS アクティビティのルーティング

分散ルーティング・プログラムを使用して、CICS ビジネス・トランザクション・サービス (BTS) のプロセスとアクティビティを動的にルーティングできます。

### 動的にルーティングできる BTS アクティビティ

すべての BTS プロセスおよびアクティビティのアクティブ化をルーティングできるわけではありません。

RUN ASYNCHRONOUS コマンドを使用して要求側により非同期にアクティブ化されたプロセスおよびアクティビティは、動的にも静的にもルーティングできます。

要求側によって同期的にアクティブにされたプロセスやアクティビティ (RUN SYNCHRONOUS または LINK コマンドによる) は、常にローカルで実行されます。それらは、動的にも静的にもルーティングできません。関連付けられたトランザクションが DYNAMIC(YES) として定義されているか、リモート領域にあるとして定義されているアクティビティに対して RUN SYNCHRONOUS または LINK コマンドを発行すると、そのアクティビティはローカルで実行されます。

そのため、動的ルーティング可能にするには、以下のようになります。

1. BTS プロセスまたはアクティビティは、RUN ASYNCHRONOUS コマンドを使用して、要求側によって非同期的に実行する必要があります。
2. プロセスまたはアクティビティに関連付けられたトランザクションの TRANSACTION 定義には、DYNAMIC(YES) を指定する必要があります。

「デ이지ー・チェーン」はサポートされていません。つまり、BTS アクティビティがターゲット領域にルーティングされると、関連付けられたトランザクションが DYNAMIC(YES) として定義されていても、そのアクティビティをターゲット領域から第 3 の領域に再ルーティングすることはできません。

### 分散ルーティング・プログラムを呼び出す場合

RUN ASYNCHRONOUS コマンドによって開始された BTS プロセスおよびアクティビティでは、CICS は以下のポイントで分散ルーティング・プログラムを呼び出します。

ルーティング領域では、次のとおりです。

1. 以下のうちのどちらか。
  - アクティビティのルーティングのため。これは、アクティビティに関連付けられたトランザクションが DYNAMIC(YES) と定義されている場合に起こります。
  - 静的にルーティングされた要求の通知のため。これは、アクティビティに関連付けられたトランザクションが DYNAMIC(NO) と定義されている場合に起こります。ルーティング・プログラムはアクティビティをルーティングすることができません。ただし、その他は実行できます。
2. 経路選択でエラーが発生した場合。例えば、経路選択呼び出しでルーティング・プログラムによって戻されたターゲット領域が利用できない場合などです。これにより、ルーティング・プログラムは代替ターゲットを指定することができます。このプロセスは、ルーティング・プログラムが使用可能なターゲットを選択するか、ゼロ以外の戻りコードを設定するまで繰り返されます。
3. CICS がアクティビティをターゲット領域にルーティングしようとした後 (結果が成功か失敗かに関係なく)。

この呼び出しは (ルーティング領域とターゲット領域が同一である場合を除き)、このトランザクションに対するルーティング領域の責任を破棄するように指示します。例えば、ルーティング・プログラムはこの呼び出しを使用して、トランザクションのために獲得したリソースを解放することができます。

ターゲット領域では、次のとおりです。

これらの呼び出しは、ルーティング領域内のルーティング・プログラムで、そのプログラムがターゲット領域で再呼び出しされる必要があると指定されている場合にのみ起こります。

1. ターゲット領域でアクティブ化が開始された場合 (つまり、アクティビティを実装するトランザクションが開始された場合)。
2. ルーティングされた活動化 (トランザクション) が正常に終了した場合。



3. ルーティングされた活動化(トランザクション)が異常終了した場合。

230 ページの図 57 は、分散ルーティング・プログラムが呼び出されるポイントと、各呼び出しが発生する領域を示しています。「ターゲット領域」は、必ずしもリモートではないということにご注意ください。ルーティング・プログラムがアクティビティをローカルで実行することを選択する場合は、ローカル(ルーティング)領域の場合があります。

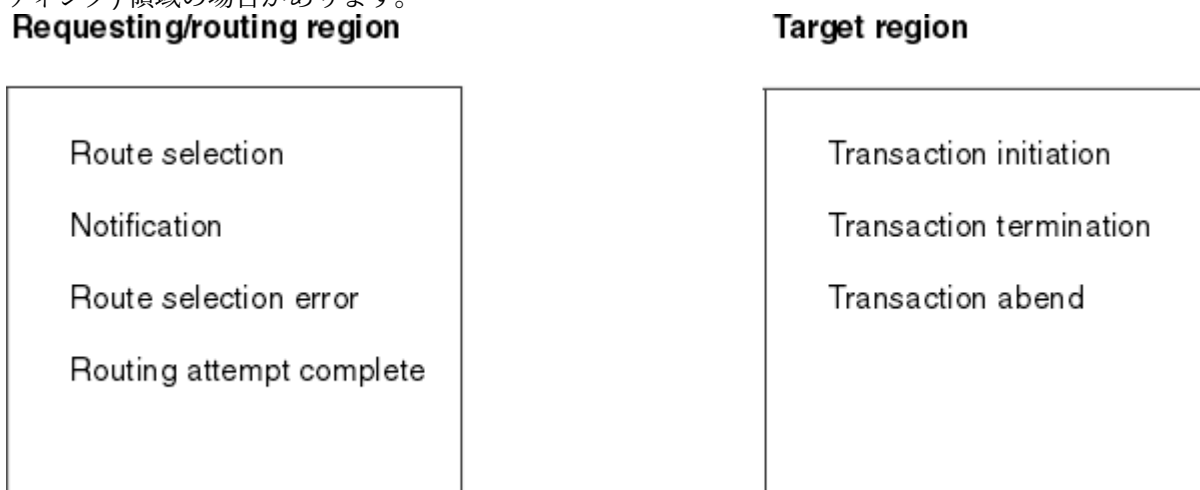


図 57. 分散ルーティング・プログラムが呼び出される場合と場所

### ターゲット CICS 領域の変更

初期設定では、分散ルーティング・プログラムに渡される通信域の DYRSYSID フィールドには、プロセスまたはアクティビティのルーティング先となるデフォルトのターゲット領域のシステム ID (sysid) が含まれています。これはルーティング領域にインストールされたトランザクション定義の REMOTESYSTEM オプションの値から派生します。REMOTESYSTEM が指定されていない場合、ローカル CICS 領域の sysid が渡されます。

経路選択のために呼び出された場合、分散ルーティング・プログラムは DYRSYSID の値を変更することでターゲット領域を変更できます。

指定された sysid が無効であるか、見つからない場合、分散ルーティング・プログラムに SYSIDERR が返されます。別の sysid を戻すことによって、このエラーに対応することができます。231 ページの『ルート選択でエラーが発生した場合』を参照してください。

ルーティング・プログラムが、通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了のために呼び出された場合に sysid を変更しても、その変更は無効です。

### アクティビティをルーティングするかどうかについての CICS への指示

ルーティング・プログラムがルーティングのために呼び出される際に、(値を変更したかどうかにかかわらず) プロセスまたはアクティビティをルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。

戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された sysid と自身のローカル sysid を次のように比較します。

- sysid が同じ(または、戻された sysid がブランク)である場合、CICS は RUN 要求をローカルで実行します。要求がローカルで実行される場合、CICS はメッセージ DFHSH0102 を CSSH 一時データ・キューに書き込みます。
- 2 つの sysid が異なる場合、CICS は要求をリモートの CICS 領域にルーティングします。

CICS が要求をサービス不能として扱うようにするには、ゼロ以外の値を返します。サービス不能の要求について詳しくは、[サービスできない要求の処理](#)を参照してください。

ルーティング・プログラムが通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了時に呼び出された場合は、DYRRETC に値を戻しても無効です。



## ルート選択でエラーが発生した場合

ルート選択でエラーが発生した場合、例えば分散ルーティング・プログラムから戻されたシステム ID が使用不可または不明の場合、分散ルーティング・プログラムが再度呼び出されます。

プログラムが再呼び出しされたら、以下のアクションのいずれかを選択できます。

1. SYSID を変更し、DYRRETC で戻りコード「0」を発行することで、異なるターゲット領域への要求のルーティングを試行できます。

その領域も使用できない場合は、ルート選択エラーのためにルーティング・プログラムがもう一度起動されます。この要求のルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド DYRCOUNT で渡されます。この回数を使用して、要求のルーティングの試行をいつ中止するかを決定できます。

2. DYRRETC でゼロ以外の戻りコードを発行することで、要求を「サービス不能」として扱うよう CICS に通知できます。

トランザクション・アフィニティーなどの影響で、特定のターゲット領域でアクティビティーを実行し、他の領域では実行しないことが必要な場合があります。これに該当する場合、ターゲット領域が使用不可であれば、要求をサービス不能に分類してください。ルート選択エラーのためにルーティング・プログラムを再呼び出しする代わりに、CICS は以下の処理を行います。

- a. 指定されたターゲット領域への要求のルーティングを 1 分間隔で繰り返し試行します。

これらの試みが 1 回成功すると、CICS はメッセージ DFHSH0108 を発行します。ルーティング・プログラムは、「ルーティング試行完了」のためにルーティング領域で呼び出され、指定された場合は、「トランザクション開始」のためにターゲット領域で呼び出されます。

- b. ターゲット領域が依然として使用不可の場合は、1 時間ごとにメッセージ DFHSH0106 を発行します。

- c. 要求が発行されてから 24 時間経過してもターゲット領域が使用不可の場合は、メッセージ DFHSH0107 を発行し、要求のルーティングの試行を中止します。要求は破棄されます。「ルーティング試行完了」のためにルーティング領域でルーティング・プログラムが呼び出されます。

## ターゲット領域での分散ルーティング・プログラムの呼び出し

経路選択、通知、経路選択エラー、およびルーティング完了による分散ルーティング・プログラムの呼び出しは、すべてルーティング領域で起こります。ルーティング・プログラムがターゲット領域で再度呼び出されるようにするには、通信域の DYROPTER フィールドを 'Y' に設定します。これは、プログラムの初期 (ルート選択または通知) 呼び出しで行う必要があります。さらに、ルート選択エラーで再呼び出しされる場合は、再び行う必要があります。

ルーティング・プログラムが DYROPTER を 'Y' に設定する場合、次の状況でターゲット領域で再呼び出されます。

- 活動化がターゲット領域で開始されようとしている場合
- ルーティングされた活動化 (トランザクション) が正常に終了した場合
- ルーティングされた活動化 (トランザクション) が異常終了した場合

ターゲット領域で呼び出されるたびに、ルーティング・プログラムはその領域で現在実行中の BTS アクティビティーの数を更新できます。ルーティングのために呼び出される場合、ルーティング・プログラムはルーティング・セット内のすべての領域に保持されている数 (それ自体を含む) をそのルーティングの判断への入力として使用できます。これには、ルーティング・セット内の各領域に、数が記録されている共通データ・セットへのアクセス権限がある必要があります。

## 非端末関連 START 要求のルーティング

分散ルーティング・プログラムを使用して、非端末関連 EXEC CICS START 要求を動的にルーティングできます。

### 動的にルーティングできる要求

非端末関連 START 要求が動的ルーティング可能であるためには、次の条件をすべて満たしていなければなりません。

- 要求が拡張型 ルーティング可能である。EXEC CICS START コマンドによって呼び出される「拡張型」ルーティング・トランザクション方式の概要、および拡張型ルーティング可能な非端末関連の START 要求固有の情報については、START コマンドで呼び出されたトランザクションのルーティングを参照してください。
- ルーティング領域のトランザクション定義では、ROUTABLE(YES) と DYNAMIC(YES) の両方が指定されています。
- START コマンドの SYSID オプションで、リモート領域の名前が指定されていない (つまり、トランザクションを開始するリモート領域が明示的に指定されていない)。

要求が完全に動的ルーティング可能な場合には、分散ルーティング・プログラムがルーティングのために呼び出されます。START 要求は、ルーティング・プログラムから戻されたターゲット領域に機能シップされます。

注：

1. 要求が拡張ルーティング可能でない場合には、分散ルーティング・プログラムは呼び出されません。START コマンドの SYSID オプションでリモート領域を明示的に指定していない限り、START 要求は REMOTESYSTEM オプションで指定されたターゲット領域に機能シップされます。REMOTESYSTEM が指定されていない場合は、START はローカルで実行されます。
2. 要求が拡張ルーティング可能であるが、動的ルーティング可能ではない場合 (例えば、トランザクションが DYNAMIC(NO) と定義されている)、分散ルーティング・プログラムは 通知の場合にのみ呼び出されます。要求をルーティングすることはできません。START コマンドの SYSID オプションでリモート領域を明示的に指定していない限り、START 要求は REMOTESYSTEM オプションで指定されたターゲット領域に機能シップされます。REMOTESYSTEM が指定されていない場合は、START はローカルで実行されます。

「デ이지ー・チェーン」はサポートされていません。つまり、非端末関連の START 要求がターゲット領域に動的にルーティングされると、トランザクションが ROUTABLE(YES) および DYNAMIC(YES) として定義されていても、その要求をターゲット領域から第 3 の領域へ動的にルーティングすることはできません。ただし、トランザクションをターゲット領域から第 3 の領域へ静的にルーティングすることは可能です。

動的ルーティング可能な非端末関連 START 要求に関する具体的な情報については、非端末関連の START コマンドを参照してください。

### 分散ルーティング・プログラムを呼び出す場合

拡張型ルーティング可能な非端末関連 START 要求では、CICS は以下のポイントで分散ルーティング・プログラムを呼び出します。

ルーティング領域では、次のとおりです。

1. 以下のうちのどちらか。
  - 要求のルーティングのため。
  - 静的にルーティングされた要求の通知のため。これは、以下のいずれか、または両方が当てはまることが理由で、ROUTABLE(YES) として定義されたトランザクションが拡張型 ルーティング可能であるものの動的ルーティング不可である場合に起こります。
    - トランザクション定義で DYNAMIC(NO) が指定されている。
    - START コマンドの SYSID オプションで明示的にリモート領域が指定されている。
2. 経路選択でエラーが発生した場合。例えば、経路選択呼び出しでルーティング・プログラムによって戻されたターゲット領域が利用できない場合などです。これにより、ルーティング・プログラムは代替ターゲットを指定することができます。このプロセスは、ルーティング・プログラムが使用可能なターゲットを選択するか、ゼロ以外の戻りコードを設定するまで繰り返されます。
3. CICS が要求をターゲット領域にルーティングしようとした後 (結果が成功か失敗かに関係なく)。

この呼び出しは (ルーティング領域とターゲット領域が同一である場合を除き)、このトランザクションに対するルーティング領域の責任を破棄するように指示します。例えば、ルーティング・プロ

グラムはこの呼び出しを使用して、トランザクションのために獲得したリソースを解放することができます。

**ターゲット領域では、次のとおりです。**

これらの呼び出しは、ルーティング領域内のルーティング・プログラムで、そのプログラムがターゲット領域で再呼び出しされる必要があると指定されている場合にのみ起こります。

1. 要求に関連付けられたトランザクションがターゲット領域で開始された場合。
2. トランザクションが正常に終了した場合。
3. トランザクションが異常終了した場合。

233 ページの図 58 は、分散ルーティング・プログラムが呼び出されるポイントと、各呼び出しが発生する領域を示しています。「ターゲット領域」は、必ずしもリモートではないということにご注意ください。ルーティング・プログラムが START 要求をローカルで実行することを選択する場合は、ローカル (ルーティング) 領域の場合があります。

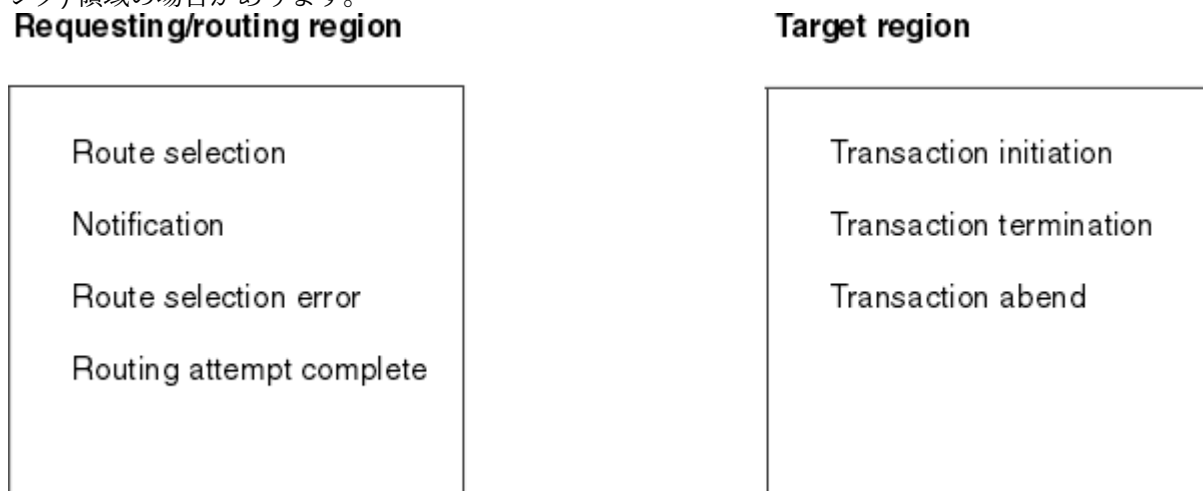


図 58. 分散ルーティング・プログラムが呼び出される場合と場所

### ターゲット CICS 領域の変更

初期設定では、分散ルーティング・プログラムに渡される通信域の DYRSYSID フィールドには、要求のルーティング先となるデフォルトのターゲット領域のシステム ID (sysid) が含まれています。これはルーティング領域にインストールされたトランザクション定義の REMOTESYSTEM オプションの値から派生します。REMOTESYSTEM が指定されていない場合、ローカル CICS 領域の sysid が渡されます。

経路選択のために呼び出された場合、分散ルーティング・プログラムは DYRSYSID の値を変更することでターゲット領域を変更できます。

指定された sysid が無効であるか、見つからない場合、分散ルーティング・プログラムに SYSIDERR が返されます。別の sysid を戻すことによって、このエラーに対応することができます。[234 ページの『ルート選択でエラーが発生した場合』](#)を参照してください。

ルーティング・プログラムが、通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了のために呼び出された場合に sysid を変更しても、その変更は無効です。

### 要求をルーティングするかどうかについての CICS への指示

ルーティング・プログラムがルーティングのために呼び出される際に、(値を変更したかどうかにかかわらず) 要求をルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。

戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された sysid と自身のローカル sysid を次のように比較します。

- sysid が同じである場合、CICS は要求をローカルで実行します。
- 2 つの sysid が異なる場合、CICS は要求をリモートの CICS 領域にルーティングします。

CICS が START 要求を拒否するようするには、ゼロ以外の値を返します。EXEC CICS START コマンドは、START 要求がルーティング・プログラムによって拒否されたことを示す RESP2 値の SYSIDERR 条件を受け取ります。

ルーティング・プログラムが通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了時に呼び出された場合は、DYRRETC に値を戻しても無効です。

### ルート選択でエラーが発生した場合

ルート選択でエラーが発生した場合 (例えば分散ルーティング・プログラムから返された SYSID が利用できない場合や認識されない場合など) は、ルーティング・プログラムがもう一度起動されます。

ルーティング・プログラムが再呼び出しされたら、以下のアクションのいずれかを選択できます。

1. SYSID を変更し、DYRRETC で戻りコード「0」を発行することで、異なるターゲット領域への要求のルーティングを試行できます。

その領域も使用できない場合は、ルート選択エラーのためにルーティング・プログラムがもう一度起動されます。この要求のルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド DYRCOUNT で渡されます。この回数を使用して、要求のルーティングの試行をいつ中止するかを決定できます。

2. DYRRETC でゼロ以外の戻りコードを発行することで、要求のルーティングの試行を続行しないよう CICS に指示できます。

### XICERES 出口を使用したターゲット領域でのリソースの使用可否の検査

XICERES グローバル・ユーザー出口プログラムを使用して、開始されたトランザクションに必要なすべてのリソースが、ターゲット領域で使用可能であることを確認できます。

XICERES 出口は (有効な場合)、動的にルーティングされる START 要求を CICS が処理する前に、ターゲット領域で起動されます。

例えば、ターゲット領域で開始するトランザクションが無効な場合や、必要なファイルがない場合などに、出口プログラムにより、分散ルーティング・プログラムが要求を別の領域にルーティングする機会を与えることができます。このためには、出口プログラムで戻りコード UERCRESU を設定する必要があります。こうすることにより、CICS は次のことを実行します。

1. ミラーによりターゲット領域に対して実行された EXEC CICS START コマンドに RESUNAVAIL 状態を戻します。(この状態は、アプリケーション・プログラムには戻されません。)
2. ルーティング・プログラムの通信域の DYRERROR フィールドを「F」(リソース使用不可) に設定します。
3. ルート選択エラーのために、ルーティング領域でルーティング・プログラムを再起動します ([234 ページの『ルート選択でエラーが発生した場合』](#)を参照)。

XICERES グローバル・ユーザー出口プログラムの作成方法については、[インターバル制御機能 EXEC インターフェース・プログラム出口 \(XICEREQ、XICERES、および XICEREQC\)](#)を参照してください。

必要なリソースがターゲット領域になく、XICERES 出口が使用不可または無効になっている場合 (または有効であるが UERCRESU 戻りコードが設定されていない場合) には、クライアント・プログラムはエラー応答を受け取ります。

### ターゲット領域での分散ルーティング・プログラムの呼び出し

経路選択、通知、経路選択エラー、およびルーティング完了による分散ルーティング・プログラムの呼び出しは、すべてルーティング領域で起こります。ルーティング・プログラムがターゲット領域で再度呼び出されるようするには、通信域の DYROPTER フィールドを 'Y' に設定します。これは、プログラムの初期 (ルート選択または通知) 呼び出しで行う必要があります。さらに、ルート選択エラーで再呼び出しされる場合は、再び行う必要があります。

ルーティング・プログラムが DYROPTER を 'Y' に設定する場合、次の状況でターゲット領域で再呼び出されます。

- ルーティングされた要求に関連付けられたトランザクションがターゲット領域で開始されようとしている場合
- トランザクションが正常に終了した場合



- トランザクションが異常終了した場合

ターゲット領域で呼び出されるたびに、ルーティング・プログラムは その領域で現在実行中のトランザクションの数を更新できます。ルーティングのために呼び出される場合、ルーティング・プログラムはルーティング・セット内のすべての領域に保持されている数 (それ自体を含む) を そのルーティングの判断への入力として使用できます。これには、ルーティング・セット内の各領域に、数が記録されている共通データ・セットへのアクセス権限がある必要があります。

## インバウンド Web サービス要求のルーティング

分散ルーティング・プログラムを使用して、インバウンド Web サービス要求を動的にルーティングすることができます。

### 端末ハンドラーでのインバウンド要求の動的ルーティング

サービス・プロバイダー・パイプラインの端末ハンドラーが CICS 提供の SOAP メッセージ・ハンドラーの 1 つであるとき、コンテナ **DFHWS-APPHANDLER** に指定されるターゲット・アプリケーションのハンドラー・プログラムが、動的ルーティングに適格である場合があります。アプリケーション・ハンドラー・プログラムより前のパイプライン処理はすべて、常に SOAP メッセージを受け取った CICS 領域でローカルに実行されます。

ターゲット・アプリケーションのハンドラー・プログラムを実行する トランザクションは、以下の条件のいずれかが真である場合に、ルーティングに適格です。

- パイプラインがメッセージを処理するトランザクションが、DYNAMIC または REMOTE として 定義される。このトランザクションは、インバウンド SOAP メッセージからの URI のマップに使用される URIMAP に定義されます。
- パイプラインのプログラムが、コンテナ **DFHWS-USERID** の内容を初期値から変更した。
- パイプラインのプログラムが、コンテナ **DFHWS-TRANID** の内容を初期値から変更した。
- WS-AT SOAP ヘッダーがインバウンド SOAP メッセージ内にある。

前のすべてのシナリオで、パイプラインの処理中にタスク切り替えが起こります。2 番目のタスクは、**DFHWS-TRANID** コンテナに指定されたトランザクションの下で実行します。このタスク切り替えによって動的ルーティングが起こる機会が生じますが、CICS 領域同士の接続に MRO が使用される場合に限ります。さらに、ルーティング先の CICS 領域が、チャンネルおよびコンテナをサポートする必要があります。

**DFHWS-TRANID** に指定されたトランザクションの TRANSACTION 定義が以下の一連の属性のいずれかを指定する場合にのみ、ルーティングが起こります。

#### DYNAMIC(YES)

トランザクションは、ルーティング・プログラムが **DSRTPGM** システム 初期設定パラメーターに指定された、分散ルーティング・モデルを使用してルーティングされます。

#### DYNAMIC(NO) REMOTESYSTEM(sysid)

トランザクションは、**sysid** で識別されるシステムにルーティングされます。

Web サービス要求のルーティングについて詳しくは、技術情報 [プロバイダー・モード CICS Web サービスのルーティング](#) を参照してください。

CICS Web サービス・アシスタントを使用して配置されたアプリケーションでは、CICS がユーザー・プログラムにリンクする際に、要求を動的にルーティングする 2 番目の機会があります。このとき要求は、ルーティング・プログラムが **DTRPGM** システム 初期設定パラメーターに指定された、動的ルーティング・モデルを使用してルーティングされます。このケースでは、プログラムの特性によってルーティングが適格であると判断されます。プログラムにリンクする際にチャンネルおよびコンテナを使用する場合は、V3.1 以上の CICS 領域にのみ要求を動的にルーティングできます。COMMAREA を使用する場合は、この制限は適用されません。

要求がターゲット領域に動的にルーティングされると、トランザクションが ROUTABLE(YES) および DYNAMIC(YES) として定義されていても、その要求をターゲット領域から第 3 の領域へ動的にルーティングすることはできません。ただし、トランザクションを静的にターゲット領域から第 3 の領域へルーティングすることは可能です。

## 分散ルーティング・プログラムを呼び出す場合

拡張型ルーティング可能なインバウンド Web サービス要求では、CICS は以下のポイントで分散ルーティング・プログラムを呼び出します。

ルーティング領域では、次のとおりです。

1. 要求のルーティングのため。
2. 経路選択でエラーが発生した場合。例えば、経路選択呼び出しでルーティング・プログラムによって戻されたターゲット領域が利用できない場合などです。これにより、ルーティング・プログラムは代替ターゲットを指定することができます。このプロセスは、ルーティング・プログラムが使用可能なターゲットを選択するか、ゼロ以外の戻りコードを設定するまで繰り返されます。
3. CICS が要求をターゲット領域にルーティングしようとした後 (結果が成功か失敗かに関係なく)。

この呼び出しは (ルーティング領域とターゲット領域が同一である場合を除き)、このトランザクションに対するルーティング領域の責任を破棄するように指示します。例えば、ルーティング・プログラムはこの呼び出しを使用して、トランザクションのために獲得したリソースを解放することができます。

ターゲット領域では、次のとおりです。

1. 要求に関連付けられたトランザクションがターゲット領域で開始された場合。
2. トランザクションが正常に終了した場合。
3. トランザクションが異常終了した場合

236 ページの図 59 は、分散ルーティング・プログラムが呼び出されるポイントと、各呼び出しが発生する領域を示しています。「ターゲット領域」は、必ずしもリモートではないということにご注意ください。ルーティング・プログラムが要求をローカルで実行することを選択する場合は、ローカル (ルーティング) 領域の場合があります。

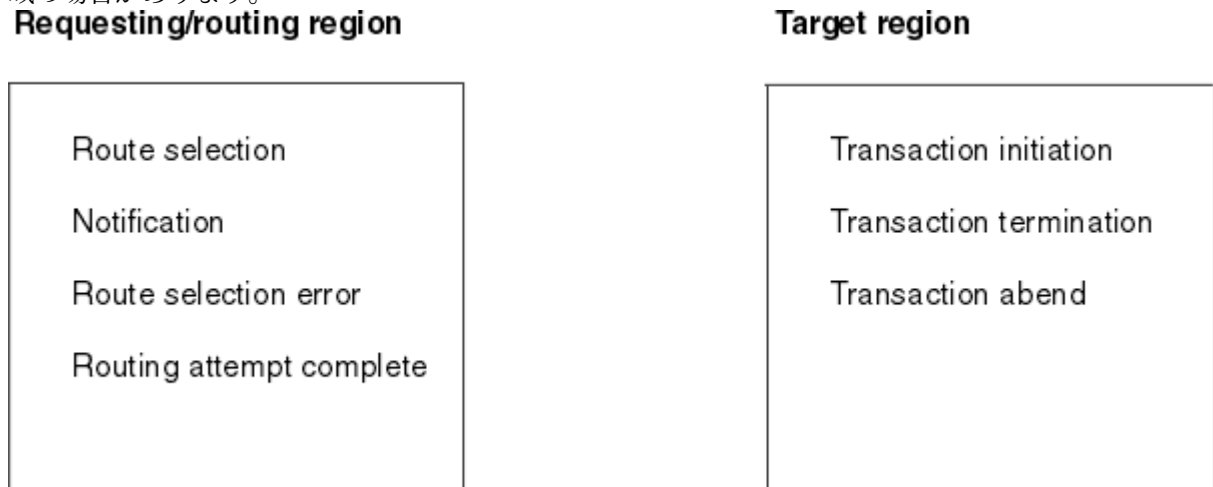


図 59. 分散ルーティング・プログラムが呼び出される場合と場所

## ターゲット CICS 領域の変更

初期設定では、分散ルーティング・プログラムに渡される通信域の DYRSYSID フィールドには、要求のルーティング先となるデフォルトのターゲット領域のシステム ID (sysid) が含まれています。これはルーティング領域にインストールされたトランザクション定義の REMOTESYSTEM オプション の値から派生します。REMOTESYSTEM が指定されていない場合、ローカル CICS 領域の sysid が渡されます。

経路選択のために呼び出された場合、分散ルーティング・プログラムは DYRSYSID の値を変更することでターゲット領域を変更できます。

指定された sysid が無効であるか、見つからない場合、分散ルーティング・プログラムに SYSIDERR が返されます。別の sysid を戻すことによって、このエラーに対応することができます。237 ページの『ルート選択でエラーが発生した場合』を参照してください。



ルーティング・プログラムが、通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了のために呼び出された場合に sysid を変更しても、その変更は無効です。

### 要求をルーティングするかどうかについての CICS への指示

ルーティング・プログラムがルーティングのために呼び出される際に、(値を変更したかどうかにかかわらず) 要求をルーティングするには、通信域のフィールド DYRRETC で、CICS にゼロの値を返します。

戻りコードをゼロにして制御を CICS に戻すと、CICS はまず、戻された sysid と自身のローカル sysid を次のように比較します。

- sysid が同じである場合、CICS は要求をローカルで実行します。
- 2 つの sysid が異なる場合、CICS は 要求をリモートの CICS 領域にルーティングします。

CICS が要求を拒否するようにするには、ゼロ以外の値を返します。

ルーティング・プログラムが通知、ルーティングの完了、トランザクションの開始、トランザクションの終了、または異常終了時に呼び出された場合は、DYRRETC に値を戻しても無効です。

### ルート選択でエラーが発生した場合

ルート選択でエラーが発生した場合 (例えば分散ルーティング・プログラムから返された SYSID が利用できない場合や認識されない場合など) は、ルーティング・プログラムがもう一度起動されます。

ルーティング・プログラムをもう一度起動する場合は、以下のアクションのいずれかを選択できます。

1. SYSID を変更し、DYRRETC で戻りコード「0」を発行することで、異なるターゲット領域への要求のルーティングを試行できます。

その領域も使用できない場合は、ルート選択エラーのためにルーティング・プログラムがもう一度起動されます。この要求のルーティングの目的でルーティング・プログラムが呼び出された回数は、フィールド DYRCOUNT で渡されます。この回数を使用して、要求のルーティングの試行をいつ中止するかを決定できます。

2. DYRRETC でゼロ以外の戻りコードを発行することで、要求のルーティングの試行を続行しないよう CICS に指示できます。

### ターゲット領域での分散ルーティング・プログラムの呼び出し

経路選択、通知、経路選択エラー、およびルーティング完了による分散ルーティング・プログラムの呼び出しは、すべてルーティング領域で起こります。ただし、ルーティング・プログラムをターゲット領域で再度呼び出すことができます。

**前提条件:** 分散ルーティング・プログラムは、可能性があるすべてのターゲット領域において **DSRTPGM** システム初期設定パラメーターで指定する必要があります。

### ターゲット領域でルーティング・プログラムを再度呼び出す方法

通信域の DYROPTER フィールドを 'Y' に設定します。これは、プログラムの初期 (ルート選択または通知) 呼び出しで行う必要があります。さらに、ルート選択エラーで再呼び出しされる場合は、再び行う必要があります。

### ターゲット領域でルーティング・プログラムが再度呼び出されるタイミング

ルーティング・プログラムが DYROPTER を 'Y' に設定する場合、次のいずれかの状況でターゲット領域で再呼び出しされます。

- ルーティングされた要求に関連付けられたトランザクションがターゲット領域で開始されようとしている場合
- トランザクションが正常に終了した場合
- トランザクションが異常終了した場合

### ターゲット領域でルーティング・プログラムが呼び出されるときに生じる事柄

ターゲット領域で呼び出されるたびに、ルーティング・プログラムはその領域で現在実行中のトランザクションの数を更新できます。ルーティングのために呼び出される場合、ルーティング・プログラムはルー

ティング・セット内のすべての領域に保持されている数(それ自体を含む)をそのルーティングの判断への入力として使用できます。これには、ルーティング・セット内の各領域に、数が記録されている共通データ・セットへのアクセス権限がある必要があります。

## ユーザー ID によるルーティング

オプションで、ルーティング・プログラムは、要求に関連付けられた CICS ユーザー ID (userid) によって要求をルーティングすることができます。通信域の DYRUSERID フィールドには、ユーザー ID が含まれています。ルーティングのため、またはルート選択エラーのために起動された場合、ルーティング・プログラムはこのフィールドの内容に基づいてルーティングを決定できます。

さまざまなタイプの要求でユーザー ID が設定される方法について詳しくは、[239 ページの『分散ルーティング・プログラムに渡されるパラメーター』](#)で DYRUSERID フィールドの説明を参照してください。

## ターゲット領域での異常終了の処理

ルーティングされた要求がターゲット領域で失敗した場合、CICS はトランザクションの異常終了でルーティング・プログラムを起動して、通信域のフィールド DYRABCDE に異常終了コードを返します。

この起動はターゲット領域で行われます。また、この起動が行われるのは、ルーティング領域で実行されたその前の呼び出しで、ターゲット領域で再起動するようにルーティング・プログラムが指定していた場合のみです。

ターゲット領域での異常終了を処理する方法として、以下の方法をお勧めします。

1. すべてのルート選択(およびルート選択エラー)呼び出しについて、ターゲット領域でルーティング・プログラムを(トランザクションの開始、終了、および異常終了で)再起動するようにルーティング・プログラムをコーディングします。
2. トランザクションの異常終了の場合に、ターゲット領域でルーティング・プログラムを呼び出すと、失敗した要求の詳細情報がルーティング領域に送信されます。例えば、RLS ファイルまたは共用データ・テーブルなどの共用リソースへの通信域を作成することがあります。
3. ルーティング領域のルーティング・プログラムは、事前設定された間隔で共用リソースを調べます。
4. ルーティング領域のルーティング・プログラムは、ルーティングされた要求が失敗したことを検出すると、以下のステップを実行します。
  - a. そのルーティング・セットからターゲット領域を除去します。
  - b. 別の領域で要求を再試行します。要求が成功するか、あるいは考えられるすべての AOR の試行が失敗するまで繰り返し試行します。後者の場合は、エラー応答をクライアントに戻します。

## 分散ルーティング・プログラムのリンク・チェックおよび情報

分散ルーティング・プログラムを作成する場合は、要求をルーティングする前にリンクが使用可能かどうかをチェックできます。また、要求がどのようにルーティングされたかについての情報を保持することもできます。

- ルーティング・プログラムを作成する場合は、要求をルーティングする前に、CICS Explorer 「**ISC/MRO 接続**」操作ビュー、または **EXEC CICS INQUIRE CONNECTION** および **INQUIRE IRC** コマンドを使用して、リンクが使用可能であることを確認できます。**EXEC CICS INQUIRE** および **EXEC CICS SET** コマンドについては、[システム・コマンド](#)で説明しています。
- 分散ルーティング・プログラムは作業単位環境の外部で実行されるので、プログラムでリカバリー可能なリソースを変更したり、ファイル制御要求や一時ストレージ要求を発行したりしてはいけません。
- 要求がどのようにルーティングされたかについての情報を保持するには、ユーザー・ルーティング・プログラム内で、情報をデータ・セットに書き込むなどして保持する必要があります。ルーティング・プログラムは分散されるので、トランザクション・ルーティング・セット内のすべての CICS 領域がそのデータ・セットにアクセスできなければなりません。
- 分散ルーティング・プログラムは RMODE ANY にすることができますが、AMODE 31 でなければなりません。

## 分散ルーティング・プログラムに渡されるパラメーター

さまざまなパラメーターが分散ルーティング・プログラムに渡されます。通信域はコピーブック DFHDYPDS によってマップされます。このコピーブックは、サポートされるどのプログラミング言語についても、該当する CICS ライブラリーにあります。

分散ルーティング・プログラムにも動的ルーティング・プログラムにも同じ通信域が渡されます。一方のルーティング・プログラムにとっては意味があっても、もう一方にとっては意味がないパラメーターもあります。一方のルーティング・プログラムには渡されるものの、もう一方には渡されることがないパラメーター値もあります。以下のリストでは、分散ルーティング・プログラムにとって意味のあるパラメーターのみ詳細に説明しています。分散ルーティング・プログラムに渡されることのないパラメーター値はリストしていません。例えば、**DYRTYPE** パラメーターの項目に値 X'4' がリストされていません。この値が分散ルーティング・プログラムに渡されることはないからです。動的ルーティング・プログラムに対するプログラム・リンク関連呼び出しでのみ渡されます。

分散ルーティング・プログラムとしても動的ルーティング・プログラムとしても同じプログラムを使用する場合、動的ルーティング呼び出しで意味のあるパラメーターと値の説明については、[213 ページの『動的ルーティング・プログラムに渡されるパラメーター』](#)を参照してください。

### DYRABCDE

ルーティングされた要求に関連付けられたトランザクションがターゲット領域で異常終了したときに戻される異常終了コードです。

このフィールドは、ルーティングされた要求を停止するために分散ルーティング・プログラムが呼び出されたときに意味を持ちます。ブランク以外のどの値も、ターゲット領域(分散ルーティング・プログラムの場合、ルーティング・プログラムが呼び出される領域でもある)でトランザクションが異常終了したことを示します。

他のタイプの異常終了の処理方法に関する一般情報については、[238 ページの『ターゲット領域での異常終了の処理』](#)を参照してください。

### DYRABNLC

異常イベント・コード、またはヌル。

このフィールドは、ルーティングされた要求を停止するために分散ルーティング・プログラムが呼び出されたときに意味を持ちます。ヌル以外のどの値も、要求がルーティングされた領域(分散ルーティング・プログラムの場合、ルーティング・プログラムが呼び出される領域でもある)で異常イベント(トランザクション異常終了以外のイベント)が発生したことを示します。エラーの原因が調べられて修正されるまで、ルーティング・プログラムは同じ領域にさらに要求をルーティングしてはなりません。

このフィールドは CICSplex System Manager で使用するためのものです。現時点では、リソース・マネージャーとしての Db2、IMS、IBM MQ、または VSAM RLS への接続が使用できないことの結果として設定されます。詳しくは、[ストーム・ドレーン作用の回避](#)を参照してください。

### DYRACMAA

分散ルーティング・プログラムでは使用されません。呼び出し時はゼロに設定されています。

### DYRACMAL

分散ルーティング・プログラムでは使用されません。呼び出し時はゼロに設定されています。

### DYRACTCMP

BTS アクティビティーが完了しつつあるかどうかを示します。プロセスがルーティングされているとき、DYRACTCMP はルート・アクティビティーが完了しつつあるかどうかを示します。

このフィールドは、BTS プロセスおよびアクティビティーのルーティングにのみ適用されます。このフィールドの内容は、トランザクションを停止するための呼び出しでのみ意味を持ちます。

可能な値は以下のとおりです。

**Y**

ルート・アクティビティーは BTS アクティビティーの最終活動化です。

**N**

ルート・アクティビティーは BTS アクティビティーの最終活動化ではありません。

## DYRACTID

ルーティングされている BTS アクティビティに対して CICS が割り当てた 52 文字のアクティビティ ID です。プロセスがルーティングされているとき、DYRACTID はルート・アクティビティの ID を戻します。

このフィールドは、BTS プロセスおよびアクティビティのルーティングにのみ適用されます。

## DYRACTN

ルーティングされている BTS アクティビティの名前です。プロセスがルーティングされているとき、DYRACTN はルート・アクティビティの名前 (DFHROOT) を戻します。

このフィールドは、BTS プロセスおよびアクティビティのルーティングにのみ適用されます。

## DYRBLGTH

分散ルーティング・プログラムでは使用されません。呼び出し時はゼロに設定されています。

## DYRBPNTR

分散ルーティング・プログラムでは使用されません。呼び出し時はゼロに設定されています。

## DYRCABP

標準の異常終了処理を CICS に続行させるかどうかを示します。

このフィールドは、分散ルーティング・プログラムでは使用されません。呼び出し時は Y に設定されています。

## DYRCHANL

プログラム・リンクまたは START コマンドにチャンネルが関連付けられていれば、そのチャンネルの名前です。このフィールドは、DPL 要求、非端末関連 START 要求、および端末関連 START 要求によって開始されたトランザクションのルーティングにのみ適用されます。他のタイプの要求の場合、またはコマンドにチャンネルが関連付けられていない場合、このフィールドにはブランクが入ります。

ルーティング・プログラムに渡されるのはチャンネルの名前であり、チャンネルのアドレスではないことに注意してください。そのため、コンテナの内容をルーティング・プログラムが調べたり変更したりすることはできません。

## DYRCOMP

CICS コンポーネント・コード。分散ルーティング・プログラムの呼び出しの場合は、以下のいずれかに設定されます。

### SH

スケジューラー・サービス・ドメイン。BTS プロセスおよびアクティビティのルーティングと、非端末関連 START 要求の場合。

### RZ

要求ストリーム・ドメイン。インバウンド Web サービス要求でのメソッド要求のルーティングの場合。

## DYRCOUNT

DYRFUNC が 0、1、または 3 に設定されたこの要求のために分散ルーティング・プログラムが呼び出された回数。プログラムが要求のルーティングを試行する回数を制限するには、このフィールドを使用します。

## DYRDTRRJ

**DTRTRAN** システム初期設定パラメーターで指定された共通トランザクション定義で定義されたトランザクションの処理を拒否するか受け入れるかを示します。

このフィールドは、分散ルーティング・プログラムでは使用されません。呼び出し時は N に設定されています。

## DYRDTRXN

ルーティングされるトランザクションが、**DTRTRAN** システム初期設定パラメーターで指定された共通トランザクション定義で定義されているか、特定のトランザクション定義で定義されているかを示します。

このフィールドは、分散ルーティング・プログラムでは使用されません。呼び出し時は N に設定されています。

## DYRERROR

DYRFUNC が 1 に設定された場合のみ、値があります。最後のルート選択試行時に発生したエラーのタイプを示します。以下の値を指定できます。

0

選択された SYSID が不明です。

1

選択されたシステムはサービス中ではありません。

2

選択されたシステムはサービス中ですが、使用可能なセッションがありません。

3

割り振り要求が拒否され、SYSIDERR がアプリケーション・プログラムに戻されました。このエラーは、以下のいずれかの理由で発生します。

- XZIQUE グローバル・ユーザー出口プログラムが、割り振りの拒否を要求した。
- CONNECTION リソース定義で指定された QUEUELIMIT 値に達したため、CICS が割り振り要求を自動的に拒否した。

4

割り振り要求のキューがパージされており、SYSIDERR が待機中のすべてのアプリケーション・プログラムに戻されました。このエラーは、以下のいずれかの理由で発生します。

- XZIQUE グローバル・ユーザー出口プログラムが、キューのパージを要求した。
- CONNECTION リソース定義で指定された MAXQTIME 限度に達したため、CICS がキューを自動的にパージした。

5

選択されたシステムはこの機能をサポートしていません。

BTS プロセスおよびアクティビティー、非端末関連 START 要求では、このエラーは、MRO または APPC 並列セッション・リンクによって接続されていない CICS 領域に分散ルーティング・プログラムが要求をルーティングしようとした場合に発生します。

インバウンド Web サービス要求では、このエラーは、分散ルーティング・プログラムが CICS TS for z/OS バージョン 3.1 より前の領域に要求をルーティングしようとした場合に発生します。

次の 6 つの値はすべて、START 要求のルーティングの試行に適用されます。これらのエラー状態の意味については、[START](#) を参照してください。

6

**EXEC CICS START** コマンドが LENGERR を戻しました。

8

**EXEC CICS START** コマンドが INVREQ を戻しました。

9

**EXEC CICS START** コマンドが NOTAUTH を戻しました。

C

**EXEC CICS START** コマンドが TRANSIDERR を戻しました。

D

**EXEC CICS START** コマンドが IOERR を戻しました。

E

**EXEC CICS START** コマンドが USERIDERR を戻しました。

F

ターゲット領域の XPCERES または XICERES グローバル・ユーザー出口プログラムが、戻りコード UERCRESU を設定しました。これは、ターゲット領域上の必要なリソースが使用不可であることを意味します。このエラー・コードは、プログラム・リンク、Link3270 ブリッジ、および非端末関連 START 要求の場合に設定されることがあります。

## DYRFUNC

分散ルーティング・プログラムのこの呼び出しの理由を通知します。可能な値は以下のとおりです。



0

ルート選択のために呼び出されました。  
この呼び出しはルーティング領域で発生します。

1

ルート選択でエラーが発生したために呼び出されました。  
この呼び出しはルーティング領域で発生します。

2

既にルーティングされた要求に関連付けられたトランザクションが正常に終了したので呼び出されました。  
この呼び出しはターゲット領域で発生します。

3

静的にルーティングされた要求の宛先を通知するために呼び出されました。  
この呼び出しはルーティング領域で発生します。以下の場合に適用されます。

#### **BTS のプロセスおよびアクティビティ**

**RUN ASYNCHRONOUS** コマンドが発行されたものの、BTS プロセスまたはアクティビティに関連付けられたトランザクションが DYNAMIC(NO) と定義されている。

#### **インバウンド Web サービス要求**

要求に関連付けられたトランザクションが DYNAMIC(NO) と定義されている。

#### **非端末関連の START 要求**

ROUTABLE(YES) と定義されたトランザクションは拡張 ルーティングの対象となるが、次の一方または両方が該当するため、動的 ルーティングの対象にならない。

- トランザクション定義で DYNAMIC(NO) が指定されている。
- START コマンドの SYSID オプションで明示的にリモート領域が指定されている。

動的ルーティングの対象となる非端末関連 START 要求について詳しくは、[START コマンドで呼び出されたトランザクションのルーティング](#)を参照してください。

4

ルーティングされた要求に関連付けられたトランザクションが異常終了したために呼び出されました。  
この呼び出しはターゲット領域で発生します。

5

トランザクション開始のために呼び出されました。ルーティングされた要求に関連付けられたトランザクションは、ターゲット領域で開始される場所です。  
この呼び出しはターゲット領域で発生します。

6

ターゲット領域に要求をルーティングしようとする試みを CICS が (成功であれ不成功であれ) 終了したために呼び出されました。

この呼び出しはルーティング領域で発生します。これは、ルーティング領域とターゲット領域が同じでない限り、このトランザクションに対するルーティング領域の責任が果たされたことを意味します。例えば、ルーティング・プログラムはこの呼び出しを使用して、トランザクションのために獲得したリソースを解放することができます。

DYRTYPE フィールドは、ルーティング要求または通知要求のタイプを通知します。

#### **DYRLEVEL**

ルーティングされた要求をターゲット AOR が正常に処理するために必要な、CICS のレベル。可能な値は以下のとおりです。

#### **X'00'**

現在サポートされているすべてのバージョンの CICS が要求を処理できます。

### **X'03'**

CICS TS は CICS TS for z/OS バージョン 3.1 以上でなければなりません。この値は、以下の要求の場合に設定されます。

- チャンネルが関連付けられている DPL 要求。

注：DPL 要求のルーティングは、動的 ルーティング・プログラムで扱われます。

- チャンネルが関連付けられている START 要求。
- インバウンド Web サービス要求。

### **DYRLPROG**

分散ルーティング・プログラムでは使用されません。呼び出し時はヌル文字に設定されています。

### **DYRNETNM**

分散ルーティング・プログラムでは使用されません。呼び出し時はヌル文字に設定されています。ターゲット領域を設定するためには、分散ルーティング・プログラムは DYRSYSID フィールドを使用する必要があります。

### **DYROPTER**

ルーティングされた要求に関連付けられたトランザクションがターゲット領域で開始されるか、(成功であれ不成功であれ) 終了するときに、ターゲット領域で分散ルーティング・プログラムを再呼び出しするかどうかを指定します。

このフィールドは、ルート選択、通知、およびルート選択エラー呼び出しで使用されます。可能な値は以下のとおりです。

#### **N**

トランザクションを開始、停止、または異常終了するための分散ルーティング・プログラム呼び出しは行われません。つまり、ターゲット領域では分散ルーティング・プログラムは呼び出されません。N がデフォルトです。

#### **Y**

ターゲット領域で分散ルーティング・プログラムが再呼び出しされます。

リモート CICS 領域にルーティングされる要求のほか、ローカルで実行される要求の場合にも、このオプションを指定できます。

### **DYRPROCCMP**

BTS プロセスが完了しつつあるかどうかを示します。

このフィールドは、BTS プロセスおよびアクティビティのルーティングにのみ適用されます。このフィールドの内容は、トランザクションを停止するための呼び出しでのみ意味を持ちます。

可能な値は以下のとおりです。

#### **Y**

この呼び出しは BTS プロセスの最終活動化です。

#### **N**

この呼び出しは BTS プロセスの最終活動化ではありません。

トランザクションの終了時にルーティング・プログラムが呼び出されたとき、ルーティング・プログラムはこのフィールドの値を使用して、トランザクション・アフィニティを終了するかどうかを判別できます。

### **DYRPROCID**

ルーティングされているアクティビティが属する BTS プロセスに対して CICS が割り当てた 52 文字の ID です。

このフィールドは、BTS プロセスおよびアクティビティのルーティングにのみ適用されます。

### **DYRPROCN**

ルーティングされているアクティビティが属する BTS プロセスの名前です。

このフィールドは、BTS プロセスおよびアクティビティのルーティングにのみ適用されます。

## DYRPROCT

ルーティングされているプロセスまたはアクティビティーが 属する BTS プロセスのプロセス・タイプです。

このフィールドは、BTS プロセスおよびアクティビティーのルーティングにのみ 適用されます。

## DYRPRTY

分散ルーティング・プログラムでは使用されません。呼び出し時はゼロに設定 されています。

## DYRQUEUE

DYRSYSID で識別されるリモート・システムが直ちに使用できるセッションがない場合に要求をキュー に入れるかどうかを指定します。

このフィールドは、分散ルーティング・プログラムでは使用されません。呼び出し時は Y に設定されて います。

## DYRRETC

進め方を CICS に通知する戻りコードが入ります。可能な 値は以下のとおりです。

**0**

要求をルーティングします。

### ゼロ以外の値

要求をルーティングしません。CICS は、BTS プロセスおよびアクティビティーの 要求をサービス 不能として扱います。231 ページの『ルート選択でエラーが発生した場合』のサービス不能要求の 説明を参照してください。START 要求は SYSIDERR 状態を受け取ります。

ルーティング・プログラムが呼び出されたときは必ず、DYRRETC が 0 に設定されています。ルート選 択のために呼び出された場合、またはルート選択でエラーが発生したために呼び出された場合、 DYRSYSID フィールドで指定された領域に CICS が要求をルーティングするようにするには、0 に設定 されたままにしておく必要があります。

通知、ルーティング完了、トランザクションの開始または停止、あるいは異常終了でルーティング・プ ログラムが呼び出されたときは、戻りコードを設定する必要はありません。どのようなコードが設定 されても CICS は無視します。

## DYRRTPRI

端末専有領域とアプリケーション専有領域の間の接続が MRO または IPIC である場合に、トランザクシ ョンのディスパッチ優先順位をアプリケーション専有領域に渡すかどうかを示します。

このフィールドは、分散ルーティング・プログラムでは使用されません。呼び出し時は N に設定されて います。

## DYRSRCTK

ルーティングされたトランザクションに使用されている MVS ワークロード管理サービスおよびレポー トのクラス・トークンです。ルーティング・プログラムはこの値を変更してはなりません。値は CICS によって設定され、CICSplex SM によって使用されます。非端末関連 START 要求の場合、このフィー ルドはゼロに設定されます。この値を変更しないでください。

## DYRSYSID

CICS 領域のシステム ID (SYSID)。このパラメーターの正確な意味は、以下のように DYRFUNC の値に よって異なります。

- DYRFUNC が 0 (ルート選択) に設定されている場合は、次の名前のいずれかが DYRSYSID に含まれま す。
  - インストールされているトランザクション定義の REMOTESYSTEM オプションに指定されている CICS 領域名
  - REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。

分散ルーティング・プログラムは、DYRSYSID の値を受け入れることも、CICS に戻る前に DYRSYSID を変更することもできます。

CICS に戻す SYSID がローカル SYSID と同じである場合、CICS はローカル領域で要求を実行します。

- DYRFUNC が 1 (ルート選択エラー) に設定されている場合は、直前の呼び出しで分散ルーティング・プログラムによって CICS に戻された CICS 領域名が DYRSYSID に含まれます。CICS がルーティングを再試行するようにするには、CICS に戻る前に DYRSYSID を変更する必要があります。
- DYRFUNC が 2 (ルーティングされた要求の終了) に設定されている場合は、完了したトランザクションが実行されたターゲット領域の名前が DYRSYSID に含まれます。この領域は、分散ルーティング・プログラムが呼び出された領域でもあります。
- DYRFUNC が 3 (通知) に設定されている場合:
  - BTS プロセスおよびアクティビティーの場合は、次の名前のいずれかが DYRSYSID に含まれます。
    - インストールされているトランザクション定義の REMOTESYSTEM オプションに指定されている CICS 領域名。
    - REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。
  - インバウンド Web サービス要求の場合は、次の名前のいずれかが DYRSYSID に含まれます。
    - インストールされているトランザクション定義 (DFHWS-TRANID コンテナで指定されたトランザクション用のトランザクション定義) の REMOTESYSTEM オプションで指定されている CICS 領域名。
    - REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。
  - 非端末関連 START 要求の場合は、次の名前のいずれかが DYRSYSID に含まれます。
    - **EXEC CICS START** コマンドの SYSID オプションに指定されているリモート CICS 領域名。
    - SYSID が指定されていない場合は、インストールされているトランザクション定義の REMOTESYSTEM オプションに指定されているリモート CICS 領域名。
    - REMOTESYSTEM が指定されていない場合は、ローカル CICS 領域のシステム名。

DYRSYSID の値を変更しても無視されます。

- DYRFUNC が 4 (トランザクション異常終了) に設定されている場合は、トランザクションが異常終了したターゲット領域の名前が DYRSYSID に含まれます。この領域は、分散ルーティング・プログラムが呼び出された領域です。
- DYRFUNC が 5 (トランザクション開始) に設定されている場合は、ルーティングされた要求が実行されるターゲット領域の名前が DYRSYSID に含まれます。この領域は、分散ルーティング・プログラムが呼び出された領域です。
- DYRFUNC が 6 (ルーティング完了) に設定されている場合は、CICS が (成功であれ不成功であれ) 要求をルーティングしようとしたターゲット領域の名前が DYRSYSID に含まれます。

## DYRTRAN

トランザクション名が含まれます。

これはルーティング領域でトランザクションを識別できる名前であることを注意してください。動的ルーティング・プログラムとは異なり、分散ルーティング・プログラムに渡されるのは、リモート・トランザクション名ではなくローカル・トランザクション名です。ターゲット領域への転送のために代替リモート・トランザクション名を指定することはできません。

## DYRTYPE

呼び出されているプログラムを必要としているルーティング要求のタイプです。分散ルーティング・プログラムに渡せる値は、以下のとおりです。

- 5** BTS プロセスまたはアクティビティー。
- 6** 非端末関連 START 要求 (データの有無にかかわらずチャンネルなし)。
- 7** インバウンド Web サービス要求でのメソッド要求。
- B** 非端末関連 START 要求 (チャンネルあり)。

## DYRUAPTR

DYRVER が 7 以上の場合、このフィールドには新しいユーザー域 DYRUSERN のアドレスが含まれます。新しいユーザー域のメカニズムにより、ルーティング・プログラムのソースが、通信域を作成した CICS リリースから独立します。これまでのユーザー域フィールド DYRUSER は、互換性の目的でのみ保持されています。

ユーザー域は、DYRUAREA DSECT によってマップできます。

DYRUAPTR が 7 より小さいシステムの場合、DYRUAPTR の内容は予測不能です。

## DYRUSER

1024 バイトのユーザー域。

このフィールドは互換性の目的でのみ保持されています。DYRUAPTR および DYRUSERN フィールドの説明を参照してください。

## DYRUSERID

要求に関連付けられた CICS ユーザー ID。

- BTS プロセスおよびアクティビティーの場合は、次のユーザー ID のいずれかが DYRUSERID に含まれます。
  - LINK ACQPROCESS または LINK ACTIVITY コマンドによって BTS プロセスまたはアクティビティーがアクティブ化された場合は、LINK を発行したトランザクションのユーザー ID。
  - DEFINE PROCESS または DEFINE ACTIVITY コマンドの USERID オプションで指定されたユーザー ID。
  - DEFINE コマンドで USERID が指定されなかった場合は、DEFINE コマンドを発行したトランザクションを実行しているユーザー ID。

BTS プロセスおよびアクティビティーに関連付けられるユーザー ID について詳しくは、IBM Knowledge Center の [プロセスおよびアクティビティーのユーザー ID](#) を参照してください。

- インバウンド Web サービス要求の場合は、要求ストリームに関連付けられたユーザー ID が DYRUSERID に含まれます。
- 非端末関連 START 要求の場合、DYRUSERID には以下が含まれます。
  - EXEC CICS START コマンドの USERID オプションで指定されたユーザー ID。
  - USERID が指定されていない場合は、START コマンドを発行したトランザクションを実行しているユーザー ID。

ルーティングまたはルート選択エラー(それぞれ DYRFUNC=0 または 1)のために呼び出された場合、このフィールドを検査することで、ルーティング・プログラムは、要求に関連付けられたユーザー ID に基づいて要求をルーティングできます。

## DYRUSERN

1024 バイトのユーザー域。

CICS は、特定のタスクのために分散ルーティング・プログラムを呼び出す前に、このユーザー域をゼロに初期化します。このユーザー域をルーティング・プログラムで変更することができます。変更された領域は、同じ要求のための後続のルーティング・プログラム呼び出しに渡されます。

1. ターゲット領域での最初の呼び出し(トランザクション開始)でルーティング・プログラムに渡されたユーザー域は、ルーティング領域でのユーザー域のコピーです。したがって、ターゲット領域でユーザー域に加えられた変更は、ルーティング領域内のユーザー域に影響を及ぼしません。例えば、トランザクションを開始するための呼び出しでユーザー域に加えられた変更は、ルーティング完了呼び出しに渡されるユーザー域に影響を及ぼしません。トランザクションを開始するための呼び出し後に後者の呼び出しが行われたとしても、同じです。
2. ターゲット領域での最初の(トランザクション開始)呼び出しに渡されるユーザー域は、トランザクション開始呼び出しが行われたルーティング領域での呼び出しから戻されたユーザー域のコピーです。具体的には、以下のとおりです。
  - ルート選択にエラーが含まれていなかった場合は、ルート選択呼び出しまたは通知呼び出しから戻されたユーザー域のコピーになります。



- ・ ルート選択エラーが発生した場合は、最後のルート選択エラー呼び出しから戻されたユーザー域のコピーになります。
- ・ ルーティング領域でのルーティング試行完了呼び出しから戻されたユーザー域のコピーになることはありません。ターゲット領域でのトランザクション開始呼び出しの前に後者の呼び出しが行われたとしても、同じです。

## DYRVER

動的ルーティング・インターフェースのバージョン 番号。 CICS Transaction Server for z/OS, バージョン 5 リリース 6 の場合、番号は 10 です。

## 分散ルーティング・プログラムの名前指定

提供されている分散ルーティング・プログラム・サンプルには、DFHDSRP という名前が付けられています。このプログラムのユーザー独自のバージョンを作成する場合は、異なる名前にすることができます。

### このタスクについて

システムがロードされた後、CICS Explorer 「領域」操作ビューまたは **EXEC CICS INQUIRE SYSTEM** コマンドを使用して、現在 CICS のものと特定される分散ルーティング・プログラムの名前を見つけます。フィールド DSRTPROGRAM には、現在のプログラムの名前が入っています。

### 手順

現在のプログラムを変更するには、以下のいずれかの方法を使用します。

- CICS Explorer 「領域」操作ビューを使用します。
- DSRTPGM システム初期設定パラメーターを使用します。
- EXEC CICS SET SYSTEM DSRTPROGRAM** コマンドを使用します。このコマンドに関するプログラミング情報については、[SET SYSTEM](#) を参照してください。

DFHDSRP 用のサンプル定義が提供されていますが、カスタマイズした分散ルーティング・プログラム用に新しいリソース定義をインストールする必要があります。

## 分散トランザクション・ルーティング・サンプル・プログラム

CICS 提供の分散ルーティング・プログラム・サンプルには、DFHDSRP という名前が付けられています。通信域を定義する、対応するコピーブックは DFHDYPDS です。

アセンブラ言語、COBOL、PL/I、C のソース・レベルのサンプルとコピーブックがあります。提供されているプログラムとコピーブック、およびそれらが収められている CICSTS56.CICS ライブラリーを、以下の表に要約しています。

表 19. 分散ルーティング・プログラム		
言語	メンバー名	ライブラリー
アセンブラ	DFHDSRP	SDFHSAMP
COBOL	DFHDSRP	SDFHCOB
PL/I	DFHDSRP	SDFHPL1
C	DFHDSRP	SDFHC370

表 20. コピーブック		
言語	メンバー名	ライブラリー
アセンブラ	DFHDYPDS	SDFHMAC
COBOL	DFHDYPDS	SDFHCOB
PL/I	DFHDYPDS	SDFHPL1

表 20. コピーブック (続き)		
言語	メンバー名	ライブラリー
C	DFHDYPDS	SDFHC370

ユーザー独自の分散ルーティング・プログラムを COBOL、PL/I、C、またはアセンブラー言語で作成でき、プログラムの名前を変更することができます。

DYRFUNC が 0 に設定されて呼び出されると、サンプル・プログラムは通信域のフィールド DYRSYSID で渡された SYSID を受け入れ、CICS に戻る前に DYRRETIC を 0 に設定します。DYRFUNC が 2、3、5、または 6 に設定されて呼び出されると、サンプル・プログラムは戻りコード 0 を設定します。DYRFUNC が 1 または 4 に設定されて呼び出されると、戻りコード 8 を設定します。

要求を動的にルーティングする場合は、DFHDSRP をカスタマイズするか、ユーザー独自のルーティング・プログラムに完全に置き換える必要があります。

## CICS-DBCTL インターフェース 状況プログラムの作成

CICS-DBCTL インターフェース 状況プログラム DFHDBUEX は、CICS-DBCTL インターフェースのサポートの一部を形成するユーザー取り替え可能プログラムです。これは、CICS が正常に DBCTL に接続したり、DBCTL から切断したりするときに、常にユーザー提供のコードを呼び出すように設計されています。これは CICS アプリケーション環境で実行され、特定のポイントで駆動されます。これにより、CICS-DBCTL インターフェースが初期化または終了するときに CICS-DL/I トランザクションを使用可能および使用不可にすることができます。

ENABLE コマンドでは、以下の場合に DFHDBUEX が呼び出されます。

- CICS が DBCTL に正常に接続した。これは、接続要求が CICS から DBCTL に発行された後に生じます。初期化の完了のために、制御出口 (DFHDBCTX) がデータベース・リソース・アダプター (DRA) によって呼び出されます。制御出口は制御トランザクション (CDBO) をポストします。そして、制御プログラム (DFHDBCT) は DFHDBUEX を呼び出します。

DISABLE コマンドでは、以下の場合に DFHDBUEX が呼び出されます。

- DBCTL からの切断要求が発行された。CICS-DBCTL メニュー・プログラム (DFHDBME) は、DBCTL から切断するための切断トランザクション (CDBT) を開始します。切断プログラム (DFHDBDSC) は DFHDBUEX を呼び出してから、インターフェース終了要求をアダプターに発行します。
- 制御トランザクション (CDBO) が以下のいずれかのイベントの通知を受けた。
  - DBCTL へのチェックポイント・フリーズ要求
  - DRA の異常終了
  - DBCTL の異常終了。

これらの各ケースで、制御プログラム (DFHDBCT) は DFHDBUEX を呼び出します。

DFHDBUEX への入力、DFHEICAP によってアドレス指定される通信域によって行われます。通信域のレイアウトを [248 ページの図 60](#) に示します。

DBUSHEAD	DS	OCL4	Standard Header	
DBUREQT	DS	CL1	Function Code	
DBUCOMP	DS	CL2	Component Code	Always "DB"
DBURES	DS	CL1	Reserved	
DBUREAS	DS	CL1	Reason for disconnection	
DBUSUFF	DS	CL2	DRA startup table suffix	
DBUDBCTL	DS	CL4	DBCTL identifier	

図 60. DFHDBUEX 通信域

パラメーター・リストには、次の情報が含まれています。

### DBUREQT

要求タイプ。機能コードには、以下の値のうちの 1 つがあります。

**DBUCONN (X'01')**

接続

**DBUDISC (X'02')**

切断。

**DBUREAS**

切断の理由。以下のフラグが含まれます。

**DBUMENU (X'01')**

メニューからの切断

**DBUDBCC (X'02')**

DBCTL へのチェックポイント・フリーズ入力

**DBUDRAF (X'03')**

DRA 失敗の発生

**DBUDBCF (X'04')**

DBCTL 失敗の発生。

**DBUSUFF**

DRA 開始テーブル接尾部。

**DBUDBCTL**

DBCTL 識別子。

## サンプルの CICS-DBCTL インターフェース 状況プログラム

提供されている CICS-DBCTL インターフェース 状況プログラム DFHDBUEX のソース・コードが、アセンブラ言語でのみ CICSTS56.CICS.SDFHSAMP ライブラリーに用意されています。通信域をマップするための対応するコピーブック DFHDBUCA が、CICSTS56.CICS.SDFHMAC にあります。

サンプル・プログラムは、(通信域で渡される) 入力パラメーターの存在を検査します。存在しなければ、呼び出し側プログラムに制御が戻ります。

次に、要求のタイプ (CONNECTION|DISCONNECTION) が判別され、該当する機能ルーチン (CONPROC|DISPROC) に分岐します。

トランザクションを有効にする方法と無効にする方法の例が、コメントの一部としてサンプルに含まれています。このプログラムを使用するには、DBCTL を使用するトランザクションが CSD で DISABLED と定義されている必要があります。

CICS でサポートされているどの言語でも、ユーザー独自の CICS-DBCTL インターフェース 状況プログラムをコーディングできます。ユーザー置換可能プログラムのアセンブルとリンク・エディットに必要なジョブ制御ステートメントに関する情報は、[337 ページの『ユーザー置換可能プログラムのアセンブルとリンク・エディット』](#)を参照してください。

## 3270 ブリッジ出口プログラムの作成

3270 ブリッジは、3270 端末を使用せずに 3270 ベースの CICS トランザクションを実行できるインターフェースを提供します。ブリッジ環境で代行受信端末コマンドを受け取るプログラムを作成できます。

ブリッジ出口は、3270 トランザクションを実行するために必要なデータを外部リソースとの間で送受信するためのメカニズムとして機能します。例えば、ブリッジ出口で IBM MQ コマンドを使用できます。そうすることで、CICS は IBM MQ キューに対してメッセージを GET および PUT することができます。

3270 ブリッジ出口とそのインターフェースについての 詳細な説明は、[3270 ブリッジの概要](#)を参照してください。

# XPLink プログラム用に Language Environment ランタイム・オプションをカスタマイズするためのプログラムの作成

ユーザー置き換え可能プログラム DFHAPXPO は、XPLINK オプションを指定してコンパイルされた C または C++ プログラムのための Language Environment エンクレープの初期化時に、CICS によって呼び出されます。

## DFHAPXPO

このプログラムは、XPLINK オプションを使用してコンパイルされた C または C++ プログラムを実行する各 Language Environment エンクレープの PIPI 事前初期設定フェーズの間にロードされます。これにより、デフォルトの Language Environment の実行時オプションを変更することができます。リセット可能な Language Environment オプションの詳細については、「[z/OS Language Environment プログラミング・ガイド](#)」を参照してください。このプログラムはアセンブラ言語で作成する必要があります。

### 実行時オプションの定義

このオプションは、人が読んで理解できるストリングで指定され、2 バイトのストリングの長さと、それに続く実行時オプションで構成されます。

すべての Language Environment 実行時オプションに許可される最大長は 255 バイトなので、各オプションはその省略バージョンを使用して指定し、変更内容は全体で 200 バイトに制限するようお勧めします。指定された値は、CICS によってチェックされずに Language Environment に渡されます。

いくつかの実行時オプションが設定されている、CICS 提供の DFHAPXPO モジュールが提供されています。モジュール・ソースは、これらのオプションを設定する方法の例を示します。次のようなコメントが含まれるセクションを見つけます。

```
* WHERE DO THE PIPI RUNTIME OPTIONS COME FROM ?
**
** 1. CEEDOPT - Installation default options
** 2. DFHAPXPO - These runtime options
** 3. CICS - Xplink options forced by DFHAPLX
** The options string passed to PIPI is shown at AP trace level 2. *
```

DFHAPLX は、内部 CICS モジュールです。

## アナライザー・プログラム

アナライザー・プログラムは、TCPIP SERVICE 定義と関連付けられます。これらのプログラムの主な役割は、URIMAP 定義によってアナライザー・プログラムの使用が指定されている場合や、URIMAP 定義が存在しない場合に、HTTP 要求の解釈を行うことです。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

アナライザー・プログラムは、CICS が HTTP クライアントであるときは呼び出すことができず、また Web サービス処理に対しても呼び出すことができません。アナライザー・プログラムは、CICS が HTTP サーバーであるときのみ呼び出すことができます。CICS に対し、CICS Web サポート・プロセス中のアナライザー・プログラムが HTTP サーバーとしての役割をすることについては、[HTTP サーバーとしての CICS に対する HTTP 要求および応答の処理](#)で説明されています。[Enabling CICS web support for CICS as a HTTP server](#) には、HTTP サーバーとしての CICS のアーキテクチャーを計画するのに役立つ情報が含まれています。

### アナライザー・プログラムと URIMAP 定義の関係

CICS Transaction Server for z/OS バージョン 3 リリース 1 の前は、HTTP サーバーとしての CICS に対するすべての HTTP 要求を、アナライザー・プログラムで解釈しました。現在では、[URIMAP](#) リソースが、HTTP 要求の処理を制御するための戦略的な方式です。これらの定義では、要求の URL を、要求を処理するアプリケーション・プログラムにマッチングすること、およびコンバーター・プログラムと別名トランザクションの使用を指定することで、アナライザー・プログラムの主要な機能を置き換えます。

ただし、処理ステージの一部を引き継ぐため、またその他のアクション (モニターまたは監査のアクションなど) を実行するために、選択した HTTP 要求について URIMAP 定義でアナライザー・プログラムを呼び出



すことがあります。アナライザー・プログラムに対して、アナライザーの機能を複製する URIMAP 定義の属性、すなわち CONVERTER (コンバーター・プログラム名)、TRANSACTION (別名トランザクション)、USERID (別名トランザクションのユーザー ID)、および PROGRAM (要求を処理するアプリケーション・プログラムの名前) を渡すことができます。アナライザー・プログラムでは、これらの属性のオーバーライドを選択することもできます。

CICS Transaction Server for z/OS バージョン 3 リリース 1 の前に CICS Web サポートが使用したのと同じプロセスに従って、URIMAP 定義を使用せずに HTTP 要求を直接、アナライザー・プログラムに渡すという選択も可能です。ただし、URIMAP 定義を使用しない場合、特定の HTTP 要求に対する CICS の応答方法を変更するには、アナライザー・プログラム内のロジックを変更することが必要になります。URIMAP 定義を使用すると、これらの変更をシステム管理タスクとして動的に実行することができます。また、URIMAP 定義の代わりにアナライザー・プログラムを使用して要求の処理を継続するにあたって、この観点から、HTTP/1.1 への準拠が必要な場合は、HTTP/1.1 仕様 (RFC 2616) に記述されている規則に従って URL 比較を実行するように、アナライザー・プログラムをコーディングする必要があります。

**注:** URIMAP で ANALYZER(YES) が指定されていても、その要求と一致する URIMAP 定義が検出された場合は、CICS 提供のサンプル・アナライザー・プログラム DFHWBADX および CICS 提供のデフォルト・アナライザー・プログラム DFHWBAAX は、提供時には、要求の分析をまったく実行しません。

アナライザー・プログラムを選択すると、リスナー・タスクは、高速で着信する HTTP 要求に対してユーザー・トランザクションを直接接続することができません。詳しくは、[直接接続ユーザー・トランザクションを使用した HTTP 要求の処理](#)を参照してください。

## エラー処理に対するアナライザー・プログラムの使用

現在、すべての HTTP 要求のパス処理にアナライザー・プログラムが必要なわけではありませんが、CICS Web サポートに使用されるそれぞれの [TCPIPService](#) リソースでは、依然としてアナライザー・プログラムを指定する必要があります。

リソース定義の URM 属性で、アナライザー・プログラムの名前を指定します。各 TCPIPService 定義に別々のアナライザーを指定することもできますし、複数の TCPIPService 定義に同じアナライザーを指定することもできます。URIMAP 定義からアナライザー・プログラムを呼び出す場合には、複数の異なるアナライザー・プログラムの中から選択することはできません。その TCPIPService 定義用に指定されているアナライザー・プログラムを使用するか、使用しないかのみの選択となります。

CICS が HTTP 要求にマッチングする URIMAP 定義を検出できない場合、その要求を処理するために、TCPIPService 定義に指定したアナライザー・プログラムが呼び出されます。この定義が見つからない状態は、要求の URL 入力時のユーザー・エラーが原因か、または適切な URIMAP 定義がインストールされていないために発生した可能性があります (URIMAP 定義は存在するが、使用不可に設定されている場合、その要求はアナライザー・プログラムではなく、Web エラー・プログラムによって処理されます)。

そのため、最小限の処置として、各 TCPIPService 定義に指定するアナライザー・プログラムには、認識できない HTTP 要求を処理するためのプロシーチャーを組み込み、適切なエラー応答を提供する必要があります。またユーザーは、URIMAP 定義によって処理されるはずであったと思われる要求を具体的に識別し、より適切なエラー応答を提供することができます。エラー状態でアナライザー・プログラムから出された出力は、Web エラー・プログラムに渡されます。この Web エラー・プログラムを使用して、HTTP 応答を変更することができます。[Web エラー・プログラム](#)では、この調整方法が説明されています。

CICS 提供のデフォルト・アナライザー・プログラム DFHWBAAX は、TCPIPService 定義で PROTOCOL(HTTP) が指定されている場合のデフォルトです。DFHWBAAX は、そのポートを使用するすべての要求を URIMAP 定義で処理する必要がある場合の基本エラー処理を行います。CICS Web サポートが CICS TS 3.1 以前で使用した URL 形式を使用しては、要求をサポートしません。URIMAP 定義で処理されない要求の処理を、アナライザー・プログラムで提供する場合、TCPIPService 定義で指定するアナライザー・プログラムは、CICS 提供のサンプル・アナライザー・プログラム DFHWBADX か、またはカスタマイズした独自のアナライザー・プログラムである必要があります。

## 一部の Web 非対応アプリケーション用、および非 HTTP メッセージ用にアナライザー・プログラムを使用する方法

Web 非対応のアプリケーションは、URIMAP 定義から直接呼び出されれば、正常に機能する可能性があります。しかし、一部は、アナライザー・プログラムからのみ提供可能な機能に依存している場合があります。



す。以下のような環境では、HTTP 要求の処理パスの中でのアナライザー・プログラムの使用が必要になる可能性があります。

- Web 非対応のアプリケーションおよびコンバーター・プログラムを使用して応答を作成する場合で、CICS TS バージョン 3 以前に受信したであろう応答と同一の応答を Web クライアントが必要とするために、CICS TS バージョン 3 以前との互換性処理用のフラグを立てる必要がある場合 (例えば、ユーザー作成のクライアントが、新しいエラー応答または追加の HTTP ヘッダーに関する作業を行う際に、問題が発生する可能性があります)。このフラグ `wbra_commarea` は、コンバーター・プログラムがストレージ・ブロック内で手動で応答を作成する場合にのみ機能します。コンバーター・プログラムが **EXEC CICS WEB API** コマンドを使用して応答を送信する場合、このフラグは無効です。
- Web 非対応のアプリケーションとコンバーター・プログラムを使用して応答を作成する場合で、ストレージ・ブロック内のコンバーター・プログラムに渡される Web クライアントの要求か、またはストレージ・ブロック内でコンバーター・プログラムが手動で作成する HTTP 応答のいずれかが、標準でないコード・ページ変換を必要とする場合。コンバーター・プログラムは、ストレージ・ブロック内で渡される HTTP 要求または応答のコード・ページ変換の設定を指定することはできません。処理パスにアナライザー・プログラムがない場合に CICS がコード・ページ変換用に使用する標準設定については、[261 ページの『コンバーター・プログラムの作成』](#)で説明しています。これらの標準設定が適していない場合や、コード・ページ変換が不要の場合は、処理パスの中でアナライザー・プログラムを使用して、代わりのコード・ページ変換設定を指定することができます。アナライザー・プログラムを使用する方法の代わりの選択として、ストレージ・ブロックを使用するのではなくコンバーター・プログラムで **EXEC CICS WEB API** コマンドを使用して、Web クライアントの要求の検査や、応答の作成を行うことができます。この場合、コード・ページ変換は通常どおりに **EXEC CICS WEB API** コマンドで指定することができます。

アナライザー・プログラムでこうした状態のいずれかに対処する必要がある場合、要求に対して URIMAP 定義をセットアップすることも可能ですが、その定義ではアナライザー・プログラムを指定する必要があります。

TCIPISERVICE 定義でユーザー定義の (USER) プロトコルを使用する非 HTTP 要求の場合、要求の処理にはアナライザー・プログラムが常に必要であり、URIMAP 定義は使用できません。非 HTTP 要求の処理方法については、[Web エラー・プログラム](#)で説明しています。

### 追加の処理に対するアナライザー・プログラムの使用

処理パスの中でアナライザー・プログラムを使用するかどうかは任意指定であるような状況では、以下の理由で、アナライザー・プログラムの使用を選択する場合があります。

- 要求の内容を基に、処理パスの要素への動的な変更を行うため。HTTP 要求の各 URL は、単一の処理パスを定義する単一の URIMAP 定義と突き合わせされます。アナライザー・プログラムは、要求の内容を解釈し、要素を変更することができます。そうした要素には、要求を処理するアプリケーション・プログラム、コンバーター・プログラムの関与、またはその要求に使用される別名トランザクションとユーザー ID などがあります。
- プロセスにモニター・アクションまたは監査アクションを導入するため。アナライザー・プログラムは、これを行う場所として適しています。
- 既存の CICS Web サポート・アーキテクチャーを CICS TS バージョン 2 からアップグレードするため、および既存のアナライザー・プログラムで、要求を処理している間に維持したい追加機能 (コンバーター・プログラムへの情報の引き渡しなど) が提供されているため。

[253 ページの『アナライザー・プログラムの作成』](#)では、アナライザー・プログラムが実行できる全範囲の機能について説明しています。

## アナライザー・プログラムと URIMAP 定義の置き換え

アナライザー・プログラムの要求処理機能は、URIMAP リソース定義に置き換えることができます。URIMAP リソース定義は、CICS システム・プログラミング・コマンドを使用して変更および制御できます。

URIMAP 定義は、要求の URL を一致させ、それらをアプリケーション・プログラムにマップしたり、コンバーター・プログラム、別名トランザクション、およびユーザー ID を指定するために使用できます。使用

しているアナライザー・プログラムが追加機能を提供している場合、引き続き URIMAP 定義の代わりにその機能を使用したり、その機能を URIMAP 定義と結合することができます。

URIMAP を使用するようにマイグレーションする場合は、以下の点に注意してください。

- 一度に少数の要求に対して、徐々に URIMAP リソース定義を導入できます。アナライザー・プログラムで実行される処理のタイプ、および要求を処理するアプリケーションのタイプに応じて、各要求の処理パスでアナライザー・プログラムの使用を続行するかどうかを選択できます。
- URIMAP リソース定義で処理される要求について、既存の URL を保持する代わりに、新しい URL を選択して公開することもできます。要求について古い処理パスの使用を中止する準備ができたなら、URIMAP 定義をセットアップして、古い URL から新しい URL に要求を永久的にリダイレクトすることができます。
- アナライザー・プログラムがすべての要求の処理パスに関係しなくなった場合でも、認識されない要求の基本的な処理手続きがアナライザー・プログラムに含まれていることを確認します。アナライザー・プログラムは、TCIPSERVICE 定義では引き続き必要であり、ユーザーが URL を誤って入力した場合などに要求を受け取ります。

URIMAP 定義を使用すると (アナライザー・プログラムは使用しない)、要求が直接接続経路指定の処理対象になります。つまり、CWBA またはそのトランザクションがソケット・リスナー・タスクによって直接接続されて、この要求が処理されます。このオプションにより、要求の処理に必要な CPU 時間が削減されます。詳しくは、[直接接続ユーザー・トランザクションを使用した HTTP 要求の処理](#)を参照してください。

## アナライザー・プログラムの作成

アナライザー・プログラムは、アセンブラー、C、COBOL、または PL/I で作成できます。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

### このタスクについて

アナライザー・プログラムの入出力パラメーターは、COMMAREA で渡されます。言語依存のヘッダー・ファイル、組み込みファイル、および COMMAREA をマップするコピーブックについては、[アナライザー・プログラムに関する参照情報](#)で説明します。

アナライザー・プログラムが実行できる機能の全範囲は、以下のとおりです。

- 要求の処理を継続するか、CICS がエラー応答を Web クライアント返すかを決定する。
- 要求の内容、および URIMAP 定義からコンバーター・プログラムに渡されたパラメーターを分析して、必要な後続の処理ステージ、および各ステージの実行に必要な CICS リソースを決定する (**EXEC CICS WEB API** コマンドをこの分析の間に使用する場合があります)。
- アプリケーション・プログラムに要求を渡す前に要求を処理するコンバーター・プログラムの名前を指定する。コンバーター・プログラムは、通常、Web 非対応のアプリケーション・プログラムと共に使用されます。ユーザー・トークンがアナライザー・プログラムに対して提供され、必要に応じてコンバーター・プログラムと通信します。Web クライアントの要求は、パラメーター・リスト内のポインターが指す 32 K のストレージ・ブロックを介してコンバーター・プログラムに渡されます。[コンバーター・プログラム](#)で、コンバーター・プログラムの機能について説明しています。
- 要求を処理して応答を提供するユーザー作成アプリケーション・プログラムの名前を指定する。
- 処理の残りのステージを処理する別名トランザクションのトランザクション ID を指定する。
- 別名トランザクションに関連付けるユーザー ID を指定する。
- ストレージ・ブロックを介してコンバーター・プログラムに渡された要求、およびコンバーター・プログラムでストレージ・ブロック内に手動で作成する応答のコード・ページ変換を指定または抑止する。このことは、**EXEC CICS WEB API** コマンドを使用して HTTP 要求を表示し、応答を作成するコンバーター・プログラムまたはユーザー作成アプリケーションに影響を与えません。コード・ページ変換は CICS から直接要求されます。[CICS Web サポートのコード・ページ変換](#)では、コード・ページ変換の処理について説明しています。
- アップグレードの目的で提供されている、Web 非対応アプリケーションが CICS TS バージョン 3 以前の互換性処理を必要とする場所を示すフラグ `wbra_commaarea` を指定します。このことは、**EXEC CICS**

WEB API コマンドを使用して HTTP 要求を表示し、応答を作成するコンバーター・プログラムまたはユーザー作成アプリケーションに影響を与えません。

- 要求ボディを変更する。変更されたすべての内容は、コンバーター・プログラムにストレージのブロックで渡されたデータで見ることができますが、**EXEC CICS** WEB API コマンドに渡されたデータでは見ることはできません。

CICS は、デフォルト・アナライザー・プログラム DFHWBAAX (詳しくは 258 ページの『[CICS 提供のアナライザー・プログラム DFHWBAAX](#)』を参照)、およびサンプル・アナライザー・プログラム DFHWBADX (詳しくは 259 ページの『[CICS 提供のサンプル・アナライザー・プログラム DFHWBADX](#)』を参照) を提供します。これらのアナライザーが要求を満たさない場合、独自のアナライザーを作成する必要があります。DFHWBADX を例として使用することができる場合があります。

ユーザーによる置換が可能なプログラムはすべて、CICS Web をサポートするシステムのローカルに存在しなければなりません。プログラムの自動インストールを使用しない場合は、アナライザー・プログラムおよびコンバーター・プログラムを含む CICS Web サポートが使用するすべてのユーザー置換可能プログラムを定義し、そのプログラム定義をインストールする必要があります。プログラムの自動インストールを使用する場合は、ユーザーによる置換が可能なプログラムを正しい属性でインストールする必要があります。アナライザー・プログラムは EXECKEY(CICS) を指定して定義する必要があることに注意してください。

ユーザーによる置換が可能なプログラムの詳細については、『[Developing system programs](#)』の『[Customizing with user-replaceable programs](#)』を参照してください。

## アナライザー・プログラムへの入力

入力パラメーターは COMMAREA のアナライザー・プログラムに渡され、要求の性質と内容、および URIMAP 定義によって提供されるすべての入力に関する情報を与えます。アナライザー・プログラムは、これらの値を受け取って出力パラメーターとして渡すことも、あるいは要求内容の分析に基づいて動的に指定変更することもできます。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

COMMAREA 内のすべてのパラメーターのリストと技術的説明については、[アナライザー・プログラム用のパラメーター](#)を参照してください。

入力パラメーターには、次の項目またはこれらの項目へのポインターが含まれます。

- アナライザー・パラメーター・リストの目印。
- クライアントおよびサーバー (HTTP サーバーとしての CICS) のコロン 16 進またはドット 10 進 IP アドレス。
- 要求が HTTP 要求であるかどうかを示す標識
- 要求について一致する URIMAP 定義が見つかったかどうかを示す標識。この標識が正の場合は、URIMAP 定義が、アナライザー・プログラムに追加の入力パラメーターを渡した可能性があります。
- HTTP バージョン。
- 要求方式。
- 要求に指定されたホスト名。ホスト・ヘッダー (絶対 URI の場合は要求 URL) から取得されます。HTTP/1.1 要求では、ホスト名は必須であるため、このパラメーターは常にアナライザーに渡されます。HTTP/1.0 要求では、ホスト名が提供されない場合があります。
- URL のパス構成要素。
- 要求で指定された照会ストリング。
- 要求の HTTP ヘッダー。

チャンク転送コーディングを使用して要求が送信された場合、後続のヘッダーがメインの要求ヘッダーと共にアナライザー・プログラムに渡されることはありません。

- 要求ボディ、またはそのうちの 32 KB のストレージ・ブロックに収まる部分。この要求ボディは、要求が含まれている別個のストレージ・ブロックへのポインターです。

SSL クライアント 認証を使用した接続で受け取られた HTTP 要求 の場合は、次のパラメーターも渡されます。

- クライアント 証明書から入手したユーザー ID

要求に一致する URIMAP 定義が見つかり、それによってアナライザー・プログラムが呼び出された場合は、その URIMAP 定義の後続のパラメーター (存在する場合) がアナライザー・プログラムに渡されます。

- アプリケーション・プログラムに渡す前に要求を処理する推奨されるコンバーター・プログラムの名前 (URIMAP 定義の CONVERTER 属性)。
- 要求を処理し、応答を提供する推奨されるユーザー作成アプリケーション・プログラムの 名前 (URIMAP 定義の PROGRAM 属性)。
- 処理の残りの段階をカバーする推奨される別名トランザクションのトランザクション ID (URIMAP 定義の TRANSACTION 属性)。
- 別名トランザクションに関連付ける推奨されるユーザー ID (URIMAP 定義の USERID 属性)。このユーザー ID は、クライアントによってユーザー ID が提供される場合はオーバーライドされることがあります。

**wbra\_urimap** 入力パラメーターを使用して、要求の処理パスで URIMAP 定義が使用されたかどうかをテストすることができます。

URIMAP 定義の代わりにアナライザー・プログラムを使用して要求を処理する場合、この点に関して HTTP/1.1 に準拠するには、HTTP/1.1 仕様の規則に従って URL 比較を実行するようにアナライザー・プログラムをコーディングする必要があります。これらの規則では、スキーム名とホスト名は大文字小文字を区別せず比較されますが、パスは大文字小文字を区別して比較されます。比較の前にすべての構成要素がアンエスケープされます。CICS は、URL と URIMAP 定義を比較する場合、これらの規則に従います。

必要に応じて、**EXEC CICS WEB API** コマンドを使用して HTTP 要求を調べることもできます。**EXEC CICS WEB** コマンドを使用すると要求の分析の精度と完全性が上昇する場合があります。特に、幅広い内容と用法がある HTTP ヘッダーを調べる際に有効です。また、**EXEC CICS WEB** コマンドを使用すると、後続処理の決定要素になる可能性のある、要求の照会ストリングまたはフォーム・フィールド情報を見つけ、抽出する処理が単純化されます。

EXTRACT TCPIP コマンドを使用して、処理中のクライアント 要求に関する以下の 情報を入手できます。

- Web クライアントの IP アドレス
- DNS サーバーによって認識されている Web クライアントのホスト名
- Web クライアントが接続要求の送信に使用したポートの番号
- サーバー (HTTP サーバーとしての CICS) の IP アドレス
- 使用されている認証のタイプ
- 使用している SSL サポートのレベル
- 要求に関連付けされた TCPIP SERVICE リソース定義

### アナライザー・プログラムからの出力

アナライザー・プログラムは、出力を COMMAREA で提供します。出力 には、応答コード、および処理ステージをさらに指定して、情報をコンバーター・プログラムと共用するためのオプションの出力パラメーターの範囲が含まれます。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと 関連 ガイダンス情報が含まれています。

COMMAREA 内のすべてのパラメーター のリストと技術的説明については、[アナライザー・プログラム用のパラメーター](#)を参照してください。

アナライザー・プログラムは、COMMAREA で以下のような出力を生成する必要があります。

- 応答コード。
  - アナライザー・プログラムが URP\_OK の応答コードを戻した場合、処理は次のステップへ進みます。
  - アナライザー・プログラムが他の値を戻した場合、CICS は Web クライアントにエラー 応答を戻します。応答はユーザー置換可能 Web エラー・プログラムを使用して変更できます。[CICS Web サポート](#)



のデフォルトの状況コードおよびエラー応答では、アナライザーからの戻りコードが CICS により Web クライアントに戻される状況コードにマップする方法を説明します。

アナライザー・プログラムは、以下のような出力を提供する場合もあります。

- ユーザー作成アプリケーション・プログラムに渡す前に要求を処理する場合に使用されるコンバーター・プログラムの名前。
  - コンバーター・プログラム名が URIMAP 定義から入力された場合、この名前を受け入れるか指定変更できます。
  - コンバーター・プログラムは不要であるということをアナライザーが示している場合は、要求の先頭の 32K バイトがストレージのブロックでユーザー作成アプリケーション・プログラムに渡されます。Web 対応アプリケーションは、これを無視して、**EXEC CICS WEB API** コマンドを使用し、要求を読み取ります。
- 要求を処理して応答を提供するアプリケーション・プログラムの名前。
  - プログラム名が URIMAP 定義から入力された場合、この名前を受け入れるか指定変更できます。
  - コンバーター・プログラムを使用している場合は、コンバーター・プログラムからプログラム名を指定または指定変更できます。コンバーターをこのように使用して、要求で処理中の複数のプログラムを含めることができます。
- 処理の残りのステージを処理する別名トランザクションのトランザクション ID。トランザクション ID が URIMAP 定義から入力された場合、この名前を受け入れるか指定変更できます。
- 別名トランザクションに関連付けるユーザー ID。ユーザー ID が URIMAP 定義から入力された場合、この名前を受け入れるか指定変更できます。ユーザー ID が指定されていない場合に、CICS がどのようにして決定するかを以下に示します。
  - ユーザー ID が URIMAP 定義から入力された場合、この ID が使用されます。
  - HTTP 要求がクライアント認証に SSL を使用している場合は、ユーザー ID はクライアント証明書から入手されます。
  - その他の場合は、CICS のデフォルト・ユーザー ID が使用されます。
- コンバーター・プログラムがストレージのブロックで手動で作成する場合の、要求を含むストレージの 32K ブロックのコード・ページ変換に関連するパラメーター、および応答ボディのコード・ページ変換に関連するパラメーター。

**注：**このことは、**EXEC CICS WEB API** コマンドを使用して HTTP 要求を表示し、応答を作成するコンバーター・プログラムまたはユーザー作成アプリケーションに影響を与えません。コード・ページ変換は CICS から直接要求されます。

以下の 2 つのいずれかの方法で、要求を含むストレージのブロックの変換のためのパラメーターを指定できます。

- Web クライアントにより使用される文字セットを指定するパラメーターのペア (wbra\_characterset)、およびアプリケーション・プログラムに適切なホスト・コード・ページ (wbra\_hostcodepage) として。この方法でパラメーターを指定すると、コード・ページ変換テーブル (DFHCNV) のエントリーは必要とされません。
- DFHCNV コード・ページ変換テーブルのエントリーに対するキーとして (wbra\_dfhcnv\_key)。これはアップグレードを目的とする場合を除いて推奨されません。

これらのパラメーターのいずれも指定しないと、CICS では、250 ページの『アナライザー・プログラム』に記載された標準設定を使用したテキスト・メッセージの変換をデフォルトで行います。ストレージのブロックの要求および応答のコード・ページ変換を抑止する場合は、wbra\_dfhcnv\_key をヌルまたはブランクに設定します。

- コンバーター・プログラムを使用する Web 非対応アプリケーションを示すフラグでは、CICS TS バージョン 3 以前の互換性処理 (wbra\_commarea) が必要になります。このフラグはアップグレードの目的のために提供されます。このフラグは、Web クライアントが CICS TS バージョン 3 以前に受信した応答と同一の応答を必要とする特定の状況で、**EXEC CICS WEB API** コマンドを使用しないアプリケーションによってのみ使用されます (つまり、ストレージのブロックで応答を手動で作成します)。このフラグを設定すると、以下のようになります。



- CICS は、HTTP/1.1 メッセージに通常挿入される応答ヘッダーを追加しません。CICS Transaction Server for z/OS バージョン 3 リリース 1 の前にクライアントに送信されるヘッダーのみが使用されます。
- エラー処理が必要な場合、CICS は、Web クライアントの HTTP バージョンに関係なく、HTTP/1.0 応答に適切で、HTTP/1.0 応答とラベルが張られたエラー応答を送信します。CICS は、通常、HTTP/1.1 クライアントに HTTP/1.1 エラー応答を使用して応答しますが、このことにより、アプリケーションが通常 HTTP/1.1 応答を送信するとクライアントに見なされる場合があります。
- アナライザー・プログラムとコンバーター・プログラムで情報を共有するために使用される 8 バイトのユーザー・トークン。このメカニズムについては、257 ページの『アナライザー・プログラムとコンバーター・プログラムでのデータの共用』を参照してください。
- 要求ボディの長さの変更値。

アナライザーは、要求の内容を変更できます。

- 変更されたデータは、オリジナルのデータよりも短くなる場合もあれば、同じ長さの場合もあります。要求ボディを長くすることはできません。
- 変更されたすべての内容は、コンバーター・プログラムに渡されたデータで見ることができますが、**EXEC CICS WEB API** コマンドに渡されたデータでは見ることはできません。

## アナライザー・プログラムとコンバーター・プログラムでのデータの共用

CICS は、処理ステージによるデータの共用を可能にする 3 つのパラメーターを アナライザー・プログラムとコンバーター・プログラム間でやり取りします。

### user\_data ポインター

このパラメーターには、ステージ間でやり取りする 32K のストレージ・ブロックのアドレスが含まれています。アナライザー・プログラムへの入り口で、ポインターは HTTP 要求が含まれているストレージ・ブロックを指し示します。コンバーター・プログラムのエンコード機能が完了すると、CICS Web サポートはこのポインターを使用して、HTTP 応答が含まれているストレージ・ブロックを見つけて（代わりに、応答を作成するために **EXEC CICS WEB API** コマンドが使用された場合は除きます）。

アナライザー・プログラムではポインターの値を変更してはなりません。ただし、ポインターによって示されたストレージ・ブロックの内容は変更できます。

コンバーター・プログラムとユーザー作成アプリケーション・プログラムの間で、ポインターを無変更のままステージ間でやり取りすることもできますし、1 つのプログラムで GETMAIN コマンドを出し、新規に取得したストレージをポインターに渡すこともできます。

### user\_data 長

このパラメーターは、user\_data ポインターによって示されたストレージ・ブロックの長さです。

### ユーザー・トークン

ユーザー・トークンは、アナライザー・プログラムとコンバーター・プログラムによって共用される 8 バイトのフィールドです。ユーザー・トークンには、以下のような任意の情報を含めることができます。

- 少量の共用情報はユーザー・トークンに直接渡すことができます。
- 大量の情報を渡すには、1 つのプログラムで GETMAIN コマンドを出して共用作業域用のストレージを獲得することができます。ユーザー・トークンを使用して共用ストレージのアドレスを渡します。

各プログラムでユーザー・トークンの内容を変更することができます。例えば、アナライザー・プログラムからコンバーター・プログラムのデコード機能へ渡すときのユーザー・トークンの意味と、エンコード機能へ渡すときのユーザー・トークンの意味を違ったものにすることができます。

アナライザー・プログラムは、コンバーター・プログラムに渡されるパラメーター・リスト内の任意のパラメーターを変更することができます。ポインターを変更することはできませんが、そのポインターによって指し示されるデータは変更可能です。各フィールドの長さを変更することはできません。

**注:** アナライザー・プログラムとコンバーター・プログラムはそれぞれ異なる CICS タスクのもとで実行されます。したがって、アナライザー・プログラムで GETMAIN コマンドを出す場合は、ストレージがコンバーター・プログラムで可視になるのであれば、SHARED オプションをコーディングする必要があります。一般に、SHARED オプションで獲得されたストレージは、CICS によって自動的に解放されないため、ユ

ーザー・プログラムがストレージを必要としなくなった時点で FREEMAIN コマンドを出す必要があります。しかし、HTTP 応答が Web クライアントへ送信された後は、user\_data ポインターによって示されたストレージを CICS が解放します。

## アナライザー・プログラムからのエスケープ・データまたはアンエスケープ・データの選択

構文解析のためにアナライザー・プログラムに渡された HTTP 要求はエスケープ形式になっています。URL またはメッセージ・ボディのフォーム・データ内の予約文字または除外文字は、%xx シーケンス として表されます。ここで、xx は、予約文字の ASCII 16 進表記です。アナライザーは、ストレージの 32K ブロック内の要求を、エスケープ・シーケンスをそのまま使用したエスケープ形式で以後の処理段階に渡したり、エスケープ・シーケンスを元の文字に戻したアンエスケープ形式で渡したりできます。**EXEC CICS WEB API** コマンドを使用した Web アプリケーション・プログラムは応答を受け取る際にこの仕組みを使用せず、CICS から直接、アンエスケープの処理を要求します。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

エスケープとその目的については、[DPL のローカル・リソースの定義](#)を参照してください。エスケープおよびアンエスケープは HTTP 要求の以下の要素にのみ適用されます。

- ・照会ストリングを含む、照会行の URL 部分。照会ストリングは GET メソッドを持つフォームからのデータの場合があります。
- ・POST メソッドを持ったフォームから戻されたフォーム・データおよびデフォルトのエンコードの **application/x-www-form-urlencoded**。このデータはメッセージ・ボディで表されます。フォーム・データについて詳しくは、[MRO のアクセス方式の選択](#)を参照してください。

ストレージの 32K ブロック内の要求をアンエスケープ形式で渡す場合、アナライザーはデータをエスケープ形式から アンエスケープ形式に変換することもできるし、CICS に変換を行わせることもできます。

- ・要求をエスケープ形式で渡すには、アナライザーで WBRA\_UNESCAPE を WBRA\_UNESCAPE\_NOT\_REQUIRED に設定します。WBRA\_UNESCAPE\_NOT\_REQUIRED はデフォルト値です。
- ・要求をアンエスケープ形式で渡し、その変換を CICS に行わせる場合は、アナライザーで WBRA\_UNESCAPE を WBRA\_UNESCAPE\_REQUIRED に設定します。
- ・アナライザーによる変換が行われた後で要求をアンエスケープ形式で渡すには、WBRA\_UNESCAPE を WBRA\_UNESCAPE\_NOT\_REQUIRED に設定します。

**EXEC CICS WEB API** コマンドを使用した Web 対応アプリケーション・プログラムは、応答の送受信の際に COMMAREA の仕組みを使用せず、CICS から直接、アンエスケープの処理を要求します。**EXEC CICS WEB API** コマンドを使用する Web 対応アプリケーション の場合、WEB READ FORMFIELD コマンドまたはフォーム・フィールド・ブラウズ・コマンドを使用して要求からデータを抽出すると、CICS がアンエスケープ処理を実行し、データはアンエスケープ形式で戻されます。WEB EXTRACT コマンドを使用して要求から照会ストリングを抽出する場合、データはエスケープ形式で戻されます。

CICS Web サポート または CICS ビジネス・ロジック・インターフェースのいずれかを介して実行できる COMMAREA インターフェースを持つアプリケーション を作成する場合は、WBRA\_UNESCAPE が WBRA\_UNESCAPE\_NOT\_REQUIRED に設定され、すべてのアンエスケープ処理がアプリケーションに委任されていることを確認する必要があります。この処置が行われていないと、CICS ビジネス・ロジック・インターフェース はアンエスケープ・データをアプリケーションに渡し、CICS Web サポートはエスケープ・データをアプリケーションに渡すことになり、予期しない結果が生じることがあります。

## CICS 提供のアナライザー・プログラム DFHWBAAX

CICS は、デフォルトのアナライザー・プログラム DFHWBAAX を提供しています。DFHWBAAX は、CICS Web サポート用に使用される TCP/IP SERVICE リソース定義のエラー処理関数を提供します。これは、1 つのポートを使用するすべての要求が、URIMAP 定義を使用して処理される場合に適しています。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

CICS は、アセンブラの DFHWBAAX のソース・コードのみ、提供しています。

DFHWBAAX は、PROTOCOL(HTTP) を指定する TCIPSERVICE 定義のデフォルトのアナライザー・プログラムです。

DFHWBAAX は、COMMAREA の標準アナライザー・プログラムと同じ入出力パラメーターを受け取ります。提供時の状態では、これらのパラメーターのほとんどは利用されていません。その代わりに、以下のような簡素化された処置が行われます。

- DFHWBAAX は、URIMAP で ANALYZER(YES) が指定されている場合でも、その要求と一致する URIMAP 定義が検出された場合は、それ以上の処理は行いません。wbra\_urimap 入力パラメーターを使用して URIMAP 定義の存在を検査し、結果が肯定の場合は、その要求 URL に対する一切の分析を行わずに戻します。つまり、別名トランザクション、コンバーター・プログラム (使用されている場合)、およびアプリケーション・プログラム用の URIMAP 定義で指定されている設定が自動的に受け入れられ、それ以降の処理工程を決定するために使用されます。
- 一致する URIMAP 定義が見つからない場合、DFHWBAAX はユーザーによる置換が可能な Web エラー・トランザクション・プログラム DFHWBERX に制御を渡して、エラー応答を作成します。これは、wbra\_server\_program 出力パラメーターを使用して、DFHWBERX を、その要求を処理するアプリケーション・プログラムとして設定することで達成されます。DFHWBAAX はそれ以外の変更を COMMAREA に加えません。制御を受け取ると、DFHWBERX は 404 (Not Found (未検出)) 状況コードを持つ HTTP 応答か SOAP 障害応答のいずれかを、Web クライアントによって作成される要求に応じて提供します。

DFHWBAAX は、標準的な範囲の応答、URP\_OK、URP\_EXCEPTION、および URP\_INVALID を使用します。提供時、DFHWBAAX には理由値は作成されません。理由が URP\_OK 以外の場合は、処理中のエラーがあり、制御は、Web エラー・アプリケーション・プログラム DFHWBERX ではなく、ユーザーによる置換が可能な Web エラー・プログラム DFHWBEP に渡されることを意味します。

## CICS 提供のサンプル・アナライザー・プログラム DFHWBADX

CICS には、作業サンプル・アナライザー・プログラム DFHWBADX が提供されています。アナライザー・プログラムによる要求処理を、URIMAP 定義とともに、または URIMAP 定義の代わりに、提供する必要がある場合は、独自のアナライザー・プログラムを作成する際の開始点として、DFHWBADX を使用することができます。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

CICS は、以下の言語のソース・コードを提供しています。

- DFHWBADX (アセンブラー)
- DFHWBAHX (C)
- DFHWBALX (PL/I)
- DFHWBAOX (COBOL)

DFHWBADX は、提供された状態では、URIMAP で ANALYZER(YES) が指定されている場合でも、その要求と一致する URIMAP 定義が見つかったときは要求の分析を行いません。つまり、別名トランザクション、コンバーター・プログラム、およびアプリケーション・プログラム用の URIMAP 定義で指定されている設定が自動的に受け入れられ、それ以降の処理工程を決定するために使用されます。

DFHWBADX は wbra\_urimap 入力パラメーターを使用して URIMAP 定義の存在を検査し、結果が肯定の場合は、その要求 URL に対する一切の分析を行わずに戻ります。独自のアナライザー・プログラムを作成して URIMAP 定義と相互作用するようにする場合、DFHWBADX の処理のこの特徴はコピーしないでください。アナライザー・プログラムの処理を別の方法で変更するために、wbra\_urimap 入力パラメーターを検査したい場合があります。例えば、URIMAP 定義からの入力パラメーターを基に分析を行うか、要求 URL で直接分析を行うかを決定するために、パラメーターの検査を行う場合があります。

### DFHWBADX による要求 URL の解釈方法

DFHWBADX は、URL のパス構成要素が以下のような構文を持つ HTTP 要求を解釈します。

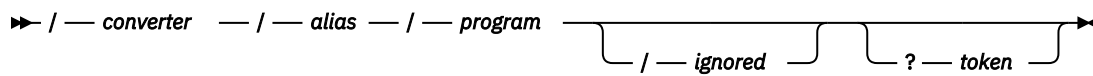


図 61. DFHWPBX によって解釈されるパス構成要素の構文

アナライザー・プログラムによって処理されたフィールドはすべて、大文字に変換されます。変換後は、以下のようになります。

#### converter

要求に使用するコンバーター・プログラムの名前を指定します。最高 8 文字の長さにすることができます。

特殊なケースとして、4 文字値 'CICS' は、コンバーター・プログラムを使用していないことを表します。URIMAP 定義が設定されているコンバーター・プログラムを使用する方法については、[コンバーター・プログラム](#)を参照してください。

#### alias

以降の要求処理のための別名トランザクションのトランザクション ID を指定します。最高 4 文字の長さにすることができます。

#### program

要求を処理するために使用する CICS アプリケーション・プログラムの名前を指定します。最高 8 文字の長さにすることができます。

#### ignored

パスのこの部分を、DFHWPBX は無視します (ただし、コンバーター・プログラムまたはアプリケーション・プログラムが使用する可能性があります)。

#### token

最初の 8 バイトは、コンバーター・プログラムに渡されるユーザー・トークンを指定します。トークンの先頭 8 バイトの後に続くデータを、DFHWPBX は無視します (ただし、コンバーター・プログラムまたはアプリケーション・プログラムが使用する可能性があります)。

パスの例として、/cics/cwba/dfh\$wb1a では、以下のとおりです。

- コンバーター・プログラムは使用されません。
- 別名トランザクションは CWBA です。
- CICS アプリケーション・プログラムは DFH\$WB1A です。

オリジナルの HTTP 要求から派生した出力に加え、DFHWPBX は以下の出力も設定します。

- コード・ページ変換テンプレートは DFHWPBU です。このテンプレートは、サンプル変換テーブル DFHCNVW\$ に定義されており、ASCII Latin-1 文字セット (コード・ページ ISO 8859-1) と EBCDIC Latin 文字セット (コード・ページ 037) のデータを変換します。このサンプル変換テーブルは、なにも構成を加えずに使用できますが、出力パラメーター wbra\_characterstet および wbra\_hostcodepage を wbra\_dfhcnv\_key 出力パラメーターの代わりに使用すると、より強力な制御が提供され、変換テーブルの使用を回避できることに、注意してください。
- DFHWPBX は、要求をエスケープ形式で渡し、WBRA\_UNESCAPE\_NOT\_REQUIRED を設定します。

### DFHWPBX からの応答

DFHWPBX によって作成された応答の意味は、以下のとおりです。

#### URP\_OK

アナライザーは、要求がデフォルトの HTTP 要求形式に準拠していることを検出し、別名用の適切な出力を生成しました。

#### URP\_EXCEPTION

アナライザーは、要求がデフォルト形式に準拠していないことを検出しました。次のような理由コードが提供されます。



- 1 リソースの長さが 6 未満でした (DFHWBADX によって認識される URL 形式で、可能な最短のリソース指定は /A/B/C です。これは、コンバーター A を使って トランザクション B のもとでプログラム C を実行するよう要求します)。この応答と理由は、着信要求が HTTP 要求でないときに使用されるものです。
- 2 リソース指定が "/" で始まっていませんでした。
- 3 リソース指定が 1 つの "/" を含んでいましたが、3 個未満です。
- 4 リソース指定の中のコンバーター名の長さが 0 であったか、または 8 を超えていました。
- 5 リソース指定の中の トランザクション名の長さが 0 であったか、または 4 を超えていました。
- 6 リソース指定の中の CICS アプリケーション・プログラム名の長さが 0 であったか、または 8 を超えていました。

応答および理由コードは、メッセージ DFHWB0723 に表示されます。状況コードが 400 (Bad Request (不正な要求)) のエラー応答が、Web クライアントに返されます。ユーザーによる置換が可能な Web エラー・プログラム DFHWBEP を使用すると、この動作を変更することができます。

#### URP\_INVALID

目印が無効です。これは内部エラーを示します。

## コンバーター・プログラムの作成

コンバーター・プログラムを作成するには、デコード機能およびエンコード機能を構成し、またコード・ページ変換について考慮する必要があります。

コンバーター・プログラムは、アセンブラー、C、COBOL、または PL/I で作成することができます。言語依存のヘッダー・ファイル、組み込みファイル、およびコピーブックについては、[コンバーター・プログラムに関する参照情報](#)で説明しています。



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

### デコード機能: HTTP 要求の表示および処理

コンバーター・プログラムのデコード機能は、Web クライアントから HTTP 要求とその要求に関する詳しい情報を提供するパラメーター・リストを受け取ります。HTTP 要求は、パラメーター・リストによって指示されている 32K のストレージ・ブロック内のコンバーター・プログラムに渡されます。要求はすでに、メソッド、要求ヘッダー、およびボディなどの、いくつかの別々の要素に分割されています (要求がストレージ・ブロックに対して長すぎて収まらない場合、残りのデータはコンバーター・プログラムに渡されないことに、注意してください)。処理パスでアナライザー・プログラムが使用される場合、アナライザー・プログラムが要求の内容を変更している可能性があります。

CICS Web サポートのコンバーター・プログラムでは、必要であれば、**EXEC CICS WEB API** コマンドを使用して HTTP 要求を検査することができます。WEB EXTRACT コマンドは、その要求に関する情報 (メソッドやバージョンなど) を取得します。WEB READ HTTPHEADER コマンドまたは HTTPHEADER 表示コマンドは、HTTP ヘッダーの読み取りに使用できます。WEB RECEIVE コマンドは、要求のボディの受け取りに使用することができます。いずれかの **EXEC CICS WEB API** コマンドを使用する場合、これらのコマンドが Web クライアントの要求からオリジナルの情報を戻すこと、および、アナライザー・プログラムによって加えられた変更を参照するためには使用できないことに、注意してください。アナライザー・プログラムによる変更は、コンバーター・プログラムに直接渡されるパラメーター・リストおよびストレージ・ブロックでのみ見ることができます。

応答にデータを提供するユーザー作成のアプリケーション・プログラムの名前は、パラメーター・リストで提供します。パラメーター・リストは、URIMAP 定義から取得されるか、アナライザー・プログラムに



よって設定されます。アナライザー・プログラムを使用する場合、このアナライザー・プログラムが追加情報をユーザー・トークン内のコンバーター・プログラムに直接提供することができます。

Web クライアントの要求に関して取得した情報を使用して、コンバーター・プログラムのデコード機能は以下のことを行う必要があります。

- 要求の処理を継続するか、CICS がエラー応答を Web クライアント返すかを決定する。
- 要求を処理して応答を提供するユーザー作成アプリケーション・プログラムの名前を指定する。すでに URIMAP 定義またはアナライザー・プログラムから名前が入力されている場合、コンバーター・プログラムはその名前を受け入れるか、または変更することができます。
- ユーザー作成のアプリケーション・プログラムに渡される COMMAREA を構成する。COMMAREA には、そのアプリケーション・プログラムが受け入れられる入力形式に変換された、Web クライアント要求からのデータが入っています。HTTP 要求が入っているストレージ・ブロックを再使用することもできますし、新しい COMMAREA を指定することもできます。

## エンコード機能: 応答の作成

ユーザー作成のアプリケーション・プログラムが、コンバーター・プログラムによって提供された入力を使用してその処理を行った場合、コンバーター・プログラムのエンコード機能がアプリケーション・プログラムから出力 COMMAREA を受け取ります。このデータを使用して、コンバーター・プログラムのエンコード機能は以下のことを行う必要があります。

- データを供するため給に複数のアプリケーション・プログラムが必要な場合は、残りのアプリケーション・プログラムを呼び出す。これを行うために、エンコード機能は、デコード機能を再度呼び出すためのループ応答を設定します。デコード機能はアプリケーション・プログラムの名前を変更して、COMMAREA 内の適切な入力を提供します。エンコード機能に再度出力が戻されます。詳しくは、[265 ページの『コンバーター・プログラムからの複数のアプリケーション・プログラムの呼び出し』](#)を参照してください。
- Web クライアントに送信される HTTP 応答を構成する。

CICS Web サポートのコンバーター・プログラムでは、**EXEC CICS WEB API** コマンドを使用して、Web クライアントへの応答を作成し、送信することができます。WEB WRITE HTTPHEADER は、応答の HTTP ヘッダーの作成に使用できます。WEB SEND コマンドは、応答のアセンブルと送信に使用することができます。

また、コンバーター・プログラムでは、ストレージのバッファー内に手動で HTTP 応答を構成し、それを CICS に戻して、CICS から Web クライアントに送信させることもできます。応答には HTTP バージョン、状況コード、状況テキスト、必要な HTTP ヘッダー、およびメッセージ・ボディを含める必要があります。応答の形式は、作業を行っている HTTP プロトコルの仕様 (HTTP/1.0 または HTTP/1.1) に準拠している必要があります。以下の方法で、HTTP 応答用にストレージのバッファーを取得することができます。

- GETMAIN コマンドを発行してストレージを取得する。
- 初期の処理ステージ (アナライザー・プログラムなど) で獲得したストレージを使用する。
- ユーザー作成のアプリケーション・プログラムによって戻された COMMAREA で応答を作成する。

応答に使用される領域の先頭のワードには、その領域の長さ (つまり、HTTP 応答の長さに 4 を加えたもの) が含まれていなければなりません。コンバーター・プログラムのエンコード機能を終了するときには、パラメーター・リスト内のデータ・ポインターがこのストレージのブロックを指示している必要があります。代わりに **EXEC CICS WEB API** コマンドを使用して応答を送信する場合、CICS はこのポインターが示すストレージ・ブロックを無視し、破棄します。

どちらの方法を使用して HTTP 応答を構成するとしても、CICS は通常、HTTP/1.0 または HTTP/1.1 応答に適したいくつかの HTTP ヘッダーを挿入します (それらのヘッダーは、CICS Web サポートにおける HTTP ヘッダーの解説にリストされています)。コンバーター・プログラムによって作成された応答にこれらのヘッダーが既に含まれている場合、CICS はそれらのヘッダーを置き換えません。アナライザー・プログラムが、CICS TS バージョン 3 より前との互換性処理のためのフラグを応答に立てた場合は、CICS TS バージョン 3 より前に受信していた応答と同じ応答が Web クライアントに必要であるため、CICS Transaction Server for z/OS バージョン 3 リリース 1 より前にクライアントに送信されたヘッダーのみが使用されます。このフラグ `wbra_commarea` は、コンバーター・プログラムがストレージ・ブロック内で手動で応答

を作成する場合にのみ機能します。コンバーター・プログラムが **EXEC CICS** WEB API コマンドを使用して応答を送信する場合、このフラグは無効です。

## コード・ページ変換

コンバーター・プログラムで **EXEC CICS** WEB API コマンドを使用して HTTP 要求を表示し、応答を作成する場合には、そのコマンドでの指定に従って、**EXEC CICS** WEB API コマンドを使用する他のすべてのプログラムと同じ方法でコード・ページ変換が行われます。

コンバーター・プログラムでは、32K のストレージ・ブロック内で渡される HTTP 要求のコード・ページ変換の設定を指定することはできません。コンバーター・プログラムが URIMAP 定義から直接呼び出される場合で、Web クライアント要求用のヘッダーが、メッセージ・ボディがテキストであることを示している場合、CICS は以下の標準設定を使用して、ストレージ・ブロックで提供されるメッセージ・ボディを変換します。

- 文字セットの場合、Web クライアントの要求に、CICS でサポートされている文字セットを指名する Content-Type ヘッダーがあれば、その文字セットが使用されます。Content-Type ヘッダーが Web クライアントの要求に付帯していないか、または指定した文字セットがサポートされていない場合、ISO-8859-1 文字セットが使用されます。
- ホスト・コード・ページの場合、CICS は、LOCALCCSID 初期設定パラメーターに指定されている、そのローカル CICS 領域のデフォルトのコード・ページを使用します。

これらの標準設定が適していない場合や、コード・ページ変換が不要の場合は、処理パスの中でアナライザー・プログラムを使用して、代わりのコード・ページ変換設定を指定するか、あるいは **EXEC CICS** WEB API コマンドを使用して要求を処理します。

コンバーター・プログラムが HTTP 応答をストレージのバッファ内に手動で構成する場合、CICS は 32K ストレージ・ブロックで渡された要求用に行われたコード・ページ変換をミラーリングします。応答は、アナライザー・プログラムに指定されている文字セットおよびホスト・コード・ページ設定を使用して、Web クライアントに送信されます。また、アナライザー・プログラムがない場合は、このトピックで説明している標準設定が使用されます。アナライザー・プログラムが要求に対するコード・ページ変換を抑止した場合、応答ボディのコード・ページ変換は実行されません。この結果が適切でない場合は、代わりに **EXEC CICS** WEB API コマンドを使用して、応答を作成します。

## CICS ビジネス・ロジック・インターフェース用のコンバーター・プログラム

CICS ビジネス・ロジック・インターフェースのあるコンバーター・プログラムを使用するときには制約事項があり、ユーザー作成のアプリケーション・プログラムに渡される COMMAREA の作成方法と、応答が含まれているストレージのバッファの作成方法が、その制限による影響を受ける場合があります。詳しくは、[オフセット・モードとポインター・モード](#)を参照してください。

CICS ビジネス・ロジック・インターフェース用に作成されたコンバーター・プログラムで、**EXEC CICS** WEB API コマンドを使用しないでください。

## コンバーター・プログラム・デコード機能の入力パラメーター



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

入力パラメーターは、パラメーター・リストでデコード機能に渡されます。

入力パラメーターには、以下のものがあります。

- Web クライアントの IP アドレス。
- Web クライアント要求の HTTP バージョンを指すポインター。
- 要求方式を指すポインター。
- URL のパス構成要素を指すポインター。
- 要求の HTTP ヘッダーを指すポインター。
- 要求メッセージのエンティティ・ボディを指すポインター。

- 要求のデータを提供する CICS アプリケーション・プログラムの名前 (アナライザー・プログラムによって設定されるか、URIMAP 定義で指定されています)。
- アナライザー・プログラムとコンバーター・プログラムで情報を共有するために使用される 8 バイトのユーザー・トークン。257 ページの『アナライザー・プログラムとコンバーター・プログラムでのデータの共用』を参照してください。
- 各 HTTP 要求ごとにデコード機能が起動された回数を記録する反復カウンター。このカウンターは、デコード機能を初めて呼び出す前に 1 に設定され、後続の各呼び出しを行う前に増分されます。
- エンティティ・ボディのアドレスを FREEMAIN コマンドのターゲットにすることができるかどうかの指示。

アナライザー・プログラムは、パラメーター・リストをコンバーター・プログラムに渡す前に、以上のパラメーターの値を変更することがあります。Web クライアントからの元の要求を調べたい場合は、コンバーター・プログラムで **EXEC CICS WEB API** コマンドを使用します。

## コンバーター・プログラム・デコード機能の出力パラメーター



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

デコード機能は、COMMAREA に出力として、応答コード、および COMMAREA のアドレスと長さを提供する必要があります。

- 応答コード (オプションで、理由コードによって修飾されます)。

デコード機能が URP\_OK の応答コードを戻した場合、処理は次のステップへ進みます。

デコード機能が他の値を戻した場合は、HTTP 要求がリジェクトされ、エラー応答が出ます。この場合に CICS が行う応答の詳細は、[CICS Web サポートのデフォルトの状況コードおよびエラー応答](#)を参照してください。

- ユーザー作成アプリケーション・プログラムに渡される COMMAREA のアドレスと長さ。アプリケーション・プログラムが呼び出されなかった場合は、COMMAREA は未変更のままエンコード機能に渡されます。

デコード機能は次のような出力を生成する場合もあります。

- 要求のデータを生成するユーザー作成アプリケーション・プログラムの名前。アナライザー・プログラムが名前を提供した場合、コンバーター・プログラムはその名前を変更したり、アプリケーション・プログラムを呼び出さないことを指定したりできます。
- アナライザー・プログラムとコンバーター・プログラムで情報を共有するために使用される 8 バイトのユーザー・トークン。257 ページの『アナライザー・プログラムとコンバーター・プログラムでのデータの共用』を参照してください。

## コンバーター・プログラム・エンコード機能の入力パラメーター



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

入力パラメーターは、COMMAREA のエンコード機能に渡されます。

入力パラメーターには、以下のものがあります。

- ユーザー作成アプリケーション・プログラムによって戻される COMMAREA のアドレスと長さ。アプリケーション・プログラムが呼び出されなかった場合は、COMMAREA は、デコード機能から未変更のまま渡されます。
- アナライザー・プログラムとコンバーター・プログラムで情報を共有するために使用される 8 バイトのユーザー・トークン。257 ページの『アナライザー・プログラムとコンバーター・プログラムでのデータの共用』を参照してください。
- 各 HTTP 要求ごとにエンコード機能が起動された回数を記録する反復カウンター。このカウンターは、エンコード機能を初めて呼び出す前に 1 に設定され、後続の各呼び出しを行う前に増分されます。

## コンバーター・プログラム・エンコード機能の出力パラメーター



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

エンコード機能は、出力として、応答コード、ストレージ・バッファのアドレス、HTTP 応答の長さ、および 8 バイトのユーザー・トークンを提供することができます。

- 応答コード (オプションで、理由コードによって修飾されます)。
  - エンコード機能が **URP\_OK** の応答コードを戻すと、CICS は、送信のために **EXEC CICS WEB API** コマンドがすでに使用された場合をのぞいて、提供された HTTP 応答を Web クライアントに送信します。
  - エンコード機能が **URP\_OK\_LOOP** の応答コードを戻した場合、デコード機能による処理へ進みます。詳細については、[265 ページの『コンバーター・プログラムからの複数のアプリケーション・プログラムの呼び出し』](#)を参照してください。
  - エンコード機能が他の値を戻した場合は、HTTP 要求がリジェクトされ、エラー応答が出ます。この場合に CICS が行う応答の詳細は、[CICS Web サポートのデフォルトの状況コードおよびエラー応答](#)を参照してください。
- ストレージのバッファに手動で HTTP 応答を作成した場合は、ストレージのバッファのアドレス、および HTTP 応答の長さ。このバッファの先頭のワードには、データの長さ (つまり、HTTP 応答の長さに 4 を加えたもの) が含まれていなければなりません。代わりに **EXEC CICS WEB API** コマンドを使用して応答を送信する場合、CICS はこのポインターが示すストレージ・ブロックを無視し、破棄します。
- アナライザー・プログラムとコンバーター・プログラムで情報を共用するために使用される 8 バイトのユーザー・トークン。[257 ページの『アナライザー・プログラムとコンバーター・プログラムでのデータの共用』](#)を参照してください。

## コンバーター・プログラムからの複数のアプリケーション・プログラムの呼び出し

HTTP 要求に対する応答を作成するために必要なデータは、場合によっては、複数のユーザー作成アプリケーション・プログラムから出力されることがあります。

### このタスクについて



**重要:** このトピックには、プロダクト・センシティブ・プログラミング・インターフェースと関連ガイダンス情報が含まれています。

このような場合には、必要に応じて以下の順序を繰り返すことができます。

- コンバーターのデコード機能。
- アプリケーション・プログラム。
- コンバーターのエンコード機能。

これを行うには、エンコード機能で応答を **URP\_OK\_LOOP** に設定します。HTTP 応答が完成したら、応答を **URP\_OK** に設定します。

デコード機能が 2 番目およびそれ以降に呼び出されるときは、以下の入力パラメーターは使用できません。

- HTTP バージョン。
- メソッド。
- URL のパス構成要素。
- 要求ヘッダー。
- エンティティ・ボディ。

ただし、**WEB EXTRACT** コマンドを使用すると、同じ情報を取得することができます。

パラメーター・リストのデータ・ポインターとユーザー・トークンを使用して、デコード機能とエンコード機能でデータを共用します。ストレージのバッファに HTTP 応答を手動で作成する際に、エンコード機能の最終呼び出しを行ったら、データ・ポインター (**encode\_data\_ptr**) が有効な HTTP 応答のアドレスを指し示していることを確認する必要があります。 **EXEC CICS WEB API** コマンドを使用して応答の生成

および送信を行う場合は、このステージで確認を行います。この場合、CICS は、このポインターが示すストレージ・ブロックを無視し、破棄します。



## 第 4 章 統計収集分析プログラムの作成

CICS 統計の収集と分析は、独自の統計収集分析プログラムを記述することによってカスタマイズできます。

統計収集分析プログラムを記述するときは、1 次スペース変換モードで CICS が常に制御を受け取るようにしなければなりません。すべてのアクセス・レジスターの元の内容は復元する必要があり、すべての汎用レジスターは復元する必要があります。ただし、戻りコードまたはリンケージ情報を提供するレジスターは例外です。変換モードについては、[z/Architecture 解説書](#)を参照してください。

### CICS 統計を収集するプログラムの作成

#### CICS 統計を収集する理由

CICS 統計には、CICS システム全体の情報が含まれます。例えば、そのパフォーマンスやリソースの使用などです。そのため、統計データは、パフォーマンスを調整する場合にも、キャパシティー・プランニングを行う場合にも役立ちます。

統計は、後のオフライン分析用として、CICS オンライン処理中に収集されます。統計ドメインは、統計レコードをシステム管理機能 (SMF) データ・セットに書き込みます。レコードは SMF のタイプ 110、サブタイプ 0002 から構成されます。

**注:** モニター・レコードと、一時記憶域共用キュー・サーバーによって生成される統計レコードも、タイプ 110 のレコードとして SMF データ・セットに書き込まれます。(いくつかのジャーナリング・タイプ 110 のレコードも、そこに書き込まれます。) このデータ・セットは、統計レコードおよびモニター・レコードと一緒に処理する場合に役立ちます。これは、統計は、CICS モニターが作成したトランザクション・データを補完するリソース情報およびシステム情報を提供するためです。

統計レコードは、以下によっても書き込まれます。

- ・一時記憶域 (TS) データ共用プール・サーバー領域。これらのレコードは SMF のタイプ 110、サブタイプ 0003 から構成されます。
- ・カップリング・ファシリティ・データ・テーブル (CFDT) サーバー領域。これらのレコードは SMF のタイプ 110、サブタイプ 0004 から構成されます。
- ・名前付きカウンター・シーケンス番号サーバー領域。これらのレコードは SMF のタイプ 110、サブタイプ 0005 から構成されます。

CICS は、5 つのタイプの統計 (間隔、1 日の終わり、要求、要求されたリセット、および非送信請求) を作成します。

#### 重要

CICS 統計のタイプ、収集のタイミング、およびその収集の制御方法については、[CICS 統計の概要](#)を参照してください。

#### 統計カウンターのリセット・オプション

統計カウンターは以下の状況でリセットされます。

- ・ CICS の始動時
- ・ 間隔統計の書き込み時 (ただし、間隔の開始時に統計が書き込まれていない場合は除く)
- ・ 1 日の終わり
- ・ 要求されたリセット統計の書き込み時

ただし、SMF データ・セットにレコードを書き込まずに、統計カウンターをリセットすることができます。以下のいずれかのオプションを使用して、統計記録状況を変更します。

- ・ CICS Explorer の「領域」操作ビューの「時間をリセット」オプション

- **CEMT SET STATISTICS ON|OFF RESETNOW** コマンド
- **EXEC CICS SET STATISTICS ON|OFF RESETNOW** コマンド

上記の方法で、ユーザー・プログラムはすべての統計カウンタをリセットすることができます。

記録状況を本当に変更する場合にのみ、RESETNOW オプションを指定することができます。例えば、STATISTICS が既に ON に設定されている状態で、**EXEC CICS SET STATISTICS ON RESETNOW** をコーディングすると、エラー応答が返されます。

## 重要

統計カウンタをリセットする方法は複数あります。特定のカウンタについては、以下の値にリセットすることができます。

- 0
- 1
- 新規ピーク値
- リセットされない
- 上記以外

特定の統計カウンタのリセット方法については、[統計カウンタのリセット特性](#)を参照してください。

## CICS 統計の収集と抽出

CICS によって収集される統計は、SMF データ・セットに書き込まれます。ただし、ユーザー・プログラムは CICS Explorer の「領域」操作ビュー、**EXEC CICS COLLECT STATISTICS** コマンド、および **EXEC CICS EXTRACT STATISTICS** コマンドを使用して、特定のリソースに関する現行の統計、または特定のタイプのリソースに関する全統計を収集することもできます。

### このタスクについて

収集できる統計の種類として、CICS 領域内のグローバル・トランザクション・アクティビティーに関する統計(接続しているトランザクションの総数など)があります。関心のある単一トランザクションを指定することもできます。統計は、開始アプリケーションに返されます。これらのコマンドのプログラミング情報については、[COLLECT STATISTICS](#) を参照してください。

## 手順

統計分析を実行します。CICS は、**EXEC CICS COLLECT STATISTICS**、**EXEC CICS EXTRACT STATISTICS**、および **EXEC CICS INQUIRE** コマンドを使用して CICS システムの役立つ分析を生成する方法を示す 15 個のサンプル・プログラムを提供します。

以下のプログラムが提供されます。

- DFHOSTAT
- DFHOSTDB
- DFHOSTEJ
- DFHOSTEP
- DFHOSTGN
- DFHOSTLK
- DFHOSTPR
- DFHOSTSA
- DFHOSTSY
- DFHOSTTP
- DFHOSTTS
- DFHOSTWB

- DFH\$STAS
- DFH\$STCN
- DFH\$STTB

サンプル・プログラムは、CICS ディスパッチャーから重要なシステム・パラメーターを示すレポートを、ローダー統計および CICS ストレージ・マネージャーの分析と共に生成します。DFH\$STAS、DFH\$STCN、および DFH\$STTB は、アセンブラー言語で提供されます。その他の 12 個のプログラムは COBOL で提供されます。

サンプル統計プログラムのインストールと操作、およびプログラムによって生成されるデータについて詳しくは、[CICS 統計の概要](#)を参照してください。

## CICS 統計レコードの形式

このセクションでは、CICS 統計 SMF タイプ 110 レコードの形式について詳細に説明します。この情報は、統計データを分析するための独自のプログラムを作成する場合に必要になります。CICS 統計レコードには、SMF ヘッダー、SMF プロダクト・セクション、および CICS データ・セクションの 3 つの構成要素が含まれています。

269 ページの図 62 は、CICS 統計レコードの構成要素を示しています。以下のセクションで、それらを 1 つずつ説明します。

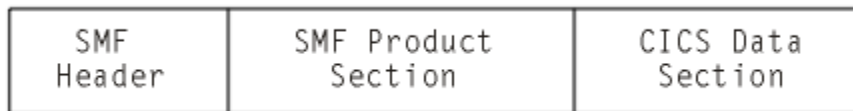


図 62. SMF タイプ 110 統計レコードの形式

### SMF ヘッダーおよび SMF 製品セクション

SMF ヘッダーは、出力を作成しているシステムを説明します。SMF 製品セクションは、統計データが関連付けられるサブシステムを識別します。CICS 統計の場合、これは CICS 領域、TS データ共用サーバー、CFDT サーバー、または名前付きカウンター・シーケンス番号サーバーです。

SMF ヘッダーと SMF 製品セクションはどちらも、DSECT STSMFDS でマップできます。これは、次のような DFHSTSMF マクロを使用して生成できます。

```
STSMFDS DFHSTSMF PREFIX=SMF
```

ラベル「STSMFDS」はデフォルトの DSECT 名で、SMF はデフォルトの PREFIX 値です。したがって、DFHSTSMF をコーディングして DSECT を生成することもできます。

STSMFDS DSECT のフォーマットを [270 ページの図 63](#) に示します。

```

*          START THE SMF HEADER
*
STSMFDS  DSECT
SMFSTLEN DS      XL2          RECORD LENGTH
SMFSTSEQ DS      XL2          SEGMENT DESCRIPTOR
SMFSTFLG DS      X            OPERATING SYSTEM INDICATOR (see note 1)
SMFSTRTY DC      X'6E'        RECORD TYPE 110 FOR CICS
SMFSTTME DS      XL4          TIME RECORD MOVED TO SMF
SMFSTDTE DS      XL4          DATE RECORD MOVED TO SMF
SMFSTSID DS      XL4          SYSTEM IDENTIFICATION
SMFSTSSI DS      CL4'CICS'    SUBSYSTEM IDENTIFICATION
SMFSTSTY DS      XL2          RECORD SUBTYPE X'0002' FOR STATISTICS
*                                     (see note 4)
SMFSTRRN DS      XL2          NUMBER OF TRIPLETS
DS      XL2          RESERVED
SMFSTAPS DS      XL4          OFFSET TO PRODUCT SECTION
SMFSTLPS DS      XL2          LENGTH OF PRODUCT SECTION
SMFSTNPS DS      XL2          NUMBER OF PRODUCT SECTIONS
SMFSTASS DS      XL4          OFFSET TO DATA SECTION
SMFSTASL DS      XL2          LENGTH OF DATA SECTION
SMFSTASN DS      XL2          NUMBER OF DATA SECTIONS
*
*          THIS CONCLUDES THE SMF HEADER
*
*          START THE SMF PRODUCT SECTION
*
SMFSTRVN DS      XL2          RECORD VERSION
SMFSTPRN DS      CL8          PRODUCT NAME (GENERIC APPLID)
SMFSTSPN DS      CL8          PRODUCT NAME (SPECIFIC APPLID)
SMFSTMFL DS      XL2          RECORD MAINTENANCE INDICATOR
DS      XL2          RESERVED
DS      XL2          RESERVED
SMFSTDTK DS      XL4          DOMAIN TOKEN
SMFSTDID DS      CL2          DOMAIN ID
SMFSTRQT DS      CL3          USS/EOD/REQ/INT STATISTICS TYPE
SMFSTICD DS      CL3          YES IF INCOMPLETE DATA RECORDED
SMFSTDAT DS      CL8          COLLECTION DATE MMDDYYYY
SMFSTCLT DS      CL6          COLLECTION TIME HHMMSS
SMFSTINT DS      CL6          INTERVAL HHMMSS. See note 3.
SMFSTINO DS      XL4          INTERVAL NUMBER. See note 3.
SMFSTRTK DS      XL8          REQUEST TOKEN
SMFSTLRT DS      CL6          LAST RESET TIME HHMMSS
SMFSTCST DS      XL8          CICS START TIME
SMFSTJBN DS      CL8          JOBNAME
SMFSTRSD DS      XL4          JOB DATE
SMFSTRST DS      XL4          JOB TIME
SMFSTUIF DS      CL8          USER IDENTIFICATION
SMFSTPDN DS      CL8          OPERATING SYSTEM PRODUCT LEVEL
*
*          THIS CONCLUDES THE SMF PRODUCT SECTION

```

図 63. 統計レコードの SMF ヘッダーと製品セクションのフォーマット

注:

1. CICS は、SMF ヘッダー (SMFSTFLG) にオペレーティング・システム 標識フラグ・バイトのサブシステム関連ビットのみを設定します。SMF は、オペレーティング・システムのレベルおよびその他の要因に従って、残りのバイトを設定します。他のビットの設定に関する説明については、[z/OS MVS システム管理機能 \(SMF\)](#)を参照してください。
2. コピーブック DFHSMFDS も提供されており、これを使用して、CICS ジャーナリング、CICS モニター、および CICS 統計によって作成された、SMF 110 レコードの 6 つのサブタイプのすべての SMF ヘッダーおよび SMF 製品セクションをマップできます。
3. フィールド SMFSTINT および SMFSTINO は、SMFSTRQT が 'INT' の場合にのみ関係します。それ以外の場合は、どちらの値も無視してください。
4. TS データ共用の場合、レコードのサブタイプは X'0003' であり、フィールドによっては設定されないものや、別の方法で使用されるものがあります。SMFSTPRN と SMFSTSPN には、サーバー接頭部 (DFHXQ) およびプール名が含まれます。
5. カップリング・ファシリティ・データ・テーブル (CFDT) サーバーの場合、レコードのサブタイプは X'0004' であり、フィールドによっては設定されないものや、別の方法で使用されるものがあります。SMFSTPRN および SMFSTSPN には、サーバー接頭部 (DFHCF) とカップリング・ファシリティ・データ・テーブルのプール名が含まれます。

6. 名前付きカウンター・シーケンス番号サーバーの場合、レコードのサブタイプは X'0005' であり、フィールドによっては設定されないものや、別の方法で 사용되는ものがあります。SMFSTPRN と SMFSTSPN には、サーバー接頭部 (DFHNC) とプール名が含まれます。

### CICS 統計データ・セクション

CICS 統計レコードの統計データ・セクションは、複数の統計レコードで構成されています。

CICS 統計データ・セクションの形式については、[271 ページの図 64](#) に示されています。

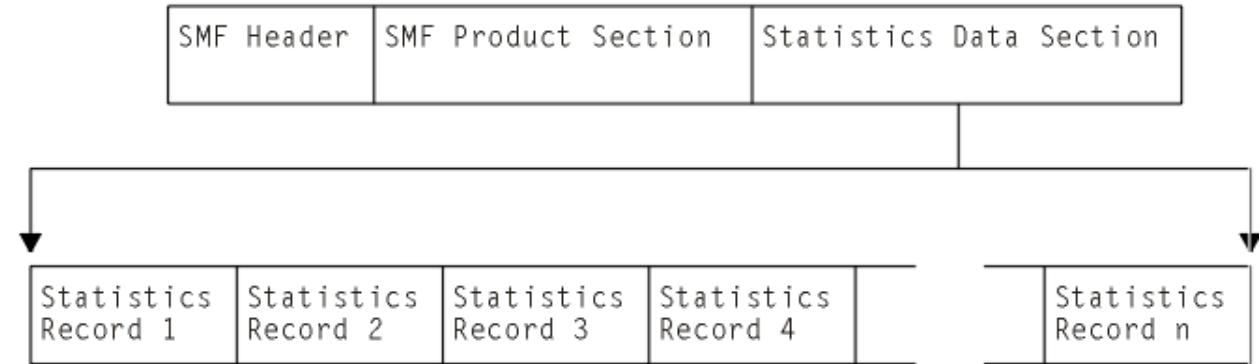


図 64. 統計データ・セクションの形式

データ・レコードが不完全な場合、フラグ・フィールド SMFSTICD は YES に設定されます。この場合、統計データ・セクションは表示されません。

完全なデータ・レコードの場合、統計データ・セクションは 1 つ以上の統計データ・レコードで構成されます。データ・レコードにはさまざまな形式があります。各データ・レコードの最初の 5 バイトの形式は共通しています。これらの 5 バイトは、[271 ページの図 65](#) にあるサンプル集 DFHSTIDS からの抜粋で説明されています。

DFHSTIDS	DSECT		Statistics record header
*			
	DS	0F	Fullword alignment
STILEN	DS	H	Length of the record
STID	DS	AL2	Statistics identifier
STIVERS	DS	CL1	Statistics record version

図 65. コピーブック DFHSTIDS からの抜粋

#### STILEN

データ・レコードの長さ。

#### STID

所有する統計レコードのタイプを識別する名前または値 ([271 ページの表 21](#) を参照)。

STID シンボル名または値を使用して、統計データ・レコードを処理するときに使用するサンプル集を決定することができます。STID 名または STID 値とサンプル集との関係について詳しくは、[271 ページの表 21](#) を参照してください。統計データ・レコード内のフィールドについての指針は、[DSECTS](#) および [DFHSTUP](#) レポートの [CICS 統計](#) を参照してください。

#### STIVERS

レコードのバージョン。STIVERS は、このリリースの CICS では値 1 を取ります。

### STID 名と STID 値に関連した統計データ・レコード・サンプル集

表 21. STID 名と STID 値に関連した統計データ・レコード・サンプル集			
STID シンボル名	STID 値	コピーブック	レコードのタイプ
STIXMG	10	DFHXMGDS	Transaction manager (Globals) id



表 21. STID 名と STID 値に関連した統計データ・レコード・サンプル集 (続き)

STID シンボル 名	STID 値	コピーブック	レコードのタイプ
STIXMR	11	DFHXRDS	Transaction manager (Trans) id
STIXMC	12	DFHXMCDs	Transaction manager (Tclass) id
STIFEPI	16	DFHA22DS	FEPI pool id
STIFEPI	17	DFHA23DS	FEPI connection id
STIFEPI	18	DFHA24DS	FEPI target id
STISMD	19	DFHSMDDs	Storage mgr domain subpool id
STISMT	20	DFHSMTDS	Storage manager task subpool id
STIVT	21	DFHA03DS	z/OS Communications Server stats id
STIPAU	23	DFHPGGDS	Program Autoinstall id
STIAU	24	DFHA04DS	Terminal Autoinstall stats id
STILDR	25	DFHLDRDS	Public Loader (Resid) id
STIDBU	28	DFHDBUDs	DBCTL USS id
STISMDS	29	DFHMSDS	Storage manager DSA id
STILDG	30	DFHLDGDS	Loader (Globals) id
STILDB	31	DFHLDBDS	LIBRARY resources - public
STILDY	32	DFHLDYDS	LIBRARY resources - private
STITCR	34	DFHA06DS	Terminal control (resid) id
STILDP	36	DFHLDPDS	Private Loader (Resid) id
STILSR	39	DFHA08DS	LSRPOOL pool stats (resid) id
STILSRFR	40	DFHA09DS	LSRPOOL File statistics (by file)
STITDQR	42	DFHTQRDS	TDQUEUE (Resid) id
STITDQG	45	DFHTQGDS	TDQUEUE (globals) id
STITSQ	48	DFHTSGDS	TSQUEUE statistics id
STICONSR	52	DFHA14DS	ISC/IRC system entry (resid) id
STICONSS	54	DFHA21DS	ISC connection - system security
STIUSG	61	DFHUSGDS	User domain stats id
STIDS	62	DFHDSGDS	Dispatcher stats id
STITM	63	DFHA16DS	Table manager statistics id
STIDST	64	DFHDSTDS	Dispatcher TCB (global)id
STIDSR	65	DFHDSRDS	Dispatcher TCB (resid)id
STIST	66	DFHSTGDS	Statistics statistics id
STIFCR	67	DFHA17DS	File Control (resid) id
STIMQG	74	DFHMQGDS	MQ connection stats (global) id
STICONMR	76	DFHA20DS	ISC/IRC mode entry (resid) id

表 21. STID 名と STID 値に関連した統計データ・レコード・サンプル集 (続き)

STID シンボル 名	STID 値	コピーブック	レコードのタイプ
STIM	81	DFHMGDS	Monitoring stats (global) id
STIMNR	84	DFHMNTDS	Monitoring stats (Resid) id
STITDR	85	DFHTDRDS	Transaction dump (resid) id
STITDG	87	DFHTDGDS	Transaction dump (global) id
STISDR	88	DFHSDRDS	System dump (resid) id
STISDG	90	DFHSDGDS	System dump (global) id
STILGG	92	DFHLGGDS	Logstream stats (global) id
STILGR	93	DFHLGRDS	Logger stats (resid) id
STILGS	94	DFHLGSDS	Logstream stats (resid) id
STINQG	97	DFHNQGDS	Enqueue mgr stats (global) id
STIRMG	99	DFHRMGDS	Recovery mgr stats (global) id
STIRLR	100	DFHRLRDS	BUNDLES (resource) id
STIWBG	101	DFHWBGDS	URIMAPs (global) id
STID2G	102	DFHD2GDS	DB2® connection stats (global) id
STID2R	103	DFHD2RDS	DB2 entry stats (resource) id
STIWBR	104	DFHWBRDS	URIMAPs (resource) id
STIPIR	105	DFHPIRDS	PIPELINE (resource) id
STIPIW	106	DFHPIWDS	WEBSERVICE (resource) id
STISOG	107	DFHSOGDS	TCP/IP (global) id
STISOR	108	DFHSORDS	TCPIP services (resource) id
STIISR	109	DFHISRDS	IPCONN (resource) id
STIW2R	110	DFHW2RDS	ATOMSERVICE (resource) id
STIDHD	112	DFHDHDDS	DOCTEMPLATE (resource) id
STIMLR	113	DFHMLRDS	XMLTRANSFORM (resource) id
STISJS	116	DFHSJSDS	JVMSERVER stats (resource) id
STIPGR	119	DFHPGRDS	JVMPROGRAM stats (resource) id
STIPGD	120	DFHPGDDS	PROGRAMDEF stats (resource) id
STIECG	140	DFHECGDS	EVENTBINDINGS (global) id
STIECR	141	DFHECRDS	EVENTBINDINGS (resource) id
STIEPG	142	DFHEPGDS	EVENTPROCESS (global) id
STIECC	143	DFHECCDS	CAPTURESPECS (resource) id
STIEPR	144	DFHEPRDS	EPADAPTERs (resource) id
STIMPR	145	DFHMPRDS	POLICYs (Resource) id

表 21. STID 名と STID 値に関連した統計データ・レコード・サンプル集 (続き)

STID シンボル名	STID 値	コピーブック	レコードのタイプ
STIPGP	146	DFHPGPDS	JVM programs - private
STIPGE	147	DFHPGEDS	Program definitions - private
STIMQR	148	DFHMQRDS	MQMONITORs (Resource) id
STIASG	149	DFHASGDS	ASYNCSERVICE (Global) id
STISJN	150	DFHSJNDS	NODEJSAPP (Resource) id

**STID に関連した TS データ共用統計**

TS データ共用統計はシンボル名を使用しませんが、以下のように STID 値に関連しています。

表 22. STID に関連した TS データ共用統計

STID シンボル名	STID 値	コピーブック	レコードのタイプ
-	121	DFHXQS1D	TS server list structure stats id
-	122	DFHXQS2D	TS buffer stats id
-	123	DFHXQS3D	TS storage stats id

**STID に関連したカップリング・ファシリティ・データ・テーブルのサーバー統計**

カップリング・ファシリティ・データ・テーブルのサーバー統計はシンボル名を使用しませんが、以下のように STID 値に関連しています。

表 23. STID に関連したカップリング・ファシリティ・データ・テーブルのサーバー統計

STID シンボル名	STID 値	コピーブック	レコードのタイプ
-	126	DFHCFS6D	CFDT server list stats
-	127	DFHCFS7D	CFDT buffer stats id
-	128	DFHCFS8D	CFDT request stats id
-	129	DFHCFS9D	CFDT storage stats id

**STID に関連した名前付きシーケンスのサーバー統計**

名前付きシーケンス番号のサーバー統計はシンボル名を使用しませんが、以下のように STID 値に関連しています。

表 24. STID に関連した名前付きシーケンスのサーバー統計

STID シンボル名	STID 値	コピーブック	レコードのタイプ
-	124	DFHNCS4D	NC server list structure stats id
-	125	DFHNCS5D	NC server storage stats id

# XSTOUT グローバル・ユーザー出口プログラムを使用した統計レコードのフィルタリング

## このタスクについて

CICS 統計ドメインには 1 つのグローバル・ユーザー出口 (XSTOUT) があります。この出口は、統計データ・バッファの内容が SMF に書き込まれる前に開始します。この出口では、以下の情報を使用できます。

- 統計バッファのアドレス
- 統計バッファの長さ
- 統計タイプのアドレス

上記の内容は、統計の 5 つのタイプ (間隔統計、1 日の終わり統計、要求された統計、要求されたりセット統計、非送信請求統計) のすべてに適用されます。

グローバル・ユーザー出口プログラムをこの出口で開始するように作成する場合は、この情報を調べて、バッファの内容を SMF に書き込むか、または出力を抑制するかを CICS に指示することができます。

グローバル・ユーザー出口についての全般的な情報、および特に統計出口について調べる場合は、[1 ページの『グローバル・ユーザー出口プログラム』](#)を参照してください。

## CICS 統計の出力の処理

統計出力の処理にはいくつかのオプションがあります。

次のものを使用することができます。

### CICS 提供の DFHSTUP プログラム

DFHSTUP を実行する方法については、[統計ユーティリティ・プログラム \(DFHSTUP\)](#)を参照してください。DFHSTUP によって作成されるレポートを解釈する方法については、[DSECTS および DFHSTUP レポートの CICS 統計](#)を参照してください。

### IBM Z® Decision Support

CICS 統計 (および他のデータ) を Db2 データ・セットに保管し、そのデータをいろいろな形式にして提示することができます。IBM Z Decision Support については、[「Improving performance」の『IBM Z Decision Support』](#)を参照してください。

### 独自のプログラム

統計レコードのデータをレポートおよび分析します。CICS 統計レコードのフォーマットについては、[269 ページの『CICS 統計レコードの形式』](#)に記載してあります。

## CICS TS フォーマットのジャーナル・レコードの構造および内容

注:

**SMF レコード**については、[307 ページの『SMF に書き込まれたジャーナル・レコードの形式』](#)を参照してください。

このセクションは SMF データ・セットに書き込まれるジャーナル・レコードには適用されません。

システム・ログは常に CICS TS フォーマットで表示されます。各一般ログは、ジャーナル・データの連続ブロックのストリームで構成されます。各ブロックは、ブロック・ヘッダーとそれに続く可変数の CICS ジャーナル・レコードで構成されます。各 CICS ジャーナル・レコードは、レコード・ヘッダーとそれに続く呼び出し元データで構成されます。

276 ページの図 66 は、一般ログを大まかに表した図です。ブロック全体のフォーマットとジャーナル・レコード全体のフォーマットを示しています。

呼び出し元データのフォーマットは、ジャーナル・レコードを発行する CICS コンポーネントと、その時にジャーナリングされる機能によって異なります。つまり、例えばファイル制御によって発行されたジャーナル・レコードの呼び出し元データのフォーマットは、FEPI によって発行されたジャーナル・レコードの呼び出し元データのフォーマットとは異なります。

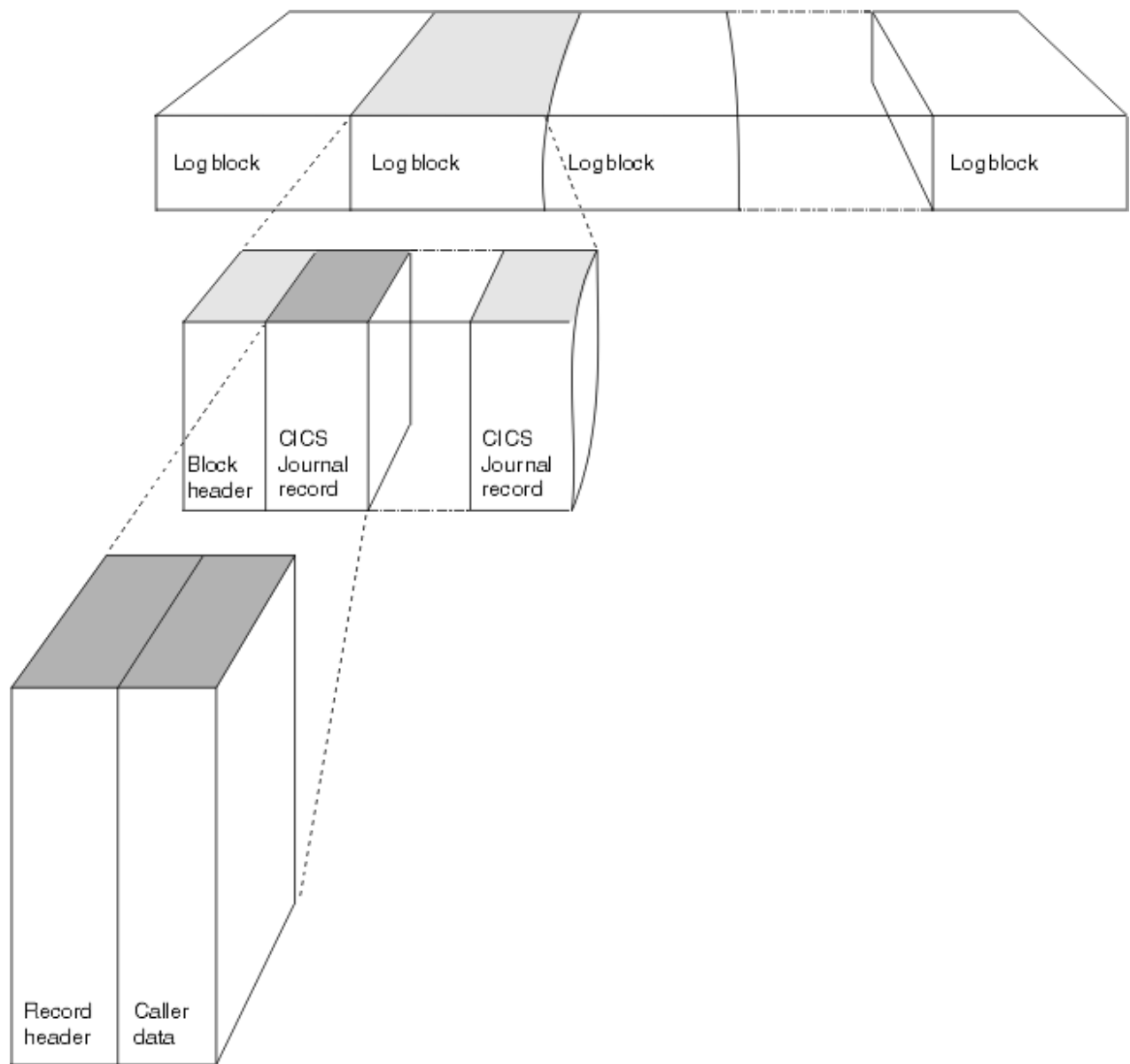


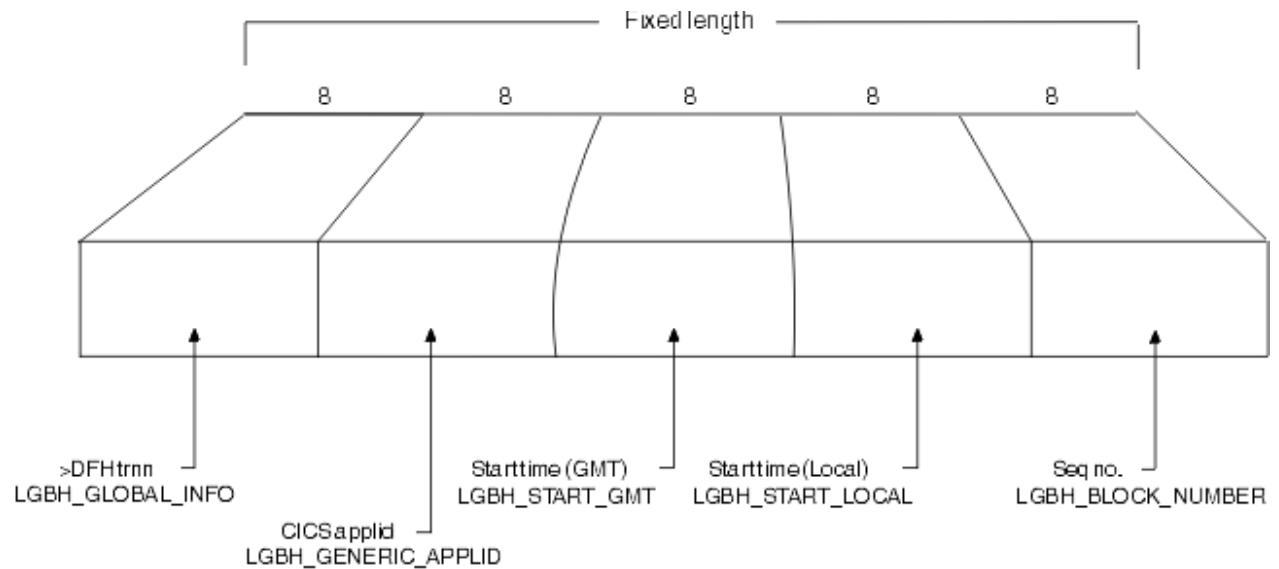
図 66. 一般ログのレイアウト

### 汎用ログ・ブロック・ヘッダー

ログ・ブロック・ヘッダーには、ジャーナル・ブロックを書き込む CICS applid など、汎用のシステム全体の性質に関する情報が含まれます。

277 ページの図 67 に、ログ・ブロック・ヘッダーのフォーマットを示します。





Where  $t$  = Log type  
 $r$  = reserved  
 $nn$  = block version number

図 67. 汎用ログ・ブロック・ヘッダーのフォーマット

#### LGBH\_GLOBAL\_INFO

'>DFHtrnn' を含む 8 バイト。ここで、各文字は以下を示します。

```
t = log type
r = reserved
nn = block version number
```

#### LGBH\_GENERIC\_APPLID

8 バイトの CICS applid。

#### LGBH\_START\_GMT

8 バイトの開始時刻 (GMT)。

#### LGBH\_START\_LOCAL

8 バイトの開始時刻 (ローカル)。

#### LGBH\_BLOCK\_NUMBER

8 バイトのシーケンス番号。

### 汎用ログ・ジャーナル・レコード

ジャーナル・レコードは、レコード・ヘッダーと、それに続く呼び出し側のデータで構成されます。レコード・ヘッダーには、書き込まれた時刻などの、レコードの属性を説明する情報が含まれます。呼び出し側のデータは、レコードを発行する CICS コンポーネント、およびジャーナル処理される機能によって異なります。

278 ページの図 68 に、レコード・ヘッダーのフォーマットを示します。

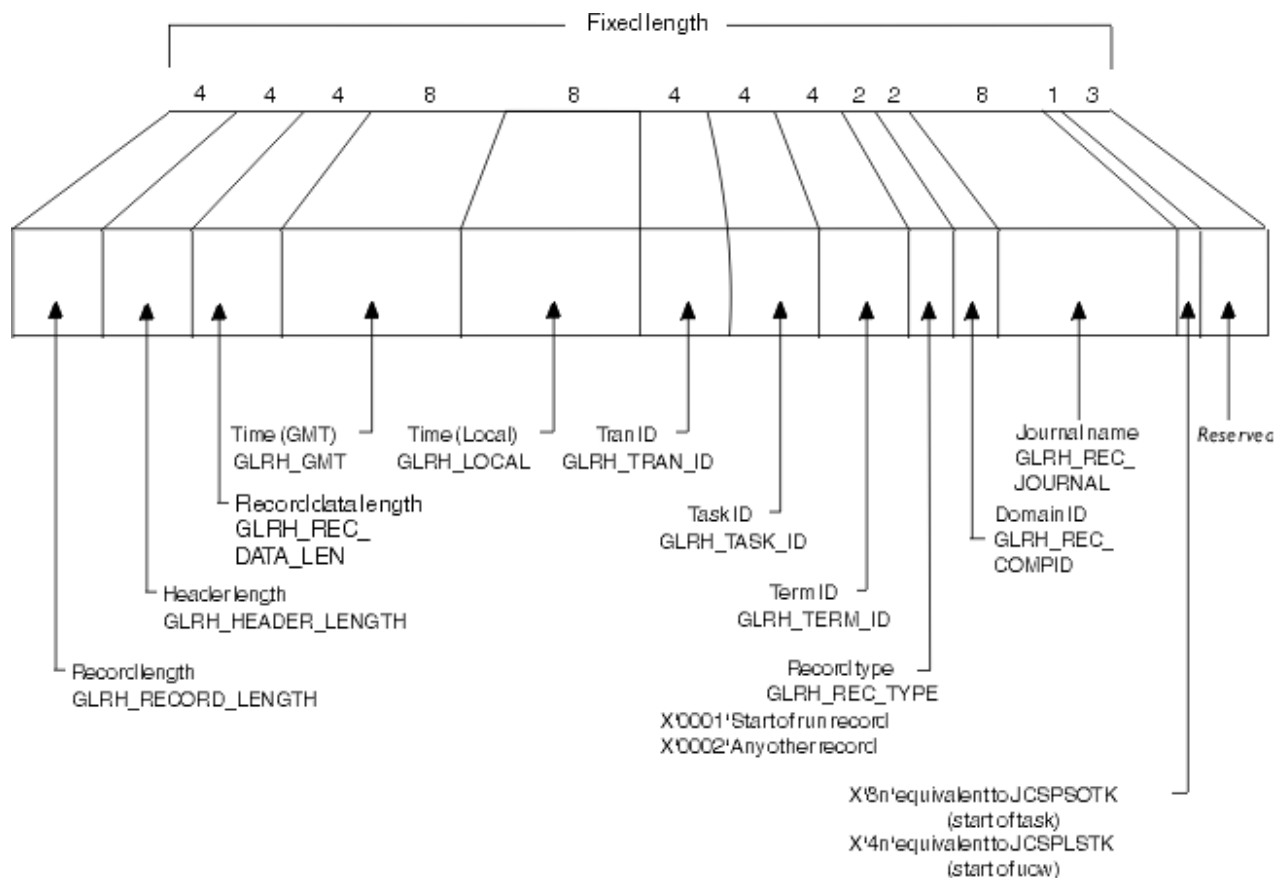


図 68. 汎用ログ・レコード・ヘッダーのフォーマット

#### GLRH\_RECORD\_LENGTH

4 バイトのレコード長。

#### GLRH\_HEADER\_LENGTH

4 バイトのヘッダー長。

#### GLRH\_REC\_DATA\_LEN

4 バイトのレコード・データ長。

#### GLRH\_GMT

8 バイトの時刻 (GMT)。

#### GLRH\_LOCAL

8 バイトの時刻 (ローカル)。

#### GLRH\_TRAN\_ID

4 バイトのトランザクション ID。

#### GLRH\_TASK\_ID

4 バイトのタスク ID。

#### GLRH\_TERM\_ID

4 バイトの端末 ID。

#### GLRH\_REC\_TYPE

2 バイトのレコード・タイプ。以下のいずれかの方法で行います。

```
X'0001' Start of run record
X'0002' Any other record
```

#### GLRH\_REC\_COMPID

2 バイトのドメイン ID。

#### GLRH\_REC\_JOURNAL

8 バイトのジャーナル名。

## タスク標識の開始

次のものを含めることができる、1 バイト。

X'8n'	Equivalent to JCSPS0TK (start of task)
X'4n'	Equivalent to JCSPSTK (start of UOW)

## 予約済み

3 バイトの予約フィールド。

## 呼び出し元データ

呼び出し元データはレコード・ヘッダーに続く部分です。一般ログ・ジャーナル・レコードの呼び出し元データ部分のフォーマットは、レコードを書き込む CICS コンポーネントと、ジャーナリングされる機能によって異なります。

ジャーナル・レコードを書き込む CICS コンポーネントは、ロガー、ジャーナル管理 (ユーザーから発行された要求の場合)、ファイル制御、フロントエンド・プログラミング・インターフェース (FEPI)、端末制御です。レコード・ヘッダーのフィールド GLRH\_REC\_COMPID から、レコードを書き込んだコンポーネントがわかります (それぞれ LG、UJ、FC、SZ、TC)。

ファイル制御の場合は、実際のジャーナル・データの先頭に情報が追加されます (280 ページの『ファイル制御により書き込まれた呼び出し元データ』を参照)。他のコンポーネント (ジャーナル管理、FEPI、端末制御) はジャーナル・データにその他の情報を追加しません。

## API ユーザー・ヘッダー

レコードが CICS API によって書き込まれた場合は、呼び出し元データのセクションは API ユーザー・ヘッダーで始まります。このフォーマットは 279 ページの図 69 に示されています。

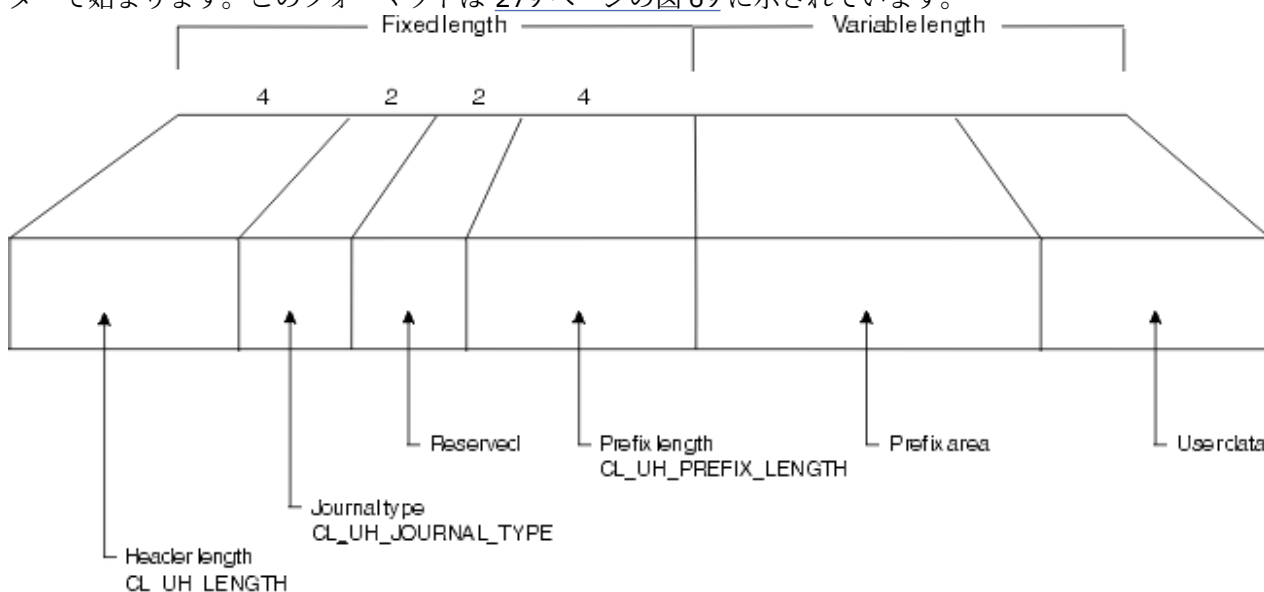


図 69. API ユーザー・ヘッダーのフォーマット

### CL\_UH\_LENGTH

4 バイトのヘッダー長。

### CL\_UH\_JOURNAL\_TYPE

2 バイトのジャーナル・タイプ

## 予約済み

2 バイトの予約フィールド。

### CL\_UH\_PREFIX\_LENGTH

接頭部の長さは 4 バイトです。

## 接頭部

接頭部域の長さは可変長です。

## ユーザー・データ

ユーザー・データの長さは可変長です。

### ファイル制御により書き込まれた呼び出し元データ

ファイルのログおよびジャーナル・ブロック (FLJB) は、ファイル制御がそのジャーナル・レコードの一部として書き込む呼び出し元データを記述したものです。コピーブック DFHFCLGD は FLJB DSECT を定義します。

FLJB には 2 つのセクションがあります。最初のセクションにはファイル制御により書き込まれたすべてのジャーナル・レコードに該当するデータが含まれています。2 番目のセクションにはレコード・タイプに固有の情報が含まれています。いずれのセクションも固定長です。

一部のレコードには可変長の 3 番目および 4 番目のセクションもあります。

280 ページの表 25 は、ファイル制御により書き込まれるジャーナル・レコードの各セクションの概要を示しています。

表 25. ファイル制御により書き込まれたジャーナル・レコードの FLJB セクション				
レコード・タイプ	最初のセクション	2 番目のセクション	3 番目のセクション	4 番目のセクション
<a href="#">281 ページ</a> の『読み取り専用、読み取り更新、書き込み更新、書き込み追加、書き込み追加の完了の各レコード・タイプ』	FLJB_GENERAL_DATA	FLJB_COMMON_DATA	FLJB_CD_KEY	FLJB_CD_DATA
<a href="#">284 ページ</a> の『書き込み削除レコード・タイプ』	FLJB_GENERAL_DATA	FLJB_WRITE_DELETE_DATA	FLJB_WDD_BASE_KEY	FLJB_WDD_PATH_KEY
<a href="#">286 ページ</a> の『コミットおよびバックアウトのレコード・タイプ』	FLJB_GENERAL_DATA	なし	なし	なし
<a href="#">286 ページ</a> の『Unlock record types』	FLJB_GENERAL_DATA	FLJB_UNLOCK_DATA	FLJB_UND_BASE_KEY	FLJB_UND_PATH_KEY
<a href="#">288 ページ</a> の『ファイル・クローズ・レコード・タイプ』	FLJB_GENERAL_DATA	FLJB_FILE_CLOSE_DATA	なし	なし
<a href="#">289 ページ</a> の『タイアップ・レコード・タイプ』	FLJB_GENERAL_DATA	FLJB_TIE_UP_RECORD_DATA	なし	なし

**読み取り専用、読み取り更新、書き込み更新、書き込み追加、書き込み追加の完了の各レコード・タイプ**  
読み取り専用、読み取り更新、書き込み更新、書き込み追加、および書き込み追加完了の各レコード・タイプ用に書き込まれるジャーナル・レコードは、4つのセクションで構成されます。

これらのセクションは以下のとおりです。

- FLJB\_GENERAL\_DATA セクション
  - FLJB\_COMMON\_DATA セクション
  - 2つの呼び出し元データ・イメージ・セクション。
    - FLJB\_CD\_KEY (その長さは FLJB\_COMMON\_DATA で与えられる)
    - FLJB\_CD\_DATA (その長さは FLJB\_COMMON\_DATA で与えられる)
- このセクションには、呼び出し元データのイメージが含まれます。

## レコード形式

これらのレコード・タイプ用に書き込まれるレコードのフォーマットを、[281 ページの図 70](#) に示します。

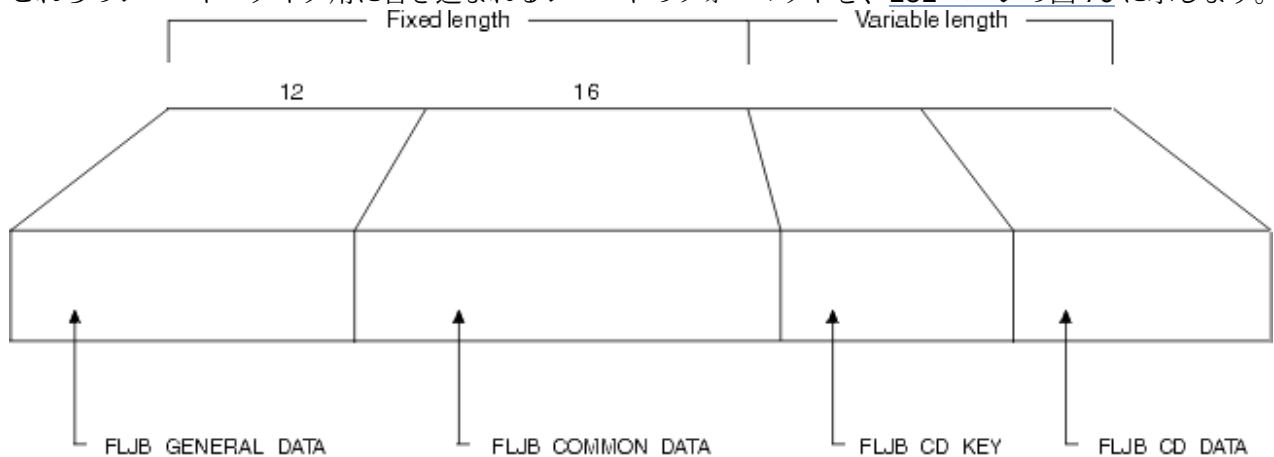


図 70. 読み取り専用、読み取り更新、書き込み更新、書き込み追加、書き込み追加の完了の各レコード・タイプ用に書き込まれるレコードのレイアウト

### FLJB\_GENERAL\_DATA

12 バイトの汎用データ。

[281 ページの『FLJB\\_GENERAL\\_DATA』](#)を参照してください。

### FLJB\_COMMON\_DATA

16 バイトの共通データ。

[283 ページの『FLJB\\_COMMON\\_DATA』](#)を参照してください。

### FLJB\_CD\_KEY

可変長の呼び出し側データ・キー。

### FLJB\_CD\_DATA

可変長の呼び出し側データ。

### FLJB\_GENERAL\_DATA

FLJB\_GENERAL\_DATA セクションのフォーマットが、[282 ページの図 71](#) に示されています。



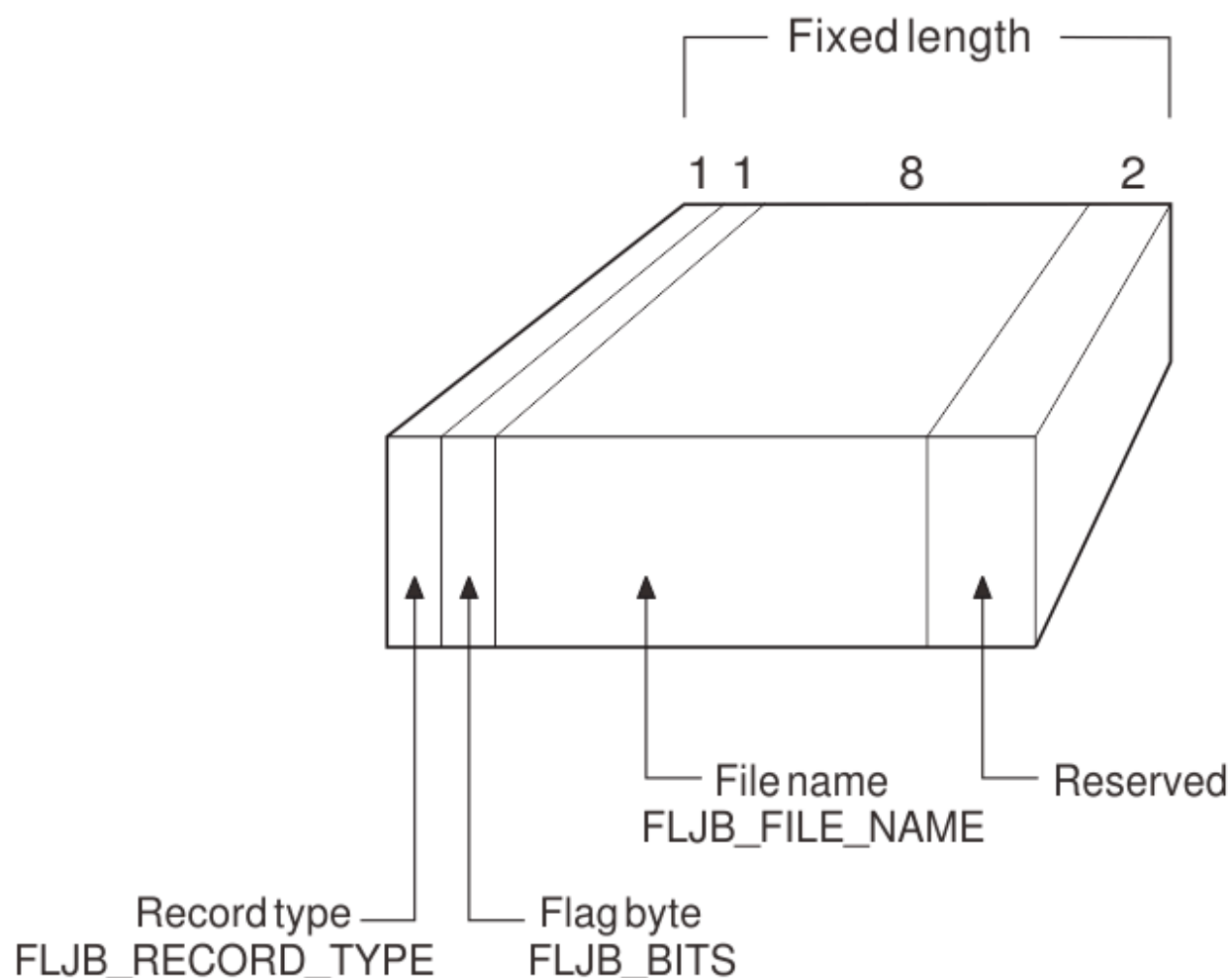


図 71. FLJB\_GENERAL\_DATA セクションのフォーマット

## FLJB\_RECORD\_TYPE

1 バイトのレコード・タイプ:

X'80'	Readonly
X'81'	Read update
X'82'	Write update
X'83'	Write add
X'84'	Write add complete
X'86'	Write delete
X'87'	Commit (replication only)
X'88'	Backout (replication only)
X'89'	Unlock (replication only)
X'8E'	File close
X'8F'	File tie-up record

## FLJB\_BITS

1 バイト・フラグ・フィールド。

X'80'	File control autojournal record
X'40'	Forward recovery log record
X'20'	System log record
X'10'	Log-of-log record
X'08'	Written in backout
X'04'	Data set is extended addressing ESDS
X'02'	Replication record
X'01'	Replication record written by replication tran

## FLJB\_FILE\_NAME

8 バイトのファイル名。

### 予約済み

2 バイトの予約フィールド。

### FLJB\_COMMON\_DATA

FLJB\_COMMON\_DATA セクションのフォーマットが、[283 ページの図 72](#) に示されています。

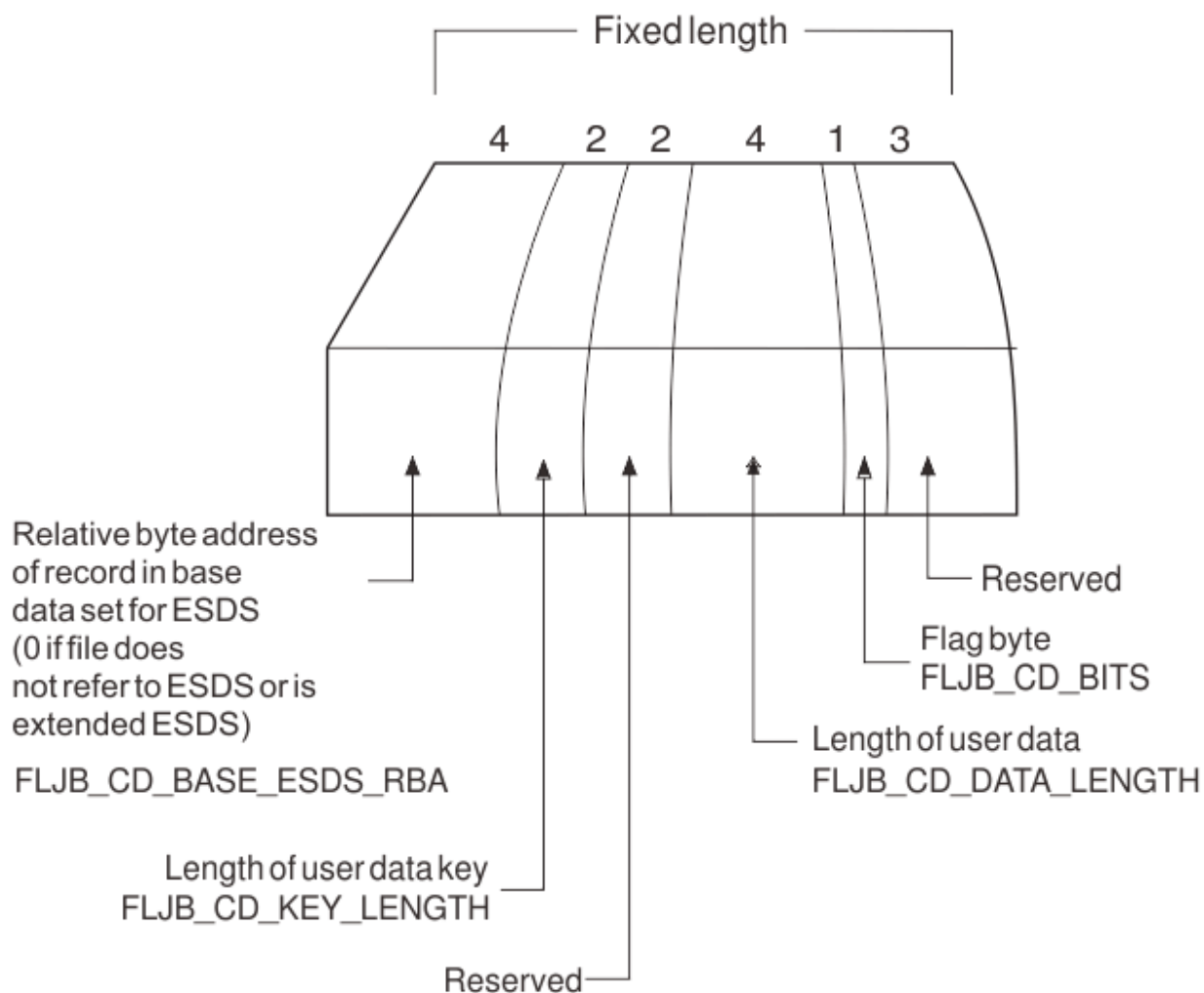


図 72. FLJB\_COMMON\_DATA セクションのフォーマット

### FLJB\_CD\_BASE\_ESDS\_RBA

4 バイトの、ESDS の基本データ・セットのレコードの相対バイト・アドレス。

ファイルが ESDS を参照していない場合、またはそれが拡張アドレッシング ESDS である場合、相対バイト・アドレスは 0 です。

### FLJB\_CD\_KEY\_LENGTH

2 バイトのユーザー・データ・キーの長さ。

キーの長さは、RRDS、VRRDS または標準の ESDS では 4、拡張アドレッシング ESDS では 8 です。

### 予約済み

2 バイトの予約フィールド。

### FLJB\_CD\_DATA\_LENGTH

4 バイトのユーザー・データの長さ。

### FLJB\_CD\_BITS

1 バイト・フラグ・フィールド。

```

X'80'    UOW has been shunted at least once
X'40'    Write massinsert
X'20'    First write-add-complete in massinsert sequence
X'10'    End of massinsert sequence
X'08'    Fixed length record
X'04'    Replication record is auto committed
X'02'    Token used on READ-UPDATE (replication records only)

```

設定を組み合わせる使用することが可能です。

#### 予約済み

3 バイトの予約フィールド。

#### 書き込み削除レコード・タイプ

書き込み削除レコード・タイプ用に書き込まれるジャーナル・レコードは、4つのセクションで構成されます。

これらのセクションは以下のとおりです。

- FLJB\_GENERAL\_DATA セクション
- FLJB\_WRITE\_DELETE\_DATA セクション
- 2つの呼び出し元データ・イメージ・セクション。
  - FLJB\_WDD\_BASE\_KEY (その長さは FLJB\_WRITE\_DELETE\_DATA 内の FLJB\_WDD\_BASE\_KEY\_LENGTH で与えられる)
  - FLJB\_WDD\_PATH\_KEY (その長さは FLJB\_WRITE\_DELETE\_DATA 内の FLJB\_WDD\_PATH\_KEY\_LENGTH で与えられる)

これらのセクションには、呼び出し側のデータのイメージが基本キーとして含まれます。データ・セットがパスの場合は、パスとして含まれます。

#### 書き込み削除レコードのレコード・フォーマット

書き込み削除レコード・タイプ用に書き込まれるレコードのフォーマットを、[284 ページの図 73](#) に示します。

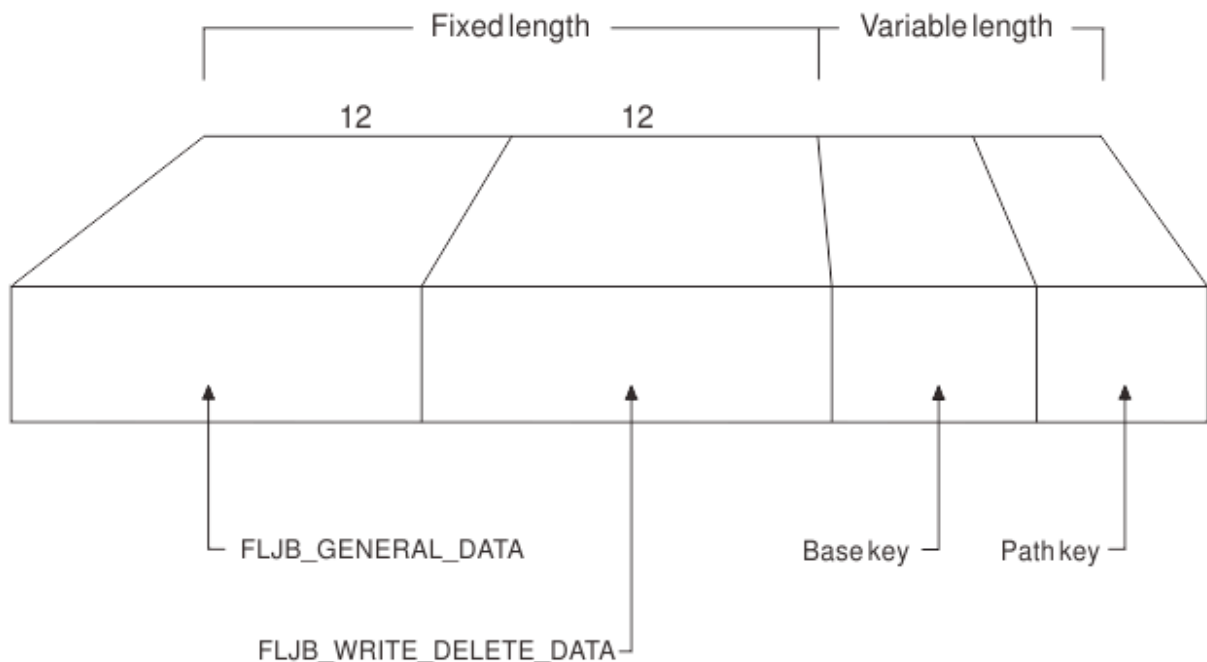


図 73. 書き込み削除レコード・タイプ用に書き込まれるレコードのレイアウト

### FLJB\_GENERAL\_DATA

12 バイトの汎用データ・セクション。

281 ページの『FLJB\_GENERAL\_DATA』を参照してください。

### FLJB\_WRITE\_DELETE\_DATA

12 バイトの書き込み削除データ・セクション。

285 ページの『FLJB\_WRITE\_DELETE\_DATA』を参照してください。

### FLJB\_WDD\_BASE\_KEY

可変長の基本キー。

### FLJB\_WDD\_PATH\_KEY

可変長のパス・キー。

### FLJB\_WRITE\_DELETE\_DATA

FLJB\_WRITE\_DELETE\_DATA セクションのフォーマットが、285 ページの図 74 に示されています。

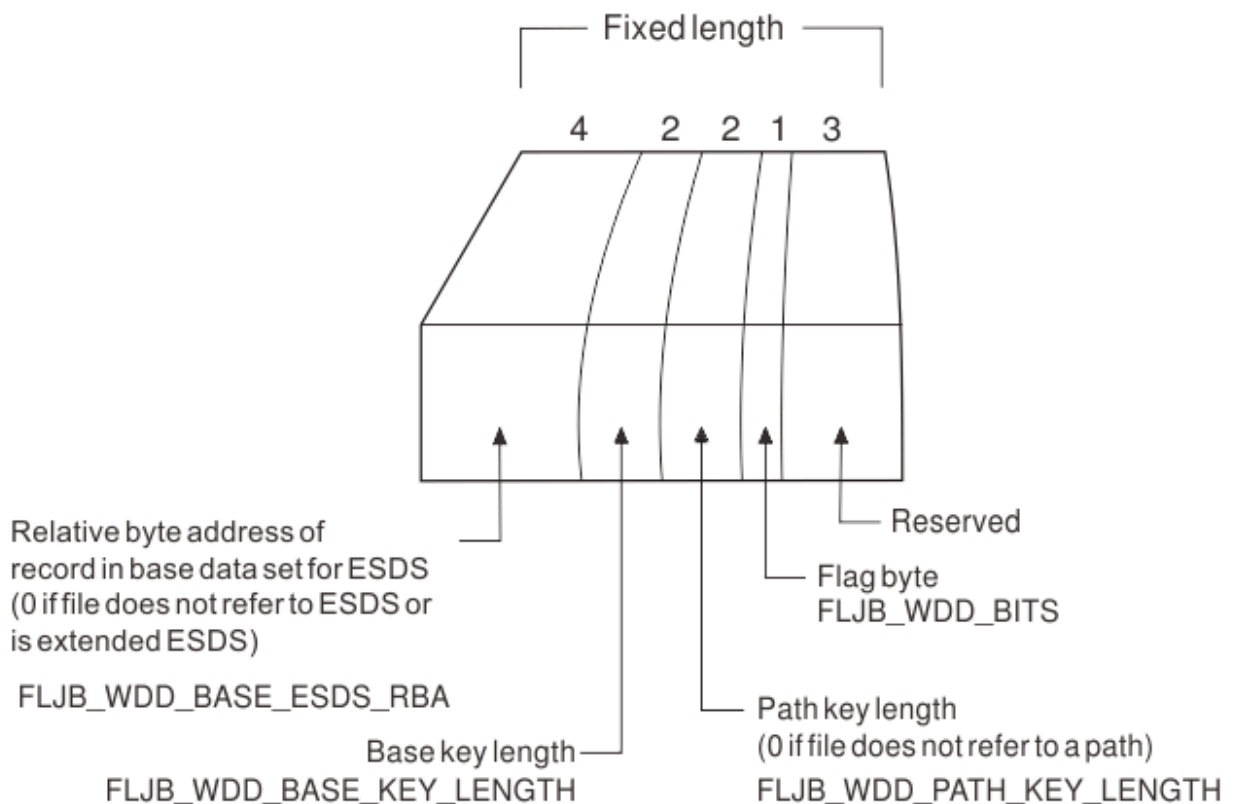


図 74. FLJB\_WRITE\_DELETE\_DATA セクションのフォーマット

### FLJB\_WDD\_BASE\_ESDS\_RBA

4 バイトの、ESDS の基本データ・セットのレコードの相対バイト・アドレス。

ファイルが ESDS を参照していない場合、またはそれが拡張アドレッシング ESDS である場合、相対バイト・アドレスは 0 です。

### FLJB\_WDD\_BASE\_KEY\_LENGTH

2 バイトの基本キーの長さ。

キーの長さは、RRDS、VRRDS または標準の ESDS では 4、拡張アドレッシング ESDS では 8 です。

### FLJB\_WDD\_PATH\_KEY\_LENGTH

2 バイトのパス・キーの長さ。

ファイルがパスを参照していない場合、キーの長さは 0 です。

## FLJB\_WDD\_BITS

1 バイト・フラグ・フィールド。

X'80'	UOW has been shunted at least once
X'40'	Fixed-length record
X'20'	Replication record is auto committed

## 予約済み

3 バイトの予約フィールド。

## コミットおよびバックアウトのレコード・タイプ

コミットおよびバックアウト用書き込まれるジャーナル・レコードは複製ロギングのためにのみ書き込まれ、1つのセクションで構成されます。

- FLJB\_GENERAL\_DATA セクション

## FLJB\_GENERAL\_DATA

12 バイトの汎用データ・セクション。

281 ページの『FLJB GENERAL DATA』を参照してください。

## Unlock record types

アンロック用書き込まれるジャーナル・レコードは複製ロギングのためにのみ書き込まれ、4つのセクションで構成されます。

これらのセクションは以下のとおりです。

- FLJB\_GENERAL\_DATA セクション
- FLJB\_UNLOCK\_DATA セクション
- 2つの呼び出し元データ・イメージ・セクション。
  - FLJB\_UND\_BASE\_KEY (その長さは FLJB\_UNLOCK\_DELETE\_DATA の FLJB\_UND\_BASE\_KEY\_LENGTH で与えられる)
  - FLJB\_UND\_PATH\_KEY (その長さは FLJB\_UNLOCK\_DATA の FLJB\_UND\_PATH\_KEY\_LENGTH で与えられる)

これらのセクションには、呼び出し側のデータのイメージが基本キーとして含まれます。データ・セットがパスの場合は、パスとして含まれます。

## アンロック・レコードのレコード・フォーマット

アンロック・レコード・タイプ用書き込まれるレコードのフォーマットを、286 ページの図 75 に示します。

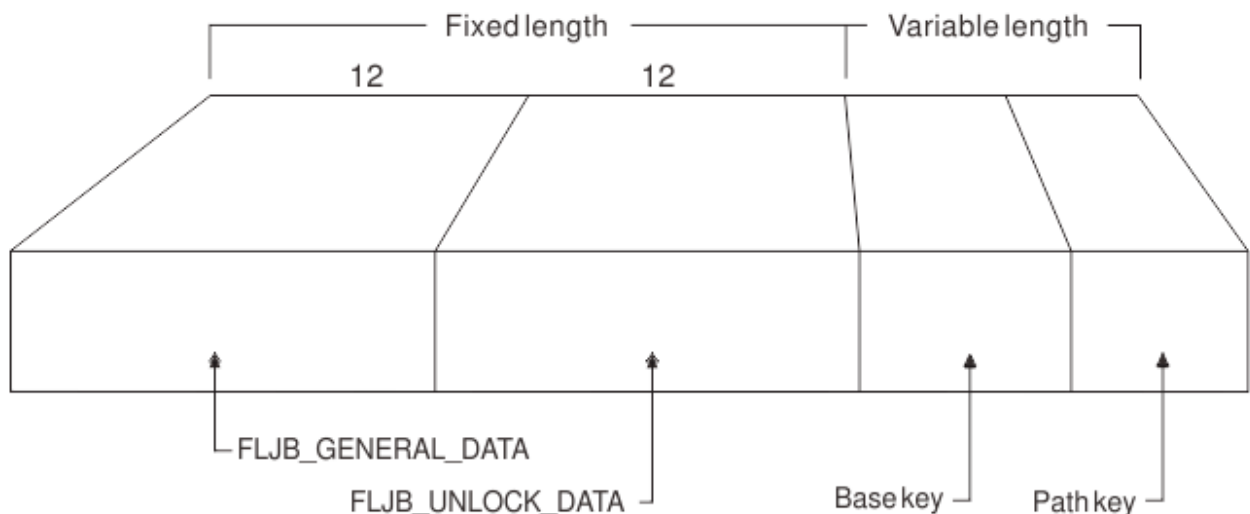


図 75. アンロック・レコード・タイプ用書き込まれるレコードのレイアウト



### FLJB\_GENERAL\_DATA

12 バイトの汎用データ・セクション。

281 ページの『FLJB\_GENERAL\_DATA』を参照してください。

### FLJB\_UNLOCK\_DATA

12 バイトのアンロック・データ・セクション。

287 ページの『FLJB\_UNLOCK\_DATA』を参照してください。

### FLJB\_UND\_BASE\_KEY

可変長の基本キー。

### FLJB\_UND\_PATH\_KEY

可変長のパス・キー。

### FLJB\_UNLOCK\_DATA

FLJB\_UNLOCK\_DATA セクションのフォーマットが、287 ページの図 76 に示されています。

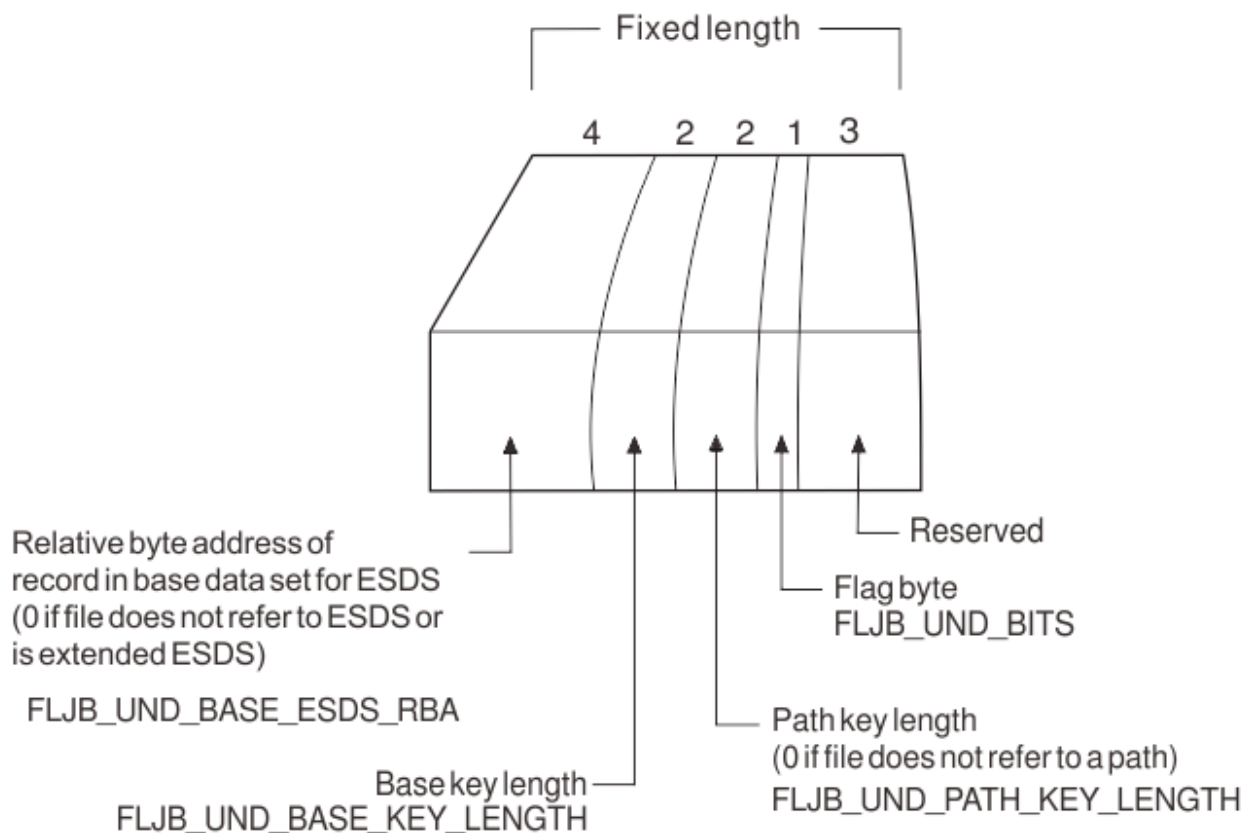


図 76. FLJB\_UNLOCK\_DATA セクションのフォーマット

### FLJB\_UND\_BASE\_ESDS\_RBA

4 バイトの、ESDS の基本データ・セットのレコードの相対バイト・アドレス。

ファイルが ESDS を参照していない場合、またはそれが拡張アドレッシング ESDS である場合、相対バイト・アドレスは 0 です。

### FLJB\_UND\_BASE\_KEY\_LENGTH

2 バイトの基本キーの長さ。

キーの長さは、RRDS、VRRDS または標準の ESDS では 4、拡張アドレッシング ESDS では 8 です。

### FLJB\_UND\_PATH\_KEY\_LENGTH

2 バイトのパス・キーの長さ。

ファイルがパスを参照していない場合、キーの長さは 0 です。

## FLJB\_UND\_BITS

1 バイト・フラグ・フィールド。

X'80'	UOW has been shunted at least once
X'40'	Fixed-length record
X'20'	Replication record is auto committed
X'10'	Unlock follows a read-update
X'08'	Unlock follows a massinsert

## 予約済み

3 バイトの予約フィールド。

## ファイル・クローズ・レコード・タイプ

ファイル・クローズ・レコード・タイプ用書き込まれるジャーナル・レコードは、2つのセクションで構成されます。

これらのセクションは以下のとおりです。

- FLJB\_GENERAL\_DATA セクション
- FLJB\_CLOSE\_DATA セクション

## ファイル・クローズ・レコードのレコード・フォーマット

ファイル・クローズ・レコード・タイプ用書き込まれるレコードのフォーマットを、[288 ページの図 77](#)に示します。

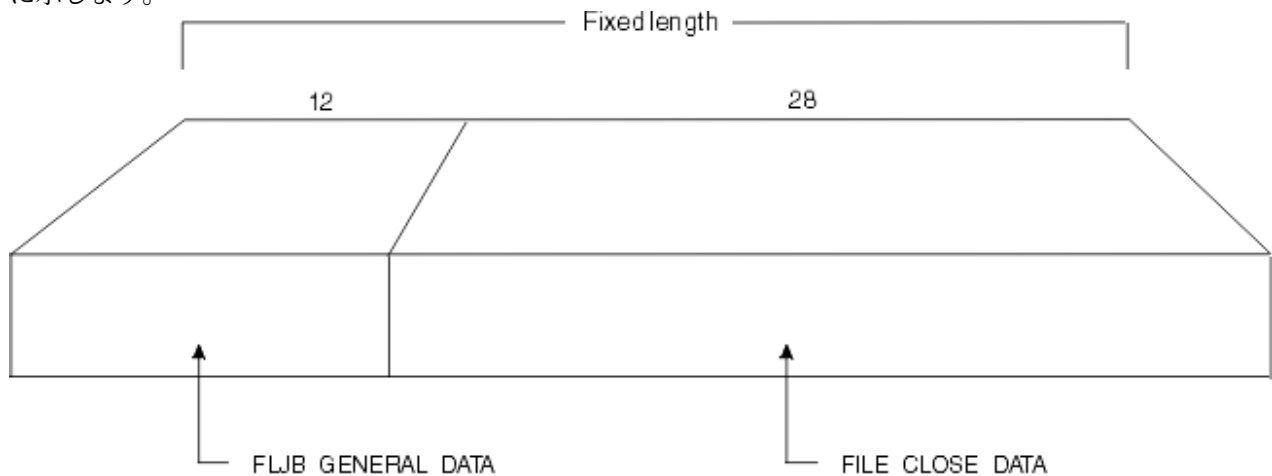


図 77. ファイル・クローズ・レコード・タイプ用書き込まれるレコードのレイアウト

## FLJB\_GENERAL\_DATA

12 バイトの汎用データ・セクション。

[281 ページの『FLJB\\_GENERAL\\_DATA』](#)を参照してください。

## FLJB\_CLOSE\_DATA

28 バイトのクローズ・データ・セクション。

[288 ページの『FLJB\\_CLOSE\\_DATA』](#)を参照してください。

## FLJB\_CLOSE\_DATA

FLJB\_CLOSE\_DATA セクションのフォーマットが、[289 ページの図 78](#)に示されています。

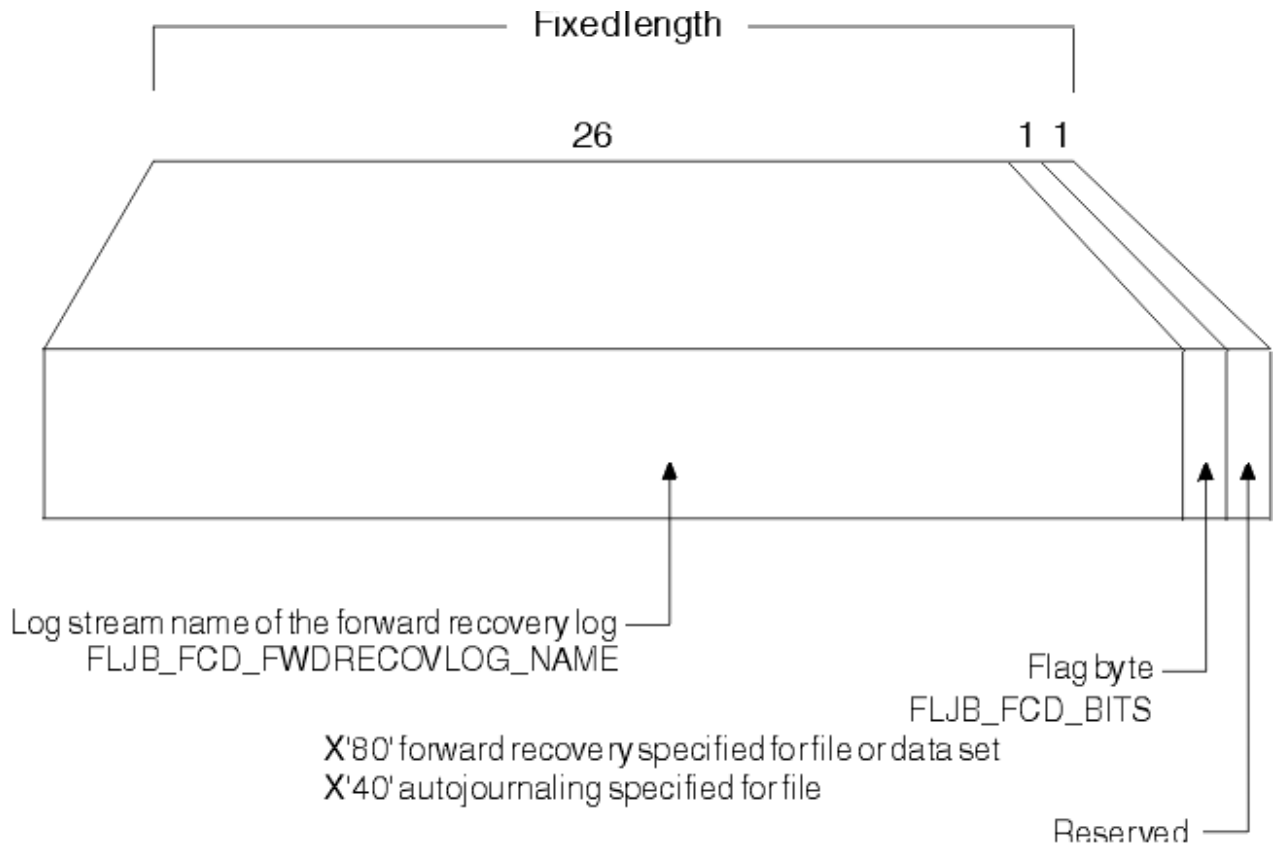


図 78. FILE\_CLOSE\_DATA セクションのフォーマット

#### FLJB\_FCD\_FWDRECOVLOG\_NAME

26 バイトの順方向リカバリー・ログのログ・ストリーム名。

#### FLJB\_FCD\_BITS

1 バイト・フラグ・フィールド。

X'80'	Forward recovery specified for file or data set
X'40'	Autojournaling specified for file

#### 予約済み

1 バイトの予約フィールド。

#### タイアップ・レコード・タイプ

タイアップ・レコード・タイプ用書き込まれるジャーナル・レコードは、2つのセクションで構成されます。

これらのセクションは以下のとおりです。

- FLJB\_GENERAL\_DATA セクション
- TIE\_UP\_RECORD\_DATA セクション

#### タイアップ・レコードのレコード・フォーマット

タイアップ・レコード・タイプ用書き込まれるレコードのフォーマットを、[290 ページの図 79](#) に示します。

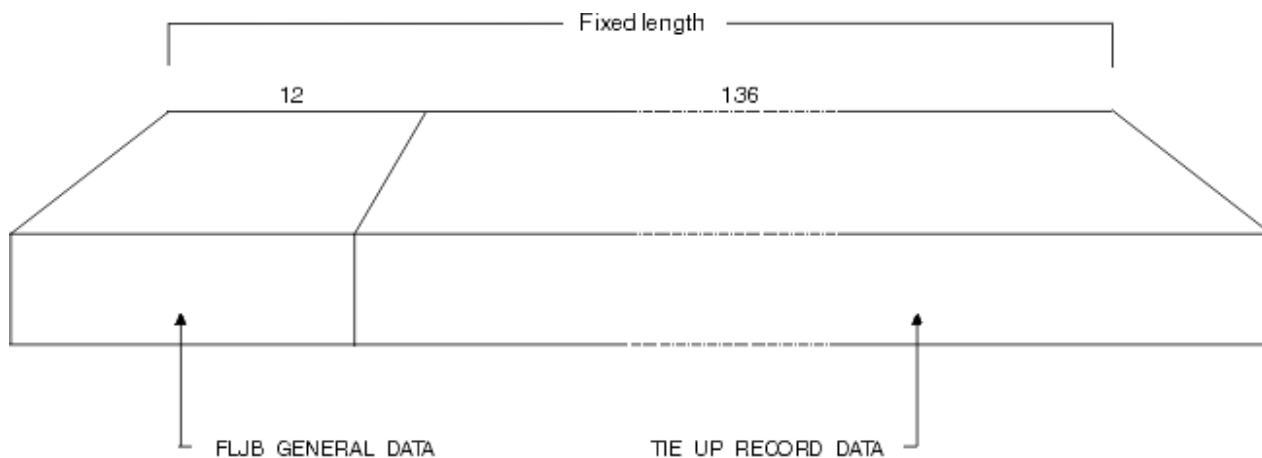


図 79. タイアップ・レコード・タイプ用に書き込まれるレコードのレイアウト

#### **FLJB\_GENERAL\_DATA**

12 バイトの汎用データ。

[281 ページの『FLJB GENERAL DATA』](#)を参照してください。

#### **TIE\_UP\_RECORD\_DATA**

136 バイトのタイアップ・レコード・データ。

[290 ページの『TIE UP RECORD DATA』](#)を参照してください。

#### **TIE\_UP\_RECORD\_DATA**

TIE\_UP\_RECORD\_DATA セクションの フォーマットが、[291 ページの図 80](#) に示されています。

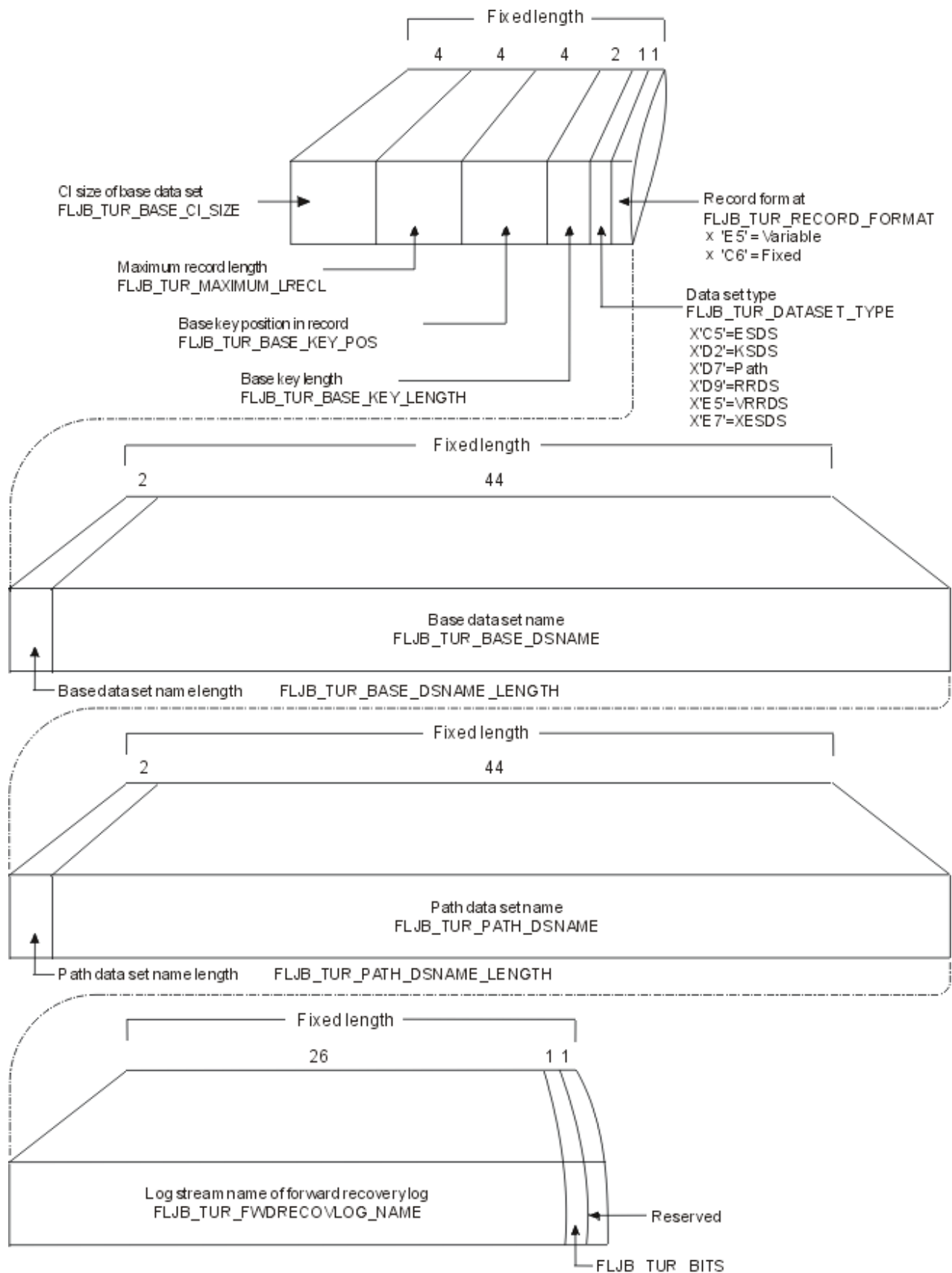


図 80. TIE\_UP\_RECORD\_DATA セクションのフォーマット

#### FLJB\_TUR\_BASE\_CI\_SIZE

4 バイトの基本データ・セットの CI のサイズ。

**FLJB\_TUR\_MAXIMUM\_LRECL**

4 バイトの最大レコード長。

**FLJB\_TUR\_BASE\_KEY\_POSITION**

4 バイトのレコード内の基本キー位置。

**FLJB\_TUR\_BASE\_KEY\_LENGTH**

2 バイトの基本キーの長さ。

**FLJB\_TUR\_DATASET\_TYPE**

1 バイトのデータ・セット・タイプ:

X'C5'	Standard ESDS
X'D2'	KSDS
X'D7'	Path
X'D9'	RRDS
X'E5'	VRRDS
X'E7'	Extended ESDS

**FLJB\_TUR\_RECORD\_FORMAT**

1 バイトのレコード・フォーマット:

X'E5'	Variable
X'C6'	Fixed

**FLJB\_TUR\_BASE\_DSNAME\_LENGTH**

2 バイトの基本データ・セット名の長さ。

**FLJB\_TUR\_BASE\_DSNAME**

44 バイトの基本データ・セット名。

**FLJB\_TUR\_PATH\_DSNAME\_LENGTH**

2 バイトのパス・データ・セット名の長さ。

**FLJB\_TUR\_PATH\_DSNAME**

44 バイトのパス・データ・セット名。

**FLJB\_TUR\_FWDRECOVLOG\_NAME**

26 バイトの順方向リカバリー・ログのログ・ストリーム名。

**FLJB\_TUR\_BITS**

1 バイト・フラグ・フィールド。

**予約済み**

1 バイトの予約フィールド。

注: RLS モードのファイル制御によって書き込まれたジャーナル・レコード内の呼び出し側データのフォーマットは、非 RLS モードのファイル制御によって書き込まれたジャーナル・レコードのものと同一です。ただし FLJB\_TUR\_BITS は例外で、値 X'80' は RLS アクセスを表します。

**端末管理接頭部データ**

CICS 端末管理 (TC) は、ジャーナル・レコードを書き込んで、発行するメッセージを追跡します。各 TC ジャーナル・レコードには、API ユーザー・ヘッダーの接頭部域の位置にある接頭部域が含まれます。

LU6.1 関連のレコードの場合のみ、接頭部域には同期点時間に VTAM® 物理シーケンス番号が含まれます。それ以外のすべての TC ジャーナル・レコードの場合は、2 進ゼロが含まれます。293 ページの図 81 に、TC 接頭部域のフォーマットを示します。



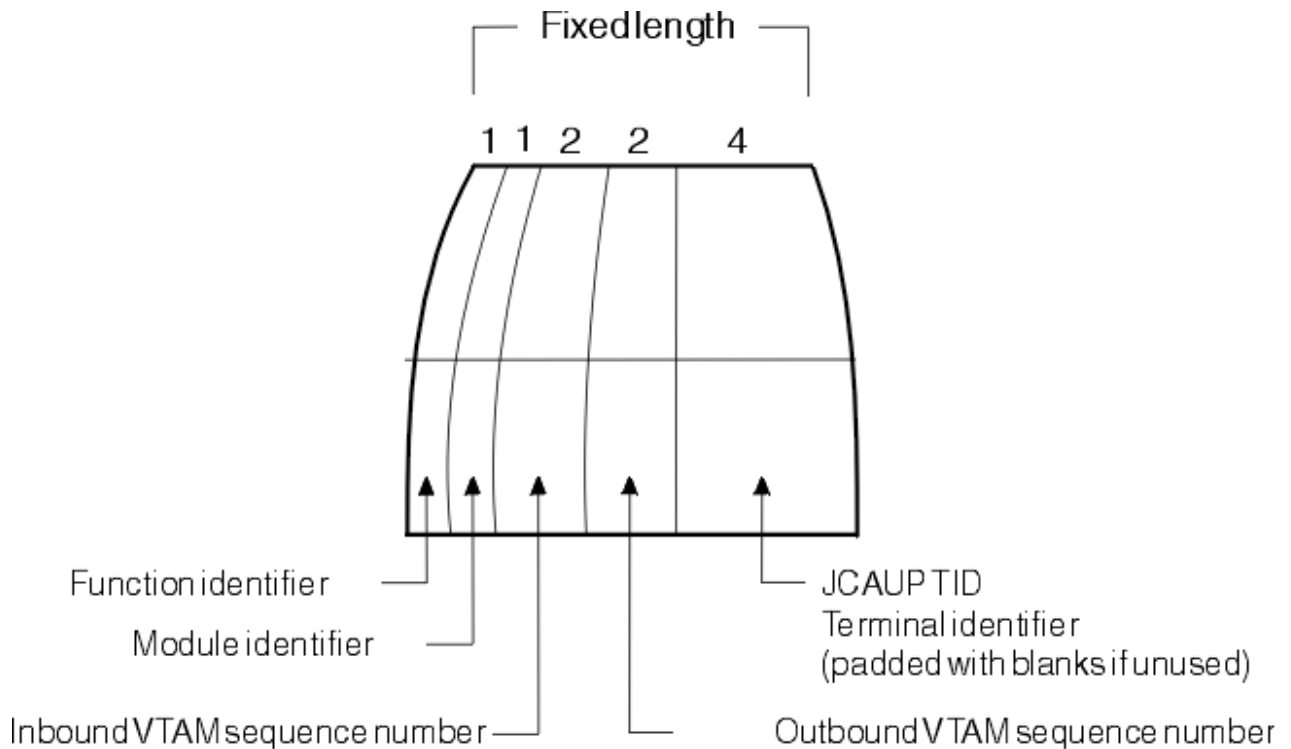


図 81. 端末管理接頭部域のフォーマット

#### 機能 ID

1 バイトの機能 ID。

#### モジュール ID

1 バイトのモジュール ID。

#### インバウンド VTAM SN

2 バイトのインバウンド VTAM シーケンス番号。

#### アウトバウンド VTAM SN

2 バイトのアウトバウンド VTAM のシーケンス番号。

#### JCAUP TID

4 バイトの端末 ID (未使用の場合はブランクで埋められます)。

#### FEPI 接頭部データ

各 FEPI ジャーナル・レコードに含まれる接頭部域によって、データがジャーナル処理された FEPI 会話を識別できます。

この接頭部域は、API ユーザー・ヘッダーの接頭部域の位置にあります。そのフォーマットを [294 ページの図 82](#) に示します。

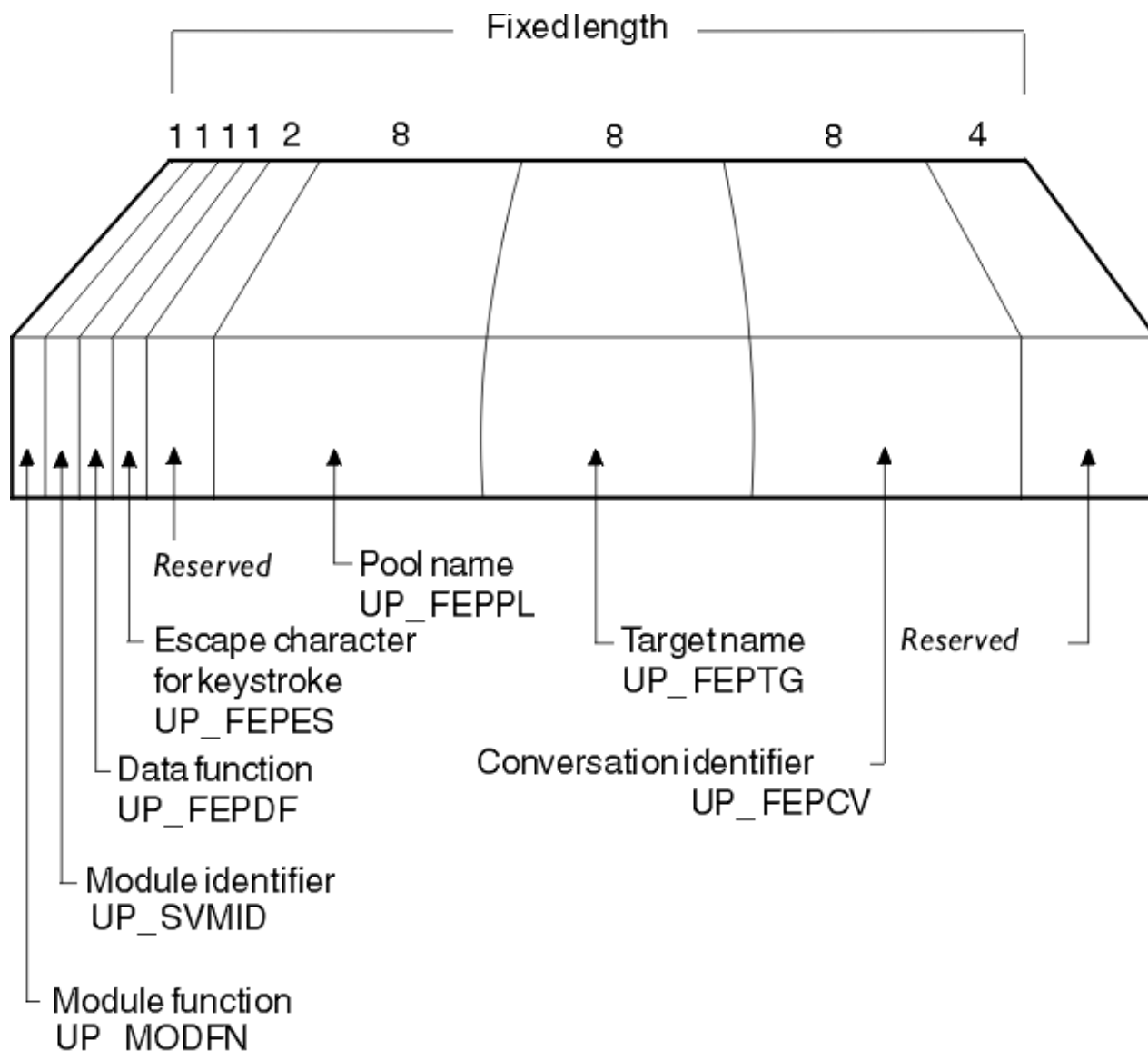


図 82. FEPI 接頭領域の形式

#### UP\_MODFN

1 バイトのモジュール関数

#### UP\_SVMID

1 バイトのモジュール ID

#### UP\_FEPDF

1 バイトのデータ関数

#### UP\_FEPES

キー・ストローク用の 1 バイトのエスケープ文字

#### 予約済み

2 バイトの予約フィールド

#### UP\_FEPPL

8 バイトのプール名

#### UP\_FEPTG

8 バイトのターゲット名

#### UP\_FEPCV

8 バイトの会話 ID

## 予約済み

4 バイトの予約フィールド

## 実行開始時レコード

CICS は、汎用ログに接続されると、CICS のこの実行の最初のレコードとして実行開始時レコードを書き込みます。このレコードは、レコード・ヘッダー (他のすべての汎用ログ・ジャーナル・レコードと同じフォーマットを持つ) と、それに続く実行開始時本体で構成されています。

295 ページの図 83 に、実行開始時レコードのフォーマットを示します。

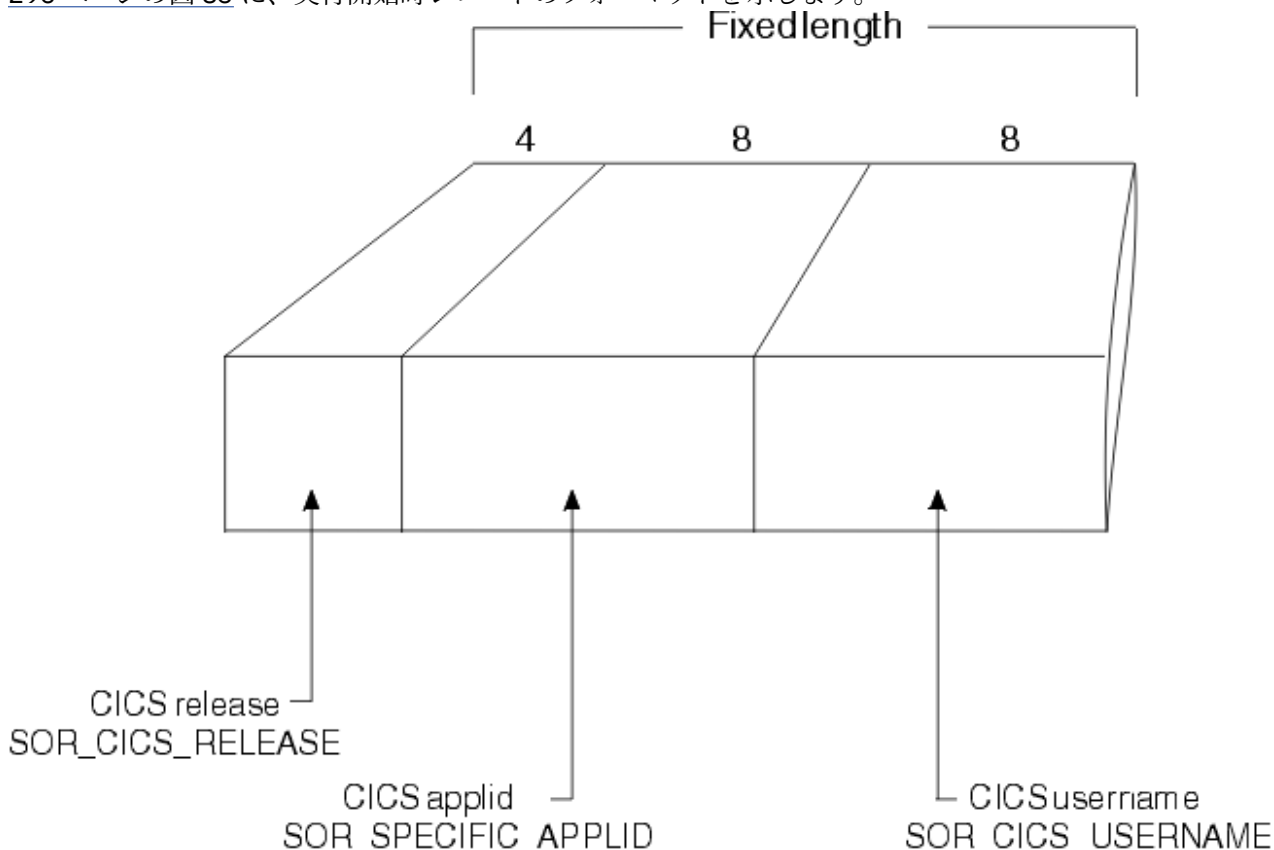


図 83. 実行開始時レコードのフォーマット

### **SOR\_CICS\_RELEASE**

4 バイトの CICS リリース。

### **SOR\_SPECIFIC\_APPLID**

8 バイトの CICS applid。

### **SOR\_CICS\_USERNAME**

8 バイトの CICS ユーザー名。

実行開始時レコードでは、CICS はドメイン ID 「LG」 (「ロガー」の場合) をレコード・ヘッダーの GLRH\_REC\_COMPID フィールドに 配置します。

## COMPAT41 フォーマットのジャーナル・レコードの構造および内容

CICS では、CICS/ESA 4.1 で使用されるフォーマットで表示されるようにジャーナル・レコードのフォーマットを設定できます。JCL の SUBSYS=(LOGR...) ステップで COMPAT41 オプションを使用します。

注：

**SMF レコード**については、307 ページの『SMF に書き込まれたジャーナル・レコードの形式』を参照してください。

このセクションは SMF データ・セットに書き込まれるジャーナル・レコードには適用されません。

## データ内に表示されないフィールド

データのうち、296 ページの表 26 に示されている特定のフィールドは表示されません。それらはフォーマットされた出力で X'00' と表示されます。

表 26. X'00' とフォーマットされるフィールド
フィールド
JCLRJFID
JCLRTBAL
JCSPEMER
JCLRVCD
JCRBB
JCSPMIDT
JCLRVSN
JCSPFS
JCSPRRIF
JCLRLBW
JCSPDSP

## ジャーナル・レコードのフォーマット

各一般ログは、ジャーナル・データの連続ブロックのストリームで構成されます。各ブロックは、ジャーナル管理ラベル・ヘッダーとそれに続く可変数の CICS ジャーナル・レコードで構成されます。各 CICS ジャーナル・レコードは、システム・ヘッダー、システム接頭部、ユーザー接頭部、およびジャーナル・データで構成されます。

296 ページの図 84 に、一般ログの大まかなフォーマットを図示しています。ブロック全体のフォーマットを示しています。

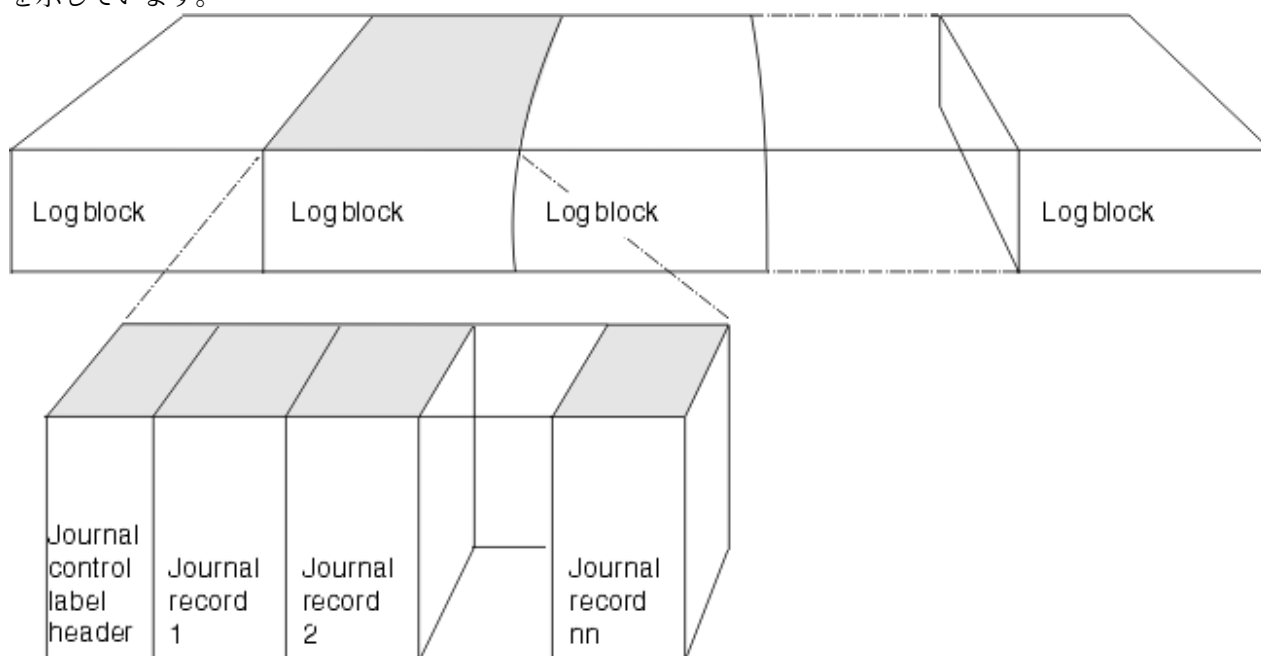


図 84. COMPAT41 オプションを使用してフォーマットされた一般ログのフォーマット

## COMPAT41 ジャーナル管理ラベル・ヘッダー

各ログ・ブロックは、ジャーナル管理ラベル・ヘッダーから始まります。ログ・ブロックごとに1つのジャーナル管理ラベル・ヘッダーがあります。長さは42バイトで、長さフィールド、ラベル・ヘッダー、およびラベル接頭部で構成されています。

ジャーナル管理ラベル・ヘッダーのフォーマットが、[297 ページの図 85](#) に示されています。

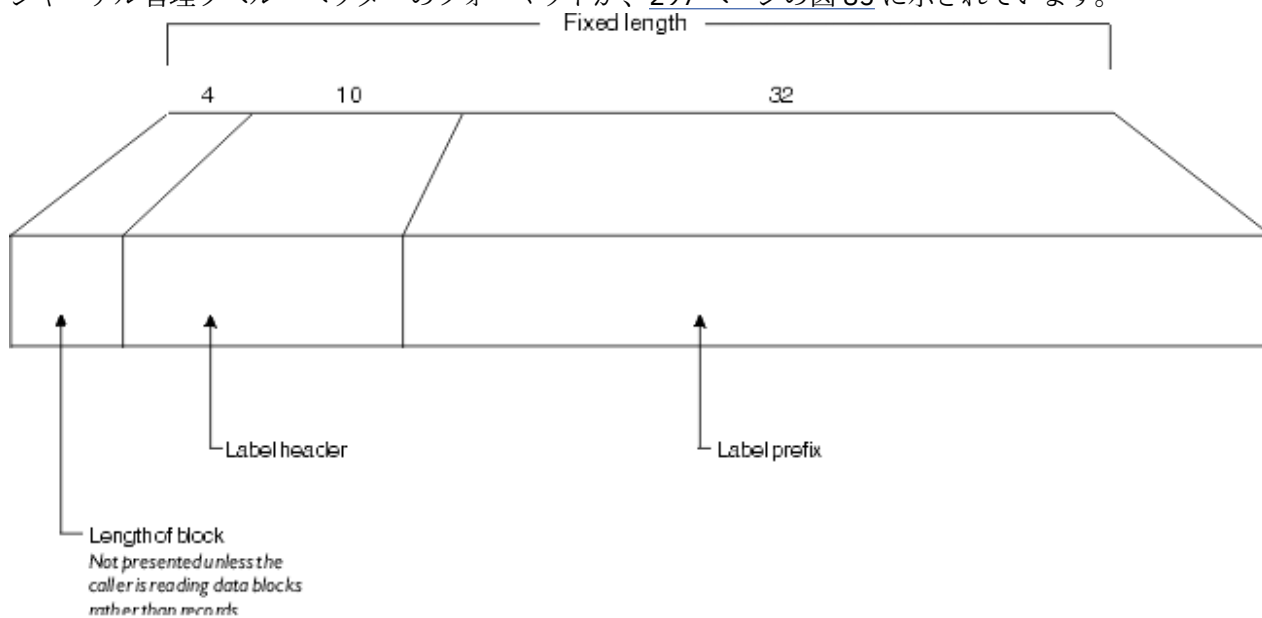


図 85. ジャーナル管理ラベル・ヘッダーのフォーマット

### ラベル・ヘッダー部分

ジャーナル管理ラベル・ヘッダーのラベル・ヘッダー部分の長さは10バイトで、フォーマットは [298 ページの図 86](#) に示されています。

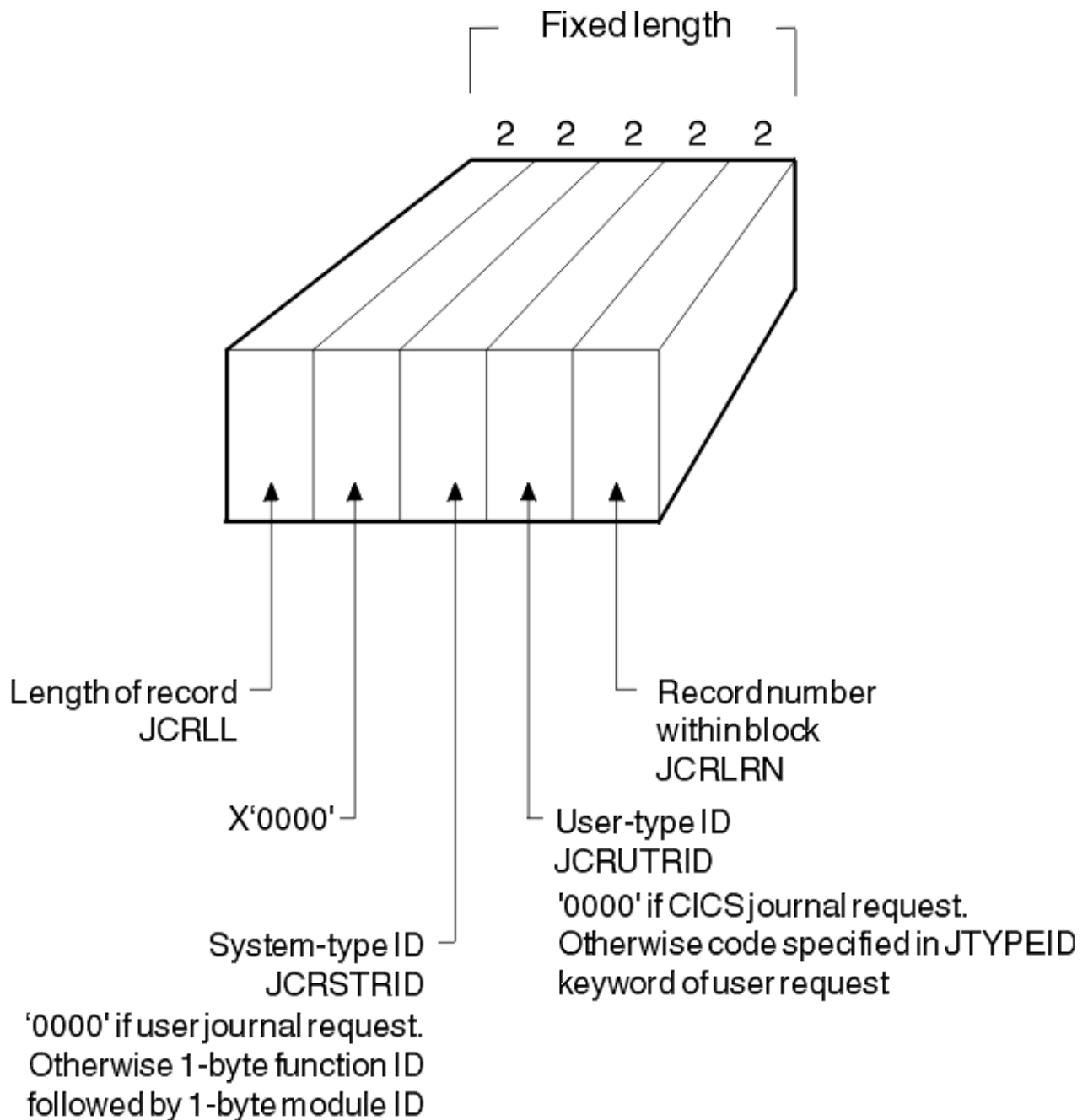


図 86. ジャーナル管理ラベル・ヘッダーのラベル・ヘッダー部分のフォーマット

#### JCRL

2 バイト長のレコード。

#### X'0000'

X'0000' を含む 2 バイト。

#### JCRSTRID

2 バイトのシステム・タイプ ID。ユーザー・ジャーナル要求の場合、これは X'0000' です。それ以外の場合、これは 1 バイトの機能 ID と、それに続く 1 バイトのモジュール ID で構成されています。

#### JCRUTRID

2 バイトのユーザー・タイプ ID。CICS ジャーナル要求の場合、これは X'0000' です。それ以外の場合、ユーザー要求の JTYPEID キーワードによって指定されたコードが含まれます。

#### JCRLRN

ブロック内の 2 バイトのレコード数。



## ラベル接頭部の部分

ジャーナル管理ラベル・ヘッダーのラベル接頭部の部分の長さは 32 バイトで、フォーマットは [299 ページの図 87](#) に示されています。

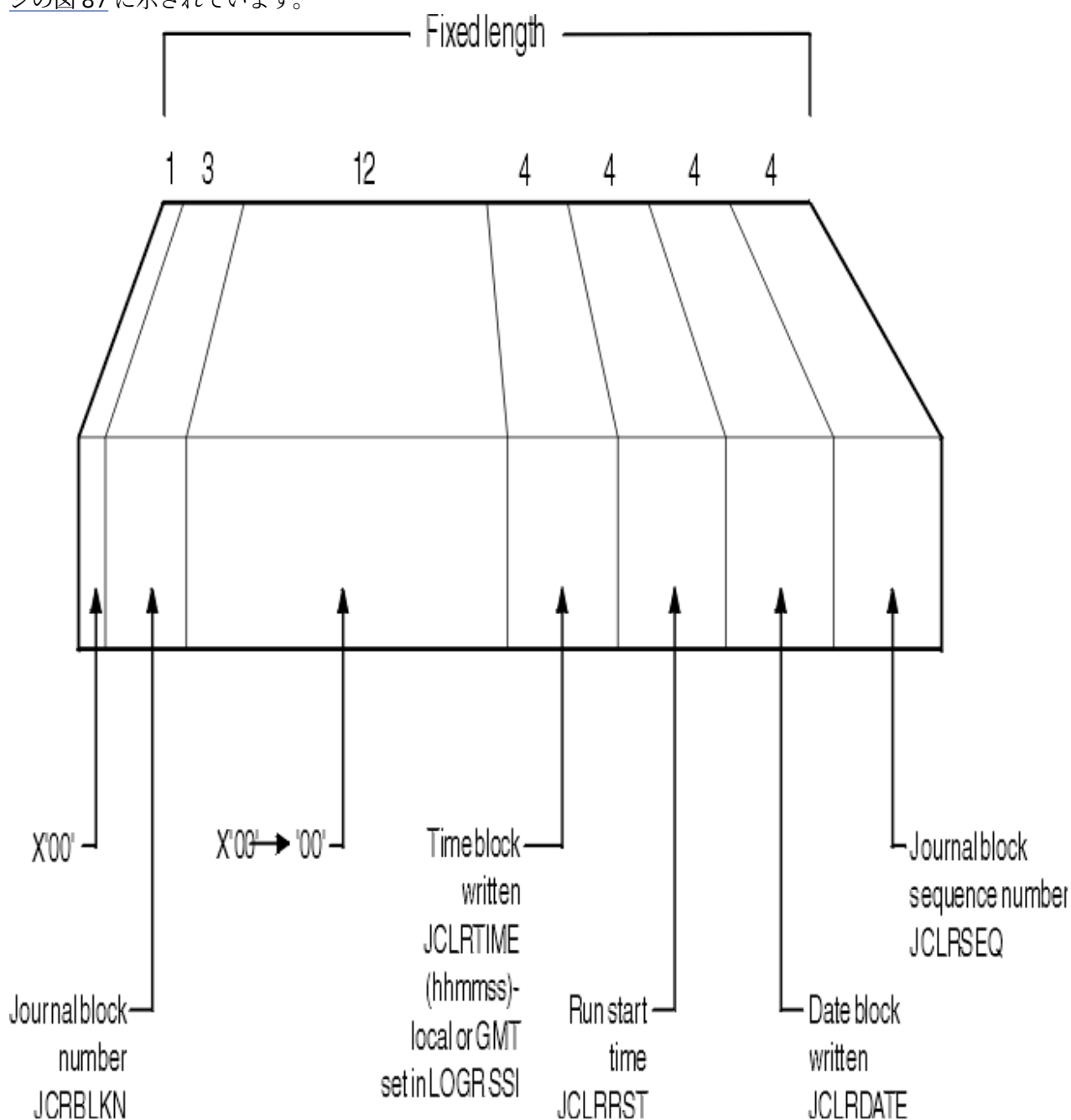


図 87. ジャーナル管理ラベル・ヘッダーのラベル接頭部部分のフォーマット

### X'00'

X'00' を含む 1 バイト。

### JCRBLKN

3 バイトのジャーナル・ブロック番号。

### X'00' (複数)

各バイトに X'00' を含む 12 バイト。

### JCLRTIME

ブロックが書き込まれた時刻 (hhmmss 形式) を含む 4 バイト。(LOGR SSI に設定されたローカルまたは GMT。)

### JCLRRST

実行開始時刻を含む 4 バイト。

### JCLRDATE

ブロックが書き込まれた日付を含む 4 バイト。

### JCLRSEQ

4 バイトのジャーナル・ブロック・シーケンス番号。

## ジャーナル・レコードのフォーマット

各 CICS ジャーナル・レコードは、システム・ヘッダー、システム接頭部、ユーザー接頭部、およびジャーナル・データで構成されます。

300 ページの図 88 に、ジャーナル・レコードのフォーマットを示します。

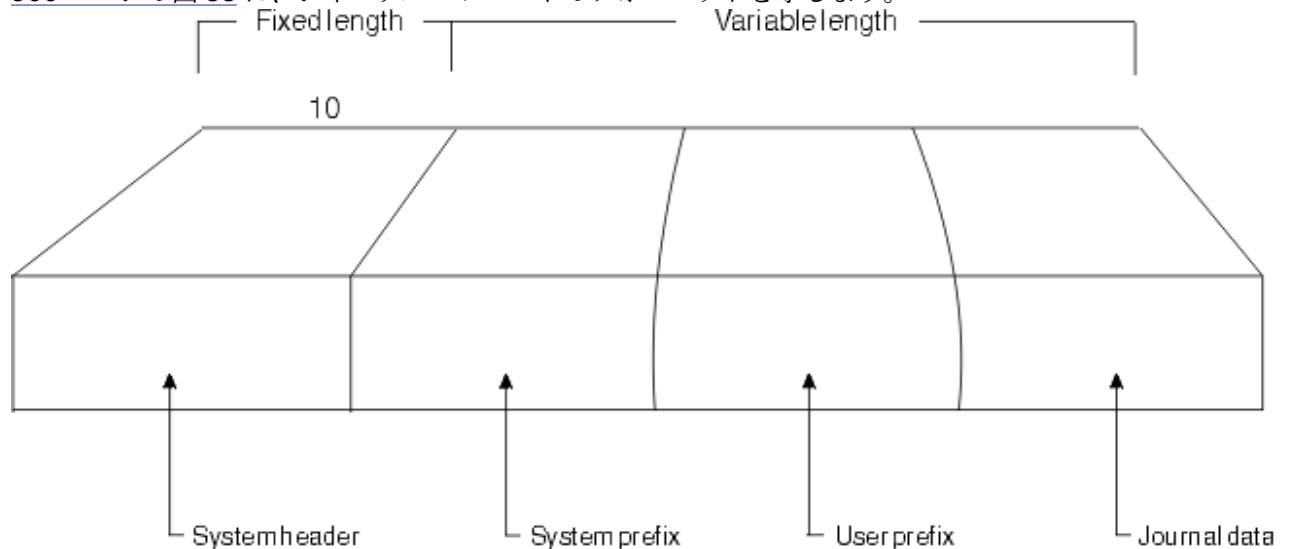


図 88. COMPAT41 ジャーナル・レコードのフォーマット

システム・ヘッダーの長さは 10 バイトです。そのフォーマットを 301 ページの図 89 に示します。

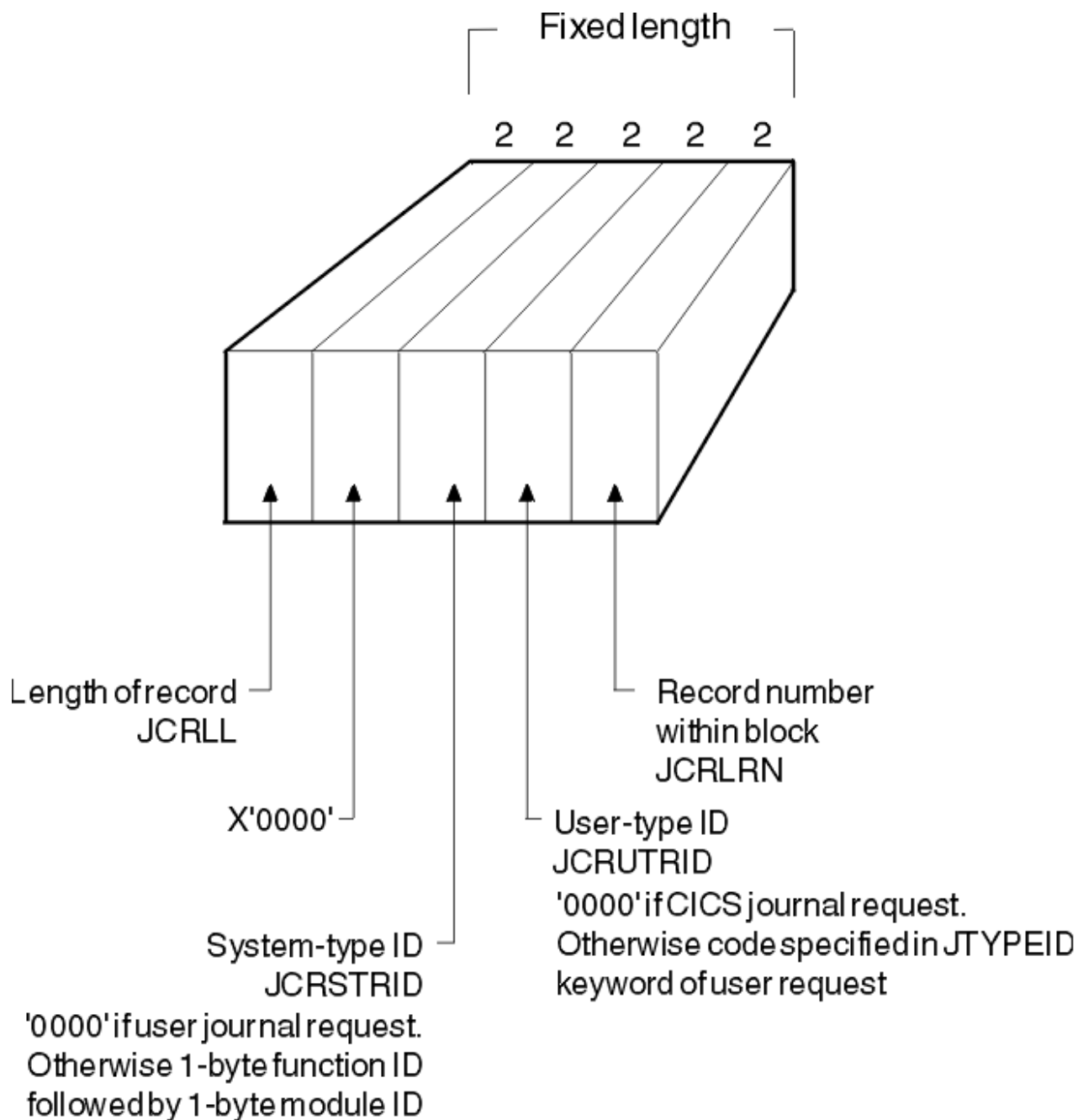


図 89. システム・ヘッダーのフォーマット

#### JCRL

2 バイト長のレコード。

#### X'0000'

X'0000' を含む 2 バイト。

#### JCRSTRID

2 バイトのシステム・タイプ ID。ユーザー・ジャーナル要求の場合、これは X'0000' です。それ以外の場合、これは 1 バイトの機能 ID と、それに続く 1 バイトのモジュール ID で構成されています。

#### JCRUTRID

2 バイトのユーザー・タイプ ID。CICS ジャーナル要求の場合、これは X'0000' です。それ以外の場合、ユーザー要求の JTYPEID キーワードによって指定されたコードが含まれます。

#### JCRLRN

ブロック内の 2 バイトのレコード数。

システム・ヘッダーのフィールド JCRSTRID (システム・タイプ ID) およびフィールド JCRUTRID (ユーザー・タイプ ID) を使用すると、CICS によるこれらのジャーナル・レコード出力 (端末管理などのコンポーネントによる) を、直接のユーザー要求によって発行された出力と区別できます。

CICS ジャーナル要求の場合、JCRUTRID には 2 進ゼロが含まれ、JCRSTRID には 1 バイトの機能コードと、続けて 1 バイトのモジュール・コードが含まれます。機能コードは、どの機能がジャーナル処理されたかを示し、モジュール・コードは、どのモジュールがレコードの書き込みの原因となったかを示します。これらのコードの有効な設定は、CICS アセンブラー言語のマクロ・ライブラリーのメンバー DFHFMIDS に含まれています。305 ページの図 92 に、ジャーナル要求を発行したこれらの CICS コンポーネントの有効な機能 ID が示されています。306 ページの図 94 に、有効なモジュール ID を示しています。

ユーザー・ジャーナル要求の場合、JCRSTRID には常に 2 進ゼロが含まれ、JCRUTRID にはアプリケーション・プログラムの WRITE JOURNALNAME 要求の JTYPEID キーワードによって指定された 2 バイトの 16 進コードが含まれます。

システム接頭部の長さは 20 バイトです。そのフォーマットを 302 ページの図 90 に示します。

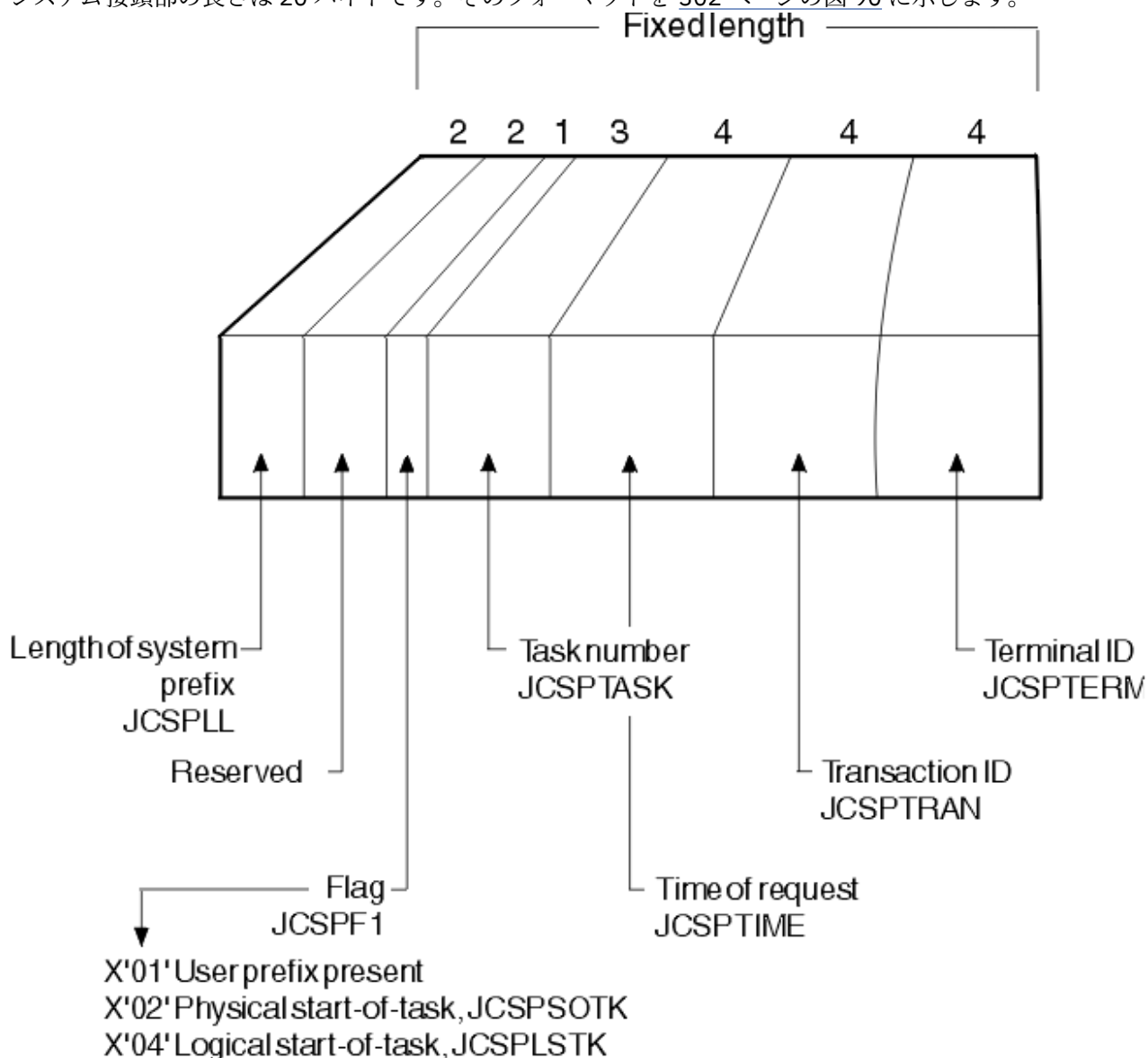


図 90. システム接頭部のフォーマット

#### JCSPLL

2 バイト長のシステム接頭部。

#### 予約済み

2 バイトの予約フィールド。

## JCSPF1

1 バイト・フラグ・フィールド。

X'01'	User prefix present
X'02'	Physical start-of-task, JCSPS0TK
X'04'	Logical start-of-task, JCSPSTK

## JCSPTASK

3 バイトのタスク番号。

## JCSPTIME

4 バイトの要求の時間。

## JCSPTRAN

4 バイトのトランザクション ID。

## JCSPTERM

4 バイトの端末 ID。

一部の CICS ジャーナル要求には、要求の発信元をより具体的に識別するための追加のデータがシステム接頭部に含まれています。この追加のデータはシステム接頭部の共通フィールドの 後ろに続き、通常は可変長です。したがって、システム接頭部の先頭には、長さフィールド JCSPLL が必要です。以降に続くデータは、すべて独自の接頭部レイアウトを持ち、それらは診断とリカバリーの目的で [データ域](#)に説明されています。

ユーザー接頭部は、可変長の領域です。これは、このレコードがユーザー要求 EXEC CICS WRITE JOURNALNAME コマンドによって書き込まれた場合に存在します。レコードに含まれる情報は、JTYPEID、PREFIX、および PFXLENG の各パラメーターを使用して、コマンドの範囲内でユーザーが設定します。そのフォーマットを [304 ページの図 91](#) に示します。

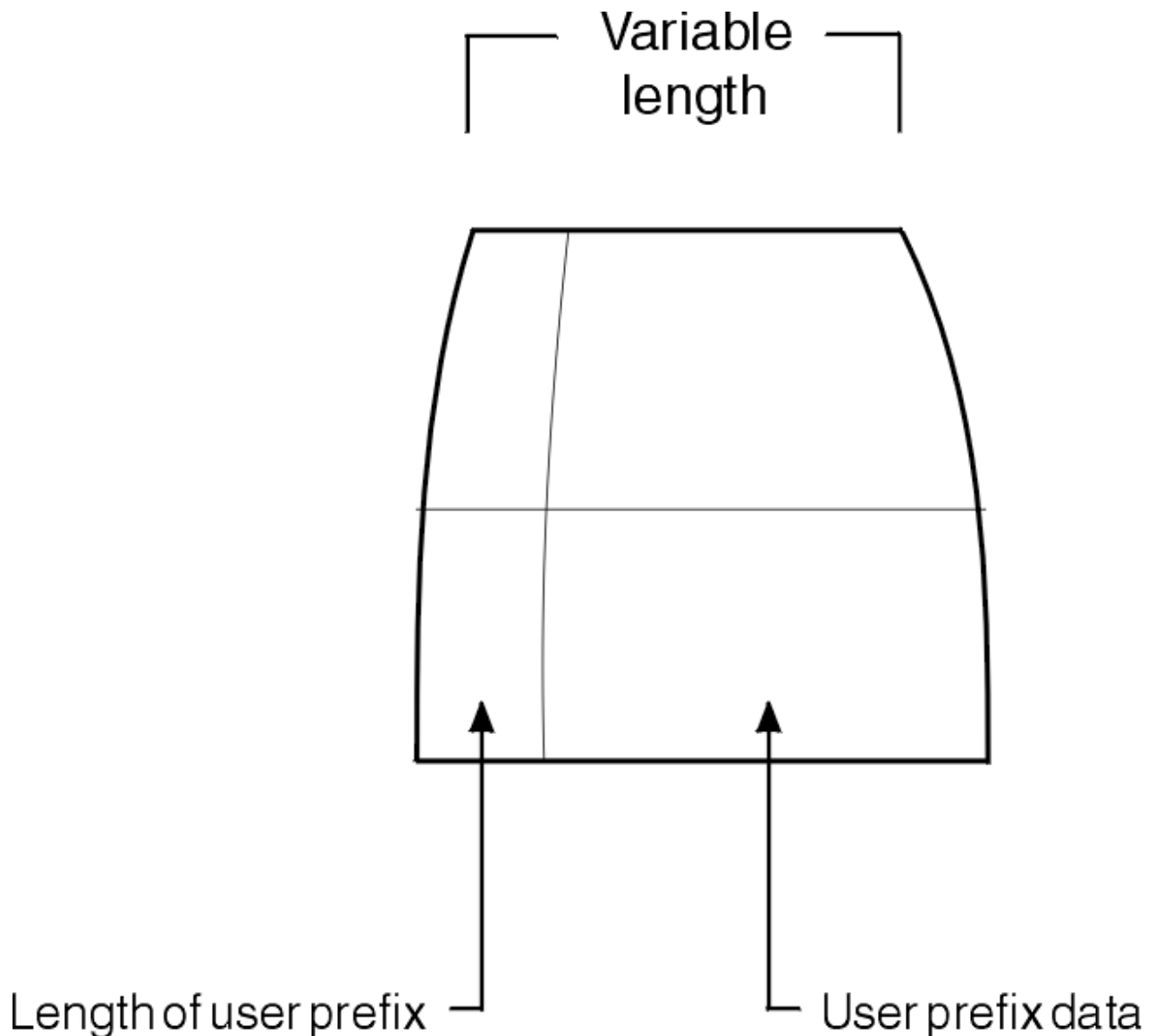


図 91. ユーザー接頭部のフォーマット

フィールド JCSPUP は、ユーザー接頭部がジャーナル・レコードにある場合は、システム接頭部領域に設定されます。

ジャーナル・データがその後に続きます。データの長さフィールドが必要な場合は、それをデータに含める必要があります。または、次のように、システム・ヘッダーの長さ (10 バイト) にシステム接頭部の長さ (JCSPLL) を足し、さらにユーザー接頭部の長さ (ある場合。ユーザー自身が定義したフィールド内のもの) を足した合計をジャーナル・レコードの長さ (JCRLL) から引くことで、ジャーナル・レコードのデータ部分の長さを計算することができます。

JCRLL - (system header (10) bytes + JCSPLL + user prefix)

すべてのジャーナル・レコードに、ジャーナル・データが含まれるわけではありません。

ジャーナル処理要求を発行する CICS コンポーネントは、ジャーナル管理、ファイル管理、FEPI、および端末管理です。



```

*****
* *      F U N C T I O N   I D E N T I F I E R S      * *
*****
*
*      X'20' PLUS X'8-' ...USE FOR AUTOMATIC JOURNALING      *
*      X'40' PLUS X'8-' ...USE FOR AUTOMATIC LOGGING        *
*****
* *      JOURNAL CONTROL      * *
*****
FIDJCLAB EQU  X'80'      ...JOURNAL CONTROL LABEL
*                      RECORD (DFHJCR)      *
*****
* *      F I L E   C O N T R O L      * *
*****
FIDALOG EQU  X'40'      ...AUTOMATICALLY LOGGED
FIDAJRN EQU  X'20'      ...AUTOMATICALLY JOURNALED
FIDMASS EQU  X'10'      ...MASSINSERT REQ. (FIDFCWA ONLY)
*                      PLUS ONE OF...      *
FIDFCRO EQU  X'80'      ...FILE CONTROL READ-ONLY
FIDFCRU EQU  X'81'      ...FILE CONTROL READ-UPDATE
FIDFCWU EQU  X'82'      ...FILE CONTROL WRITE-UPDATE
FIDFCWA EQU  X'83'      ...FILE CONTROL WRITE-ADD
FIDFCWAC EQU  X'84'      ...FILE CONTROL WRITE-ADD-COMPLETE
FIDFCWD EQU  X'86'      ...FILE CONTROL WRITE DELETE
FIDFCBOF EQU  X'88'      ...BACKOUT FAILED LOG RECORD
FIDFCDSN EQU  X'8F'      ...DSNAME RECORD
*
*      NOTE THAT FID* VALUES (AS ABOVE) ARE OFTEN USED BOTH TO
*      IDENTIFY THE FUNCTION OF THE DWE AND THE FUNCTION OF THE
*      LOG RECORD.  IN THE CASE OF THE FIDFC* EQU'S ABOVE, THEY
*      ARE USED FOR LOG RECORDS ONLY.  THOSE BELOW APPLY ONLY
*      TO DWE'S
*
FIDFCVWA EQU  X'80'      THIS DWE ADDRESSES A VSWA.
FIDFCRVY EQU  X'40'      THIS DWE IS ASSOCIATED WITH A
                          RECOVERABLE CHANGE.

```

図 92. ジャーナル機能 ID (パート 1)

```

*****
*                TERMINAL CONTROL FUNCTION IDENTIFIERS                *
*                                                                 *
FIDTCML EQU X'F0'          SYNCPOINT - LOG SEQUENCE
*                            NUMBERS                                *
*                            CAN BE OR'ED WITH ANY OF                *
*                            THE FOLLOWING THREE FIELDS:
FIDTCDWL EQU X'01'          ...DEFERRED WRITE DATA
FIDTCFMH EQU X'02'          ...+ FUNCTION MANAGEMENT
*                            HEADER
FIDTCDIP EQU X'04'          ...+ DIP REQUEST
*                                                                 *
*                            EQU X'08'          ...DYNAMIC BACKOUT MASK
*                            RESERVED
FIDTCAL EQU X'40'          AUTOMATIC LOGGING MASK...
FIDTCAJ EQU X'20'          AUTOMATIC JOURNALING MASK..
*                            ...THE ABOVE 2 PLUS 1 OF FOLLOWING SET
FIDTCTL EQU X'80'          ...SEQUENCE NUMBER ONLY
*                            (LOG ONLY)
FIDTCIM EQU X'81'          ...INPUT MESSAGE (LOG AND
*                            JOURNAL)
FIDTCOM EQU X'82'          ...OUTPUT MESSAGE (JOURNAL
*                            ONLY)
FIDTCWP EQU X'83'          ...WRITE WAS PURGED (LOG
*                            ONLY)
FIDTCPRR EQU X'84'          ...POSITIVE RESPONSE
*                            RECEIVED (LOG ONLY)
FIDTCIMF EQU X'85'          ...INPUT MESSAGE (W/FMH,
*                            LOG AND JOURNAL)
FIDTCOMN EQU X'86'          ...OUTPUT MESSAGE, (W/O
*                            FMH, JOURNAL ONLY)
FIDTCOJ EQU X'87'          ...OUTPUT MESSAGE, FMH,
*                            CCOMPL=NO
FIDTCOJ EQU X'88'          ...OUTPUT MESSAGE, W/O FMH,
*                            ...CCOMPL=NO
FIDTCUA EQU X'89'          ...INITIAL TCT USER AREA
FIDTCEIB EQU X'8A'          ...INITIAL EXEC COMM AREA
FIDTCIMN EQU X'8B'          INPUT MSG, NO FMH, COMPLETE
FIDTCINN EQU X'8C'          INPUT MSG, NO FMH, INCOMPLETE
*****
*                FRONT END PROGRAMMING INTERFACE IDENTIFIERS        *
*                                                                 *
FIDFEPIN EQU X'F0'          FEPI INBOUND DATA  API <--- FEPI
FIDFEPOU EQU X'F1'          FEPI OUTBOUND DATA  API ---> FEPI
*****

```

図 93. ジャーナル機能 ID (パート 2)

```

*****
* *                MODULE IDENTIFIERS                * *
*****
*                                                                 *
MODIDTC EQU X'10'          ...TERMINAL CONTROL
MODIDFC EQU X'11'          ...FILE CONTROL
MODIDJC EQU X'45'          ...JOURNAL CONTROL
MODIDFEP EQU X'50'          ...FEPI
*                                                                 *
*****

```

図 94. ジャーナル・モジュール ID

注:

1. 自動ジャーナル処理と自動ログ記録によって作成されたレコードは、それぞれ X'20' (FIDAJRN) と X'40' (FIDALOG) の値によって識別され、機能 ID の「基本」の値に追加されます。
2. 順方向リカバリー・プロセス (書き込み削除が行われた場所) によって作成されたファイル制御書き込み削除レコードの機能 ID は X'86' (FIDFCWD) です。ただし、レコードが自動ジャーナル・プロセスによって作成された場合、機能 ID は X'A2' で、X'82' (書き込み更新または FIDFCWU) と X'20' (FIDAJRN) を足したもので構成されます。

## タスクおよび UOW の開始のレコードの特定

システム 接頭部フィールド JCSPF1 の値を調べることで、タスクの開始の目印となるレコードを特定できます。JCSPSOTK ビットが設定されている場合、このレコードはタスクの開始時に書き込まれています。

JCSPLSTK ビットがフィールド JCSPF1 に設定されている場合、そのレコードは UOW の開始時に書き込まれています。

## SMF に書き込まれたジャーナル・レコードの形式

SMF データ・セットに書き込まれたジャーナル・レコードを分析する独自のプログラムを作成するには、データの形式を知っている必要があります。

ジャーナル・レコードには、SMF ブロック・ヘッダー、CICS プロダクト・セクション、CICS データ・セクションの 3 つの構成要素があります。[308 ページの図 95](#) は MVS SMF ログのレイアウトです。ログ・ブロックと CICS セクションが示されています。

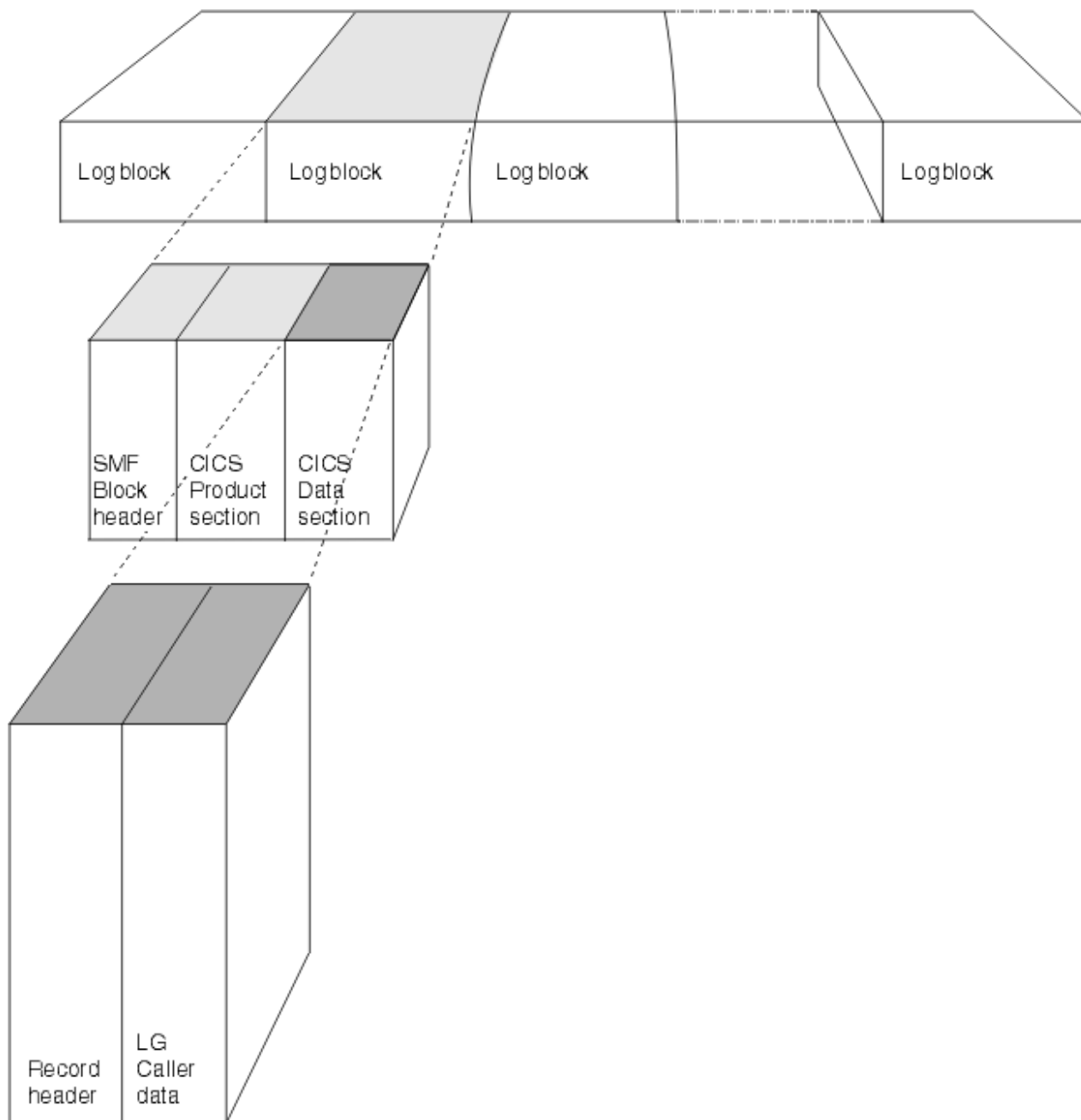


図 95. MVS SMF に書き込まれた CICS ログのレイアウト

SMF に書き込まれたジャーナル・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。このようなプログラムに `INCLUDE DFHLGMSD` ステートメントを組み込んでジャーナル・レコードをマップすることができます。それにより、アセンブラー・バージョンの DSECT が生成されます。

## SMF ブロック・ヘッダー

SMF ブロック・ヘッダーは、出力を作成するシステムを記述します。

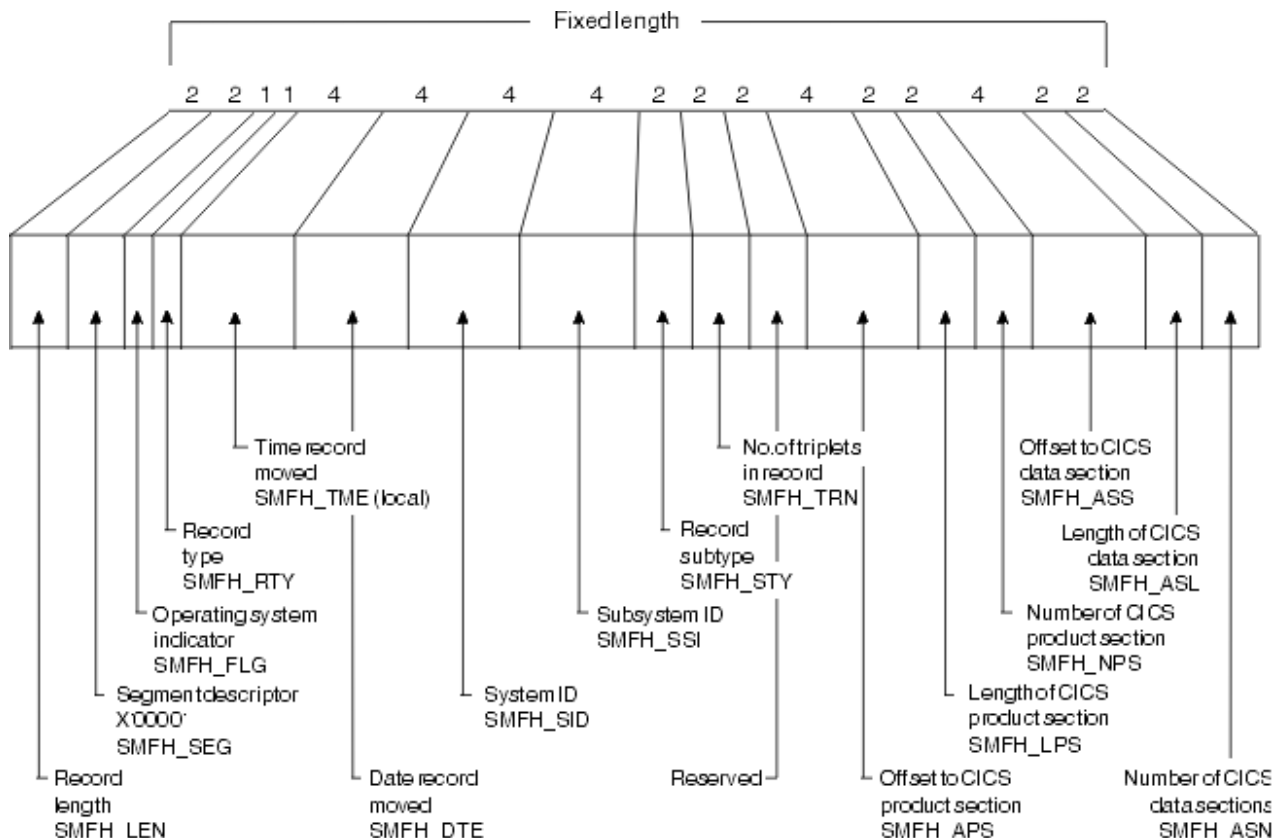


図 96. SMF ブロック・ヘッダーの形式

SMF ブロック・ヘッダーの形式は次のとおりです。

### SMFH\_LENGTH

2 バイトのレコード長。

### SMFH\_SEG

2 バイトのセグメント記述子 (X'0000')。

### SMFH\_FLG

1 バイトのオペレーティング・システム 標識。

### SMFH\_RTY

1 バイトのレコード・タイプ。

### SMFH\_TME

移動した 4 バイトの (ローカル) 時刻レコード。

### SMFH\_DTE

移動した 4 バイトの日付レコード。

### SMFH\_SID

4 バイトのシステム ID。

### SMFH\_SSI

4 バイトのサブシステム ID。

### SMFH\_STY

2 バイトのレコード・サブタイプ。

### SMFH\_TRN

レコード内の 2 バイトのトリプレット数。

**予約済み**

2 バイトの予約フィールド。

**SMFH\_APS**

CICS 製品セクションへの 4 バイトのオフセット。

**SMFH\_LPS**

CICS 製品セクションの 2 バイトの長さ。

**SMFH\_NPS**

2 バイトの CICS 製品セクション番号。

**SMFH\_ASS**

CICS データ・セクションへの 4 バイトのオフセット。

**SMFH\_ASF**

CICS データ・セクションの 2 バイトの長さ。

**SMFH\_ASN**

2 バイトの CICS データ・セクション数。

注：CICS は、オペレーティング・システム標識フラグ・バイトのサブシステム関連ビットのみを SMF ヘッダー (SMFH\_LG) に設定します。SMF は、オペレーティング・システムのレベルおよびその他の要因に従って、残りのバイトを設定します。他のビットの設定に関する説明については、[z/OS MVS システム管理機能 \(SMF\)](#)を参照してください。

**CICS 製品セクション**

CICS 製品セクションは、ジャーナリング・データが関連付けられるサブシステムを識別します。

そのフォーマットを [311 ページの図 97](#) に示します。



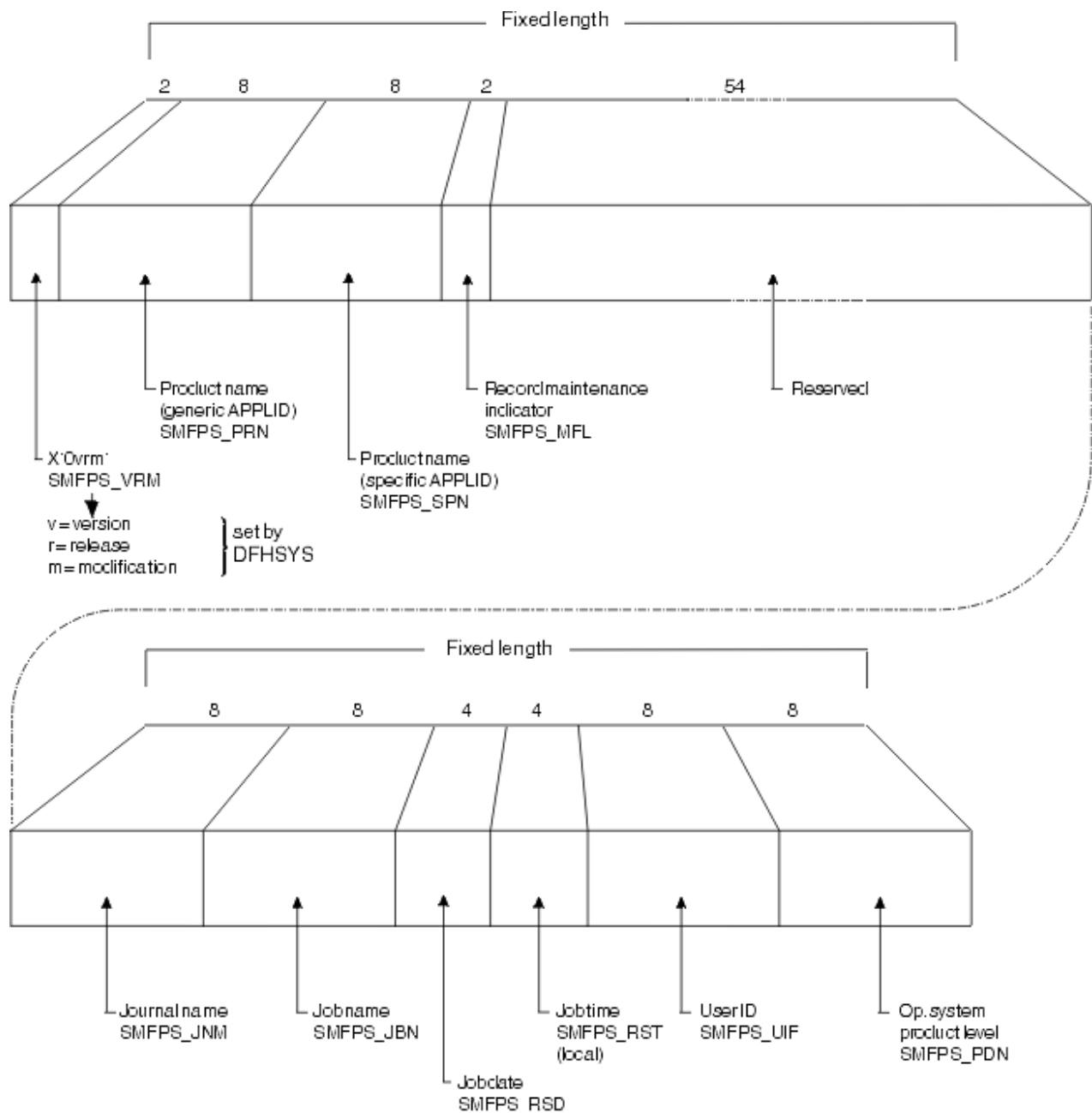


図 97. CICS 製品セクションの形式

CICS 製品セクションの形式は次のとおりです。

#### SMFPS\_VRM

2 バイトの CICS バージョン、リリース、およびモディフィケーションの情報 (X'0vrn' の形式)。

#### SMFPS\_PRN

8 バイトの製品名 (汎用 APPLID)。

#### SMFPS\_SPN

8 バイトの製品名 (特定の APPLID)。

#### SMFPS\_MFL

2 バイトのレコード保守標識。

#### 予約済み

54 バイトの予約フィールド。

#### SMFPS\_JNM

8 バイトのジャーナル名。

**SMFPS\_JBN**

8 バイトのジョブ名。

**SMFPS\_RSD**

4 バイトのジョブ日付。

**SMFPS\_RST**

4 バイトのジョブ時間 (ローカル)。

**SMFPS\_UIF**

8 バイトのユーザー ID。

**SMFPS\_PDN**

8 バイトのオペレーティング・システム製品レベル。

**CICS データ・セクション**

このセクションには、可変数の CICS ジャーナル・レコードが含まれます。

各レコードは汎用ログ・レコード・ヘッダーで始まり、その後にユーザー・ヘッダーが続きます。ユーザー・ヘッダーの後に呼び出し元データが続きます。

これが、CICS を初期設定した後にジャーナルに書き込まれる最初のレコードである場合、レコードは汎用ログ・レコード・ヘッダーと、その後の実行開始レコードで構成されます。それ以降のレコードの形式は、前述のとおりです。

## 第 5 章 CICS 互換インターフェースの開発

データ・セットを動的に割り振りおよび割り振り解除するために使用する動的割り振りサンプル・アプリケーション・プログラムをカスタマイズする方法は多数あります。

### 動的割り振りプログラムの概要

動的割り振り (DYNALLOC) のサンプル・アプリケーション・プログラムは、CICS 端末オペレーターが DYNALLOC (SVC 99) のほとんどの機能を使用できるようにします。

DYNALLOC の機能については、[z/OS MVS Programming: Authorized Assembler Services Reference \(Volume 1\)](#)で説明されています。許可プログラム機能 (APF) による許可が必要な機能はサポートされていません。

このアプリケーションは、DFH99 というコマンド・レベルのアセンブラー言語プログラム 1 つで構成されています。DFH99 はトランザクション ADYN により開始されます。ソース・コードは、CICSTS56.CICS.SDFHSAMP にあります。

端末オペレーターは、DYNALLOC 機能を使用して、CICS が開いたり閉じたりするデータ・セット (区画外一時データ・セット、ジャーナル、ダンプ、トレース・データ・セットなど) を動的に割り振り/割り振り解除できます。

ファイル制御で管理されるファイルに関連付けるデータ・セットの割り振り/割り振り解除には、この動的割り振りプログラムを使用しないでください。代わりに、CICS の一部である動的割り振りと割り振り解除の機能を使用してください。CICS 始動中にファイルが割り振られなかった場合、リソース定義内にそのファイルに関する十分な情報があれば、そのファイルを開く直前に CICS による動的割り振りが行われます。必要な情報は、データ・セット名とファイルの属性指定です。この情報を設定するには、**CEMT SET FILE** マスター端末トランザクション ([CEMT SET FILE](#) を参照) を使用するか、または **EXEC CICS INQUIRE FILE** コマンドと **EXEC CICS SET FILE** コマンドを使用します。これらのコマンドは、追加の照会機能や制御機能を備えています ([SET FILE](#) を参照)。

動的割り振りサンプル・プログラムを効果的に使用するには、端末オペレーターが MVS ジョブ制御言語か **TSO ALLOCATE** コマンドと **FREE** コマンドについて理解する必要があります。DYNALLOC 機能が返すエラー・コードや理由コードなどの詳細については、[z/OS MVS Programming: Authorized Assembler Services Reference \(Volume 1\)](#)を参照してください。

サンプル・プログラムは、3270 表示画面端末を使用し、画面サイズに合うようにフォーマットを調整します。BMS は不要です。このプログラムは、サポートされる機能をインストール環境の規格に合わせて簡単に変更できるように設計されています。

### プログラム定義とトランザクション定義のインストール

動的割り振りサンプル・プログラムのトランザクション定義とプログラム定義は、サンプル・ユーティリティ・リソース定義グループ DFH\$UTIL 内に用意されています。

これらの定義は、以下の CEDA コマンドを使用してインストールされます。

```
CEDA INSTALL GROUP(DFH$UTIL)
```

注:

1. DFH99 は EXECKEY(CICS) を指定して定義する必要があります。
2. サンプル・プログラムに変更を加える場合は、ADYN トランザクションを使用する前に DFH99BLD プロシージャを実行しなければなりません。

## 動的割り振りプログラム: 端末操作

端末でトランザクション ADYN を入力すると、フォーマットが調整された画面がオペレーターに表示されます。この表示画面の上部でコマンドを入力し、下部でプログラムからのメッセージを受け取ります。

オペレーターは、TSO に似た構文でコマンドを入力します。例えば、以下のようになります。

```
verb {keyword[(value...)]}...
```

入力後、ENTER キーを押します。プログラムは、構文が正しいかコマンドを検査し、DYNALLOC パラメーター・リストを作成し、重大なエラーが検出されなければ DYNALLOC SVC を発行します。その後、メッセージが表示されて構文エラーが診断され、DYNALLOC 戻りコードが示され、DYNALLOC 情報取得機能から返された値が表示されます。コマンドは表示画面上に残るので、端末の編集機能を使用して訂正してから再入力したり、別のコマンドを入力したりできます。

メッセージが多すぎて画面のメッセージ領域に収まらない場合、表示できないメッセージはキューに入れられ、既に画面上に出力されたメッセージは、出力すべきメッセージがまだあることを示すために高輝度で表示されます。オペレーターは表示されたエラーを訂正し、コマンドを再入力して追加検査を行うことができます。すると、キューに入れられているメッセージは再生成されます (ある場合)。

ヌル・コマンドを入力するとこのプログラムは終了します。ヌル・コマンドを入力するには、ERASE INPUT キーを押した後に ENTER キーを押します。PA キー 1 と 2 はプログラムでは無視されます。CLEAR キーを押すと、最後に入力したコマンドが再表示されます。PF キーを押すことは ENTER を押すことと同じです。

## 動的割り振りプログラムのヘルプ機能の使用

このプログラムには、プログラムのキーワード・テーブルによって駆動される限られた「ヘルプ」機能が組み込まれています。

「?」に対する応答として、verb キーワードが表示されます。「verb?」に対する応答として、その verb のオペランド・キーワードがすべて表示されます。「verb operand(?)」とすると、そのオペランドの予期される値についての簡単な説明が表示されます。「?」を含むコマンドを入力した場合、DYNALLOC SVC は発行されません。「?」が認識されるのは、上記の位置にある場合だけです。コマンドの残りの部分は無視されます。

## 動的割り振りプログラム: 値

値は以下のように分類されます。

### キーワード値

キーワードの中には、キーワード値を指定しなければならないものもあります。例えば、STATUS キーワードの場合は、SHR、NEW、MOD、または OLD (いずれも省略形が可能) というキーワード値を指定できます。

### キー文字のストリング

値として、任意の順序の文字のストリングを使用できます。指定された文字の組み合わせが意味を持つかどうかをプログラムは検査しません。例えば、RECFM キーワードの場合、A、B、D、F、G、M、R、S、T、U、V の中から選択した文字のストリングを値にすることができます。

### 戻り値

端末オペレーターが値を指定すべきではありません。このキーワードでは、DYNALLOC 情報検索機能によって値が戻される必要があるためです。以降は、戻される値の種類についての説明です。値は通常、オペレーターが入力する際の形式になりますが、16 進数ストリングの場合もあります。

### 不許可

値を必要としないキーワードもあります。この場合は、値を指定してはなりません。

### Required

コーディングされるキーワードが値を必要とすると指定されている場合は、値を指定する必要があります。

### オプション

キーワードの中には、値の指定がオプションになっているものもあります。

## 単一

キーワードの中には、指定できる値が1つだけのものもあります。

## 複数

キーワードの中には、複数の値を指定できるものもあります。(DYNALLOC が複数の値を必要とする場合もありますが、この動的割り振りサンプル・プログラムはそれを強制していません。)

## 文字ストリング

このタイプの値には任意の文字を使用できますが、ほとんどの場合、従うべき追加の規則があります (例えば DSNAME キーワードの場合)。

## 数値ストリング

このタイプの値には、数字のみが許可されます (例えば EXPDT キーワードの場合)。

## 最大長と最小長

文字値と数値の場合、値の最大長と最小長がプログラムによって検査されます。固定長ストリングの場合、これらの値は同じです。値は指定されたとおりに DYNALLOC に渡されます。

## n バイト・バイナリーに変換可能

指定されたバイト数のバイナリーで表せる大きさの数値を必要とします。大きすぎる値は、許容最大幅になるように切り捨てられます。

動的割り振りサンプル・プログラムは、負の数をサポートしていません。オペランド・キーワードのクロスチェックは行われません。このタイプのエラーが発生した場合は通常、DYNALLOC は 03xx 形式のエラー・コードを戻します。

## キーワードの省略規則

キーワードは短縮することができます。コマンドの単語は、次の場合にキーワードと一致します。

- スペルが同じである。
- 最初の文字が同じであり、単語の残りの文字がキーワードの場合と同じ順序で現れる。

あいまいさが発生する場合、プログラムはあいまいさを診断し、可能性のあるキーワードをリストします。

## システム・プログラミングに関する考慮事項

キーワードのスペルはプログラムのテーブル DFH99T で定義されています。これは、プログラムと共にリンク・エディットされます。可能な場合には、これらに対応するジョブ制御または TSO キーワードと同じになります。DFH99T のソース・コードのコメントでは、システム・プログラマーが以下を行う方法が説明されています。

- キーワードのスペルを変更する
- キーワードの代わりのスペルを定義する
- 動詞の機能をサブセットに分割する
- サブセット機能で新しい動詞を追加する
- SVC で使用可能になるように新しいオペランドを追加する

CICSTS56.CICS.SDFHINST のメンバー DFH99BLD は、プログラムをビルドするために使用されるジョブ・ストリームです。プログラムの一部が変更された場合、その部分を再アセンブルし、プログラムを再びリンク・エディットします。

MVS によって提供されるマクロ IEFZB4D0 (DYNALLOC パラメーター・リスト構造) および IEFZB4D2 (シンボリック・キーと同等) は、動的割り振りプログラムとそのキーワード・テーブルで使用されます。テーブルの各キーワードの意味は、シンボリック名で定義されます。これは、マクロ IEFZB4D0 または IEFZB4D2 のいずれかによって定義されます。そのマニュアルで示されるコマンド・キーワードの定義は、他のソースからのものよりも優先する必要があります。MVS マニュアルの相互参照として使用するためにコマンド・キーワードとそのシンボリック値のリストを取得するには、DFH99T をオプション SYSPARM(LIST) と共にアセンブルし、それによって生成されるオブジェクト・コードを出力します。テーブルが変更される場合、アセンブリーを繰り返して新しいリストを入手してください。

## DYNALLOC 要求が出されたときの制御の流れ

DYNALLOC 要求が出されると、メインプログラム DFH99M は、一連の下位プログラムを呼び出して要求を処理します。

通常の呼び出しにおける流れは、以下のとおりです。メインプログラム DFH99M が CICS から制御を受け取り、初期化を行います。この初期化には、画面サイズの決定、入力バッファと出力バッファの各セクションの割り振り、最初のメッセージの発行が含まれます。次に、端末から入力コマンドを取得する DFH99GI を呼び出します。戻りでコマンドがヌルだった場合、メインプログラムは終了して最終メッセージを発行します。

取得したコマンドの開始アドレスと終了アドレスが、グローバル通信域 COMM に保管されます。メインプログラムはトークン化テキストのためのストレージを割り振り、DFH99TK を呼び出してコマンドをトークン化します。この段階でエラーが検出されると、コマンドのこれ以上の分析はバイパスされます。

トークン化が成功すると、メインプログラムは、verb キーワードを分析する DFH99FP を呼び出します。DFH99FP は、テーブル DFH99T で verb キーワードを検索する DFH99LK を呼び出します。省略形が可能な場合、DFH99LK は DFH99MT を呼び出します。一致する verb が見つかり、DFH99FP はテーブルのオペランド・セクションのアドレスを COMM に書き込み、機能コードを DYNALLOC 要求ブロックに書き込みます。

ここでメインプログラムは、オペランド・キーワードを処理する DFH99KO を呼び出します。DFH99LK を呼び出すことによってテーブルで各キーワードが順に検索され、キーワードのコード化値がテーブル内の属性と照合されます。DFH99KO は次に、適切なコードを使用してテキスト・ユニットを開始し、値が持つべき属性に応じて変換ルーチンを呼び出します。

文字ストリングと数値ストリングの場合は、DFH99CC が呼び出されます。このプログラムはストリングの妥当性検査を行い、その長さと値をテキスト・ユニットに書き込みます。

バイナリー変数の場合は、DFH99BC が呼び出されます。このプログラムは値の妥当性検査を行い、必要な長さのバイナリーに値を変換して、その長さと値をテキスト・ユニットに書き込みます。

キーワード値の場合は、DFH99KC が呼び出されます。このプログラムは、DFH99LK を使用してキーワード・テーブルの記述部分で値を検索し、コード化等価値とその長さをテキスト・ユニットに書き込みます。

戻り値を指定しているキーワードが見つかり、DFH99KO は COMM に固定されている戻り値チェーンにエントリーを作成します。このエントリーは、DFH99T でのキーワード・エントリー、値が戻されるテキスト・ユニット、および次のエントリーをアドレッシングします。この場合、変換ルーチンはまだ呼び出されたままですが、この変換ルーチンはテキスト・ユニットでストレージのみを予約し、長さを最大値に、値をゼロにそれぞれ設定します。

すべてのオペランド・キーワードが処理されると、DFH99KO はメインプログラムに戻り、メインプログラムは DYNALLOC 要求を発行するための DFH99DY を呼び出します。

DFH99DY はパラメーター・リストの残りの部分をセットアップし、非常に重大なエラーが検出されなければ、DYNALLOC SVC を発行するためのサブタスクが接続されます。次に、サブタスク終了 ECB に対して WAIT EVENT が発行されます。サブタスクが終了し、CICS がプログラムを再度ディスパッチすると、サブタスク ECB からの DYNALLOC 戻りコード、および DYNALLOC 要求ブロックからのエラー・コードと理由コードがキャプチャーされ、これらの値を端末に表示するためのメッセージが発行されます。

次に DFH99DY はメインプログラムに戻り、メインプログラムは戻り値を処理するための DFH99RP を呼び出します。DFH99RP は戻り値チェーンをスキャンし、テキスト・ユニットで見つかったキーワードと値を含むメッセージをエレメントごとに発行します。戻り値がキーワード値に対応していれば、記述で値を検索してメッセージを発行するための DFH99KR が呼び出されます。

コマンドの処理はこれで完了です。メインプログラムは次のコマンドのために再初期化され、DFH99GI を呼び出すポイントまでループバックします。

さまざまな場所でマクロを使用してメッセージが発行されます。マクロ展開は DFH99MP の呼び出しで終了します。このプログラムは、新しいメッセージごとに新しい行が始まるようにし、メッセージ・エディターの DFH99ML を呼び出します。メッセージ・エディターへの入力トークンのリストです。各トークンが順にピックアップされて表示可能テキストに変換されます。テキストの部分ごとに DFH99TX が呼び出されます。このプログラムは、出力バッファにテキストを挿入し、必要であれば新しい行を開始します。これにより、単語が 2 行に分かれることがなくなります。



コマンドの処理の終わりに、メインプログラムはパラメーターなしで DFH99MP を呼び出します。これにより、出力バッファが端末に送られ、初期化されて空になります。



## 第 6 章 ユーザー作成プログラムによるリソース定義操作のカスタマイズ

CSD でのリソース定義に関する操作に使用される CEDA トランザクション・プログラムおよび DFHCSDUP ユーティリティ・プログラムの動作をカスタマイズできます。

- プログラム DFHEDAP にリンクすることにより、アプリケーション・プログラムから RDO 機能呼び出すことができます。
- DFHCSDUP からユーザー・プログラムを呼び出すことも、ユーザー・プログラムから DFHCSDUP を呼び出すこともできます。
- プログラムの処理内のキーポイントで出口ルーチンに制御を渡すことにより、DFHCSDUP の動作を変更できます。

### ユーザー作成プログラムに関する一般注意

次のコメントは、本書の第 8 部で取り上げているすべてのユーザー作成プログラムに適用されます。

- ユーザー作成プログラムからの戻りでは、CICS は常に 1 次スペース変換モードで制御を受け取らなければなりません。このとき、すべてのアクセス・レジスターの元の内容が復元されていて、すべての汎用レジスター (戻りコードおよびリンケージ情報のためのレジスターを除く) が復元されている必要があります。

変換モードについては、[z/Architecture 解説書](#)を参照してください。

## CEDA とのプログラマブル・インターフェースの使用

リソース定義オンライン (RDO) トランザクションである CEDA は、RDO が提供する機能をアプリケーション・プログラムから呼び出すために使用できるプログラマブル・インターフェースを備えています。

オフライン・ユーティリティ・プログラム DFHCSDUP および EXEC CICS CSD コマンドは、CSD ファイルを調査および修正するための方法として優先的に使用されます。DFHCSDUP は、CSD ファイルを一括で更新するときに使用できます。バッチ・モードまたは TSO の下で実行中のユーザー・プログラムから、DFHCSDUP を呼び出すことができます。

CEDA とのプログラマブル・インターフェースを呼び出すには、次のコマンドを使用します。

```
EXEC CICS LINK PROGRAM('DFHEDAP')  
              COMMAREA(cedaparm)
```

ここで、DFHEDAP は RDO プログラムのエントリー・ポイントの名前で、*cedaparm* は次の 5 つの 31 ビット・アドレス (それぞれ 1 つのフルワードに収まる) からなるパラメーター・リストのユーザー定義名です。

1. ソース形式の RDO コマンドが格納されているフィールドのアドレス。
2. このコマンドの長さを指定するハーフワードのバイナリー・フィールドのアドレス。入力コマンドの最大長は 1022 バイトです。
3. 次のように定義される、1 バイトの標識フィールドのアドレス。

**X'80'**

出力を呼び出し元に返さず、端末に表示します。

**X'00'**

出力を端末に表示しません。

4. DFHEDAP による出力の配置先となるフィールドのアドレス。
5. アプリケーションで処理可能な出力の最大長を指定するハーフワードのバイナリー・フィールドのアドレス。

アドレス 3 の標識が X'80' である場合、出力は端末に表示されます。この場合、端末の画面に表示される出力への応答として、CEDA コマンドをいくつでも入力できます。PF3 を押すと、制御はアプリケーション・プログラムに返されます。

しかし、標識が X'00' (出力は端末に表示されない) である場合、最初のアドレスに指定されている RDO コマンドの処理が終わるとすぐに、DFHEDAP は制御をアプリケーション・プログラムに返します。同時に、DFHEDAP は出力を 1 つまたは 2 つの連結された構造化フィールドとして返します。1 つの要求の出力は、変換段階で生成された 1 つのフィールドと、実行段階で生成されたゼロ個または 1 つのフィールドで構成されます。各フィールドの形式は次のとおりです。

- フィールドの包括的な長さを格納するバイナリー・ハーフワード。
- 作成されるメッセージの数を格納するバイナリー・ハーフワード。
- メッセージ最大重大度を格納するバイナリー・ハーフワード。'0' と '4' の場合は実行を継続し、'8' と '12' の場合は実行を継続しません。
- 以下を格納する可変長データ。
  - 変換段階の場合: 診断メッセージ (存在する場合)
  - 実行段階の場合: 通常、CEDA 画面に表示されるデータ (メッセージを含む)。各行は改行 (NL) 文字で始まるか、それ以外の場合はブランクと大文字の英数字が入ります。

このデータのフォーマットはリリースごとに変化する可能性があります。CEDA によって表示される内容と同じです。(このデータの分析は通常は必要ありません。一般的にプログラムにとって重要なのは、コマンドが正常に実行されたかどうかの情報のみです。) ユーザーが指定した最大長より出力の合計が長い場合、出力は切り捨てられます。

注:

1. EXEC CICS START コマンドがアプリケーション・プログラムから CEDA を開始しようとしても、失敗するはずですが、これは、CEDA が最初に行うアクションは関連した端末からの入力を要求する処理であるのに対して、自動で開始されるトランザクションは最初に端末にデータを送信する必要があるからです。  
同様の理由で、EXEC CICS START コマンドが CECI の下で CEDA を開始しようとしても失敗します。
2. CEDAPARM パラメーター・リストのアドレス 1 に渡される RDO コマンドは、有効でなければなりません。(例えば、プログラマブル・インターフェースを使用している場合、PROGRAM を PRORGAM とミススペルをしても自動修正されません。)
3. 本リリースの CICS で廃止されたものの旧リリースとの互換性を保つために維持されている CEDA キーワードの値は、プログラマブル・インターフェースを使用して変更することはできません。つまり、このインターフェースは互換性モードをサポートしていません。
4. CEDA は処理の一環として各種の同期点を発行します。そのため、プログラムが DFHEDAP にリンクされると、トランザクションの現在の作業単位 (UOW) は完了します。そのため、VSAM データ・セットに対する参照操作で未処理のものがある場合などには、問題が発生する可能性があります。

## DTP 環境での DFHEDAP の使用

LINK DFHEDAP 機能は、単一環境で使用するためのものです。分散トランザクション・プログラミング (DTP) 環境ではサポートされません。そのような環境で使用すると、異常終了する可能性があります。

DTP 環境の場合、CICS は、複数のセッションにわたる SYNCPOINT および SYNCPOINT ROLLBACK 要求を他のシステムに伝搬させようとする可能性があります。これらの要求は、LINK DFHEDAP を使用して呼び出される CEDA モジュールによって発行されます。SYNCPOINT ROLLBACK の発行があるなら、LU6.1 リンクを所有する DTP 環境では LINK DFHEDAP を使用できないことになります。

一般に、セッションは SYNCPOINT を開始するために SEND 状態である必要がありますが、LINK DFHEDAP コマンドが発行されると、セッションの SEND 状態が維持されなくなる可能性があります。有効なコマンドと状態については、[分散トランザクション処理の概要](#)を参照してください。

LINK DFHEDAP によって呼び出されるコードにより、コマンドの順序が正しくなくなる可能性があります。例えば、DFHEDAP によって呼び出されたコードによって、セッションが SEND 状態にある (そして SYNCPOINT をまだ発行していない) バックエンド・アプリケーション・プログラムから SYNCPOINT ROLLBACK が発行される場合、そのセッションは RECEIVE 状態となります。それから、DFHEDAP によ

て呼び出されたコードにより、SYNCPOINT が発行されると、異常終了が発生します。SYNCPOINT 要求を発行する DTP アプリケーションのすべてでこれを回避できるのは、それらのアプリケーションが LINK DFHEDAP コマンドを発行する前に、(それらのセッションのすべてで) SEND 状態となっている場合です。

DTP アプリケーション・プログラムが、フロントエンド 1 つとバックエンド 1 つからなるペアよりも多い場合は、LINK DFHEDAP の使用を試みないでください。

単純な DTP 環境 (フロントエンド 1 つとバックエンド 1 つ) 内で LINK DFHEDAP を使用するための通則は、LINK DFHEDAP コマンドが発行されるときに DTP 環境内のすべてのセッションが SEND 状態であること、および DFHEDAP コードによって発行される SYNCPOINT ROLLBACK が発生したら SEND 状態に戻す必要があることです。

## システム定義ユーティリティ・プログラム (DFHCSDUP) 用ユーザー・プログラム

CICS システム定義ユーティリティ・プログラム (DFHCSDUP) を変更または拡張する独自のプログラムを作成することができます。

DFHCSDUP の概要については、[システム定義ファイル・ユーティリティ・プログラム \(DFHCSDUP\)](#) を参照してください。

### DFHCSDUP からのユーザー・プログラムの呼び出し

3 つのサンプル・プログラムのいずれかを EXTRACT 処理中に呼び出すことができます。

抽出コマンドについて詳しくは、[DFHSTUP 抽出統計レポート機能](#)を参照してください。

#### EXTRACT の処理の際に呼び出されるプログラムの作成

DFHCSDUP LIST コマンドを使用すると、ユーザーが指定する入力パラメーターによってのみ変化する CSD ファイルの現在の状況に関するレポートが作成されます。もう 1 つの DFHCSDUP コマンドである EXTRACT を使用すると、選択した CSD データが不定形式でユーザー・プログラムに渡されます。その後、ユーザー・プログラムは、ローカル要件を満たす CSD データのレポートを作成することができます。

例えば、関連した定義 (TERMINAL、TYPETERM など) を相互参照したり、セキュリティ・キーや処理優先順位などの属性値によってデータをソートしたりすることができます。ユーザー・プログラムは、要求されたリソース属性を、SQL、DB2、または Data Extract プログラム製品などのデータベース製品への入力として使用されるデータ・セットに書き込むこともできます。

ユーザー・プログラムは RMODE(24) でリンクされる必要があります。これは、24 ビットの 1 次スペース変換モードで制御を受け取ります。(変換モードの詳細については、[z/Architecture 解説書](#)を参照してください) アクセス・レジスターの内容は予測できません。プログラムは 24 ビットの 1 次スペース変換モードで制御を返す必要があり、(汎用レジスターを復元することに加えて) 変更したすべてのアクセス・レジスターを復元する必要があります。

EXTRACT の処理中に DFHCSDUP から呼び出されるサンプル・プログラムが 3 つあります。サンプル・プログラム、およびそれを独自のバージョンで置き換える方法については、[323 ページの『サンプル EXTRACT プログラム』](#)で説明されています。

#### ユーザー・プログラムが呼び出された場合

DFHCSDUP による EXTRACT コマンドの処理中に、ユーザー・プログラムを 9 つの異なる時点で呼び出すことができます。ただし、それらのすべての時点でユーザー・プログラムが呼び出されるのは、EXTRACT コマンドに LIST と OBJECTS の両方が指定されている場合に限られます。呼び出しの時点は以下のとおりです。

1. EXTRACT 処理の開始時。これは、ファイル・オープンやストレージ獲得などのアクティビティを可能にするためです。
2. LIST 処理の開始時。ただし、EXTRACT コマンドに LIST 値を指定した場合に限られます。
3. EXTRACT コマンドで処理される各グループの開始時。
4. 処理される各オブジェクト (つまり、リソース・タイプ TERMINAL、PROGRAM など) の開始時。これは、オブジェクトまたはグループに基づいた選択を可能にするためです。

注: EXTRACT コマンドに LIST を指定しても、OBJECTS を指定しなかった場合、この呼び出しは生じません。

5. 抽出されたオブジェクト内の各キーワード (属性) に応じて。ただし、EXTRACT コマンドに OBJECTS を指定した場合に限られます。これは、相互参照で必要になることがある詳細な処理を可能にするためです。
6. 各オブジェクトの終了時。つまり、オブジェクト内のすべてのキーワードが処理された時点。これは、詳細な項目から構築されたデータの処理を可能にするためで、オブジェクトごとに 1 回ずつ生じます。
7. 各グループの終了時。これは、累積データの処理を可能にするためです。
8. LIST 処理の終了時。EXTRACT コマンドに LIST 値を指定した場合に生じます。
9. EXTRACT 処理の完了時。これは、ファイルのクローズやストレージの解放などを可能にするためです。

### DFHCSDUP からユーザー・プログラムに渡されるパラメーター

ユーザー・プログラムを呼び出すたびに、DFHCSDUP は汎用レジスター 1 によってアドレス指定されたパラメーター・リストを渡します。このパラメーター・リストは、フィールドをアドレス指定する一連のフルワードで構成されます (フィールドについて、詳しくは後述)。パラメーター・リストに設定されるアドレスは、EXTRACT 処理が到達した地点に応じて異なります。

パラメーター・リストには、次のフィールドが含まれています。

#### Function Type Ptr (機能タイプ・パラメーター)

EXTRACT 処理が到達した地点を定義するコードを含んだハーフワード・フィールドのアドレス。

機能コードは、以下のとおりです。

- 0 初期呼び出し
- 2 リスト開始呼び出し
- 4 グループ開始呼び出し
- 6 オブジェクト開始呼び出し
- 8 キーワード詳細呼び出し
- 10 オブジェクト終了呼び出し
- 12 グループ終了呼び出し
- 14 リスト終了呼び出し
- 16 最終呼び出し

#### Workarea Ptr (作業域パラメーター)

これは、ユーザーが獲得した作業域のアドレスを保管するためにユーザー・アプリケーションによって使用される、フルワードのアドレスを入れたフィールドのアドレスです。

#### Back translated command Ptr (逆変換済みコマンド・パラメーター)

処理中の EXTRACT コマンドが入るストレージの 75 バイト領域のアドレスが入るフルワードのアドレス。

#### List name Ptr (リスト名パラメーター)

現在のオブジェクトの取得元の RDO リストを識別する 8 バイト・フィールドのアドレス。この値は、「リスト開始」および「リスト終了」呼び出し時にのみ設定されます。

#### Group name Ptr (グループ名パラメーター)

現在のオブジェクトの取得元の RDO グループを識別する 8 バイト・フィールドのアドレス。この値は、「グループ開始」、「グループ終了」、「オブジェクト開始」、「オブジェクト終了」、および「キーワード」の呼び出し時に設定されます。

#### Object type Ptr (オブジェクト・タイプ・パラメーター)

オブジェクトのタイプ (TRANSACTION、PROGRAM など) を識別する 12 バイト・フィールドのアドレスで、「オブジェクト開始」、「オブジェクト終了」、および「キーワード」の呼び出し時にのみ設定されます。

#### Object name Ptr (オブジェクト名パラメーター)

オブジェクトの名前が入る 8 バイト・フィールドのアドレスで、「オブジェクト開始」、「オブジェクト終了」、および「キーワード」の呼び出し時にのみ設定されます。



**Keyword name Ptr (キーワード名パラメーター)**

処理中のキーワードの名前が入る 12 バイト・フィールドのアドレスで、「キーワード」呼び出し時のみ設定されます。

**Keyword length Ptr (キーワード長パラメーター)**

キーワードに関連付けられている値の長さが入る ハーフワード・フィールドのアドレスで、「キーワード」呼び出し時のみ設定されます。

**Keyword Value Ptr (キーワード値パラメーター)**

キーワードに関連付けられている値が入る ストレージ域のアドレスで、「キーワード」呼び出し時のみ設定されます。

注: ポインター値で設定されていないフィールドには、ヌル値が入ります。

**サンプル EXTRACT プログラム**

DFHCSDUP EXTRACT 処理中に 3 つの CICS 提供サンプル・プログラムを呼び出すことができます。

これらのうちの 2 つは COBOL、PL/I、およびアセンブラ言語で提供されており、もう 1 つは COBOL のみ提供されています。

表 27. DFHCSDUP ユーティリティ・プログラム用のサンプル EXTRACT ユーザー・プログラム		
プログラム名	言語	説明
DFH\$CRFA DFH0CRFC DFH\$CRFP	アセンブラ COBOL PL/I	EXTRACT コマンドに指定したグループまたはリストに定義されているリソース定義の相互参照リストを作成します。
DFH\$FORA DFH0FORC DFH\$FORP	アセンブラ COBOL PL/I	EXTRACT コマンドに指定したリソース定義のグループまたはリストを、Db2 テーブル・ロード・ユーティリティに適したフォーマットに設定します。
DFH0CBDC	COBOL	EXTRACT コマンドに指定したリソース定義のリストまたはグループを、抽出されたリソースのバックアップ・コピーとして使用するのに適した、DEFINE コマンドの形式で書き出します。

サンプル・プログラムは、提供されたまま使用することも、それをモデルにして独自のプログラムを作成することもできます。

CSD 相互参照プログラムのアセンブラ言語バージョン (DFH\$CRFA) と COBOL バージョン (DFH0CRFC) は、CICSTS56.CICS.SDFHLOAD の実行可能形式で提供されています。PL/I バージョン (DFH\$CRFP) はソース形式のみで提供されています。

Db2 フォーマット設定プログラムのアセンブラ言語バージョン (DFH\$FORA) と COBOL バージョン (DFH0FORC) は、CICSTS56.CICS.SDFHLOAD の実行可能形式で提供されています。PL/I バージョン (DFH\$FORP) はソース形式のみで提供されています。

CSD バックアップ・ユーティリティ・プログラム (DFH0CBDC) は、CICSTS56.CICS.SDFHLOAD の実行可能形式で提供されています。

すべてのサンプル・プログラムのすべてのバージョンのソース・ステートメントは、CICSTS56.CICS.SDFHSAMP で提供されています。

CICS 提供サンプル Db2 フォーマット設定プログラム (DFH\$FORx) は、DFHCSDUP ユーティリティ・プログラムに CSD 互換性オプション (COMPAT) が指定されている場合には使用できません。CSD 相互参照リスト作成プログラムおよび CSD バックアップ・ユーティリティ・プログラムからの出力は、互換性オプションが指定されているかどうかによって異なります。互換性オプションが指定されている場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 より前のリリースで使用されている廃止済みキーワードが出力に含まれますが、このオプションが指定されていない場合は、CICS Transaction Server for z/OS, バージョン 5 リリース 6 で使用されているキーワードのみが出力されます。

すべてのサンプル抽出ユーザー・プログラムは、定義シグニチャー・フィールドをサポートします。

サンプル・プログラムを使用するときは、DFHCSDUP EXTRACT コマンドに OBJECTS キーワードを指定する必要があることに注意してください。

サンプル・プログラムの出力データ定義名 (ddname) は、以下のとおりです。

#### **CRFOUT**

CSD 相互参照プログラム

#### **FOROUT**

Db2 フォーマット設定プログラム

#### **CBDOUT**

CSD バックアップ・ユーティリティ・プログラム

#### **CSD 相互参照プログラム**

CICS 提供のサンプル CSD 相互参照プログラムを使用して、EXTRACT コマンドで指定されるグループまたはリストで定義されるリソース定義の相互参照リストを生成します。アセンブラーを使用している場合は相互参照プログラムの DFH\$CRFA バージョンを実行し、COBOL を使用している場合は DFH0CRFC バージョンを実行し、PL/I を使用している場合は DFH\$CRFP バージョンを実行します。

CICS 提供のサンプル CSD 相互参照プログラムは、CSD のオブジェクトおよびキーワードの相互参照リストを生成します。EXTRACT コマンドによって収集されるデータはサンプル・プログラムに渡され、そのプログラムで相互参照テーブルに保管されます。このサンプル・プログラムの最終呼び出しの際に、テーブルの内容が照合シーケンスで出力されます。

プログラムは、次の形式で EXTRACT コマンドに対して実行する必要があります。

```
EXTRACT GROUP(group name) OBJECTS USERPROGRAM(program-name)
```

または、

```
EXTRACT LIST(list name) OBJECTS USERPROGRAM(program-name)
```

なお、サンプル・プログラムでは OBJECTS キーワードを指定する必要があります。

このプログラムに関してのみ、EXTRACT コマンドの他に、相互参照リストの対象とするオブジェクトとキーワードを順次データ・セットに定義する必要があります。データ・セットは、CRFINPT という DD 名を使用してサンプル・プログラムによって読み取られます。

CRFINPT は、80 バイトのレコードを含む順次ファイルです。各レコードには、相互参照される 1 つのオブジェクトまたはキーワードが含まれます。CEDA によって認識される有効なリソース・タイプまたは属性があればそれも相互参照できます。例えば、CRFINPT ファイルには以下の項目を含めることができます (1 行に 1 つ)。

PROGRAM  
TRANSACTION  
TYPETERM  
DSNAME

COBOL サンプル・プログラム DFH0CRFC を使用する際、CRFINPT ファイルには最大 10 個の項目を含めることができます。

ファイル内のレコードごとに、キーワードに割り当てられるさまざまな値の詳細を記述したレポートが生成されます。つまり、値がどこで定義され、どこで使われるかが報告されます。なお、44 文字を超えるキーワード値は切り捨てられます。

CRFINPT の DCB サブパラメーターは、DSORG=PS、RECFM=F、LRECL=80、および BLKSIZE=80 として定義する必要があります。

#### **Db2 フォーマット設定プログラム**

CICS 提供のサンプル Db2 フォーマット設定プログラムを使用して、Db2 表のロード・ユーティリティに適した形式に CSD データを編成します。アセンブラーを使用している場合 DFH\$FORA サンプル・プログラムを実行し、COBOL を使用している場合は DFH0FORC サンプル・プログラムを実行し、PL/I を使用している場合は DFH\$FORP サンプル・プログラムを実行します。

CICS 提供のサンプル Db2 フォーマット設定プログラムは、ロード・ユーティリティの入力で定義された列に対応する列にデータを編成します。リソースが選択されるたびに、レコードがこのプログラムの出力ファイルに書き込まれます。最初の 4 文字がリソース・タイプを識別します。

プログラムは、次の形式で EXTRACT コマンドに対して実行する必要があります。

```
EXTRACT GROUP(group name) OBJECTS USERPROGRAM(program-name)
```

または、

```
EXTRACT LIST(list name) OBJECTS USERPROGRAM(program-name)
```

なお、サンプル・プログラムでは OBJECTS キーワードを指定する必要があります。

### Db2 への CSD データの保管

データを Db2 データベースに保管する場合は、データのフォーマットを調整し、Db2 内に表を作成し、フォーマットを調整したデータをそれらの表に追加する必要があります。

### このタスクについて

DFHCSDUP による CSD データ出力を Db2 に保管する場合は、以下の手順を行います。

#### 手順

1. Db2 に表を作成します。

SDFHSAMP(DFH\$DB2T) に用意されているサンプル Db2 表定義を使用して、Db2 に表を作成します。

- a) DFH\$DB2T サンプルを更新します。

DFH\$DB2T サンプル・コードを更新して、<data base name>.<table space> の出現箇所を、使用環境に応じたデータベース名および表スペース名 (TESTDB.TESTDBTSP など) に置き換えます。

2. DFHCSDUP および Db2 フォーマット設定プログラムを実行します。

用意されているサンプル Db2 フォーマット設定プログラムを使用して、DFHCSDUP から得られた CSD データを、Db2 表ロード・ユーティリティに適したフォーマットに編成します。サンプルのフォーマット設定プログラムには、アセンブラ用の DFH\$FORA、COBOL 用の DFH0FORC、および PL/I 用の DFH\$FORP という 3 つのバージョンがあります。

FOROUT DD ステートメントは、レコード・サイズが 1536 のデータ・セットを指す必要があります。

3. フォーマットを調整したデータを、Db2 表に取り込みます。

SDFHSAMP(DFH\$SQLT) に用意されているサンプルの Db2 ロード・ロジックを使用して、Db2 内の表にデータを取り込みます。Db2 フォーマット設定プログラムの出力を書き込んだデータ・セットを指す、SYSREC という DD カードが必要です。これは、直前のステップで FOROUT DD で指定したデータ・セットと同じです。

- a) DFH\$SQLT サンプルを更新します。

DFH\$SQLT サンプル・コードを更新して、<owner> の出現箇所を、データベースの許可ユーザー ID に置き換えます。

### タスクの結果

Db2 に作成した表に、データが保管されました。

#### 例

データをフォーマットして Db2 に移動するために使用できる JCL コードのサンプルを以下に示します。

```
//DB2EXT JOB MSGCLASS=H,CLASS=A,NOTIFY=&SYSUID
//*-----
//*CSD EXTRACT
//*-----
//EXTRACT EXEC PGM=DFHCSDUP,REGION=2000K
//SYSPRINT DD SYSOUT=A
//*
//STEPLIB DD DSN=<cicsshlq>..SDFHLOAD,DISP=SHR
//DFHCSD DD DSN=TEST.RTSREG.CTS410.CICS660.CSD,DISP=SHR
//SYSOUT DD *
//FOROUT DD DSN=USER1.TEST.DB2.INPUT,DISP=SHR
//SYSIN DD *
EXTRACT GROUP(TESTGRP) USERPROGRAM(DFH0FORC) OBJECTS
/*
```

```

//
//DB2STG JOB 1,USER=TEST,CLASS=A,MSGCLASS=A,
//      MSGLEVEL=(1,1),REGION=7M,NOTIFY=&SYSUID
//*
//JOB LIB DD DSN=SYS2.DB2.V910.SDSNLOAD,DISP=SHR
//*-----
//* CREATE STORAGE GROUP/DATABASES/TABLESPACES
//*-----
//CREATDB EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
      DSN SYSTEM(DHP2)
      RUN PROGRAM(DSNTIAD) PLAN(DSNTIA91) -
        LIB('DSN910P2.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
      DROP DATABASE TESTDB;
      DROP STOGROUP TESTDBST;
      CREATE STOGROUP TESTDBST VOLUMES (SYSDA) VCAT DSN910P2;
      CREATE DATABASE TESTDB STOGROUP TESTDBST;
      COMMIT;
      CREATE TABLESPACE TESTDBTS IN TESTDB
      LOCKSIZE ROW;
      COMMIT;
//*
//
//*-----
//* CREATE TABLES AND INDEXES
//*-----
//CREATTAB EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//DBRMLIB DD DSN=DSN910P2.DBRMLIB.DATA,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
      DSN SYSTEM(DHP2)
      RUN PROGRAM(DSNTIAD) PLAN(DSNTIA91) -
        LIB('DSN910P2.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
//* INCLUDE CONTENTS OF SDFHSAMP(DFH$DB2T) HERE
//*e.g.
//*CREATE TABLE ATOMSERVICE
//* (ATOMSERVICE CHAR(8),
//* RDOGROUP CHAR(8),
//* DESCRIPTION CHAR(58),
//* ATOMTYPE CHAR(10),
//* STATUS CHAR(8),
//* CONFIGFILE CHAR(255)
//* RESOURCENAME CHAR(16),
//* RESOURCETYPE CHAR(7),
//* BINDFILE CHAR(255)
//* DEFINETIME CHAR(17),
//* CHANGETIME CHAR(17),
//* CHANGEUSRID CHAR(8),
//* CHANGEAGENT CHAR(8),
//* CHANGEAGREL CHAR(4))
//* IN TESTDB.TESTDBTSP;
//*
//*CREATE INDEX ATOMI ON ATOMSERVICE
//* (ATOMSERVICE ASC);
//*
//* COMMIT;
//* etc ...
//
//GRANTACC EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
      DSN SYSTEM(DHP2)
      RUN PROGRAM(DSNTIAD) PLAN(DSNTIA91) -
        LIB('DSN910P2.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN DD *
      GRANT DBADM ON DATABASE TESTDB TO PUBLIC;
      GRANT USE OF TABLESPACE TESTDB.TESTDBTS TO PUBLIC;
      GRANT ALL PRIVILEGES ON TABLE TESTDBTB TO PUBLIC;
//*
//
//*-----
//*LOAD DATA INTO TABLES
//*-----

```

```
//DB2TBL JOB CLASS=A,MSGCLASS=H,NOTIFY=&SYSUID,REGION=4096K,
//      USER=TEST
//*
//JOB LIB DD DSN=SYS2.DB2.V910.SDSNLOAD,DISP=SHR
//*
//*
//***** LOAD ITMNUMBR - TRANS WKLD *****
//*
//STEPITM EXEC PGM=DSNUTILB,PARM='DHP2'
//UTPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSUT1 DD DSN=ST.ITM02.SYSUT1,DISP=(MOD,DELETE,CATLG),
// UNIT=SYSDA,SPACE=(CYL,(10,5))
//SYSREC DD DSN=USER1.TEST.DB2.INPUT,DISP=SHR
//SORTOUT DD UNIT=SYSDA,SPACE=(CYL,(40,10),,ROUND)
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK04 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SYSIN DD * << INCLUDE CONTENTS OF SDFHSAMP(DFH$SQLT) HERE >>
//* e.g.
//*LOAD DATA
//*RESUME NO REPLACE
//*INTO TABLE ATOMSERVICE
//*WHEN (1:4) = 'ATOM'
//*(ATOMSERVICE POSITION (5:12) CHAR,
//* RDOGROUP POSITION (13:20) CHAR,
//* DESCRIPTION POSITION (21:78) CHAR,
//* ATOMTYPE POSITION (79:88) CHAR,
//* STATUS POSITION (89:96) CHAR,
//* CONFIGFILE POSITION (97:351) CHAR,
//* RESOURCENAME POSITION (352:367) CHAR,
//* RESOURCETYPE POSITION (368:374) CHAR,
//* BINDFILE POSITION (375:629) CHAR,
//* DEFINETIME POSITION (630:646) CHAR,
//* CHANGETIME POSITION (647:663) CHAR,
//* CHANGEUSRID POSITION (664:671) CHAR,
//* CHANGEAGENT POSITION (672:679) CHAR,
//* CHANGEAGREL POSITION (680:683) CHAR)
//*INTO TABLE BUNDLE
//*
//* etc...
//*
//
```

### CSD バックアップ・ユーティリティー・プログラム

CICS 提供のサンプル CSD バックアップ・ユーティリティー・プログラム DFH0CBDC を使用して、EXTRACT コマンドで指定されたリソース定義のリストまたはグループを、バックアップ・コピーとしての使用に適した DEFINE コマンドの形式で作成します。

CICS 提供のサンプル CSD バックアップ・ユーティリティー・プログラムは、DFHCSDUP DEFINE 制御ステートメントのファイルを生成します。このファイルは、次の目的で使用できます。

- ドキュメント CSD リソースを後で編集またはコメントする
- 他の CICS インストール済み環境に一部または全体を配布する
- DFHCSDUP を使用して、いずれかの CSD にリソース定義を再作成または追加する

プログラムは、次の形式で EXTRACT コマンドに対して実行する必要があります。

```
EXTRACT GROUP(group name) OBJECTS USERPROGRAM(program-name)
```

または、

```
EXTRACT LIST(list name) OBJECTS USERPROGRAM(program-name)
```

なお、サンプル・プログラムでは OBJECTS キーワードを指定する必要があります。

DFH0CBDC を使用する場合は、以下の点に注意してください。

- DFHCSDUP を呼び出すときに処理できるデータのセット数は、毎回 1 つだけです。2 つの EXTRACT コマンドが発行される場合、2 番目のデータ・セットは最初のデータ・セットを上書きします。

- DFHOCBDC によって生成されるファイルにおいて、カラム 1 の CICS 提供のリソースに関連した DEFINE ステートメントの先頭にアスタリスク (\*) が付きます。つまり、コメント化されます。リソースを CSD に定義するためにファイルを入力として使用する場合、これは重要な意味を持ちます。(CICS 提供の定義は、初期化されたときに自動的に生成されているので、既に CSD に存在しています。)
- (DEFINE ステートメントを復元するために) アスタリスクをカラム 1 から削除する場合、ブランクで上書きするのではなく、実際に削除してください。これにより、結果として生成されるコマンドの長さが 72 文字以下に収まるようになります。これより長いと、出力が DFHCSDUP によって戻されたときにエラーが発生します。

### EXTRACT プログラムのアセンブルおよびリンク・エディット

DFHCSDUP ユーザー・プログラムは、CICS アプリケーションではなくバッチ・プログラムとしてアセンブル(またはコンパイル)してリンク・エディットする必要があり、プログラムの作成に使用した言語に適した制御ステートメントをリンク・エディットする必要があります。

### このタスクについて

注: DFHCSDUP ユーザー・プログラムは変換しないでください。変換すると、予期しない結果になる可能性があります。

COBOL バージョンのサンプル・プログラムをコンパイルするときには、コンパイラ属性 RENT および NORES を指定する必要があります。<sup>3</sup>

プログラムをリンク・エディットするときには、次のリンク・エディット制御ステートメントを指定する必要があります。

- 入り口名を DFHEXTRA として定義する ENTRY ステートメント
- ユーザー・プログラムに組み込む必要のある CICS 提供スタブの INCLUDE ステートメント
- CICS 提供スタブのダミー CSECT 名を EXITEP からユーザー・プログラムの名前に変更する CHANGE ステートメント

COBOL バージョンのサンプル・プログラムをリンク・エディットするときには、RMODE(24) を指定する必要があります。

上記の要件については、以下のサンプル・ジョブ・ストリームで、言語(アセンブラー、COBOL、および PL/I) ごとにより詳しく説明しています。

### アセンブラー言語のバージョン

サンプル・ジョブは、DFHCSDUP ユーザー・プログラムのアセンブラー言語のバージョンに必要なリンク・エディット・ステートメントを示しています。

```
//DFHCRFA JOB (accounting information),CLASS=A,MSGCLASS=A,NOTIFY=userid
//* .
//* Assembler job step here
//* .
//LINK EXEC PGM=IEWL,PARM='XREF,LIST,LET'
//OBJLIB DD DSN=object.module.library,DISP=SHR
//SYSLIB DD DSN=CICSTS56.CICS.SDFHLOAD,DISP=SHR
//SYSLMOD DD DSN=user.library,DISP=SHR
//SYSUT1 DD UNIT=SYSDA,SPACE=(1024,(100,10))
//SYSPRINT DD SYSOUT=A
//SYSLIN DD *
ENTRY DFHEXTRA 1
CHANGE EXITEP(csectname) 2
INCLUDE SYSLIB(DFHEXAI) 3
INCLUDE OBJLIB(obj-name) 4
NAME progname(R) 5
```

図 98. DFHCSDUP ユーザー・プログラムのリンク・エディット制御ステートメント(アセンブラー言語)

### アセンブラー・ジョブについての注意事項:

<sup>3</sup> RENT コンパイラ属性を指定すると、サンプル・プログラムが B37 ('Data set size is smaller than output (データ・セット・サイズが出力より小さいです)') などの異常終了を受け取っても異常終了 C03 ('Data set was not closed properly (データ・セットが正しく閉じませんでした)') になりません。



1 項目名を DFHEXTRA として指定します。これは、CICS 提供のスタブ DFHEXAI の項目名です。(3 を参照。)

2 CICS 提供のスタブ DFHEXAI が、ダミー CSECT 名 (EXITEP) を使用するユーザー・プログラムへのリンクと一緒に生成されます。リンク・エディット CHANGE ステートメントを使用して、CSECT 名を EXITEP からユーザー・プログラムの CSECT の名前に変更してください。2 つの CICS 提供アセンブラ言語サンプル・プログラムでは、これらの名前は次のとおりです。

#### CREFCSD

相互参照リストのユーザー・プログラム DFH\$CRFA 内の CSECT 名。

#### FORMCSD

Db2 書式設定ユーザー・プログラム DFH\$FORA 内の CSECT 名。

3 DFHCSDUP EXTRACT コマンドで使用するために作成するアセンブラ言語ユーザー・プログラムに DFHEXAI を組み込みます。DFHEXAI は、DFHCULIS、DFHCSDUP 内のモジュール、およびユーザー・プログラムの間のインターフェース・スタブです。

4 obj-name は、アセンブルされたオブジェクト・モジュールを含むライブラリー・メンバーの名前です。

5 progname は、ロード・モジュールを呼び出すための希望の名前です。これは、EXTRACT コマンドの USERPROGRAM パラメーターで指定する名前です。

### 言語環境プログラムのバージョン

329 ページの図 99 のサンプル・ジョブは、言語環境プログラムに準拠する高水準言語で書かれた DFHCSDUP ユーザー・プログラムに必要なリンク・エディット・ステートメントを示しています。

```
//DFHCRFA JOB (accounting information),CLASS=A,MSGCLASS=A,NOTIFY=userid
//* .
//* Compile job step here
//* .
//LINK EXEC PGM=IEWL,PARM='XREF,LIST,LET'
//SYSLIB DD DSN=PP.ADLE370.OS39025.SCEELKED
//CICSLIB DD DSN=CICSTS56.CICS.CICS.SDFHLOAD,DISP=SHR
//OBJLIB DD DSN=object.module.library,DISP=SHR
//SYSLMOD DD DSN=user.library,DISP=SHR
//SYSUT1 DD UNIT=SYSDA,SPACE=(1024,(100,10))
//SYSPRINT DD SYSOUT=A
//SYSLIN DD *
ENTRY DFHEXTRA 1
CHANGE EXITEP(prof-id) 2
INCLUDE CICSLIB(DFHEXLE) 3
INCLUDE OBJLIB(obj-prog) 4
NAME progname(R) 5
```

図 99. DFHCSDUP ユーザー・プログラムのリンク・エディット制御ステートメント (言語環境プログラム)

#### 言語環境プログラム・ジョブについての注意事項:

1 項目名を DFHEXTRA として指定します。これは、CICS 提供のスタブ DFHEXLE の項目名です (3 を参照)。

2 CICS 提供のスタブ DFHEXLE が、ダミー CSECT 名 (EXITEP) を使用するユーザー・プログラムへのリンクと一緒に生成されます。リンク・エディット CHANGE ステートメントを使用して、CSECT 名を EXITEP からユーザー・プログラムの PROC ステートメントで指定した名前に変更してください。

3 DFHCSDUP EXTRACT コマンドで使用するために作成する LE 準拠のユーザー・プログラムに DFHEXLE を組み込みます。DFHEXLE は、DFHCULIS、DFHCSDUP のモジュール、および言語環境プログラムのユーザー・プログラムの間のインターフェース・スタブです。

4 obj-prog は、オブジェクト・プログラムの名前です。

5 progname は、希望するロード・モジュール名です。これは、EXTRACT コマンドの USERPROGRAM パラメーターで指定する名前です。

## ユーザー・プログラムからの DFHCSDUP の呼び出し

ユーザー・プログラムから DFHCSDUP を呼び出すことができます。それによってユーティリティとのインターフェースを柔軟に作成できます。

ユーザー・プログラムで適切な入力パラメーターを指定することにより、DFHCSDUP が 5 つの任意の出口点で制御を出口ルーチンに渡すように設定できます。これらの出口を使用して、DFHCSDUP にコマンドを渡す、DFHCSDUP 処理で生成されたメッセージに応答する、などの操作を実行できます。

ユーザー・プログラムは次の方法で実行できます。

- バッチ・モードで
- TSO の下で

注:

1. TSO 環境では、通常、端末オペレーターが ATTENTION 割り込みを使用していつでも処理に割り込むことができます。DFHCSDUP は、CSD ファイルの保全性を守るために、現在実行中のコマンドに関連した処理が完了するまでそのような割り込みに応答しません。処理が完了すると、DFHCSDUP はメッセージ番号'DFH5618'をメッセージ書き込み出口 (可能な場合、[335 ページの『メッセージ書き出し出口』](#)を参照) およびデフォルト出力ファイルに書き込みます。

AN ATTENTION INTERRUPT WAS  
REQUESTED DURING DFHCSDUP PROCESSING

メッセージ書き込み出口ルーチンで DFHCSDUP を終了させることも可能です。(ATTENTION 割り込み後にオペレーターに制御を返す場合は、メッセージ書き込みルーチンを設定する必要があります。)

2. 適切な権限が付与されている TSO オペレーターは、CEDA INSTALL トランザクションを使用して、事前に DFHCSDUP で定義されているリソースをインストールできます。

CICS 提供のサンプル・プログラム DFH\$CUS1 は、DFHCSDUP プログラムをユーザー・プログラムから起動する方法を例示しています。これは TSO/E オペレーティング・システムの下で実行されるコマンド・プロセッサ (CP) として作成されています。詳しくは、[336 ページの『サンプル・プログラム DFH\\$CUS1』](#)を参照してください。

### DFHCSDUP の入力パラメーター

プログラムが DFHCSDUP を呼び出すときに、レジスター 1 でアドレッシングされるパラメーター・リストを渡します。プログラムは、最大 5 個のパラメーターを渡すことができます。

#### OPTIONS

コンマで区切られた文字ストリングのリスト。この情報は、ここで渡さなければ JCL の EXEC ステートメントの PARM キーワードで渡す情報です。最大で次の 4 つのオプションを指定できます。

##### CSD({READWRITE|READONLY})

CSD への読み取り/書き込みアクセスが必要か、読み取り専用アクセスが必要かを指定します。

##### PAGESIZE(nnnn)

出力リストの 1 ページ当たりの行数を指定します。有効な値は 4 から 9999 です。デフォルト値は 60 です。

##### NOCOMPAT|COMPAT

互換モードで DFHCSDUP が呼び出されるかどうかを指定します。デフォルトでは、互換モードで呼び出されません。互換モードについて詳しくは、[異なる CICS リリース間での CSD の共用](#)を参照してください。

##### UPPERCASE

出力リストがすべて大文字で出力されるように指定します。デフォルトでは、大/小文字混合で出力されます。

#### DDNAMES

通常 DFHCSDUP によって使用されるデータ定義名 (DD 名) を置換するデータ定義名のリスト (指定された場合)。

## HDING

DFHCSDUP によってリストが生成された場合、そのリストの開始ページ番号。このパラメーターを使用して、今後の呼び出しが行われたときに論理的に番号が付けられたリストが生成されるようにします。このパラメーターが指定されない場合、開始ページ番号は 1 に設定されます。

ページ番号データの長さ (331 ページの図 100 のフィールド bb) は 0 または 4 である必要があります。ページ番号 (指定する場合) は、4 つの数字の EBCDIC 文字である必要があります。このフィールド (存在する場合) は、DFHCSDUP から出るときに、最後に出力されたページの番号より 1 だけ大きい番号で更新されます。

## DCBS

DFHCSDUP で内部使用するデータ制御ブロックのセットのアドレス。DCB (または ACB) を指定する場合、それらは DFHCSDUP で通常使用されるものの代わりに内部的に使用されます。

置換 DD 名と置換 DCB の両方を指定する場合、代替 DCB は使用されますが、代替 DD 名は無視されます。

## EXITS

DFHCSDUP の処理の際に呼び出されるユーザー出口ルーチンのセットのアドレス。

パラメーター・リストの構造が 331 ページの図 100 に示されています。

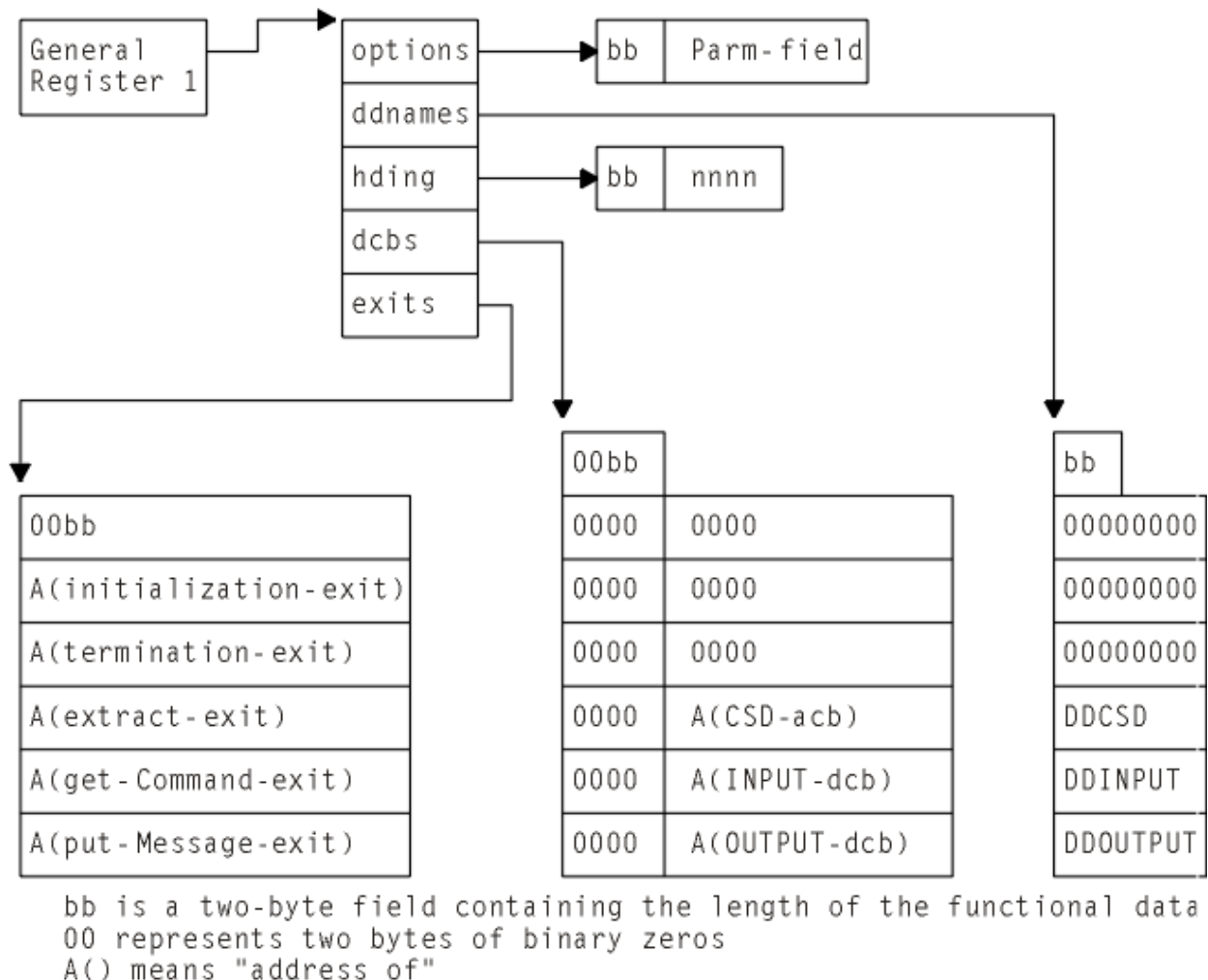


図 100. DFHCSDUP の入力パラメーター

以下の点に注意してください。

- 各パラメーターには長さフィールドが含まれており、それに何らかの機能データが続いています。
- DDNAMES、DCBS、および EXITS パラメーターの機能データには、複数の副項目が含まれています。

- パラメーター OPTIONS、DDNAMES、および HDING は、ハーフワード境界で位置合わせされます。最初の 2 バイト bb には、以下の機能データのバイトの 2 進数が入ります。
- パラメーター DCBS および EXITS は、フルワード境界で位置合わせされます。最初の 4 バイト「00bb」には、以下の機能データのフルワードの 2 進数が入ります。
- bb フィールドがゼロになっているパラメーターがあれば、そのパラメーターは無視されます。
- 機能データの副項目がすべて 2 進ゼロの場合、無視されます。
- 副項目が bb によって示される長さに収まらない場合、無視されます。
- DDNAMES 機能データの各副項目は、DFHCSDUP によって使用されるデフォルトの DD 名を置き換える 8 バイトの DD 名で構成されます。DFHCSDUP は DDNAMES パラメーターの最初の 3 つの副項目を使用しません。4 番目、5 番目、および 6 番目の副項目が存在する場合、それぞれ DFHCSD、SYSIN、SYSPRINT の DD 名を置き換えます。
- DCBS 機能データの各副項目は、2 つのフルワードで構成されます。最初のワードは CICS で使用されません。2 番目のワードには、オープン DCB または ACB のアドレスが含まれます。DCB または ACB が、次に示す正しい属性で開かれていることを確認する必要があります。

#### PRIMARY CSD

AM=VSAM,MACRF=(KEY,DIR,SEQ,IN,OUT)

#### INPUT FILE

DSORG=PS,MACRF=GL,LRECL=80,RECFM=FB

DCB にデータ終了ルーチン (EODAD) または I/O エラー・ルーチン (SYNAD) のアドレスがある場合、そのアドレスは DFHCSDUP によってオーバーレイされます。

#### OUTPUT FILE

DSORG=PS,MACRF=PL,LRECL=125,RECFM=VBA

DFHCSDUP は DCBS パラメーターの最初の 3 つの副項目を使用しません。4 番目、5 番目、および 6 番目の副項目が存在する場合、それぞれ DFHCSD、SYSIN、SYSPRINT の内部 DCB または ACB の代わりに使用されます。

- EXITS パラメーターの各副項目は、出口ルーチンのアドレスを含む単一のフルワードで構成されます。出口ルーチンは、[331 ページの図 100](#) で示されている順序で指定する必要があります。(ユーザー出口については、[332 ページの『DFHCSDUP のユーザー出口点』](#)で説明されています。)

### ユーザー・プログラムの責任

DFHCSDUP を呼び出す前に、呼び出し側プログラムについて、以下の点を事前に確認しておく必要があります。

- RMODE(24) が強制されている。
- オペレーティング・システムのレジスター規則に従っている。
- EXITS パラメーターが渡される場合、出口ルーチンで必要なプログラミング環境が初期設定されている。
- DFHCSDUP で使用するために渡されている ACB または DCB がある場合、それらが OPEN になっている。

### DFHCSDUP のユーザー出口点

DFHCSDUP には 5 つのユーザー出口点があります。適切な入力パラメーターを指定することにより、DFHCSDUP がこれらの任意の出口点で制御を出口ルーチンに渡すように設定できます。

どのユーザー出口も XPI 呼び出しをサポートしていません。

#### ユーザー出口ルーチンに渡されるパラメーター

パラメーター・リストのアドレスはレジスター 1 のユーザー出口ルーチンに渡されます。このリストには、すべての出口ルーチンに渡されるいくつかの標準パラメーターが含まれており、出口ルーチンを呼び出している出口点に固有の出口固有パラメーターもいくつか含まれている場合があります。

標準パラメーター・リストは、CICS グローバル・ユーザー出口で使用されているものとは異なります。次の DFHUEXIT DSECT は、DFHCSDUP およびサンプル・プログラム DFH\$CUS1 で使用されている標準パラ

メーター・リストをマップします。(UEPCMDA フィールドと UEPCMDL フィールドはコマンド取得出口のみで使用されます。)

DFHUEEXIT	DSECT		
UEPEXN	DS	A	ADDRESS OF EXIT NUMBER
UEPGAA	DS	A	ADDRESS OF GLOBAL AREA
UEPGAL	DS	A	ADDRESS OF GLOBAL AREA LENGTH
UEPCRC	DS	A	ADDRESS OF CURRENT RETURN-CODE
UEPTCA	DS	A	ADDRESS OF TCA
UEPCSA	DS	A	ADDRESS OF CSA
UEPHMSA	DS	A	ADDRESS OF SAVE AREA USED BY HOST
UEPSTACK	DS	A	ADDRESS OF KERNEL STACK ENTRY
UEPXSTOR	DS	A	ADDRESS OF STORAGE OF XPI PARMS
UEPTRACE	DS	A	ADDRESS OF TRACE FLAG
*			
UEPCMDA	DS	A	ADDRESS OF UTILITY COMMAND
UEPCMDL	DS	A	ADDRESS OF LENGTH OF UTILITY
*			COMMAND

出口固有パラメーターについては、個々の出口についての記述の中で説明されています。

### 初期設定の出口

DFHCSDUP の初期設定の際に一度初期設定の出口が呼び出されます。その目的は、ルーチンが出口に関連した初期設定を実行できるようにすることです。

例えば、ルーチンはその独自のグローバル作業域を取得して、そのアドレスを UEPGAA に保管し、その長さを UEPGAL によって指示されるハーフワードに保管する場合があります。これらの値は DFHCSDUP によって保管され、他の出口点で使用できるようになります。

### 呼び出される状況

DFHCSDUP の入り口で一度呼び出されます。

### 出口固有のパラメーター

なし。

### 戻りコード

#### UERCNORM (X'00')

処理が続行される。

#### UERCERR

リカバリー不能エラー。これにより、DFHCSDUP は戻りコード‘8’で終了します。

### XPI 呼び出し

使用できません。

### GET コマンド出口

GET コマンド出口の目的は、コマンド行で読み取りを行うことです。指定される場合、コマンドは SYSIN から読み取られません。

呼び出しを行う際、出口ルーチンは、**完全な**コマンドのアドレスと長さを指定する必要があります。通常の戻りコード‘UERCNORM’、またはこれ以上渡すコマンドがないことを意味するコード‘UERCDONE’のいずれかを指定して制御を戻す必要があります。各コマンドを処理した後、DFHCSDUP は、戻りコード‘UERCDONE’を受け取るまで出口を呼び出し続けます。

### 呼び出される状況

本来 DFHCSDUP がコマンドを SYSIN から読み取りを行うポイントで複数回呼び出されます。

### 出口固有のパラメーター

#### UEPCMDA

コマンドのアドレスへのポインター。

#### UEPCMDL

コマンド・テキストの長さを含むハーフワードのアドレス。指定できる最大長は 1536 バイトです。

## 戻りコード

### **UERCNORM (X'00')**

処理が続行される。

### **UERCDONE (X'04')**

処理すべきコマンドはこれ以上ありません。(これは、SYSIN ファイルの終わりに達したことに相当します。)

### **UERCERR**

リカバリー不能エラー。これにより、DFHCSDUP は戻りコード'8'で終了します。

## XPI 呼び出し

使用できません。

## 抽出出口

抽出出口は、EXTRACT コマンドの処理中、いろいろな地点で呼び出されます。

これらの地点については、[321 ページの『ユーザー・プログラムが呼び出された場合』](#)にリストされています。

### 注:

1. EXTRACT ユーザー出口ルーチンを DFHCSDUP への入り口リンケージまたは USERPROGRAM キーワードで指定していないと、構文エラーが発生します。
2. USERPROGRAM キーワードで指定されるユーザー出口ルーチンは、入り口リンケージで指定されるものに優先して使用されます。

## 呼び出される状況

EXTRACT コマンドの処理の際に複数回呼び出されます。

## 出口固有のパラメーター

### **EXTRACT\_FUNCTION\_CODE\_PTR**

到達した EXTRACT の処理の地点を定義するコードを含むハーフワードのアドレス。EXTRACT 機能コードは、[EXTRACT 機能コード](#)でリストされています。

### **EXTRACT\_WORK\_AREA\_PTR**

EXTRACT 作業域のアドレスを含むフルワードのアドレス。

### **EXTRACT\_BACKTRAN\_COMMAND\_PTR**

処理される EXTRACT コマンドのアドレスを含むフルワードのアドレス。

### **EXTRACT\_CSD\_LIST\_NAME\_PTR**

データが抽出されるリストの名前が含まれる 8 バイト・フィールドのアドレス。この値は「リスト開始」および「リスト終了」呼び出しでのみ設定されます。

### **EXTRACT\_CSD\_GROUP\_NAME\_PTR**

データが抽出されるグループの名前が含まれる 8 バイト・フィールドのアドレス。この値は「グループ開始」、「グループ終了」、「オブジェクト開始」、「オブジェクト終了」、および「キーワード」呼び出しで設定されます。

### **EXTRACT\_CSD\_OBJECT\_TYPE\_PTR**

オブジェクトのタイプを識別する 12 バイトのフィールドのアドレス (TRANSACTION や PROGRAM など)。この値は「オブジェクト開始」、「オブジェクト終了」、および「キーワード」呼び出しでのみ設定されます。

### **EXTRACT\_CSD\_OBJECT\_NAME\_PTR**

オブジェクトの名前が含まれる 8 バイトのフィールドのアドレス。この値は「オブジェクト開始」、「オブジェクト終了」、および「キーワード」呼び出しでのみ設定されます。

### **EXTRACT\_KEYWORD\_NAME\_PTR**

処理されるキーワードの名前が含まれる 12 バイトのフィールドのアドレス。この値は「キーワード」呼び出しでのみ設定されます。

### **EXTRACT\_KEYWORD\_LENGTH\_PTR**

キーワードに関連付けられた値の長さを含むハーフワードのアドレス。この値は「キーワード」呼び出しでのみ設定されます。



## EXTRACT\_KEYWORD\_VALUE\_PTR

キーワードに関連付けられた値を含む文字ストリングのアドレス。この値は「キーワード」呼び出しでのみ設定されます。

これらのパラメーターは、DFHCSDUP がバッチ・プログラムとして呼び出されるときに渡されるものと似ています。(322 ページの『DFHCSDUP からユーザー・プログラムに渡されるパラメーター』を参照してください。)ただし、DFHCSDUP がユーザー・プログラムから呼び出されるときには、332 ページの『ユーザー出口ルーチンに渡されるパラメーター』で示されている標準パラメーターもパラメーター・リストに含まれます。

## 戻りコード

### UERCNORM (X'00')

処理が続行される。

### UERCERR

リカバリー不能エラー。これにより、DFHCSDUP は戻りコード‘8’で終了します。

## XPI 呼び出し

使用できません。

## メッセージ書き出し出口

メッセージ書き出し出口は、DFHCSDUP がメッセージを発行するときに必ず呼び出されます。

TSO の下で実行中の場合は、この出口を使用して、オペレーターが ATTENTION 割り込みを入力した後に、DFHCSDUP を終了させることもできます。(330 ページの『ユーザー・プログラムからの DFHCSDUP の呼び出し』を参照してください。)または、この出口を使用して、オペレーターの各国語でメッセージを提供することもできます。

この出口が提供されている場合でも、メッセージは必ずデフォルトの出力ファイル (つまり SYSPRINT、または DFHCSDUP への入力リンケージで指定された置換 DD 名) にも追加で書き込まれます。

## 呼び出される状況

メッセージが発行されるときに呼び出される。

## 出口固有のパラメーター

### UEPMNUM

メッセージ番号が入る 4 文字フィールドのアドレス

### UEPMDOM

予約済み

### UEPINSN

挿入フィールドの数が入る 2 バイト・フィールドのアドレス

### UEPINS A

次のメッセージ構造のアドレス:

	DS	F	Reserved
INS_1_TEXT_PTR	DS	A	Address of insert 1
INS_1_LEN_PTR	DS	A	Address of a fullword containing the length of insert 1
	DS	F	Reserved
	DS	F	Reserved
INS_2_TEXT_PTR	DS	A	Address of insert 2
INS_2_LEN_PTR	DS	A	Address of a fullword containing the length of insert 2
	DS	F	Reserved
	...		
	DS	F	Reserved
INS_n_TEXT_PTR	DS	A	Address of insert n
INS_n_LEN_PTR	DS	A	Address of a fullword containing the length of insert n
	DS	F	Reserved

出口固有パラメーターにはメッセージ番号と挿入フィールド数のみが用意されていて、TSO オペレーターの言語でメッセージを提供することができます。UEPINS A が指す構造は、UEPINSN で要求されている回数まで繰り返されます。

## 戻りコード

### **UERCNORM (X'00')**

処理が続行される。

### **UERCERR**

リカバリー不能エラー。これにより、DFHCSDUP は戻りコード'8'で終了します。

### **XPI 呼び出し**

使用できません。

## 終了出口

終了出口の目的は、最終ハウスキーピング処理を実行できるようにすることです。これは、DFHCSDUP の正常終了または異常終了の前に呼び出されます。

## 呼び出される状況

DFHCSDUP の終了前に一度呼び出されます。

## 出口固有のパラメーター

### **UEPTRMFL**

終了のモードを示す 1 バイト・フィールドのアドレス。可能な値は次のとおりです。

#### **X'00'**

正常終了。

#### **X'F0'**

異常終了。

出口プログラムは、このフィールドの値をリセットできません。

## 戻りコード

### **UERCNORM (X'00')**

処理が続行される。

### **UERCERR**

リカバリー不能エラー。これにより、DFHCSDUP は戻りコード'8'で終了します。

### **XPI 呼び出し**

使用できません。

## サンプル・プログラム DFH\$CUS1

CICS 提供のサンプル・プログラム DFH\$CUS1 は、DFHCSDUP をユーザー・プログラムから起動する方法を例示しています。これは TSO/E オペレーティング・システムの下で実行されるコマンド・プロセッサ (CP) として作成されています。

DFH\$CUS1 は、DFHCSDUP で通常使用されるものとは異なる DCB および ACB 名を使用することに注意してください。プログラムを TSO/E の下で実行する前に、これらが割り振られていることを確認してください。

DFH\$CUS1 は TSO から実行するように作成されていますが、例えば REXX EXEC などから実行することも可能です。これを行う前に、DFH\$CUS1、DFHCSDUP、および DFHEITCU を含むロード・ライブラリーが、ユーザーの検索チェーン、LOGON プロシージャ、またはリンク・リスト内にあることを確認してください。336 ページの図 101 は、DFH\$CUS1 を起動する REXX EXEC のサンプルです。

```
/*REXX*/  
"ALLOCATE DSN('XXXXX.CICS730.DFHCSO') DD(ALTACB) SHR"  
"ALLOC DD(SIN) DA(*) BLKSIZE(80)"  
"ALLOC DD(SPRINT) DA(*) BLKSIZE(80)"  
X = PROMPT('ON')  
ADDRESS TSO "DFH$CUS1"  
"FREE DD(ALTACB)"  
"FREE DD(SIN)"  
"FREE DD(SPRINT)"  
RETURN 0
```

図 101. DFH\$CUS1 サンプル・プログラムを起動する REXX プログラム

## ユーザー置換可能プログラムのアセンブルとリンク・エディット

ほとんどのユーザー置換可能プログラムは、コマンド・レベル・プログラムとして用意されており、変換、アセンブル、リンク・エディットが必要です。CICS は、ユーザー置換可能プログラムを変換、アセンブル、およびリンク・エディットするためのプロシージャーを提供しています。

### このタスクについて

DFHAPXPO 以外のすべてのプログラムは、コマンド・レベル・プログラムとして用意されており、アセンブリーとリンク・エディットの前に変換が必要です。独自のバージョンの DFHZNEP、DFHTEP、DFHXCURM では、変換プログラム・オプション NOPROLOG と NOEPILOG をコーディングしなければなりません。

### 手順

1. 使用する CICS 提供のユーザー置換可能プログラムをコピーして、そのコピーを編集します。

CICS 提供のユーザー置換可能プログラムのソースは、CICSTS56.CICS.SDFHSAMP ライブラリーにインストールされています。元の SDFHSAMP がサービスで使用されていて、ユーザー置換可能プログラムが変更されている場合は、独自のコードにその変更内容を組み込むこともできます。

2. 独自のバージョンのプログラムを変換し、アセンブルし、リンク・エディットします。

- DFHAXPO を置換する場合は、プログラムを変換したり、EXEC インターフェース・モジュールを使用してリンク・エディットしたりする必要はありません。DFHASMVS プロシージャーを使用してこれらのプログラムをコンパイルできます。
- その他のプログラムを置換する場合は、CICS 提供の該当するプロシージャーを使用して、プログラムの変換、アセンブル、リンク・エディットを実行します。例えば、AMODE(24) や AMODE(31) のアセンブラー・プログラムの場合は、DFHEITAL プロシージャーを使用します。プログラムを EXEC インターフェース・モジュール・スタブにリンク・エディットすることも必要です。プログラムは、そのスタブを使用して EXEC インターフェース・プログラム DFHEIP と通信できるようになります。プログラムを EXEC インターフェース・スタブにリンク・エディットする時には、DFHEITAL プロシージャーを使用します。SMP/E を使用している場合は、変換とアセンブリーの後にオブジェクト・デック出力を SMP/E に渡してリンク・エディットを実行することもできます。

言語ごとに用意されているプロシージャーの使用法については、[アプリケーション・プログラムをインストールするための CICS 提供プロシージャーの使用](#)を参照してください。

### 例

ユーザー置換可能プログラムのアセンブリーとリンク・エディットのためのジョブ・ストリームの例を [337 ページの図 102](#) に示します。この図の後に説明のための注記があります。

```
//ASSEMBLE EXEC DFHEITAL,
//  ASMBLR=ASMA90,
//  INDEX='CICSTS56.CICS',
//  PROGLIB='your_loadlib',
//  DSCTLIB='your_copylib',
//  PARM.TRN='NOPROLOG,NOEPILOG',
//  PARM.ASM='DECK,NOOBJECT,LIST,XREF(SHORT),RENT,ALIGN',
//  LNKPARM='LIST,XREF,RENT,MAP,AMODE(31),RMODE(ANY)'
//TRN.SYSIN DD DSN=your_sourcelib(program_name),DISP=SHR 5 6
//LKED.SYSIN DD *
  ENTRY program_name
  NAME program_name(R)
//*
```

図 102. ユーザー置換可能プログラムのアセンブルとリンク・エディットのためのジョブ・ストリーム

注:

- 1 CICS ライブラリーの高位修飾子。
- 2 ロード・モジュールのリンク・エディット先のライブラリー。
- 3 ローカルのアセンブラー・マクロと COPY メンバーが入るライブラリーの名前(オプション)。

**4** このオプションは、DFHXCURM の場合に必須です。提供されているサンプル・バージョンの DFHTEP と DFHZNEP の場合にも必須です。

**5** `your_sourcelib` は、独自の変更バージョンのプログラムが入るライブラリーの名前です。

**6** `program_name` は、アセンブルするユーザー置換可能プログラムのソース・メンバー名です。提供されている DFHTEP サンプルのソース・メンバーは DFHXTEP です。提供されている DFHZNEP サンプルのソース・メンバーは DFHZNEP0 です。

**7** 通常、リンケージ・エディターの入力には、ここにある 2 つのステートメントを含めます。`program_name` の部分は、コンパイルするユーザー置換可能プログラムの名前に置き換えます。ただし、CICS 提供の一部のサンプル・プログラムについては例外があります。[338 ページの図 103](#) をご覧ください。

図 103. DFHTEP と DFHZNEP のリンク・エディット・ステートメント

DFHTEP のリンク・エディット・ステートメント:

```
ENTRY DFHTEPNA  
NAME DFHTEP(R)
```

DFHZNEP のリンク・エディット・ステートメント:

```
ENTRY DFHZNENA  
NAME DFHZNEP(R)
```

## 第7章 セキュリティー処理のカスタマイズ

CICS Transaction Server for z/OS, バージョン 5 リリース 6 では、CICS でサポートされるセキュリティー形式は、RACF などの外部セキュリティー・マネージャー (ESM) によって提供されたものだけです。CICS では、RACROUTE マクロを使用することによって、MVS System Authorization Facility (SAF) インターフェースで許可要求を外部セキュリティー・マネージャーにルーティングできます。SAF は MVS ルーターを、リソース制御を提供および要求するすべての製品に共通のシステム・インターフェースとして使用します。CICS では、いくつかの RACF 呼び出し可能サービスも使用されます。このセクションでは、RACF 出口プログラムを使用してセキュリティー処理をカスタマイズする方法について説明します。

SAF レベルでセキュリティーをカスタマイズするには、MVS ルーター出口 (ICHRTX00) を作成して RACROUTE 呼び出しをカスタマイズし、RACF 呼び出し可能インストール・システム出口 (IRRSXT00) を作成して RACF 呼び出し可能サービスをカスタマイズします。

### ユーザー提供の ESM への制御の受け渡し

通常、呼び出し元 (CICSplex SM など) は、MVS ルーターを呼び出し、RACROUTE 出口パラメーター・リストを介してそのルーターに要求タイプ、リクエスター、およびサブシステムのパラメーターを渡します。MVS ルーターは、これらのパラメーターを使用して、ルーター出口を呼び出します。ルーター出口は、その処理を完了すると、ルーターに戻りコードを渡します。

戻りコードが 0 の場合、ルーターは RACF を呼び出します。RACF は、レジスター 15 と 0 にそれぞれ戻りコードと理由コードを入力して、その呼び出しの結果をルーターに報告します。ルーターは、RACF の戻りコードと理由コードをルーターの戻りコードと理由コードに変換して、それらを呼び出し元に渡します。ルーターは、未変換の RACF の戻りコードと理由コードをルーター入力パラメーター・リストの 1 番目と 2 番目のワードに入れて、呼び出し元に追加情報を提供します。

インストール済み環境で RACF を使用していない場合は、MVS ルーター出口を作成して、代替の ESM に制御を渡すことができます。ただし、これを行った場合、引き続き、CICSplex SM に、受け取ることが予期される RACF の戻りコードと理由コードを提供する必要があります。RACF が呼び出されないように、ルーター出口を設定します。また、ルーター入力パラメーター・リストの 1 番目と 2 番目のフルワードに RACF の戻りコードと理由コードを入れるように出口をコーディングして、RACF 呼び出しの結果をシミュレートします。RACF の戻りコードと理由コードについては、z/OS Security Server RACF メッセージおよびコードの資料で取り上げられています。

ユーザー提供の ESM への制御の受け渡しについて詳しくは、『z/OS MVS Installation Exits』の『出口ルーチン処理』を参照してください。

### 非 RACF ユーザーの場合 – ESM パラメーター・リスト

CICS (または別の呼び出し元) によって、ESM パラメーター・リスト内のご使用の外部セキュリティー・マネージャーに情報が渡されます。そのアドレスは、MVS ルーター・パラメーター・リストのフィールド SAFPRACP を使用して計算できます。

呼び出し元が CICS である場合、ESM パラメーター・リストの「INSTLN」フィールドは、インストール・データのパラメーター・リストを指します。このリストには、ESM 出口プログラムで使用できる CICS 関連情報が含まれています。

ESM パラメーター・リストの形式と、「INSTLN」フィールドの実際の名前は、処理されている CICS セキュリティー・イベントによって異なります。(ルーター・パラメーター・リストの「要求タイプ」フィールド (SAFPREQT) には、RACROUTE REQUEST タイプを示すことにより、ESM が呼び出されている理由が表示されます。) 340 ページの表 28 に、MVS マクロを使用して ESM パラメーター・リストのいくつかの形式をマップする方法を示します。

表 28. ESM パラメーター・リストのマッピング

RACROUTE REQUEST タイプ	パラメーター・リストのマッピング・マクロ	INSTLN フィールド名
VERIFY	IRRPRIPL	INITIPTR (X'10')
AUTH	ICHACHKL	ACHKIN31 (X'20')
FASTAUTH	使用不可	オフセット X'18'
LIST	使用不可	オフセット X'0C'
EXTRACT	使用不可	なし

注：INSTLN フィールドは、ESMEXITS システム 初期化パラメーターに INSTLN を指定した場合にのみ、インストール・パラメーター・リストを指します。このパラメーターのデフォルト値は NOINSTLN です。これは、初期化データが渡されないことを意味します。

## RACF ユーザーの場合 – RACF ユーザー出口パラメーター・リスト

RACF ユーザーである場合は、RACF ユーザー出口パラメーター・リストから直接、初期化データ・パラメーター・リストのアドレスを見つけることができます。ユーザー出口パラメーター・リスト内の関連するフィールドの名前は、RACROUTE REQUEST タイプと、起動される RACF ユーザー出口に応じて異なります。

340 ページの表 29 に、REQUEST タイプ、出口名、およびフィールド名の間の関係を示します。

表 29. インストール・データ・パラメーター・リストのアドレスの取得

RACROUTE REQUEST タイプ	RACF 出口	出口リスト・マッピング・マクロ	パラメーター・リスト・フィールド名
VERIFY	ICHRIX01	ICHRIXP	RIXINSTL
VERIFY	ICHRIX02	ICHRIXP	RIXINSTL
AUTH	ICHRCX01	ICHRCXP	RCXINSTL
AUTH	ICHRCX02	ICHRCXP	RCXINSTL
FASTAUTH	ICHRFX01	ICHRFXP	RFXANSTL
FASTAUTH	ICHRFX02	ICHRFXP	RFXANSTL
LIST	ICHRLX01	ICHRLX1P	RLX1INST
LIST	ICHRLX02	ICHRLX2P	RLX2PRPA (注 2 を参照)。
EXTRACT	使用不可	使用不可	なし

注：

- xxxINSTL フィールドは、ESMEXITS システム 初期化パラメーターに INSTLN を指定した場合にのみ、インストール・パラメーター・リストを指します。このパラメーターのデフォルト値は NOINSTLN です。これは、初期化データが渡されないことを意味します。
- RLX2PRPA には、ICHRLX01 ユーザー出口パラメーター・リスト (RLX1P) のアドレスが含まれています。次に、RLX1P のフィールド RLX1INST がインストール・データ・パラメーター・リストを指します。
- RACF APAR OA43999 の結果として、パスワードは、パスワードが有効な場合に ICHRIX01 ユーザー出口では使用できなくなります。通常の使用法では、この出口は、パスワードが無効であった場合にのみパスワードにアクセスできます。これは、パスワードの確認と変更がサインオンとは別個に行われるようになったためです。これにより、サインオン中に行われる RACF 呼び出しが、これらの呼び出しの一



部として起動されるユーザー出口で利用できるデータとともに変更されました。以下の手順が実行されます。

- 提供されたパスワードを確認するために RACF サービス IRRSPW00 が呼び出されます。このサービスではユーザー出口は駆動されません。パスワードの確認に失敗した場合や、提供されたパスワードがパスチケットである場合、あるいはパスワードが有効であっても以前に失敗したことがある場合には、RACROUTE REQUEST=VERIFYX 呼び出しが行われます。ICHRIX01 ユーザー出口が呼び出され、インストール・データが渡されます。
  - パスワード変更操作が要求された場合、オリジナルのパスワードを確認したり、パスワード変更操作を実行したりするため、RACROUTE REQUEST=VERIFYX の呼び出しがなされます。ICHRIX01 ユーザー出口が呼び出され、インストール・データが渡されます。
  - サインオンでは RACROUTE REQUEST=VERIFY が使用されます。この呼び出しによって ICHRIX01 ユーザー出口が呼び出され、インストール・データが渡されます。このパスワードと新規パスワードは使用できません。
4. REQUEST=EXTRACT の RACF ユーザー出口はありません。また、インストール・パラメーター・データの受け渡しも行われません。カスタマイズは、MVS ルーター出口 ICHRTX00 を使用して行う必要があります。

RACF 出口パラメーター・リストについて詳しくは、[z/OS Security Server RACF セキュリティー管理者のガイド](#)を参照してください。RACF を使用した CICS セキュリティーの処理について詳しくは、[RACF ファシリティー](#)を参照してください。

## インストール・データ・パラメーター・リストのマッピング

インストール・データ・パラメーター・リストを使用すると、ご使用の ESM 出口プログラムで、処理中の CICS セキュリティー・イベントと、現在の CICS 環境の詳細情報にアクセスできます。インストール・パラメーター・リストは、マクロ DFHXSUXP を使用してマップできます。

[インストール・データ・パラメーター・リスト](#)を参照してください。

DSECT DFHXSUXP には、以下のフィールドが含まれています。

### UXPLEN

このパラメーター・リストの長さ (バイト単位) が入っているハーフワード。

### UXPARROW

矢印の「目印」(>)。

### UXPDFHXS

所有しているコンポーネントの名称 (DFHXS)。

### UXPBLKID

ブロック ID の名称 (UXPARMS)。

### UXPPHASE

ESM の呼び出しの理由 (つまり、処理されているセキュリティ・イベント) を示す 1 バイト・コードのアドレス。このコードの値は、以下のいずれかです。

#### DEFAULT\_SIGN\_ON (X'01')

デフォルト・ユーザー ID のサインオン

#### PRESET\_SIGN\_ON (X'02')

事前設定セキュリティ端末のサインオン

#### IRC\_SIGN\_ON (X'03')

IRC (MRO) リンクのリンク・サインオン

#### LU61\_SIGN\_ON (X'04')

LUTYPE6.1 リンクのリンク・サインオン

#### LU62\_SIGN\_ON (X'05')

APPC リンクのリンク・サインオン

#### XRF\_SIGN\_ON (X'06')

サインオンの XRF 追跡

**ATTACH\_SIGN\_ON (X'07')**

リンク・ユーザーの接続時間サインオン

**NON\_TERMINAL\_SIGN\_ON (X'08')**

非端末ユーザー ID のサインオン

**USER\_SIGN\_ON (X'10')**

通常のユーザー・サインオン

**PRESET\_SIGN\_OFF (X'22')**

端末が削除された場合のサインオフ

**LINK\_SIGN\_OFF (X'25')**

リンクがクローズされた場合のサインオフ

**XRF\_SIGN\_OFF (X'26')**

サインオフの XRF 追跡

**ATTACH\_SIGN\_OFF (X'27')**

リンク・ユーザーのタスクの終了サインオフ

**NON\_TERMINAL\_SIGN\_OFF (X'28')**

非端末ユーザー ID のサインオフ

**USER\_SIGN\_OFF (X'30')**

通常のユーザー・サインオフ

**TIMEOUT\_SIGN\_OFF (X'31')**

端末異常条件プログラムによって強制されたサインオフ。または CSSC トランザクションによるタイムアウトで強制されたサインオフ

**USRDELAY\_SIGN\_OFF (X'32')**

USRDELAY 間隔の期限切れによるサインオフ

**DEFERRED\_SIGN\_OFF (X'33')**

タスクの終わりまで据え置かれたサインオフ

**USER\_ATTACH\_CHECK (X'40')**

ユーザーに対するトランザクション攻撃検査

**LINK\_ATTACH\_CHECK (X'41')**

リンクに対するトランザクション攻撃検査

**EDF\_ATTACH\_CHECK (X'42')**

CEDF に対するトランザクション攻撃検査

**USER\_COMMAND\_CHECK (X'50')**

ユーザーに対するコマンド検査

**LINK\_COMMAND\_CHECK (X'51')**

リンクに対するコマンド検査

**EDF\_COMMAND\_CHECK (X'52')**

EDF に対するコマンド検査

**USER\_RESOURCE\_CHECK (X'60')**

ユーザーに対するリソース検査

**LINK\_RESOURCE\_CHECK (X'61')**

リンクに対するリソース検査

**EDF\_RESOURCE\_CHECK (X'62')**

EDF に対するリソース検査

**USER\_SURROGATE\_CHECK (X'68')**

ユーザーに対する代理検査

**LINK\_SURROGATE\_CHECK (X'69')**

リンクに対する代理検査

**EDF\_SURROGATE\_CHECK (X'6A')**

EDF に対する代理検査

**USER\_QUERY\_CHECK (X'70')**

ユーザーに対する照会検査

**LINK\_QUERY\_CHECK (X'71')**

リンクに対する照会検査

**EDF\_QUERY\_CHECK (X'72')**

EDF に対する照会検査

**INITIALIZE\_SECURITY (X'80')**

CICS セキュリティーの初期化

**REBUILD\_SECURITY (X'81')**

CEMT またはコマンド・レベルの SECURITY REBUILD

**XRF\_TRACK\_INITIALIZE (X'82')**

初期または再構築の XRF 追跡。

**PASSWORD\_CHANGE (X'90')**

パスワードの変更

**PASSWORD\_VERIFICATION (X'91')**

パスワードの確認

**UXPSUBSY**

CICS サブシステム ID が含まれている領域のアドレス。

**UXPAPPL**

CICS アプリケーション ID が含まれている領域のアドレス。

注：CICS が z/OS Communications Server for SNA 総称リソースのメンバーである場合、UXPAPPL が指す領域には、固有のアプリケーション ID ではなく、総称 アプリケーション ID が含まれています。

**UXPCWA**

共通作業域のアドレス。

**UXPTRAN**

トランザクション ID が含まれている領域のアドレス。

**UXPPROG**

プログラム名が含まれている領域のアドレス。プログラム名が識別できない場合、このアドレスはゼロの場合があります。

**UXPTERM**

端末 ID が含まれている領域のアドレス。端末が要求に関連付けられていない場合、このアドレスはゼロの場合があります。

**UXPLUNAM**

SNA LU 名が含まれている領域のアドレス。端末が要求に関連付けられていない場合、このアドレスはゼロの場合があります。あるいは、端末が SNA 端末でない場合、この領域はブランクの場合があります。

**UXPTCTUA**

TCT ユーザー域のアドレス。

**UXPTCTUL**

TCTUA の長さが含まれているフルワードのアドレス。

**UXPCOMM**

2 ワードの通信域のアドレス。

## 早期検証処理の使用

CICS サインオン・ルーチンでは、問題プログラム状態の ENVIR=VERIFY オプションを指定した RACROUTE REQUEST=VERIFY マクロを使用して、SAF インターフェースが呼び出されます。ESM が RACF である場合、このバージョンのマクロを呼び出しても効果はありませんが、その他の外部セキュリティー・マネージャ製品では、SAF 出口インターフェースを介した制御が可能であり、その独自の早期検証 ルーチンを実行できます。

CICS では、通常検証 ルーチンを実行するために ENVIR=CREATE オプションを指定した RACROUTE REQUEST=VERIFY マクロが発行されるまで、アクセサ環境エレメントの作成が据え置かれます。このマクロの ENVIR=CREATE バージョンは、監視プログラム状態で実行されているセキュリティー・マネージャー・ドメインによって発行されます。

CICS によって、RACROUTE REQUEST=VERIFY マクロの ENVIR=VERIFY バージョンに関する以下の情報が渡されます。

#### **USERID**

CICS 領域にサインオンしているユーザーのユーザー ID。

#### **GROUP**

ユーザーのサインオン先となるグループのグループ名 (指定された場合)。

#### **PASSWRD**

ユーザー ID を検証するためのユーザーのパスワード。

#### **NEWPASS**

ユーザーのパスワードの新規値 (指定された場合)。これによって、既存のパスワードが変更され、以後のサインオンで使用されます。

#### **OIDCARD**

オペレーター ID カードの内容 (指定された場合)。

#### **APPL**

ユーザーがサインオンしている CICS 領域の APPLID。渡される APPLID は、指定されているシステム初期化パラメーターによって異なります。

#### **INSTLN**

CICS 関連情報のベクトルを指すポインター。これは、DFHXSUXP マッピング・マクロを使用してマップできます。このポインターは、CICS 領域のシステム初期化パラメーターとして ESMEXITS=INSTLN が指定されている場合にのみ有効です。

INSTLN パラメーターによって参照されるインストール・データには、ポインター UXPCOMM が含まれます。このポインターは、サインオン検証プロセスの 2 つのフェーズ (ENVIR=VERIFY で開始される早期検証ルーチンと、ENVIR=CREATE で開始される通常検証ルーチン) 間で情報の受け渡しに使用できる 2 ワードの通信域を指します。

CICS では、各タスク用に、CICS キー・ストレージ内に別個の通信域が維持されます。

## **早期検証ルーチンの作成**

ENVIR=VERIFY オプションに対して作成された早期検証ルーチンは、SAF ベクトル・テーブルのフィールド SAFVRACR でアドレス指定される入り口点を持つ外部セキュリティー・マネージャーから通常受け取るのと同じ方法で、SAF から制御を受け取ります。

この検証ルーチンでは、以下のような、その呼び出し元と同じ状態で制御を受け取ります。

- 問題プログラム状態
- タスク・モード (通常、CICS 準再入可能 TCB)
- PSW ストレージ・キー 8
- 31 ビット・アドレッシング・モード
- 基本アドレス変換モード

レジスター 13 は、標準の 18 ワード保管域を指しています。レジスター 1 は、2 ワードのパラメーター・リストを指しています。ここで、

- 最初のワードは、VERIFY 関数の SAF パラメーター・リストのアドレスです。
- 2 番目のワードは、152 バイト作業域のアドレスです。

## **早期検証ルーチンでの CICS API コマンドの使用**

早期検証ルーチンでは、以下のインターフェース規則に従って CICS アプリケーション・プログラミング・インターフェース (API) コマンドを使用できます。

- このルーチンは、アセンブラーで作成する必要があります。
- ルーチンへの入り口は、DFHEIENT マクロを経由する必要があります。このマクロによって、呼び出し元のレジスターが保管され、CICS の早期検証 API 環境が確立されます。
- ルーチンからの出口は、DFHEIRET マクロを経由する必要があります。このマクロによって、CICS の早期検証 API 環境が解放され、呼び出し元のレジスターが復元されます。
- ルーチンは、通常の CICS API スタブ DFHEAIO ではなく、特殊なセキュリティー・ドメイン API スタブ DFHXSEAI でリンク・エディットする必要があります。この CICS の早期検証スタブは、SAF インターフェースのリンケージ要件を満たす特殊なインターフェース・ルーチンへのリンクを生じさせ、現在の CICS コマンド環境を保管します。さらに、標準の EXEC インターフェース・スタブ DFHEAI も、以下のように ORDER ステートメントを使用して早期検証ルーチンの直前に入れる必要があります。

```
INCLUDE SYSLIB(DFHXSEAI)
INCLUDE SYSLIB(DFHEAI)
ORDER   DFHEAI,verify-program,DFHEAIO
ENTRY   verify-program
```

DFHEIENT マクロおよび DFHEIRET マクロは、CICS 変換プログラムによって挿入されます。ただし、

```
*ASM XOPTS(NOPROLOG,NOEPILOG)
```

がプログラムの最初のステートメントとして指定されている場合を除きます。DFHEIENT マクロでは、レジスター 15 がその最初の実行可能な指示を指すと想定されます。

DFHEIENT マクロから戻された時点で、DFHEISTG マクロによってマップされた CICS ストレージ域が確立されています。ポインター DFHEIBP (および DFHEIENT の EIBREG パラメーターで指定されたレジスター) には、EXEC インターフェース・ブロック (EIB) のアドレスが含まれています。DFHEICAP には、SAF インターフェースによって提供された元のパラメーター・リストを指すポインターが含まれています。

## 早期検証ルーチンからの戻りコードと理由コード

制御を戻す前に、早期検証ルーチンで、SAF パラメーター・リストの SAFPRRET フィールドと SAFPRREA フィールドに戻りコードと理由コードを設定する必要があります。さらに、プログラムの終了に使用される DFHEIRET マクロの RCREG キーワードで指定されたレジスターに、SAF 戻りコードとして戻される値を渡す必要もあります。

これらの戻りコードは、CICS サインオン機能によって検査され、SAFPRRET 内の非ゼロの値はすべて検証失敗と解釈され、サインオンは失敗します。戻りコードがゼロの場合は、サインオンの続行が許可され、最終的に CICS によって、監視プログラム状態および CICS リソース所有 TCB の制御のもとで、RACROUTE REQUEST=VERIFY, ENVIR=CREATE マクロが発行されます。CICS で外部セキュリティー・マネージャーからの ACEE アドレスが受け入れられるのは、この呼び出しの場合のみです。

## CICS セキュリティー制御点

CICS では、RACROUTE マクロと RACF 呼び出し可能サービスを使用して、外部セキュリティー・マネージャー (ESM) が呼び出されます。これらの呼び出しは、いくつかの制御点で発行されます。一部の呼び出しは、常に発行される訳ではありません。これは、CICS では、既に CICS 領域にサインインしている 適切な ユーザー ID のエントリーが再使用されるためです。

### RACROUTE マクロ

#### RACROUTE

このマクロは、以下に示すマクロへの「フロントエンド」です。このマクロは、MVS ルーターを呼び出します。

#### RACROUTE REQUEST=VERIFY

このマクロは、オペレーターのサインオンではパラメーター ENVIR=CREATE を、サインオフではパラメーター ENVIR=DELETE をそれぞれ指定して、発行されます。このマクロは、ACEE (アクセス制御環境エレメント) を作成または破棄します。このマクロは、EXEC CICS SIGNON コマンドを介した通常のサインオンの初期に、パラメーター ENVIR=VERIFY を指定して発行されますが、このコマンドは RACF では無視されます。

このマクロは、以下の CICS 制御点で発行されます。

以下に示す制御点は、それぞれ ENVIR=CREATE に関連しています。

- **EXEC CICS SIGNON** を介した通常のサインオン。
- デフォルト・ユーザー ID DFLTUSER のサインオン。
- 事前設定セキュリティ端末のサインオン。
- MRO セッションのサインオン。
- LU6.1 セッションのサインオン。
- LU6.2 セッションのサインオン。
- 前述のいずれかの XRF 追跡のためのサインオン。
- LOCAL 以外の ATTACHSEC のすべてのオペランドに対して、接続要求を出したユーザー ID に関連しているサインオン。
- **VERIFY TOKEN** (BASICAUTH または JWT 用)。オプション IDT は、要求に応じて JWT の入出力を実行するために使用されます。
- 毎日、初回にユーザー ID が認証される。

以下に示す制御点は、それぞれ ENVIR=DELETE に関連しています。

- **EXEC CICS SIGNOFF** を介した通常のサインオフ。
- 端末を削除する場合のサインオフ。
- TIMEOUT が期限切れになった場合のサインオフ。
- USRDELAY が期限切れになった場合のサインオフ。
- MRO セッションのサインオフ。
- LU6.1 セッションのサインオフ。
- LU6.2 セッションのサインオフ。
- 前述のいずれかの XRF 追跡のためのサインオフ。
- LOCAL 以外のすべての ATTACHSEC のオペランドに対して、接続要求を出したユーザー ID に関連しているサインオフ。
- LOCAL 以外のすべての ATTACHSEC のオペランドに対して、RACF から CICS に対してユーザー・プロファイルの変更が通知され、そのサインオンしたユーザー ID に関連する接続済み要求が完了したことによるサインオフ。
- RACF から CICS に対してユーザー・プロファイルの変更が通知され、新規接続要求が出されて、**USRDELAY** システム初期化パラメーター内のその値が期限切れになっていないことによるサインオフ。このサインオフの後に、サインオンが続きます。
- 毎日、初回にユーザー ID が認証される。
- **VERIFY TOKEN** (BASICAUTH または JWT 用)。ACEE は、制御が呼び出し元に戻される前に即時に削除されます。

#### **RACROUTE REQUEST=VERIFYX**

このマクロは、単一の呼び出しで ACEE を作成および削除します。このマクロは、以下の制御点で発行されます。

- パスワードの確認を含む認証プロセスを使用する場合は、以下のいずれかの条件が適用されます。
  - パスワードが無効である (R\_Password または RACROUTE REQUEST=EXTRACT の後)。
  - 以前に試行したログインが無効だった。
- VERIFY の代替としてのサインオン (ATTACHSEC(VERIFY) または ATTACHSEC(PERSISTENT) を使用した LU6.2 リンク間での以後の接続サインオンに対して、最適化されたサインオンが実行された場合)。
- パスワードまたはパスワード・フレーズの変更



### **RACROUTE REQUEST=FASTAUTH**

このマクロは、ACEE で識別されたユーザーの代わりに、リソース検査中に発行されます。このマクロはストレージ内のリソース・プロファイルを使用する高性能形式の REQUEST=AUTH であり、監査は行われません。このマクロは、以下の CICS 制御点で発行されます。

- ローカル・トランザクションの接続時
- トランザクション接続のためのリンク・セキュリティの検査時
- MRO タスクのトランザクションの検証
- CICS リソース検査
- CICS リソースのリンク・セキュリティ検査
- EDF のトランザクションの検証
- (EDF による) テスト中のトランザクションのトランザクション検証
- DBCTL PSB スケジューリングのリソース・セキュリティ検査
- DBCTL PSB スケジューリングのリンク・セキュリティ検査
- リモート DL/I PSB スケジューリングのセキュリティ検査
- 代理ユーザー権限の検査時
- RESTYPE オプションを指定した QUERY SECURITY

### **RACROUTE REQUEST=AUTH**

このマクロは、パス長がより長いリソース検査の形式を提供し、監査を行います。このマクロは、以下の場合に使用します。

- FASTAUTH の呼び出しで、ロギングを必要とするアクセス障害が示された後。
- RESCLASS オプションを指定した QUERY SECURITY 要求が使用された場合。このオプションは、CICS によってストレージ内のプロファイルが作成されていないリソースに対する要求を示します。

### **RACROUTE REQUEST=LIST**

このマクロは、REQUEST=FASTAUTH で必要なストレージ内のプロファイル・リストを作成および削除する場合に発行されます。リソース・クラスごとに 1 つの REQUEST=LIST マクロが必要です。このマクロは、以下の CICS 制御点で発行されます。

- CICS セキュリティの初期化時
- EXEC CICS PERFORM SECURITY REBUILD コマンドの発行時
- XRF によるこれらいずれかのイベントの追跡時

### **RACROUTE REQUEST=EXTRACT**

このマクロは、R\_Password が使用できない場合に R\_Password の代わりに使用されます (注 1 を参照してください)。

RACROUTE REQUEST=EXTRACT マクロは、各国語とユーザー名を取得する場合にも、SEGMENT=CICS,CLASS=USER パラメーターと SEGMENT=BASE,CLASS=USER パラメーターを指定して、以下のすべての制御点で発行されます。

- EXEC CICS SIGNON を介した通常のサインオン
- デフォルト・ユーザー ID DFLTUSER のサインオン
- 事前設定セキュリティ端末のサインオン
- MRO セッションのサインオン
- LU6.1 セッションのサインオン
- LU6.2 セッションのサインオン
- 前述のいずれかの XRF 追跡のためのサインオン
- LOCAL 以外の ATTACHSEC のすべてのオペランドに対して、接続要求を出したユーザー ID に関連しているサインオン

このマクロは、LU6.2 バインド・セキュリティ検査の際にも、LU6.2 セッションのバインドの CICS 制御点で SEGMENT=SESSION,CLASS=APPCLU パラメーターを指定して発行されます。

このマクロを使用すると、USRDELAY 期間内にユーザー・テーブルのエントリが再使用された場合に、ユーザーのパスワードを確認できます。

REQUEST=EXTRACT パラメーターには、関連付けられた RACF ユーザー出口がないため、インストール・パラメーター・データは渡されません。MVS ルーター出口 ICHRTX00 は、カスタマイズに使用します。

これらすべてのマクロの詳細な説明については、「[z/OS Security Server RACROUTE マクロ 解説書](#)」を参照してください。

## **z/OS Security Services RACF 呼び出し可能サービス**

CICS では、ESM を呼び出す際に、さまざまな目的で以下の呼び出し可能インターフェースを使用します。

### **deleteUSP (IRRSU00): USP の削除**

HFS ファイル・セキュリティに使用します。

### **initACEE (IRRSIA00): ACEE の初期化**

証明書からユーザー ID を取得するために使用します。

### **initUSP (IRRSIU00): USP の初期化**

HFS ファイル・セキュリティに使用します。

### **R\_admin (IRRSEQ00): RACF 管理 API**

証明書ラベルの検証に使用します。

### **R\_cacheserv (IRRSCH00): キャッシュ・サービス**

ACEE に関連付けられた ICRX を取得または削除するために使用します。

### **R\_datalib (IRRSDL00): OCSF データ・ライブラリー**

CICS 鍵リングから証明書の情報を取り出すために使用します。

### **R\_dcekey (IRRSK00): RACF 以外のパスワードの検索と設定**

LDAP 処理で使用します。

### **R\_GenSec (IRRSGS00): 汎用セキュリティ API インターフェース**

Kerberos サポートに使用します。CICS は、**VERIFY TOKEN** および **SIGNON TOKEN** API コマンドを介して、また、Web サービス構成を介して、Kerberos サポートを提供します。

### **R\_kerbinfo (IRRSMK00): セキュリティ・サーバー・ネットワーク認証の検索または設定**

領域のプリンシパル名を取得するために Kerberos サポートに使用されます。

### **R\_ticketserv (IRRSRK00): 構文解析または抽出**

Kerberos サポートに使用します。CICS は、**VERIFY TOKEN** および **SIGNON TOKEN** API コマンドを介して、また、Web サービス構成を介して、Kerberos サポートを提供します。

### **R\_usermap (IRRSIM00): アプリケーション・ユーザーのマップ**

Kerberos の検査で Kerberos トークンに関連付けられているユーザーを取得するために使用します。CICS は、**VERIFY TOKEN** および **SIGNON TOKEN** API コマンドを介して、また、Web サービス構成を介して、Kerberos サポートを提供します。

### **R\_Password (IRRSRW00): 平文パスワードまたはパスワード・フレーズの評価または暗号化**

**VERIFY PASSWORD** および **VERIFY PHRASE** API コマンド、および、PASSWORD または PHRASE を指定した **SIGNON** API コマンドに使用します (注 1 を参照してください)。

これらの呼び出しの詳細については、[z/OS Security Server RACF 呼び出し可能サービスを参照してください](#)。

注:

1. APAR CA43999 が適用された PTF が含まれる z/OS 2.2 または z/OS 2.1 が必要です。

## **非端末トランザクションの接続検査の抑止**

CICS は、トランザクションに関連する端末がない場合でも、常に各トランザクション接続に対してトランザクション接続セキュリティ検査を実行します。ただし、端末に接続されたトランザクションに対する完全なトランザクション接続セキュリティを保持し続ける一方で、非端末トランザクションに対するトランザクション接続セキュリティ検査をバイパスできます。

CICS では常に RACROUTE REQUEST=FASTAUTH によるトランザクション接続リソース検査が実行されるので、ICHRFX01 ユーザー出口を提供するだけで済みます。ICHRFX01 ルーチンは、リソース検査処理を続行することを示すゼロの戻りコードか、検査を成功と見なすことを示す戻りコード 8 を発行する必要があります。

ICHRFX01 出口が呼び出された状況を判別できるようにするため、トランザクション接続セキュリティを制御する CICS 領域について ESMEXITS=INSTLN システム初期設定パラメーターを指定します。その後、ICHRFX01 ルーチンは以下の処理を行う必要があります。

1. CICS インストール・データ・パラメーター・リストのアドレスを取得します (ESM 出口プログラムによる CICS 関連情報へのアクセス方法を参照)。このアドレスがゼロの場合、RACROUTE マクロの呼び出し元は CICS でないか、あるいは動作を変更したくない CICS 領域であるため、ゼロの戻りコードで終了します。
2. DFHXSUXP マクロを使用して、インストール・データ・パラメーター・リスト内のフィールドをマップします。
3. インストール・データが CICS によって作成されたことを確認するため、UXPDFHXS が「DFHXS」に等しいかどうかを調べます。等しくない場合は、ゼロの戻りコードで終了します。
4. インストール・データのフィールド UXPPHASE を調べます。値が USER\_ATTACH\_CHECK (X'40') に等しくない場合、これはトランザクション接続でないため、ゼロの戻りコードで終了します。
5. インストール・データのフィールド UXPTerm を調べます。値がゼロ以外の場合、これは端末関連のトランザクション接続であるため、ゼロの戻りコードで終了します。
6. UXPPHASE が USER\_ATTACH\_CHECK で、UXPTerm がゼロの場合は、非端末トランザクションが接続されています。この検査が成功したことを RACF に示すために、戻りコード 8 で終了します。その後、機能 RACROUTE REQUEST=FASTAUTH はゼロの戻りコードで完了し、CICS は非端末トランザクションの接続を続行します。

## サインオンおよびサインオフのグローバル・ユーザー出口

CICS は、EXEC CICS SIGNON 処理で XSNON グローバル・ユーザー出口を提供し、EXEC CICS SIGNOFF 処理で XSNOFF グローバル・ユーザー出口を提供します。これらの出口ではサインオンまたはサインオフの結果に影響を及ぼすことはできませんが、端末に関連付けられたユーザー ID が変更された場合に通知されます。

これらの出口については、グローバル・ユーザー出口プログラムに詳しい説明があります。



# 付録 A z/OS Communications Server LOGON モード・ テーブルのエントリーのコーディング

端末が自動的にインストールされるようにするには、z/OS Communications Server LOGON モード・テーブルを正しくコーディングする必要があります。

## z/OS Communications Server LOGON モード・テーブルの概要

CICS は、自動インストール (autoinstall) 要求を処理するときに、z/OS Communications Server LOGON モード・テーブルにコード化したデータを使用します。自動インストールは、z/OS Communications Server に定義したログモード・エントリーが、CICS に指定した TYPETERM およびモデル TERMINAL 定義の間で一致する場合のみ、適切に機能します。

候補となるさまざまな端末装置において、自動インストールを使用する場合に、ログモード・テーブルを定義する z/OS Communications Server MODEENT マクロに何をコード化する必要があるかを、以下の表で示します。それらの間で、MODEENT マクロのオペランドごとに指定する必要がある値を示しています。CICS にとってオペランドの値のすべてのビット設定値に意味がある場合には、データは 16 進形式で表示されています。オペランドの一部のビット設定値は CICS にとって意味がない場合には、データ・バイトはビット・パターンとして表示されています。CICS にとって意味があるビット設定値は、CICS が予想する値に設定されて表示されています。CICS にとって意味がないビットは、ピリオド(.)として表示されています。したがって、例えば

```
01..0011
```

は、対象バイトの 6 ビットには特定の値を指定する必要があることを示しています。残りの 2 ビットには意味がありません。

ここに示した例のいくつかは、ISTINCLM という CICS 提供の LOGON モード・テーブルのエントリーに正確に対応します。これに該当する場合、表では ISTINCLM でのエントリーの名前を記載しています。

PSERVIC 設定は、aaaaaaaa, bbbbbbbb (以下続く) というフィールドを示しています。これらのフィールドの内容は、端末の特定の属性をどのように指定したかによって、LUTYPE0、LUTYPE2、および LUTYPE3 装置で異なります。LUTYPEx デバイスに関する PSERVIC 画面サイズ値を参考にして、必要な値を設定できます。

## z/OS Communications Server MODEENT マクロ・オペランド

z/OS Communications Server LOGON モード・テーブルは、関連した CICS TYPETERM リソースに必要な MODEENT マクロ・エントリーをリストします。

表の左側の、356 ページの表 31 に基づく参照番号 (RN) を下に向かってご覧ください。番号が見つかったら、その行の真ん中の列をご覧ください。この列には、CICS で自動インストールが処理される方法に影響を与えるマクロ・オペランドが示されています。インストールするデバイスに関する MODEENT マクロ・エントリーは、この列で指定されているマクロ・オペランドと一致していなければなりません。一部の参照番号に対応する PSERVIC などの、表に示されていない MODEENT マクロ・エントリーは、CICS によるテストが行われていません。自動インストールされる端末に関するバインド分析中に CICS で処理されないビット設定は、ピリオド(.)として表わされています。

注: LUTYPE0、LUTYPE2、LUTYPE3 デバイスに関する PSERVIC データ内の一部のフィールドには、そのデバイスの ALTSCREEN 特性と DEFSCREEN 特性に応じて異なる値があります。この理由から、358 ページの『LUTYPEx デバイスに関する PSERVIC 画面サイズ値』を参照して、aaaaaaaa, bbbbbbbb, cccccccc, dddddddd, eeeeeeee の代わりに指定する必要がある値を見つけてください。

表の右側の列は、ニーズに見合う可能性がある、用意されている LOGON モード・テーブル内のエントリーの名前を示しています。用意されているテーブルの名前は ISTINCLM です。

表 30. LOGON モード・テーブルと ISTINCLM エントリー

RN	関連した CICS TYPETERM 定義に必要な z/OS Communications Server MODEENT マクロ・エントリー	用意されている エントリーのう ち適切なもの
1	FMPROF=X'02' TSPROF=X'02' PRIPROT=X'70' SECPROT=X'40' COMPROT=B'0000.000 00000.00'	
2	FMPROF=X'02' TSPROF=X'02' PRIPROT=X'71' SECPROT=X'40' COMPROT=B'0010.000 00000.00'	DSILGMOD D4B32781 D4B32782 D4B32783 D4B32784 D4B32785 NSX32702 S3270
3	FMPROF=X'04' TSPROF=X'04' PRIPROT=X'B0' SECPROT=X'B0' COMPROT=B'0000.000 00000.00'	
4	FMPROF=X'04' TSPROF=X'03' PRIPROT=X'B0' SECPROT=X'90' COMPROT=B'0100.000 00000.00'	
5	FMPROF=X'04' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0110.000 00000.00'	
6	FMPROF=X'04' TSPROF=X'04' PRIPROT=X'31' SECPROT=X'30' COMPROT=B'0110.000 00000.00'	INTRUSER
7	FMPROF=X'04' TSPROF=X'04' PRIPROT=X'B0' SECPROT=X'30' COMPROT=B'0100.000 00000.00'	
8	FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0011.000 01000.00' PSERVIC=B'00000001 00000000 00000000 00000000. ..... 00000000 00000000 00000000 00000000 ..... 00000000 00000000'	



表 30. LOGON モード・テーブルと ISTINCLM エントリー (続き)

RN	関連した CICS TYPETERM 定義に必要な z/OS Communications Server MODEENT マクロ・エントリー	用意されている エントリーのう ち適切なもの
9	<pre> FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0011.000 10000.00' PSERVIC=B'00000001 00000000 00000000 00000000. ..... 00000000 00000000 00000000 00000000 ..... 00000000 00000000' </pre>	SCS
10	<pre> FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0011.000 01000.00' PSERVIC=X'01' </pre>	
11	<pre> FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0011.000 10000.00' PSERVIC=X'01' </pre>	SCS  注 2 を参照
12	<pre> FMPROF=X'07' TSPROF=X'07' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0101.000 10000.01' PSERVIC=B'00000100 10101000 01000000 10100000 ..... 10101000 01000000 10100000 00000000 ..... 00001100 00000000' </pre>	
13	<pre> FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0011.000 10000.00' PSERVIC=X'01' </pre>	SCS3790  注 2 を参照
14	<pre> FMPROF=X'03' TSPROF=X'04' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0111.000 10000.00' PSERVIC=B'00000001 00110001 00011000 01000000. ..... 00000000 10010010 00000000 00000000 ..... 00000000 01010000' </pre>	
15	<pre> FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0111.000 10000.00' PSERVIC=B'00000001 00110001 00001100 01110000. ..... 00000000 11010010 00000000 00000000 ..... 00000000 11010000' </pre>	

表 30. LOGON モード・テーブルと ISTINCLM エントリー (続き)

RN	関連した CICS TYPETERM 定義に必要な z/OS Communications Server MODEENT マクロ・エントリー	用意されている エントリーのう ち適切なもの
16	FMPROF=X'04' TSPROF=X'04' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0111.000 10000.00'	注 3 を参照
17	FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=X'90' COMPROT=B'0111.000 10000.00' PSERVIC=B'00000001 00100000 00000000 00000000. ..... 00000000 11000010 00000000 00000000 ..... 00000000 11000000'	
18	FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=B'10..0000' COMPROT=B'0011.000 10000.00' PSERVIC=B'00000010 10000000 00000000 00000000 00000000 00000000 aaaaaaaa bbbbbbbb cccccccc dddddddd eeeeeeee'	D329001 D4A32771 D4A32772 D4A32781 D4A32782 D4A32783 D4A32784 D4A32785 D4C32771 D4C32772 D4C32781 D4C32782 D4C32783 D4C32784 D4C32785 D6327801 D6327802 D6327803 D6327804 D6327805 EMUDPCX EMU3790 SNX32702  注 1 を参照
19	FMPROF=X'03' TSPROF=X'03' PRIPROT=X'B1' SECPROT=B'10..0000' COMPROT=B'0011.000 10000.00' PSERVIC=B'00000011 10000000 00000000 00000000 00000000 00000000 aaaaaaaa bbbbbbbb cccccccc dddddddd eeeeeeee'	BLK3790 DSC2K DSC4K D6328902 D6328904  注 1 を参照

表 30. LOGON モード・テーブルと ISTINCLM エントリー (続き)

RN	関連した CICS TYPETERM 定義に必要な z/OS Communications Server MODEENT マクロ・エントリー	用意されている エントリーのう ち適切なもの
20	<pre> FMPROF=X'04' TSPROF=X'03' PRIPROT=X'31' SECPROT=X'B0' COMPROT=B'0111.000' </pre>	
21	<pre> FMPROF=X'04' TSPROF=X'04' PRIPROT=X'50' SECPROT=X'10' COMPROT=B'0000.000 00000.00' </pre>	
22	<pre> FMPROF=X'04' TSPROF=X'04' PRIPROT=X'B0' SECPROT=X'B0' COMPROT=B'0100.000 00000.00' </pre>	IBMS3650
23	<pre> FMPROF=X'04' TSPROF=X'04' PRIPROT=X'B1' SECPROT=X'B0' COMPROT=B'0111.000 00000.00' </pre>	
24	<pre> TYPE=X'00' FMPROF=X'13' TSPROF=X'07' PRIPROT=X'B0' SECPROT=X'B0' COMPROT=B'.101.000 10110.01' PSERVIC=B'00000110 00000010           ..... 00000000 00000000 00000000           ..... 00000000 00000000 00000000           ..... 0010..00' </pre>	

注:

1. PSERVIC (RN 18 と 19): 拡張データ・ストリーム (EXTDS) サポートが必要な位置では BYTE 2 BIT 0 を設定する必要があります。
2. RN 11 または 13 は、デバイス SCSPRINT の MODEENT マクロ・オペランドの判別に使用します。しかし、TYPETERM に属性 EXTENDED DS、COLOR、PROGSYMBOLS、HIGHLIGHT、SOSI、OUTLINE、QUERY(COLD)、または QUERY(ALL) を指定している場合は、COMPROT=B'0111.000 10000.00' を読み取るように RN 13 の COMPROT パラメーターを変更する必要があります。
3. この LOGMODE は、半二重モードのデバイス・タイプ 4700 か、SESSIONTYPE (USERPROG) が指定されたデバイス・タイプ BCHLU、3770、3770B、3790 の場合に使用できます。正しいモデルを使用してこれらのデバイスを自動インストールできるようにするために、自動インストール出口に提供されるモデル名リストでは、DEVICE(3600) として定義されているモデルの名前が、その他の適格なすべてのモデル名の後にリストされます。4700 半二重デバイスのリストの末尾から名前を選択するように、この出口をコーディングできます。

## TYPETERM デバイス・タイプと関連 LOGON モード・データを指すポインター

TYPETERM デバイスのタイプごとに、z/OS Communications Server MODEENT マクロ上でコーディングする必要のある参照番号があります。この情報は、自動インストールする端末を判別する際に使用できます。

356 ページの表 31 は TYPETERM デバイス・タイプの完全なリストです。自動インストールには使用できないタイプも含まれています。使用できないタイプにはアスタリスク (\*) が付いています。TYPETERM 定義のコーディングについての詳細と、自動インストールできる端末のリストについては、[z/OS Communications Server 端末の自動インストール](#)を参照してください。

表 31. TYPETERM デバイス・タイプと、z/OS Communications Server ログモード・エントリーに対する相互参照	
TYPETERM デバイス・タイプ	352 ページの表 30 の参照番号
DEVICE(APPC)	24
DEVICE(BCHLU)	17
DEVICE(BCHLU) SESSIONTYPE(BATCHDI)	15
DEVICE(BCHLU) SESSIONTYPE(USERPROG)	16
DEVICE(CONTLU)	10
DEVICE(INTLU)	11
DEVICE(LUTYPE2)	18
DEVICE(LUTYPE2) TERMMODEL(1)	18
DEVICE(LUTYPE3)	19
DEVICE(LUTYPE3) TERMMODEL(1)	19
DEVICE(LUTYPE4)	12
DEVICE(SCSPRINT)	11, 13
DEVICE(TLX)	8
DEVICE(TLX) SESSIONTYPE(CONTLU)	8
DEVICE(TLX) SESSIONTYPE(INTLU)	9
DEVICE(TWX)	8
DEVICE(TWX) SESSIONTYPE(CONTLU)	8
DEVICE(TWX) SESSIONTYPE(INTLU)	9
DEVICE(3270)	2
DEVICE(3270) BRACKET(NO)	1
DEVICE(3270) TERMMODEL(1)	2
DEVICE(3270) TERMMODEL(1) BRACKET(NO)	1
DEVICE(3270P)	2
DEVICE(3270P) BRACKET(NO)	1
DEVICE(3270P) TERMMODEL(1)	2
DEVICE(3270P) TERMMODEL(1) BRACKET(NO)	1
DEVICE(3275)	2

表 31. TYPETERM デバイス・タイプと、z/OS Communications Server ログモード・エントリーに対する相互参照 (続き)

TYPETERM デバイス・タイプ	<a href="#">352 ページの表 30</a> の参照番号
DEVICE(3275) BRACKET(NO)	1
DEVICE(3275) TERMMODEL(1)	2
DEVICE(3275) TERMMODEL(1) BRACKET(NO)	1
DEVICE(3600)	16, 22, 23
DEVICE(3600) SESSIONTYPE(PIPELINE) *	21
DEVICE(3600) SESSIONTYPE(PIPELN) *	21
DEVICE(3614) *	3
DEVICE(3650) SESSIONTYPE(PIPELINE) *	21
DEVICE(3650) SESSIONTYPE(PIPELN) *	21
DEVICE(3650) SESSIONTYPE(USERPROG) BRACKET(YES)	6
DEVICE(3650) SESSIONTYPE(USERPROG) BRACKET(NO)	7
DEVICE(3650) SESSIONTYPE(3270)	5
DEVICE(3650) SESSIONTYPE(3270) BRACKET(NO)	4
DEVICE(3650) SESSIONTYPE(3653)	5
DEVICE(3650) SESSIONTYPE(3653) BRACKET(NO)	4
DEVICE(3767)	11
DEVICE(3767C)	10
DEVICE(3767I)	11
DEVICE(3770)	17
DEVICE(3770) SESSIONTYPE(BATCHDI)	15
DEVICE(3770) SESSIONTYPE(USERPROG)	16
DEVICE(3770B)	17
DEVICE(3770B) SESSIONTYPE(BATCHDI)	15
DEVICE(3770B) SESSIONTYPE(USERPROG)	16
DEVICE(3770C)	10
DEVICE(3770I)	11
DEVICE(3790)	20
DEVICE(3790) SESSIONTYPE(BATCHDI)	14
DEVICE(3790) SESSIONTYPE(SCSPRT)	13
DEVICE(3790) SESSIONTYPE(SCSPRINT)	13
DEVICE(3790) SESSIONTYPE(USERPROG)	16
DEVICE(3790) SESSIONTYPE(3277CM)	18
DEVICE(3790) SESSIONTYPE(3284CM)	19

表 31. TYPETERM デバイス・タイプと、z/OS Communications Server ログモード・エントリーに対する相互参照 (続き)	
<b>TYPETERM デバイス・タイプ</b>	<b>352 ページの表 30 の参照番号</b>
DEVICE(3790) SESSIONTYPE(3286CM)	19

## LUTYPEx デバイスに関する PSERVIC 画面サイズ値

自動インストール・モデル・デバイス定義オプション・テーブルを使用して、z/OS Communications Server MODEENT マクロの PSERVIC オペランド上で指定する LUTYPE0、LUTYPE2、LUTYPE3 デバイスの画面サイズ値の判別に役立てることができます。

CICS TYPETERM 定義で、358 ページの表 32 の 1 列目から 4 列目に示されている値をコーディングすると、CICS モデル・バインド・イメージの画面サイズ値は 5 列目の値になります。この値と一致する値を、PSERVIC オペランド上の画面サイズ値としてコーディングしなければなりません。

PSERVIC 画面サイズの仕様で違う値がコーディングされていても、CICS で同等に扱われる値があります。359 ページの表 33 を参照してください。

表 32. 自動インストール・モデル・デバイス定義オプション				
デバイス・タイプ	DEFSCRN	ALTSCRN	QUERY	MODEL BIND
0,2,3	00,00	?	?	INVALID
0,2,3	12,40	,	?	0000000001
0,2,3	12,40	00,00	?	0C2800007E
0,2,3	12,40	YY,YY	?	0C28YYYY7F
0,2,3	24,80	,	NO	0000000002
3	24,80	,	COLD/ALL	0000000002
0,2	24,80	,	COLD/ALL	0000000003
0,2,3	24,80	00,00	?	185000007E
0,2,3	24,80	YY,YY	?	1850YYYY7F
0,2,3	XX,XX	,	?	XXXX00007E
0,2,3	XX,XX	00,00	?	XXXX00007E
0,2,3	XX,XX	YY,YY	?	XXXXYYYY7F

ここで、

**0**

ローカルの非 SNA 3270 を示します

**2**

LUTYPE2 を示します

**3**

LUTYPE3 を示します

,

デフォルトを示します

**XX,XX**

12,40 と 24,80 以外の画面サイズを示します

**YY,YY**

00,00 とブランク以外の画面サイズを示します



?

任意を意味します (つまり QUERY=ALL|COLD|NO や ALTSCRN=any)

表 33. 同等の PSERVIC 画面サイズ値	
CICS モデル・バインドの 20 - 24 バイト	PSERVIC 定義上の有効な画面サイズ値
0000 0000 01	0000 0000 00 0000 0000 01 0C28 0000 7E
0000 0000 02	0000 0000 00 0000 0000 02 1850 0000 7E
0000 0000 03	0000 0000 00 0000 0000 03 1850 0000 03
xxxx 0000 7E で、xxxx=1850 の場合	0000 0000 00 xxxx 0000 7E 0000 0000 02
xxxx yyyy 7F	0000 0000 00 xxxx yyyy 7F

ここで、

**xxxx**

デフォルト画面サイズが含まれる 16 進数の 2 バイトを示します

**yyyy**

代替画面サイズが含まれる 16 進数の 2 バイトを示します

## 対応するモデルと LOGON モード・エンタリー

このセクションには、一連の z/OS Communications Server LOGON モード・テーブル定義と、対応する CICS 自動インストール定義が示されています。各エンタリーは、z/OS Communications Server ログモード定義、対応する CICS の TYPETERM 定義とモデルの TERMINAL 定義、指定されたモデル定義に基づいて CICS が送信する BIND (通知用) で構成されます。

CICS 固有の属性は完全に任意であることに注意してください。デバイス属性だけがマッチング・アルゴリズムに影響を及ぼします。自動インストール・ユーザー・プログラムで、対応するモデルの区別を行う必要があります。

```
*****
1) LOCAL NON-SNA 3277 / 3278 / 3279 (without special features)
*****
MT32772  MODEENT LOGMODE=MT32772,  3277/8          MODEL 2
          TYPE=1,
          FMPROF=X'02',
          TSPROF=X'02',
          PRIPROT=X'71',
          SECPROT=X'40',
          COMPROT=X'2000',
          PSERVIC=X'000000000000000000000000200'
OR
          PSERVIC=X'00000000000000000000000018502B507F00' Others
OR
          PSERVIC=X'000000000000000000000000185000007E00' Model 2, no Altscreen
```

### TERMINAL 定義

```
*****
AUTINSTNAME  ==> M3278A
AUTINSTMODEL ==> ONLY
GROUP        ==> PDATD
TYPETERM     ==> T3278
INSERVICE    ==> YES
```

### TYPETERM 定義

```
*****
TYPETERM     ==> T3278
GROUP        ==> PDATD
DEVICE       ==> 3270
TERMMODEL    ==> 2
LIGHTPEN     ==> YES
AUDIBLEALARM ==> YES
UCTRAN       ==> YES
IOAREALEN    ==> 2000,2000
ERRLASTLINE  ==> YES
```

ERRINTENSIFY	==>	YES
USERAREALEN	==>	32
ATI	==>	YES
TTI	==>	YES
AUTOCONNECT	==>	NO
LOGONMSG	==>	YES

```

BIND SENT BY CICS depends on PSERVIC value on LOGMODE definition above:
EITHER      :      01020271 40200000 00000080 00000000
              00000000 00000002 00009300 00300000
OR          :      01020271 40200000 00000080 00000000
              00000018 502B507F 00009300 00300000
OR          :      01020271 40200000 00000080 00000000
Real Model 2 00000018 5000007E 00009300 00300000

```

```
*****
2) LOCAL SNA 3277/78/79 (without special features) LUTYPE2
*****
S32782    MODEENT LOGMODE=S32782,    SNA LUTYPE2 3270
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'B0',
          COMPROT=X'3080',
          RUSIZES=X'8585',
          PSERVIC=X'028000000000185018507F00'
```

```

TERMINAL 定義
*****
AUTINSTNAME  ==> M32782
AUTINSTMODEL ==> ONLY
GROUP         ==> PDATD
TYPETERM      ==> T32782
INSERVICE     ==> YES

```

```

TYPETERM 定義
*****
TYPETERM      ==>  T32782
GROUP          ==>  PDATD
DEVICE         ==>  LUTYPE2
TERMMODEL     ==>  2
LIGHTPEN      ==>  YES
AUDIBLEALARM  ==>  YES
UCTRAN        ==>  YES
IOAREALEN     ==>  256, 256
ERRRLASTLINE  ==>  YES
ERRINTENSIFY  ==>  YES
USERAREALEN   ==>  32
ATI           ==>  YES
TTI           ==>  YES
LOGONMSG      ==>  YES
DISCREQ       ==>  YES
RECEIVESIZE   ==>  256
BUILDCHAIN    ==>  YES

```

```
BIND SENT BY CICS :      010303B1 B0308000 0085C780 00028000  
                        00000018 5018507F 00000000 00000000
```

```

*****
3) 3770 BATCH LU (3777)
*****
BATCH      MODEENT LOGMODE=BATCH,      3770 BATCH
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'B0',
          COMPROT=X'7080',
          PSERVIC=X'01310C70E100D20000E100D0'

```

```

TERMINAL 定義
*****
AUTINSTNAME ==> M3770

```

```

AUTINSTMODEL ==> ONLY
GROUP        ==> PDATD
TYPETERM     ==> T3770
INSERVICE    ==> YES

```

#### TYPETERM 定義

\*\*\*\*\*

```

TYPETERM     ==> T3770
GROUP        ==> PDATD
DEVICE       ==> 3770
SESSIONTYPE  ==> BATCHDI
PAGESIZE     ==> 12,80
DISCREQ      ==> YES
AUTOPAGE     ==> YES
RECEIVESIZE  ==> 256
SENDSIZE     ==> 256
IOAREALEN    ==> 256,2048
BUILDCHAIN   ==> YES
BRACKET      ==> YES
ATI          ==> YES
TTI          ==> YES
AUTOCONNECT  ==> NO
HORIZFORM    ==> YES
VERTFORM     ==> YES
LDCLIST      ==> LDC2
Needs LDC declaration in TCT :
LDC2  DFHTCT TYPE=LDC,LOCAL=INITIAL
      DFHTCT TYPE=LDC,LDC=BCHLU
      DFHTCT TYPE=LDC,LOCAL=FINAL

```

```

BIND SENT BY CICS :      010303B1 B0708000 00000080 0001310C
                        70E100D2 0000E100 D0000000 00000000

```

\*\*\*\*\*  
4) 6670 LUTYPE4

\*\*\*\*\*

```

S6670  MODEENT LOGMODE=S6670,      6670 LUTYPE4
      TYPE=1,
      FMPROF=X'07',
      TSPROF=X'07',
      RUSIZES=X'8585',
      PRIPROT=X'B1',
      SECPROT=X'B0',
      COMPROT=X'5081',
      PSERVIC=X'04A840A000A840A0000000C00'

```

#### TERMINAL 定義

\*\*\*\*\*

```

AUTINSTNAME  ==> M6670
AUTINSTMODEL ==> ONLY
GROUP        ==> PDATD
TYPETERM     ==> T6670
INSERVICE    ==> YES

```

#### TYPETERM 定義

\*\*\*\*\*

```

TYPETERM     ==> T6670
GROUP        ==> PDATD
DEVICE       ==> LUTYPE4
BUILDCHAIN   ==> YES
DISCREQ      ==> YES
RECEIVESIZE  ==> 256
UCTRAN       ==> YES
IOAREALEN    ==> 256,4096
FORMFEED     ==> YES
HORIZFORM    ==> YES
VERTFORM     ==> YES
ATI          ==> YES
TTI          ==> YES
PAGESIZE     ==> 50,80
AUTOPAGE     ==> YES
LOGONMSG     ==> NO
LDCLIST      ==> LDC1

```

```
Needs LDC declaration in TCT :
LDCS      DFHTCT TYPE=LDC,LDC=SYSTEM
LDC1      DFHTCT TYPE=LDC,LOCAL=INITIAL
          DFHTCT TYPE=LDC,DVC=(BLUCON,01),PROFILE=DEFAULT,LDC=PC,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(BLUPRT,02),PROFILE=BASE,LDC=PP,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(BLUPRT,08),PROFILE=BASE,LDC=P8,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(BLUPRT,08),PROFILE=DEFAULT,LDC=DP,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(BLUPCH,03),PROFILE=JOB,LDC=PM,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(BLUPCH,03),PROFILE=DEFAULT,LDC=DM,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED1,04),PROFILE=WPRAW,LDC=P1,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED1,04),PROFILE=DEFAULT,LDC=D1,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED2,05),PROFILE=OII1,LDC=P2,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED2,05),PROFILE=DEFAULT,LDC=D2,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED3,06),PROFILE=OII2,LDC=P3,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,DVC=(WPMED4,07),PROFILE=OII3,LDC=P4,
            PGESIZE=(50,80),PGESTAT=AUTOPAGE
          DFHTCT TYPE=LDC,LOCAL=FINAL
```

```
BIND SENT BY CICS :      010707B1 B0508100 00858580 0004A840
                        A000A840 A000000C 00000000 00000000
```

```
*****
5) 3790 FULL FUNCTION LU
*****
S3790A  MODEENT LOGMODE=S3790A,  3790 FULL FUNCTION LU
          TYPE=1,
          FMPROF=X'04',
          TSPROF=X'04',
          PRIPROT=X'B1',
          SECPROT=X'B0',
          RUSIZES=X'8585',
          COMPROT=X'7080'
```

```
TERMINAL 定義
*****
AUTINSTNAME ==> M3790A
AUTINSTMODEL ==> ONLY
GROUP       ==> PDATD
TYPETERM    ==> T3790A
INSERVICE   ==> YES
```

```
TYPETERM 定義
*****
TYPETERM    ==> T3790A
GROUP       ==> PDATD
DEVICE      ==> 3790
SENDSIZE    ==> 256
RECEIVESIZE ==> 256
SESSIONTYPE ==> USERPROG
BRACKET     ==> YES
IOAREALEN   ==> 256
ATI         ==> YES
TTI         ==> YES
```

```
BIND SENT BY CICS :      010404B1 B0708000 00858580 00000000
```

```
*****
6) 3790 BATCH DATA INTERCHANGE
*****
S3790B  MODEENT LOGMODE=S3790B,  3790 BATCH
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'04',
          PRIPROT=X'B1',
```

```
SECROT=X'B0',
COMPROT=X'7080',
RUSIZES=X'8585',
PSERVIC=X'013118400000920000E10050'
```

#### TERMINAL 定義

```
*****
AUTINSTNAME ==> M3790B
AUTINSTMODEL ==> ONLY
GROUP ==> PDATD
TYPETERM ==> T3790B
INSERVICE ==> YES
TERMPRIORITY ==> 50
```

#### TYPETERM 定義

```
*****
TYPETERM ==> T3790B
GROUP ==> PDATD
DEVICE ==> 3790
SESSIONTYPE ==> BATCHDI
AUTOPAGE ==> YES
BUILDCHAIN ==> YES
OBOPERID ==> YES
IOAREALEN ==> 256,2048
RELREQ ==> YES
SENDSIZE ==> 256
RECEIVESIZE ==> 256
ATI ==> YES
TTI ==> YES
LDCLIST ==> LDC2
```

Needs LDC declaration in TCT :

```
LDC2 DFHTCT TYPE=LDC,LOCAL=INITIAL
      DFHTCT TYPE=LDC,LDC=BCHLU
      DFHTCT TYPE=LDC,LOCAL=FINAL
```

```
BIND SENT BY CICS :      010304B1 B0708000 00858580 00013118
                        40000092 0000E100 50000000 00000000
```

```
*****
7) 3790 SCSPT
*****
S3790C  MODEENT LOGMODE=S3790C,  3790 WITH SCS
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECROT=X'B0',
          COMPROT=X'3080',
          RUSIZES=X'8585',
          PSERVIC=X'010000000000000000000000'
```

#### TERMINAL 定義

```
*****
AUTINSTNAME ==> M3790C
AUTINSTMODEL ==> ONLY
GROUP ==> PDATD
TYPETERM ==> T3790C
INSERVICE ==> YES
```

#### TYPETERM 定義

```
*****
TYPETERM ==> T3790C
GROUP ==> PDATD
DEVICE ==> 3790
SESSIONTYPE ==> SCSPT
BRACKET ==> YES
SENDSIZE ==> 256
RECEIVESIZE ==> 256
ATI ==> YES
TTI ==> YES
```

Note that CEDA changes DEVICE=3790,  
SESSIONTYPE=SCSPRT to DEVICE=SCSPRINT,  
SESSIONTYPE=blanks, PRINTERTYPE=3284.

```
BIND SENT BY CICS :      010303B1 B0308000 00858580 00010000
```

```

*****
8) 3767 INTERACTIVE (FLIP-FLOP) LU
*****
S3767    MODEENT LOGMODE=S3767,    3767 INTERACTIVE
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'90',
          COMPROT=X'3080',
          PSERVIC=X'010000000000000000000000'

```

#### TERMINAL 定義

```

*****
AUTINSTNAME ==> M3767
AUTINSTMODEL ==> ONLY
GROUP ==> PDATD
TERMPRIORITY ==> 60
TYPETERM ==> T3767
INSERVICE ==> YES

```

#### TYPETERM 定義

```

*****
TYPETERM ==> T3767
GROUP ==> PDATD
DEVICE ==> 3767
VERTFORM ==> YES
HORIZFORM ==> YES
RELREQ ==> YES
DISCREQ ==> YES
IOAREALEN ==> 256
AUTOPAGE ==> NO
PAGESIZE ==> 12,80
ATI ==> YES
TTI ==> YES
BRACKET ==> YES
RECEIVESIZE ==> 256
SENDSIZE ==> 256

```

```

BIND SENT BY CICS :          010303B1 90308000 00000080 00010000

```

```

*****
9) 3650 INTERPRETER LU
   (SESTYPE = USERPROG BRACKET = YES)
*****
S3650A    MODEENT LOGMODE=S3650A,    3650 SESTYPE=USERPROG
          TYPE=1,                      BRACKET=YES
          FMPROF=X'04',
          TSPROF=X'04',
          PRIPROT=X'31',
          SECPROT=X'30',
          COMPROT=X'6000'

```

#### TERMINAL 定義

```

*****
AUTINSTNAME ==> M3650A
AUTINSTMODEL ==> ONLY
GROUP ==> PDATD
TYPETERM ==> T3650A
INSERVICE ==> YES

```

#### TYPETERM 定義

```

*****
TYPETERM ==> T3650A
GROUP ==> PDATD
DEVICE ==> 3650
SESSIONTYPE ==> USERPROG
ROUTEDMSGSGS ==> SPECIFIC
FMHPARM ==> YES
RELREQ ==> YES
DISCREQ ==> YES
BRACKET ==> YES
RECEIVESIZE ==> 256
IOAREALEN ==> 256,256
ATI ==> YES

```



TTI ==> YES  
AUTOCONNECT ==> NO

BIND SENT BY CICS : 01040431 30600000 00000080 00000000

\*\*\*\*\*  
10) 3650 HOST CONVERSATIONAL (3270) LU  
\*\*\*\*\*  
S3650B MODEENT LOGMODE=S3650B, 3650 SESTYPE=3270  
TYPE=1, AND SESTYPE=3653  
FMPROF=X'04',  
TSPROF=X'03',  
PRIPROT=X'B1',  
SECPROT=X'90',  
COMPROT=X'6000'

#### TERMINAL 定義

\*\*\*\*\*  
AUTINSTNAME ==> M3650B1  
AUTINSTMODEL ==> ONLY  
GROUP ==> PDATD  
TYPETERM ==> T3650B1  
INSERVICE ==> YES

#### TYPETERM 定義

\*\*\*\*\*  
TYPETERM ==> T3650B1  
GROUP ==> PDATD  
DEVICE ==> 3650  
OBFORMAT ==> YES  
SESSIONTYPE ==> 3270  
RELREQ ==> YES  
DISCREQ ==> YES  
IOAREALEN ==> 256  
BRACKET ==> YES  
RECEIVESIZE ==> 240  
ATI ==> NO  
TTI ==> YES

BIND SENT BY CICS : 010403B1 90600000 00000080 00000000

\*\*\*\*\*  
11) 3650 HOST CONVERSATIONAL (3653) LU  
(N.B. LOGMODE SAME AS HC (3270) LU)  
\*\*\*\*\*  
S3650B MODEENT LOGMODE=S3650B, 3650 SESTYPE=3270  
TYPE=1, AND SESTYPE=3653  
FMPROF=X'04',  
TSPROF=X'03',  
PRIPROT=X'B1',  
SECPROT=X'90',  
COMPROT=X'6000'

#### TERMINAL 定義

\*\*\*\*\*  
AUTINSTNAME ==> M3650B2  
AUTINSTMODEL ==> ONLY  
GROUP ==> PDATD  
TYPETERM ==> T3650B2  
INSERVICE ==> YES

#### TYPETERM 定義

\*\*\*\*\*  
TYPETERM ==> T3650B2  
GROUP ==> PDATD  
DEVICE ==> 3650  
SESSIONTYPE ==> 3653  
RELREQ ==> YES  
DISCREQ ==> NO  
BRACKET ==> YES  
IOAREALEN ==> 256  
RECEIVESIZE ==> 240  
ROUTEDMSGs ==> NONE

```
ATI      ==> NO
TTI      ==> YES
```

```
BIND SENT BY CICS :      010403B1 90600000 00000080 00000000
```

```
*****
12) 3650 HOST COMMAND PROCESSOR LU
    (SESTYPE = USERPROG  BRACKET = NO)
*****
S3650C  MODEENT LOGMODE=S3650C,    3650 SESTYPE=USERPROG
          TYPE=1,                  BRACKET=NO
          FMPROF=X'04',
          TSPROF=X'04',
          PRIPROT=X'B0',
          SECPROT=X'30',
          COMPROT=X'4000'
```

#### TERMINAL 定義

```
*****
AUTINSTNAME ==> M3650C
AUTINSTMODEL ==> ONLY
GROUP       ==> PDATD
TYPETERM    ==> T3650C
INSERVICE   ==> YES
```

#### TYPETERM 定義

```
*****
TYPETERM    ==> T3650C
GROUP       ==> PDATD
DEVICE      ==> 3650
SESSIONTYPE ==> USERPROG
BRACKET     ==> NO
RELREQ      ==> NO
DISCREQ     ==> NO
RECEIVESIZE ==> 256
IOAREALEN   ==> 256
ATI         ==> YES
TTI         ==> YES
```

```
BIND SENT BY CICS :      01040430 30400000 00000080 00000000
```

```
*****
13) 8815 SCANMASTER (APPC SINGLE SESSION)
*****
SIN62  MODEENT LOGMODE=SIN62,      8815 SCANMASTER.
          TYPE=0,
          FMPROF=X'13',
          TSPROF=X'07',
          PRIPROT=X'B0',
          SECPROT=X'B0',
          COMPROT=X'50B1',
          PSNDPAC=X'00',
          SRCVPAC=X'00',
          SSNDPAC=X'00',
          RUSIZES=X'8585',
          PSERVIC=X'06020000000000000000002C00'
```

#### TERMINAL 定義

```
*****
AUTINSTNAME ==> MLU62
AUTINSTMODEL ==> ONLY
GROUP       ==> PDATD
TYPETERM    ==> SINLU62
INSERVICE   ==> YES
```

#### TYPETERM 定義

```
*****
TYPETERM    ==> SINLU62
GROUP       ==> PDATD
DEVICE      ==> APPC
RECEIVESIZE ==> 2048
SENDSIZE    ==> 2048
ATI         ==> YES
```

```
TTI          ==> YES
Note: There is no RDO keyword equivalent of the MACRO
keyword 'FEATURE=Single', because this is assumed with
RDO DEFINE TYPETERM when DEVICE=APPC.
```

[illegible]

```

*****
14) 3290 (SDLC)
*****
S3290      MODEENT LOGMODE=S3290,      3290 SDLC
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'90',
          COMPROT=X'3080',
          RUSIZES=X'8787',
          PSERVIC=X'02800000000018503EA07F00'

```

```

TERMINAL 定義
*****
AUTINSTNAME  ==> M3290
AUTINSTMODEL ==> ONLY
GROUP        ==> PDATD
TYPETERM     ==> T3290
INSERVICE    ==> YES

```

```

TYPETERM 定義
*****
TYPETERM      ==>  T3290
GROUP          ==>  PDATD
DEVICE         ==>  LUTYPE2
TERMMODEL      ==>  2
ALTSCREEN      ==>  62,160
DEFSCREEN      ==>  24,80
AUDIBLEALARM   ==>  YES
UCTRAN         ==>  YES
IOAREALEN      ==>  2000,2000
ERRLASTLINE    ==>  YES
ERRINTENSIFY   ==>  YES
USERAREALEN    ==>  32
ATI            ==>  YES
TTI            ==>  YES
LOGONMSG       ==>  YES
ERRHLIGHT      ==>  BLINK
RECEIVEIZE     ==>  1024

```

```
BIND SENT BY CICS :      010303B1 90308000 00878780 00028000  
                        00000018 503EA07F 00000000 00000000
```

```
*****
15) 3601 WITH A 3604 ATTACHED
*****
S3600      MODEENT LOGMODE=S3600,      3601
          TYPE=1,
          FMPPROF=X'04',
          TSPROF=X'04',
          PRIPROT=X'B1',
          SECPR0T=X'B0',
          COMPROT=X'7000',
          RUSIZES=X'0000'
```

```

TERMINAL 定義
*****
AUTINSTNAME    ==> M3600
AUTINSTMODEL   ==> ONLY
GROUP          ==> PDATD
TERMPRIORITY    ==> 50

```

```
TYPETERM      ==> T3600
INSERVICE     ==> YES
```

```
TYPETERM 定義
*****
TYPETERM      ==> T3600
GROUP         ==> PDATD
DEVICE        ==> 3600
AUTOPAGE      ==> NO
PAGESIZE      ==> 6,40
RELREQ        ==> YES
DISCREQ       ==> NO
IOAREALEN     ==> 256
SENDSIZE      ==> 224
RECEIVESIZE   ==> 256
USERAREALEN   ==> 100
ATI           ==> NO
TTI           ==> YES
BRACKET       ==> YES
LDCLIST       ==> BMSLLDC1
```

```
Needs LDC declaration in TCT :
BMSLLDC1 DFHTCT TYPE=LDCLIST,
          LDC=(DS,JP,PB=5,LP,MS)
          DFHTCT TYPE=LDC,
          LDC=(DS=1),
          DVC=3604,
          PGESIZE=(6,40),
          PGESTAT=PAGE
          DFHTCT TYPE=LDC,LDC=SYSTEM
```

```
BIND SENT BY CICS :      010404B1 B0700000 00000080 00000000
```

## CICS 提供の自動インストール・モデル用の LOGON モード定義

この章には、自動インストールのための CICS 提供の TYPETERM およびモデル TERMINAL 定義と適合する、z/OS Communications Server LOGON モード・テーブル定義の例が含まれています。

最初の 6 つの項目は定義例です。これらは、z/OS Communications Server に含まれていません。

```
DFHLU3  MODEENT LOGMODE=DFHLU3, LU TYPE 3 PRINTER.
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'B0',
          COMPROT=X'3080',
          RUSIZES=X'8585',
          PSERVIC=X'0380000000000000000000200'
```

```
DFHSCSP MODEENT LOGMODE=DFHSCSP, LU TYPE 1 SCS PRINTER
          TYPE=1,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'B0',
          COMPROT=X'7080',
          RUSIZES=X'8585',
          PSERVIC=X'0100000100000000000000000'
```

```
DFHLU62T MODEENT LOGMODE=DFHLU62T, APPC SINGLE-SESSION
          TYPE=0,
          FMPROF=X'13',
          TSPROF=X'07',
          PRIPROT=X'B0',
          SECPROT=X'B0',
          COMPROT=X'50B1',
          RUSIZES=X'8888',
          PSERVIC=X'06020000000000000000002C00'
```

```
DFH3270  MODEENT LOGMODE=DFH3270, 3270
          TYPE=1,
          FMPROF=X'02',
```

```
DFH3270P  MODEENT LOGMODE=DFH3270P, 3284/3286 BISYNC 3270P (QUERY)
          TYPE=1,
          FMPROF=X'02',
          TSPROF=X'02',
          PRIPROT=X'71',
          SECPRROT=X'40',
          COMPROT=X'2000',
          RUSIZES=X'0000'
```

以下の項目は、CICS 提供の TYPETERM 定義と適合する、z/OS Communications Server 提供の LOGMODE 定義です。

```
DFHLU0M2 MODEENT LOGMODE=D4B32782, LU0 model 2 nonqueryable
      FMPPROF=X'02',
      TSPPROF=X'02',
      PRIPROT=X'71',
      SECPR0T=X'40',
      COMPROT=X'2000',
      RUSIZES=X'0000',
      PSERVIC=X'0000000000000185000007E00'
```

```
DFHLU0M3 MODEENT LOGMODE=D4B32783, LU0 model 3 nonqueryable
      FMPPROF=X'02',
      TSPPROF=X'02',
      PRIPROT=X'71',
      SECPROT=X'40',
      COMPROT=X'2000',
      RUSIZES=X'0000',
      PSERVIC=X'00000000000000185020507F00'
```

```
DFHLU0M4 MODEENT LOGMODE=D4B32784, LU0 model 4 nonqueryable
      FMPPROF=X'02',
      TSPPROF=X'02',
      PRIPROT=X'71',
      SECPROT=X'40',
      COMPROT=X'2000',
      RUSIZES=X'0000',
      PSERVIC=X'0000000000000018502B507F00'
```

付録 A z/OS Communications Server LOGON モード・テーブルのエントリーのコーディング 369

```
DFHLU2E2 MODEENT LOGMODE=SNX32702, LU2 model 2 queryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87F8',
      PSERVIC=X'028000000000185000007E00'
```

```
DFHLU2E3 MODEENT LOGMODE=SNX32703, LU2 model 3 queryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87F8',
      PSERVIC=X'028000000000185020507F00'
```

```
DFHLU2E4 MODEENT LOGMODE=SNX32704, LU2 model 4 queryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87F8',
      PSERVIC=X'02800000000018502B507F00'
```

```
DFHLU2M2 MODEENT LOGMODE=D4A32782, LU2 model 2 nonqueryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87C7',
      PSERVIC=X'020000000000185000007E00'
```

```
DFHLU2M3 MODEENT LOGMODE=D4A32783, LU2 model 3 nonqueryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87C7',
      PSERVIC=X'020000000000185020507F00'
```

```
DFHLU2M4 MODEENT LOGMODE=D4A32784, LU2 model 4 nonqueryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87C7',
      PSERVIC=X'02000000000018502B507F00'
```

```
DFHLU2M5 MODEENT LOGMODE=D4A32785, LU2 model 5 nonqueryable
      FMPROF=X'03',
      TSPROF=X'03',
      PRIPROT=X'B1',
      SECPROT=X'90',
      COMPROT=X'3080',
      RUSIZES=X'87C7',
      PSERVIC=X'02000000000018501B847F00'
```



## 付録 B ノード異常条件プログラムのデフォルト・アクション

ノード異常条件プログラム DFHZNAC のデフォルト・アクションは、z/OS Communications Server から受け取る端末エラー・コードとシステム・センス・コードに応じて異なります。

ほとんどの場合、DFHZNAC はメッセージを発行し、ノード・エラー・プログラム DFHZNEP に渡される通信域に 1 つ以上の「アクション・フラグ」を設定します。その後、DFHZNEP でフラグを設定したり再設定したりして、デフォルトのアクションを変更できます (メッセージは変更できません)。 (しかし、エラー発生時のノードの状態に応じて、場合によってはアクション設定とは異なるアクションが行われる可能性があることに注意してください。)

DFHZNAC と DFHZNEP について詳しくは、[ノード・エラー・プログラムの作成](#)を参照してください。

### DFHZNAC: 端末エラー・コードに関するデフォルトのアクション

z/OS Communications Server からの端末エラー・コードは、DFHZNEP に渡される通信域の 1 バイト・フィールド (TWAEC) に入れます。

371 ページの表 34 は、端末エラー・コードごとに、発行されるメッセージと、DFHZNAC で設定されるアクション・フラグを示しています。

「設定されるアクション・フラグ」列の数値は、ビット設定値に変換されて 386 ページの表 37 で説明されています。

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ			
エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'10'	TCZSRCTU	DFHZNAC2405	18
X'11'	TCZSRCBF	DFHZNAC2403	2 5 7 18 24
X'13'	TCZSRCVH	DFHZNAC2416	7 18 24
X'14'	TCZLRCER	DFHZNAC2404	2 3 7 9 10 11 23 24
X'15'	TCZSRCPF	DFHZNAC2407	2 3 7 9 10 11 24
X'16'	TCZDMIT	DFHZNAC3492	なし
X'18'	TCZLRCNR	DFHZNAC2404	2 3 7 9 10 11 23 24
X'19'	TCZSRCTS	DFHZNAC2406	9 10 11 18
X'1A'	TCZSRCVE	DFHZNAC2408	2 3 7 9 10 11 24
X'1D'	TCZSRCVI	DFHZNAC2417	2 7 24
X'1E'	TCZSRCV2	DFHZNAC2408	2 3 7 9 10 11 24
X'20'	TCZVTAMI	DFHZNAC2417	なし
X'21'	TCZLUCF1	DFHZNAC4902	3 7 9 10 11 24
X'22'	TCZLUCF2	DFHZNAC4903	3 7 9 10 11 24
X'23'	TCZFMSBE	DFHZNAC4904	3 7 9 10 11 24
X'24'	TCZFMSCS	DFHZNAC4905	3 7 9 10 11 24
X'25'	TCZFSMCR	DFHZNAC4906	3 7 9 10 11 24
X'26'	TCZSDLER	DFHZNAC4907	3 7 9 10 11 24

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)

エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'28'	TCZRVLER	DFHZC4909	3 7 9 10 11 24
X'29'	TCZRVLRB	DFHZC4910	3 7 9 10 11 24
X'2A'	TCZRLPEX	DFHZC4911	2 3 7 9 10 11 24
X'2B'	TCZRLPBD	DFHZC4912	2 3 7 9 10 11 24
X'2C'	TCZRLPDR	DFHZC4913	2 3 7 9 10 11 24
X'2D'	TCZRLPIL	DFHZC4914	2 3 7 9 10 11 24
X'2E'	TCZRLPEC	DFHZC4915	2 3 7 9 10 11 24
X'2F'	TCZRLPRR	DFHZC4916	2 3 7 9 10 11 24
X'30'	TCZRLPIF	DFHZC4917	2 3 7 9 10 11 24
X'31'	TCZRLPIR	DFHZC4918	2 3 7 9 10 11 24
X'32'	TCZRLXCL	DFHZC4922	7 20
X'33'	TCZIVIND	DFHZC4919	2 3 7 9 10 11 24
X'34'	TCZIVDAT	DFHZC4920	2 3 7 9 10 11 24
X'35'	TCZRTMT	DFHZC4930	2 3 7 9 10 11 24
X'36'	TCZXSBL	なし	24
X'37'	TCZXSHRA	DFHZC3470	9 10 11 24
X'38'	TCZXSWS	DFHZC6596	2 3 7 15 24
X'39'	TCZXSABN	DFHZC6595	2 3 5 7 24
X'3A'	TCZXSHR	DFHZC6594	7 24
X'3B'	TCZXSBC	DFHZC6593	なし
X'3C'	TCZXUVAR	DFHZC3488	2 3 7 9 10 11 24
X'3D'	TCZXMSG	なし	なし
X'3E'	TCZXERR	DFHZC6591	7 9 10 11 15 24
X'3F'	TCZXRST	DFHZC6590	なし
X'40'	TCZINCPY	DFHZC2489	3 9 11
X'41'	TCZTOLRQ	DFHZC2490	2 3 7 9 10 11 15 24
X'42'	TCZUNPRT	DFHZC2497 - <a href="#">377 ページの『1』</a> を参照	なし
X'43'	TCZCPYNS	DFHZC2434	3 11
X'44'	TCZSRCDE	DFHZC2456	2 3 7 9 10 11 24
X'45'	TCZCHMX	DFHZC3400	3 10 11 22
X'46'	TCZOCIR	DFHZC3402	3 9 10 11
X'47'	TCZGMMS	なし <a href="#">377 ページの『2』</a>	13
X'48'	TCZOPSIN	DFHZC3461	7, 8
X'49'	TCZCLSIN	DFHZC3462	7

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)

エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'4A'	TCZOPACB	DFHZNAC3463	なし
X'4B'	TCZICPUT	DFHZNAC2498	なし
X'4C'	TCZDSPCL	DFHZNAC3481	2 3 7 9 10 11 24
X'4D'	TCZSLRSL	DFHZNAC3473	なし
X'4E'	TCZUNBFE	DFHZNAC3479	2 3 7 9 10 11 24
X'4F'	TCZCNOS0	なし	なし
X'50'	TCZSDRE3	DFHZNAC3417	3 7 9 10 11 24
X'51'	TCZBDPRI	DFHZNAC3418	3 7 9 10 11 24
X'52'	TCZBDUAC	DFHZNAC3419	2 3 5 7
X'53'	TCZBDTOS	DFHZNAC3420	7 20
X'54'	TCZUNBIS	DFHZNAC3434	2 3 7 9 10 11 24
X'55'	TCZEMWBK	DFHZNAC3440	なし
X'56'	TCZXRFVS	DFHZNAC6598	なし
X'57'	TCZRELIS	DFHZNAC3464	7 20
X'58'	TCZERMGR	DFHZNAC3433	7
X'59'	TCZROCT	DFHZNAC2443	2 3 7 9 10 11 24
X'5A'	TCZSBIRV	DFHZNAC3421	7 20
X'5B'	TCZNSP01	DFHZNAC3422	2 3 7 9 10 11 18 24
X'5C'	TCZNSP02	DFHZNAC3424	7 9 10 11 15 24
X'5D'	TCZPRDIO	DFHZNAC0101	なし
X'5E'	TCZBRUAC	DFHZNAC3454	2 3 5 7 18 24
X'5F'	TCZBDSQP	DFHZNAC3455	2 3 5 7 18 24
X'60'	TCZUNCMD	DFHZNAC2421	2 3 7 9 10 11 24
X'62'	TCZVTAMQ	なし 377 ページの『3』	24
X'63'	TCZVTAMO	DFHZNAC3441	なし
X'64'	TCZVTAMA	DFHZNAC3443	なし
X'65'	TCZINVR	DFHZNAC2448	2 3 7 10 11 22 23 24
X'66'	TCZSIGH	DFHZNAC3452	なし
X'67'	TCZVTAMK	DFHZNAC3442	なし
X'69'	TCZSEXOS	DFHZNAC3466	7 20 23
X'6A'	TCZTIOAE	DFHZNAC3444	1 2 3 7 9 10 11 24
X'6B'	TCZNOTNA	DFHZNAC3495	7 24
X'6C'	TCZPSAF	DFHZNAC0155	3 6 7 9 10 11 24
X'6D'	TCZPSAR	DFHZNAC0156	7

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)

エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'70'	TCZCLRRV	DFHZNAC3468	7 9 10 11 15 24
X'71'	TCZPSLE	DFHZNAC0147	3 6 7 9 10 11 24
X'72'	TCZPSVF	DFHZNAC0148	7 9 10 11 24
X'73'	TCZSDSE4	DFHZNAC2437	3 9 11
X'74'	TCZSDSE5	DFHZNAC2423	3 7 9 10 11 24
X'75'	TCZSESE1	DFHZNAC2424	3 7 9 10 11 15 24
X'76'	TCZLGNA	DFHZNAC2487	3
X'77'	TCZDMRY	DFHZNAC2488	なし
X'78'	TCZSDRE2	DFHZNAC2430	3 9 11 22
X'79'	TCZPSRAF	DFHZNAC0145	3 6 7 9 10 11 24
X'7A'	TCZPSRAC	DFHZNAC0144	7 11
X'7C'	TCZPSANR	DFHZNAC0157	3 7 9 10 11 24
X'7D'	TCZRABUS	DFHZNAC4949	2 3 7 9 10 11 24
X'80'	TCZSRCSP	DFHZNAC2414	なし
X'81'	TCZSSXNR	DFHZNAC2432	なし
X'82'	TCZSSXUC	DFHZNAC2419	2 3 7 9 10 11 23 24
X'83'	TCZSSXAR	DFHZNAC2450	なし
X'84'	TCZSSXIB	DFHZNAC2446	2 3 7 9 10 11 23 24
X'85'	TCZUNEGR	DFHZNAC3409	2 3 7 9 10 11 23 24
X'88'	TCZLEXCI	DFHZNAC2467	2 3 7 9 10 11 23 24
X'89'	TCZLEXUS	DFHZNAC2468	2 3 7 9 10 11 24
X'8A'	TCZLUSRR	DFHZNAC4937	2 3 5 7 24
X'8B'	TCZLUSRF	DFHZNAC4938	2 3 5 7 24
X'8C'	TCZLUPUN	DFHZNAC4939	2 3 5 7 24
X'8D'	TCZLUPLK	DFHZNAC4941	2 3 5 7 24
X'8E'	TCZLUPEX	DFHZNAC4942	2 3 5 7 24
X'8F'	TCZLUSKN	DFHZNAC4940	2 3 5 7 24
X'90'	TCZLGCER	DFHZNAC2422	1 2 3 6 9 10 11 23 24
X'91'	TCZRSTLE	DFHZNAC2429	3 10 11
X'92'	TCZSDSE6	DFHZNAC2428	3 9 11
X'93'	TCZRACET	DFHZNAC2455	2 3 9 10 11
X'94'	TCZRACES	DFHZNAC2426	2 3 9 10 11 22
X'95'	TCZSDSE8	DFHZNAC2445	3 9 11
X'96'	TCZRVSZ1	DFHZNAC2435	3 7 10 11 24

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)

エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'97'	TCZRVSZ3	DFHZNAC2436	3 10 11
X'98'	TCZACT01	DFHZNAC2439	2 18
X'99'	TCZSDSE7	DFHZNAC2459	3 9 11
X'9A'	TCZDOMCF	DFHZNAC2447	3 9 10 11 23
X'9B'	TCZRACNL	DFHZNAC2486	3
X'9D'	TCZRSPER	DFHZNAC3465	1 2 3 7 9 10 11 23
X'9E'	TCZDEVND	DFHZNAC3472	なし
X'A0'	TCZNOISC	DFHZNAC3480	7 23 24
X'A1'	TCZRVSZ2	DFHZNAC2438	3 10 11
X'A2'	TCZPRGE	DFHZNAC4945	3 7 9 10 11 24
X'A3'	TCZBKTSSE	DFHZNAC2444	2 3 7 9 10 11 24
X'A7'	TCZBOEB	DFHZNAC2449	2 3 7 11 18 22 24
X'A8'	TCZFMHLE	DFHZNAC2471	2 3 4 7 10 11 22 24
X'A9'	TCZRACRF	DFHZNAC2472	11
X'AA'	TCZSDSE9	DFHZNAC2473	3 9 11
X'AB'	TCZLUERR	DFHZNAC3470	7 9 10 11 24
X'AC'	TCZVRDAC	DFHZNAC3474	7 9 10 11 24
X'AD'	TCZNRLUF	DFHZNAC3475	7 9 10 11 24
X'AE'	TCZRCLUF	DFHZNAC3476	7 9 10 11 24
X'AF'	TCZCLEAN	DFHZNAC3477	7 9 10 11 24
X'B0'	TCZEXRO	DFHZNAC3491	7 15 24
X'B1'	TCZRPLAC	DFHZNAC2401	2 3 7 9 10 11 23 24
X'B2'	TCZSDAUC	DFHZNAC2425	3 7 9 10 11 15 24
X'B3'	TCZBDBND	DFHZNAC4929	2 3 5 7 24
X'B4'	TCZRSNE	DFHZNAC2402	3 11
X'B5'	TCZSAXUC	DFHZNAC2420	2 3 7 9 10 11 23 24
X'B6'	TCZNSEED	DFHZNAC4924	2 3 5 7 24
X'B7'	TCZASINC	DFHZNAC4925	2 3 5 7 24
X'B8'	TCZEVBAD	DFHZNAC4926	2 3 5 7 24
X'B9'	TCZFMH12	DFHZNAC4927	2 3 5 7 24
X'BB'	TCZSEXUC	DFHZNAC2418	2 3 7 9 10 11 23 24
X'BC'	TCZINIIR	DFHZNAC3410	2 3 9 10 11
X'BD'	TCZDESGM	DFHZNAC4928	7 24
X'BE'	TCZBFAIL	DFHZNAC4944	2 3 5 24

表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)

エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'BF'	TCZCPFAL	DFHZNAC3490	7 24
X'CO'	TCZDWEGF	DFHZNAC3499	なし
X'C1'	TCZSRCAT	DFHZNAC2400	2 3 7 9 10 11 23 24
X'C2'	TCZLUINP	DFHZNAC3486	7 24
X'C3'	TCZCPFAL	DFHZNAC3490	24
X'C5'	TCZSRCNA	DFHZNAC2427	2
X'C6'	TCZPASSD	DFHZNAC3484	なし
X'C7'	TCZPSPRE	DFHZNAC3485	7 24
X'C8'	TCZLUINH	DFHZNAC3489	7 18 24
X'C9'	TCZNPSAU	DFHZNAC3487	7 24
X'CB'	TCZSRCTC	DFHZNAC2431	2 3 9 10 11
X'CC'	TCZSRCCI	DFHZNAC2451	2 3 9 10 11
X'CD'	TCZSRCCX	DFHZNAC2454	2 3 9 10 11
X'CE'	TCZVHOLD	DFHZNAC3469	7 9 10 11 24
X'CF'	TCZVRNOP	DFHZNAC3471	7 9 10 11 24
X'D0'	TCZTXCS	DFHZNAC2409	2 3 7 9 10 11 15 24
X'D1'	TCZTXCU	DFHZNAC2410	2 3 7 9 10 11 24
X'D3'	TCZDMPD	DFHZNAC2463	なし
X'D4'	TCZCXRR	DFHZNAC2453	1 2 3 9 10
X'D5'	TCZCXE2	DFHZNAC2452	3 7 9 10 11 18 24
X'D6'	TCZSXC2	DFHZNAC2441	なし
X'D7'	TCZSXC1	DFHZNAC2440	なし
X'D8'	TCZRNCH	DFHZNAC2457	2 3 7 9 10 11 24
X'D9'	TCZYX43	DFHZNAC2469	2 3 9 10 11
X'DA'	TCZSXC3	DFHZNAC2470	7 9 10 11 24
X'DB'	TCZPIPL	DFHZNAC2117	7 9 10 11 23 24
X'DC'	TCZPXE1	DFHZNAC2442	なし
X'DD'	TCZPXE2	DFHZNAC2458	なし
X'DE'	TCZPIPP	DFHZNAC2119	7 9 10 11 23 24
X'DF'	TCZDMGF	DFHZNAC3482	なし
X'E0'	TCZDMSN	DFHZNAC2411	なし
X'E1'	TCZDMRA	DFHZNAC2412	なし
X'E2'	TCZDMCL	DFHZNAC2413	2
X'E3'	TCZCNCL	DFHZNAC2485	3 9 10 11



表 34. 特定のエラー・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)			
エラー・コード	シンボル・ラベル	メッセージ	アクション・フラグ設定
X'E4'	TCZAIER	DFHZC2433	なし
X'E6'	TCZDMLG	DFHZC2404	なし
X'E8'	TCZDMSLE	DFHZC3416	2 3
X'E9'	TCZSTIND	DFHZC2102	3
X'EA'	TCZSTLER	DFHZC3432	2 3
X'EB'	TCZSTRMH	DFHZC3428	3
X'EC'	TCZSTRMM	DFHZC3429	2 3 7
X'ED'	TCZRTHS	DFHZC3403	2 3 7 9 10 11 23 24
X'EF'	TCZSTIN	DFHZC3431	2 3
X'F1'	TCZBDMOD	DFHZC4931	7 18 24
X'F2'	TCZEXRVT	DFHZC2469	2 3 9 10 11
X'F3'	TCZICTYP	DFHZC4932	2 3 7 24
X'F4'	TCZIDBA	DFHZC4933	2 3 24
X'F5'	TCZISYNL	DFHZC4934	2 3 7 24
X'F6'	TCZIUOW	DFHZC4935	2 3 7 24
X'F7'	TCZIFMHL	DFHZC4936	2 3 7 24
X'F8'	TCZFMRB	DFHZC4943	3 7 9 10 11 24
X'F9'	TCZINVAT	DFHZC4946	2 3 7 24
X'FA'	TCZLUSEC	DFHZC4947	2 3 7 24
X'FB'	TCZPSUNB	DFHZC0125	7
X'FC'	TCZPSOPN	DFHZC0131	7
X'FD'	TCZPSRC	DFHZC0146	7
X'FE'	TCZPSRF	DFHZC0150	3 6 7 9 10 11 15 24
X'FF'	TCZPSPE	DFHZC0149	7

注:

1. デバイス・タイプに応じて、メッセージ DFHZC2497 または DFHZC3493 を参照してください。
2. 「Good morning」メッセージが送信されます。
3. クイック・クローズまたは異常終了のために、タスクを取り消して、z/OS Communications Server セッションを閉じます。

## z/OS Communications Server エラーに関連した CICS メッセージ

表 35. z/OS Communications Server エラーに関連した CICS メッセージ			
メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC0101	TCZPRDTO	X'5D'	なし
DFHZC0125	TCZPSUNB	X'FB'	7

表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC0131	TCZPSOPN	X'FC'	7
DFHZC0144	TCZPSRAC	X'7A'	7 11
DFHZC0145	TCZPSRAF	X'79'	3 6 7 9 10 11 24
DFHZC0146	TCZPSRC	X'FD'	7
DFHZC0147	TCZPSLE	X'71'	3 6 7 9 10 11 24
DFHZC0148	TCZPSVF	X'72'	7 9 10 11 24
DFHZC0149	TCZPSPE	X'FF'	7
DFHZC0150	TCZPSRF	X'FE'	3 6 7 9 10 11 15 24
DFHZC0155	TCZPSAF	X'6C'	3 6 7 9 10 11 24
DFHZC0156	TCZPSAR	X'6D'	7
DFHZC0157	TCZPSANR	X'7C'	3 7 9 10 11 24
DFHZC2102	TCZSTIND	X'E9'	3
DFHZC2117	TCZPIPL	X'DB'	7 9 10 11 23 24
DFHZC2119	TCZPIPP	X'DE'	7 9 10 11 23 24
DFHZC2400	TCZSRCAT	X'C1'	2 3 7 9 10 11 23 24
DFHZC2401	TCZRPLAC	X'B1'	2 3 7 9 10 11 23 24
DFHZC2402	TCZRSNE	X'B4'	3 11
DFHZC2403	TCZSRCBF	X'11'	2 5 7 18 24
DFHZC2404	TCZLRCER	X'14'	2 3 7 9 10 11 23 24
DFHZC2404	TCZLRCNR	X'18'	2 3 7 9 10 11 23 24
DFHZC2404	TCZDMLG	X'E6'	なし
DFHZC2405	TCZSRCTU	X'10'	18
DFHZC2406	TCZSRCTS	X'19'	9 10 11 18
DFHZC2407	TCZSRCPF	X'15'	2 3 7 9 10 11 24
DFHZC2408	TCZSRCVE	X'1A'	2 3 7 9 10 11 24
DFHZC2408	TCZSRCV2	X'1E'	2 3 7 9 10 11 24
DFHZC2409	TCZTXCS	X'D0'	2 3 7 9 10 11 15 24
DFHZC2410	TCZTXCU	X'D1'	2 3 7 9 10 11 24
DFHZC2411	TCZDMSN	X'E0'	なし
DFHZC2412	TCZDMRA	X'E1'	なし
DFHZC2413	TCZDMCL	X'E2'	2
DFHZC2414	TCZSRCSP	X'80'	なし
DFHZC2416	TCZSRCVH	X'13'	7 18 24
DFHZC2417	TCZSRCVI	X'1D'	2 7 24

表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC2417	TCZVTAMI	X'20'	なし
DFHZC2418	TCZSEXUC	X'BB'	2 3 7 9 10 11 23 24
DFHZC2419	TCZSSXUC	X'82'	2 3 7 9 10 11 23 24
DFHZC2420	TCZSAXUC	X'B5'	2 3 7 9 10 11 23 24
DFHZC2421	TCZUNCMD	X'60'	2 3 7 9 10 11 24
DFHZC2422	TCZLGCER	X'90'	1 2 3 6 9 10 11 23 24
DFHZC2423	TCZSDSE5	X'74'	3 7 9 10 11 24
DFHZC2424	TCZSESE1	X'75'	3 7 9 10 11 15 24
DFHZC2425	TCZSDAUC	X'B2'	3 7 9 10 11 15 24
DFHZC2426	TCZRACES	X'94'	2 3 9 10 11 22
DFHZC2427	TCZSRCNA	X'C5'	2
DFHZC2428	TCZSDSE6	X'92'	3 9 11
DFHZC2429	TCZRSTLE	X'91'	3 10 11
DFHZC2430	TCZSDRE2	X'78'	3 9 11 22
DFHZC2431	TCZSRCTC	X'CB'	2 3 9 10 11
DFHZC2432	TCZSSXNR	X'81'	なし
DFHZC2433	TCZAIER	X'E4'	なし
DFHZC2434	TCZCPYNS	X'43'	3 11
DFHZC2435	TCZRVSZ1	X'96'	3 7 10 11 24
DFHZC2436	TCZRVSZ3	X'97'	3 10 11
DFHZC2437	TCZSDSE4	X'73'	3 9 11
DFHZC2438	TCZRVSZ2	X'A1'	3 10 11
DFHZC2439	TCZACT01	X'98'	2 18
DFHZC2440	TCZSXC1	X'D7'	なし
DFHZC2441	TCZSXC2	X'D6'	なし
DFHZC2442	TCZPXE1	X'DC'	なし
DFHZC2443	TCZROCT	X'59'	2 3 7 9 10 11 24
DFHZC2444	TCZBKTSE	X'A3'	2 3 7 9 10 11 24
DFHZC2445	TCZSDSE8	X'95'	3 9 11
DFHZC2446	TCZSSXIB	X'84'	2 3 7 9 10 11 23 24
DFHZC2447	TCZDOMCF	X'9A'	3 9 10 11 23
DFHZC2448	TCZINVR	X'65'	2 3 7 10 11 22 23 24
DFHZC2449	TCZBOEB	X'A7'	2 3 7 11 18 22 24
DFHZC2450	TCZSSXAR	X'83'	なし

表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC2451	TCZSRCCI	X'CC'	2 3 9 10 11
DFHZC2452	TCZCXE2	X'D5'	3 7 9 10 11 18 24
DFHZC2453	TCZCXRR	X'D4'	1 2 3 9 10
DFHZC2454	TCZSRCCX	X'CD'	2 3 9 10 11
DFHZC2455	TCZRACET	X'93'	2 3 9 10 11
DFHZC2456	TCZSRCDE	X'44'	2 3 7 9 10 11 24
DFHZC2457	TCZRNCH	X'D8'	2 3 7 9 10 11 24
DFHZC2458	TCZPX2	X'DD'	なし
DFHZC2459	TCZSDSE7	X'99'	3 9 11
DFHZC2463	TCZDMPD	X'D3'	なし
DFHZC2467	TCZLEXCI	X'88'	2 3 7 9 10 11 23 24
DFHZC2468	TCZLEXUS	X'89'	2 3 7 9 10 11 24
DFHZC2469	TCZYX43	X'D9'	2 3 9 10 11
DFHZC2469	TCZEXRVT	X'F2'	2 3 9 10 11
DFHZC2470	TCZSXC3	X'DA'	7 9 10 11 24
DFHZC2471	TCZFMHLE	X'A8'	2 3 4 7 10 11 22 24
DFHZC2472	TCZRACRF	X'A9'	11
DFHZC2473	TCZSDSE9	X'AA'	3 9 11
DFHZC2485	TCZCNCL	X'E3'	3 9 10 11
DFHZC2486	TCZRACNL	X'9B'	3
DFHZC2487	TCZLGNA	X'76'	3
DFHZC2488	TCZDMRY	X'77'	なし
DFHZC2489	TCZINCPY	X'40'	3 9 11
DFHZC2490	TCZTOLRQ	X'41'	2 3 7 9 10 11 15 24
DFHZC2497	TCZUNPRT	X'42'	なし
DFHZC2498	TCZICPUT	X'4B'	なし
DFHZC3400	TCZCHMX	X'45'	3 10 11 22
DFHZC3402	TCZOCIR	X'46'	3 9 10 11
DFHZC3403	TCZRTHS	X'ED'	2 3 7 9 10 11 23 24
DFHZC3409	TCZUNEGR	X'85'	2 3 7 9 10 11 23 24
DFHZC3410	TCZINIIR	X'BC'	2 3 9 10 11
DFHZC3416	TCZDMSLE	X'E8'	2 3
DFHZC3417	TCZSDRE3	X'50'	3 7 9 10 11 24
DFHZC3418	TCZBDPRI	X'51'	3 7 9 10 11 24

表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC3419	TCZBDUAC	X'52'	2 3 5 7
DFHZC3420	TCZBDTOS	X'53'	7 20
DFHZC3421	TCZSBIRV	X'5A'	7 20
DFHZC3422	TCZNSP01	X'5B'	2 3 7 9 10 11 18 24
DFHZC3424	TCZNSP02	X'5C'	7 9 10 11 15 24
DFHZC3428	TCZSTRMH	X'EB'	3
DFHZC3429	TCZSTRMM	X'EC'	2 3 7
DFHZC3431	TCZSTIN	X'EF'	2 3
DFHZC3432	TCZSTLER	X'EA'	2 3
DFHZC3433	TCZERMGR	X'58'	7
DFHZC3434	TCZUNBIS	X'54'	2 3 7 9 10 11 24
DFHZC3440	TCZEMWBK	X'55'	なし
DFHZC3441	TCZVTAMO	X'63'	なし
DFHZC3442	TCZVTAMK	X'67'	なし
DFHZC3443	TCZVTAMA	X'64'	なし
DFHZC3444	TCZTIOAE	X'6A'	1 2 3 7 9 10 11 24
DFHZC3452	TCZSigr	X'66'	なし
DFHZC3454	TCZBRUAC	X'5E'	2 3 5 7 18 24
DFHZC3455	TCZBDSQP	X'5F'	2 3 5 7 18 24
DFHZC3461	TCZOPSIN	X'48'	7, 8
DFHZC3462	TCZCLSIN	X'49'	7
DFHZC3463	TCZOPACB	X'4A'	なし
DFHZC3464	TCZRELIS	X'57'	7 20
DFHZC3465	TCZRSPER	X'9D'	1 2 3 7 9 10 11 23
DFHZC3466	TCZSEXOS	X'69'	7 20 23
DFHZC3468	TCZCLRRV	X'70'	7 9 10 11 15 24
DFHZC3469	TCZVHOLD	X'CE'	7 9 10 11 24
DFHZC3470	TCZXSHRA	X'37'	9 10 11 24
DFHZC3470	TCZLUERR	X'AB'	7 9 10 11 24
DFHZC3471	TCZVRNOP	X'CF'	7 9 10 11 24
DFHZC3472	TCZDEVND	X'9E'	なし
DFHZC3473	TCZSLSRL	X'4D'	なし
DFHZC3474	TCZVRDAC	X'AC'	7 9 10 11 24
DFHZC3475	TCZNRLUF	X'AD'	7 9 10 11 24

表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC3476	TCZRCLUF	X'AE'	7 9 10 11 24
DFHZC3477	TCZCLEAN	X'AF'	7 9 10 11 24
DFHZC3479	TCZUNBFE	X'4E'	2 3 7 9 10 11 24
DFHZC3480	TCZNOISC	X'AO'	7 23 24
DFHZC3481	TCZDSPCL	X'4C'	2 3 7 9 10 11 24
DFHZC3482	TCZDMGF	X'DF'	なし
DFHZC3484	TCZPASSD	X'C6'	なし
DFHZC3485	TCZPSPRE	X'C7'	7 24
DFHZC3486	TCZLUINP	X'C2'	7 24
DFHZC3487	TCZNPSAU	X'C9'	7 24
DFHZC3488	TCZXUVAR	X'3C'	2 3 7 9 10 11 24
DFHZC3489	TCZLUINH	X'C8'	7 18 24
DFHZC3490	TCZCPFAL	X'C3'	7 24
DFHZC3491	TCZEXRO	X'B0'	7 15 24
DFHZC3492	TCZDMIT	X'16'	なし
DFHZC3495	TCZNOTNA	X'6B'	7 24
DFHZC3499	TCZDWEGF	X'C0'	なし
DFHZC4902	TCZLUCF1	X'21'	3 7 9 10 11 24
DFHZC4903	TCZLUCF2	X'22'	3 7 9 10 11 24
DFHZC4904	TCZFMSBE	X'23'	3 7 10 11 9 24
DFHZC4905	TCZFSMCS	X'24'	3 7 10 11 9 24
DFHZC4906	TCZFSMCR	X'25'	3 7 10 11 9 24
DFHZC4907	TCZSDLER	X'26'	3 7 10 11 9 24
DFHZC4909	TCZRVLER	X'28'	3 7 10 11 9 24
DFHZC4910	TCZRVLRB	X'29'	3 7 10 11 9 24
DFHZC4911	TCZRLPEX	X'2A'	2 3 7 9 10 11 24
DFHZC4912	TCZRLPBD	X'2B'	2 3 7 9 10 11 24
DFHZC4913	TCZRLPDR	X'2C'	2 3 7 9 10 11 24
DFHZC4914	TCZRLPIL	X'2D'	2 3 7 9 10 11 24
DFHZC4915	TCZRLPEC	X'2E'	2 3 7 9 10 11 24
DFHZC4916	TCZRLPRR	X'2F'	2 3 7 9 10 11 24
DFHZC4917	TCZRLPIF	X'30'	2 3 7 9 10 11 24
DFHZC4918	TCZRLPIR	X'31'	2 3 7 9 10 11 24
DFHZC4919	TCZIVIND	X'33'	2 3 7 9 10 11 24



表 35. z/OS Communications Server エラーに関連した CICS メッセージ (続き)

メッセージ	シンボル・ラベル	エラー・コード	アクション・フラグ設定
DFHZC4920	TCZIVDAT	X'34'	2 3 7 9 10 11 24
DFHZC4922	TCZRLXCL	X'32'	7 20
DFHZC4924	TCZNSEED	X'B6'	2 3 5 7 24
DFHZC4925	TCZASINC	X'B7'	2 3 5 7 24
DFHZC4926	TCZEVBAD	X'B8'	2 3 5 7 24
DFHZC4927	TCZFMH12	X'B9'	2 3 5 7 24
DFHZC4928	TCZDESGM	X'BD'	7 24
DFHZC4929	TCZBDBND	X'B3'	2 3 5 7 24
DFHZC4930	TCZRTMT	X'35'	2 3 7 9 10 11 24
DFHZC4931	TCZBDMOD	X'F1'	7 18 24
DFHZC4932	TCZICTYP	X'F3'	2 3 7 24
DFHZC4933	TCZIDBA	X'F4'	2 3 24
DFHZC4934	TCZISYNL	X'F5'	2 3 7 24
DFHZC4935	TCZIUOW	X'F6'	2 3 7 24
DFHZC4936	TCZIFMHL	X'F7'	2 3 7 24
DFHZC4937	TCZLUSRR	X'8A'	2 3 5 7 24
DFHZC4938	TCZLUSRF	X'8B'	2 3 5 7 24
DFHZC4939	TCZLUPUN	X'8C'	2 3 5 7 24
DFHZC4940	TCZLUSKN	X'8F'	2 3 5 7 24
DFHZC4941	TCZLUPLK	X'8D'	2 3 5 7 24
DFHZC4942	TCZLUPEX	X'8E'	2 3 5 7 24
DFHZC4943	TCZFMSMRB	X'F8'	3 7 9 10 11 24
DFHZC4944	TCZBFAIL	X'BE'	2 3 5 7 24
DFHZC4945	TCZPRGE	X'A2'	3 7 9 10 11 24
DFHZC4946	TCZINVAT	X'F9'	2 3 7 24
DFHZC4947	TCZLUSEC	X'FA'	2 3 7 24
DFHZC4949	TCZRABUS	X'7D'	2 3 7 9 10 11 24
DFHZC6590	TCZXRST	X'3F'	なし
DFHZC6591	TCZXERR	X'3E'	7 9 10 11 15 24
DFHZC6593	TCZXSBC	X'3B'	なし
DFHZC6594	TCZXSHR	X'3A'	7 24
DFHZC6595	TCZXSABN	X'39'	2 3 5 7 24
DFHZC6596	TCZXSWAS	X'38'	2 3 7 15 24
DFHZC6598	TCZXRFVS	X'56'	なし

## DFHZNAC: システム・センス・コードに関するデフォルトのアクション

384 ページの表 36 は、受け取るインバウンド・システム・センス・コードごとに、発行されるメッセージと、DFHZNAC で設定されるアクション・フラグを示しています。「設定されるアクション・フラグ」列の数値は、ビット設定値に変換されて 386 ページの表 37 で説明されています。

表 36. 特定のセンス・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ		
センス・コード	メッセージ	アクション・フラグ設定
X'0001' <sup>1</sup>	DFHZC3401	2
X'0002' <sup>1</sup>	DFHZC3415	2, 3, 10, 11
X'0003' <sup>1</sup>	DFHZC3449	なし
X'0004' <sup>1</sup>	DFHZC3450	なし
X'0007' <sup>1</sup>	DFHZC3451	なし <sup>2</sup>
X'00FF'	DFHZC3446	2, 3, 7, 9, 10, 11, 23, 24
X'0801'	DFHZC2476	3, 9, 10, 11
X'0802'	DFHZC2461	なし
X'0806'	DFHZC3426	なし
X'0807'	DFHZC3411	なし
X'080B'	DFHZC2462	2, 3, 7, 9, 10, 11, 15, 24
X'080E'	DFHZC3448	23
X'080F'	DFHZC3436	9, 10, 11
X'0811'	DFHZC2464	9, 10, 11
X'0812'	DFHZC2465	2, 3
X'081B'	DFHZC2483	2, 3 <sup>3</sup>
X'081C'	DFHZC2466	2, 3, 9, 10, 11
X'0824'	DFHZC2475	3, 9, 10, 11
X'0825'	DFHZC2484	2, 3, 9, 10, 11
X'0826'	DFHZC3423	2, 3, 9, 10, 11
X'0827'	DFHZC2480	3
X'0829'	DFHZC3407	1, 2, 3, 7, 10, 11, 24
X'082A'	なし <sup>4</sup>	9
X'082B'	DFHZC3408	2, 3, 10, 11, 13
X'082D'	DFHZC3413	なし
X'082E'	DFHZC3412	なし
X'082F'	DFHZC3414	2, 3, 9, 10, 11
X'0831'	DFHZC3438	なし
X'0833'	DFHZC3427	なし
X'0847'	DFHZC3439	なし
X'084A'	なし <sup>5</sup>	なし

表 36. 特定のセンス・コードに関する、発行されるメッセージと DFHZNAC で設定されるフラグ (続き)		
センス・コード	メッセージ	アクション・フラグ設定
X'084C'	DFHZC3467	9, 10, 11
X'0860'	DFHZC3459	なし
X'0863'	DFHZC3460	9, 10, 11
X'0864'	DFHZC2475	3, 9, 10, 11
X'0865'	DFHZC2465	3, 9, 10, 11
X'0866'	DFHZC2475	3, 9, 10, 11
X'0867'	なし <sup>6</sup>	9, 10, 11
X'0868'	DFHZC3456	2, 9, 10, 11
X'0869'	DFHZC3457	2, 9, 10, 11
X'08FF'	DFHZC3447	2, 3, 7, 9, 10, 11, 24
X'1000'	DFHZC3494	2, 3, 9, 10, 11
X'1001'	DFHZC2481	2, 3, 9, 10, 11, 14
X'1002'	DFHZC2481	2, 3, 9, 10, 11, 14
X'1003'	DFHZC2479	2, 3, 9, 10, 11, 14
X'1005'	DFHZC3406	2, 3, 4, 9, 10, 11, 14
X'1008'	DFHZC2478	なし
X'1009'	DFHZC3458	2, 9, 10, 11
X'10FF'	DFHZC3446	2, 3, 7, 9, 10, 11, 23, 24
X'2003'	DFHZC3405	2, 3, 7, 9, 10, 11, 15, 24
X'20FF'	DFHZC3445	2, 3, 7, 9, 10, 11, 23, 24
X'400B'	DFHZC2477	1, 3, 11
X'40FF'	DFHZC3453	2, 3, 7, 9, 10, 11, 23, 24
X'8000'	DFHZC3435	2, 3, 7, 9, 10, 11, 18, 24
X'8005'	DFHZC3435	2, 3, 7, 9, 10, 11, 18, 24
X'80FF'	DFHZC3435	2, 3, 7, 9, 10, 11, 18, 23, 24
X'FFFF'	DFHZC2460	2, 3, 7, 9, 10, 11, 23, 24

注:

1. システム・センス・コードは、LUSTATUS コマンド・コードの形式になります。
2. タスクが追加される場合や、未解決の操作が完了する場合には、アクション・フラグは設定されません。それ以外の場合には、フラグ 21 が設定されます。
3. アクション・フラグ 2 と 3 は、SEND で確定応答を要求して否定応答を受け取る場合に設定されます。
4. 表示スペース・エラー。
5. 読み取り時の表示エラー。オペレーター介入によるディスプレイ・バッファの変更が、互換モードの論理装置に対する READ コマンド上で検出されました。
6. デバイスから機能の異常終了を受け取りました。チェーンに対する否定応答が送信されましたが、パージされました。

## アクション・フラグの設定値および意味

386 ページの表 37 では、DFHZNEP に渡される通信域で DFHZNAC によって設定できる「アクション・フラグ」について説明しています。DFHZNAC で設定されたフラグは、DFHZNEP で設定を変更していない場合に実行されるデフォルトのアクションを表しています。

「フラグ」列の図は、371 ページの表 34 および 384 ページの表 36 の列 3 にあるフラグを参照しています。

表 37. DFHZNAC によって設定されるアクション・フラグの意味				
フラグ	フィールド	ビット・マスク	16 進ビット設定値	アクション
1	TWAOPT1	1... ..	X'80'	アクション・フラグを印刷する
2		.1.. ..	X'40'	z/OS Communications Server RPL を印刷する
3		..1. ....	X'20'	TCTTE を印刷する
4		...1. ...	X'10'	TIOA を印刷する
5		.... 1...	X'08'	BIND 域を印刷する
6		.... .1..	X'04'	システム・ダンプ (付加されたタスクが存在しない場合)
7		.... ..1.	X'02'	ネットワーク修飾名 (NQNAME) を印刷する
8		.... ...1	X'01'	TN3270 IP アドレス (TNADDR) を印刷する
9	TWAOPT2	1... ..	X'80'	この端末に関するすべての送信を取り消す
10		.1.. ..	X'40'	この端末に関するすべての受信を取り消す
11		..1. ....	X'20'	TCTTE に接続されているすべてのタスクを異常終了させます。
12		...1. ...	X'10'	TCTTE に接続されているすべてのタスクを取り消します。
13		.... 1...	X'08'	Good Morning メッセージを送信する
14		.... .1..	X'04'	この TCTTE の BMS ページをすべてページする
15		.... ..1.	X'02'	SIMLOGON が必要
17	TWAOPT3	1... ..	X'80'	現在許可されている INTLOG を設定する
18		.1.. ..	X'40'	内部汎用ログオンを設定しない
20		...1. ...	X'10'	通常の CLSDST です (リセット不能)。
21		.... 1...	X'08'	通常の CLSDST です (リセット可能)。
22		.... .1..	X'04'	否定応答を送信する

表 37. DFHZNAC によって設定されるアクション・フラグの意味 (続き)				
フラグ	フィールド	ビット・マスク	16進ビット設定値	アクション
23		.... ..1.	X'02'	AOS - ノードをサービス休止のままにする
24		.... ...1	X'01'	CLSDST ノード





## 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涉外

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](http://ibm.com)<sup>®</sup> 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<sup>®</sup> 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を提供され、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]

アクション・フラグ名、DFHTEP [89](#)  
アクセス・レジスター [2](#)  
アダプター、タスク関連ユーザー出口 [12](#)  
アドレッシング・モードの影響 [33](#)  
アナライザー・プログラム  
    エスケープ・データおよびアンエスケープ・データ [258](#)  
    機能 [253](#)  
    共用、コンバーター・プログラムとのデータの [257](#)  
    作成 [253](#)  
    出力 [255](#)  
    入力 [254](#)  
    CICS 提供の  
        DFHWBAAX [258](#)  
        DFHWBADX [259](#)  
    URIMAP 定義による置き換え [252](#)  
    URIMAP リソース定義との関連 [250](#), [254](#)  
アンエスケープ  
    アナライザー・プログラムでの [258](#)  
異常終了  
    トランザクション・ビット [102](#)  
異常条件  
    サンプル端末エラー・プログラム [83](#)  
    サンプル・ノード・エラー・プログラム [121](#)  
    端末エラー・プログラム [100](#)  
    ユーザー作成のノード・エラー・プログラム [129](#)  
エスケープ  
    アナライザー・プログラムでの [258](#)  
エラー・グループ [110](#)  
エラー・グループ・インデックス [122](#), [128](#)  
エラー状況エレメント (ESE) [90](#)  
エラー状況エレメント (ESE) (error status element (ESE))  
    DFHTEPT TYPE=PERMCODE|ERRCODE [96](#)  
エラー状況ブロック (ESB) (error status block (ESB)) [128](#)  
エラー処理  
    アナライザー・プログラムの役割 [250](#)  
    端末エラー・プログラム (TEP) で [82](#)  
    ノード・エラー・プログラム (NEP) 内 [121](#)  
エンコード機能、コンバーター・プログラムの  
    出力パラメーター [265](#)  
    入力パラメーター [264](#)

## [カ行]

書き込み打ち切りビット [102](#)  
仮想端末、自動インストール [180](#)  
関連データ  
    発信元 データ [48](#)  
共通サブルーチン・ベクトル・テーブル (CSV) [122](#), [129](#)  
グローバル・ユーザー出口  
    概要 [1](#)  
    サンプル・プログラム  
        API 呼び出しと XPI 呼び出しの混合 [3](#)

グローバル・ユーザー出口 (続き)  
    ストレージ保護との併用  
        実行キー [8](#)  
        データ記憶キー [9](#)  
    出口点  
        統計ドメイン内 [275](#)  
    出口プログラム  
        CICS サービスの使用 [2](#)  
        EDF の使用 [4](#)  
        アドレス指定に関する考慮事項 [2](#)  
        エラー [9](#)  
        グローバル作業域 [4](#)  
        定義、有効化、無効化 [10](#)  
        複数の出口で [1](#) つ [11](#)  
        プログラミング・インターフェースの制約事項 [8](#)  
        レジスター規則 [1](#)  
        渡されるパラメーター [5](#)  
        1 出口で複数 [11](#)  
        CICS に値を戻す [8](#)  
    トレース・テーブルのエントリー [4](#)  
    XSNOFF サインオフ出口 [349](#)  
    XSNON サインオン出口 [349](#)  
コード・ページ変換  
    アナライザー・プログラムによる指定 [255](#)  
    アナライザー・プログラムを使用する Web 非対応のアプリケーションの場合 [250](#)  
    DFHWBADX における、サンプル・アナライザー・プログラム [259](#)  
コンソール、自動インストール [156](#)  
コンバーター・プログラム  
    エンコード機能  
        出力パラメーター [265](#)  
        入力パラメーター [264](#)  
    共用、アナライザー・プログラムとのデータの [257](#)  
    構成、応答の [261](#)  
    作成 [261](#)  
    デコード機能  
        出力パラメーター [264](#)  
        入力パラメーター [263](#)  
    呼び出し、複数のアプリケーション・プログラムの [265](#)  
    API コマンド [261](#)

## [サ行]

再帰的再試行ルーチン、DFHTEP  
    例 [106](#)  
サンプル DFHTEP の生成のためのジョブ制御 [90](#)  
サンプル・プログラム  
    「good night」トランザクション (DFH0GNIT) [149](#)  
    システム定義ユーティリティ・プログラム  
        DFHCSDUP 用  
        CSD 相互参照プログラム [324](#)  
        CSD バックアップ・ユーティリティ・プログラム [327](#)  
        Db2 フォーマット設定プログラム [324](#), [325](#)  
        DFH\$CRFA [323](#)  
        DFH\$CRFP [323](#)

サンプル・プログラム (続き)	システム定義ユーティリティ・プログラム (DFHCSDUP) (続き)
システム定義ユーティリティ・プログラム DFHCSDUP 用 (続き)	ユーザー・プログラムからの呼び出し (続き)
DFH\$FORA 323	DFHCSDUP の入力パラメーター 330
DFH\$FORP 323	DFHCSDUP のユーザー出口 332
DFH0CBDC 323	TSO の下での実行 330
DFH0CRFC 323	システムによる自動インストール 190
DFH0FORC 323	システム・ヘッダー、ジャーナル・レコード 300
端末エラー・プログラム (DFHXTEP) 83	持続セッション・サポート
端末の自動インストール用 150	ノード・エラー・プログラム 135
動的ルーティングの 226	実行開始時レコード、ジャーナル・レコード 295
動的割り振り用 (DYNALLOC) 313	実行診断機能 (EDF)
トランザクション再始動プログラム (DFHREST) 81	タスク関連ユーザー出口 15
ノード・エラー・プログラム (DFHSNEP) 121	ユーザー置き換え可能プログラム 69
プログラム・エラー・プログラム (DFHPEP) 75	実行診断機能 (EDF) (Execution Diagnostic Facility (EDF))
プログラムの自動インストール 195	グローバル・ユーザー出口による 4
分散ルーティング用 247	シッパされた端末、自動インストール 174
APPC 接続の自動インストール用 167	自動インストール、APPC 接続の
CICS-DBCTL インターフェース 状況プログラム (DFHDBUEX) 249	概要 162
IPCONN の自動インストール用 173	サンプル・プログラム
システム管理機能 (SMF) (system management facility (SMF))	デフォルトのアクション 167
製品セクション 310	制御プログラム
ヘッダー 309	インストール時のパラメーター・リスト 163
システム初期設定パラメーター	削除時 166
AIEXIT 137, 163	目的 163
AIRDELAY 135	単一セッション
CLSDSTP 120	BIND によって開始された 162
DSRTPGM 247	CINIT によって開始された 162
DTRPGM 226	テンプレート 162
DTRTRAN 200	に関する要件 163
GMTRAN 118	並列セッション 162
GNTRAN 147, 150	モデル定義 162
PGAICTLG 191	用意されているリソース定義 168
PGAIXIT 191	リカバリーと再始動 163
PGAIPGM 64, 191	利点 162
PLTPI 63	自動インストール、仮想端末の
PLTSD 66	概要 180
RMTRAN 136	制御プログラム
TBEXITS 10	インストール時のパラメーター・リスト 183
システム接頭部、ジャーナル・レコード 302	削除時のパラメーター・リスト 186
システム定義ユーティリティ・プログラム (DFHCSDUP)	自動インストール、シッパされた端末の
サンプル・プログラム	概要 174
CSD 相互参照プログラム 324	制御プログラム
CSD バックアップ・ユーティリティ・プログラム 327	インストール時のパラメーター・リスト 177
Db2 フォーマット設定プログラム 324, 325	削除時のパラメーター・リスト 178
DFH\$CRFA 323	自動インストールに関する ISTINCLM エントリー 351
DFH\$CRFP 323	自動インストールに関する PSERVIC 値 358
DFH\$FORA 323-325	自動インストール・ユーザー置き換え可能プログラム
DFH\$FORP 323-325	仮想端末の
DFH0CBDC 323	DFHZATDX 180
DFH0CRFC 323	DFHZATDY 180
DFH0FORC 323-325	シッパされた端末の
バッチ・プログラムとして	DFHZATDX 174
ユーザー・プログラムが呼び出された場合 321	DFHZATDY 174
ユーザー・プログラム用のリンク・エディット・ステートメント 328	端末用 (DFHZATDX) 150
DFHCSDUP からユーザー・プログラムに渡されるパラメーター 322	ブリッジ・ファシリティーの 180
EXTRACT 処理用のプログラムの作成 321	プログラムの (DFHPDAD) 188
ユーザー・プログラムからの呼び出し	APPC 接続用 (DFHZATDY) 162
概要 330	IPIC 接続 (DFHISAIP) 用 169
ユーザー・プログラムの責任 332	ジャーナル管理ラベル・ヘッダー 297
	ジャーナル・モジュール ID 306
	ジャーナル・レコード
	データ・セクションのフォーマット 312
	モジュール ID 306
	SMF に書き込まれた 307

- ジャーナル・レコード、旧式フォーマット [300](#)
- ジャーナル・レコード・フォーマット
  - フォーマット [275](#)
  - 旧式フォーマットのジャーナル・レコード [300](#)
  - システム接頭部 [302](#)
  - システム・ヘッダー [300](#)
  - 実行開始時レコード [295](#)
  - ジャーナル管理ラベル・ヘッダー [297](#)
  - 新規フォーマットのジャーナル・レコード [277](#)
  - 端末管理接頭部 [292](#)
  - ユーザー接頭部 [303](#)
  - 呼び出し元データ、ファイル制御 [280](#)
  - ラベル接頭部 [297](#)
  - ラベル・ヘッダー [297](#)
  - ログ・ブロック・ヘッダー [276](#)
  - FEPI 接頭部 [293](#)
- シャットダウン (PLTSD) プログラム
  - 作成に関する注意点 [66](#)
- シャットダウン補助トランザクション [67](#)
- シャットダウン補助プログラム、DFHCESD [67](#)
- 初期設定プログラム
  - 作成に関する注意点 [63](#)
- スケジュール・フラグ・ワード [31](#)
- スタブ・プログラム、タスク関連ユーザー出口の [12, 14](#)
- ストレージ内のプロファイル
  - QUERY SECURITY RESCLASS [347](#)
- ストレージ保護機能
  - グローバル・ユーザー出口プログラムとの併用 [8](#)
  - タスク関連ユーザー出口プログラム [34](#)
  - ユーザー置き換え可能プログラム [69](#)
  - PLT プログラムを使用 [67](#)
- セキュリティ検査のカスタマイズ
  - ユーザー ID 変更の通知 [349](#)
- セッション障害、ユーザー・アクション [132](#)
- 総称リソース、z/OS Communications Server
  - ノード・エラー・プログラム [136](#)
- 総称リソース、Communications Server [165](#)

## [タ行]

- 対話式論理装置エラー・プロセッサ [124](#)
- タスク関連のユーザー出口 [12](#)
- タスク関連ユーザー出口
  - EDF の使用 [45](#)
  - アダプター
    - インストールと削除 [46](#)
    - 構造とコンポーネント [12](#)
    - 呼び出し側への応答 [33](#)
  - アドレッシング・モードの影響 [33](#)
  - アプリケーション・プログラムのパラメーター [24](#)
  - 管理 [12, 46](#)
  - グローバル作業域 [36, 47](#)
  - 作業域 [36](#)
  - スケジュール・フラグ・ワード [31](#)
  - スタブ・プログラム
    - ename [14](#)
    - statname [14](#)
  - ストレージ保護との併用
    - 実行キー [34](#)
    - データ記憶キー [34, 68](#)
  - タスク・マネージャーのパラメーター [27](#)
  - タスク・マネージャーの呼び出し
    - 制限 [41](#)
  - 同期点マネージャーのパラメーター [24](#)

- タスク関連ユーザー出口 (続き)
  - 同期点マネージャー呼び出し
    - 再始動再同期 [41](#)
    - サンプル疑似コード [40](#)
    - 変更のコミット [40](#)
    - 変更のバックアウト [40](#)
  - 同期点マネージャー呼び出しのサンプル・コード [40](#)
  - パラメーター・リスト [19](#)
  - パラメーター・リストのアドレッシング機能 [19](#)
  - プロトコル
    - 単一更新 [38](#)
    - 読み取り専用 [38](#)
  - 有効化と開始 [46](#)
  - 有効化と無効化
    - EXEC CICS DISABLE コマンド [47](#)
    - EXEC CICS ENABLE コマンド [47](#)
  - 呼び出し元のパラメーター・リスト [24](#)
  - リカバリーに関する考慮事項 [38](#)
  - ローカル作業域 (local work area) [36](#)
  - CICS コマンドの使用 [35](#)
  - CICS 終了呼び出し
    - サンプル・コード [43](#)
    - 制限 [42](#)
    - DFHEIENT の使用 [43](#)
  - CICS 終了呼び出しのサンプル・コード [43](#)
  - DFHEIENT マクロ [35](#)
  - DFHUEPAR DSECT [19](#)
  - DFHUERTR、機能定義 [23](#)
  - DFHUEXIT TYPE=RM マクロ [19](#)
  - EDF [15](#)
  - EXTRACT コマンド [47](#)
  - SPI パラメーター [24](#)
  - UEPCALAM、呼び出し元の AMODE 標識バイトのアドレス [21](#)
  - UEPEIB、EIB のアドレス [21](#)
  - UEPEXN、機能定義のアドレス [20](#)
  - UEPFLAGS、スケジュール・フラグ・ワードのアドレス [21](#)
  - UEPGAA、グローバル作業域のアドレス [20](#)
  - UEPGAL、グローバル作業域の長さ [20](#)
  - UEPHMSA、レジスター保管域のアドレス [20](#)
  - UEPPBTOK、パフォーマンス・ブロック・トークンのアドレス [23](#)
  - UEPRMQUA、リソース・マネージャーの修飾子名のアドレス [21](#)
  - UEPRMSTK、カーネル・スタック・エントリーのアドレス [21](#)
  - UEPSECBK、ユーザー・セキュリティ・ブロックをアドレス指定するフルワードのアドレス。 [21](#)
  - UEPSECFLG、ユーザー・セキュリティ・フラグのアドレス [21](#)
  - UEPSYNCA、単一更新の標識バイトのアドレス [21](#)
  - UEPTAA、ローカル作業域のアドレス [20](#)
  - UEPTAL、ローカル作業域の長さ [20](#)
  - UEPTIND、呼び出し元のタスク標識のアドレス [22](#)
  - UEPUOWDS、APPC ID のアドレス [21](#)
  - UEPURID、リカバリー単位 ID のアドレス [21](#)
  - UERTFGP、機能グループ標識 [23](#)
  - UERTFID、呼び出し元 ID [23](#)
- タスク関連ユーザー出口の作業域 [36](#)
- タスク関連ユーザー出口のタスク・マネージャー・パラメーター [27](#)
- 端末 ID とエラー・コードの検索 [86](#)
- 端末、自動インストール

端末、自動インストール (続き)  
     SNA 動的 LU 別名 [143](#)  
 端末異常条件回線項目 (TACLE) [83](#)  
 端末異常条件プログラム (DFHTACP) [82](#)  
 端末エラー・プログラム (TEP)  
     異常条件 [82](#)  
     エラー・テーブル [84](#)  
     エラー・プロセッサの置き換え、DFHTEPM  
     TYPE=ERRPROC [94](#)  
     エラー・プロセッサ・のソース [93](#)  
     サンプル  
         アクション・フラグ名 [89](#)  
         入り口と初期設定 [86](#)  
         エラー状況エレメント (ESE) (error status element  
         (ESE)) [84](#)  
         エラー処理の実行 [86](#)  
         エラー・プロセッサの選択 [86](#)  
         概要 [86](#)  
         起動する [86](#)  
         共通のサブルーチン [87](#)  
         コンポーネント [84](#)  
         サンプル・モジュールの生成 [91](#)  
         端末 ID とエラー・コードの検索 [86](#)  
         端末エラー・ブロック (TEB) [84](#)  
         メッセージ [89](#)  
         DECB オペランド [89](#)  
         DECB 情報 [89](#)  
         DFHTEPM TYPE=INITIAL [91](#)  
         ESE 情報 [90](#)  
         TACLE 情報 [90](#)  
     サンプル DFHTEP の生成のためのジョブ制御 [90](#)  
     生成 [90](#)  
     端末異常条件回線項目 (TACLE) [83](#)  
     端末エラー・ブロックの定義  
         テーブル、DFHTEPT TYPE=PERMTID [96](#)  
 通信域  
     内容のアドレッシング [100](#)  
 テーブル  
     デフォルトのしきい値のカウンタ制限 [98](#)  
     DFHTEPT TYPE=BUCKET [98](#)  
     DFHTEPT TYPE=INITIAL [95](#)  
     DFHTEPT マクロの例 [99](#)  
 デフォルト・テーブル [85](#)  
 ユーザー作成プログラム  
     異常条件 [100](#)  
     書き込み打ち切りビット [102](#)  
     ダミー端末標識 [102](#)  
     通信域の内容のアドレッシング [100](#)  
     トランザクション異常終了ビット [102](#)  
     ページ不能タスク [102](#)  
     例 [105](#)  
     TACLE DSECT の形式の説明 [103](#)  
     TACLE の内容のアドレッシング [102](#)  
 CICS コンポーネント [82](#)  
 DFHTEP 再帰的再試行ルーチン  
     システム・カウンタ (TCTTENI) [105](#)  
     ユーザー・フィールド A (PCISAVE) [105](#)  
     ユーザー・フィールド B (PCICNT) [105](#)  
     例 [106](#)  
     DFHTEP テーブル [95](#)  
     DFHTEPM TYPE=ENTRY [93](#)  
     DFHTEPM TYPE=EXIT [93](#)  
     DFHTEPT TYPE=PERMCODE|ERRCODE [96](#)  
 端末エラー・ブロック (TEB) [84](#)

端末エラー・ブロックの定義 [96](#)  
 端末管理  
     ジャーナル・レコード  
         prefix area [292](#)  
 端末の自動インストール  
     サンプル・プログラム  
         カスタマイズ [151](#)  
         DFHZATDX [150](#)  
         DFHZCTDX [150](#)  
         DFHZDZDX [150](#)  
         DFHZPTDX [150](#)  
     制御プログラム  
         インストール時の処理 [138](#)  
         削除時の処理 [145](#)  
         テストおよびデバッグ [146](#)  
         命名 [146](#)  
         戻ったときのアクション [144](#)  
         CICS に返される情報 [141](#)  
         ログオン時のパラメーター・リスト [139](#)  
         z/OS Communications Server LOGON モード・テーブル  
         [137](#)  
 通信域  
     自動インストール制御プログラム (autoinstall control  
     program)  
         端末 [139](#)  
         プログラム [192](#)  
         APPC 接続 [164](#)  
     自動インストール・ユーザー・プログラム  
         IPCONN [171](#)  
     端末エラー・プログラム [83](#), [100](#)  
     動的ルーティング・プログラム [213](#)  
     トランザクション再始動プログラム [80](#)  
     ノード・エラー・プログラム [114](#)  
     分散ルーティング・プログラム [239](#)  
     CICSDBCTL インターフェース 状況プログラム [248](#)  
 データベース制御 (DBCTL)  
     インターフェース 状況 [248](#)  
     DFHDBUEX [248](#)  
 デコード機能、コンバーター・プログラムの  
     出力パラメーター [264](#)  
     入力パラメーター [263](#)  
 デフォルトのしきい値のカウンタ制限  
     DFHTEP (端末エラー・プログラム) [97](#)  
 テンプレート、APPC 接続の自動インストール用の [162](#), [169](#)  
 同期点の管理  
     同期点マネージャのパラメーター [24](#)  
 統計  
     概要 [267](#)  
     グローバル・ユーザー出口 [275](#)  
     そこからの出力の処理 [275](#)  
     データ・セクションのフォーマット [271](#)  
     目的 [267](#)  
     レコード・タイプ [267](#)  
     レコードの形式 [269](#)  
     CICS 統計を収集するプログラムの作成 [267](#)  
     SMF 製品セクション [269](#)  
     SMF ヘッダー [269](#)  
 動的トランザクション [198](#)  
 動的ルーティング  
     概要 [197](#)  
     サービス・プロバイダーの場合 [235](#)  
     サンプル・プログラム [226](#)  
     端末ハンドラーでの [235](#)  
     トランザクションの

## 動的ルーティング (続き)

### トランザクションの (続き)

エラー処理 [201](#)

概要 [197](#)

ターゲット領域の変更 [200](#)

トランザクションの終了 [201](#)

プログラム名の変更 [201](#)

ユーザー・プログラム [198](#)

リソース定義 [198](#)

ルーティング・プログラムに渡される情報 [199](#)

### ブリッジ要求の

エラー処理 [211](#)

再呼び出し [212](#)

パラメーターの変更 [210](#)

要求の終了 [211](#)

ルーティングに対し適格 [209](#)

### プログラム・リンク要求の

エラー処理 [207](#)

ターゲット領域の変更 [206](#)

プログラム名の変更 [206](#)

要求の終了 [207](#)

ルーティングに対し適格 [204](#)

ルーティング・プログラムを呼び出す場合 [205](#)

### ユーザー・プログラム

エラー処理手順 [201, 207](#)

テスト [226](#)

パラメーター [213](#)

命名 [226](#)

呼び出される時点 [198, 205](#)

## 動的ルーティング・プログラム (DFHDYP)

### ブリッジに関する考慮事項 [212](#)

アプリケーションの通信域の変更 [203, 209](#)

異常終了での呼び出し [202, 208](#)

エラー処理 [201, 207](#)

概要 [197](#)

カスタマイズ・バージョンのテスト [226](#)

カスタマイズ・バージョンの名前変更 [226](#)

サンプル・プログラム [226](#)

初期端末データの変更 [202](#)

処理に関する考慮事項 [204, 209](#)

ターゲット領域の変更 [200, 206](#)

通信域 [213](#)

トランザクションのルーティング [201](#)

ブリッジ・パラメーターの変更 [210](#)

ブリッジ要求のルーティング [211, 212](#)

プログラム名の変更 [201, 206](#)

プログラム・リンク要求のルーティング [207](#)

呼び出される時点 [198, 205](#)

ルーティングされた DPL 要求からの情報の受け取り

出力 COMMAREA のモニター [209](#)

ルーティングされたトランザクションからの情報の受信

出力 COMMAREA のモニター [203](#)

出力 TIOA のモニター [204](#)

渡される情報 [199](#)

UOW 考慮事項 [204, 209](#)

## 動的割り振りサンプル・プログラム (DYNALLOC)

値 [314](#)

概要 [313](#)

キーワード、省略規則 [315](#)

システム・プログラミングに関する考慮事項 [315](#)

制御の流れ [316](#)

端末操作 [314](#)

ヘルプ機能 [314](#)

## トランザクション異常終了

プログラム・エラー・プログラム (PEP) [70](#)

トランザクション・クラス・エラー処理ルーチン [114, 133](#)

トランザクション再始動プログラム (DFHREST)

概要 [79](#)

再始動に適したトランザクション [79](#)

通信域 [80](#)

デフォルト・プログラム [81](#)

呼び出される時点 [80](#)

トレース・テーブルのエントリー、グローバル・ユーザー出口インターフェース [4](#)

## [ナ行]

ノード異常条件プログラム (NACP) [113](#)

ノード・エラー・テーブル (NET) [110](#)

ノード・エラー・ハンドラー (CSNE トランザクション) [108](#)

ノード・エラー・プログラム (NEP)

アプリケーション・ルーティング障害 [120](#)

異常条件が発生した場合 [113](#)

エラー・グループ [110](#)

エラー状況ブロック [129](#)

エラー・テーブル・ヘッダー [128](#)

共通サブルーチン・ベクトル・テーブル (CSVT) [129](#)

作成の概要 [109](#)

サンプル

アドレッシング機能 [122](#)

エラー状況情報 [122](#)

エラー・プロセッサー、DFHSNEP TYPE=DEF3270

[125](#)

エラー・プロセッサー・ベクトル・テーブル (EPVT)

[122](#)

エラー・プロセッサー・ベクトル・テーブル (EPVT)

(error processor vector table (EPVT)) [126](#)

オプションの共通サブルーチン [123](#)

共通サブルーチン・ベクトル・テーブル (CSVT) [122](#)

コーディングの記述 [110](#)

コンポーネント [121](#)

サンプル TEP との互換性 [121](#)

状態 [112](#)

ノード・エラー・テーブル [122](#)

ルーティング・メカニズム (ACF/SNA) [122](#)

3270 用のオプションのエラー・プロセッサー [123](#)

CSVT (共通サブルーチン・ベクトル・テーブル) [122](#)

DFHSNEP TYPE=INITIAL マクロ [125](#)

DFHSNEP TYPE=USTOR マクロ [124](#)

DFHSNEP TYPE=USTOREND マクロ [124](#)

DFHSNEP による生成 [124](#)

INTLU のエラー・プロセッサー、DFHSNEP

TYPE=DEFILU [125](#)

INTLU 用オプション・エラー・プロセッサー [124](#)

持続セッション・サポート [135](#)

セッション障害 [132](#)

端末管理プログラム (ACF/SNA セクション) [113](#)

通信域 [114](#)

デフォルト・トランザクション・クラス・ルーチン [133](#)

デフォルトのノード・エラー・プログラム [109](#)

独自に作成する理由 [108](#)

ノード異常条件プログラム [113](#)

ノード・エラー・テーブル

生成 [110](#)

フォーマット [123](#)

ノード・エラー・ブロック [128](#)

ノード・エラー・ブロック、形式 [123](#)



## ノード・エラー・プログラム (NEP) (続き)

### 複数の NEP [112](#)

#### ユーザー作成

アドレッシング機能 [130](#)

使用に関する制限 [130](#)

ユーザー作成エラー・プロセッサ [126](#)

ユーザー提供エラー・プロセッサ、DFHSNEP

TYPE=ERRPROC [126](#)

ルーティングに関する注意点 [113](#)

レジスターの規則 [126](#)

3270: unavailable printer [130](#)

ACF/Communications Server のエラー処理

背景情報 [108](#)

DFHNET DSECT [128](#)

DFHSNET [127](#)

DFHZNAC [113](#)

DFHZNAC アクション・フラグの設定値 [386](#)

DFHZNAC のデフォルトのアクション

システム・センス・コードに関する [384](#)

端末エラー・コードに関する [371](#)

DFHZNAC ログ機能 [120](#)

DFHZNAC/DFHZNAP インターフェース [108](#)

DFHZNAP [108](#), [113](#)

DFHZNAPI TYPE=INITIAL [133](#)

DFHZNAPI インターフェース・モジュール [132](#)

DFHZNAPI マクロ [132](#)

DSECT [128](#)

NEPCLASS [112](#)

NET 生成 [110](#)

TERMERR 状態 [107](#)

XRF 環境における

リカバリー通知の変更 [135](#)

リカバリー・トランザクションの変更 [136](#)

リカバリー・メッセージの変更 [135](#)

z/OS Communications Server 総称リソースを指定 [136](#)

## ノード・エラー・ブロック (NEB) [128](#)

## [ハ行]

ページ不能タスク [102](#)

### パス

DFHWPBAXD による解釈、サンプル・アナライザー・プログラム [259](#)

### 非端末セキュリティ

接続検査の抑止 [348](#)

### ファイル制御

ジャーナル・レコード

書き込み更新レコード・タイプ [281](#)

書き込み削除レコード・タイプ [284](#)

書き込み追加の完了レコード・タイプ [281](#)

書き込み追加レコード・タイプ [281](#)

タイアップ・レコード・タイプ [289](#)

ファイル・クローズ・レコード・タイプ [288](#)

読み取り更新レコード・タイプ [281](#)

読み取り専用レコード・タイプ [281](#)

FILE\_CLOSE\_DATA セクション [288](#)

FLJB [280](#)

FLJB\_COMMON\_DATA セクション [281](#)

FLJB\_GENERAL\_DATA セクション [281](#)

FLJB\_WRITE\_DELETE\_DATA セクション [284](#)

TIE\_UP\_RECORD\_DATA セクション [289](#)

ブリッジ (3270)

ブリッジ出口 [249](#)

ブリッジ・ファシリティー

## ブリッジ・ファシリティー (続き)

自動インストール制御プログラム

インストール時のパラメーター・リスト [185](#)

自動インストール・ユーザー置き換え可能プログラム [180](#)

ブリッジ要求の動的ルーティング

ルーティングに対し適格 [209](#)

プログラム、自動インストール [188](#)

プログラム・エラー・プログラム (PEP)

作成 [70](#)

ソース・コード [70](#)

プログラムの自動インストール

概要 [188](#)

サンプル・プログラム

カスタマイズ [195](#)

DFHPGADX [195](#)

DFHPGAHX [195](#)

DFHPGALX [195](#)

DFHPGAOX [195](#)

システムによる自動インストール [190](#)

制御プログラム

インストール時のパラメーター・リスト [192](#)

テスト [197](#)

マップ・セットのインストール [190](#)

モデル定義 [189](#)

用意されているリソース定義 [196](#)

要件 [191](#)

利点 [190](#)

プログラム・リスト・テーブル (PLT) プログラム

一般的な考慮事項 [67](#)

ストレージ保護との併用

実行キー [68](#)

データ記憶キー [68](#)

PLTPI プログラム

概要 [63](#)

第 1 フェーズ [63](#)

第 2 フェーズ [64](#)

PLTSD プログラム

概要 [66](#)

第 1 静止フェーズ [66](#)

第 2 フェーズ [66](#)

分散プログラム・リンク (DPL)

要求の動的ルーティング

エラー処理 [207](#)

ターゲット領域の変更 [206](#)

プログラム名の変更 [206](#)

要求の終了 [207](#)

ルーティングに対し適格 [204](#)

ルーティング・プログラムを呼び出す場合 [205](#)

分散ルーティング

インバウンド Web サービス要求の

エラー処理 [237](#)

ターゲット領域の変更 [236](#)

ルーティングに対し適格 [235](#)

ルーティング・プログラムを呼び出す場合 [236](#)

ローカルでのトランザクションの実行 [237](#)

概要 [227](#)

サンプル・プログラム [247](#)

非端末関連 START 要求

エラー処理 [234](#)

ターゲット領域の変更 [233](#)

ルーティングに対し適格 [231](#)

ルーティング・プログラムを呼び出す場合 [232](#)

ローカルでのトランザクションの実行 [233](#)



## 分散ルーティング (続き)

### ユーザー・プログラム

エラー処理手順 [231](#), [234](#), [237](#)

パラメーター [239](#)

命名 [247](#)

呼び出される時点 [229](#), [232](#), [236](#)

### BTS アクティビティの

エラー処理 [231](#)

ターゲット領域の変更 [230](#)

ルーティングに対し適格 [229](#)

ルーティング・プログラムを呼び出す場合 [229](#)

ローカルでのアクティビティの実行 [230](#)

## 分散ルーティング・プログラム (DFHDSRP)

異常終了での呼び出し [231](#), [235](#), [237](#)

インバウンド Web サービス要求のルーティング [237](#)

エラー処理 [231](#), [234](#), [237](#)

カスタマイズ・バージョンの名前変更 [247](#)

サンプル・プログラム [247](#)

処理に関する考慮事項 [238](#)

ターゲット領域の変更 [230](#), [233](#), [236](#)

通信域 [239](#)

非端末関連 START 要求のルーティング [233](#)

呼び出される時点 [229](#), [232](#), [236](#)

BTS アクティビティのルーティング [230](#)

## 分散ルーティング・プログラム、DFHDSRP

概要 [227](#)

動的ルーティング・プログラムとの相違点 [227](#)

## [マ行]

### モデル端末サポート

エントリーのコーディング [138](#)

### モデル定義

プログラムの自動インストール [189](#)

APPC 接続の自動インストール [162](#)

### 問題判別

CICS セキュリティー制御点 [345](#)

## [ヤ行]

### ユーザー置き換え可能プログラム

アセンブルとリンク・エディット [337](#)

コンソールの自動インストール用 (DFHZATDX) [156](#)

再作成 [69](#)

自動インストール用、APPC 接続の (DFHZATDY) [162](#)

自動インストール用、仮想端末の

DFHZATDX [180](#)

DFHZATDY [180](#)

自動インストール用、シッパされた端末の

DFHZATDX [174](#)

DFHZATDY [174](#)

ストレージ保護との併用

実行キー [69](#)

データ記憶キー [69](#)

端末エラー・プログラム (DFHTEP) [82](#)

端末の自動インストール (DFHZATDX) [136](#)

動的ルーティング・プログラム (DFHDYP) [197](#)

トランザクション再始動プログラム (DFHREST) [79](#)

ノード・エラー・プログラム (DFHZNEP) [107](#)

プログラム・エラー・プログラム (DFHPEP) [70](#)

プログラムの自動インストール用 (DFHPGADX) [188](#)

分散ルーティング・プログラム (DFHDSRP) [227](#)

3270 ブリッジ出口 [249](#)

### ユーザー置き換え可能プログラム (続き)

DBCTL インターフェース 状況プログラム (DFHDBUEX) [248](#)

EDF による検査 [69](#)

IPIC 接続 (DFHISAIP) の自動インストール用 [169](#)

XPLINK ランタイム・オプション・プログラム

(DFHAPXPO) [250](#)

ユーザー作成のノード・エラー・プログラム [129](#)

ユーザー接頭部、ジャーナル・レコード [303](#)

ユーザー置換可能プログラムのアセンブル [337](#)

ユーザー置換可能プログラムのアセンブルとリンク・エディット [337](#)

ユーザー置換可能プログラムのリンク・エディット [337](#)

ユーザー提供エラー・プロセッサ、DFHSNEP

TYPE=ERRPROC [126](#)

ユーザー出口

グローバル [1](#)

タスク関連 [12](#)

DFHCSDUP の [332](#)

ICHRFX01 RACF ユーザー出口 [348](#)

XSNOFF グローバル・ユーザー出口 [349](#)

XSNON グローバル・ユーザー出口 [349](#)

ユーティリティー・プログラム

シャットダウン補助、DFHCESD [67](#)

抑止、非端末トランザクションの接続検査の [348](#)

呼び出し側プログラムのレジスター [33](#)

## [ラ行]

ラベル接頭部 [297](#)

ラベル・ヘッダー [297](#)

リカバリーと再始動

ノード・エラー・プログラム (DFHZNEP) [107](#)

プログラム・エラー・プログラム (PEP) [70](#)

ルーティング・メカニズム (ACF/SNA) [122](#)

リソース定義

アナライザー・プログラム [254](#)

リソース定義オンライン・トランザクション

それに対するプログラマブル・インターフェース [319](#)

DFHEDAP への EXEC CICS LINK [319](#)

リソース・マネージャー・インターフェース (RMI) [12](#)

ルーティング・メカニズム、z/OS Communications Server [122](#)

ログ・ブロック・ヘッダー、ジャーナル・レコード [276](#)

論理装置 (LU)

ノード・エラー・プログラム [113](#)

## [ワ行]

ワークロード管理

サービス・プロバイダーの場合 [235](#)

端末ハンドラーでの [235](#)

## [数字]

3270 情報表示システム

エラー・プロセッサ (オプション) [123](#)

unavailable printer

DFHZNEP [130](#)

3270 ブリッジ

ブリッジ出口 [249](#)

ブリッジ出口プログラム [249](#)

## A

ACEE (アクセス制御環境エレメント) [345](#)  
ACF/ z/OS Communications Server  
  ISTINCLM 値 [351](#)  
  LOGON モード・テーブル内の項目 [351](#)  
  z/OS Communications Server LOGON モード・テーブル [137](#)  
ACF/Communications Server  
  エラー処理  
    DFHZNAC/DFHZNEP インターフェース [108](#)  
  総称リソース [165](#)  
  デフォルト DFHZNEP [108](#)  
ACF/SNA  
  アプリケーション・ルーティング障害 [120](#)  
  自動インストール [136](#)  
  セッション障害  
    ユーザー作成の NEP [132](#)  
  総称リソース  
    ノード・エラー・プログラム [136](#)  
  トランザクション・クラス・エラー処理ルーチン [114](#)  
  ノード・エラー・プログラム (DFHZNEP) [121](#)  
  CLSDST PASS 機能 [120](#)  
  DFHZNAC ログ機能 [120](#)  
ACF/SNA z/OS Communications Server  
  エラー処理  
    DFHZNAC/DFHZNEP インターフェース・アクション・フラグ [109](#)  
ACF/z/OS Communications Server  
  PSERVIC 値 [358](#)  
ADYN、動的割り振りトランザクション [314](#)  
AIEXIT、システム初期設定パラメーター [137](#), [163](#)  
AIRDELAY、システム初期設定パラメーター [135](#)  
API コマンド、コンバーター・プログラムにおける [261](#)  
APPC 接続、自動インストール [162](#)

## B

BTS アクティビティの分散ルーティング  
  ルーティングに対し適格 [229](#)

## C

CEDA トランザクション  
  それに対するプログラマブル・インターフェース [319](#)  
CEMT INQUIRE AUTOINSTALL [146](#)  
CEMT SET AUTOINSTALL [146](#)  
CESD、デフォルトのシャットダウン補助トランザクション [67](#)  
CICS システム定義ユーティリティ・プログラム (DFHCSDUP)  
  サンプル・プログラム  
    CSD 相互参照プログラム [324](#)  
    CSD バックアップ・ユーティリティ・プログラム [327](#)  
    Db2 フォーマット設定プログラム [324](#), [325](#)  
    DFH\$CRFA [323](#)  
    DFH\$CRFP [323](#)  
    DFH\$FORA [323](#)  
    DFH\$FORP [323](#)  
    DFH0CBDC [323](#)  
    DFH0CRFC [323](#)  
    DFH0FORC [323](#)

CICS システム定義ユーティリティ・プログラム (DFHCSDUP) (続き)  
  バッチ・プログラムとして  
    ユーザー・プログラムが呼び出された場合 [321](#)  
    ユーザー・プログラム用のリンク・エディット・ステートメント [328](#)  
  DFHCSDUP からユーザー・プログラムに渡されるパラメーター [322](#)  
  EXTRACT 処理用のプログラムの作成 [321](#)  
  ユーザー・プログラムからの呼び出し  
    概要 [330](#)  
    ユーザー・プログラムの責任 [332](#)  
    DFHCSDUP の入力パラメーター [330](#)  
    DFHCSDUP のユーザー出口 [332](#)  
  CSD 相互参照プログラム  
    DFH\$CRFA [324](#)  
    DFH\$CRFP [324](#)  
    DFH0CRFC [324](#)  
  TSO の下での実行 [330](#)  
CICS 統計の出力の処理 [275](#)  
CICS レジスター [33](#)  
CICS-RACF インターフェースのカスタマイズ  
  概要 [339](#)  
  CICS セキュリティ制御点 [345](#)  
  RACROUTE マクロ [345](#)  
CICS-RACF セキュリティ・インターフェース  
  CICS セキュリティ制御点 [345](#)  
  RACROUTE マクロ [345](#)  
CICS-DBCTL インターフェース 状況プログラム (DFHDBUEX)  
  概要 [248](#)  
  サンプル・プログラム [249](#)  
  通信域 [248](#)  
CINIT、z/OS Communications Server [146](#)  
CLSDSTP、システム初期設定パラメーター [120](#)  
CODE オペランド  
  DFHSNEP TYPE=ERRPROC [126](#)  
  DFHTEPM TYPE=ERRPROC [94](#)  
  DFHTEPT TYPE=BUCKET [99](#)  
  DFHTEPT TYPE=PERMCODE|ERRCODE [97](#)  
COUNT オペランド  
  制限、TEP のデフォルトのしきい値 [97](#)  
  DFHSNET マクロ [128](#)  
  DFHTEPT TYPE=PERMCODE|ERRCODE [97](#)  
CS オペランド  
  DFHSNEP TYPE=INITIAL [125](#)  
CSD 相互参照プログラム  
  DFHCSDUP 用 [324](#)  
CSD バックアップ・ユーティリティ・プログラム  
  DFH0CBDC [327](#)  
  DFHCSDUP 用 [327](#)  
CSD ユーティリティ・プログラム (DFHCSDUP) [327](#)  
CSNE トランザクション [108](#)  
CSV (共通サブルーチン・ベクトル・テーブル) [122](#)

## D

Db2 フォーマット設定プログラム  
  DFHCSDUP 用 [324](#), [325](#)  
DBCTL (データベース制御)  
  インターフェース 状況 [248](#)  
  DFHDBUEX [248](#)  
DECB、端末エラー・プログラム  
  オペランド [89](#)  
  情報 [89](#)

DEFAULT オペランド  
     DFHZNEPI TYPE=INITIAL [133](#)  
 DFH\$APDT  
     アダプター 追跡サンプル [48](#)  
 DFH\$CRFA、相互参照プログラム、アセンブラー言語 [323](#)  
 DFH\$CRFP、相互参照プログラム、PL/I [323](#)  
 DFH\$FORA [325](#)  
 DFH\$FORA、フォーマット設定プログラム、アセンブラー言語 [323](#)  
 DFH\$FORP [325](#)  
 DFH\$FORP、フォーマット設定プログラム、PL/I [323](#)  
 DFH\$ISAI、ユーザーが置換可能な自動インストール・プログラム [174](#)  
 DFH0CBDC プログラム、COBOL 用 DEFINE コマンドの作成 [323](#)  
 DFH0CRFC、相互参照プログラム、COBOL [323](#)  
 DFH0FORC [325](#)  
 DFH0FORC、フォーマット設定プログラム、COBOL [323](#)  
 DFH0GNIT、サンプル「good night」プログラム [149](#)  
 DFH0ISAI、ユーザーが置換可能な自動インストール・プログラム [174](#)  
 DFHAPXPO、XPLINK ランタイム・オプション・プログラム [250](#)  
 DFHCESD、シャットダウン補助プログラム [67](#)  
 DFHCSDUP、システム定義ユーティリティ・プログラム  
     サンプル・プログラム  
         CSD 相互参照プログラム [324](#)  
         CSD バックアップ・ユーティリティ・プログラム [327](#)  
         Db2 フォーマット設定プログラム [324](#), [325](#)  
         DFH\$CRFA [323](#)  
         DFH\$CRFP [323](#)  
         DFH\$FORA [323](#)  
         DFH\$FORP [323](#)  
         DFH0CBDC [323](#)  
         DFH0CRFC [323](#)  
         DFH0FORC [323](#)  
     バッチ・プログラムとして  
         ユーザー・プログラムが呼び出された場合 [321](#)  
         ユーザー・プログラム用のリンク・エディット・ステートメント [328](#)  
         DFHCSDUP からユーザー・プログラムに渡されるパラメーター [322](#)  
         EXTRACT 処理用のプログラムの作成 [321](#)  
     ユーザー・プログラムからの呼び出し  
         概要 [330](#)  
         ユーザー・プログラムの責任 [332](#)  
         DFHCSDUP の入力パラメーター [330](#)  
     TSO の下での実行 [330](#)  
 DFHDBUEX、CICS-DBCTL インターフェース 状況プログラム  
     概要 [248](#)  
     サンプル・プログラム [249](#)  
     通信域 [248](#)  
 DFHDSRP、分散ルーティング・プログラム  
     異常終了での呼び出し [231](#), [235](#), [237](#)  
     インバウンド Web サービス要求のルーティング [237](#)  
     エラー処理 [231](#), [234](#), [237](#)  
     概要 [227](#)  
     カスタマイズ・バージョンの名前変更 [247](#)  
     サンプル・プログラム [247](#)  
     処理に関する考慮事項 [238](#)  
     ターゲット領域の変更 [230](#), [233](#), [236](#)  
     通信域 [239](#)  
     動的ルーティング・プログラムとの相違点 [227](#)  
 DFHDSRP、分散ルーティング・プログラム (続き)  
     非端末関連 START 要求のルーティング [233](#)  
     呼び出される時点 [229](#), [232](#), [236](#)  
     BTS アクティビティのルーティング [230](#)  
 DFHDYP、動的ルーティング・プログラム  
     ブリッジに関する考慮事項 [212](#)  
     アプリケーションの通信域の変更 [203](#), [209](#)  
     異常終了での呼び出し [202](#), [208](#)  
     エラー処理 [201](#), [207](#)  
     概要 [197](#)  
     カスタマイズ・バージョンのテスト [226](#)  
     カスタマイズ・バージョンの名前変更 [226](#)  
     サンプル・プログラム [226](#)  
     初期端末データの変更 [202](#)  
     処理に関する考慮事項 [204](#), [209](#)  
     ターゲット領域の変更 [200](#), [206](#)  
     通信域 [213](#)  
     トランザクションのルーティング [201](#)  
     ブリッジ・パラメーターの変更 [210](#)  
     ブリッジ要求のルーティング [211](#), [212](#)  
     プログラム名の変更 [201](#), [206](#)  
     プログラム・リンク要求のルーティング [207](#)  
     呼び出される時点 [198](#), [205](#)  
     ルーティングされた DPL 要求からの情報の受け取り  
         出力 COMMAREA のモニター [209](#)  
     ルーティングされたトランザクションからの情報の受信  
         出力 COMMAREA のモニター [203](#)  
         出力 TIOA のモニター [204](#)  
     渡される情報 [199](#)  
     UOW 考慮事項 [204](#), [209](#)  
 DFHEIP、EXEC インターフェース・プログラム [337](#)  
 DFHISAIP、ユーザー置き換え可能自動インストール・プログラム  
     概要 [169](#)  
     サンプル・プログラム [173](#)  
     通信域 [171](#)  
     デフォルトのアクション [173](#)  
     目的 [169](#)  
     用意されている定義 [173](#)  
     呼び出される時点 [171](#)  
 DFHNET DSECT [128](#)  
 DFHPEP、プログラム・エラー・プログラム  
     作成 [70](#)  
     ソース・コード [70](#)  
 DFHPGADX、ユーザー置換可能自動インストール・プログラム  
     インストール時のパラメーター・リスト [192](#)  
     概要 [188](#)  
     カスタマイズ [195](#)  
     サンプル・プログラム [195](#)  
     マップ・セットのインストール [190](#)  
     モデル定義の使用 [189](#)  
     用意されている定義 [196](#)  
     呼び出される時点 [188](#)  
 DFHREST、トランザクション再始動プログラム  
     概要 [79](#)  
     再始動に適したトランザクション [79](#)  
     通信域 [80](#)  
     デフォルト・プログラム [81](#)  
     呼び出される時点 [80](#)  
 DFHRMCAL マクロ [12](#)  
 DFHSNEP マクロ  
     TYPE=DEF3270 [125](#)

DFHSNEP マクロ (続き)  
 TYPE=DEFILU [126](#)  
 TYPE=ERRPROC [111](#), [126](#)  
 TYPE=FINAL [126](#)  
 TYPE=INITIAL [110](#), [125](#)  
 TYPE=USTOR [124](#)  
 TYPE=USTOREND [124](#)  
 DFHSNEP、サンプル・ノード・エラー・プログラム [124](#)  
 DFHSNET マクロ  
 COUNT オペランド [128](#)  
 ESB 構造 [128](#)  
 ESBS オペランド [128](#)  
 NAME オペランド [127](#)  
 NEBNAME オペランド [128](#)  
 NEBS オペランド [128](#)  
 TIME オペランド [128](#)  
 DFHSTUP、統計処理プログラム [275](#)  
 DFHTACP、端末異常条件プログラム  
 端末エラー処理 [82](#)  
 DFHTEP、端末エラー・プログラム  
 リンク・エディット・ステートメント [338](#)  
 DFHTEPM マクロ  
 例 [94](#)  
 TYPE=ENTRY [93](#)  
 TYPE=ERRPROC [94](#)  
 TYPE=EXIT [93](#)  
 TYPE=FINAL [94](#)  
 TYPE=INITIAL [91](#)  
 DFHTEPT マクロ  
 例 [99](#)  
 TYPE=BUCKET [98](#)  
 TYPE=FINAL [99](#)  
 TYPE=INITIAL [95](#)  
 TYPE=PERMCODE|ERRCODE [96](#)  
 TYPE=PERMTID [96](#)  
 DFHUEPAR DSECT [5](#), [20](#)  
 DFHUERTR DSECT [23](#)  
 DFHUEXIT マクロ [5](#)  
 DFHWBAAX、デフォルトのアナライザー・プログラム  
 概要 [250](#)  
 動作 [258](#)  
 DFHWBADX、サンプル・アナライザー・プログラム  
 応答 [259](#)  
 概要 [250](#)  
 要求 URL 形式 [259](#)  
 DFHXTEP、サンプル端末エラー・プログラム [83](#)  
 DFHZATDX、ユーザー置き換え可能自動インストール・プログラム  
 インストール時の処理 [138](#)  
 インストール用に使用、仮想端末を [180](#)  
 インストール用に使用、シップされた端末を [174](#)  
 概要 [136](#)  
 カスタマイズ [151](#)  
 コンソール用 [156](#)  
 削除時の処理 [145](#)  
 サンプル制御プログラム [151](#)  
 使用の提案 [151](#)  
 ソース・コード [150](#)  
 通信域 [145](#)  
 DFHZATDY、ユーザー置き換え可能自動インストール・プログラム  
 インストール用に使用、仮想端末を [180](#)  
 インストール用に使用、シップされた端末を [174](#)  
 概要 [162](#)

DFHZATDY、ユーザー置き換え可能自動インストール・プログラム (続き)  
 サンプル・プログラム [167](#)  
 単一セッションの APPC 接続  
 BIND によって開始された [162](#)  
 通信域 [164](#)  
 デフォルトのアクション [167](#)  
 並列セッションの APPC 接続 [162](#)  
 目的 [163](#)  
 用意されている定義 [168](#)  
 呼び出される時点 [163](#)  
 APPC 単一セッション接続  
 CINIT によって開始された [162](#)  
 DFHZNAC、ノード異常条件プログラム  
 アクション・フラグの設定値 [386](#)  
 持続セッションがサポートされる場合の実行 [135](#)  
 端末エラー処理 [113](#)  
 デフォルトのアクション  
 システム・センス・コードに関する [384](#)  
 端末エラー・コードに関する [371](#)  
 ログ機能 [120](#)  
 z/OS Communications Server 総称リソースを指定した実行 [136](#)  
 DFHZNEP、ノード・エラー・プログラム  
 リンク・エディット・ステートメント [338](#)  
 DFHZNEP、ユーザー置き換え可能ノード・エラー・プログラム [107](#)  
 DFHZNEPI マクロ  
 TYPE=ENTRY [133](#)  
 TYPE=FINAL [133](#)  
 TYPE=INITIAL [133](#)  
 DPL 要求の動的ルーティング  
 ルーティングに対し適格 [204](#)  
 ルーティング・プログラムを呼び出す場合 [205](#)  
 DSECTPR オペランド  
 DFHTEPM TYPE=INITIAL [91](#)  
 DSRTPGM、初期設定パラメーター [247](#)  
 DTRPGM、システム初期設定パラメーター [226](#)  
 DTRTRAN、システム初期設定パラメーター [200](#)  
 DYNAMIC オプション [198](#)

## E

EDF (実行診断機能)  
 グローバル・ユーザー出口による [4](#)  
 タスク関連ユーザー出口 [15](#)  
 ESB (エラー状況ブロック) [128](#)  
 ESBS オペランド  
 DFHSNET マクロ [128](#)  
 ESE (エラー状況エレメント)  
 DFHTEPT TYPE=PERMCODE|ERRCODE [96](#)  
 EXEC CICS HANDLE コマンド  
 ノード・エラー・プログラムの代替として [107](#)  
 EXEC CICS INQUIRE コマンド  
 自動インストールの場合 [146](#)  
 EXEC CICS SET コマンド  
 自動インストールの場合 [146](#)  
 EXEC インターフェース・プログラム (DFHEIP) [337](#)  
 EXTRACT TCPIP コマンド  
 使用、アナライザー・プログラムでの [254](#)  
 EXTRACT コマンド  
 タスク関連のユーザー出口の [47](#)

## F

### FEPI

ジャーナル・レコード

prefix area [293](#)

FILE\_CLOSE\_DATA セクション、ジャーナル・レコード [288](#)

FLJB\_COMMON\_DATA セクション、ジャーナル・レコード [281](#)

FLJB\_GENERAL\_DATA セクション、ジャーナル・レコード [281](#)

FLJB\_WRITE\_DELETE\_DATA セクション、ジャーナル・レコード [284](#)

FLJB、ファイル制御ジャーナル・ブロック [280](#)

## G

### GLUE 1

GLUE、作成 [1](#)

GMTRAN、システム初期設定パラメーター [118](#)

GNTRAN、システム初期設定パラメーター [147](#), [150](#)

「good night」トランザクション

概要 [147](#)

サンプル・プログラム、DFH0GNIT [149](#)

サンプル・プログラムのカスタマイズ [149](#)

GROUP オペランド

DFHSNEP TYPE=ERRPROC [126](#)

## I

ICHRFX01 RACF ユーザー出口 [349](#)

INTLU エラー・プロセッサ [124](#)

IP 接続の自動インストール

用意されているリソース定義 [173](#)

IPCONN の自動インストール

サンプル・プログラム

接続テンプレート [174](#)

デフォルトのアクション [173](#)

自動インストールの利点 [169](#)

テンプレート [169](#)

ユーザー・プログラム

インストール時のパラメーター・リスト [171](#)

削除時 [172](#)

目的 [169](#)

要件 [169](#)

IPIC 接続、自動インストール [169](#)

IPIC 接続の自動インストール

概要 [169](#)

事前の考慮事項 [169](#)

ISSUE PASS コマンド [120](#)

## L

LOGON モード・テーブル、z/OS Communications Server [351](#)

LU 別名 [143](#)

LU、自動インストール [136](#)

## M

MAXERRS オペランド

DFHTEPT TYPE=INITIAL [96](#)

MAXTIDS オペランド

DFHTEPT TYPE=INITIAL [96](#)

MVS コンソール

自動インストール [156](#)

## N

NAME オペランド

DFHSNEP TYPE=INITIAL [125](#)

DFHSNET マクロ [127](#)

NEB (ノード・エラー・ブロック) [128](#)

NEBNAME オペランド

DFHSNET マクロ [128](#)

NEBS オペランド

DFHSNET マクロ [128](#)

NEP (ノード・エラー・プログラム)

アプリケーション・ルーティング障害 [120](#)

異常条件が発生した場合 [113](#)

エラー・グループ [110](#)

エラー状況ブロック [129](#)

エラー・テーブル・ヘッダー [128](#)

共通サブルーチン・ベクトル・テーブル (CSV) [129](#)

作成の概要 [109](#)

サンプル

アドレッシング機能 [122](#)

エラー状況情報 [122](#)

エラー・プロセッサ、DFHSNEP TYPE=DEF3270 [125](#)

エラー・プロセッサ・ベクトル・テーブル (EPVT) [122](#)

エラー・プロセッサ・ベクトル・テーブル (EPVT) (error processor vector table (EPVT)) [126](#)

オプションの共通サブルーチン [123](#)

共通サブルーチン・ベクトル・テーブル (CSV) [122](#)

コーディングの記述 [110](#)

コンポーネント [121](#)

サンプル TEP との互換性 [121](#)

状態 [112](#)

ノード・エラー・テーブル [122](#)

ルーティング・メカニズム (ACF/SNA) [122](#)

3270 用のオプションのエラー・プロセッサ [123](#)

CSV (共通サブルーチン・ベクトル・テーブル) [122](#)

DFHSNEP TYPE=INITIAL マクロ [125](#)

DFHSNEP TYPE=USTOR マクロ [124](#)

DFHSNEP TYPE=USTOREND マクロ [124](#)

DFHSNEP による生成 [124](#)

INTLU のエラー・プロセッサ、DFHSNEP

TYPE=DEFILU [125](#)

INTLU 用オプション・エラー・プロセッサ [124](#)

持続セッション・サポート [135](#)

セッション障害 [132](#)

端末管理プログラム (ACF/SNA セクション) [113](#)

通信域 [114](#)

デフォルト・トランザクション・クラス・ルーチン [133](#)

デフォルトのノード・エラー・プログラム [109](#)

独自に作成する理由 [108](#)

ノード異常条件プログラム [113](#)

ノード・エラー・テーブル

生成 [110](#)

フォーマット [123](#)

ノード・エラー・ブロック [128](#)

ノード・エラー・ブロック、形式 [123](#)

複数の NEP [112](#)

ユーザー作成

アドレッシング機能 [130](#)

使用に関する制限 [130](#)



NEP (ノード・エラー・プログラム) (続き)  
ユーザー作成エラー・プロセッサ [126](#)  
ユーザー提供エラー・プロセッサ、DFHSNEP  
TYPE=ERRPROC [126](#)  
ルーティングに関する注意点 [113](#)  
レジスタの規則 [126](#)  
3270: unavailable printer [130](#)  
ACF/Communications Server のエラー処理  
背景情報 [108](#)  
DFHNET DSECT [128](#)  
DFHSNET [127](#)  
DFHZNAC [113](#)  
DFHZNAC アクション・フラグの設定値 [386](#)  
DFHZNAC のデフォルトのアクション  
システム・センス・コードに関する [384](#)  
端末エラー・コードに関する [371](#)  
DFHZNAC ログ機能 [120](#)  
DFHZNAC/DFHZNAP インターフェース [108](#)  
DFHZNAP [108](#), [113](#)  
DFHZNAPI TYPE=INITIAL [133](#)  
DFHZNAPI インターフェース・モジュール [132](#)  
DFHZNAPI マクロ [132](#)  
DSECT [128](#)  
NEPCLASS [112](#)  
NET 生成 [110](#)  
TERMERR 状態 [107](#)  
XRF 環境における  
リカバリー通知の変更 [135](#)  
リカバリー・トランザクションの変更 [136](#)  
リカバリー・メッセージの変更 [135](#)  
z/OS Communications Server 総称リソースを指定 [136](#)  
NEPCLAS オペランド  
DFHZNAPI TYPE=ENTRY [133](#)  
NEPCLASS オペランド  
CEDA [112](#)  
NEPNAME オペランド  
DFHZNAPI TYPE=ENTRY [133](#)  
NET (ノード・エラー・テーブル) [110](#)  
NETNAME オペランド  
DFHSNEP TYPE=INITIAL [125](#)

## O

OPTIONS オペランド  
DFHTEPM TYPE=INITIAL [92](#)  
DFHTEPT TYPE=INITIAL [96](#)

## P

PEP (プログラム・エラー・プログラム)  
作成 [70](#)  
ソース・コード [70](#)  
PGAICTLG、システム初期設定パラメーター [191](#)  
PGAIXIT、システム初期設定パラメーター [191](#)  
PGAIPGM、システム初期設定パラメーター [64](#), [191](#)  
PLT プログラム [67](#)  
PLTPI プログラム  
一般的な考慮事項 [67](#)  
概要 [63](#)  
第1 フェーズ [63](#)  
第2 フェーズ [64](#)  
PLTPI、システム初期設定パラメーター [63](#)  
PLTSD プログラム

PLTSD プログラム (続き)  
一般的な考慮事項 [67](#)  
概要 [66](#)  
第1 静止フェーズ [66](#)  
第2 フェーズ [66](#)  
PLTSD、システム初期設定パラメーター [66](#)  
PRINT オペランド  
DFHTEPM TYPE=INITIAL [92](#)  
PROGRAM 定義  
アナライザー・プログラム [254](#)

## R

RACROUTE マクロ [345](#)  
RDO トランザクション  
それに対するプログラマブル・インターフェース [319](#)  
DFHEDAP への EXEC CICS LINK [319](#)  
RDO トランザクションとのプログラマブル・インターフェース [319](#)  
RMI (リソース・マネージャー・インターフェース) [12](#)  
RMI レジスター [33](#)  
RMTRAN、システム初期設定パラメーター [136](#)

## S

SMF (システム管理機能)  
製品セクション [269](#), [310](#)  
ヘッダー [269](#), [309](#)  
SNA 動的 LU 別名 [143](#)

## T

TACLE (端末異常条件回線項目)  
端末エラー・プログラム [83](#)  
内容のアドレッシング [102](#)  
DSECT、形式の説明 [103](#)  
TASKSTART [48](#)  
TBEXITS、システム初期設定パラメーター [10](#)  
TCP (端末管理プログラム)  
ACF/SNA セクション [113](#)  
TACLE (端末異常条件回線項目) [82](#)  
TCIPSERVICE リソース定義  
アナライザー・プログラムの役割 (URM) [250](#)  
TEB (端末エラー・ブロック) [84](#)  
TEP (端末エラー・プログラム)  
異常条件 [82](#)  
エラー・テーブル [84](#)  
エラー・プロセッサの置き換え、DFHTEPM  
TYPE=ERRPROC [94](#)  
エラー・プロセッサ・のソース [93](#)  
サンプル  
アクション・フラグ名 [89](#)  
入り口と初期設定 [86](#)  
エラー状況エレメント (ESE) (error status element  
(ESE)) [84](#)  
エラー処理の実行 [86](#)  
エラー・プロセッサの選択 [86](#)  
概要 [86](#)  
起動する [86](#)  
共通のサブルーチン [87](#)  
コンポーネント [84](#)  
サンプル・モジュールの生成 [91](#)  
端末 ID とエラー・コードの検索 [86](#)



## TEP (端末エラー・プログラム) (続き)

### サンプル (続き)

端末エラー・ブロック (TEB) [84](#)

メッセージ [89](#)

DECB オペランド [89](#)

DECB 情報 [89](#)

DFHTEPM TYPE=INITIAL [91](#)

ESE 情報 [90](#)

TACLE 情報 [90](#)

サンプル DFHTEP の生成のためのジョブ制御 [90](#)  
生成 [90](#)

端末異常条件回線項目 (TACLE) [83](#)

端末エラー・ブロックの定義

テーブル、DFHTEPT TYPE=PERMTID [96](#)

通信域

内容のアドレッシング [100](#)

テーブル

デフォルトのしきい値のカウント制限 [98](#)

DFHTEPT TYPE=BUCKET [98](#)

DFHTEPT TYPE=INITIAL [95](#)

DFHTEPT マクロの例 [99](#)

デフォルト・テーブル [85](#)

ユーザー作成プログラム

異常条件 [100](#)

書き込み打ち切りビット [102](#)

ダミー端末標識 [102](#)

通信域の内容のアドレッシング [100](#)

トランザクション異常終了ビット [102](#)

ページ不能タスク [102](#)

例 [105](#)

TACLE DSECT の形式の説明 [103](#)

TACLE の内容のアドレッシング [102](#)

CICS コンポーネント [82](#)

DFHTEP 再帰的再試行ルーチン

システム・カウント (TCTTENI) [105](#)

ユーザー・フィールド A (PCISAVE) [105](#)

ユーザー・フィールド B (PCICNT) [105](#)

例 [106](#)

DFHTEP テーブル [95](#)

DFHTEPM TYPE=ENTRY [93](#)

DFHTEPM TYPE=EXIT [93](#)

DFHTEPT TYPE=PERMCODE|ERRCODE [96](#)

TERMERR 状態 [107](#)

TIE\_UP\_RECORD\_DATA セクション、ジャーナル・レコード  
[289](#)

TIME オペランド

DFHSNET マクロ [128](#)

DFHTEPT TYPE=PERMCODE|ERRCODE マクロ [97](#)

TRMIDNT オペランド

DFHTEPT TYPE=PERMTID [96](#)

True

オープン TCB [16](#)

準再入可能 [16](#)

TSO

DFHCSDUP [330](#)

## U

URIMAP リソース定義

アナライザー・プログラムとの関係 [250](#)

アナライザー・プログラムの置き換え [252](#)

バイパス [252](#)

## W

Web 非対応のアプリケーション・プログラム  
アナライザー・プログラム [250](#)

## X

XICERES、グローバル・ユーザー出口

ターゲット領域でのリソースの使用可否の検査 [234](#)

XPCERES、グローバル・ユーザー出口

ターゲット領域でのリソースの使用可否の検査 [208](#),  
[212](#)

XPI (出口プログラミング・インターフェース)

概要 [48](#)

プログラミング例 [54](#)

API 呼び出しと XPI 呼び出しの混合 [3](#)

RELENSCALL [53](#)

XPI 呼び出しの形式 [49](#)

XPLINK ランタイム・オプション・プログラム、DFHAPXPO  
[250](#)

XSNOFF グローバル・ユーザー出口 [349](#)

XSNON グローバル・ユーザー出口 [349](#)

XSTOUT、グローバル・ユーザー出口 [275](#)

## Z

z/OS Communications Server [120](#)

## [特殊文字]

ブリッジ

要求の動的ルーティング

エラー処理 [211](#)

再呼び出し [212](#)

パラメーターの変更 [210](#)

要求の終了 [211](#)

ルーティングに対し適格 [209](#)





