CICS® ファミリー

クライアント / サーバー・プログラミング

SC88-8998-00
(英文原典：SC34-5947-00)

CICS® ファミリー

クライアント / サーバー・プログラミング

SC88-8998-00
(英文原典：SC34-5947-00)

ご注意

本書の情報およびそれによってサポートされる製品を使用する前に、213ページの『付録D. 特記事項』に記載する一般情報をお読みください。

本書は、SC88-7429 の改訂版です。ページの左側にある縦線は、この版で新規の内容であることを示しています。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

http://www.ibm.com/jp/manuals/main/mail.html

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

http://www.ibm.com/jp/manuals/ の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

原　典： SC34-5947-00
CICS® Family
Client/Server Programming

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2001.6

目次

図

表

改訂の要約

以下は、CICS ユニバーサル・クライアント バージョン 4.0 で行われた機能変更で、本書で適用されるものです。

- 以下のサポートはなくなりました。
  - Cobol
  - PL1
  - REXX
  - CICSTELD
  - ECI_VERSION_0
- 本書で言及するクライアントとは、CICS Transaction Gateway 4.0 の CICS クライアント・コンポーネントのことであり、旧版のような、別個の CICS ユニバーサル・クライアントまたは CICS クライアント製品ではありません。

この版は、SC88-7429-04 の改訂版です。ページの左側にある縦線は、この版で新規の内容であることを示しています。

本書について

本書は、クライアント・サーバー環境で、CICS 以外のアプリケーションから CICS の機能を使用可能にする CICS ファミリーのプログラミング・インターフェースについて説明しています。説明するインターフェースは、以下のとおりです。

- 外部呼び出しインターフェース (ECI)
- 外部表示インターフェース (EPI)
- 外部セキュリティー・インターフェース (ESI)

本書の対象読者

本書は、クライアント・サーバー環境用アプリケーションの設計者およびプログラマーを対象としています。

読者は、以下についての十分な知識が必要です。

- CICS、およびアプリケーションで使用する CICS サーバー
- クライアント・サーバー・プログラミングの概念
- アプリケーション作成に使用するプログラミング言語
- プログラムが作動するプログラミング環境

本書で使用する規約および用語

本書では、用語 *CICS* ユニバーサル・クライアント は、CICS Transaction Gateway のクライアント・コンポーネントを表しています。

- CICS® for MVS/ESA™
- CICS® Transaction Server for OS/390®
- CICS® Transaction Server for VSE/ESA™
- CICS/VSE®

CICS ユニバーサル・クライアントは、Windows NT® および Windows 2000 上で実行します。本書で Windows® は、特定の Windows のバージョンが指定されていない限り、Windows NT および Windows 2000 の両方を指しています。

本書で OS/390 は、OS/390 オペレーティング・システムおよび z/OS オペレーティング・システムの両方を指しています。

前提条件および関連情報

一般情報

- *CICS ファミリー: 相互通信の手引き*、*SC88-7260*

クライアント・サーバー・システムのセットアップ

- *CICS クライアント 管理の手引き*
  - *CICS Transaction Gateway: Windows® クライアント管理*、*SC88-8991*
  - *CICS Transaction Gateway: AIX® クライアント管理*、*SC88-8992*
  - *CICS Transaction Gateway: HP-UX クライアント管理*、*SC88-8994*
  - *CICS Transaction Gateway: Linux クライアント管理*、*SC88-8995*
  - *CICS Transaction Gateway: Solaris クライアント管理*、*SC88-8993*
- *IBM Transaction Server for Windows NT™ V4.0 導入および実行*、GC88-7697
- *IBM Transaction Server for Windows NT Version 4 Administration Guide*、GC33-1881
- *IBM Transaction Server for Windows NT V4 CICS 管理 解説書*、SC88-7764
- *CICS/400® Administration and Operations Guide*、SC33-1387
- *CICS/400 V3.1 相互通信*、SC88-7313
- *CICS Transaction Server for OS/390 CICS リソース定義ガイド*、SC88-7687
- *CICS Transaction Server for OS/390 カスタマイズ・ガイド*、SC88-7686
- *CICS/VSE リソース定義 (オンライン)*、SC88-7128
- *CICS/VSE カストマイズの手引き*、SC88-7127

使用しているクライアントおよびサーバー・システムに関連するその他の資料を参照する必要がある場合があります。

クライアント・アプリケーション・プログラミング

クライアント・システムでのアプリケーション・プログラミングに関する詳細については、該当するクライアント・システムの資料を参照してください。

CICS サーバー上でのアプリケーション・プログラミング

CICS サーバーでのアプリケーション・プログラミングについては、サーバー環境ごとにライブラリーにある資料で説明しています。

その他

- *IBM 3270 情報表示システム 入門*、N:GA27-2739

- *IBM3270 情報表示システム データストリーム プログラマー用解説書*、N:GA23-0059

第1章 外部アクセス・インターフェースの紹介

本章では、CICS ファミリーのクライアント・サーバー・プログラミングの概要を説明します。CICS ファミリーのクライアント・サーバー・プログラミングには以下の 2 つのアプリケーション・プログラミング・インターフェース (API) が含まれ、CICS の機能への外部アクセスを提供します。

- 外部呼び出しインターフェース (ECI)
- 外部表示インターフェース (EPI)
- 外部セキュリティー・インターフェース (ESI)

本章の構成は以下のとおりです。

概要

ECI および EPI は、CICS 以外のアプリケーションから CICS の機能およびデータへのアクセスを可能にするインターフェースです。

2ページの図1 は、クライアント・システム上の CICS 以外のアプリケーションが外部インターフェースを使用している様子を示しています。このアプリケーションからサーバー・システム上の CICS の機能が使用されています。アプリケーションの ECI 要求および EPI 要求は、CICS クライアント・ソフトウェアによって処理され、適切な通信プロトコルを使ってサーバー・システムに転送されます。図のクライアント・システムとサーバー・システムは、それぞれ別のワークステーション上にありますが、図の全構成を 1 台のワークステーション上で実現することもできます。

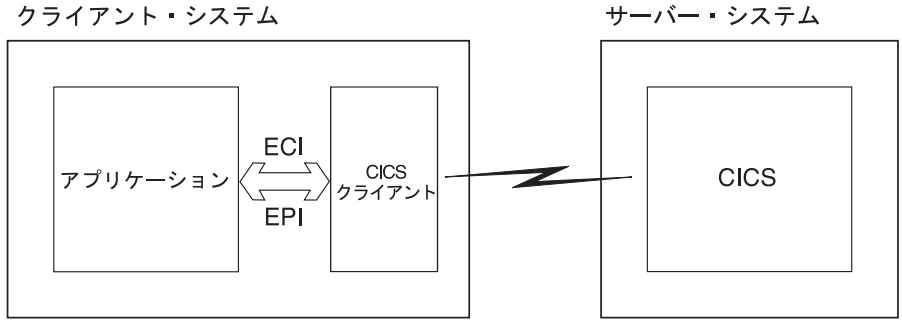

図 1. CICS クライアント・サーバー構成における外部アクセス・インターフェース

CICS ファミリー・プロダクトには、CICS クライアントを使用しなくても、CICS 以外のアプリケーションから外部インターフェースが使用可能なものもあります。この場合、CICS 以外のアプリケーションは、サーバーと同じワークステーション上になければなりません。また、他のサーバーと通信することはできません。図2 は、この様子を示しています。アプリケーションはサーバー上に実現した外部インターフェースを使用しています。

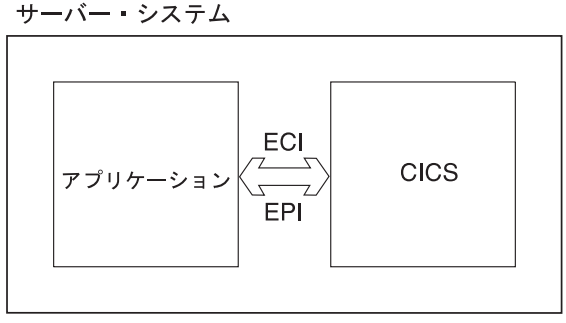

図 2. 外部インターフェースのサーバー上での実現

以下のクライアント・システムを使用すれば、どの CICS サーバーにも接続できます。これらのシステムには、適切な CICS クライアントをインストールする必要があります。「*CICS クライアント 管理の手引き*」を参照してください。

- IBM CICS ユニバーサル・クライアント (Windows 版)
- IBM CICS ユニバーサル・クライアント (AIX 版)®
- IBM CICS ユニバーサル・クライアント (HP-UX 版)
- IBM CICS ユニバーサル・クライアント (Linux 版)
- IBM CICS ユニバーサル・クライアント (Solaris 版)

外部呼び出しインターフェース

ECI は、CICS 以外のアプリケーションから CICS サーバー上の CICS プログラムの呼び出しを可能にするインターフェースです。アプリケーションから複数のサーバーへ同時に接続が可能で、さらに同時に複数の未解決のプログラム呼び出しがサポートされます。

CICS プログラムで端末入出力を行うことはできませんが、その他すべての CICS リソースに対しては、アクセスおよび更新が可能です。

図3 は、同一 CICS プログラムが、 CICS 以外のアプリケーションおよび CICS プログラムの両方から呼び出される様子を示しています。 CICS 以外のアプリケーションは外部呼び出しインターフェース、 CICS プログラムは EXEC CICS LINK をそれぞれ使用しています。 2 つのプログラム間のデータの受け渡しでは、CICS プログラム間通信と同様に COMMAREA が使用されます。

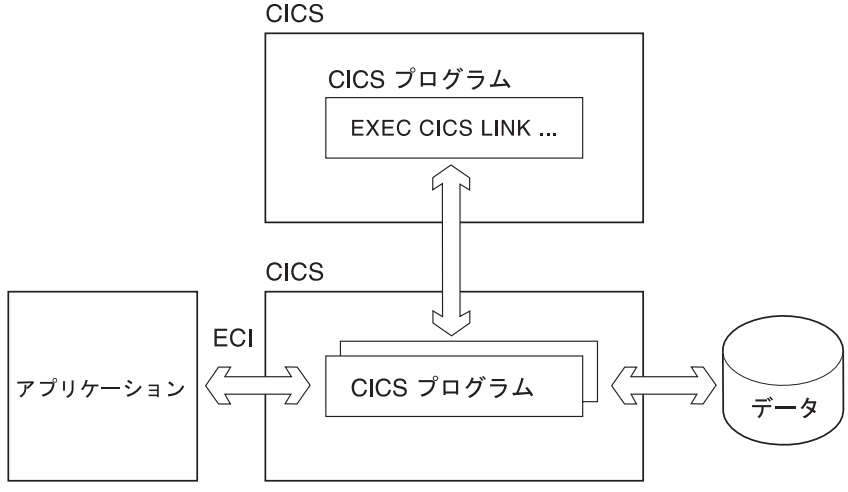

図 3. 外部呼び出しインターフェース

呼び出しには、同期呼び出しと非同期呼び出しがあります。同期呼び出しでは、呼び出されたプログラムが完了すると、呼び出し側に制御が戻り、戻り情報はすぐに使用できます。非同期呼び出しでは、呼び出されたプログラムの完了をチェックすることなく、呼び出し側に制御が戻されます。アプリケーションから戻り情報が使用可能になったかどうかの確認が可能です。

呼び出しは*拡張* することができます。呼び出しの拡張とは、単一の論理作業単位で複数の呼び出しを連続して行うことです。ただしそれぞれの論理作業単位

に対して、一度に 1 つの呼び出ししかアクティブになりません。非同期呼び出しを使えば、アプリケーションから多くの論理作業単位を並行して処理することができます。

呼び出されたプログラムでは、同一システム上のリソースの更新、および分散プログラム・リンク (DPL) による他のシステム上の CICS プログラムの呼び出しが可能です。また、機能シップ、分散トランザクション処理 (DTP) または (CICS/MVS® および CICS/ESA® 環境の場合には) フロントエンド・プログラミング・インターフェース (FEPI) を使用して、他の CICS システム上のリソースにアクセスすることができます。

外部表示インターフェース

EPI は、CICS 以外のアプリケーション・プログラムを、接続先の CICS サーバー・システムに 3270 端末として認識させるインターフェースです。 図4 は、EPI アプリケーションと CICS 端末の両方が CICS サーバーでトランザクションをスケジュールする方法を示しています。

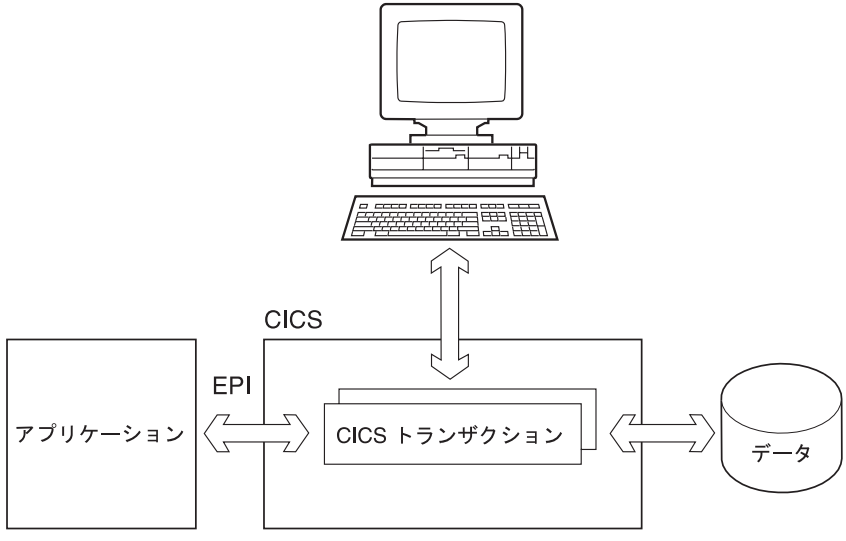

図 4. 外部表示インターフェース

アプリケーションは複数のサーバーの機能を同時に使用し、それらを複数の 3270 端末のように処理することができます。

さらにアプリケーションは、CICS トランザクションをスケジュールし、トランザクションに基本機能を提供します。

EPI を介してのアクセスをサポートしている CICS サーバー上では、そこで稼働している CICS トランザクションで CICS の START コマンドを使えば、CICS 以外のアプリケーションを起動端末とする他のトランザクションをスケジュールすることが可能です。

CICS 以外のアプリケーションが、CICS サーバー上のトランザクションを EPI を使って開始する場合には、3270 データ・ストリームとイベントがサーバーとアプリケーション間で受け渡されます。この場合、端末入出力の内容は、アプリケーションの操作環境に適した方法でユーザー側に表示できます。

トランザクションは、標準的なルーティング方法で、他の CICS システムにルーティングすることができます。他の CICS システム上のリソースへは、機能シップでアクセスできます。

サーバー上のトランザクションとして、3270 データの入出力を行う既存トランザクションを使用することもできます。(この場合には多少制限があります。)

外部セキュリティー・インターフェース

ESI によって、拡張プログラム間通信機能 (APPC) のパスワード有効期限管理 (PEM) が提供するサービスを非 CICS アプリケーションから呼び出すことができます。

CICS では、APPC PEM は、CICS サーバーに対するユーザー ID にサインオンする APPC アーキテクチャー・サインオン・トランザクションのためのサポートを提供し、以下によってパスワード変更の要求を処理します。

- ユーザーを識別して、そのユーザーの ID を認証する。
- 認証時に、特定のユーザーにパスワードの有効期限が切れたことを通知する。
- パスワードの有効期限が切れたとき (または切れる前) に、ユーザーにパスワードを変更させる。
- 現在のパスワードが有効とされる期間をユーザーに通知する。
- 特定のユーザー ID を使用したサーバーへの認証されていないアクセス試行に関する情報を提供する。

ESI 呼び出しを ECI または EPI アプリケーションに組み込むことができます。

外部アクセス・インターフェースの使用方法

外部アクセス・インターフェースは、CICS 以外のアプリケーションから CICS トランザクションの初期化または CICS プログラムの呼び出しを行って、CICS リソースへのアクセスおよびリソースの更新を可能にします。さらに外部インターフェースと CICS 通信機能を併用すると、CICS 以外のアプリケーションから、あらゆる CICS システム上のリソースへのアクセスおよび更新が可能になります。この機能によって、以下のような処理が可能になります。

1. プレゼンテーション・マネージャーや他の表示システムを使用して、グラフィカル・ユーザー・インターフェース (GUI) フロントエンドを CICS アプリケーション用に開発する。
2. バー・コード読み取り装置などの外部機器と CICS システムとの接続
3. CICS システムと CICS 以外のシステムの統合

EPI を使用することによって、既存 CICS システム用および新規アプリケーション用の GUI 機能作成が可能になります。特に、変更を行わない既存 CICS トランザクション用の新規 GUI フロントエンド開発に有用です。 EPI を使用するアプリケーションでは、CICS トランザクションとの通信、およびクライアント・システムの表示機能によるエンド・ユーザーとの対話が可能です。

バー・コード読み取り装置などの外部機器を接続した場合には、アプリケーションから EPI を使って、外部機器の入出力の処理、および外部装置の生成データの処理用の作成済みトランザクションの開始などを行うことができます。外部機器からの入力は、アプリケーションによって 3270 データ・ストリームに変換され、開始したトランザクションに渡されます。トランザクションから出力される 3270 データ・ストリームは、信号とデータ・ストリームに変換されて外部機器の操作に使用されます。

CICS システムと CICS 以外のシステムの統合には、通常 CICS プログラムと CICS 以外のシステムのプログラム間でのユーザー定義データの受け渡しが含まれます。このために ECI を使用できます。

上記のいずれの場合でも、EPI と ECI のいずれを選択するかについての明確な基準はありません。どちらのインターフェースでも、CICS アプリケーションと CICS 以外のアプリケーション間のデータ受け渡しはサポートされます。ただし、データ受け渡しの方法は、EPI では 3270 データ・ストリーム、ECI では COMMAREA を使ったアプリケーションになります。

ECI および EPI 出口

CICS ユニバーサル・クライアントの場合、EPI および ECI の操作は、ユーザー出口でカスタマイズすることができます。 169ページの『付録C. ECI および EPI 出口』を参照してください。

第2章 外部呼び出しインターフェース

本章では、外部呼び出しインターフェース (ECI) についての解説をします。

インターフェースのエレメントに選んだ名前、つまり、関数、パラメーター、データ構造、フィールド、および定数などは、プログラミングで提供される名前と似ていますが、ここで解説するインターフェースの適用範囲は、特定の言語に限定されません。特定言語に関連した情報については、137ページの『第4章 ECI および EPI アプリケーション・プログラムの作成』を参照してください。

また、プラットフォームによっては、 CICS ファミリーのクライアント・サーバー・プログラミングに含まれない ECI 機能が使用できる場合もあります。このような機能については、159ページの『付録A. 環境に依存する ECI の拡張関数』を参照してください。

本章の構成は以下のとおりです。

概要

ECI 関数

ECI は、CICS 以外のアプリケーションから CICS サーバー上の CICS プログラムを呼び出し可能にするインターフェースです。呼び出し側の CICS 以外のアプリケーションは CICS コマンドは出しません。サーバー上で稼働する呼び出されたプログラムから出されます。したがって、呼び出されたプログラムは、COMMAREA オプションを使用した EXEC CICS LINK による呼び出しのような形をとります。 ECI アプリケーションは、分散プログラム・リンクの規則に従う既存の CICS プログラムを利用することができ、ECI アプリケー

ションから呼び出される新規 CICS プログラムを書くこともできます。ただし呼び出し先プログラムは、DPL から呼び出される CICS プログラムと同じ規則に従います。

さらに使用環境によっては、アプリケーションから、異なる CICS サーバー上に分散している複数の CICS プログラムを並行して稼働することもできます。

ECI から提供される関数は、以下の 2 つに分けてパッケージされています。

CICS_ExternalCall
ECI のほとんどの関数を提供します。パラメーターは ECI パラメーター・ブロックのみで、その中のさまざまなフィールドで実行する関数および入出力を表します。以下のセクションで説明する関数は、大部分がこの **CICS_ExternalCall** の使用に関連するものです。

CICS_EciListSystems
CICS_ExternalCall 要求が向けられる使用可能なサーバーを検索するのに使われます。詳細については、58ページの『CICS_EciListSystems』を参照してください。

ECI 呼び出しのタイプ

CICS_ExternalCall には、以下の 3 つのタイプの呼び出しがあります。
- プログラム・リンク呼び出し
- 状況情報呼び出し
- 応答請求呼び出し

CICS_ExternalCall の呼び出しのタイプは、ECI パラメーター・ブロックの **eci_call_type** パラメーターの設定によって制御します。

特定の操作環境での呼び出しタイプ制約については、各呼び出しタイプの説明を参照してください。

プログラム・リンク呼び出し

プログラム・リンク呼び出しは、CICS プログラムを CICS サーバー上で実行します。この呼び出しは、以下のいずれかです。

- 同期呼び出し -- 呼び出し側のアプリケーションは、呼び出されたプログラムが完了するまで待ちます。戻り情報は、すぐに使用できます。
- 非同期呼び出し -- 呼び出し側のアプリケーションは、呼び出されたプログラムが完了したかどうかに関係なく、制御を戻されます。呼び出し側のアプリケーションは、後に情報が使用可能になったときに通知するよう要求することができます。非同期呼び出しの結果を確認する場合には、応答請求呼び出しを使用します。

状況情報呼び出し

状況情報呼び出しは、アプリケーションが稼働しているシステムの種類、およびその状況情報を検索します。この呼び出しは、以下のいずれかです。

- 同期呼び出し -- 呼び出し側のアプリケーションは、状況情報が使用可能になるまで待ちます。
- 非同期呼び出し -- 呼び出し側のアプリケーションは、状況情報の検索されている間に、制御を戻されます。呼び出し側のアプリケーションは、後に情報が使用可能になったときに通知するよう要求することができます。非同期呼び出しの結果を確認する場合には、応答請求呼び出しを使用します。

応答請求呼び出し

応答請求呼び出しは、非同期プログラム・リンクまたは非同期状況情報呼び出しの後に、情報を戻します。応答請求呼び出しには、以下の処理があります。

- 汎用呼び出し -- 未解決の情報すべてを検索します。
- 特殊呼び出し -- 指定した非同期要求の情報を検索します。

非同期呼び出しを行うアプリケーションはいつでも、複数の未解決のプログラム・リンク呼び出しおよび状況情報呼び出しを、行うことができます。ECI パラメーター・ブロック内の **eci_message_qualifier** パラメーターを各非同期呼び出しで使用すれば、その呼び出しにユーザー定義の ID を割り当てることができます。1 つのアプリケーション内の異なる非同期呼び出しで別々の ID を使うようにするのは、プログラマーの責任です。汎用の応答請求呼び出しを行うときには、ECI は **eci_message_qualifier** フィールドを使用して、応答が属する呼び出しの名前が戻します。特定の応答請求呼び出しでは、呼び出し時に **eci_message_qualifier** フィールドに値を指定し、検索中の情報がある非同期呼び出しを識別する必要があります。

プログラム・リンク呼び出し

プログラム・リンク呼び出しには、同期呼び出しと非同期呼び出しの 2 種類があります。非同期呼び出しで応答を要求するには、呼び出し側のアプリケーションから応答請求呼び出しを使う必要があります。(17ページの『応答請求呼び出し』を参照してください。)

作業論理単位の管理

ECI アプリケーションは、多くの場合、サーバー上のリカバリー可能リソースの更新を必要としています。したがって、プログラマーは、論理作業単位を管理するために ECI から提供される機能について理解しておく必要があります。作業論理単位とは、リカバリー可能リソースに対して一連の変更を行うために必要なサーバー上のすべての処理のことです。1 つの作業論理単位が正常

に終了すると、行われた変更がすべてコミットされます。作業論理単位が異常終了した場合、たとえばプログラムの異常終了時などには、変更はバックアウトされます。サーバー上の作業論理単位は、ECI を使用して開始および終了することができます。

作業論理単位でのリカバリー可能リソースへの変更は、以下のいずれかの呼び出しによって行われます。
- 単一のプログラム・リンク呼び出し
- 一連のプログラム・リンク呼び出し

作業論理単位に対する一連の呼び出しの最初の呼び出しが正常終了すると、制御ブロック内の **eci_luw_token** フィールドにトークンが格納されます。このトークンは、その同じ作業論理単位に関連する以降の呼び出しで使用します。同じ作業論理単位を対象とした一連のプログラム・リンク呼び出しは、すべて同一のサーバーに送信されます。

重要: 1 つの作業論理単位に対する一連のプログラム・リンク呼び出しで作業論理単位の拡張のために長い時間を要する場合 (たとえば、ユーザーの考慮時間が長い場合) には注意が必要です。作業論理単位は、いろいろのロックや、サーバー上の多くの CICS リソースを保留状態にするため、同一ロックおよびリソースを待っているユーザーの待ち時間が長くなってしまうからです。

作業論理単位が終了すると、CICS サーバーは、行われた変更をコミットしようとします。したがって、作業論理単位に対する一連のプログラム・リンク呼び出しの最後の呼び出し、または作業論理単位の単独の呼び出しで、このコミットが行われたかどうかの確認処理を行ってください。

作業論理単位当り 1 つのプログラム・リンク呼び出しだけが常に未解決状態です。(非同期プログラム・リンク呼び出しの処理は、応答請求呼び出しによって応答処理が行われるまで未解決です。)

13ページの表1 は、**eci_extend_mode**、 **eci_program_name**、および **eci_luw_token** パラメーター値の組み合わせを使用して、 ECI を介した作業論理単位の管理に関連するタスクを実行する方法を示しています。いずれの場合でも、使用する呼び出しのタイプに応じて、適切な値を別のフィールドに保管する必要があります。

表 1. ECI における作業論理単位

行うタスク	使用パラメーター
対象作業論理単位に対する単独プログラムの呼び出し。 1 つの要求がクライアントからサーバーへ送信される。指定されたプログラムによるすべての変更がコミットされた後にはじめて応答がクライアントに送信される。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_NO_EXTEND • **eci_program_name**: 適切な値を指定 • **eci_luw_token**: 0
新たに拡張作業論理単位を開始するプログラムの呼び出し。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_EXTENDED • **eci_program_name**: 適切な値を指定 • **eci_luw_token**: 0 このあと **eci_luw_token** から戻されるトークンを保管します。
既存の作業論理単位を継続するプログラムの呼び出し。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_EXTENDED • **eci_program_name**: 適切な値を指定 • **eci_luw_token**: 保管したトークンを指定
既存の作業論理単位に対する最後のプログラムの呼び出し、および変更のコミット。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_NO_EXTEND • **eci_program_name**: 適切な値を指定 • **eci_luw_token**: 保管したトークンを指定
既存作業論理単位の終了 (他のプログラムは呼び出さない)、およびリカバリー可能リソースへの変更のコミット。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_COMMIT • **eci_program_name**: ヌル • **eci_luw_token**: 保管したトークンを指定
既存の作業論理単位の終了 (他のプログラムは呼び出さない)、およびリカバリー可能リソースへの変更のバックアウト。	以下のようにパラメーターを指定します。 • **eci_extend_mode**: ECI_BACKOUT • **eci_program_name**: ヌル • **eci_luw_token**: 保管したトークンを指定

拡張作業論理単位の呼び出しの 1 つでエラーが生じた場合、 **eci_luw_token** フィールドを使用すると、行った変更がバックアウトされているのか、まだ保留状態なのかを調べることができます。**eci_luw_token** フィールドの詳細については、22ページの『ECI_SYNC 呼び出しタイプ』および 30ページの『ECI_ASYNC 呼び出しタイプ』を参照してください。行った変更が保留状態

になっている場合には、変更をコミットまたはバックアウトするプログラム・リンク呼び出しを実行して、その作業論理単位を終了させなければなりません。

各作業論理単位の実行時には、 CICS 機能以外のタスクが実行されます。したがって、並行呼び出しの期待する最大数を得るためには、CICS サーバー上で十分な数のフリー・タスクを指定してください。

図5 は、アプリケーション・プログラムによる非同期プログラム呼び出しを示しています。このプログラムからの 2 つの呼び出しが、それぞれ異なる 2 つのサーバー上で未解決になっています。このプログラムは、要求を記録するためのメッセージ修飾子を生成し、さらに **eci_luw_token** フィールドに戻された LUW トークンを使用しています。

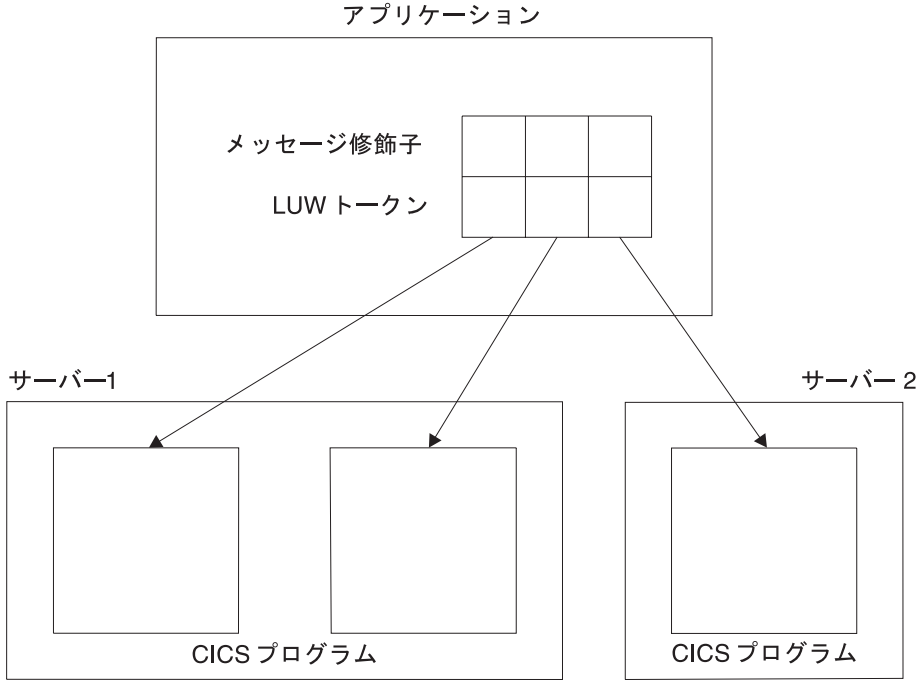

図 5. メッセージ修飾子および LUW トークンを使用した非同期プログラム・リンク

プログラム・リンク呼び出しの詳細については、以下の各タイプの呼び出しの説明を参照してください。

22ページの『ECI_SYNC 呼び出しタイプ』(同期プログラム呼び出し)

30ページの『ECI_ASYNC 呼び出しタイプ』(非同期プログラム呼び出し)

ECI でのセキュリティー

ECI_SYNC および ECI_ASYNC 呼び出しは、CICS クライアントおよびサーバーの間の一連システム間会話としてインプリメントされます。つまり、クライアント・アプリケーションが作成する各作業単位に対して、 1 つの会話が割り振られます。

各システム間会話ごとにユーザー ID とパスワードが必要になることがあります。この要件は、CICS クライアントと CICS サーバーの構成の仕方によって決まります。

クライアント・アプリケーションは、作業単位内の最初の呼び出し、または単独の呼び出しに対して、ECI パラメーター・ブロック内の **eci_userid** (または **eci_userid2**) および **eci_password** (または **eci_password2**) でユーザー ID およびパスワードを指定できます。

クライアント・アプリケーションは、ユーザー ID とパスワードを **CICS_SetDefaultSecurity** 関数に渡すことができます。これらの値はサーバー固有で、ECI パラメーター・ブロックでこれらが省略されている場合に必ず使用されます。これらの値は、 **CICS_SetDefaultSecurity** 関数によっていつでも変更することができます。

ユーザー ID とパスワードが、システム間会話で必要とされ、**CICS_SetDefaultSecurity** 関数で設定されていない場合は、CICS クライアントは他の方法 (たとえば、ポップアップ・ウィンドウ) を使用してこれらの値を決定することができます。 (ポップアップ・ウィンドウは、UNIX® オペレーティング・システムではサポートされませんのでご注意ください)。

状況情報呼び出し

状況情報呼び出しは、同期呼び出し、非同期呼び出しのいずれかです。非同期呼び出しで応答を得るためには、呼び出し側のアプリケーションから応答請求呼び出しを行う必要があります。 (17ページの『応答請求呼び出し』を参照してください。)

状況情報の提供および使用方法

呼び出しの実行後、状況情報は ECI 状況ブロック内の **eci_commarea** パラメーターに格納され、インターフェース経由で受け渡されます。

ECI 状況ブロック内に格納される状況情報を以下に示します。詳細については、56ページの『ECI 状況ブロック』を参照してください。

- 接続の種類 (ECI プログラムの CICS サーバーへのローカル接続、CICS クライアントへのローカル接続、または接続なし)
- CICS サーバーの状態 (使用可能、使用不可、不明)
- CICS クライアントの状態 (使用可能、使用不可、不明)

状況情報呼び出しでは、以下の 3 種類のタスクを行うことがきます。

- *アプリケーションが稼働しているシステムの種類、および指定したサーバーとの接続状況の照会*。これを行うために、状況情報の格納先として COMMAREA を指定してください。
- *状況が指定した値以外に変化した場合の通知要求の設定*。これを行うために、指定状況を入れる COMMAREA を設定します。状況が指定した値以外に変化すると、変化後の状況が通知されます。これに使うのは、非同期呼び出しのみです。
- *状況変更の通知要求の取り消し*。COMMAREA は必要ありません。

表2 は、**eci_extend_mode**、 **eci_commarea**、**eci_commarea_length**、および **eci_luw_token** パラメーター値の組み合わせを使用して、状況照会に関連したタスクを実行する方法を示しています。いずれの場合でも、使用する呼び出しのタイプに応じて、適切な値を別のフィールドに保管する必要があります。

表 2. CICS_ExternalCall による状況照会

行うタスク	使用パラメーター
現在の状況の照会	以下のようにパラメーターを指定します。 • **eci_commarea**: ヌル • **eci_commarea_length**: ECI_STATUS の長さ • **eci_extend_mode**: ECI_STATE_IMMEDIATE このあと、現在の状況を COMMAREA 内で調べます。
状況変更の通知要求の設定	以下のようにパラメーターを指定します。 • **eci_commarea**: 指定状況 • **eci_commarea_length**: ECI_STATUS の長さ • **eci_extend_mode**: ECI_STATE_CHANGED • **eci_luw_token**: 0 このあと、**eci_luw_token** に戻されるトークンを保管します。保管したトークンは要求処理の取り消し時に使用します。

表 2. CICS_ExternalCall による状況照会 (続き)

行うタスク	使用パラメーター
状況変更の通知取り消し	以下のようにパラメーターを指定します。 • **eci_commarea**: ヌル • **eci_commarea_length**: 0 • **eci_extend_mode**: ECI_STATE_CANCEL • **eci_luw_token**: 保管したトークンを指定

状況情報呼び出しの詳細については、以下の各呼び出しタイプの説明を参照してください。

38ページの『ECI_STATE_SYNC 呼び出しタイプ』(同期状況呼び出し)

42ページの『ECI_STATE_ASYNC 呼び出しタイプ』(非同期状況呼び出し)

応答請求呼び出し

非同期プログラム・リンク呼び出しまたは非同期状況情報呼び出しの後に応答を得るためには、呼び出し側のアプリケーションから応答請求呼び出しを行う必要があります。応答請求呼び出しでは、呼び出しで指定した選択基準を満たす未解決の応答すべてが戻されます。

応答請求呼び出しの詳細については、以下の呼び出しの各タイプの説明を参照してください。

47ページの『ECI_GET_REPLY 呼び出しタイプ』(応答が使用可能の場合、いずれの非同期呼び出しにも未解決の応答すべての獲得)

51ページの『ECI_GET_REPLY_WAIT 呼び出しタイプ』(応答が使用可能になるのを待っているいずれの非同期呼び出しにも未解決の応答すべての獲得)

52ページの『ECI_GET_SPECIFIC_REPLY 呼び出しタイプ』(応答が使用可能の場合、指定した非同期呼び出しの未解決の応答すべての獲得)

56ページの『ECI_GET_SPECIFIC_REPLY_WAIT 呼び出しタイプ』(応答が使用可能になるのを待っている指定非同期呼び出しの未解決の応答すべての獲得)

CICS_ExternalCall

CICS_ExternalCall	ECI_Parms

目的

CICS_ExternalCall は上記のプログラム・リンク呼び出し、状況情報呼び出し、および応答請求呼び出しへのアクセスを可能にします。実行する関数は、ECI パラメーター・ブロック内の **eci_call_type** フィールドで指定します。

パラメーター

ECI_Parms

ECI パラメーター・ブロックを指すポインターです。*使用する前に、ECI パラメーター・ブロックの各フィールドにヌルを指定してください。*入出力用のパラメーター・ブロックの各フィールドの詳細については、以降のセクションの呼び出しの各タイプの説明を参照してください。各フィールドの概要は以下のとおりです。

eci_call_type

使用する呼び出しのタイプを指定する整数フィールドです。提供される関数に関する詳細については、 10ページの『ECI 呼び出しのタイプ』を参照してください。

eci_program_name

呼び出し対象のプログラム名です。

eci_userid

セキュリティー・チェック用のユーザー ID です。

eci_password

セキュリティー・チェック用のパスワードです。

eci_transid

トランザクション ID

eci_abend_code

プログラム異常発生時に異常終了コードが格納されます。

eci_commarea

COMMAREA 格納域です。呼び出されたプログラムで使用されるか、または戻された状況情報が格納されます。

eci_commarea_length

COMMAREA の長さが入ります。 COMMAREA のサイズは、入力データまたは出力データの最大サイズに設定しなければな

りません。最大長は、32 500 バイトです。入力データが COMMAREA の長さよりも小さい場合は、COMMAREA をヌルで埋め込まなければなりません。 CICS クライアントは、このヌル埋め込みを除去し、ECI 要求のデータのみを CICS サーバーに送信します。

eci_timeout
CICS サーバーからの応答を待つ時間。 ECI タイムアウト・サポートの詳細については、161ページの『タイムアウト機能』を参照してください。

reserved1
予期せぬエラーの発生時にその詳細情報を与える戻りコード。

このフィールドは、以前は **eci_system_ return_code** でした。CICS ユニバーサル・クライアントのバージョン 3.1 およびそれ以降のバージョンでは、このフィールドは後方互換性のために予約されています。このフィールドに戻される情報はありません。すべてのシステム・エラーは、クライアントのエラー・ログに書き込まれます。

eci_extend_mode
さまざまな方法で実行される関数を修飾するためのフィールドです。

eci_message_qualifier
非同期呼び出しで参照するユーザー提供の ID です。

eci_luw_token
作業論理単位の ID です。

eci_sysid
このバージョンでは使用されません。ヌルを指定してください。

eci_version
アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

eci_system_name
CICS サーバー名を格納します。

eci_callback
非同期要求のコールバック・ルーチンを指すポインターです。

eci_userid2
セキュリティー・チェック用のユーザー ID です。

eci_password2
セキュリティー・チェック用のパスワードです。

eci_tpn
ミラー・トランザクションのトランザクション識別名。このフィールドは、**eci_version** の値が ECI_VERSION_1A の場合にのみ使用することができます。

戻りコード

以降のセクションで説明する各タイプの呼び出しの戻りコード以外に、以下のコードが戻される場合があります。

ECI_ERR_INVALID_CALL_TYPE
呼び出しタイプが無効です。

ECI_ERR_CALL_FROM_CALLBACK
コールバック・ルーチンから呼び出しが行われました。

ECI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。クライアントまたはサーバーにエラーが発生している可能性があります。プログラマーは、クライアントのエラー・ログに戻された情報を保管する必要があります。この情報は、サービス担当員がエラーを診断するときに役立ちます。

ECI_ERR_INVALID_VERSION
eci_version の値が無効です。

ECI_ERR_REQUEST_TIMEOUT
ECI パラメーター・ブロックの **eci_timeout** フィールドの値が 0 以下です。

使用環境によっては、上記および各タイプの呼び出しで説明される戻りコードがサポートされていない場合もあります。

Windows® オペレーティング・システムの場合、実際の戻りコード値を記号名にマッピングしたものが、以下のファイルに含まれています。

C ¥include¥cics_eci.h

UNIX® オペレーティング・システムの場合は、以下のファイルです。

C /include/cics_eci.h

ECI_SYNC 呼び出しタイプ

環境

ECI_SYNC 呼び出しは、あらゆる環境で実行できます。

目的

ECI_SYNC 呼び出しは同期プログラム・リンクを実行し、作業論理単位が開始、継続、または終了されます。呼び出し側のアプリケーションには、呼び出された CICS プログラムが終了するまで制御が戻りません。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックの各フィールドをヌルに設定する必要があります。

eci_call_type

必須の入力パラメーターです。

ECI_SYNC に設定します。

eci_program_name

入力パラメーターです。**eci_extend_mode** が ECI_COMMIT または ECI_BACKOUT 以外の場合は必須です。 (13ページの表1 を参照してください。)

呼び出し対象のプログラム名を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。このフィールドの値は、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

eci_userid

必須の入力パラメーターです。

ユーザー ID を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。

クライアントおよびサーバーの関連資料を参照し、このフィールドの値がサーバー伝送時に大文字に変換されるかどうかを確認してください。(8 文字以上のユーザー ID またはパスワードが必要な場合には、**eci_userid** および **eci_password** をヌル・ストリングに指定し、**eci_userid2** and **eci_password2** フィールドを代わりに指定します。)

ユーザー ID を指定すると、サーバーはユーザー ID および (指定されている場合には) パスワードを使用してユーザーの検証を行います。これらのユーザー ID およびパスワードは、以降のセキュリティー・チェックの際にも使用されます。

eci_password

必須の入力パラメーターです。

パスワードを格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。

クライアントおよびサーバーの関連資料を参照し、このフィールドの値がサーバー伝送時に大文字に変換されるかどうかを確認してください。(8 文字以上のユーザー ID またはパスワードが必要な場合には、このフィールドおよび **eci_userid** をヌル・ストリングに指定し、**eci_userid2** および **eci_password2** フィールドを代わりに使用します。)

eci_transid

任意指定の入力パラメーターです。

CICS トランザクションの ID を格納する 4 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。 **eci_tpn** が使用されている (ヌル・ストリング以外の値に設定されている) 場合には、このパラメーターは無視されます。このパラメーターの使用方法は、要求を送るクライアントによって異なります。**eci_transid** の値は、**eci_program_name** で指定されているプログラムへの LINK 時に、大文字変換なしで ASCII から EBCDIC に変換され、EIBTRNID に保管されます。

呼び出されたプログラムはミラー・トランザクション CPMI のもとで実行されますが、**eci_transid** トランザクション名のもとでリンクされます。この名前は、呼び出されたプログラムがトランザクション ID を照会するときに使用されます。サーバーによっては、トランザクション ID を使用して、呼び出されたプログラムのセキュリティー属性およびパフォーマンス属性を判別する場合があります。これらのサーバーでは、呼び出し先プログラムの処理を制御する場合に、このパラメーターを使用することをお勧めします。

ECI 要求を拡張する (**eci_extend_mode** の説明を参照) 場合は、**eci_transid** パラメーターは作業単位の最初の呼び出しに対してだけ意味をもちます。

このフィールドの値がすべてヌル・ストリングで、**eci_tpn** が指定されていない場合、デフォルトのサーバー・トランザクション ID が使用されます。

eci_abend_code
出力パラメーターです。

呼び出されたプログラムを実行するトランザクションが異常終了した際に、CICS 異常終了コードが戻される 4 文字のフィールドです。異常終了コードが 4 文字未満の場合には、残りにはスペースが格納されます。

eci_commarea
任意指定の入力パラメーターです。

呼び出された CICS プログラムに渡される COMMAREA データを指すポインターです。COMMAREA は、呼び出されたプログラムからアプリケーションに戻される情報が格納されます。

COMMAREA が必要ない場合はヌル・ポインターを指定し、長さ (**eci_commarea_length** で指定されます) を 0 にします。

アプリケーションとサーバーのコード・ページが一致しない場合には、サーバーで、データ伝送時にデータ変換を行う必要があります。このデータ変換では、DFHCNV マクロ定義などの CICS 提供のリソース変換機能を使用してください。

eci_commarea_length
任意指定の入力パラメーターです。

COMMAREA のバイト長です。最大長は、32 500 バイトです。 (クライアント / サーバー環境によっては、これ以上の長さの COMMAREA を指定することが可能な場合もありますが、「CICS ファミリー: クライアント / サーバー・プログラミング」で正常に使用できることは保証されません。)

COMMAREA が必要ない場合には、このフィールドには 0 を指定し、**eci_commarea** フィールドにヌル・ポインターを指定します。

eci_timeout
CICS サーバーからの応答を待つ時間 (秒単位)。 0 の値は、限度が設定されていないことを示します。

タイムアウトが発生すると、会話は異常終了します。

reserved1

出力パラメーターです。

このフィールドは、以前は **eci_system_ return_code** でした。CICS ユニバーサル・クライアントのバージョン 3.1 およびそれ以降のバージョンでは、このフィールドは後方互換性のために予約されています。このフィールドに戻される情報はありません。すべてのシステム・エラーは、クライアントのエラー・ログに書き込まれます。

eci_extend_mode

必須の入力パラメーターです。

作業論理単位が、呼び出しの後に終了するかどうかを決める整数フィールドです。(13ページの表1 を参照してください。)

このフィールドに格納される値 (記号名) を以下に示します。

ECI_NO_EXTEND

1. **eci_luw_token** フィールドの入力値が 0 の場合は、作業論理単位に対する単独呼び出しを示します。
2. **eci_luw_token** の入力値が 0 以外の場合は、指定作業論理単位に対する最後の呼び出しを示します。

いずれの場合でも、リカバリー可能リソースへの変更は、CICS タスク終了同期点でコミットされ、作業論理単位が終了します。

eci_extend_mode を ECI_NO_EXTEND に設定し、eci_luw_token を 0 に設定した場合には、クライアントからサーバーへの要求フローが 1 つと、サーバーからクライアントへの応答フローが 1 つ発生します。サーバーは、eci_program_name で指定されたプログラムが起動され、そのプログラムによって行われた変更がコミットされると、応答を送信します。

ECI_EXTENDED

1. **eci_luw_token** フィールドの入力値が 0 の場合は、継続する拡張作業論理単位に対する最初の呼び出しを示します。
2. **eci_luw_token** フィールドの入力値が 0 以外の場合は、作業論理単位の継続する呼び出しを示します。

いずれの場合でも、呼び出されたプログラムの正常終了後、作業論理単位の処理は継続され、リカバリー可能リソースへの変更はこの段階ではコミットされません。

ECI_COMMIT
eci_luw_token フィールドの入力値で識別される作業論理単位を終了し、リカバリー可能リソースへの変更すべてをコミットします。

ECI_BACKOUT
eci_luw_token フィールドの入力値で識別される作業論理単位を終了し、リカバリー可能リソースへの変更をバックアウトします。

eci_luw_token
必須の入出力パラメーターです。

呼び出しの対象となる作業論理単位を識別する整数フィールドです。このパラメーターの値は、作業論理単位が拡張されるかどうかに関係なく、作業論理単位の開始時に必ず 0 に指定します。拡張作業論理単位の場合には、最初の呼び出し時に ECI が **eci_luw_token** フィールドに有効な値を格納します。この値は、この作業論理単位に対する継続呼び出しの入力値として指定してください。(13ページの表1 を参照してください。)

戻りコードが ECI_NO_ERROR 以外で、呼び出しが作業論理単位を継続または終了する場合、このフィールドは作業論理単位の状態をレポートする出力として使用されます。この出力値が 0 の場合は、作業論理単位が終了し、更新がバックアウトされたことを示します。0 以外の場合は、入力値と同じ値であり、作業論理単位が継続され、更新が保留中であることを示します。

eci_sysid
必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version
必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。
ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能に

よっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

eci_system_name
任意指定の入力パラメーターです。

ECI 要求の対象サーバーの名前を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。このパラメーターを指定する場合は、**CICS_EciListSystems** 関数が戻すサーバー名のいずれかを指定します。この値は、**eci_luw_token** が 0 の場合には必ず指定します。(**eci_luw_token** が 0 以外の場合は、対象作業論理単位の開始時にサーバーが確立するため、このパラメーターを指定しても無視されます。)

このフィールドの値がヌルの場合は、現行のデフォルト・サーバーが選択され、このサーバーの名前がこのフィールドに格納されます。この名前は、以降に実行する一連の ECI 要求で使用します。他の作業論理単位に対する ECI 要求でも、対象のサーバーが同じ場合は、このサーバーの名前を **eci_system_name** に指定します。

eci_userid2
任意指定の入力パラメーターです。

eci_userid フィールドにヌルを指定すると、**eci_userid2** フィールドの値でユーザー ID (存在する場合) が識別されます。このユーザー ID は、サーバー側の各種権限の検証で使用されます。ユーザー ID の最大長は 16 文字です。.

ユーザー ID の使用法については、**eci_userid** フィールドの説明を参照してください。

eci_password2
任意指定の入力パラメーターです。

eci_password フィールドがヌルの場合は、 **eci_password2** フィールドの値でパスワード (存在する場合) が指定されます。パスワードは、サーバー側の各種権限の検証で使用されます。パスワードの最大長は 16 文字です。

パスワードの使用法については、**eci_password** フィールドの説明を参照してください。

eci_tpn
任意指定の入力パラメーターです。このパラメーターは、**eci_version** の値が ECI_VERSION_1A の場合にだけ使用することができます。

サーバーが ECI 要求の処理に使用するトランザクションの ID が格納される 4 文字のフィールドです。このトランザクションは、CICS ミラー・トランザクションとしてサーバーで定義されていなければなりません。このフィールドがヌル・ストリングの場合は、デフォルト・ミラー・トランザクション CPMI が使用されます。

ECI 要求が拡張された場合 (**eci_extend_mode** の説明を参照)、このパラメーターは最初の要求に対してだけ意味をもちます。

このフィールドを使用すると、**eci_transid** の内容は無視されます。

戻りコード

18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

ECI_NO_ERROR
呼び出しが正常に終了しました。

ECI_ERR_INVALID_DATA_LENGTH
eci_commarea_length フィールドに指定した値が無効です (**eci_commarea** フィールドの値と整合性がありません)。ヌル・ポインターではない **eci_commarea** に 0 を指定している、あるいはヌル・ポインターである **eci_commarea** に 0 以外の値を指定しています。

ECI_ERR_INVALID_EXTEND_MODE
eci_extend_mode フィールドの値が無効です。

ECI_ERR_NO_CICS
クライアントまたはサーバーが使用できない、または作業論理単位の開始時に **eci_system_name** で指定される CICS サーバーが使用できない、のいずれかの場合を示します。リソースの更新は行われていません。

ECI_ERR_CICS_DIED
作業論理単位の開始または継続時に、対象 CICS サーバーが使用できませんでした。 **eci_extend_mode** の値が ECI_EXTENDED の場合は、変更はバックアウトされ、作業論理単位が終了します。
eci_extend_mode の値が ECI_NO_EXTEND、ECI_COMMIT、または ECI_BACKOUT である場合には、アプリケーションは変更がコミットされたかバックアウトされたかを判別できません。したがって、今後手動リカバリーを行わなければならない機会があったときのため、この状態を記録しておく必要があります。

ECI_ERR_TRANSACTION_ABEND
要求したプログラムを実行した CICS トランザクションが異常終了しました。異常終了コードが、**eci_abend_code** に格納されます。異常終了コードの種類と意味については、要求対象のサーバー・システムの関連資料を参照してください。

ECI_ERR_LUW_TOKEN
eci_luw_token の値が無効です。

ECI_ERR_ALREADY_ACTIVE
作業論理単位の継続要求時に、その作業論理単位に未完了の非同期呼び出しが存在します。

ECI_ERR_RESOURCE_SHORTAGE
サーバーまたはクライアント上に、要求の完了に必要なリソースが不足しています。

ECI_ERR_NO_SESSIONS
最大数の作業論理単位を実行しているアプリケーションから、さらに新規作業論理単位の作成が要求されました。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_ERR_ROLLEDBACK
サーバーによる作業論理単位のコミット処理で、変更がコミットできずバックアウトされました。

ECI_ERR_UNKNOWN_SERVER
要求対象のサーバーが見つかりません。**CICS_EciListSystems** で戻されたサーバーのみが受け入れ可能です。

ECI_ERR_MAX_SESSIONS
要求の実行に必要な通信リソースが不足しています。通信リソースの制御方法については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_MAX_SYSTEMS
構成の上限以上の数のサーバーに対する要求が行われました。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_SECURITY_ERROR
指定したユーザー ID とパスワードの組み合わせが、サーバー側で無効と判断されました。

ECI_ASYNC 呼び出しタイプ

環境

目的

ECI_ASYNC 呼び出しは、非同期プログラム・リンク呼び出しを提供し、作業論理単位を開始、継続、または終了させます。呼び出し側のアプリケーションの処理は、ECI が要求を受け取りしだい再開されます。この時点でパラメーターが検証されます。要求は後の処理のためにキューに入ります。

コールバック・ルーチンを提供しない場合は、アプリケーションから応答請求呼び出しを実行し、要求の終了および結果を確認します。

コールバック・ルーチンを提供する場合には、応答が有効になりしだい **eci_callback** が呼び出されます。

注: コンパイラーによっては、コールバック・ルーチンをサポートしないものもあります。詳細については、使用するコンパイラーの関連資料を参照してください。

注: 未解決の ECI_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果の受信前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

コールバック・ルーチンが呼び出されると、 **eci_message_qualifier** で指定された値が単一のパラメーターとして渡されます。コールバック・ルーチンは、このパラメーターの値によって、完了している非同期呼び出しを認識します。コールバック・ルーチンの使用時には、以下の点を留意してください。

1. コールバック・ルーチンでは必要な最小限の処理を行ってください。
2. コールバック・ルーチンからは、ECI 関数の呼び出しはできません。
3. コールバック・ルーチンには、ECI アプリケーションを実行しているオペレーティング・システムに応じた方法で、応答が有効かどうかをアプリケーションに知らせる機能が必要です。たとえばマルチスレッド環境では、セマフォーをポストすれば、コールバック・ルーチンから他のスレッドにイベント発生を知らせることができます。プレゼンテーション・マネージャー上では、メッセージをウィンドウにポストし、ウィンドウ・プロシージャーにイベント発生を知らせる方法があります。他の環境では、対応する方法で処理してください。
4. コールバック・ルーチンでは、応答の内容を得ることはできません。アプリケーションから応答請求呼び出しを実行する必要があります。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックの各フィールドをヌルに設定する必要があります。

eci_call_type

必須の入力パラメーターです。

必ず ECI_ASYNC に指定します。

eci_program_name

入力パラメーターです。**eci_extend_mode** が ECI_COMMIT または ECI_BACKOUT 以外の場合は必須です。 (13ページの表1 を参照してください。)

呼び出し対象のプログラム名を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。このフィールドの値は、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

eci_userid

必須の入力パラメーターです。

ユーザー ID を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。

クライアントおよびサーバーの関連資料を参照し、このフィールドの値がサーバー伝送時に大文字に変換されるかどうかを確認してください。(8 文字以上のユーザー ID またはパスワードが必要な場合には、**eci_userid** および **eci_password** をヌル・ストリングに指定し、**eci_userid2** and **eci_password2** フィールドを代わりに指定します。)

ユーザー ID を指定すると、サーバーはユーザー ID および (指定されている場合には) パスワードを使用してユーザーの検証を行います。これらのユーザー ID およびパスワードは、以降のセキュリティー・チェックの際にも使用されます。

eci_password
必須の入力パラメーターです。

パスワードを格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。

クライアントおよびサーバーの関連資料を参照し、このフィールドの値がサーバー伝送時に大文字に変換されるかどうかを確認してください。(8 文字以上のユーザー ID またはパスワードが必要な場合には、このフィールドおよび **eci_userid** をヌル・ストリングに指定し、**eci_userid2** および **eci_password2** フィールドを代わりに使用します。)

eci_transid
任意指定の入力パラメーターです。

CICS トランザクションの ID を格納する 4 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。 **eci_tpn** が使用されている (ヌル・ストリング以外の値に設定されている) 場合には、このパラメーターは無視されます。このパラメーターの使用方法は、要求を送るクライアントによって異なります。**eci_transid** の値は、**eci_program_name** で指定されているプログラムへの LINK 時に、大文字変換なしで ASCII から EBCDIC に変換され、EIBTRNID に保管されます。

呼び出されたプログラムはミラー・トランザクション CPMI のもとで実行されますが、**eci_transid** トランザクション名のもとでリンクされます。この名前は、呼び出されたプログラムがトランザクション ID を照会するときに使用されます。サーバーによっては、トランザクション ID を使用して、呼び出されたプログラムのセキュリティー属性およびパフォーマンス属性を判別する場合があります。これらのサーバーでは、呼び出し先プログラムの処理を制御する場合に、このパラメーターを使用することをお勧めします。

ECI 要求を拡張する (**eci_extend_mode** の説明を参照) 場合は、**eci_transid** パラメーターは作業単位 (UOW) の最初の呼び出しに対してだけ意味をもちます。

このフィールドの値がヌル・ストリングで、**eci_tpn** が指定されていない場合、デフォルトのサーバー・トランザクション ID が使用されます。

eci_commarea
必須の入力パラメーターです。

呼び出された CICS プログラムに渡される COMMAREA データを指すポインターです。

COMMAREA が必要ない場合はヌル・ポインターを指定し、長さ (**eci_commarea_length** で指定されます) を 0 にします。

アプリケーションとサーバーのコード・ページが一致しない場合には、サーバーで、データ伝送時にデータ変換を行う必要があります。このデータ変換では、DFHCNV マクロ定義などの CICS 提供のリソース変換機能を使用してください。

eci_commarea_length

必須の入力パラメーターです。

COMMAREA のバイト長です。最大長は、32 500 バイトです。 (クライアント / サーバー環境によっては、これ以上の長さの COMMAREA を指定することが可能な場合もありますが、「CICS ファミリー: クライアント / サーバー・プログラミング」で正常に使用できることは保証されません。)

COMMAREA が必要ない場合には、このフィールドには 0 を指定し、**eci_commarea** フィールドにヌル・ポインターを指定します。

eci_timeout

CICS サーバーからの応答を待つ時間 (秒単位)。 0 の値は、限度が設定されていないことを示します。

タイムアウトが発生すると、会話は異常終了します。

reserved1

出力パラメーターです。

このフィールドは、以前は **eci_system_ return_code** でした。CICS ユニバーサル・クライアントのバージョン 3.1 およびそれ以降のバージョンでは、このフィールドは後方互換性のために予約されています。このフィールドに戻される情報はありません。すべてのシステム・エラーは、クライアントのエラー・ログに書き込まれます。

eci_extend_mode

必須の入力パラメーターです。

作業論理単位が、呼び出しの後に終了するかどうかを決める整数フィールドです。(13ページの表1 を参照してください。)

このフィールドの指定値 (記号名) を以下に示します。

ECI_NO_EXTEND

1. **eci_luw_token** フィールドの入力値が 0 の場合は、作業論理単位に対する単独呼び出しを示します。
2. **eci_luw_token** の入力値が 0 以外の場合は、指定作業論理単位に対する最後の呼び出しを示します。

いずれの場合でも、リカバリー可能リソースへの変更は、CICS タスク終了同期点でコミットされ、作業論理単位が終了します。

ECI_EXTENDED

1. **eci_luw_token** フィールドの入力値が 0 の場合は、継続する拡張作業論理単位に対する最初の呼び出しを示します。
2. **eci_luw_token** フィールドの入力値が 0 以外の場合は、作業論理単位の継続する呼び出しを示します。

いずれの場合でも、呼び出されたプログラムの正常終了後、作業論理単位が継続され、リカバリー可能リソースへの変更は、この時点ではコミットされません。

ECI_COMMIT

eci_luw_token フィールドの入力値で識別される作業論理単位を終了し、リカバリー可能リソースへの変更すべてをコミットします。

ECI_BACKOUT

eci_luw_token フィールドの入力値で識別される作業論理単位を終了し、リカバリー可能リソースへの変更をバックアウトします。

eci_message_qualifier

任意指定の入力パラメーターです。

アプリケーションで複数の非同期呼び出しを行う場合に、各呼び出しを識別する整数フィールドです。コールバック・ルーチンを実行する場合は、このフィールドの値が通知処理の間にコールバック・ルーチンに戻されます。

eci_luw_token

必須の入出力パラメーターです。

呼び出しの対象となる作業論理単位を識別する整数フィールドです。作業論理単位の最初で、作業論理単位が拡張されるかどうかに関係なく、必ず 0 に指定します。 ECI は、最初の (または単独の) 論理作業単位の呼び出し時にこの値を更新します。論理作業単位が拡張される場合、

同じ論理作業単位と関連するそれ以降のすべての呼び出しへの入力値としてこの値を使用します。(13ページの表1 を参照してください。)

戻りコードが ECI_NO_ERROR 以外で、呼び出しが作業論理単位を継続または終了する場合は、このフィールドは作業論理単位の状態をレポートする出力として使用されます。この出力値が 0 の場合は、作業論理単位が終了し、更新がバックアウトされたことを示します。0 以外の場合は、入力値と同じ値であり、作業論理単位が継続され、更新が保留中であることを示します。

eci_sysid

必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version

必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

eci_system_name

任意指定の入力パラメーターです。

ECI 要求の対象サーバーの名前を格納する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。この値は、**eci_luw_token** が 0 の場合に必ず指定します。(**eci_luw_token** が 0 以外の場合は、対象作業論理単位の開始時にサーバーが確立するため、このパラメーターを指定しても無視されます。)

このフィールドがヌルの場合は、現行のデフォルト・サーバーが選択されます。選択されたサーバーの名前は、この非同期要求の応答を獲得する応答請求呼び出しを実行すれば、**eci_system_name** フィールドに格納されます。(この名前は、この要求以降に同一サーバー上の異なる作業論理単位に対して実行する ECI 要求の **eci_system_name** フィールドに指定します。)

eci_callback

任意指定の入力パラメーターです。

非同期要求の完了時に呼び出すルーチンを指すポインターです。 (コールバック・ルーチンは、戻りコードが ECI_NO_ERROR で、このフィールドがヌル・ポインター以外の場合のみ呼び出し可能です。)

eci_userid2

任意指定の入力パラメーターです。

eci_userid フィールドにヌルを指定すると、**eci_userid2** フィールドの値でユーザー ID (存在する場合) が識別されます。ユーザー ID は、サーバー側の各種権限の検証で使用されます。ユーザー ID の最大長は 16 文字です。

ユーザー ID の使用法については、**eci_userid** フィールドの説明を参照してください。

eci_password2

任意指定の入力パラメーターです。

eci_password フィールドがヌルの場合は、 **eci_password2** フィールドの値でパスワード (存在する場合) が指定されます。パスワードは、サーバー側の各種権限の検証で使用されます。パスワードの最大長は 16 文字です。

パスワードの使用法については、**eci_password** フィールドの説明を参照してください。

eci_tpn

任意指定の入力パラメーターです。このパラメーターは、**eci_version** の値が ECI_VERSION_1A の場合にだけ使用することができます。

サーバーが ECI 要求の処理に使用するトランザクションの ID が格納される 4 文字のフィールドです。このトランザクションは、CICS ミラー・トランザクションとしてサーバーで定義されていなければなりません。このフィールドがヌル・ストリングの場合は、デフォルト・ミラー・トランザクション CPMI が使用されます。

ECI 要求が拡張された場合 (**eci_extend_mode** の説明を参照)、このパラメーターは最初の要求に対してだけ意味をもちます。

このフィールドを使用すると、**eci_transid** の内容は無視されます。

戻りコード

18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

戻りコードが ECI_NO_ERROR 以外の場合は、コールバック・ルーチンは呼び出されず、非同期要求への応答も生成されません。

ECI_NO_ERROR
ECI への呼び出しが正常に終了しました。エラーは検出されませんでした。要求の完了時に、コールバック・ルーチンが呼び出されます。

ECI_ERR_INVALID_DATA_LENGTH
eci_commarea_length フィールドに指定した値が無効です (**eci_commarea** フィールドの値と整合性がありません)。ヌル・ポインターではない **eci_commarea** に 0 を指定している、あるいはヌル・ポインターである **eci_commarea** に 0 以外の値を指定しています。

ECI_ERR_INVALID_EXTEND_MODE
eci_extend_mode フィールドの値が無効です。

ECI_ERR_NO_CICS
クライアントまたはサーバーは使用可能ではありません。

ECI_ERR_LUW_TOKEN
eci_luw_token の値が無効です。

ECI_ERR_THREAD_CREATE_ERROR
サーバーまたはクライアントが、要求を処理するためのスレッドを生成できません。

ECI_ERR_ALREADY_ACTIVE
作業論理単位の継続要求時に、その作業論理単位に未完了の非同期呼び出しが存在します。

ECI_ERR_RESOURCE_SHORTAGE
サーバーまたはクライアント上に、要求の完了に必要なリソースが不足しています。

ECI_ERR_NO_SESSIONS
最大数の作業論理単位を実行しているアプリケーションから、さらに新規作業論理単位の作成が要求されました。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_STATE_SYNC 呼び出しタイプ

環境

ECI_STATE_SYNC 呼び出しは、あらゆる環境で使用可能です。

目的

ECI_STATE_SYNC 呼び出しは同期状況情報呼び出しを実行します。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックの各フィールドをヌルに設定する必要があります。

eci_call_type

必須の入力パラメーターです。

必ず ECI_STATE_SYNC に設定します。

eci_commarea

入力パラメーターです。**eci_extend_mode** の値が ECI_STATE_CANCEL 以外の場合は必ず指定します。

アプリケーションが受信する COMMAREA の格納域を指すポインターです。 COMMAREA には状況情報が格納されます。 (詳細については、15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)

eci_extend_mode に ECI_STATE_CANCEL を指定する場合は、このフィールドにヌル・ポインターを指定し、長さ (**eci_commarea_length**) を 0 に指定します。

eci_commarea_length

入力パラメーターです。**eci_extend_mode** の値が ECI_STATE_CANCEL 以外の場合は必ず指定します。

COMMAREA のバイト長。 COMMAREA の状況情報のレイアウトを与える ECI_STATUS 構造体のバイト長を指定します。 (詳細については、15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。) 領域サイズの最大長は、32 500 バイトです。

COMMAREA が必要ない場合には、このフィールドには 0 を指定し、**eci_commarea** フィールドにヌル・ポインターを指定します。

reserved1

出力パラメーターです。

このフィールドは、以前は **eci_system_ return_code** でした。CICS ユニバーサル・クライアントのバージョン 3.1 およびそれ以降のバージョンでは、このフィールドは後方互換性のために予約されています。このフィールドに戻される情報はありません。すべてのシステム・エラーは、クライアントのエラー・ログに書き込まれます。

eci_extend_mode

必須の入力パラメーターです。

呼び出しタイプの詳細を指定する整数フィールドです。(16ページの表2を参照してください。) このフィールドに格納される値 (記号名) を以下に示します。

ECI_STATE_IMMEDIATE

状況応答が使用可能になりしだい、強制的に送信します。戻された COMMAREA のレイアウトは、ECI_STATUS 構造体で定義されています (15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)

ECI_STATE_CHANGED

状況が変更されしだい、状況情報を強制的に送信します (据え置き状況要求)。アプリケーションは、COMMAREA に戻される状況情報をサポートする必要があります。状況情報の送信は、アプリケーションが指定した状況が変化した場合のみ実行されます。 COMMAREA のレイアウトは、ECI_STATUS 構造体で定義されます。 (詳細については、15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)要求の間に、**eci_luw_token** フィールドに要求を識別するトークンが格納されます。

ECI_STATE_CANCEL

ECI_STATE_CHANGED 呼び出しの操作を取り消します。この要求では、COMMAREA は必要ありません。**eci_luw_token** フィールドには、据え置き呼び出しの間に戻されるトークンを指定します。

eci_luw_token

任意指定の入出力パラメーターです。

据え置き状況要求が、(**eci_extend_mode** を ECI_STATE_CHANGED に設定して) セットアップされると、要求を識別するトークンが **eci_luw_token** フィールドに戻されます。

据え置き状況要求の取り消しが行われると (**eci_extend_mode** を ECI_STATE_CANCEL に設定して)、ECI_STATE_CHANGED 呼び出しで戻されたトークンを **eci_luw_token** フィールドに指定します。

eci_extend_mode にそれ以外の値を指定した場合、このフィールドは使用しません。

eci_sysid
必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version
必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

eci_system_name
任意指定の入力パラメーターです。

状況情報が要求されているサーバーの名前を指定する 8 文字のフィールドです。このパラメーターを指定する場合は、名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。**CICS_EciListSystems** 関数が戻すサーバー名のいずれかを指定します。この値は、**eci_luw_token** が 0 の場合に必ず指定します。

このフィールドの値がヌルの場合は、現行のデフォルト・サーバーが選択され、このサーバーの名前がこのフィールドに格納されます。

戻りコード
18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

ECI_NO_ERROR
呼び出しが正常に終了しました。

ECI_ERR_INVALID_DATA_LENGTH
eci_commarea_length フィールドに指定した値が無効です (**eci_commarea** フィールドの値と整合性がありません)。ヌル・ポイ

ンターではない **eci_commarea** に 0 を指定している、あるいはヌル・ポインターである **eci_commarea** に 0 以外の値を指定しています。

ECI_ERR_INVALID_EXTEND_MODE
eci_extend_mode フィールドの値が無効です。

ECI_ERR_LUW_TOKEN
eci_luw_token の値が無効です。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_ERR_UNKNOWN_SERVER
要求対象のサーバーが見つかりません。**CICS_EciListSystems** で戻されたサーバーのみが受け入れ可能です。

ECI_STATE_ASYNC 呼び出しタイプ

環境

目的

ECI_STATE_ASYNC 呼び出しは、非同期状況情報呼び出しを提供します。呼び出し側のアプリケーションの処理は、ECI が要求を受け取りしだい再開されます。この時点でパラメーターが検証されます。要求は後の処理のためにキューに入ります。

コールバック・ルーチンを提供しない場合は、アプリケーションから応答請求呼び出しを実行し、要求の終了および結果を確認します。

コールバック・ルーチンを提供する場合には、応答が有効になりしだい **eci_callback** が呼び出されます。

注: コンパイラーによっては、コールバック・ルーチンをサポートしないものもあります。詳細については、使用するコンパイラーの関連資料を参照してください。

注: 未解決の ECI_STATE_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果の受信前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

コールバック・ルーチンが呼び出されると、 **eci_message_qualifier** で指定された値が単一のパラメーターとして渡されます。コールバック・ルーチンは、このパラメーターの値によって、完了している非同期呼び出しを認識します。コールバック・ルーチンの使用時には、以下の点を留意してください。

1. コールバック・ルーチンでは必要な最小限の処理を行ってください。
2. コールバック・ルーチンからは、ECI 関数の呼び出しはできません。
3. コールバック・ルーチンには、ECI アプリケーションを実行しているオペレーティング・システムに応じた方法で、応答が有効かどうかをアプリケーションに知らせる機能が必要です。たとえばマルチスレッド環境では、セマフォーをポストすれば、コールバック・ルーチンから他のスレッドにイベント発生を知らせることができます。プレゼンテーション・マネージャー上では、メッセージをウィンドウにポストし、ウィンドウ・プロシージャーにイベント発生を知らせる方法があります。他の環境では、対応する方法で処理してください。
4. コールバック・ルーチンでは、応答の内容を得ることはできません。アプリケーションから応答請求呼び出しを実行する必要があります。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックをヌルに設定する必要があります。

eci_call_type

必須の入力パラメーターです。

必ず ECI_STATE_ASYNC に設定します。

eci_commarea

入力パラメーターです。**eci_extend_mode** の値が ECI_STATE_CANCEL 以外の場合は必ず指定します。

アプリケーションが受信する COMMAREA の格納域を指すポインターです。 COMMAREA には状況情報が格納されます。 (詳細については、15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)

eci_extend_mode に ECI_STATE_CANCEL を指定する場合は、このフィールドにヌル・ポインターを指定し、長さ (**eci_commarea_length**) を 0 に指定します。

eci_commarea_length

入力パラメーターです。**eci_extend_mode** の値が ECI_STATE_CANCEL 以外の場合は必ず指定します。

COMMAREA のバイト長。 COMMAREA の状況情報のレイアウトを与える ECI_STATUS 構造体のバイト長を指定します。 (詳細については、15ページの『状況情報呼び出し』 および 56ページの『ECI 状況ブロック』を参照してください。) 領域サイズの最大長は、32 500 バイトです。

COMMAREA が必要ない場合には、このフィールドには 0 を指定し、**eci_commarea** フィールドにヌル・ポインターを指定します。

reserved1

出力パラメーターです。

このフィールドは、以前は **eci_system_ return_code** でした。CICS ユニバーサル・クライアントのバージョン 3.1 およびそれ以降のバージョンでは、このフィールドは後方互換性のために予約されています。このフィールドに戻される情報はありません。すべてのシステム・エラーは、クライアントのエラー・ログに書き込まれます。

eci_extend_mode

必須の入力パラメーターです。

呼び出しタイプの詳細を指定する整数フィールドです。(16ページの表2を参照してください。) このフィールドに格納される値 (記号名) を以下に示します。

ECI_STATE_IMMEDIATE
状況応答が使用可能になりしだい、強制的に送信します。戻された COMMAREA のレイアウトは、 ECI_STATUS 構造体で定義されます (15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)

ECI_STATE_CHANGED
状況が変更されしだい、状況情報を強制的に送信します (据え置き状況要求)。アプリケーションは、COMMAREA に戻される状況情報をサポートする必要があります。状況情報の送信は、アプリケーションが指定した状況が変化した場合のみ実行されます。 COMMAREA のレイアウトは、ECI_STATUS 構造体で定義されます。 (詳細については、15ページの『状況情報呼び出し』および 56ページの『ECI 状況ブロック』を参照してください。)即時応答で戻される、**eci_luw_token** フィールドは対象の作業論理単位を識別します。

ECI_STATE_CANCEL
ECI_STATE_CHANGED 呼び出しの操作を取り消します。この要求では、COMMAREA は必要ありません。**eci_luw_token** フィールドには、据え置き呼び出しの間に戻されるトークンを指定します。

eci_message_qualifier
任意指定の入力パラメーターです。

複数の非同期呼び出しを行う際に、各呼び出しの識別に使用する整数フィールドです。コールバック・ルーチンを実行する場合は、このフィールドの値が通知処理の間にコールバック・ルーチンに戻されます。

eci_luw_token
任意指定の入出力パラメーターです。

据え置き状況要求が、(**eci_extend_mode** を ECI_STATE_CHANGED に設定して) セットアップされると、要求を識別するトークンが **eci_luw_token** フィールドに戻されます。

据え置き状況要求の取り消しが行われると (**eci_extend_mode** を ECI_STATE_CANCEL に設定して)、ECI_STATE_CHANGED 呼び出しで戻されたトークンを **eci_luw_token** フィールドに指定します。

eci_extend_mode にそれ以外の値を指定した場合、このフィールドは使用しません。

eci_sysid
必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version
必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

eci_system_name
任意指定の入力パラメーターです。

状況情報要求の対象サーバーの名前を指定する 8 文字のフィールドです。名前が規定の文字数に足りない場合には、残りをスペースで埋め込みます。このパラメーターを指定する場合は、**CICS_EciListSystems** 関数が戻すサーバー名のいずれかを指定します。この値は、**eci_luw_token** が 0 の場合に必ず指定します。

このフィールドがヌルの場合は、現行のデフォルト・サーバーが選択されます。 サーバーの名前は、この非同期要求の結果を入手するために使用する応答請求呼び出しの **eci_system_name** フィールドに入っています。

eci_callback
任意指定の入力パラメーターです。

非同期要求の完了時に呼び出すルーチンを指すポインターです。 (コールバック・ルーチンは、戻りコードが ECI_NO_ERROR で、このフィールドがヌル・ポインター以外の場合のみ呼び出し可能です。)

戻りコード

18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

戻りコードが ECI_NO_ERROR 以外の場合は、コールバック・ルーチンは呼び出されず、非同期要求への応答も生成されません。

ECI_NO_ERROR
呼び出しが正常に終了しました。

ECI_ERR_INVALID_DATA_LENGTH
eci_commarea_length フィールドに指定した値が無効です (**eci_commarea** フィールドの値と整合性がありません)。ヌル・ポインターではない **eci_commarea** に 0 を指定している、あるいはヌル・ポインターである **eci_commarea** に 0 以外の値を指定しています。

ECI_ERR_INVALID_EXTEND_MODE
eci_extend_mode フィールドの値が無効です。

ECI_ERR_LUW_TOKEN
eci_luw_token の値が無効です。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_GET_REPLY 呼び出しタイプ

環境

目的

ECI_GET_REPLY 呼び出しは応答を請求する呼び出しを行い、未解決の非同期要求への応答として適している情報すべてを戻します。適切な応答がない場合には、ECI_ERR_NO_REPLY を戻します。(代わりに ECI_GET_REPLY_WAIT 呼び出しを使用すれば、アプリケーションは応答が使用可能になるまで処理を中断します。)

注: 未解決の ECI_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果を受信する (たとえば、この GET REPLY 呼び出しを使用して) 前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックの各フィールドをヌルに設定する必要があります。

この呼び出しの実行時に指定可能な ECI パラメーター・ブロックの入力フィールドを以下に示します。

ECI_GET_REPLY 呼び出しは、ECI パラメーターの値を以下のように更新します。

1. 応答の全出力がフィールドに追加されます。入力フィールドの値が置き換えられる場合もあります。これらは対象の非同期要求に対応する同期要求でも出力されるフィールドです。
2. 応答に関連する非同期要求の入力として指定された **eci_message_qualifier** の値がリストアされます。
3. COMMAREA を指すポインター以外の、更新されていない入力フィールドが未定義状態になります。これらのフィールドの値は使用しないでください。

eci_call_type

必須の入力パラメーターです。

必ず ECI_GET_REPLY に設定します。

eci_commarea

任意指定の入力パラメーターです。

アプリケーションが受け取る COMMAREA の格納域を指すポインターです。戻される COMMAREA の内容は、対象の応答に関連する非同期呼び出しのタイプによって異なります。プログラム・リンク呼び出しの場合には、呼び出されたプログラムの戻りデータ (戻りデータがない場合もあります) が格納されます。状況情報呼び出しでは、**eci_extend_mode** の値が ECI_STATE_CANCEL 以外の場合、状況情報が格納されます (詳しくは 15ページの『状況情報呼び出し』および56ページの『ECI 状況ブロック』を参照してください)。

COMMAREA が必要ない場合はヌル・ポインターを指定し、長さ (**eci_commarea_length** で指定されます) を 0 にします。

アプリケーションとサーバーのコード・ページが一致しない場合には、サーバーで、データ伝送時にデータ変換を行う必要があります。このデータ変換では、DFHCNV マクロ定義などの CICS 提供のリソース変換機能を使用してください。

eci_commarea_length
必須の入力パラメーターです。

COMMAREA のバイト長です。最大長は、32 500 バイトです。 (クライアント / サーバー環境によっては、これ以上の長さのデータを指定することが可能な場合もありますが、「CICS ファミリー: クライアント / サーバー・プログラミング」で正常に使用できることは保証されません。)

COMMAREA が必要ない場合には、このフィールドには 0 を指定し、**eci_commarea** フィールドにヌル・ポインターを指定します。

eci_sysid
必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version
必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

戻りコード

18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

ECI_NO_ERROR
この応答に関連する非同期要求が正常に終了しました。

ECI_ERR_INVALID_DATA_LENGTH
eci_commarea_length フィールドの値が受け入れられません。このフィールドの値が、以下のいずれかの状態になっています。

- 有効な範囲を外れている。
- **eci_commarea** フィールドの値と整合性がない、つまり、0 が指定されているが、**eci_commarea** がヌル・ポインターではない、または 0 以外の値が指定されているが、**eci_commarea** ポインターがヌル・ポインターです。
- 指定値が小さく、応答に関連する非同期呼び出しが出力を COMMAREA に格納できない。

指定値が小さすぎる場合は、**eci_commarea_length** に出力される値を参考にして COMMAREA 格納域のサイズを拡大します。
eci_message_qualifier の出力で対象の非同期要求が識別できる場合は、その値を使用して ECI_GET_SPECIFIC_REPLY 呼び出しを再度出せば応答が獲得できます。

ECI_ERR_NO_CICS
応答に関連する非同期要求の **eci_system_name** に指定された CICS サーバーが使用可能ではありません。リソースの更新は行われていません。

ECI_ERR_CICS_DIED
応答に関連する非同期要求によって作業論理単位の開始または継続が要求されましたが、対象の CICS サーバーは使用可能ではありませんでした。 **eci_extend_mode** の値が ECI_EXTENDED の場合は、変更はバックアウトされ、作業論理単位が終了します。 **eci_extend_mode** の値が ECI_NO_EXTEND、ECI_COMMIT、または ECI_BACKOUT である場合には、アプリケーションは変更がコミットされたかバックアウトされたかを判別できません。したがって、今後手動リカバリーを行わなければならない機会があったときのため、この状態を記録しておく必要があります。

ECI_ERR_NO_REPLY
未解決の応答がありません。

ECI_ERR_TRANSACTION_ABEND
応答に関連する非同期要求でサーバー上のプログラムが開始されましたが、プログラムを実行した CICS トランザクションが異常終了しました。異常終了コードが、**eci_abend_code** に格納されます。異常終了コードの種類と意味については、要求対象のサーバー・システムの関連資料を参照してください。

ECI_ERR_THREAD_CREATE_ERROR
CICS サーバーまたはクライアントが、応答に関連する非同期呼び出しを処理するスレッド生成に失敗しました。

ECI_ERR_RESOURCE_SHORTAGE
サーバーまたはクライアント上に、応答に関連する非同期要求を完了するために必要なリソースが不足しています。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_ERR_ROLLEDBACK
応答に関連する非同期要求が作業論理単位のコミットを要求しましたが、サーバーで変更がコミットできずバックアウトされました。

ECI_ERR_UNKNOWN_SERVER
応答に関連する非同期要求で指定されたサーバーが見つかりません。**CICS_EciListSystems** で戻されたサーバーのみが受け入れ可能です。

ECI_ERR_MAX_SESSIONS
応答に関連する非同期要求のための通信リソースが不足しています。通信リソースの制御方法については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_MAX_SYSTEMS
応答に関連する非同期要求が、可能な数以上のサーバーに開始要求を送信しました。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_SECURITY_ERROR
応答に関連する非同期要求で指定されたユーザー ID とパスワードが、サーバーの要求する組合わせと一致しません。

ECI_GET_REPLY_WAIT 呼び出しタイプ

環境

目的

ECI_GET_REPLY_WAIT 呼び出しは、応答を請求する呼び出しを行い、非同期要求の未解決の応答すべてに適切な情報を戻します。適切な応答がない場合、アプリケーションは応答が使用可能になるまで処理を中断します。(代わりに、ECI_GET_REPLY 呼び出しを使用して、応答が使用可能でないという通知を取得できます。)

注: 未解決の ECI_STATE_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果の受信前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

ECI パラメーター・ブロック・フィールド

ECI_GET_REPLY と同様ですが、**eci_call_type** には ECI_GET_REPLY_WAIT を指定します。

戻りコード

ECI_GET_REPLY と同様ですが、ECI_ERR_NO_REPLY は戻されません。

ECI_GET_SPECIFIC_REPLY 呼び出しタイプ

目的

ECI_GET_SPECIFIC_REPLY 呼び出しは、応答を請求する呼び出しを行い、**eci_message_qualifier** 入力に合致する、未解決の応答すべてに適切な情報を戻します。適切な応答がない場合には、ECI_ERR_NO_REPLY を戻します。(代わりに ECI_GET_REPLY_WAIT 呼び出しを使用すれば、アプリケーションは応答が使用可能になるまで処理を中断します。)

注: 未解決の ECI_STATE_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果の受信前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

ECI パラメーター・ブロック・フィールド

入力パラメーター・フィールドの設定の前に、ECI パラメーター・ブロックの各フィールドをヌルに設定する必要があります。

この呼び出しの実行時に指定可能な ECI パラメーター・ブロックの入力フィールドを以下に示します。

ECI_GET_REPLY 呼び出しは、ECI パラメーターの値を以下のように更新します。

1. 応答の全出力がフィールドに追加されます。入力フィールドの値が置き換えられる場合もあります。これらは対象の非同期要求に対応する同期要求でも出力されるフィールドです。
2. COMMAREA を指すポインター、および入力の **eci_message_qualifier** を除く、更新されていない全入力フィールドが未定義状態になります。これらのフィールドの値は使用しないでください。

eci_call_type

必須の入力パラメーターです。

必ず ECI_GET_SPECIFIC_REPLY に設定します。

eci_commarea

任意指定の入力パラメーターです。

アプリケーションが受け取る COMMAREA の格納域を指すポインターです。戻される COMMAREA の内容は、対象の応答に関連する非同期呼び出しのタイプによって異なります。プログラム・リンク呼び出しの場合には、呼び出されたプログラムの戻りデータ (戻りデータがない場合もあります) が格納されます。状況情報呼び出しでは、

eci_extend_mode の値が ECI_STATE_CANCEL 以外の場合、状況情報が格納されます (詳しくは 15ページの『状況情報呼び出し』および56ページの『ECI 状況ブロック』を参照してください)。

アプリケーションとサーバーのコード・ページが一致しない場合には、サーバーで、データ伝送時にデータ変換を行う必要があります。このデータ変換では、DFHCNV マクロ定義などの CICS 提供のリソース変換機能を使用してください。

eci_commarea_length
必須の入力パラメーターです。

COMMAREA のバイト長です。最大長は、32 500 バイトです。 (クライアント / サーバー環境によっては、これ以上の長さのデータを指定することが可能な場合もありますが、「CICS ファミリー: クライアント / サーバー・プログラミング」で正常に使用できることは保証されません。)

eci_message_qualifier
必須の入力パラメーターです。

応答を請求する非同期呼び出しを識別する整数フィールドです。

eci_sysid
必須の入力パラメーターです。

このバージョンでは使用されません。各作業論理単位の開始時に、必ずヌルに設定してください。

eci_version
必須の入力パラメーターです。

アプリケーションに対応する ECI のバージョンを示します。ECI_VERSION_1 または ECI_VERSION_1A の値が使用できます。バージョン 1 の機能は、すべてバージョン 1A で使用可能です。機能によっては、バージョン 1A でのみ使用可能なものがあります。その場合、説明の中で使用できるバージョンを明記します。

戻りコード

18ページの『CICS_ExternalCall』の **CICS_ExternalCall** の他の戻りコードのリストも参照してください。

ECI_NO_ERROR
呼び出しが正常に終了しました。

ECI_ERR_INVALID_DATA_LENGTH

eci_commarea_length フィールドの値が受け入れられません。このフィールドの値が、以下のいずれかの状態になっています。

- 有効な範囲を外れている。
- **eci_commarea** フィールドの値と整合性がない、つまり、0 が指定されているが、**eci_commarea** がヌル・ポインターではない、または 0 以外の値が指定されているが、**eci_commarea** ポインターがヌル・ポインターです。
- 指定値が小さく、応答に関連する非同期呼び出しが出力を COMMAREA に格納できない。

指定値が小さすぎる場合には、**eci_commarea_length** の値を参考にして COMMAREA のサイズを拡大した後、ECI_GET_SPECIFIC_REPLY 呼び出しを再度出します。

ECI_ERR_NO_CICS

応答に関連する非同期要求の **eci_system_name** に指定された CICS サーバーが使用可能ではありません。リソースの更新は行われていません。

ECI_ERR_CICS_DIED

応答に関連する非同期要求によって作業論理単位の開始または継続が要求されましたが、対象の CICS サーバーは使用可能ではありませんでした。 **eci_extend_mode** の値が ECI_EXTENDED の場合は、変更はバックアウトされ、作業論理単位が終了します。 **eci_extend_mode** の値が ECI_NO_EXTEND、ECI_COMMIT、または ECI_BACKOUT である場合には、アプリケーションは変更がコミットされたかバックアウトされたかを判別できません。したがって、今後手動リカバリーを行わなければならない機会があったときのため、この状態を記録しておく必要があります。

ECI_ERR_NO_REPLY

入力の **eci_message_qualifier** に合致する未解決の応答が存在しません。

ECI_ERR_TRANSACTION_ABEND

応答に関連する非同期要求でサーバー上のプログラムが開始されましたが、プログラムを実行した CICS トランザクションが異常終了しました。異常終了コードが、**eci_abend_code** に格納されます。異常終了コードの種類と意味については、要求対象のサーバー・システムの関連資料を参照してください。

ECI_ERR_THREAD_CREATE_ERROR
CICS サーバーまたはクライアントが、応答に関連する非同期呼び出しを処理するスレッド生成に失敗しました。

ECI_ERR_RESOURCE_SHORTAGE
CICS サーバーまたはクライアントに、応答に関連する非同期要求を完了するためのリソースが不足しています。

ECI_ERR_INVALID_DATA_AREA
ECI パラメーター・ブロックを指すポインターまたは **eci_commarea** に指定されたポインターが無効です。

ECI_ERR_ROLLEDBACK
応答に関連する非同期要求が作業論理単位のコミットを要求しましたが、サーバーで変更がコミットできずバックアウトされました。

ECI_ERR_UNKNOWN_SERVER
応答に関連する非同期要求で指定されたサーバーが見つかりません。**CICS_EciListSystems** で戻されたサーバーのみが受け入れ可能です。

ECI_ERR_MAX_SESSIONS
応答に関連する非同期要求のための通信リソースが不足しています。通信リソースの制御方法については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_MAX_SYSTEMS
応答に関連する非同期要求が、可能な数以上のサーバーに開始要求を送信しました。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

ECI_ERR_SECURITY_ERROR
応答に関連する非同期要求で指定されたユーザー ID とパスワードが、サーバーの要求する組合わせと一致しません。

ECI_GET_SPECIFIC_REPLY_WAIT 呼び出しタイプ

環境

目的

ECI_GET_SPECIFIC_REPLY_WAIT 呼び出しは、応答を請求する呼び出しを行い、 **eci_message_qualifier** の入力値に合致する、未解決の応答すべてに適切な情報を戻します。適切な応答がない場合、アプリケーションは応答が使用可能になるまで処理を中断します。(代りに ECI_GET_SPECIFIC_REPLY 呼び出しを使用すると、使用可能な応答がないことが示されます。)

注: 未解決の ECI_STATE_ASYNC 呼び出しの Eci パラメーター・ブロックが、呼び出しの結果の受信前に変更されないということは、とても重要です。呼び出しの結果を受信する前にこれらのブロックが変更されると、誤った結果になります。

ECI パラメーター・ブロック・フィールド

ECI_GET_SPECIFIC_REPLY と同様ですが、**eci_call_type** には ECI_GET_SPECIFIC_REPLY_WAIT を指定します。

戻りコード

ECI_GET_SPECIFIC_REPLY と同様ですが、 ECI_ERR_NO_REPLY は戻しません。

注: **eci_extend mode** が ECI_STATE_CHANGED に設定された未解決の ECI_STATE_AYSNC 呼び出しに対して ECI_GET_SPECIFIC_REPLY_WAIT 呼び出しを出す場合は、**eci_extend_mode** が ECI_STATE_CANCEL に設定された ECI_STATE_ASYNC 呼び出しが出されると、応答はまったく受信されません。

ECI 状況ブロック

ECI 状況ブロックは、状況情報呼び出しで、ECI との情報の受け渡しに使用します。 ECI 状況ブロックには、以下のフィールドがあります。

ConnectionType

アプリケーションが稼働しているシステムの種類を示す整数フィールドです。以下のいずれかの値が格納されます。

ECI_CONNECTED_NOWHERE

アプリケーションはシステムに接続されていません。

ECI_CONNECTED_TO_CLIENT
アプリケーションはクライアント・システム上で稼働しています。

ECI_CONNECTED_TO_SERVER
アプリケーションはサーバーの ECI を使用しています。

CicsServerStatus
CICS サーバーの状態を示す整数フィールドです。以下のいずれかの値が格納されます。

ECI_SERVERSTATE_UNKNOWN
CICS サーバーの状態が特定できません。

ECI_SERVERSTATE_UP
CICS サーバー上でプログラムの稼働が可能です。

ECI_SERVERSTATE_DOWN
CICS サーバー上でプログラムの稼働に使用できません。

CicsClientStatus
クライアントの状態を示す整数フィールドです。以下のいずれかの値が格納されます。

ECI_CLIENTSTATE_UNKNOWN
クライアントの状態が特定できません。

ECI_CLIENTSTATE_UP
クライアントで ECI 呼び出しの受信が可能です。

ECI_CLIENTSTATE_INAPPLICABLE
アプリケーションがサーバーの ECI を使用しています。

CICS_EciListSystems

CICS_EciListSystems	NameSpace Systems List

目的

CICS_EciListSystems 関数は、**CICS_ExternalCall** 要求が送信される CICS サーバーのリストを戻します。リストに含まれる各サーバーには、クライアントとの間に通信リンクが存在しないもの、および要求の処理に使用可能でないものが含まれる可能性があります。

サーバーのリストは、システム情報構造の配列に格納されます。配列の各エレメントが、1 台のサーバーになります。この構造は **CICS_EciSystem_t** と呼ばれ、以下のフィールドが含まれます。

SystemName
CICS サーバーの名前が格納される、ヌルで終了するストリングを指すポインターです。サーバー名の文字数が、CICS_ECI_SYSTEM_MAX の値より少ない場合は、残りにヌルが格納されます。ヌルで終了するため、ストリングの長さは CICS_ECI_SYSTEM_MAX + 1 になります。

Description
システムの記述が (あれば) 格納される、ヌルで終了するストリングを指すポインターです。情報の文字数が、CICS_ECI_DESCRIPTION_MAX の値より小さい場合には、残りにヌルが格納されます。ヌルで終了するため、ストリングの長さは CICS_ECI_DESCRIPTION_MAX + 1 になります。

パラメーター

NameSpace
将来の使用のために予約されたポインターです。ヌル・ポインターであることを確認してください。

Systems
関数の実行時に、**List** パラメーター内の配列のエレメント数を指定します。実行が終了すると、実際に見つかったシステム数が格納されます。

List 関数の戻りデータが格納される **CICS_EciSystem_t** 構造の配列です。アプリケーション側で配列用の格納域をとり、さらに **Systems** パラメーターで配列内のエレメント数を指定する必要があります。リストの

最初の要素には、デフォルトのサーバーの名前が格納されます。デフォルトのシステムの定義は、オペレーティング・システムによって異なります。

戻りコード

ECI_NO_ERROR
関数の処理が正常に完了しました。**Systems** パラメーター入力値で指定した数以下のシステムが見つかりました。

ECI_ERR_MORE_SYSTEMS
List 配列のスペースが不足しすべての情報を格納できません。指定した配列はすべて情報が格納され、さらに見つかったシステムの数に合わせて **Systems** パラメーターが更新されました。十分なサイズの配列を再割り当てし、関数を再実行してください。

ECI_ERR_NO_SYSTEMS
CICS サーバーが見つかりません。**Systems** の戻り値は 0 になります。

ECI_ERR_NO_CICS
クライアントが活動していません。

ECI_ERR_INVALID_DATA _LENGTH
Systems パラメーターの指定値が大きすぎるため、**List** パラメーターの格納域が 32 767 バイトを超過しました。

ECI_ERR_CALL_FROM_CALLBACK
コールバック・ルーチンから呼び出しが行われました。

ECI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。

第3章 外部表示インターフェース

本章では、外部表示インターフェース (EPI) について解説します。

インターフェースのエレメントに選んだ名前、つまり、関数、パラメーター、データ構造、フィールド、および定数などは、プログラミングで提供される名前と似ていますが、ここで解説するインターフェースの適用範囲は、特定の言語に限定されません。特定言語に関連した情報については、137ページの『第4章 ECI および EPI アプリケーション・プログラムの作成』を参照してください。

特定のオペレーティング・システムのみに適用される関数制限は、関連する関数の説明に記載されています。

本章の構成は以下のとおりです。

概要

EPI は、CICS 以外のアプリケーションから複数の CICS サーバーへのアクセスを可能にするインターフェースです。アプリケーションは、各 CICS サーバー上で複数の端末リソースを確立できます。これらの端末リソースのオペレーターとしてアプリケーションは、CICS トランザクションの開始、およびトランザクションに関連する標準 3270 データ・ストリームの送受信を行うことができます。

クライアント / サーバー実施用の EPI を設計する場合は、以下の点に留意してください。

1. EPI アプリケーションにデータを送信する CICS トランザクションでは、基本マッピング・サポート (BMS) のページングを使用してはなりません。
2. EPI アプリケーションは、EPI を使って端末リソース定義のデフォルト画面サイズを決めることが可能ですが、代替画面サイズは決められません。

3. EPI アプリケーションは、CICS トランザクションとの間での送受信が、2 バイト文字セット (DBCS) の送受信を行うかどうかを EPI によって判断することはできません。
4. CICS トランザクションで生成される 3270 データ・ストリームには、構造化フィールドは格納できません。
5. CICS ユニバーサル・クライアントの EPI 端末の場合は、最大画面サイズは 27 行 x 132 列です。これは、CICS ユニバーサル・クライアントが 3270 データ・ストリーム・アーキテクチャーの ASCII-7 サブセット (これは 12 ビット・アドレッシングのみをサポートします) をサポートするからです。詳細については、130ページの『EPI の 3270 データ・ストリーム』を参照してください。
6. CICS トランザクションを、EPI アプリケーションの端末リソースによる実行診断機能 (EDF) で稼働する場合、EPI アプリケーションに通知されるイベントは、EDF が使用可能でない場合のものとは異なります。
7. CICS トランザクションで DELAY オプションとともに EXEC CICS START を使用して、EPI アプリケーションによって自動インストールされた端末リソースにトランザクションをスケジュールする場合は、EPI アプリケーションから端末リソースを削除する際に注意しなければなりません。さもなければ、遅延状態の ATI 要求が失われる場合があります。遅延状態の ATI 要求が未解決の場合に、端末リソースを削除する影響については、サーバーの関連資料を参照してください。

受信した 3270 データの表現は、アプリケーション側で行う必要があります。表現には、通常の 3270 端末のエミュレーション以外にも、さまざまな方法があります。例を以下に示します。

- Windows® アプリケーションの場合は、Windows® のグラフィカル・ユーザー・インターフェースを使用できます。
- SunOS または Solaris のアプリケーションは、Open Look が使用できます。

EPI の使用方法

EPI では、関数のおのおのに異なる関数名を提供します。

初期化と終了

EPI を使用するアプリケーションは、**CICS_EpiInitialize** 関数を呼び出して EPI を初期化する必要があります。この関数が実行されないと、他の EPI 関数も使用できません。**CICS_EpiInitialize** 関数のパラメーターとして、アプリケ

ーションを作成した EPI のバージョンを指定します。このパラメーター指定によって、EPI が拡張された場合でも、アプリケーションを変更することなく使用できます。

EPI アプリケーションを終了する際には、**CICS_EpiTerminate** 関数を呼び出して、EPI を明確に終了させます。

構成に含まれるサーバー・リストの獲得

CICS_EpiInitialize 呼び出しが正常に終了すると、アプリケーションは、**CICS_EpiListSystems** を呼び出し、どの CICS サーバーがそのアプリケーションが使用できるように構成されているかを調べます。

端末リソースの追加

EPI は、アプリケーションを 1 つ以上の 3270 端末として CICS サーバーに認識させます。

端末リソースの追加は、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** のいずれかの関数を呼び出すことによって行われます。

アプリケーション側で、端末リソースをインストールするサーバーを指定することができます。指定しないと、サーバー (現在は、デフォルトのサーバー) が選択されます。

アプリケーション側で、以下のいずれかを指定して、端末リソースをインストールするサーバーを指定することができます。

- サーバー上の既存の端末リソースのネット名。現行で使用されていない端末リソースを指定します。
- モデル端末定義の名前。この場合、サーバーは、指定されたモデル端末定義を使用して、ネット名を生成し、そのネット名を持つ端末リソースを自動インストールします。
- なにも指定しない。この場合、サーバーは、デフォルトのモデル端末定義を使用して、ネット名を生成し、その名前を持つ端末リソースを自動インストールします。

アプリケーションで端末リソースをインストールすると、他のアプリケーションは、最初のアプリケーションがその端末リソースを削除するまでそれを使用できません。

CICS_EpiAddTerminal および **CICS_EpiAddExTerminal** 関数は、*端末索引* を戻します。この端末索引は、関数が適用される端末リソースを示すために、後続の EPI 関数呼び出しに渡さなければなりません。各索引は、サーバーと端末リソースの組み合わせを識別します。端末索引は整数で、 0 から順に各端末リソースに割り当てられます。したがってアプリケーション側では、配列の検索によって、端末索引と端末リソースの組み合わせを得ることができます。

1 つのアプリケーションが使用する各端末索引は固有値ですが、他のアプリケーションの端末索引の値とは重複可能です。各アプリケーションがインストールする端末リソースには、0 から順に整数の索引が割り当てられます。

図6 では、アプリケーションと導入された 3 つの端末リソースを示しています。端末リソースは 1 つのサーバー上に 1 つ、他のサーバー上に 2 つ導入されています。アプリケーションは端末索引を使用して端末リソースを操作し、3270 データ・ストリームを送受信します。索引の順序は、端末リソースがインストールされた順序を示します。

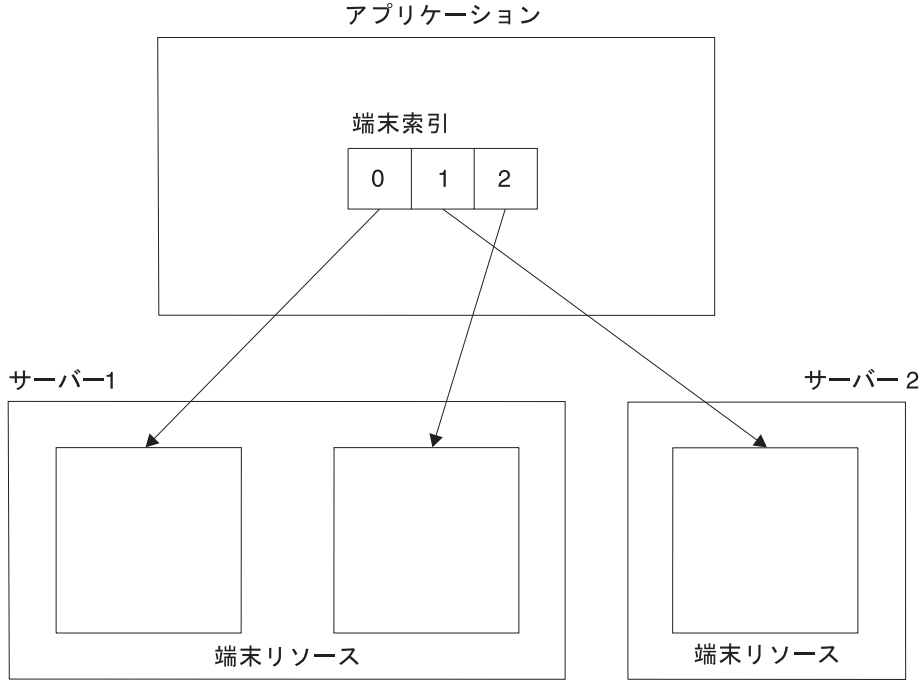

図 6. 端末索引の使用

端末属性

ほとんどの属性は、端末リソースをインストールするときにサーバーによって決定されます。サブセットは、**CICS_EpiAddTerminal** および **CICS_AddExTerminal** 関数に渡された **CICS_EpiDetails_t** 構造内のアプリケーションに戻されます。

属性によっては、アプリケーションによって決定されるものがあります。これらの属性は、 **CICS_EpiAddExTerminal** 関数に渡された **CICS_EpiAttributes_t** 構造に指定されます。たとえば、3270 のデータで使用する文字セットとエンコード体系は、 **CICS_EpiAttributes_t** 構造内の **CCSID** フィールドに指定されます。

タイムアウト

CICS_EpiAddTerminal 関数は同期的に実行されます。**CICS_EpiAddExTerminal** 関数は同期的に実行されることがあります。応答 (たとえば、端末リソースがインストールされたという応答) が使用可能になるまで、制御は呼び出しアプリケーションに戻されません。

応答時間は、以下のようないくつかの要因によって決まります。

- ネットワーク・パラメーターおよびトラフィック
- サーバー・パラメーターおよびトランザクション・ロード

応答を待つことができる時間の長さをアプリケーションで制限することができます。この制限は、**CICS_EpiAddExTerminal** 関数に渡された **CICS_EpiAttributes_t** 構造内の **InstallTimeOut** フィールドに指定されます。

クライアントが、指定時間間隔内にサーバーからの応答を受け取らないと、制御が呼び出しアプリケーションに戻され、戻りコードは CICS_EPI_ERR_RESPONSE_TIMEOUT に設定されます。

後で、端末リソースがサーバーにインストールされたことがクライアントに通知されると、クライアントはその端末リソースを削除します。この処置により、サーバーのネットワーク・トラフィックとトランザクション・ロードが増えることに注意してください。

端末リソースの削除

端末リソースが不要になったら、**CICS_EpiDelTerminal** または **CICS_EpiPurgeTerminal** のいずれかの関数を呼び出すことにより、その端末リソースを削除することができます。

CICS_EpiDelTerminal 関数は、制御した方法で端末リソースを削除したいときに使用します。この呼び出しは、端末リソースに対してトランザクションが実行されていない場合にのみ成功します。トランザクションが実行中であるか、または未処理のイベントがある場合は、この呼び出しはエラー・コードを戻します。

CICS_EpiPurgeTerminal 関数は、トランザクションが端末に対して実行されているか、あるいは端末リソースのための未処理イベントに対して実行されているかは問わず、端末リソースを削除しなければならばいときに使用されます。

端末リソースが削除されると、端末索引値が解放され、別の端末リソースが追加されたときに再使用可能になります。端末リソースが自動インストールされている場合は、サーバーがそれを削除します。

認証および許可

アプリケーションで端末リソースを定義したら、その端末リソースからトランザクションを開始できます。サーバーは、以下の操作を行う必要があります。

- 端末 ″オペレーター″ のユーザー ID とパスワードを認証する。
- 認証されたユーザー ID に基づき、各トランザクションを実行するために必要なリソースにアクセスする権限を付与する。

ユーザー ID とパスワードがサーバーによって認証される頻度は、**CICS_EpiAddExTerminal** 関数に渡された **CICS_EpiAttributes_t** 構造内の **SignOnCapability** フィールドに指定された別の属性によって異なります。

サインオン対応端末

端末リソースがサインオン対応としてインストールされている場合は、アプリケーションがサインオン・トランザクション (たとえば、 CICS 提供の CESN トランザクション) を開始しなければなりません。ユーザー ID とパスワードが 3270 データに組み込まれます。サインオン・トランザクションの前に開始されたトランザクションは、デフォルトのユーザー ID に権限が付与されて実行されます。サインオン・トランザクションの後で開始されたトランザクションは、認証されたユーザー ID に権限が付与されて実行されます。

端末をサインオン対応としてインストールすると、端末リソースがサーバー定義制限時間よりも長くアイドル状態になっている場合に、アプリケーションは、サインオフになっている ″オペレーター″ に対応できようになっていなければなりません。

サインオン非対応端末
端末リソースがサインオン非対応としてインストールされている場合は、端末リソースに対して開始された各トランザクションごとに、ユーザー ID とパスワードが認証されます。端末リソースのための初期ユーザー ID とパスワードは、 **CICS_EpiAddExTerminal** 関数に渡された **CICS_EpiAttributes_t** 構造内の **UserId** および **Password** フィールドに指定することができます。

ユーザー ID とパスワードは、 **CICS_EpiSetSecurity** 関数を呼び出すことにより、いつでも変更することができます。

ユーザー ID とパスワードが必要なのに、アプリケーションによって提供されていなければ、 CICS クライアントは、たとえば、ポップアップ・ウィンドウを使用して、ユーザー ID とパスワードを入手することができます (ただし、UNIX® オペレーティング・システムでは、ポップアップ・ウィンドウはサポートされていませんのでご注意ください)。

トランザクションの開始

アプリケーションで定義した端末リソースは、トランザクションを開始できます。この場合のトランザクションの開始は、CICS サーバー上では、実際の端末上でトランザクション・コードが入力されて AID キーが押された場合と区別されません。トランザクションを開始する場合には、**CICS_EpiStartTran** 関数を呼び出します。開始するトランザクションおよび関連データの指定では、以下の 2 つの方法のいずれかが使用できます。

1. 呼び出しの **TransId** パラメーターにトランザクション ID を指定し、さらに **Data** パラメーターにトランザクション・データを指定する。
2. トランザクション ID とトランザクション・データの両方を 3270 データ・ストリームに格納し、**呼び出しの**パラメーターとしてデータ・ストリームを指定する (DATA)。

イベントおよびコールバック

アプリケーションで定義した端末リソースには、アプリケーションの処理イベント以外に、CICS サーバーの処理によるイベントが発生する場合があります。アプリケーションからは、このようなイベントの発生は予期できませんが、適切なイベントの収集や処理はアプリケーション側で行う必要があります。発生したイベントは、EPI によって FIFO キューの中に保持されます。アプリケーションは **CICS_EpiGetEvent** 関数を実行し、これらのイベントを 1 つずつ収集することになります。呼び出しの際には、対象のイベントおよび

関連データの情報が、EPI によって **CICS_EpiEventData_t** 構造に格納されます。 FIFO キューにさらにイベントが存在するかどうかを示す情報も含まれます。

アプリケーション側では、これらのイベントを (他のアクティビティーも含めて) 同期させて処理することが可能です。以下の 3 つの同期方法があります。
- ポーリング
- ブロッキング
- コールバック通知

ポーリング
CICS_EpiGetEvent 呼び出しをポーリング・モードで行う場合には、**Wait** パラメーターに CICS_EPI_NOWAIT を指定します。収集待ち状態のイベントが存在しない場合は、関数はただちにエラー・コードを戻します。この方法は、単一ユーザー単一スレッド環境で使用します。この環境で動作するアプリケーションでは、ユーザー・アクティビティーに対してキーボードを、イベント・アクティビティーに対して EPI を、交互にポーリングする場合があるためです。

ブロッキング
CICS_EpiGetEvent 呼び出しをブロック・モードで行う場合には、**Wait** パラメーターに CICS_EPI_WAIT を指定します。待ち状態のイベントが存在しない場合には、イベントが使用可能になるまで関数は処理を待ち、戻りません。この方法は、マルチスレッド環境で使用します。このような環境では、他のスレッドをイベント処理専用にすることができます。また、この方法では、イベント情報を使えることがわかるので、コールバック処理の通知後に使用することもできます。

コールバック通知
端末リソースを **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しで定義する場合には、オプション・パラメーター **NotifyFn** を指定することにより、この端末リソースに対してイベントが生じるたびに EPI が呼び出すコールバック・ルーチンのアドレスを提供することができます。

注: コンパイラーによっては、コールバック・ルーチンをサポートしないものもあります。詳細については、使用するコンパイラーの関連資料を参照してください。

アプリケーションからコールバック・ルーチンを呼び出す場合には、処理を最小限にとどめ、EPI に処理が戻るまでは指定したルーチンでブロッキングを実行しないでください。コールバック・ルーチンから EPI 呼び出しを行うことは

できません。通知の受信後は、環境に応じた処理を行ってください。たとえばマルチスレッド環境では、セマフォーをポストすれば、コールバック・ルーチンから他のスレッドにイベント発生を知らせることができます。プレゼンテーション・マネージャー上では、メッセージをウィンドウにポストし、ウィンドウ・プロシージャーにイベント発生を知らせる方法があります。他の環境でも、対応する処理が使用できます。

コールバック・ルーチンの呼び出し時には、イベントの発生した端末リソースの端末索引のみが単一のパラメーターとして渡されます。したがって、同じコールバック・ルーチンを、さまざまな端末リソースに対して使用できます。

イベント処理

CICS_EpiGetEvent 関数から戻る時、**CICS_EpiEventData_t** 構造に、発生したイベントの詳細情報を格納します。この構造の **Event** フィールドでイベントが識別されます。このフィールドには、以下のいずれかの値が格納されます。

- CICS_EPI_EVENT_SEND
- CICS_EPI_EVENT_CONVERSE
- CICS_EPI_EVENT_END_TRAN
- CICS_EPI_EVENT_START_ATI
- CICS_EPI_EVENT_ADD_TERM
- CICS_EPI_EVENT_END_TERM

イベントには端末リソースの状態を示すものがあるので、アプリケーション側でのイベント処理はできるだけ迅速に行う必要があります。これらのイベントが処理されないと、EPI とアプリケーションの状態情報が一致しなくなる可能性があります。このため、アプリケーションが EPI 関数を実行するときに、予期せぬエラーが発生する場合があります。たとえばアプリケーションからトランザクションが開始できるのは、現在他のトランザクションを稼働していない端末リソースのみです。トランザクションの終了時には CICS_EPI_EVENT_END_TRAN イベントが発生します。アプリケーションと EPI は、このイベントによって、トランザクションを新たに開始できる状態になります。 CICS_EPI_EVENT_START_ATI イベントは、端末リソースで ATI トランザクションが開始され、この端末リソースで他のトランザクションが開始できない状態になったことを示します。アプリケーションが CICS_EPI_EVENT_START_ATI イベントの発生通知を受信しても、そのイベントの検索の前に、アプリケーションが **CICS_EpiStartTran** を呼び出すと、その **CICS_EpiStartTran** 呼び出しはエラーになります。この状態は、そのアプリケーションには、呼び出しが成功するように見える場合があります。

クライアント・アプリケーションがイベントまたはコールバックで起動されている場合は、 **CICS_EpiGetEvent** を出して関連するイベントを取得しなければなりません。特定のタイミング条件では、直前の **CICS_EpiGetEvent** から CICS_EPI_EVENT_START_ATI が既に通知されている場合があります。コールバック後に **CICS_EpiGetEvent** を出すと、 CICS_EPI_NOWAIT を **Wait** パラメーターに指定した場合は CICS_EPI_ERR_NO_EVENT を受け取り、 CICS_EPI_NOWAIT を **Wait** パラメーターに指定した場合は以降のイベントが受信されるまで待機する場合があります。これは、 CICS_EPI_EVENT_START_ATI の受信後に起こる場合があるので注意してください。

データの送受信

トランザクションから端末リソースへのデータ送信では、 EPI が CICS_EPI_EVENT_SEND または CICS_EPI_EVENT_CONVERSE のいずれかのイベントを生成します。

送信されるデータは、トランザクションからのデータ、またはエラー・メッセージなどのサーバーが生成するメッセージとなります。このようなメッセージについては、サーバーの関連資料を参照してください。 EPI アプリケーションは、データ・ストリームを分析して、エラーが起こったかどうかを確認しなければなりません。

CICS_EPI_EVENT_SEND イベントは、データ送信が行われ、応答が必要ないことを示します。通常 EXEC CICS SEND コマンドによって生成されますが、サーバーによっては EXEC CICS CONVERSE コマンドで生成される場合もあります。 (この場合には、CICS_EPI_EVENT_CONVERSE イベントが生成され、アプリケーションにサーバー上のトランザクションへのデータ・ストリームの返送が指示されます。)

CICS_EPI_EVENT_CONVERSE イベントは、応答が必要なことを示します。通常 EXEC CICS RECEIVE または EXEC CICS CONVERSE のいずれかのコマンドによって生成されます。アプリケーションは、**CICS_EpiReply** 呼び出しを出してこのイベントに応答し、応答データを提供します。**CICS_EpiReply** 関数は、CICS_EPI_EVENT_CONVERSE イベントへの応答にのみ使用します。それ以外の場合に使用するとエラーになります。

疑似会話の管理

CICS トランザクションは疑似会話モードで操作できます。端末リソースとサーバー間の会話は、複数のセグメントで構成されます。各セグメントが 1 トランザクションになります。各トランザクションは終了時に、次に端末リソース

から入力されるデータを処理するトランザクション名を指定します。(EXEC CICS RETURN が TRANSID オプションで実行されます。)アプリケーションは、CICS_EPI_EVENT_END_TRAN イベントによって、終了したトランザクションが次の入力を処理するトランザクションを指定しているかどうか、および指定されたトランザクションを確認できます。アプリケーションから、**CICS_EpiStartTran** を使用して、CICS_EPI_EVENT_END_TRAN イベントで指定されたトランザクションを開始します。指定以外のトランザクションの開始を要求しないでください。

EPI でのセキュリティー

EPI は、CICS クライアントとサーバーの間のシステム間会話のセットとしてインプリメントされます。たとえば、会話は以下のように割り振られます。

- インストールされている各端末リソースごと。 **CICS_EpiAddTerminal** および **CICS_EpiAddExTerminal** 関数を参照してください。
- 開始された各トランザクションごと。 **CICS_EpiStartTran** 関数を参照してください。
- サーバーから削除された各端末リソースごと。 **CICS_EpiDelTerminal** 関数を参照してください。

各会話ごとにユーザー ID とパスワードが必要になることがあります。この要件は、CICS クライアントと CICS サーバーの構成の仕方によって決まります。

クライアント・アプリケーションでは、**CICS_EpiAddExTerminal** 関数に渡された **CICS_EpiAttributes_t** 構造にユーザー ID とパスワードを指定することができます。これらの値は端末に固有なものであり、端末をインストールするために割り振られたシステム間会話のサーバー、およびインストールされている端末に関連する後続のシステム間会話のサーバーに渡されます。これらの値は、 **CICS_EpiSetSecurity** 関数によっていつでも変更することができます。

クライアント・アプリケーションは、ユーザー ID とパスワードを **CICS_SetDefaultSecurity** 関数に渡すことができます。これらの値はサーバー固有なものであり、端末に固有な値が使用できなくなったときに、いつでも使用することができます。これらの値は、 **CICS_SetDefaultSecurity** 関数によっていつでも変更することができます。

ユーザー ID とパスワードがシステム間会話に必要であるのに、**CICS_EpiAddExTerminal**、 **CICS_EpiSetSecurity**、および **CICS_SetDefaultSecurity** のいずれの関数でも設定されていない場合は、CICS クライアントは他の方法 (たとえば、ポップアップ・ウィンドウ) を使用

してこれらの値を決定することができます。(ポップアップ・ウィンドウは、UNIX® オペレーティング・システムではサポートされませんのでご注意ください。)

サインオン非対応端末
CICS クライアントによって決定されたユーザー ID は、サインオン非対応端末から開始された各トランザクションごとにサーバーによって認証されます。これらのトランザクションは、認証されたユーザー ID に権限が割り当てられてサーバーで実行されます。

ユーザー ID が CICS クライアントによって渡されないと、デフォルトのユーザー ID に権限が割り当てられて、トランザクションがサーバーで実行されます。

サインオン対応端末
CICS クライアントによって決定されたユーザー ID があれば、それは、サインオン対応端末から開始された各トランザクションごとに CICS サーバーによって認証されます。これらのトランザクションは、クライアント・アプリケーションによって決定されたユーザー ID、またはデフォルトのユーザー ID に権限が割り当てられて、CICS サーバーで実行されます。

端末で最初に開始されるトランザクションは、デフォルトのユーザー ID に権限が割り当てられて実行されます。

サインオン・トランザクションは、CICS 提供であるか、ユーザー作成です。いずれの場合も、ユーザー ID とパスワードはクライアント・アプリケーションによって決定され、 **CICS_EpiStartTran** または **CICS_EpiReply** 関数に渡すされた 3270 データ・ストリームに組み込まれます。ユーザー ID が認証済みであれば、端末から開始される後続のトランザクションは、認証されたユーザー ID に権限が割り当てられて CICS サーバーで実行されます。

クライアント・アプリケーションは、端末からサインオフ・トランザクションを開始することができます。事前定義された非活動期間後に、ユーザー ID をサーバーでサインオフすることができます。いずれの場合も、端末から開始される後続のトランザクションは、デフォルトのユーザー ID に割り当てられた権限で実行されます。

注: CICS サーバーは、サインオン対応端末をインストールしようとする試みを拒否することがあります。このような状態は、すべての認証がオペレーティング・システムによって管理される CICS (OS/400® 版) などで見られます。

OS/390 サーバーのセキュリティー

クライアント・アプリケーションによってインストールされた、サインオン可能な端末で開始されたトランザクションに対して、サーバーで実行されるセキュリティー検査は、そのクライアントを示す接続に対する ATTACHSEC オプションの指定内容には依存していません。セキュリティー検査は、このオプションではなく、端末を使用中に、ユーザーがサインオンしているかどうかに依存します。

ユーザーがサインオンしていない場合には、端末をインストールしたクライアントは、SIT のサーバーに対して定義されたデフォルト・ユーザーに関連付けされています。トランザクションが実行されると、このデフォルト・ユーザーに対してセキュリティー検査が実行されます。検査は、その接続に関連付けられたユーザー ID に対しても実行され、そのクライアント自体にそのリソースへのアクセス権限があるかどうかが判別されます。

ユーザーがサインオン中は、認証されたそのユーザー ID に端末が関連付けされます。リソースにアクセスするトランザクションに対しては、その接続に関連付けされているユーザー ID と、サインオン・ユーザーのユーザー ID に対するセキュリティー検査が行われます。

サインオン可能端末を使用している場合には、サーバー接続定義上の Usedfltuser パラメーターを Yes に設定し、サインオン不可能端末を使用している場合には、No にすることをお勧めします。

EPI 定数およびデータ構造

このセクションでは、EPI を使用するために必要な定数と構造について説明します。これらの定数および構造体は、84ページの『EPI 関数』の中で言及されます。

EPI 定数

以下に挙げた定数は、本章の EPI 構造、機能、およびイベントの説明で、記号名で参照されています。これらの定数の値も参考として挙げていますが、実際のプログラミングの際には、その言語用に EPI が提供している記号名を使用してください。

フィールド長

- CICS_EPI_SYSTEM_MAX (8)
- CICS_EPI_DESCRIPTION_MAX (60)
- CICS_EPI_NETNAME_MAX (8)
- CICS_EPI_TRANSID_MAX (4)

- CICS_EPI_ABEND_MAX (4)
- CICS_EPI_DEVTYPE_MAX (16)
- CICS_EPI_ERROR_MAX (60)
- CICS_EPI_PASSWORD_MAX (10)
- CICS_EPI_USERID_MAX (10)
- CICS_EPI_MAPNAME_MAX (7)
- CICS_EPI_MAPSETNAME_MAX (8)
- CICS_EPI_TERMID_MAX (4)

TermIndex 関連定数

- CICS_EPI_TERM_INDEX_NONE 0xFFFF

バージョン番号 (83ページの『EPI のバージョン』参照)

- CICS_EPI_VERSION_100
- CICS_EPI_VERSION_101
- CICS_EPI_VERSION_200.

EPI データ構造

EPI で使用できる構造を、以下に示します。

- **CICS_EpiSystem_t**
- **CICS_EpiAttributes_t**
- **CICS_EpiDetails_t**
- **CICS_EpiEventData_t**
- **CICS_EpiSysError_t**

これらの構造のストリング・フィールドはすべてヌルで終了します。

CICS_EpiSystem_t

目的: **CICS_EpiSystem_t** 構造には、CICS サーバーの名前と説明が格納されます。**CICS_EpiListSystems** 関数で、この構造の配列が戻されます。

フィールド:

SystemName

CICS サーバーを命名するストリングです。 **CICS_EpiAddTerminal** および **CICS_EpiAddExTerminal** 関数にパラメーターとして渡して、端末リソースをインストールする CICS サーバーを識別することができます。名前が CICS_EPI_SYSTEM_MAX 文字よりも短い場合は、CICS_EPI_SYSTEM_MAX + 1 の長さになるまでヌルを埋め込む必要があります。

Description

サーバーについての短い説明が格納されるストリングです。説明データの文字数が CICS_EPI_DESCRIPTION_MAX の値より小さい場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_DESCRIPTION_MAX + 1 になります。

CICS_EpiAttributes_t

目的: **CICS_EpiAttributes_t** 構造には、 **CICS_EpiAddExTerminal** 関数によってインストールされた端末リソースに関連づけられる属性に関する情報が格納されています。

データ構造は CICS_EPI_VERSION_200 に対してのみサポートされます。

フィールド:

EpiAddType
アプリケーションが、端末インストール要求の完了まで待機する準備が整っているかどうかを示します。以下のいずれかの値を指定します。

CICS_EPI_ADD_ASYNC
端末リソースのインストール要求が受け入れられると、呼び出しアプリケーションは再度制御を取得し、このとき、パラメーターの妥当性検査が行われます。

パラメーターが正しければ、端末のインストール要求が完了したときに、CICS_EPI_EVENT_ADD_TERM イベントが生成されます。

TermIndex が戻されて、**CICS_EpiGetEvent** 関数で使用されます。

CICS_EPI_ADD_SYNC
端末リソースのインストール要求が完了すると、呼び出しアプリケーションは再度制御を取得します。戻り情報は、すぐに使用できます。

InstallTimeOut
0 から 3600 までの範囲の値。端末リソースのインストールに使用できる最大時間 (秒単位)。 0 の値は、制限がないことを示します。

この範囲を超えた値を指定すると、3600 の値が使用されます。

ReadTimeOut
0 から 3600 までの範囲の値。端末リソースのための CICS_EPI_EVENT_CONVERSE イベントを通知してから、次に **CICS_EpiReply** を呼び出すまでの間の最大時間 (秒単位) を指定します。 0 の値は、制限がないことを示します。

この範囲を超えた値を指定すると、3600 の値が使用されます。

タイムアウトが発生すると、会話は異常終了します。この結果、CICS_EPI_EVENT_END_TRAN イベントが生成されます。

EndReason フィールドは CICS_EPI_READTIMEOUT_EXPIRED に設定されます。 **AbendCode** フィールドは設定されません。

SignonCapability
アプリケーションが、端末リソースから、サーバー提供のサインオンおよびサインオフ・トランザクションを開始できるかどうかを示します。以下のいずれかの値を指定します。

CICS_EPI_SIGNON_CAPABLE
端末リソースは、サインオン対応としてインストールされます。

CICS_EPI_SIGNON_INCAPABLE
端末は、サインオン対応としてインストールされます。

CCSId 1 から 65536 までの範囲の値。クライアント・アプリケーションが、端末リソースと CICS トランザクションの間でやり取りされたデータに使用したコード化図形文字セットを識別するコード化文字セット ID (CCSID) を指定します。

0 の値は、デフォルトの CCSID が使用されることを示します。

いろいろな文字セットの CCSID 値の詳細は、「CICS クライアント管理の手引き」の付録を参照してください。

UserId
端末リソースと関連付けるユーザー ID を指定するストリングです。ユーザー ID が CICS_EPI_USERID_MAX よりも短い場合は、CICS_EPI_USERID_MAX+1 の長さになるまでヌルで埋め込む必要があります。

Password
端末リソースと関連付けるパスワードを指定するストリングです。 パスワードが CICS_EPI_PASSWORD_MAX 文字よりも短い場合は、CICS_EPI_PASSWORD_MAX+1 の長さになるまでヌルで埋め込む必要があります。

CICS_EpiDetails_t

目的: **CICS_EpiDetails_t** 構造には、 **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 関数によってインストールされた端末リソースに関する情報が格納されています。

フィールド:

NetName
端末リソースの VTAM® 式のネット名を指定するストリングです。ネット名の文字数が、CICS_EPI_NETNAME_MAX の値より小さい場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_NETNAME_MAX + 1 になります。

NumLines
端末リソースがサポートする行数を格納します。

NumColumns
端末リソースがサポートする桁数を格納します。

MaxData
CICS_EpiStartTran 呼び出しまたは **CICS_EpiReply** 呼び出しによって、 CICS トランザクションからこの端末リソースに送信することができるデータの最大サイズ、およびこの端末リソースから CICS トランザクションに送信することができるデータの最大サイズです。

最大サイズは、端末リソースをサーバーにインストールした **CICS_EpiAddTerminal** 呼び出しの **DevType** パラメーターによって指定されたモデル端末定義で定義することができます。この値がモデル端末定義で指定されていない場合あるいは指定できない場合は、デフォルト値の 12000 が想定されます。

ErrLastLine
1: 端末リソースのエラー・メッセージを最後の行に表示する場合。
0: それ以外の場合。

ErrIntensify
1: 端末リソースのエラー・メッセージ表示を強調する場合。
0: それ以外の場合。

ErrColor
エラー・メッセージの表示色を定義する 3270 属性を格納します。

ErrHilight
エラー・メッセージの強調表示を定義する 3270 属性を格納します。

Hilight
1: 端末リソースが拡張強調表示をサポートしている場合。
0: それ以外の場合。

Color 1: 端末リソースがカラー表示をサポートしている場合。
0: それ以外の場合。

System
端末リソースがインストールされているサーバーの名前を指定するストリングです。名前が CICS_EPI_SYSTEM_MAX 文字よりも短い場合は、 CICS_EPI_SYSTEM_MAX + 1 の長さになるまでヌルを埋め込む必要があります。

このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

TermId
端末リソースの名前を指定するストリングです。名前が CICS_EPI_TERMID_MAX 文字よりも短い場合は、 CICS_EPI_TERMID_MAX + 1 の長さになるまでヌルを埋め込む必要があります。

このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

SignonCapability
サーバーによって端末リソースに割り当てられたサインオン機能。

CICS_EPI_SIGNON_CAPABLE
アプリケーションが、端末リソースから、サーバー提供のサインオンおよびサインオフ・トランザクションを開始できるかどうかを示します。

CICS_EPI_SIGNON_INCAPABLE
アプリケーションが、端末リソースから、サーバー提供のサインオンおよびサインオフ・トランザクションを開始できるかどうかを示します。

CICS_EPI_SIGNON_UNKNOWN
端末リソースを追加するために **CICS_EpiAddTerminal** 関数が使用されたかどうかを示します。 (端末リソースを追加するために **CICS_EpiAddExTerminal** 関数が使用され、前提変更がサーバーに適用されていない場合にも、この値が戻されます。)

このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EpiEventData_t

目的: **CICS_EpiEventData_t** 構造には、端末に関連するイベントの詳細情報が保持されます。すべてのイベントで全フィールドが使用されるわけではありません。使用されないフィールドにはヌルが格納されます。
CICS_EpiGetEvent で、この構造データが戻されます。

フィールド:

TermIndex
発生したイベントの対象端末リソースの端末索引を格納します。

Event 124ページの『EPI イベント』にリストされたイベント・コードのいずれかを格納します。このイベント・コードで、発生したイベントが識別されます。

EndReason
CICS_EPI_EVENT_END_TERM または CICS_EPI_EVENT_END_TRAN イベントが発生した場合に、終了理由が格納されます。

TransId
トランザクション名が格納されるストリングです。トランザクション名の文字数が、CICS_EPI_TRANSID_MAX の値より小さい場合は、残りにヌルが格納されます。ストリングはヌルで終了します。

Reserved1
予約フィールド。

CICS ユニバーサル・クライアントのバージョン 3.1 より前のバージョンでは、このフィールドは **AbendCode** と呼ばれていました。

Data イベントに関連する端末データ・ストリームで更新されるバッファーを指すポインターです。

入力時に **Data** パラメーターは、非同期的に追加する端末に関する最初の **CICS_EpiGetEvent** 呼び出し時に、 **CICS_EpiDetails_t** 構造を指すように設定する必要があります。詳細構造は、**CICS_EpiGetEvent** から戻ったときに更新されます。

Size **Data** でアドレスするバッファーの最大サイズです。
CICS_EpiGetEvent 呼び出しの結果として、このストリングに実際のデータ長が格納されます。

EndReturnCode
CICS_EPI_*returncode* を含むストリング。

| このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

MapName

イベントが CICS_EPI_EVENT_SEND または CICS_EPI_EVENT_CONVERSE である場合に、端末リソースで処理した SEND MAP コマンドの MAP オプションで最後に参照されたマップの名前を指定するストリングです。端末リソースが BMS によってサポートされていないか、もしくはサーバーが送信マップのレコードを持っていない場合は、戻された値はスペースになっています。名前が CICS_EPI_MAPNAME_MAX 文字よりも短い場合は、この長さになるまでスペースが埋め込まれ、その後に単一のヌル文字が入れられます。

このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

MapSetName

イベントが CICS_EPI_EVENT_SEND または CICS_EPI_EVENT_CONVERSE である場合に、端末リソースで処理した SEND MAPSET コマンドの MAPSET オプションで最後に参照されたマップ・セットの名前を指定するストリングです。最後の要求に MAPSET オプションが指定されていない場合は、 BMS はマップ名をマップ・セット名として使用しています。いずれの場合も、使用されたマップ・セット名には端末接尾部が付けられています。端末リソースが BMS によってサポートされていないか、もしくはサーバーが送信マップ・セットのレコードを持っていない場合は、戻された値はスペースになっています。名前が CICS_EPI_MAPSETNAME_MAX 文字よりも短い場合は、この長さになるまでスペースが埋め込まれ、その後に単一のヌル文字が入れられます。

このフィールドは CICS_EPI_VERSION_200 に対してのみサポートされます。

注: **Data** および **Size** の 2 つのフィールドは、 **CICS_EpiGetEvent** 呼び出しの前に指定してください。

CICS_EpiSysError_t

目的: **CICS_EpiSysError_t** 構造には、システム・エラー情報が格納されます。**CICS_EpiGetSysError** の戻りデータです。

このデータ構造は CICS_EPI_VERSION_200 に対してはサポートされません。

すべてのエラー情報はクライアント・ログ・ファイルに書き込まれます。

フィールド:

Cause 最後に発生したエラーの原因を示す値が格納されます。この値は操作環境によって異なります。

Value 最後に発生したエラーの特性を示す値です。この値は操作環境によって異なります。

Msg 最後に発生したエラーを説明するテキスト・メッセージが格納されます。この説明は、操作環境によって異なります。メッセージの文字数が CICS_EPI_ERROR_MAX の長さに満たない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_ERROR_MAX + 1 になります。

EPI のバージョン

以下の EPI 関数の説明では、EPI の 3 つのバージョンを取り上げています。使用しているクライアントまたはサーバーがサポートしている EPI のバージョンについては、クライアントまたはサーバーの関連資料を参照してください。使用する EPI のバージョンは、**CICS_EpiInitialize** 関数の実行時に **Version** パラメーターで指定します。

以下の各セクションで説明する関数は、CICS_EPI_VERSION_200 までのバージョンの関数です。以下の関数とデータ構造は、CICS_EPI_VERSION_200 に対してのみサポートされます。

- **CICS_EpiAddExTerminal**
- **CICS_EpiPurgeTerminal**
- **CICS_EpiSetSecurity**
- **CICS_EpiAttributes_t**
- **CICS_EpiDetails_t** データ構造内の **MapName**、**MapSetName**、**System**、**TermID**、および **SignonCapability** フィールド

EPI バージョン

CICS_EPI_VERSION_200 でサポートされているものは、以下のとおりです。

- **CICS_EpiGetSysError**
- **CICS_EpiSysError_t**

さらに、CICS_EPI_VERSION_100 で EPI を初期化する場合は、以下の制限が適用されます。

- **CICS_EpiInquireSystem** 関数はサポートされていません。
- EPI_ERR_IN_CALLBACK、EPI_ERR_NULL_PARM、および EPI_ERR_SECURITY の各戻りコードはサポートされていません。その代り EPI_ERR_FAILED が戻されます。
- デフォルト・サーバーの検索機能がサポートされていないため、**CICS_EpiAddTerminal** 関数の **System** パラメーターの値に、ヌル・ストリングを指定することはできません。(ヌル・ストリングを指定すると、CICS_EPI_ERR_SYSTEM が戻されます。)

EPI 関数

このセクションでは、アプリケーション・プログラムから呼び出すことができる、以下の EPI 関数について説明します。

- **CICS_EpiInitialize**
- **CICS_EpiTerminate**
- **CICS_EpiListSystems**
- **CICS_EpiAddTerminal**
- **CICS_EpiAddExTerminal**
- **CICS_EpiInquireSystem**
- **CICS_EpiDelTerminal**
- **CICS_EpiPurgeTerminal**
- **CICS_EpiSetSecurity**
- **CICS_EpiStartTran**
- **CICS_EpiReply**
- **CICS_EpiATIState**
- **CICS_EpiSenseCode**
- **CICS_EpiGetEvent**
- **CICS_EpiGetSysError**

85ページの表3 は、インターフェースの関数、各関数に渡されるパラメーター、および各関数からの戻りコードを示しています。

Windows® オペレーティング・システムの場合、実際の戻りコード値を記号名にマッピングしたものが、以下のファイルに含まれています。

C ¥include¥cics_eci.h

UNIX® オペレーティング・システムの場合は、以下のファイルです。

C /include/cics_eci.h

表 3. EPI 関数の要約

関数名	パラメーター	戻りコード: CICS_EPI_
CICS_EpiInitialize	**Version**	ERR_FAILED ERR_IS_INIT ERR_VERSION NORMAL
CICS_EpiTerminate	なし	ERR_FAILED ERR_NOT_INIT ERR_IN_CALLBACK NORMAL
CICS_EpiListSystems	**NameSpace** **Systems** **List**	ERR_FAILED ERR_MORE_SYSTEMS ERR_NO_SYSTEMS ERR_NOT_INIT ERR_NULL_PARM ERR_IN_CALLBACK NORMAL

表 3. EPI 関数の要約 (続き)

関数名	パラメーター	戻りコード: CICS_EPI_
CICS_EpiAddTerminal	**NameSpace** **System** **Netname** **DevType** **NotifyFn** **Details** **TermIndex**	ERR_FAILED ERR_MAX_TERMS ERR_NOT_INIT ERR_SYSTEM ERR_SECURITY ERR_NULL_PARM ERR_IN_CALLBACK ERR_SERVER_DOWN NORMAL CICS_EPI_VERSION_200 のみの場合: ERR_TERMID_INVALID ERR_MODELID_INVALID ERR_NOT_3270_DEVICE ERR_ALREADY_INSTALLED ERR_SERVER_BUSY ERR_RESOURCE_SHORTAGE ERR_MAX_SESSIONS ERR_MAX_SYSTEMS

表 3. EPI 関数の要約 (続き)

関数名	パラメーター	戻りコード: CICS_EPI_
CICS_EpiAddExTerminal	**System** **Netname** **DevType** **NotifyFn** **Details** **TermIndex** **Attributes**	ERR_FAILED ERR_NOT_INIT ERR_SYSTEM ERR_SECURITY ERR_VERSION ERR_IN_CALLBACK ERR_SERVER_DOWN ERR_RESPONSE_TIMEOUT ERR_SIGNON_NOT_POSS ERR_PASSWORD_INVALID ERR_ADDTYPE_INVALID ERR_SIGNONCAP_INVALID ERR_USERID_INVALID ERR_TERMID_INVALID ERR_MODELID_INVALID ERR_NOT_3270_DEVICE ERR_ALREADY_INSTALLED ERR_CCSID_INVALID ERR_SERVER_BUSY ERR_RESOURCE_SHORTAGE ERR_MAX_SESSIONS ERR_MAX_SYSTEMS NORMAL
CICS_EpiInquireSystem	**TermIndex** **System**	ERR_BAD_INDEX ERR_FAILED ERR_NOT_INIT ERR_NULL_PARM ERR_IN_CALLBACK NORMAL
CICS_EpiDelTerminal	**TermIndex**	ERR_BAD_INDEX ERR_FAILED ERR_NOT_INIT ERR_TRAN_ACTIVE ERR_IN_CALLBACK NORMAL

表 3. EPI 関数の要約 (続き)

関数名	パラメーター	戻りコード: CICS_EPI_
CICS_EpiPurgeTerminal	**TermIndex**	ERR_BAD_INDEX ERR_FAILED ERR_NOT_INIT ERR_IN_CALLBACK ERR_VERSION NORMAL
CICS_EpiSetSecurity	**TermIndex** **UserId** **Password**	ERR_NOT_INIT ERR_BAD_INDEX ERR_IN_CALLBACK ERR_SYSTEM_ERROR ERR_VERSION ERR_PASSWORD_INVALID ERR_USERID_INVALID ERR_NULL_PASSWORD ERR_NULL_USERID NORMAL
CICS_EpiStartTran	**TermIndex** **TransId** **Data** **Size**	ERR_ATI_ACTIVE ERR_BAD_INDEX ERR_FAILED ERR_NO_DATA ERR_NOT_INIT ERR_TTI_ACTIVE ERR_IN_CALLBACK ERR_SERVER_DOWN ERR_RESOURCE_SHORTAGE (v 200 のみ) ERR_MAX_SESSIONS (v 200 のみ) NORMAL
CICS_EpiReply	**TermIndex** **Data** **Size**	ERR_BAD_INDEX ERR_FAILED ERR_NO_CONVERSE ERR_NO_DATA ERR_NOT_INIT ERR_IN_CALLBACK ERR_ABENDED (v 200 のみ) ERR_SERVER_DOWN NORMAL

表 3. EPI 関数の要約 (続き)

関数名	パラメーター	戻りコード: CICS_EPI_
CICS_EpiATIState	**TermIndex** **ATIState**	ERR_ATI_STATE ERR_BAD_INDEX ERR_FAILED ERR_NOT_INIT ERR_IN_CALLBACK ERR_NULL_PARAM NORMAL
CICS_EpiSenseCode	**TermIndex** **SenseCode**	ERR_BAD_INDEX ERR_FAILED ERR_NOT_INIT ERR_SENSE_CODE ERR_IN_CALLBACK ERR_VERSION NORMAL
CICS_EpiGetEvent	**TermIndex** **Wait**	ERR_BAD_INDEX ERR_FAILED ERR_MORE_DATA ERR_MORE_EVENTS ERR_NO_EVENT ERR_NOT_INIT ERR_WAIT ERR_NULL_PARAM ERR_IN_CALLBACK NORMAL
CICS_GetSysError	**TermIndex** **SysErr**	ERR_NOT_INIT ERR_BAD_INDEX ERR_FAILED ERR_NULL_PARAM ERR_VERSION NORMAL

パラメーターのタイプおよび使用法、関数が使用するデータ構造、および戻りコードの意味については、各関数の定義を参照してください。

CICS_EpiInitialize

CICS_EpiInitialize	Version

目的

CICS_EpiInitialize 関数は EPI を初期化します。同じアプリケーションから、この関数を呼び出す前に、他の EPI 関数を実行することはできません。

パラメーター

Version

アプリケーションに対応する EPI のバージョンを示します。このパラメーターを指定することによって、EPI がアップグレードされても、アプリケーションの互換性が保証されます。ここでの記述は CICS_EPI_VERSION_200 用です。 (詳細については、83ページの『EPI のバージョン』を参照してください。)

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_FAILED

関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_IS_INIT

EPI は初期化済みです。

CICS_EPI_ERR_VERSION

EPI のバージョンが、指定値とは異なります。

CICS_EPI_NORMAL

関数の処理が正常に完了しました。

CICS_EpiTerminate

CICS_EpiTerminate

目的

CICS_EpiTerminate 関数は、アプリケーションでの EPI の使用を終了します。通常、アプリケーションの終了の直前に実行します。この呼び出しが完了すると、そのアプリケーションは、 (**CICS_EpiInitialize** 以外の) すべての EPI 呼び出しができなくなります。

アプリケーションが終了前に、**CICS_EpiDelTerminal** 呼び出しを行い、端末リソースが確実に削除されるようにしてください。

パラメーター

指定しません。

戻りコード

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_TTI_ACTIVE
EPI から開始されたトランザクションがまだアクティブのままか、CICS_EpiGetEvent 呼び出しがまだ未解決のままです。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiListSystems

CICS_EpiListSystems	NameSpace Systems List

目的

CICS_EpiListSystems 関数は、 EPI 要求の候補である CICS サーバーのリストを戻します。リストに含まれる各サーバーには、クライアントとの間に通信リンクが存在しないもの、および要求の処理に使用可能でないものが含まれる可能性があります。

リストは、各 CICS サーバーにつき 1 つのエレメントが割り当てられたシステム情報構造の配列として戻されます。この構造の内容については、75ページの『CICS_EpiSystem_t』を参照してください。

EPI アプリケーションから **CICS_EpiInitialize** 呼び出しを行った後に、この関数を呼び出し、使用可能な CICS サーバーを確認してください。

パラメーター

NameSpace
将来の使用のために予約されたポインターです。ヌル・ポインターであることを確認してください。

Systems
数値を指すポインターです。関数の呼び出し時に、この数によって、**List** パラメーターで指定された配列のエレメントの数を指定します。ここで指定する値は、EPI が結果を格納するために使用することのできるストレージの量を正確に反映するものでなければなりません。関数が戻されると、このパラメーターには実際に確認されたサーバーの数が格納されます。

このパラメーターは、EPI が入出力の両方に使用します。

List 関数の戻りデータが格納される **CICS_EpiSystem_t** 構造の配列です。アプリケーションで配列用ストレージを提供するので、**Systems** パラメーターで配列のエレメント数をセットする必要があります。リストの最初の要素には、デフォルトのサーバーの名前が格納されます。デフォルトのシステムの定義は、オペレーティング・システムによって異なります。

EPI はこのパラメーターを出力のみで使用します。

戻りコード

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_MORE_SYSTEMS
List 配列のサイズが不足しており、見つかったすべての CICS サーバーの詳細情報を格納することができません。指定したすべての配列にデータが格納され、見つかったシステムの数に合わせて **Systems** パラメーターが更新されました。適切なサイズの配列を再割り当てし、関数を再実行してください。

CICS_EPI_ERR_NO_SYSTEMS
CICS サーバーが見つかりません。**Systems** の戻り値は 0 になります。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_NULL_PARM
Systems がヌル・ポインターです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。**Systems** パラメーター入力値で指定した数以下のシステムが見つかりました。

CICS_EpiAddTerminal

CICS_EpiAddTerminal	**NameSpace** **System** **NetName** **DevType** **NotifyFn** **Details** **TermIndex**

目的

CICS_EpiAddTerminal 関数は、アプリケーションで使用する新規の端末リソースのインストール、または既存の端末リソースの予約を行います。端末索引が戻されます。この端末索引は、以降の EPI 呼び出しで端末リソースの識別に使用します。さらに、**CICS_EpiDetails_t** データ構造に定義された情報を提供します。

この操作で追加できる端末の数には、以下のような制限があります。最大数は、クライアント・システムで使用できるリソースによって異なります。

注: **CICS_EpiAddTerminal** 関数は、サインオン機能が端末リソースがインストールされたサーバーに依存している端末リソースを追加します。たとえば、端末リソースは、CICS Transaction Server (OS/390 版) サーバーではサインオン機能はありません。

パラメーター

NameSpace

将来の使用のために予約されたポインターです。ヌル・ポインターであることを確認してください。

System

端末リソースのインストールまたは予約を行うサーバーの名前を格納する、ヌルで終了するストリングを指すポインターです。名前の文字数が、CICS_EPI_SYSTEM_MAX の値より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_SYSTEM_MAX + 1 になります。

ストリングがヌル・ストリングの場合には、EPI が現行のデフォルト・サーバーを選択します。選択されたサーバーの名前を得る場合には、

CICS_EpiInquireSystem を使用します。(ヌル・ストリングの指定は、EPI が EPI_VERSION_101 以上で初期化した場合にのみ有効です。)

EPI は、このパラメーターを入力用のみに使用します。

NetName

インストールされる、または予約される端末リソースの名前、あるいはヌルを指定する、ヌル文字で終了するストリングを指すポインターです。この名前の解釈は、サーバーによって異なります。

ストリングの文字数が、CICS_EPI_NETNAME_MAX より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_NETNAME_MAX + 1 になります。

このストリングは、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。 EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

NetName の使用方法は、以下のとおりです。

1. **NetName** に指定した名前がサーバーの既存の端末リソースの名前と一致すると、サーバーはその端末リソースを予約しようとします。
2. 指定した名前がサーバー内の既存の端末リソースの名前と一致していない場合には、 サーバーは、以下で説明する **DevType** パラメーターで指定したモデル端末定義を使用して端末リソースをインストールし、この端末リソースにその名前を付けます。(**DevType** がヌル・ポインターの場合には、CICS_EPI_VERSION_200 またはそれ以降では、CICS_EPI_ERR_TERMID_INVALID が戻されます。これ以外の場合には、CICS_EPI_ERR_FAILED が戻されます。)
3. **NetName** がヌル・ポインターの場合には、 **DevType** に指定されたモデル端末定義によって端末リソースがインストールされます。**DevType** がヌル・ポインターの場合には、選択された端末タイプを予測することができません。このため、**DevType** を使用して整

合性がある結果を得るようにしてください。端末リソースの名前は、**CICS_EpiDetails_t** 構造の **NetName** フィールドに戻されます。

EPI は、このパラメーターを入力用のみに使用します。

DevType

サーバーによって端末リソース定義を生成するモデル端末定義の選択に使用されるヌルで終了するストリングを指すポインター、またはヌル・ポインターです。

ストリングの文字数が、CICS_EPI_DEVTYPE_MAX の値より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_DEVTYPE_MAX + 1 になります。

このストリングは、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。 EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

EPI は、このパラメーターを入力用のみに使用します。

NotifyFn

端末リソースに対するイベントの発生時、たとえば ATI 要求の到着などで呼び出されるコールバック・ルーチンを指すポインターです。コールバック・ルーチンが必要ない場合には、このパラメーターにヌルを設定します。

EPI は、このパラメーターを入力用のみに使用します。

Details

インストールまたは予約された端末リソースの詳細情報が、戻り時に格納される **CICS_EpiDetails_t** 構造を指すポインターです。

EPI は、この構造の各フィールドを出力用のみに使用します。

TermIndex

直前にインストールまたは予約された端末リソースの端末索引を指すポインターです。関数が戻す端末索引は、以降の EPI 関数の呼び出し

で、関数の処理対象の端末リソースの識別に使用します。端末索引は整数で、0 から順に各端末リソースに割り当てられます。

EPI はこのパラメーターを出力のみで使用します。

戻りコード

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_SYSTEM
指定されたサーバーが、クライアント側で認識できません。

CICS_EPI_ERR_SECURITY
サーバーが、セキュリティー上の理由で、要求を拒否しました。

CICS_EPI_ERR_NULL_PARM
TermIndex がヌル・ポインターです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_SERVER_DOWN
サーバーが停止したため、関数が失敗しました。

CICS_EPI_ERR_TERMID_INVALID
無効な TermId が指定されたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_MODELID_INVALID
無効なモデル端末定義が指定されたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_NOT_3270_DEVICE
指定された装置タイプが 3270 装置のものでなかったため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_ALREADY_INSTALLED
端末がすでにインストールされていたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_SERVER_BUSY
サーバーが使用中であるため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_RESOURCE_SHORTAGE
CICS サーバーまたはクライアントに、端末のインストールを完了させるためのリソースが不足しています。

CICS_EPI_ERR_MAX_SESSIONS
この要求を満足する十分な通信リソースがありません。

CICS_EPI_ERR_MAX_SYSTEMS
ユーザーの構成で許可されている数を超えるサーバーへの接続を開始しようとしました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiAddExTerminal

CICS_EpiAddExTerminal	System
	NetName
	DevType
	NotifyFn
	Details
	TermIndex
	Attributes

目的

EPI バージョン 2 以上に対してのみサポートされます。

CICS_EpiAddExTerminal 関数は、アプリケーションで使用する新規の端末リソースのインストール、または既存の端末リソースの予約を行います。端末索引が戻されます。この端末索引は、以降のすべての EPI 呼び出しで端末リソースの識別に使用します。さらに、**CICS_EpiDetails_t** データ構造に定義された情報を提供します。

一部の属性 (たとえば、3270 データで使用する文字セットやエンコード体系およびサインオン機能) は、アプリケーションによって決定することができます。これらの属性は、**CICS_EpiAttributes_t** 構造の、**CCSID** および **SignonCapability** フィールドに指定されています。

パラメーター

System

端末リソースのインストールまたは予約を行うサーバーの名前を格納する、ヌルで終了するストリングを指すポインターです。名前の文字数が、CICS_EPI_SYSTEM_MAX の値より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_SYSTEM_MAX + 1 になります。

ストリングがヌル・ストリングの場合には、EPI が現行の省力時サーバーを選択します。選択されたサーバーの名前を得る場合には、**CICS_EpiInquireSystem** を使用します。

EPI は、このパラメーターを入力用のみに使用します。

NetName

インストールされる、または予約される端末リソースの名前、あるいはヌルを指定する、ヌル文字で終了するストリングを指すポインターです。この名前の解釈は、サーバーによって異なります。

ストリングの文字数が、CICS_EPI_NETNAME_MAX より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_NETNAME_MAX + 1 になります。

このストリングは、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。 EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

NetName の使用方法は、以下のとおりです。

1. **NetName** を使用して指定した名前がサーバーの既存の端末リソースの名前と一致すると、サーバーはその端末リソースを予約しようとします。
2. 名前が指定されているにもかかわらず、その名前がサーバー内の既存の端末リソース名と一致しない場合には、サーバーは、以下で説明する **DevType** パラメーターで指定したモデル端末定義を使用して端末リソースをインストールし、この端末リソースにその入力名を付けます。(**DevType** がヌル・ポインターの場合には、CICS_EPI_VERSION_200 またはそれ以降では、CICS_EPI_ERR_TERMID_INVALID が戻されます。これ以外の場合には、CICS_EPI_ERR_FAILED が戻されます。)
3. **NetName** がヌル・ポインターの場合には、 **DevType** に指定されたモデル端末定義によって端末リソースがインストールされます。**DevType** がヌル・ポインターの場合には、選択された端末タイプを予測することができません。このため、**DevType** を使用して整合性がある結果を得るようにしてください。端末リソースの名前は、**CICS_EpiDetails_t** 構造の **NetName** フィールドに戻されます。

EPI は、このパラメーターを入力用のみに使用します。

DevType

サーバーによって端末リソース定義を生成するモデル端末定義の選択に使用されるヌルで終了するストリングを指すポインター、またはヌル・ポインターです。

ストリングの文字数が、CICS_EPI_DEVTYPE_MAX の値より少ない場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_EPI_DEVTYPE_MAX + 1 になります。

このストリングは、大文字に変換されずにサーバーに伝送されます。

使用される文字は、伝送前にクライアントのコード・ページから EBCDIC コード・ページに変換されます。サーバーが ASCII コード・ページを使用する場合は、文字が再変換されます。これらの変換において不変であることが保証されている文字は、大文字の A から Z まで、および数字の 0 から 9 までのみです。 EBCDIC サーバー (カタカナおよびヘブライ語文字セット A) の中には英小文字の標準表示を使用しないものもあるので、そのようなサーバーと通信する場合は注意して使用しなければなりません。

EPI は、このパラメーターを入力用のみに使用します。

NotifyFn

端末リソースに対するイベントの発生時、たとえば ATI 要求の到着などで呼び出されるコールバック・ルーチンを指すポインターです。コールバック・ルーチンが必要ない場合には、このパラメーターにヌルを設定します。

EPI は、このパラメーターを入力用のみに使用します。

Details

インストールまたは予約された端末リソースの詳細情報が、戻り時に格納される **CICS_EpiDetails_t** 構造を指すポインターです。非同期呼び出しの場合は、**Details** パラメーターを NULL に設定する必要があります。ポインターがヌルに設定されていない場合は、端末リソースのインストール要求が完了したときに、詳細が構造に追加されます。非同期呼び出しの場合は、CICS_EPI_EVENT_ADD_TERM イベントが発生したときに、これが行われます。

EPI は、この構造の各フィールドを出力用のみに使用します。

TermIndex

直前にインストールまたは予約された端末リソースの端末索引を指すポインターです。関数が戻す端末索引は、以降の EPI 関数の呼び出しで、関数の処理対象の端末リソースの識別に使用します。端末索引は整数で、0 から順に各端末リソースに割り当てられます。

EPI はこのパラメーターを出力のみに使用します。

Attributes

CICS_EpiAttributes_t 構造を指すポインターです。インストールする

端末リソースのクライアント・アプリケーションによる定義が可能な属性を指定します。*この構造を使用する前に、それをヌルに設定しておく必要があります。*

ポインターをヌルに設定した場合は、デフォルトの属性が使用されます。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_SYSTEM
指定されたサーバーが、クライアント側で認識できません。

CICS_EPI_ERR_SECURITY
サーバーが、セキュリティー上の理由で、要求を拒否しました。

CICS_EPI_ERR_NULL_PARM
TermIndex がヌル・ポインターです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_RESPONSE_TIMEOUT
指定時間間隔内にサーバーからの応答がありませんでした。

CICS_EPI_ERR_SIGNON_NOT_POSS
サーバーでは、端末リソースをサインオン対応としてインストールすることはできません。

CICS_EPI_ERR_SERVER_DOWN
サーバーが停止したため、関数が失敗しました。

CICS_EPI_ERR_PASSWORD_INVALID
パスワードの長さが CICS_EPI_PASSWORD_MAX を超えています。

CICS_EPI_ERR_ADDTYPE_INVALID
CICS_EpiAttributes_t 構造内の **EpiAddType** フィールドに割り当てられた値が、 CICS_EPI_ADD_ASYNC でも CICS_EPI_ADD_SYNC でもありません。

CICS_EPI_ERR_SIGNONCAP_INVALID
CICS_EpiAttributes_t 構造内の **SignonCapability** フィールドに割り

当てられた値が、 CICS_EPI_SIGNON_CAPABLE でも CICS_EPI_SIGNON_INCAPABLE でもありません。

CICS_EPI_ERR_USERID_INVALID
ユーザー ID の長さが CICS_EPI_USERID_MAX を超えています。

CICS_EPI_ERR_TERMID_INVALID
無効な TermId が指定されたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_MODELID_INVALID
無効なモデル端末定義が指定されたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_NOT_3270_DEVICE
指定された装置タイプが 3270 装置のものでなかったため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_ALREADY_INSTALLED
端末がすでにインストールされていたため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_CCSID_INVALID
無効な CCSID が指定されたため、関数が失敗しました。

各種の文字セットの CCSID 値の詳細は、「CICS クライアント 管理の手引き」の付録を参照してください。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_SERVER_BUSY
サーバーが使用中であるため、関数が失敗しました。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_VERSION
この関数は、EPI が初期化されたバージョンではサポートされません。

CICS_EPI_ERR_RESOURCE_SHORTAGE
CICS サーバーまたはクライアントに、端末のインストールを完了させるためのリソースが不足しています。

CICS_EPI_ERR_MAX_SESSIONS
この要求を満足する十分な通信リソースがありません。

CICS_EPI_ERR_MAX_SYSTEMS
ユーザーの構成で許可されている数を超えるサーバーへの接続を開始しようとしました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiInquireSystem

CICS_EpiInquireSystem	TermIndex System

目的

CICS_EpiInquireSystem 関数は、指定された端末リソース (端末索引で識別されます) をインストールするサーバーの名前を戻します。

パラメーター

TermIndex
導入場所を判別する端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

System
サーバー名が格納される、長さ CICS_ECI_SYSTEM_MAX + 1 のストリングを指すポインターです。

EPI はこのパラメーターを出力のみで使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_NULL_PARM
System がヌル・ポインターです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。サーバーの名前が **System** パラメーターに戻されます。ストリングの長さが CICS_EPI_SYSTEM_MAX +1 に満たないときは、ヌル文字で埋め込まれます。

CICS_EpiDelTerminal

CICS_EpiDelTerminal	TermIndex

目的

CICS_EpiDelTerminal 関数の呼び出しは、既存の端末リソースを削除します。アプリケーション側では、この関数の実行で CICS_EPI_EVENT_END_TERM イベントを受信するまで、削除完了を前提とした処理は行わないでください。 **CICS_EpiGetEvent** 呼び出しが CICS_EPI_EVENT_END_TERM イベントを取得するまで、端末索引は割り振られたままになります。 この関数の呼び出しは、対象の端末リソースが現在トランザクションを稼働している場合には、エラーになります。アプリケーションは、端末リソースの削除完了を確認するために、 **CICS_EpiDelTerminal** を呼び出す前に、現行のトランザクションが完了し、さらに未解決のイベントすべてを処理し終るのを待たなければなりません。

端末リソースが自動インストールされた場合は、その定義はサーバーから削除されます。対象の端末リソースに対する **CICS_EpiDelTerminal** 呼び出しが正常に終了しても、アプリケーションが対応する CICS_EPI_EVENT_END_TERM イベントを受信するまでは、端末索引は **CICS_EpiGetEvent** および **CICS_EpiGetSysError** 呼び出し以外では使用しないでください。

パラメーター

TermIndex
削除する端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_TRAN_ACTIVE
対象の端末リソースがトランザクションを実行しているか、またはその端末リソースに関する未処理のイベントがあります。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiPurgeTerminal

CICS_EpiPurgeTerminal	TermIndex

目的

EPI バージョン 2 以上に対してのみサポートされます。

CICS_EpiPurgeTerminal 関数は、以前に追加された端末リソースを除去します。アプリケーション側では、この関数の実行で CICS_EPI_EVENT_END_TERM イベントを受信するまで、削除完了を前提とした処理は行わないでください。

The **CICS_EpiPurgeTerminal** 呼び出しが **CICS_EpiDelTerminal** 呼び出しと異なる点は、アプリケーションが、現行トランザクションが終了するまで待機する必要がないこと、あるいは呼び出しを行う前にすべての未解決のイベントを処理する必要がないことです。

端末リソースが自動インストールされた場合は、その定義はサーバーから削除されます。

この除去関数は、端末にキューされた ATI 要求のキャンセルは行いません。

パラメーター

TermIndex
削除する端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_VERSION
この関数は、EPI が初期化されたバージョンではサポートされません。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiSetSecurity

CICS_EpiSetSecurity	**TermIndex**
	UserId
	Password

目的

EPI バージョン 2 以上に対してのみサポートされます。

CICS_EpiSetSecurity 関数を使用すれば、クライアント・アプリケーションでユーザー ID とパスワードを指定して、以前にサインオン対応としてインストールされた端末リソースと関連づけることができます。

CICS_EpiSetSecurity 関数は、いつでも呼び出すことができます。さらに端末リソースに関するトランザクションを開始するとき、ユーザー ID とパスワードが使用されます。端末リソースに関する関数が呼び出されなかったか、あるいは呼び出されてユーザー ID と暗黙のパスワードがヌルに設定された場合に、クライアント定義のユーザー ID とパスワードが使用されます。

クライアント・アプリケーションが、ユーザー ID とパスワードを検査しなければならない点に注意してください。

パラメーター

TermIndex

端末の端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

UserId

ユーザー ID を指定する、ヌル文字で終了するストリングを指すポインターです。ユーザー ID が CICS_EPI_USERID_MAX 文字よりも短い場合は、 CICS_EPI_USERID_MAX+1 の長さになるまでヌル文字で埋め込まなければなりません。

EPI は、このパラメーターを入力用のみに使用します。

Password

パスワードを指定する、ヌル文字で終了するストリングを指すポインターです。パスワードが CICS_EPI_PASSWORD_MAX 文字よりも短い場合は、 CICS_EPI_PASSWORD_MAX+1 の長さになるまでヌルで埋め込む必要があります。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。

CICS_EPI_ERR_VERSION
この関数は、EPI が初期化されたバージョンではサポートされません。

CICS_EPI_ERR_NULL_PASSWORD
Password がヌル・ポインターです。

CICS_EPI_ERR_NULL_USERID
Userid がヌル・ポインターです。

CICS_EPI_ERR_PASSWORD_INVALID
パスワードの長さが CICS_EPI_PASSWORD_MAX を超えています。

CICS_EPI_ERR_USERID_INVALID
ユーザー ID の長さが CICS_EPI_USERID_MAX を超えています。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiStartTran

目的

CICS_EpiStartTran	**TermIndex** **TransId** **Data** **Size**

CICS_EpiStartTran 関数は、端末リソースからの新規トランザクションの開始、または疑似会話の継続処理を行います。

- *新規トランザクションの開始* -- **CICS_EpiAddTerminal** の呼び出し後、または前のトランザクションが端末リソースからの次の入力を処理するトランザクションを**指定していない**ことを示す CICS_EPI_EVENT_END_TRAN を受信した場合に行います。
- *疑似会話の継続* -- 前のトランザクションが、端末リソースからの次の入力を処理するトランザクションを**指定している**ことを示す CICS_EPI_EVENT_END_TRAN イベントを受信した場合に行います。

呼び出しが正常に完了すると、開始したトランザクションが終了するまで、対象の端末リソースに新規トランザクションの開始を要求することはできません。この状況は、CICS_EPI_EVENT_END_TRAN イベントで判別できます。

パラメーター

TermIndex

トランザクションを稼働する端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

TransId

稼働するトランザクションを指定したストリングを指すポインター、またはヌル・ポインターです。新規のトランザクションを開始する場合、このパラメーターがヌル・ポインターであれば、**Data** パラメーターのデータ・ストリームからトランザクション名が獲得されます。疑似会話を継続する場合、ポインターがヌルでなければ、対象端末リソースに対して最後に発生した CICS_EPI_EVENT_END_TRAN イベントで戻されるトランザクション名をこのパラメーターに指定しなければなりません。ポインターがヌルでなく、ストリングが CICS_EPI_TRANSID_MAX で設定された長さよりも短い場合には、その長さになるまでスペースが埋め込まれます。

EPI は、このパラメーターを入力用のみに使用します。

Data 対象トランザクションで使用する 3270 データ・ストリームを指すポインターです。このパラメーターがヌル・ポインターであってはなりません。データ・ストリームには、少なくとも AID バイトが常に格納されます。

新規のトランザクションを開始する場合に、**TransId** パラメーターがヌル・ポインターなら、このデータ・ストリームの長さは少なくとも 4 バイトにする必要があります。開始するトランザクション名が必ず含まれ、さらに最初の EXEC CICS RECEIVE コマンドでトランザクションに渡されるデータが格納されている場合もあります。

新規のトランザクションを開始する場合に、**TransId** パラメーターがヌル・ポインター以外であれば、データ・ストリームの長さは、1 バイト (AID バイト)、 3 バイト (AID バイトおよびカーソル・アドレス)、または 4 バイト以上 (AID バイト、カーソル・アドレス、データおよび SBA コマンド) になります。 4 バイト以上の場合には、これらのデータは最初の EXEC CICS RECEIVE コマンドでトランザクション・プログラムに渡されます。

疑似会話を継続する場合には、このデータ・ストリームのデータ長は、1 バイト (AID バイト)、3 バイト (AID バイトおよびカーソル・アドレス)、または 4 バイト以上 (AID バイト、カーソル・アドレス、データおよび SBA コマンド) になります。 4 バイト以上の場合には、これらのデータは最初の EXEC CICS RECEIVE コマンドでトランザクション・プログラムに渡されます。

CICS 用 3270 データ・ストリームの形式の詳細については、130ページの『EPI の 3270 データ・ストリーム』を参照してください。

CICS_EpiAddTerminal で端末リソースをインストールした場合は、3270 データ・ストリームの長さは、**CICS_EpiDetails_t** の **MaxData** に戻された値を超えてはなりません。

EPI は、このパラメーターを入力用のみに使用します。

Size トランザクションに最初に渡されるデータのバイト長です。

EPI は、このパラメーターを入力用のみに使用します。

注: アプリケーションが、端末リソースがフリーで新規トランザクションを開始できると判断した場合でも、**CICS_EpiStartTran** の呼び出しで戻りコード (CICS_EPI_ERR_ATI_ACTIVE) を得ることがあります。その場合には、EPI がその端末リソースに対して ATI 要求を開始し、対応する

CICS_EPI_EVENT_START_ATI イベントが出されましたが、アプリケーション側では、 **CICS_EpiGetEvent** 呼び出しを出して、イベントが受信されていません。

戻りコード

CICS_EPI_ERR_ATI_ACTIVE
対象の端末リソースで ATI トランザクションがアクティブです。

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NO_DATA
初期データが指定されていません。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_TTI_ACTIVE
対象端末リソースで、EPI が開始したトランザクションがすでにアクティブです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_SERVER_DOWN
サーバーが停止したため、関数が失敗しました。

CICS_EPI_ERR_RESOURCE_SHORTAGE
CICS サーバーまたはクライアントに、端末のインストールを完了させるためのリソースが不足しています。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_ERR_MAX_SESSIONS
この要求を満足する十分な通信リソースがありません。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiReply

CICS_EpiReply	TermIndex
	Data
	Size

目的

CICS_EpiReply 関数は、端末リソースから CICS トランザクションにデータを送信します。CICS_EPI_EVENT_CONVERSE イベントへの応答の場合のみ使用します。

パラメーター

TermIndex
データ送信元の端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

Data トランザクションに送信される 3270 データ・ストリームを指すポインターです。このパラメーターがヌル・ポインターであってはなりません。データ・ストリームには、少なくとも AID バイトが常に格納されます。データ・ストリームには、1 バイト (AID バイト)、3 バイト (AID バイトおよびカーソル・アドレス)、または 4 バイト以上 (AID バイト、カーソル・アドレス、データおよび SBA コマンド) が格納されます。4 バイト以上の場合には、カーソル・アドレスに続くデータが、最初の EXEC CICS RECEIVE コマンドでトランザクション・プログラムに渡されます。

CICS_EpiAddTerminal で端末リソースをインストールした場合は、3270 データ・ストリームの長さは、**CICS_EpiDetails_t** の **MaxData** に戻された値を超えてはなりません。

EPI は、このパラメーターを入力用のみに使用します。

Size データのバイト長です。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NO_CONVERSE
端末リソースに応答は不要です。

CICS_EPI_ERR_NO_DATA
応答データがありません。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_ERR_SERVER_DOWN
サーバーが停止したため、関数が失敗しました。

CICS_EPI_ERR_ABENDED
読み取りタイムアウト期間が満了して会話が異常終了しましたが、アプリケーションはまだ CICS_EPI_EVENT_END_TRAN イベントを受け取っていません。

これは CICS_EPI_VERSION_200 に対してのみサポートされます。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiATIState

CICS_EpiATIState	TermIndex ATIState

目的

CICS_EpiATIState 関数を実行すると、呼び出し側のアプリケーションから端末リソースの ATI 要求の処理方法の検索および変更が可能です。ATI 要求が使用可能で (CICS_EPI_ATI_ON)、ATI 要求がサーバーに送信された場合は、端末リソースの解放後に要求が開始します。 ATI 要求が保留される (CICS_EPI_ATI_HOLD) と、送信された ATI 要求はすべて待ち状態になり、ATI 要求が有効になりしだい実行されます。

CICS_EpiAddTerminal 呼び出し後の ATI 要求の状態は、CICS_EPI_ATI_HOLD になります。EPI アプリケーションは、ATI 要求の処理準備が整いしだい、 ATI 要求を有効 (CICS_EPI_ATI_ON) にすることができます。 (端末リソースの ATI 状態は、EPI で維持する ATI 状態とは独自にサーバー側でも維持されています。サーバー上の ATI 状態を変更しても、EPI 側の ATI 状態はそのままです。)

パラメーター

TermIndex

ATI 状態の要求対象の端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

ATIState

このパラメーターは、以下のそれぞれの入力値によって、 EPI によって入出力の両方で使用されます。

CICS_EPI_ATI_ON

ATI 状態を使用可能にします。前の ATI 状況がパラメーターに格納されます。

CICS_EPI_ATI_HOLD

ATI 要求を次に使用可能になるまで保留します。前の ATI 状態がパラメーターに格納されます。

CICS_EPI_ATI_QUERY

ATI 状態は変更されません。現在の状態がパラメーターに格納されます。

戻りコード

CICS_EPI_ERR_ATI_STATE
提供された **ATIState** の値が無効です。

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NULL_PARAM
ATIState がヌル・ポインターです。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiSenseCode

CICS_EpiSenseCode	TermIndex SenseCode

目的

EPI バージョン 2 に対するサポートはありません。

アプリケーションは、この関数を呼び出すことによって、トランザクションが送信する 3270 データ・ストリーム内の全エラーを EPI に通知することができます。

この関数は、CICS クライアントに影響を与えません。

パラメーター

TermIndex
エラーの起こった端末リソースの端末索引です。

EPI は、このパラメーターを入力用のみに使用します。

SenseCode
センス・コード・エラーの理由を示します。以下のいずれかの値が格納されます。

CICS_EPI_SENSE_OPCHECK
3270 データ・ストリーム内でエラーが検出されました。

CICS_EPI_SENSE_REJECT
受信した 3270 コマンドが無効です。

EPI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_SENSE_CODE
提供されたセンス・コードが無効です。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_VERSION
この関数は、EPI が初期化されたバージョンではサポートされません。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

CICS_EpiGetEvent

CICS_EpiGetEvent	**TermIndex** **Wait** **Event**

目的

CICS_EpiGetEvent 関数は、指定した端末リソースに対するイベントの情報を戻します。

この関数はアプリケーションからのみ実行できます。コールバック・ルーチンからは実行できません。

パラメーター

TermIndex

イベントの獲得対象の端末リソースの端末索引です。このパラメーターを定数の CICS_EPI_TERM_INDEX_NONE に設定すれば、呼び出し側のアプリケーションが使用している端末リソースのいずれかに対し、次に戻されるイベントを示すことができます。アプリケーションは、関数が戻す **CICS_EpiEventData_t** 構造の **TermIndex** フィールドを検査すれば、イベントが発生した端末リソースを確認できます。

このパラメーターは、EPI が入出力の両方に使用します。

Wait 指定した端末リソースに対するイベントが存在しない場合の処理を指定します。以下のいずれかの値を指定します。

CICS_EPI_WAIT
次のイベントが生成されるまで戻りません。

CICS_EPI_NOWAIT
エラー・コードを出して即時に戻ります。このオプションは、アプリケーションがイベントのポーリングを行う場合に指定します。

EPI は、このパラメーターを入力用のみに使用します。

Event

戻り時に発生したイベントの詳細情報を格納する **CICS_EpiEventData_t** 構造を指すポインターです。この構造の **Data** フィールドには、対象イベントに関連する全端末データ・ストリームで更新されるデータ・バッファーを指すポインターを指定します。**Size**

フィールドには、このバッファーの最大サイズを指定します。戻り時には、実際のデータ長さで更新されます。

戻りコード

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_MORE_DATA
データ・バッファーの指定サイズが不十分で、端末データ全体を格納できません。データの未格納部分は廃棄されました。

CICS_EPI_ERR_MORE_EVENTS
イベントの獲得に成功しましたが、指定された端末リソースには、さらに未解決のイベントが存在します。

CICS_EPI_ERR_NO_EVENT
この端末リソースに対する未解決のイベントはありません。

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が実行されていません。

CICS_EPI_ERR_WAIT
Wait パラメーターの値が無効です。

CICS_EPI_ERR_NULL_PARM
Event がヌル・ポインターです。

CICS_EPI_ERR_IN_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_EPI_NORMAL
関数の処理が正常に終了しました。他にイベントはありません。

CICS_EpiGetSysError

CICS_EpiGetSysError	TermIndex SysErr

目的

EPI バージョン 2 に対するサポートはありません。

CICS_EpiGetSysError 関数は、最後に発生したエラーの詳細情報を戻します。各関数のエラー発生は、戻りコード CICS_EPI_ERR_FAILED で示されます。関数実行時に戻される **SysErr** パラメーターの **Cause** および **Value** フィールドの値は、他の EPI 関数の戻りコードの詳細情報を知るのに使われます。これらの値は環境によって異なります。使用しているクライアント環境および CICS サーバーの関連資料を参照してください。

関数実行時に指定する **SysErr** パラメーターの **Msg** フィールドには、発生したエラー・コードの詳細メッセージが戻される場合もあります。このメッセージはオペレーティング・システムによって異なります。メッセージがない場合には、このフィールドにはヌルが格納されます。

注: CICS ユニバーサル・クライアントのバージョン 3.1 では、この関数は後方互換性のために予約されています。この関数よって戻される情報はありません。すべてのシステム・エラーは、クライアント・エラー・ログに書き込まれます。

パラメーター

TermIndex

詳細エラー・コードの獲得対象の端末リソースの端末索引です。**CICS_EpiInitialize**、**CICS_EpiTerminate**、 **CICS_EpiListSystems**、または **CICS_EpiAddTerminal** の各呼び出しのエラーの詳細情報が必要な場合には、このパラメーターに定数の CICS_EPI_TERM_INDEX_NONE を指定します。

EPI は、このパラメーターを入力用のみに使用します。

SysErr

戻り時にシステム・エラー情報を格納する **CICS_EpiSysError_t** 構造を指すポインターです。

EPI は、この構造を出力用のみに使用します。

戻りコード

CICS_EPI_ERR_NOT_INIT
CICS_EpiInitialize が呼び出されていません。(**CICS_EpiInitialize** 呼び出しに失敗しても、 **CICS_EpiGetSysError** は成功する場合があります。)

CICS_EPI_ERR_BAD_INDEX
TermIndex 値が無効な端末索引です。

CICS_EPI_ERR_FAILED
関数の処理が、予期せぬ理由で失敗しました。

CICS_EPI_ERR_NULL_PARM
SysErr がヌル・ポインターです。

CICS_EPI_VERSION
この関数は、EPI が初期化されたバージョンではサポートされません。

CICS_EPI_NORMAL
関数の処理が正常に完了しました。

EPI イベント

EPI イベントは、CICS® から EPI アプリケーションに渡されるデータが存在する場合に発生します。アプリケーションは、さまざまな方法で EPI イベントを処理することができます。(67ページの『イベントおよびコールバック』を参照してください。) いずれの処理方法でも、アプリケーションは CICS® のデータ取り込みで **CICS_EpiGetEvent** 関数を呼び出します。

CICS_EPI_ADD_TERM

目的

CICS_EPI_EVENT_ADD_TERM イベントは、端末リソースのインストールの非同期要求が完了したことを示します。端末リソースがインストールされた場合は、詳細情報が **CICS_EpiDetails_t** 構造に入れられ、その位置が **Data** で示されます。

完了フィールド

Event The CICS_EPI_ADD_TERM イベント・コードです。

EndReturnCode
終了の理由です。戻りコードの詳細については、**CICS_EpiAddExTerminal** 関数を参照してください。

Data **EndReturnCode** が CICS_EPI_NORMAL である場合に、端末詳細情報によって更新された **CICS_EpiDetails_t** 構造を指すポインターです。

CICS_EPI_EVENT_SEND

目的

CICS_EPI_EVENT_SEND イベントは、トランザクションから端末リソースに 3270 データが送信されたこと (通常、EXEC CICS SEND コマンドの結果として) を示します。応答は通常必要ありませんので、応答処理は行わないようにします。

完了フィールド

Event CICS_EPI_EVENT_SEND イベント・コードです。

Data トランザクションが送信するデータで更新されるバッファーを指すポインターです。データ・ストリーム形式の詳細については、130ページの『EPI の 3270 データ・ストリーム』を参照してください。

Size **Data** バッファーのデータ長です。

CICS_EPI_EVENT_CONVERSE

目的

CICS_EPI_EVENT_CONVERSE イベントは、EXEC CICS RECEIVE コマンドまたは EXEC CICS CONVERSE コマンドの実行の結果、トランザクションが応答を要求していることを示します。

アプリケーションは、以下のように **CICS_EpiReply** 呼び出しを行い、データを CICS に戻さなければなりません。

- トランザクションが、BUFFER オプションを指定せずに EXEC CICS RECEIVE コマンドを出したとき、バッファーにはトランザクションから送信されたデータが格納されるか、あるいは空のままになります。処理対象のデータが存在する場合には、応答の前に処理を行います。応答は、送信するデータが存在する場合のみ送信します。
- トランザクションが EXEC CICS RECEIVE BUFFER コマンドを出した場合は、データ・バッファーに 3270 Read Buffer コマンドがセットされ、 **Size** フィールドに 1 がセットされます。応答はコマンド発行後すぐに送信してください。

完了フィールド

Event CICS_EPI_EVENT_CONVERSE イベント・コードです。

Data トランザクションから送信される上記のデータで更新されるバッファーを指すポインターです。

Size バッファーのデータ長です。この値が 0 の場合は、データは送信されていません。この場合も、応答は送信します。

CICS_EPI_EVENT_END_TRAN

目的

CICS_EPI_EVENT_END_TRAN イベントは、端末リソースを稼働していたトランザクションが終了したことを示します。トランザクションが失敗した場合は、 **EndReason** および **EndReturnCode** がその原因を指定します。トランザクションが正常に終了した場合は、 **EndReason** フィールドが CICS_EPI_TRAN_NO_ERROR に設定され、 **EndReturnCode** フィールドが CICS_EPI_NORMAL に設定されます。終了したトランザクションが疑似会話の場合は、 **TransId** フィールドに次に実行するトランザクションの名前が格納されます。アプリケーションで、**CICS_EpiStartTran** 呼び出しを出し、指定されたトランザクションを開始させます。

CICS_EPI_EVENT_END_TRAN イベントが発生するのは、端末リソースに対して実行されているトランザクションが異常終了したとき、または IMMEDIATE オプションが指定されていない RETURN コマンドの後で終了したときです。

完了フィールド

Event CICS_EPI_EVENT_END_TRAN イベント・コードです。

EndReason
終了トランザクション・イベントが発生した原因を示します。以下のいずれかの値が格納されます。

CICS_EPI_TRAN_NO_ERROR
トランザクションの正常終了です。

CICS_EPI_TRAN_NOT_STARTED
トランザクションの開始が失敗しました。

CICS_EPI_TRAN_STATE_UNKNOWN
トランザクションの完了が失敗しました。

CICS_EPI_READTIMEOUT_EXPIRED
読み取りタイムアウトが満了しました。

TransId
前のトランザクションが疑似会話の場合に、次に開始するトランザクションの名前が格納されます。名前は 4 文字で、最後にヌルが付加されます。次のトランザクションが存在しない場合は、このフィールドはヌル・ストリングになります。

EndReturnCode
CICS_EPI_*returncode* を含むストリング。

CICS_EPI_EVENT_START_ATI

目的

CICS_EPI_EVENT_START_ATI イベントは、端末リソースに対する ATI トランザクションが開始されたことを示します。端末リソースが他のトランザクションを稼働している途中に ATI 要求を受信した場合には、トランザクション終了まで要求は保持されます。端末リソースのための ATI トランザクションが開始されると、 CICS_EPI_EVENT_START_ATI イベントが生成され、アプリケーションに通知されます。

完了フィールド

Event CICS_EPI_EVENT_START_ATI イベント・コードです。

TransId

開始されたトランザクションの名前が格納されます。名前は 4 文字で、最後にヌルが付加されます。

CICS_EPI_EVENT_END_TERM

目的

CICS_EPI_EVENT_END_TERM イベントは、対象の端末リソースが存在しないことを示します。このイベントの生成後、その端末リソースに使われた端末索引が無効になります。EPI が CICS サーバーの遮断を検出した場合は、アプリケーションがそのサーバーにインストールしたが削除されていない既存の全端末リソースに対して CICS_EPI_EVENT_END_TERM イベントが生成されます。

完了フィールド

Event CICS_EPI_EVENT_END_TERM イベント・コードです。

EndReason
端末リソースの削除理由を示します。以下のいずれかの値が格納されます。

CICS_EPI_END_SIGNOFF
端末リソースがサインオフされました。CESF トランザクションの稼働または **CICS_EpiDelTerminal** 関数の呼び出しのいずれかの結果です。

CICS_EPI_END_SHUTDOWN
CICS サーバーが遮断されました。

CICS_EPI_END_OUTSERVICE
端末リソースがサービス不能になっています。

CICS_EPI_END_UNKNOWN
予期せぬエラーが発生しました。

CICS_EPI_END_FAILED
端末リソースの削除の試行が失敗しました。

EPI の 3270 データ・ストリーム

EPI にインプリメントされたデータ・ストリームは、「*3270 データストリーム プログラマー用解説書*」で定義されたデータ・ストリームに準拠しています。EPI への送信データは ASCII 形式で、構造化フィールドはサポートされていません。データ・フローは、「*3270 データストリーム プログラマー用解説書*」にある以下のトピックで定義されています。

- 3270 データ・ストリームの概要 (構造化フィールドの部分を除く)
- 3270 データ・ストリーム・コマンド
- 文字セット、オーダー、および属性

- キーボードおよびプリンターの操作

EPI の 3270 データ・ストリームの使い方、および標準以外のオーダーと属性については、本書で定義されています。

提供される C ヘッダー・ファイル CICS3270.H、およびインクルード・ファイル CICS3270.INC に、3270 データ・ストリームを処理するのに役立つ定数および変換テーブルが含まれています。

CICS ユーザーのトランザクションで EXEC CICS SEND コマンドまたは EXEC CICS RECEIVE コマンドを出す場合、このトランザクションと EPI アプリケーションとの間で受け渡すデータ (制御バイト以降) は、CICS の処理対象になりません。この場合には、ユーザー定義データをデータ・バッファーに格納して、プログラム間で受け渡します。ただし、CICS システム間の受け渡しでは、バッファーの内容はコード・ページの変換が行われる可能性があります。 CICS システムでは、ASCII および EBCDIC 形式のデータのみが使用可能です。

CICS トランザクションが EXEC CICS SEND MAP コマンドまたは EXEC CICS RECEIVE MAP コマンドを出す場合には、 CICS が BMS 構造体データを 3270 データ・ストリームに変換します。アプリケーションは CICS から 3270 データを受信します。送信の際にもトランザクション用に変換が行われるため、アプリケーションから有効な 3270 データを送信する必要があります。

配布された CICS プロダクト (つまり、CICS ユニバーサル・クライアント) は、3270 データ・ストリーム・アーキテクチャーの ASCII-7 サブセットをサポートします。このため、14 および 16 ビット・アドレスまたは構造化フィールド (あるいは、その両方とも) が使用できなくなり、 12 ビット SBA アドレスのみがサポートされることになります。この制限のために、EPI 端末の最大画面サイズは 27 行 x 132 列になります。 CICS/390 および CICS/400 プロダクトは、EBCDIC 3270 データ・ストリーム・アーキテクチャーをサポートします。ただし、配布されたいずれかの CICS プロダクトからトランザクションを開始する場合は、このようなサーバーのトランザクションでは、 14 および 16 ビット・アドレスまたは構造化フィールド (あるいは、その両方) の使用を避ける必要があります。

インバウンド・データ・ストリーム (EPI から CICS への送信時)

AID (1 バイト)	カーソル・アドレス (2 バイト)	データ・バッファー (可変長)

EPI アプリケーションは、以下の関数の呼び出しの際に CICS に 3270 データを送信します。

- **CICS_EpiStartTran**
- **CICS_EpiReply**

いずれの場合も、送信するデータの形式は同じです。データ・ストリームは、少なくとも AID およびカーソル・アドレスを含む 3 バイト以上のデータです (ヌルは格納できません)。ただし、AID が CLEAR または PA キーを示す場合には、カーソル・アドレスが含まれない場合もあります。これらのフィールドの値は、EIB の EIBAID および EIBCPOSN の 2 つのフィールドに格納され、CICS に渡されます。

データ・バッファーの内容は以下のとおりです。

- 3270 制御文字を含む表示可能な ASCII 文字 (EXEC CICS RECEIVE MAP コマンドに渡される場合)
- ユーザー定義データ (EXEC CICS RECEIVE コマンドに渡される場合)

トランザクションの開始時に、データ・バッファーの以下の部分からトランザクション ID が取り込まれます。

- データ・バッファーの最初に、バッファー・アドレス設定 (SBA) オーダーが格納されている場合は、データ・バッファーの 4 バイト目から 7 バイト目の部分からトランザクション ID が取り込まれます。
- データ・バッファーの最初の部分に SBA がなければ、データ・バッファーの 1 バイト目から 4 バイト目の部分からトランザクション ID が取り込まれます。

いずれの場合も、トランザクション ID のデータは 4 バイトより短く、以降の他の SBA または ASCII スペースまたストリングの終了で区切られます。

CICS トランザクションの開始時に渡されるデータ・バッファーの内容は、最初の EXEC CICS RECEIVE コマンドに応答するトランザクションで利用可能になります。

アプリケーションの応答では、データ・バッファーの内容は EXEC CICS RECEIVE コマンドへの応答では変換されない形式、 EXEC CICS RECEIVE MAP コマンドへの応答では BMS 構造に変換された形式が使用可能です。

注: EXEC CICS RECEIVE MAP コマンドを実行する場合は、正常に変換が行われるように、EPI のアプリケーション側でデータが正しくフォーマットされているかを、EPI プログラマーが確認する必要があります。

アウトバウンド・データ・ストリーム (CICS から EPI への送信時)

コマンド (1 バイト)	書き込み制御 文字 (1 バイト)	データ・バッファー (可変長)

3270 コマンドには、書き込みコマンドと読み取りコマンドがあり、それぞれ EPI に対してデータの処理、またはデータ応答を要求します。

CICS_EPI_EVENT_SEND イベントの発生時には、以下のいずれかの 3270 書き込みコマンドが送信されます。

- Write (書き込み)
- Erase/Write (消去 / 書き込み)
- Erase/Write Alternate (消去 / 書き込みのいずれか)
- Erase All Unprotected (非保護部分の全消去)

上から 3 つのコマンドの送信では、書き込み制御文字 (WCC) およびデータが続けて送信されます。Erase All Unprotected コマンドは WCC およびデータなしで送信されます。Write Structured Field (構造化フィールドの書き込み) コマンドは CICS から生成されず、EPI では使用できません。

データ・バッファーの内容は以下のとおりです。

- 3270 制御文字を含む表示可能な ASCII 文字 (EXEC CICS SEND MAP コマンドから渡される場合)
- ユーザー定義データ (EXEC CICS SEND コマンドから渡される場合)

CICS_EPI_EVENT_CONVERSE イベントでは、読み取りコマンドが指定されます。データ・ストリームの内容は、イベントのソースにより、以下のようになります。

- イベントが EXEC CICS RECEIVE コマンドの結果で生じた場合、データ・バッファーはトランザクションから送信されたデータを格納するか、または空になります。 EPI プログラムは、送られたデータが使用可能であれば応答します。
- イベントが EXEC CICS RECEIVE BUFFER コマンドの結果で生じた場合は、データ・バッファーには 3270 Read Buffer (バッファー読み取り) コマンドが格納されます。これは、「*3270 データストリーム プログラマー用解説書*」の記述に従って処理を行わなければなりません。

Read Modified (修正部分読み取り) および Read Modified All (全修正部分読み取り) コマンドは CICS からは生成されず、EPI では使用できません。

3270 オーダー・コード

3270 オーダーは、補足的な制御関数を提供するためのものであり、インバウンド・データ・ストリームにもアウトバウンド・データ・ストリームにも含まれます。表4 に、3270 データ・ストリームで使用される可能性のあるオーダー・コードをリストしており、各コードがインバウンドとアウトバウンドどちらのデータ・ストリームに関連付けられているか (あるいは両方に関連付けられているか) を示しています。

表 4. 3270 データ・ストリームのオーダー・コード

オーダー・コード	インバウンド	アウトバウンド
フィールド開始 (SF)	O	O
フィールド開始拡張 (SFE)	O	O
バッファー・アドレス設定 (SBA)	O	O
属性設定 (SA)	O	O
フィールド修正 (MF)	X	O
カーソル挿入 (IC)	X	O
プログラム・タブ (TB)	X	O
アドレスまで反復 (RA)	X	O
アドレスまで非保護部分消去 (EUA)	X	O
グラフィック・エスケープ (GE)	X	X

注: 「*3270 データストリーム プログラマー用解説書*」には、SFE、SA、および MF の各オーダーは、ASCII 形式ではサポートされないと定義されています。しかし、EPI の 3270 データ・ストリームには、この形式で格納される場合があります。この場合、それぞれ以下の値になります。

```
SFE     X'10'
SA      X'1F'
MF      X'1A'
```

以前は、MF に X'1E' の値が割り当てられていました。しかし、フィールド・マーク形式制御命令にも同じ値が割り当てられるため、MF の値を X'1A' に変更しなければ、 EPI アプリケーションは MF と FM を区別することができなくなります。

これらのオーダーには、1 つ以上の属性タイプの値の対が付加されます。属性値の対の数と属性タイプは両方とも 2 進数値であり、このことは、「*3270 データストリーム プログラマー用解説書*」で定義されています。

ただし、属性値フィールドの内容は、「*3270 データストリーム プログラマー用解説書*」で定義されているものとは、以下の点で異なる場合があります。

- 属性タイプが X'C0' 以下 (たとえば、色指定) の場合には、この属性値は、「*3270 データストリーム プログラマー用解説書*」では、EBCDIC 値として定義されています。 EPI では、EBCDIC 値に対応する ASCII を使用します。たとえば、「*3270 データストリーム プログラマー用解説書*」では、赤色は X'F2' として定義されていますが、 EPI データ・ストリームでは X'32' として定義する必要があります。
- 属性タイプの値が X'C0' より大きい (たとえばフィールド枠指定) 場合には、この属性値は 2 進数値になります。EPI は、「*3270 データストリーム プログラマー用解説書*」で定義されている値を使用します。

3270 の各オーダーおよびその他の制御文字の詳細については、次の表に示されているファイルに説明があります。

	提供されたファイル
C ヘッダー・ファイル	CICS3270.H

第4章 ECI および EPI アプリケーション・プログラムの作成

本章は、外部アクセス・インターフェースの各プログラミング言語に依存する情報について説明します。本章の構成は以下のとおりです。

本章では、ECI および EPI アプリケーションのテストおよびデバッグの方法については説明しません。作業する環境でのプログラミング解説書を参照してください。

CICS 以外のアプリケーションの作成

ECI および EPI アプリケーションは標準的な独立型プログラムで、あらゆるクライアント・システム上で稼働可能です。これらのアプリケーションから、CICS® for OS/2® サーバー上の ECI および EPI のサーバー・インプリメンテーションが使用できます。さらに、オペレーティング・システム特有の呼び出しを行わないアプリケーション・プログラムは、これらのいずれの環境でも使用できます。*付属のサンプル・プログラムおよび関連ビルド・ファイルには、使用環境に関する本章の説明内容のほとんどが盛り込まれています。*

アプリケーション・プログラムは、ECI および EPI の両方の機能を使用できます。

アプリケーションは単一プロセスとして構成されていなければなりません。ただし、1 つのプロセスでいくつかのスレッドの生成が可能な環境においては、アプリケーションをマルチスレッドすることができます。

アプリケーションは、C または C++ 言語で作成することができます。ただし、環境によっては、使用できない言語もあります。

以下の表に、各プログラミング環境での違いを示します。

- ECI 用 C ヘッダー・ファイル名
- EPI 用 C ヘッダー・ファイル名
- C プログラム用インターフェースの型定義ファイル名

使用	ファイル
ECI: C プログラム	CICS_ECI.H
EPI: C プログラム	CICS_EPI.H
タイプ定義: C プログラム	CICSTYPE.H

使用する言語に応じたファイルの内容を参照してください。各ファイルには、使用するインターフェース用のプログラム作成に必要な入り口点、タイプ、データ構造、および定数のすべてが定義されています。

各言語における、インターフェースの構造、フィールド、および定数の命名規則は、以下のとおりです。

- C -- 本書のインターフェースの説明で使用された名前の記述方式と同じです。

この規則には例外もあります。例外については掲載箇所に記述されています。

C プログラムのコンパイルの際には、CICS の外部機能に渡される構造をパック・データにする必要が生じる場合があります。その場合には、 C ヘッダー・ファイルに `#pragma pack` ディレクティブが含まれており、変更されていないことを確認してください。

ECI 呼び出しの作成

ECI 関数は以下の方法で呼び出されます。

CICS_ExternalCall

C による **CICS_ExternalCall** 関数の呼び出し方法について説明します。パラメーターを ECI に渡す際には、ECI パラメーター・ブロックが使用されます (C では ECI_PARMS です)。呼び出しおよび関連する宣言の形式を以下に示します。

C プログラムの場合

```
ECI_PARMS      EciBlock;
cics_sshort_t    Response;
.
.
.
Response = CICS_ExternalCall(&EciBlock);
```

コールバック・ルーチン

コールバック・ルーチンには、パラメーターを 1 つ渡す必要があります。コールバック・ルーチンの作成についてはサンプル・プログラムを参照してください。

CICS_EciListSystems

各言語での呼び出し形式を以下に示します。

C プログラムの場合

```
#define MAX_SYS 5
cics_sshort_t    Response;
cics_ushort_t    Count;
CICS_EciSystem_t List[MAX_SYS];
.
Count = MAX_SYS;
.
Response = CICS_EciListSystems(NULL, &Count, List);
```

EPI 呼び出しの作成

EPI 関数

EPI 関数は標準的な方法で呼び出されます。**CICS_EpiListSystems** 関数の呼び出し例を以下に示します。

C プログラムの場合

```
#define MAX_SYS 5
CICS_EpiSystem_t System;
cics_ushort-t    Count;
cics_sshort_t    Response;
.
Count = MAX_SYS;
.
Response = CICS_EpiListSystems(NULL, &Count, &System);
```

コールバック・ルーチン

コールバック・ルーチンには、パラメーターを 1 つ渡す必要があります。コールバック・ルーチンの作成についてはサンプル・プログラムを参照してください。

アプリケーションのコンパイルおよびリンク

以下にリストするディレクトリーには、ECI および EPI を呼び出すサンプル・プログラムが入っています。これらのディレクトリーは、それらのサンプル・プログラムのコンパイルおよびリンクで使用できるコマンド・ファイルまたは makefile も含んでいます。これらのファイル内のコマンドは、ユーザー独自のアプリケーションのコンパイルおよびリンクの方法を示す例としても使用できます。

- IBM CICS ユニバーサル・クライアント (Windows 版)®:
  - ...¥IBM CICS Transaction Gateway¥SAMPLES¥C
  - ...¥IBM CICS Transaction Gateway¥SAMPLES¥CPP
- CICS ユニバーサル・クライアント (AIX 版)®:
  - /usr/lpp/ctg/samples/c
  - /usr/lpp/ctg/samples/cpp
- CICS ユニバーサル・クライアント (HP-UX 版)、CICS ユニバーサル・クライアント (Linux 版)、CICS ユニバーサル・クライアント (Solaris 版):
  - /opt/ctg/samples/c
  - /opt/ctg/samples/cpp

第5章 外部セキュリティー・インターフェース

本章では、外部セキュリティー・インターフェース (ESI) について解説します。

インターフェースのエレメントに選んだ名前、つまり、関数、パラメーター、データ構造、フィールド、および定数などは、プログラミングで提供される名前と似ていますが、ここで解説するインターフェースの適用範囲は、特定の言語に限定されません。

特定のオペレーティング・システムのみに適用される関数制限は、関連する関数の説明に記載されています。

本章の構成は以下のとおりです。

概要

外部セキュリティー・インターフェース (ESI) によって、拡張プログラム間通信機能 (APPC) のパスワード有効期限管理 (PEM) が提供するサービスを非 CICS アプリケーションから呼び出すことができます。

CICS では、APPC PEM は、CICS サーバーに対するユーザー ID にサインオンする APPC アーキテクチャー・サインオン・トランザクションのためのサポートを提供し、以下によってパスワード変更の要求を処理します。

- ユーザーを識別して、そのユーザーの ID を認証する。
- 認証時に、特定のユーザーにパスワードの有効期限が切れたことを通知する。
- パスワードの有効期限が切れたとき (または切れる前) に、ユーザーにパスワードを変更させる。
- 現在のパスワードが有効とされる期間をユーザーに通知する。
- 特定のユーザー ID を使用したサーバーへの認証されていないアクセス試行に関する情報を提供する。

APPC PEM の利点

APPC PEM には、以下の利点があります。

- ユーザーが APPC リンクでパスワードを更新することができます。

  パスワードの有効期限が切れた場合には特に便利です。 APPC PEM をサポートしていない APPC リンクでは、リモート・システム上のユーザーのパスワードの有効期限が切れた場合に、ユーザー自身のシステムからパスワードを更新することができません。非 APPC PEM システムでは、LU2 3270 エミュレーション・セッションなどの非 APPC リンクを使用してリモート・システムに直接ログオンしてパスワードを更新するしかありません。
- APPC ユーザーに、サインオンの状況に関する追加情報 (ユーザーが最後にサインオンした日時など) を提供します。ユーザーが正しいパスワードまたはパスチケットを提供した場合に、そのユーザー ID が取り消されたかパスワードの有効期限が切れているかをユーザーに通知します。

ESI の利点

APPC PEM を使用するには、 APPC セッションがリンクした PEM クライアント (リクエスター) および PEM サーバーが必要です。また、リソース・アクセス管理機能 (RACF®) などの外部セキュリティー・マネージャー (ESM) または同等の ESM が PEM サーバーで使用可能でなければなりません。

ESI は、以下の関数を提供します。

- **CICS_VerifyPassword** 関数。これによって、クライアント・アプリケーションは、指定されたユーザー ID について、あるパスワードが ESM の記録したパスワードと一致していることを検査することができます。
- **CICS_ChangePassword** 関数。これによって、クライアント・アプリケーションは、特定のユーザー ID について ESM の記録したパスワードを変更することができます。
- **CICS_SetDefaultSecurity** 関数。これによって、クライアント・アプリケーションは、デフォルトのユーザー ID とパスワードをサーバーに対する ECI および EPI 要求に使用することができます。

これらの関数によって、アプリケーション・プログラマーが APPC 会話を管理しなくても、非 CICS アプリケーション・プログラムが PEM リクエスターとして機能することができます。このため、PEM の要求と応答の形式およびローカル PEM サーバーに対するインターフェースについて知っている必要がありません。

ESI の定数およびデータ構造

このセクションでは、 ESI を使用するために必要な定数とデータ構造について説明します。これらについては、146ページの『ESI 関数』に記載されています。

ESI 定数

以下に挙げた定数は、本章の ESI データ構造および関数の説明で、記号名で参照されています。これらの定数の値も参考として挙げていますが、実際のプログラミングの際には、その言語用に ESI が提供している記号名を使用してください。

フィールド長

- CICS_ESI_PASSWORD_MAX (10)
- CICS_ESI_SYSTEM_MAX (8)
- CICS_ESI_USERID_MAX (10)

ESI データ構造

ESI で使用できる構造を、以下に示します。

- **CICS_EsiDate_t**
- **CICS_EsiTime_t**
- **CICS_EsiDetails_t**

これらの構造のストリング・フィールドはすべてヌルで終了します。

CICS_EsiDate_t

目的: **CICS_EsiDate_t** 構造には、年、月、および日で表された日付が含まれます。

フィールド:

Year 4 桁の年が **cics_ushort_t** 形式で保持されます。

Month 月が **cics_ushort_t** 形式で保持されます。値の範囲は 1 から 12 で、1 が 1 月を表します。

Day 日が **cics_ushort_t** 形式で保持されます。値の範囲は 1 から 31 で、1 がその月の最初の日を表します。

CICS_EsiTime_t

目的: **CICS_EsiTime** 構造には、時間、分、秒、および 100 分の 1 秒で表された時刻が含まれます。

フィールド:

Hours 時間が **cics_ushort_t** 形式で保持されます。値の範囲は 0 から 23 です。

Minutes 分が **cics_ushort_t** 形式で保持されます。値の範囲は 0 から 59 です。

Seconds 秒が **cics_ushort_t** 形式で保持されます。値の範囲は 0 から 59 です。

Hundredths 100 分の 1 秒が **cics_ushort_t** 形式で保持されます。値の範囲は 0 から 99 です。

CICS_EsiDetails_t

目的: **CICS_EsiDetails_t** 構造には、**CICS_VerifyPassword** 関数または **CICS_ChangePassword** 関数の正常な呼び出しから戻された情報が含まれます。

フィールド:

LastVerifiedDate
パスワードが最後に検査された日付。

LastVerifiedTime
パスワードが最後に検査された時刻。

ExpiryDate
パスワードの満了日。

ExpiryTime
パスワードの満了時刻。

LastAccessDate
ユーザー ID に最後にアクセスした日付。

LastAccessTime
ユーザー ID に最後にアクセスした時刻。

InvalidCount
ユーザー ID に無効なパスワードが入力された回数。

ESI 関数

このセクションでは、アプリケーション・プログラムから呼び出すことができる、以下の ESI 関数について説明します。

- **CICS_VerifyPassword**
- **CICS_ChangePassword**
- **CICS_SetDefaultSecurity**

CICS_VerifyPassword

CICS_VerifyPassword	**UserId** **Password** **System** **Details**

目的

CICS_VerifyPassword 関数によって、クライアント・アプリケーションは、指定されたユーザー ID について、あるパスワードが外部セキュリティー・マネージャーの記録したパスワードと一致しているかどうかを検査することができます。

外部セキュリティー・マネージャーは、クライアントの接続先サーバーにあると想定されるので注意してください。

パラメーター

UserId

パスワードを検査するユーザー ID を指定する、ヌル文字で終了するストリングを指すポインターです。ユーザー ID の文字数が CICS_ESI_USERID_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_USERID_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

Password

指定されたユーザー ID に対して外部セキュリティー・マネージャーが検査するパスワードを指定する、ヌル文字で終了するストリングを指すポインターです。パスワードの文字数が CICS_ESI_PASSWORD_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_PASSWORD_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

System

パスワードを検査するサーバーの名前を指定する、ヌルで終了するストリングを指すポインターです。サーバー名の文字数が、CICS_ESI_SYSTEM_MAX の値より短い場合は、残りにヌ

ルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_SYSTEM_MAX + 1 になります。

ストリングがすべてヌルの場合には、現行のデフォルト・サーバーが選択されます。

ESI は、このパラメーターを入力用のみに使用します。

Details

外部セキュリティー・マネージャーによって戻される詳細情報が戻り時に入れられる **CICS_EsiDetails_t** 構造を指すポインターです。

ESI は、この構造の各フィールドを出力用のみに使用します。

戻りコード

CICS_ESI_NO_ERROR
関数の処理が正常に完了しました。

CICS_ESI_ERR_CALL_FROM_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_ESI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。

CICS_ESI_ERR_NO_CICS
クライアントが使用できないか、または指定されたサーバーが使用できません。

CICS_ESI_ERR_CICS_DIED
指定されたサーバーは使用できなくなりました。

CICS_ESI_ERR_RESOURCE_SHORTAGE
クライアントに、要求の完了に必要なリソースがありませんでした。

CICS_ESI_ERR_NO_SESSIONS
アプリケーションに、構成によってサポートされる数の未解決の ECI 要求および EPI 要求があります。

CICS_ESI_ERR_UNKNOWN_SERVER
要求対象のサーバーが見つかりません。**CICS_EciListSystems** および **CICS_EpiListSystems** 関数で戻されたサーバーのみが受け入れ可能です。

CICS_ESI_ERR_MAX_SESSIONS
要求の実行に必要な通信リソースが不足しています。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

CICS_ESI_ERR_MAX_SYSTEMS
構成の上限以上の数のサーバーに対する要求が行われました。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

CICS_ESI_ERR_NULL_USERID
ユーザー ID がヌルに設定されています。

CICS_ESI_ERR_NULL_PASSWORD
パスワードがヌルに設定されています。

CICS_ESI_ERR_PEM_NOT_SUPPORTED
パスワード有効期限管理は、要求されたサーバーとの SNA および TCP62 を介した通信でしかサポートされていません。

CICS_ESI_ERR_PEM_NOT_ACTIVE
要求されたサーバーがパスワード有効期限管理をサポートしていません。

CICS_ESI_ERR_PASSWORD_EXPIRED
パスワードの有効期限が切れています。

CICS_ESI_ERR_PASSWORD_INVALID
パスワードが無効です。

CICS_ESI_ERR_USERID_INVALID
このユーザー ID は外部セキュリティー・マネージャーに認識されません。

CICS_ESI_ERR_SECURITY_ERROR
外部セキュリティー・マネージャーによってエラーが検出されました。ほとんどの場合、ユーザー ID が呼び出されたことが原因であると考えられます。

実際の戻りコード値を記号名にマッピングしたものが、 CICS ユニバーサル・クライアント (Windows 版) の以下のファイルに含まれています。

C ¥include¥cics_esi.h

また、CICS ユニバーサル・クライアント (AIX 版) および CICS ユニバーサル・クライアント (Solaris 版) の以下のファイルに含まれています。

C /include/cics_esi.h

CICS_ChangePassword

CICS_ChangePassword	UserId
	OldPassword
	NewPassword
	System
	Details

目的

CICS_ChangePassword 関数によって、クライアント・アプリケーションは、指定されたユーザー ID の外部セキュリティー・マネージャーによって記録されたパスワードを変更することができます。

外部セキュリティー・マネージャーは、クライアントの接続先サーバーにあると想定されるので注意してください。

パラメーター

UserId

パスワードを変更するユーザー ID を指定する、ヌル文字で終了するストリングを指すポインターです。ユーザー ID の文字数が CICS_ESI_USERID_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_USERID_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

OldPassword

指定されたユーザー ID に対する現行のパスワードを指定する、ヌル文字で終了するストリングを指すポインターです。パスワードの文字数が CICS_ESI_PASSWORD_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_PASSWORD_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

NewPassword

指定されたユーザー ID に対する新規のパスワードを指定する、ヌル文字で終了するストリングを指すポインターです。パスワードの文字数が CICS_ESI_PASSWORD_MAX の値より短

い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_PASSWORD_MAX+1 になります。

パスワードは、現行のパスワードが正しく指定されている場合に限り変更されます。

ESI は、このパラメーターを入力用のみに使用します。

System

パスワードを検査するサーバーの名前を指定する、ヌルで終了するストリングを指すポインターです。サーバー名の文字数が、CICS_ESI_SYSTEM_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_SYSTEM_MAX + 1 になります。

ストリングがすべてヌルの場合には、現行のデフォルト・サーバーが選択されます。

ESI は、このパラメーターを入力用のみに使用します。

Details

外部セキュリティー・マネージャーによって戻される詳細情報が戻り時に入れられる **CICS_EsiDetails_t** 構造を指すポインターです。

ESI は、この構造の各フィールドを出力用のみに使用します。

戻りコード

CICS_ESI_NO_ERROR
関数の処理が正常に完了しました。

CICS_ESI_ERR_CALL_FROM_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_ESI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。

CICS_ESI_ERR_NO_CICS
クライアントが使用できないか、または指定されたサーバーが使用できません。

CICS_ESI_ERR_CICS_DIED
指定されたサーバーは使用できなくなりました。パスワードが変更されたことを確認するには、**CICS_VerifyPassword** 関数を使用します。

CICS_ESI_ERR_RESOURCE_SHORTAGE
クライアントに、要求の完了に必要なリソースがありませんでした。

CICS_ESI_ERR_NO_SESSIONS
アプリケーションに、構成によってサポートされる数の未解決の ECI 要求および EPI 要求があります。

CICS_ESI_ERR_UNKNOWN_SERVER
要求対象のサーバーが見つかりません。**CICS_EciListSystems** および **CICS_EpiListSystems** 関数で戻されたサーバーのみが受け入れ可能です。

CICS_ESI_ERR_MAX_SESSIONS
要求の実行に必要な通信リソースが不足しています。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

CICS_ESI_ERR_MAX_SYSTEMS
構成の上限以上の数のサーバーに対する要求が行われました。使用可能なサーバー数については、クライアントまたはサーバーの関連資料を参照してください。

CICS_ESI_ERR_NULL_USERID
ユーザー ID がヌルに設定されています。

CICS_ESI_ERR_NULL_OLD_PASSWORD
現行のパスワードがヌルに設定されています。

CICS_ESI_ERR_NULL_NEW_PASSWORD
新規のパスワードがヌルに設定されています。

CICS_ESI_ERR_PEM_NOT_SUPPORTED
パスワード有効期限管理は、要求されたサーバーとの SNA および TCP62 を介した通信でしかサポートされていません。

CICS_ESI_ERR_PEM_NOT_ACTIVE
要求されたサーバーがパスワード有効期限管理をサポートしていません。

CICS_ESI_ERR_PASSWORD_INVALID
パスワードが無効です。

CICS_ESI_ERR_PASSWORD_REJECTED
新しいパスワードが、外部セキュリティー・マネージャーに定義された標準に適合していません。

CICS_ESI_ERR_USERID_INVALID
このユーザー ID は外部セキュリティー・マネージャーに認識されません。

CICS_ESI_ERR_SECURITY_ERROR
外部セキュリティー・マネージャーによってエラーが検出されました。ほとんどの場合、ユーザー ID が呼び出されたことが原因であると考えられます。

実際の戻りコード値を記号名にマッピングしたものが、 CICS ユニバーサル・クライアント (Windows 版) の以下のファイルに含まれています。

C ¥include¥cics_esi.h

また、CICS ユニバーサル・クライアント (AIX 版) および CICS ユニバーサル・クライアント (Solaris 版) の以下のファイルに含まれています。

C /include/cics_esi.h

CICS_SetDefaultSecurity

CICS_SetDefaultSecurity	UserId Password System

目的

CICS_SetDefaultSecurity 関数。これによって、クライアント・アプリケーションは、デフォルトのユーザー ID とパスワードをサーバーに対する ECI および EPI 要求に使用することができます。ユーザー ID および暗黙のパスワードをヌルに設定することができます。

クライアント・アプリケーションは、ユーザー ID とパスワードを検査しなければなりません。

ユーザー ID とパスワードが必要になった場合は、いくつかの場所のうちの 1 つからそれらを入手することができます。この場合、CICS ユニバーサル・クライアントが以下の検索順序を使用することを前提にしています。

1. ECI の ECI パラメーター・ブロックまたは、**CICS_EpiSetSecurity** 関数によって設定された端末固有値のいずれか。
2. **CICS_SetDefaultSecurity** 関数によって設定されたサーバー固有の値。
3. クライアントのポップアップ・ウィンドウなどから入手したデフォルト、たとえば、Windows NT ユーザー ID。

パラメーター

UserId

設定するユーザー ID を指定する、ヌル文字で終了するストリングを指すポインターです。ユーザー ID の文字数が CICS_ESI_USERID_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_USERID_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

Password

指定されたユーザー ID に対して設定するパスワードを指定する、ヌル文字で終了するストリングを指すポインターです。パスワードの文字数が CICS_ESI_PASSWORD_MAX の値より短

い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_PASSWORD_MAX+1 になります。

ESI は、このパラメーターを入力用のみに使用します。

System

パスワードとユーザー ID が設定されるサーバーの名前を指定する、ヌルで終了するストリングを指すポインターです。サーバー名の文字数が、CICS_ESI_SYSTEM_MAX の値より短い場合は、残りにヌルが格納されます。ストリングはヌルで終了するため、長さは CICS_ESI_SYSTEM_MAX + 1 になります。

ストリングがすべてヌルの場合には、現行のデフォルト・サーバーが選択されます。

ESI は、このパラメーターを入力用のみに使用します。

戻りコード

CICS_ESI_NO_ERROR
関数の処理が正常に完了しました。

CICS_ESI_ERR_CALL_FROM_CALLBACK
関数がコールバック・ルーチンから呼び出されました。

CICS_ESI_ERR_SYSTEM_ERROR
内部システム・エラーが発生しました。

CICS_ESI_ERR_NO_CICS
クライアントが使用できないか、または指定されたサーバーが使用できません。

CICS_ESI_ERR_UNKNOWN_SERVER
要求対象のサーバーが見つかりません。**CICS_EciListSystems** および **CICS_EpiListSystems** 関数で戻されたサーバーのみが受け入れ可能です。

CICS_ESI_ERR_USERID_INVALID
ユーザー ID の長さが CICS_ESI_USERID_MAX を超えています。

CICS_ESI_ERR_PASSWORD_INVALID
パスワードの長さが CICS_ESI_PASSWORD_MAX を超えています。

実際の戻りコード値を記号名にマッピングしたものが、 CICS ユニバーサル・クライアント (Windows 版) の以下のファイルに含まれています。

C ¥include¥cics_esi.h

また、CICS ユニバーサル・クライアント (AIX 版) および CICS ユニバーサル・クライアント (Solaris 版) の以下のファイルに含まれています。

C /include/cics_esi.h

付録A. 環境に依存する ECI の拡張関数

本章では、特定の環境のみがサポートする ECI の拡張関数について説明します。**これらの拡張関数は、「CICS ファミリー : クライアント・サーバー プログラミングの手引き」には含まれません。**

本章の構成は以下のとおりです。

拡張呼び出しタイプ

非同期呼び出しの拡張呼び出しタイプを以下に示します。

プログラム・リンク呼び出しの詳細については、 164ページの表5 および 30ページの『ECI_ASYNC 呼び出しタイプ』を参照してください。

状況情報呼び出しの詳細については、 164ページの表5 および 42ページの『ECI_STATE_ASYNC 呼び出しタイプ』を参照してください。

メッセージによる通知を行う非同期プログラム・リンク呼び出し (ECI_ASYNC_NOTIFY_MSG)

この呼び出しは、windows 上で稼働するプログラムのみで使用可能です。

呼び出し側のアプリケーションの処理は、ECI が要求を受け取りしだい再開されます。ただし、プログラムは稼動を開始するわけではなく、パラメーターの検証のみが行われます。後で処理するため要求がキューに入る場合もあります。

ECI は、応答が使用可能になりしだい、指定したウィンドウに通知メッセージを送信します。(メッセージ形式の詳細については、163ページの『応答メッセージの形式』を参照してください。) 呼び出し側のアプリケーションでは、こ

の通知メッセージの受信後、ECI_GET_REPLY または ECI_GET_SPECIFIC_REPLY を使用して実際の応答を受信する必要があります。

メッセージによる通知を行う場合、以下のフィールドが必要なパラメーターです。

- **eci_async_notify.window_handle**
- **eci_message_id**: 通知処理で使用するメッセージのタイプを示します。

eci_message_qualifier を入力パラメーターとして使用して、呼び出しのユーザー定義名を指定できます。 Windows 環境では、通知メッセージの一部として戻されます。

セマフォーによる通知を行う非同期プログラム・リンク呼び出し (ECI_ASYNC_NOTIFY_SEM)

この呼び出しは、Windows 上で稼働するプログラムのみで使用可能です。

呼び出し側のアプリケーションの処理は、ECI が要求を受け取りしだい再開されます。ただし、プログラムは稼動を開始するわけではなく、パラメーターの検証のみが行われます。後で処理するため要求がキューに入る場合もあります。

ECI は、応答が使用可能になると、指定したセマフォーをポストします。呼び出し側のアプリケーションでは、この通知メッセージの受信後、ECI_GET_REPLY または ECI_GET_SPECIFIC_REPLY を使用して実際の応答を受信する必要があります。

eci_message_qualifier を入力パラメーターとして使用して、呼び出しのユーザー定義名を指定できます。

セマフォーで通知を行う場合、以下のフィールドが必要なパラメーターです。

- **eci_async_notify.sem_handle**: セマフォーを参照します。

メッセージによる通知を行う非同期状況呼び出し (ECI_STATE_ASYNC_MSG)

この呼び出しタイプは、Windows 上で稼働するプログラムのみで使用可能です。

eci_message_qualifier を入力パラメーターとして使用して、呼び出しのユーザー定義名を指定できます。

ECI は、応答が使用可能になると、指定したウィンドウに通知メッセージを送信します。 (メッセージ形式の詳細については、163ページの『応答メッセージの形式』を参照してください。) 呼び出し側のアプリケーションでは、この通知メッセージの受信後、ECI_GET_REPLY または ECI_GET_SPECIFIC_REPLY を使用して実際の応答を受信する必要があります。

メッセージによる通知に関連する追加パラメーターの詳細については、ECI_ASYNC_NOTIFY_MSG タイプの呼び出しの説明を参照してください。

セマフォーによる通知を行う非同期状況呼び出し (ECI_STATE_ASYNC_SEM)

この呼び出しは、Windows 上で稼働するプログラムのみで使用可能です。

eci_message_qualifier を入力パラメーターとして使用して、呼び出しのユーザー定義名を指定できます。

ECI は、応答が使用可能になると、指定したセマフォーをポストします。呼び出し側のアプリケーションでは、この通知メッセージの受信後、ECI_GET_REPLY または ECI_GET_SPECIFIC_REPLY を使用して実際の応答を受信する必要があります。

セマフォーで通知を行う場合、以下のフィールドが必要なパラメーターです。

- **eci_async_notify.sem_handle**: セマフォーを参照します。

タイムアウト機能

この機能では、ECI パラメーター・ブロックの **eci_timeout** を使用します。戻りコードは、ECI_ERR_RESPONSE_TIMEOUT および ECI_ERR_REQUEST_TIMEOUT の 2 つです。

タイムアウト処理では、以下の 2 つの点を留意する必要があります。

- サーバーへの転送の完了前にタイムアウトが発生する場合は、応答は ECI_ERR_REQUEST_TIMEOUT になり、要求したプログラムの呼び出しは実行されず、サーバー上のリソースも更新されません。
  - 新規の作業論理単位を開始する呼び出しの場合、または作業論理単位全体への呼び出しの場合には、作業論理単位は開始されず、リカバリー可能リソースの更新も行われません。
  - 既存の作業論理単位を継続する呼び出しの場合には、作業論理単位は継続されますが、リカバリー可能リソースの更新は行われず、作業論理単位は依然としてコミットされません。

- 既存の作業論理単位を終了する呼び出しの場合は、対象の作業論理単位は継続されます。リカバリー可能リソースの更新は行われず、作業論理単位は依然としてコミットされません。

- サーバーへの転送後にタイムアウトが発生した場合には、応答は ECI_ERR_RESPONSE_TIMEOUT になります。この応答は、同期呼び出し、非同期呼び出し、または非同期呼び出しの応答を獲得する応答請求要求に戻されます。
  - 新規の作業論理単位に対する単一呼び出しの場合は、その作業論理単位が開始されます。ただし、リソースの更新が行われたかどうかおよび更新がコミットされたか、バックアウトされたかアプリケーション側では判別できません。
  - ECI_NO_EXTEND を使った既存の作業論理単位を終了する呼び出しの場合には、対象の作業論理単位が終了します。ただし、リソースの更新が行われたかどうかおよび更新がコミットされたか、バックアウトされたかアプリケーション側では判別できません。
  - ECI_COMMIT を使った既存の作業論理単位を継続する呼び出しの場合は、対象の作業論理単位は継続されますが、リカバリー可能リソースへの変更は依然として保留されています。

ECI 拡張関数用フィールド

ECI パラメーター・ブロックの以下のフィールドは、環境に依存する拡張関数をサポートします。

eci_async_notify.window_handle

(Windows 環境では、ECI_ASYNC_NOTIFY_MSG および ECI_STATE_ASYNC_MSG 呼び出しタイプ)

応答メッセージをポストするウィンドウのハンドルです。

ECI は、このフィールドを入力にのみ使用します。

注: eci_window_handle を代わりに使用することもできます。

eci_async_notify.sem_handle

(Windows 環境では、ECI_ASYNC_NOTIFY_SEM および ECI_STATE_ASYNC_SEM 呼び出しタイプ)

Windows アプリケーションでは、イベント・オブジェクト処理を渡します。

ECI は、このフィールドを入力にのみ使用します。

eci_async_notify.win_fields.hwnd

応答メッセージをポストする Microsoft Windows NT または Windows 2000 のウィンドウのハンドルです。

ECI は、このフィールドを入力にのみ使用します。

eci_async_notify.win_fields.hinstance

プログラムの初期化の際に提供される、呼び出し側プログラムの Microsoft Windows hInstance です。

ECI は、このフィールドを入力にのみ使用します。

eci_sync_wait.hwnd

同期呼び出し中に使用不能になるウィンドウ・ハンドルです。

ECI は、このフィールドを入力にのみ使用します。

eci_message_id

(Windows 環境では、ECI_ASYNC_NOTIFY_MSG および ECI_STATE_ASYNC_MSG 呼び出しタイプ)

関連するウィンドウ・ハンドルで指定するウィンドウへの応答メッセージをポストするのに使用されるメッセージ ID です。

ECI は、このフィールドを入力にのみ使用します。

eci_timeout

(プログラム・リンク呼び出しおよび状況呼び出しのみで使用)

アプリケーションからの要求が取ることができる最大時間 (秒) を指定する整数フィールドです。要求の提供が指定時間を超過するとタイムアウトが発生します。指定時間の範囲は、0 から 32767 までです。0 を指定すると、アプリケーションが要求をサービスする時間は無制限になります。

ECI は、このフィールドを入力にのみ使用します。

応答メッセージの形式

アプリケーションが、メッセージで通知を要求する非同期要求の場合、ECI は指定されたウィンドウ・ハンドルおよびメッセージ ID を使用して、結果情報を含むメッセージをウィンドウに戻します。

Windows 環境では、メッセージは以下の 2 つのパラメーターに分割されます。

wParam

高位 16 ビット
指定メッセージ修飾子

低位 16 ビット
戻りコード

lParam
4 文字の異常終了コード (異常終了した場合のみ)

ECI 戻りコードと通知内容

表 5. CICS_ExternalCall の戻りコード -- 環境依存の拡張

戻りコード	意味
ECI_ERR_NULL_WIN_HANDLE	ウィンドウ・ハンドルを 0 にセットした非同期呼び出しが指定された。
ECI_ERR_NULL_MESSAGE_ID	メッセージ修飾子を 0 にセットした非同期呼び出しが指定された。
ECI_ERR_NULL_SEM_HANDLE	有効なハンドルの要求に対し、ヌル・セマフォー・ハンドルが渡された。
ECI_ERR_REQUEST_TIMEOUT	要求の処理終了前に指定タイムアウト時間間隔が満了した。またはタイムアウト時間間隔に負の値を指定した。
ECI_ERR_RESPONSE_TIMEOUT	プログラム稼動中に指定タイムアウト時間間隔が満了した。

入力パラメーター指定について

166ページの表6 に、ECI 拡張呼び出しと呼び出しタイプの入力パラメーターを示します。それについて、必要なパラメーターは ″R″、オプションの場合は ″O″、適用なしの場合は ″-″ で示されます。オプションまたは適用なしのパラメーターについては、初期値にヌルを指定してください。″R″ の後にアスタリスク (*) が付加されているパラメーターについては、対応する説明部分で適用法の詳細が説明されています。

表の**パラメーター**欄では、以下の省略形が使われます。

AN **async_notify**

WF **win_fields**

SW **sync_wait**

また、表中のパラメーターには、実際には最初に **eci_** が付加されます。したがって、表中で **AN.WF.hwnd** と表記されているパラメーターは、**eci_async_notify.win_fields.hwnd** を意味します。

表中の欄見出しで使用されている 3 文字の省略形は、それぞれ以下の呼び出しタイプを示します。

ANM ECI_ASYNC_NOTIFY_MSG

ANE ECI_ASYNC_NOTIFY_SEM

SAM ECI_STATE_ASYNC_MSG

SAE ECI_STATE_ASYNC_SEM

SYN ECI_SYNC

SSN ECI_STATE_SYNC

表 6. CICS_ExternalCall の入力パラメーター -- 環境依存の拡張

パラメーター (最初に eci_ を付加)	ANM	ANE	SAM	SAE	SYN	SSN
call_type	R	R	R	R	R	R
program_name	R*	R*	-	-	R*	-
userid	R	R	-	-	R	-
password	R	R	-	-	R	-
transid	O	O	-	-	O	-
commarea	O	O	R*	R*	O	R*
commarea_length	O	O	R*	R*	O	R*
timeout	O	O	O	O	O	O
extend_mode	R	R	R	R	R	R
AN.window_handle	R*	-	R*	-	-	-
AN.sem_handle	-	R	-	R	-	-
AN.WF.hwnd	R*	-	R*	-	-	-
AN.WF.hinstance	R*	-	R*	-	-	-
SW.hwnd	-	-	-	-	R*	R*
message_id	R	-	R	-	-	-
message_qualifier	O	O	O	O	O	O
luw_token	R	R	R*	R*	R	R*
version	O	O	O	O	O	O
system_name	O	O	O	O	O	O

付録B. CICS ユニバーサル・クライアントのプログラミング・サンプル

本章では、 CICS ユニバーサル・クライアントでクライアント / サーバー・プログラミングを行うためのサンプルを要約しています。サンプルは、C および C++ 言語で提供されています。

サンプル・コードは、可能なかぎり、言語標準 (たとえば、ANSI C) に準拠しています。プラットフォーム固有の GUI コードへの依存を回避するために、サンプルはすべてコマンド行から起動されます。

各言語ごとにそれぞれ異なるレベルのサンプルが提供されます。サンプルには、以下のものが含まれます。

- 簡単なサンプル。これを使用すれば、クライアントが機能しているかどうかをテストしたり、クライアント・アプリケーションの基本要件を直観的に理解できます。
- より複雑なサンプル。より高機能で有用な API 機能のいくつかを実演し、より実際的なクライアント・アプリケーションの例を提供します。

ECI、EPI、および ESI のサンプルが提供されます。

各プラットフォームでサポートされる言語とコンパイラーについては、該当する *CICS クライアント 管理の手引き* を参照してください。

サンプルは、プロダクト・ルート・ディレクトリーのサンプル・サブディレクトリーに提供されます。サンプル・ディレクトリーの下には、各プログラム言語のためのサブディレクトリーが設けられています (たとえば、¥samples¥c)。

サンプルについての詳しい説明が、コンパイルの説明やコンパイラーに関する考慮事項の説明とともに、サンプル・ディレクトリー内の `Samples.txt` ファイルに収録されています。

付録C. ECI および EPI 出口

本章では、CICS ユニバーサル・クライアントを使用するときに EPI および ECI に追加できる出口について説明します。これらの出口を使用すると、特定のアプリケーション要求に対する ECI および EPI 呼び出しの処理に影響を及ぼします。

本章の構成は以下のとおりです。

本章には、プロダクト・センシティブ・プログラミング・インターフェースおよび関連ガイダンス情報が含まれています。

出口のインストール

ECI および EPI の初期設定時に、クライアントは、表7 に示すオブジェクトを CICS バイナリー・ディレクトリーからロードし、対応する入り口点を呼び出そうとします。

表 7. ECI および EPI 出口

	オブジェクト名	入り口点名
ECI	**cicseciexit**	**CICS_EciExitInit**
EPI	**cicsepiexit**	**CICS_EpiExitInit**

各入り口点では、単一のパラメーターが渡されます。このパラメーターは、アドレスのリストが入る構造を指すポインターです。プログラムは初期設定で、すべての出口のアドレスを構造体に入れます。 ECI および EPI の処理が開始すると、該当する地点で出口が呼び出されます。出口の呼び出しは指定されたアドレスを使って行われるので、出口には任意の名前を割り当てることができます。ただし、本書では、出口に標準の名前を使用しています。

オブジェクトが見つからないと、出口処理は行われません。

CICS クライアントには以下のファイルが提供され、出口をプログラミングする際に役立ちます。

include¥cicsecix.h
以下の定義を含むヘッダー・ファイル。
- 各 ECI 出口の入出力
- ECI 出口の呼び出しに使用するアドレス・リストの形式
- ECI 出口が使用するデータ構造
- ECI 出口からの戻りコードの値

samples¥c¥cicsecix.c
cicsecix のスケルトン・コードです。

samples¥c¥cicsecix.def
cicsecix の定義ファイルです。

samples¥cicsecix.mak
cicsecix を再構築する MAKE ファイルです。

include¥cicsepix.h
以下の定義を含むヘッダー・ファイル。
- 各 EPI 出口の入出力
- EPI 出口の呼び出しに使用するアドレス・リストの形式
- EPI 出口が使用するデータ構造
- EPI 出口からの戻りコードの値

samples¥c¥cicsepix.c
cicsepix のスケルトン・コードです。

samples¥c¥cicsepix.def
cicsepix の定義ファイルです。

samples¥c¥cicsepix.mak
cicsepix を再構築する MAKE ファイルです。

(UNIX® オペレーティング・システムでは、パス名は UNIX® 規約に従います。)

注: ユーザーがユーザー出口 DLL を作成する場合には、以下の例外を除いて、すべての出口をむ組み込む必要があります。
- **CICS_EciSetProgramAlias** はオプションです。

- **CICS_EpiTermIdExit** または **CICS_EpiTermIdInfoExit** のどちらかが組み込まれている必要があります。ただし、すべての新規 DLL は、**CICS_EpiTermIdInfoExit** を使用する必要があります。

出口ルーチン環境

ユーザー出口には、以下の制約があります。

- EPI または ECI 呼び出しを行ってはなりません。
- 待機状態または実行時間の長いコードはコーディングしないでください。(アプリケーション・ユーザー・インターフェースのスレッドで出口が稼働しているときに、戻りに遅延が生じると、システムの応答性に悪影響を及ぼす場合があります。)

出口ルーチンの解説

出口ルーチンの解説は、以下の見出しから構成されています。

- **目的** -- 出口が行う処理の種類を説明します。
- **呼び出し地点** -- ECI または EPI 処理で出口が呼び出される地点を説明します。
- **パラメーター** -- 出口に渡されるパラメーターを説明します。パラメーターの種類は、以下のとおりです。
  - 入力 -- このパラメーターは出口で参照できますが、変更してはいけません。
  - 出力 -- このパラメーターは出口では参照しません。値を保管できます。
  - 入出力 -- このパラメーターは出口で参照でき、値を保管できます。
- **戻りコード** -- 出口が ECI または EPI に戻す値について説明します。各戻り値に対する ECI または EPI の対応についても説明します。

ECI 出口の解説

このセクションでは、以下の出口について説明します。

- **CICS_EciInitializeExit**
- **CICS_EciTerminateExit**
- **CICS_EciExternalCallExit1**
- **CICS_EciExternalCallExit2**
- **CICS_EciSystemIdExit**
- **CICS_EciDataSendExit**
- **CICS_EciDataReturnExit**

- **CICS_EciSetProgramAliasExit**

表8 に、出口名、各出口に渡されるパラメーター、および戻りコードの要約を示します。

表 8. ECI の要約

関数名	パラメーター	戻りコード
CICS_EciInitializeExit	**Version** **Anchor**	CICS_EXIT_OK CICS_EXIT_NO_EXIT CICS_EXIT_CANT_INIT_EXITS ユーザー定義
CICS_EciTerminateExit	**Anchor**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_STORAGE ユーザー定義
CICS_EciExternalCallExit1	**Anchor** **Token** **ParmPtr**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EciExternalCallExit2	**Anchor** **Token** **ParmPtr**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EciSystemIdExit	**Anchor** **Token** **ParmPtr** **Reason**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM CICS_EXIT_GIVE_UP ユーザー定義
CICS_EciDataSendExit	**Anchor** **Token**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EciDataReturnExit	**Anchor** **Token** **ParmPtr**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EciSetProgramAliasExit	**Anchor** **EciParms** **Program**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義

識別トークン

同じ ECI 要求に対する呼び出しと出口を関連付けるために、**CICS_EciInitializeExit** と **CICS_EciTerminateExit** を除くすべての出口に、パラメーターとして*識別トークン* が渡されます。同じ呼び出しに関連する **CICS_EciExternalCallExit1** および **CICS_EciExternalCallExit2** には、同じトークンが使用されます。また、**CICS_EciDataSendExit**、**CICS_EciDataReturnExit**、および **CICS_EciSystemIdExit** の各出口の介入でも同じトークンが使用されます。(**CICS_EciExternalCallExit1** および **CICS_EciExternalCallExit2** は、応答請求要求に対しては呼び出されません。)

トークンは、要求が実行されている間、その要求が開始されたオペレーティング・システム内で固有です。要求に対する最後の出口が呼び出された後で、トークンは再使用することができます。

拡張作業論理単位の場合、作業論理単位内の要求ごとにトークンが異なる場合があります。(トークンの再利用は可能ですが、前の非同期要求に対する ECI_GET_REPLY 要求が完了するまで新しいプログラム・リンク呼び出しが行われないので、トークンは同じ場合もあります。)

トークンの長さは、8 バイトです。8 バイトすべてヌル文字のトークンは無効で、出口には渡されません。

プロセス・モデルの実現

特定の要求 (同じ識別トークンをもつものなど) に関連する出口はすべて、アプリケーション・プロセスのコンテキストで呼び出されます。

CICS_EciInitializeExit

CICS_EciInitializeExit	Version Anchor

目的: 出口環境をセットアップします。

呼び出し地点: 各プロセスの **CICS_ExternalCall** の最初の呼び出し時に、パラメーターの妥当性検査が発生した後呼び出されます。

パラメーター:

Version 入力パラメーターです。出口の稼働する ECI のバージョンを表します。

Anchor 出力パラメーターです。ECI 出口に渡されるポインターを指すポインターです。2 番目のポインターは ECI には使用されません。そのまま出口に渡されます。この出口でのストレージを獲得して、そのアドレスを他の出口に渡すことができます。

戻りコード:

CICS_EXIT_OK
ECI は、適切な出口を呼び出して、この要求の処理を継続します。

CICS_EXIT_NO_EXIT
ECI は、他の出口は呼び出さずに、この要求の処理を継続します。

CICS_EXIT_CANT_INIT_EXITS
ECI は、 CICS クライアントのトレース・レコードを作成してからこの要求の処理を続行しますが、これ以上出口を呼び出しません。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、 CICS クライアントのトレース・レコードを作成してからこの要求の処理を続行しますが、これ以上出口を呼び出しません。

CICS_EciTerminateExit

CICS_EciTerminateExit	Anchor

目的: 出口環境の終結処理を行います。**CICS_EciInitializeExit** で獲得したストレージは、この出口ですべて解放しなければなりません。

CICS_EciTerminateExit は、CICS ユニバーサル・クライアントによって呼び出されません。

呼び出し地点: **CICS_EciInitializeExit** を出したプロセスの終了時に呼び出されます。

パラメーター:

Anchor
入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

戻りコード:

CICS_EXIT_OK
終了処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、CICS クライアントのトレース・レコードを作成してから、終了処理を続行します。

CICS_EXIT_BAD_STORAGE
CICS は、ストレージ・エラーを検出しました。 ECI は、CICS クライアントのトレース・レコードを作成してから、終了処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、 CICS クライアントのトレース・レコードを作成してから、終了処理を続行します。

CICS_EciExternalCallExit1

CICS_EciExternalCallExit1	Anchor Token ParmPtr

目的: プログラムの稼働に最適なシステムを選択します。この出口は、各プログラム・リンクまたは状況情報呼び出し時に 1 回だけ呼び出されます。応答請求呼び出しでは呼び出されません。出口は **eci_luw_token** の値が 0 でない場合に呼び出されますが、作業論理単位が開始したときにサーバーが選択されているため、 **eci_system_name** への変更は無視されます。

呼び出し地点: 各プログラム・リンク呼び出しおよび状況情報呼び出しに対する **CICS_ExternalCall** で、ECI がパラメーターの妥当性検査を実行した後に呼び出されます。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

Token 入力パラメーターです。この要求に ECI が設定した識別トークンです。

ParmPtr 入力パラメーターです。ECI パラメーター・ブロックを指すポインターです。この出口では、ECI パラメーター・ブロックのすべてのフィールドを入力として扱う必要があります。ただし、**eci_system_name** フィールドは変更される可能性があるので、入力として扱いません。

戻りコード:

CICS_EXIT_OK
ECI は、現在 ECI パラメーター・ブロックに指定されている **eci_system_name** を使用して、要求の処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、CICS クライアントのトレース・レコードを作成してから、現在 ECI パラメーター・ブロックに指定されている **eci_system_name** からの要求の処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、CICS クライ

アントのトレース・レコードを作成してから、現在 ECI パラメーター・ブロックに指定されている **eci_system_name** からの要求の処理を続行します。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、 CICS クライアントのトレース・レコードを作成してから、現在 ECI パラメーター・ブロックに指定されている **eci_system_name** からの要求の処理を続行します。

注: 出口が新しいシステムを選択する場合、いくつかの制限事項があります。出口が新しいシステムを選択できるのは、呼び出しのタイプがプログラム・リンク呼び出しか状況情報呼び出しのいずれかで、新しい作業論理単位を開始する場合だけです。これ以外の場合には、出口は CICS_EXIT_OK を戻します。

呼び出し側のアプリケーションでシステム名としてパラメーター・ブロックに 2 進ゼロをセットすると、システムが動的に選択され、出口が確実にシステムを選択できるようになります。

ただし、呼び出し側のアプリケーションがシステム名をパラメーター・ブロックにセットした場合、またはアプリケーションのバージョンが 0 である場合には、ターゲット・システムが変更されることを期待していないので、アプリケーション・エラーが生じることがあります。この場合、出口が初めから置換システムを指定せずに制御を戻すと、指定の、またはデフォルトのシステム名が使用されることになります。このとき、選択したシステムを変更しようとすると、変更することはできますが、以下のことに注意してください。

- 出口ルーチンは、ターゲット・システムの修正によって、クライアントで稼働している ECI アプリケーションでエラーが生じるか否かに注意が必要です。
- 出口ルーチンは、この修正がクライアント・アプリケーションに受け入れられるかどうか判断するために、適切なデータが位置づけできる知識ベースを維持しなければなりません。

CICS_EciExternalCallExit2

CICS_EciExternalCallExit2	Anchor Token ParmPtr

目的: 同期 ECI 呼び出しの結果を調べます。これは、情報収集の目的のために行います。この出口は、各プログラム・リンクまたは状況情報呼び出し時に 1 回だけ呼び出されます。応答請求呼び出しでは呼び出されません。

呼び出し地点: ECI パラメーター・ブロックに戻りデータが入れられてから ECI 呼び出しがアプリケーションに制御を戻すまでの間に呼び戻されます。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

Token 入力パラメーターです。この要求に ECI が設定した識別トークンです。

ParmPtr 入力パラメーターです。ECI パラメーター・ブロックを指すポインターです。出口では、ECI パラメーター・ブロックのすべてのフィールドを入力として扱う必要があります。

戻りコード:

CICS_EXIT_OK
ECI は、**CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、CICS クライアントのトレース・レコードを作成してから、**CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、CICS クライアントのトレース・レコードを作成してから、 **CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が

入っていなければなりません。 ECI は、 CICS クライアントのトレース・レコードを作成してから、 **CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

CICS_EciSystemIdExit

CICS_EciSystemIdExit	Anchor Token ParmPtr Reason

目的: ECI パラメーター・ブロックのシステム名が無効な場合に、新しいシステム名を設定します。

呼び出し地点: この出口は、新しいシステム、ユーザー ID またはパスワードの選択で修正されるエラーが生じたときに呼び出されます。これは、以下のコードのいずれかを ECI が戻した場合です。

- ECI_ERR_NO_CICS
- ECI_ERR_UNKNOWN_SERVER
- ECI_ERR_SECURITY_ERROR
- ECI_ERR_SYSTEM_ERROR
- ECI_ERR_RESOURCE_SHORTAGE
- ECI_ERR_MAX_SYSTEMS

また、クライアントがサーバーにデータを送信する前にエラーを検出した場合、あるいはサーバーから戻されたデータにエラーがあった場合に、この出口が呼び出されます。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

Token 入力パラメーターです。この要求に ECI が設定した識別トークンです。

ParmPtr 入力パラメーターです。ECI パラメーター・ブロックを指すポインターです。この出口では、ECI パラメーター・ブロックのすべてのフィールドを入力として扱う必要があります。ただし、以下のフィールドは変更される可能性があるので、入力として扱いません。

- **eci_system_name**
- **eci_userid**
- **eci_password**

Reason 入力パラメーターです。アプリケーション要求が成功しなかった理由を示す理由コードです。

戻りコード:

CICS_EXIT_OK
ECI は、ECI パラメーター・ブロックの新しいパラメーターを使用してアプリケーション呼び出しを再度試みます。(アプリケーションが **CICS_ExternalCall** に提供した CICS プログラム連絡域は、保管されます。) アプリケーション・コールバック・ルーチンは呼び出されません。また、 **CICS_EciExternalCallExit2** も呼び出されません。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、CICS クライアントのトレース・レコードを作成してから、**CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、CICS クライアントのトレース・レコードを作成してから、 **CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

CICS_EXIT_GIVE_UP
ECI は、**CICS_ExternalCall** 要求を出したアプリケーションに制御を戻します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、 CICS クライアントのトレース・レコードを作成してから、 CICS_EXIT_OK で説明するアプリケーション呼び出しを再試行します。

CICS_EciDataSendExit

CICS_EciDataSendExit	Anchor Token

目的: パフォーマンス分析用の時刻呼び出しを行います。

呼び出し地点: 要求がサーバーに送信された直後に呼び出されます。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

Token 入力パラメーターです。この要求に ECI が設定した識別トークンです。

戻りコード:

CICS_EXIT_OK
ECI は、要求の処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

CICS_EciDataReturnExit

CICS_EciDataReturnExit	Anchor Token ParmPtr

目的: パフォーマンス分析用の時刻呼び出しを行います。

呼び出し地点: サーバーからの応答が受信され、アプリケーションへの戻りに使用される ECI ブロックおよび連絡域データが作成された直後に呼び出されます。サーバーから応答がなくタイムアウトになった場合にも、この出口は呼び出されます。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

Token 入力パラメーターです。この要求に ECI が設定した識別トークンです。

ParmPtr 入力パラメーターです。ECI パラメーター・ブロックを指すポインターです。出口では、ECI パラメーター・ブロックのすべてのフィールドを入力として扱う必要があります。

戻りコード:

CICS_EXIT_OK
ECI は、要求の処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

CICS_EciSetProgramAliasExit

CICS_EciSetProgramAliasExit	Anchor EciParms Program

目的: CICS クライアント (Windows 版) のワークロード・マネージャーがロード・バランシングに使用するプログラム名を、ユーザーが変更できるようにします。

この出口は、ワークロード・マネージャーが使用可能になっているときにのみ使用できます。

呼び出し地点: ワークロード・マネージャーが接続先の ECI プログラム用のサーバーを選択する直前。

パラメーター:

Anchor 入力パラメーターです。**CICS_EciInitializeExit** で設定されたポインターです。

ECIParms ECI パラメーター・ブロックです。

Program ワークロード・マネージャーがロード・バランシングのために使用する ECI プログラムの別名。

戻りコード:

CICS_EXIT_OK
ECI は、要求の処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 ECI は、クライアントのトレース・レコードを作成してから、要求の処理を継続します。

EPI 出口の解説

このセクションでは、以下の出口について説明します。

- **CICS_EpiInitializeExit**
- **CICS_EpiTerminateExit**
- **CICS_EpiAddTerminalExit**
- **CICS_EpiTermIdExit**
- **CICS_EpiTermIdInfoExit**
- **CICS_EpiStartTranExit**
- **CICS_EpiReplyExit**
- **CICS_EpiDelTerminalExit**
- **CICS_EpiGetEventExit**
- **CICS_EpiSystemIdExit**
- **CICS_EpiTranFailedExit**

表9 に、出口名、各出口に渡されるパラメーター、および戻りコードの要約を示します。

表 9. EPI 出口の要約

関数名	パラメーター	戻りコード
CICS_EpiInitializeExit	**Version** **Anchor**	CICS_EXIT_OK CICS_EXIT_NO_EXIT CICS_EXIT_CANT_INIT_EXITS ユーザー定義
CICS_EpiTerminateExit	**Anchor**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_STORAGE ユーザー定義
CICS_EpiAddTerminalExit	**Anchor** **NameSpace** **System** **NetName** **DevType**	CICS_EXIT_OK CICS_EXIT_DONT_ADD_TERMINAL CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiTermIdExit	**Anchor** **TermIndex** **System**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiTermIdInfoExit	**Anchor** **Version** **TermIndex** **EpiDetails**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義

表 9. EPI 出口の要約 (続き)

関数名	パラメーター	戻りコード
CICS_EpiStartTranExit	**Anchor** **TransId** **Data** **Size**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiReplyExit	**Anchor** **TermIndex** **Data** **Size**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiDelTerminalExit	**Anchor** **TermIndex**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiGetEventExit	**Anchor** **TermIndex** **Wait** **Event**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiSystemIdExit	**Anchor** **NameSpace** **System** **NetName** **DevType** **FailedSystem** **Reason** **SubReason** **UserId** **PassWord**	CICS_EXIT_OK CICS_EXIT_DONT_ADD_TERMINAL CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義
CICS_EpiTranFailedExit	**Anchor** **TermIndex** **Wait** **Event**	CICS_EXIT_OK CICS_EXIT_BAD_ANCHOR CICS_EXIT_BAD_PARM ユーザー定義

CICS_EpiInitializeExit

CICS_EpiInitializeExit	Version Anchor

目的
出口環境をセットアップします。

呼び出し地点
CICS_EpiInitialize の呼び出し時に、EPI がパラメーターの妥当性検査を行った後で呼び出されます。

パラメーター

Version 入力パラメーターです。出口の稼働する EPI のバージョンを表します。

Anchor 出力パラメーターです。EPI 出口に渡されるポインターを指すポインターです。2 番目のポインターは EPI には使用されません。そのまま出口に渡されます。この出口でのストレージを獲得して、そのアドレスを他の出口に渡すことができます。

戻りコード

CICS_EXIT_OK
EPI は、適切な出口を呼び出して、この要求の処理を継続します。

CICS_EXIT_NO_EXIT
EPI は、他の出口は呼び出さずに、この要求の処理を継続します。

CICS_EXIT_CANT_INIT_EXITS
EPI は、クライアントのトレース・レコードを作成してから、この要求の処理を続行しますが、これ以上は出口を呼び出しません。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、この要求の処理を続行しますが、これ以上は出口を呼び出しません。

CICS_EpiTerminateExit

CICS_EpiTerminateExit	Anchor

目的

出口環境の終結処理を行います。**CICS_EpiInitializeExit** で獲得したストレージは、この出口ですべて解放しなければなりません。

呼び出し地点

CICS_EpiTerminate の呼び出し時に、EPI がパラメーターの妥当性検査を行った後で呼び出されます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

戻りコード

CICS_EXIT_OK
終了処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、終了処理を続行します。

CICS_EXIT_BAD_STORAGE
CICS は、ストレージ・エラーを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、終了処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、終了処理を続行します。

CICS_EpiAddTerminalExit

CICS_EpiAddTerminalExit	Anchor
	NameSpace
	System
	NetName
	DevType

目的

ユーザーがサーバーを選択するか、または、 **System** パラメーターの **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** に渡されたユーザーを指定変更できるようにします。

呼び出し地点

EPI がパラメーターの妥当性検査を行った後に、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** を呼び出すつど。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターのストレージです。

NameSpace 入出力パラメーターです。入力の場合、その値は、この出口が関連する **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しの **NameSpace** パラメーターに渡された値によって異なります。

- ヌル・ポインターが渡されている場合には、この入力パラメーターは、ヌル文字を指すポインターになります。
- 非ヌルのポインターが渡されている場合は、 **Namespace** 入力パラメーターは、このデータのコピーを指します。

出力の場合、呼び出しで指定した値が使われたのと同じ方法で、EPI はこのパラメーターを使用します。

System 入出力パラメーターです。入力の場合、この出口が関連する **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しの **System** パラメーターに渡された値です。出力の場合、呼び出しで指定した値が使われたのと同じ方法で、EPI はこのパラメーターを使用します。

NetName 入出力パラメーターです。入力の場合、この出口が関連する **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しの **NetName** パラメーターに渡された値です。出力の場

合、呼び出しで指定した値が使われたのと同じ方法で、EPI はこのパラメーターを使用します。

DevType 入出力パラメーターです。入力の場合、この出口が関連する **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しの **DevType** パラメーターに渡された値です。出力の場合、呼び出しで指定した値が使われたのと同じ方法で、EPI はこのパラメーターを使用します。

戻りコード

CICS_EXIT_OK
NameSpace、**System**、**NetName**、および **DevType** の出力パラメーターの値を使用して、処理を継続します。

CICS_EXIT_DONT_ADD_TERMINAL
CICS_EpiAddTerminal または **CICS_EpiAddExTerminal** が終了し、CICS_EPI_ERR_FAILED の戻りコードが戻されます。アプリケーションが **CICS_EpiGetSysError** を使用している場合には、値 109 が **CICS_EpiSysError_t** 構造の **Cause** フィールドに戻されます。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

注

システムの選択に関する注意

呼び出し側のアプリケーションがパラメーター・リストにあるシステム名を指定しない場合には、システムが動的に選択され、出口が確実にシステムを選択できるようになります。

ただし、呼び出し側のアプリケーションがシステム名を指定すると、ターゲット・システムが変更されることを期待していないので、アプリケーション・エラーが生じることがあります。この場合、出口が初めから置換システムを指定していないと、指定の、またはデフォルトのシステム名、装置タイプなどが使用されます。このとき、選択したシステムを変更しようとすると、変更することはできますが、以下のことに注意してください。

- 出口ルーチンは、ターゲット・システムの修正によって、クライアントで稼働している EPI アプリケーションにエラーが生じるか否かに注意が必要です。
- 出口ルーチンは、この修正がクライアント・アプリケーションに受け入れられるかどうか判断するために、適切なデータが位置づけできる知識ベースを維持しなければなりません。

CICS_EpiAddTerminalExit および CICS_EpiSystemIdExit

これらの出口間の関係は、次のとおりです。出口では、システムの選択を複数回行うことができます。最初の選択は必ず **CICS_EpiAddTerminalExit** です。この出口は、アプリケーションが **CICS_EpiAddTerminal** or **CICS_EpiAddExTerminal** に渡したパラメーターのみを受け取ります。CICS が端末を追加しようとしたときにエラーが起こると (出口が選択を行ったかどうかに関係なく)、 **CICS_EpiSystemIdExit** が呼び出されます。端末の追加を試みているときに起こったエラーが **CICS_EpiSystemIdExit** に渡され、エラーを訂正することができます。この状態は、端末が正しく追加されるか、または **CICS_EpiSystemIdExit** が放棄の信号を出されない限り、継続します。

端末の追加を試みているときにエラーが起きなければ、**CICS_EpiSystemIdExit** は呼び出されません。

CICS_EpiTermIdExit

CICS_EpiTermIdExit	Anchor TermIndex System

目的

CICS_EpiAddTerminal が正しく呼びされた後で割り振られた端末索引を通知します。

以前のバージョンのアプリケーションとの互換性を保つためのみに、**CICS_EpiTermIdExit** が提供されています。 EPI 出口を使用するすべての新規アプリケーションは、代わりに **CICS_EpiTermIdInfoExit** を使用する必要があります。

呼び出し地点

CICS_EpiAddTerminal の呼び出し時で、サーバーが端末を割り振った後で呼び出されます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TermIndex 入力パラメーターです。これは、予約時またはインストール時の端末リソースの端末索引です。

System 入力パラメーターです。端末リソースをインストールまたは予約したサーバーの名前を指定する、ヌルで終了するストリングを指すポインターです。

戻りコード

CICS_EXIT_OK
処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

CICS_EXIT_BAD_PARM

CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

CICS_EpiTermIdInfoExit

CICS_EpiTermIdInfoExit	Anchor Version TermIndex EpiDetails

目的
ユーザーが現行端末に関する情報を検索できるようにします。

呼び出し地点
CICS 端末をインストールした直後。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

Version 入力パラメーターです。EPI バージョン。

TermIndex 入力パラメーターです。インストールする端末の索引です。

EpiDetails 入力パラメーターです。**CICS_EpiDetails_t** 構造へのポインターで、インストールされた端末の詳細情報を含んでいます。

戻りコード

CICS_EXIT_OK
処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、CICS_EXIT_OK の処理を続行します。

CICS_EpiStartTranExit

CICS_EpiStartTranExit	Anchor
	TransId
	Data
	Size

目的

トランザクションの開始時刻を調べます。これは、情報収集の目的のために行います。この出口はシステムを選択しません。また、戻りデータもありません。

呼び出し地点

CICS_EpiStartTran の呼び出し時に、EPI がパラメーターの妥当性検査を行った後で呼び出されます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TransId 入力パラメーターです。この出口に関連した **CICS_EpiStartTran** 呼び出しの **TransId** パラメーターに渡された値です。

Data 入力パラメーターです。この出口に関連した **CICS_EpiStartTran** 呼び出しの **Data** パラメーターに渡された値です。

Size 入力パラメーターです。この出口に関連した **CICS_EpiStartTran** 呼び出しの **Size** パラメーターに渡された値です。

戻りコード

CICS_EXIT_OK
CICS_EpiStartTran 呼び出しの処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiStartTran** 呼び出しの処理を続行します。

CICS_EXIT_BAD_PARM

CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiStartTran** 呼び出しの処理を続行します。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、 **CICS_EpiStartTran** 呼び出しの処理を続行します。

CICS_EpiReplyExit

CICS_EpiReplyExit	**Anchor** **TermIndex** **Data** **Size**

目的

トランザクションが応答を受けた時刻を調べます。これは、情報収集の目的のために行います。

呼び出し地点

CICS_EpiReply の呼び出し時に、EPI がパラメーターの妥当性検査を行った後に呼び出されます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TermIndex 入力パラメーターです。この出口に関連した **CICS_EpiReply** 呼び出しの **TermIndex** パラメーターに渡された値です。

Data 入力パラメーターです。この出口に関連した **CICS_EpiReply** 呼び出しの **Data** パラメーターに渡された値です。

Size 入力パラメーターです。この出口に関連した **CICS_EpiReply** 呼び出しの **Size** パラメーターに渡された値です。

戻りコード

CICS_EXIT_OK
CICS_EpiReply 呼び出しの処理が継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiReply** 呼び出しの処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiReply** 呼び出しの処理を続行します。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、 **CICS_EpiReply** 呼び出しの処理を続行します。

CICS_EpiDelTerminalExit

CICS_EpiDelTerminalExit	Anchor TermIndex

目的

端末関連のデータ構造の終結処理を行います。

呼び出し地点

CICS_EpiDelTerminal または **CICS_EpiPurgeTerminal** の呼び出し時で、EPI がパラメーターの妥当性検査を行った後に呼び出されます。端末関連のデータ構造の終結処理を行います。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TermIndex 入力パラメーターです。この出口に関連した **CICS_EpiDelTerminal** または **CICS_EpiPurgeTerminal** 呼び出しの **TermIndex** パラメーターに渡された値です。

戻りコード

CICS_EXIT_OK
CICS_EpiDelTerminal または **CICS_EpiPurgeTerminal** 呼び出しが続行されます。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiDelTerminal** または **CICS_EpiPurgeTerminal** 呼び出しの処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiDelTerminal** または **CICS_EpiPurgeTerminal** 呼び出しの処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiDelTerminal** または **CICS_EpiPurgeTerminal** 呼び出しの処理を続行します。

CICS_EpiGetEventExit

CICS_EpiGetEventExit	Anchor TermIndex Wait Event

目的

受信したイベントに関するデータを収集します。

呼び出し地点

CICS_EpiGetEvent が呼び出し側に制御を戻す直前に呼び出されます。この出口は、戻されたデータ、システムからの応答時間などを調べることができます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TermIndex 入力パラメーターです。この出口が関連した **CICS_EpiGetEvent** 呼び出しの **TermIndex** パラメーターのアプリケーションに戻される値です。

Wait 入力パラメーターです。この出口に関連した **CICS_EpiGetEvent** 呼び出しの **Wait** パラメーターに渡された値です。

Event 入力パラメーターです。この出口が関連した **CICS_EpiGetEvent** 呼び出しの **Event** パラメーターのアプリケーションに戻される値です。

戻りコード

CICS_EXIT_OK
CICS_EpiGetEvent 呼び出しの処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

CICS_EXIT_BAD_PARM

CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

CICS_EpiSystemIdExit

CICS_EpiSystemIdExit	Anchor
	NameSpace
	System
	NetName
	DevType
	FailedSystem
	Reason
	SubReason
	UserId
	PassWord

目的

CICS_Epi_AddTerminal または **CICS_EpiAddExTerminal** の値が無効な場合に、ユーザーが新規の名前を提供できるようにします。

呼び出し地点

端末の追加中にエラーが発生して、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** がアプリケーションに戻る直前。考えられるエラーは、CICS_EPI_ERR_SYSTEM、 CICS_EPI_ERR_FAILED、または CICS_EPI_ERR_SERVER_DOWN です。このエラーは、**CICS_EpiAddTerminalExit** または **CICS_EpiAddExTerminal** が前に呼び出されたかどうかに関係なく発生します。

注: システムによっては、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** の完了と同期してアプリケーションに制御が戻らない場合があります。その場合、この出口は非同期で呼び出されます。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

NameSpace 入出力パラメーターです。失敗した **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** に使用した **NameSpace** パラメーターです。

System 入出力パラメーターです。失敗した **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** に使用した **System** パラメーターです。

NetName 入出力パラメーターです。失敗した **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** に使用した **NetName** パラメーターです。

DevType 入出力パラメーターです。失敗した **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** に使用した **DevType** パラメーターです。

FailedSystem 入力パラメーターです。障害の起きたシステムの ID です。

Reason 入力パラメーターです。障害の理由は、CICS_EPI_ERR_SYSTEM または CICS_EPI_ERR_FAILED です。

SubReason 入力パラメーターです。障害の詳細です。理由が CICS_EPI_ERR_FAILED の場合、**CICS_EpiSysError_t** 構造の **Cause** フィールドの値が入ります。

UserId 出力パラメーターです。使用されません。

PassWord 出力パラメーターです。使用されません。

戻りコード

CICS_EXIT_OK
EPI は、この出口の出力に指定された値を使用して、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しを再試行します。この場合、189ページの『CICS_EpiAddTerminalExit』で説明した考慮事項が適用されます。

CICS_EXIT_DONT_ADD_TERMINAL
CICS_EpiAddTerminal または **CICS_EpiAddExTerminal** が終了し、CICS_EPI_ERR_FAILED の戻りコードが戻されます。アプリケーションが **CICS_EpiGetSysError** を使用している場合には、値 109 が **CICS_EpiSysError_t** 構造の **Cause** フィールドに戻されます。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成し、出口を呼び出す原因となったエラーがアプリケーションに戻されます。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成し、出口を呼び出す原因となったエラーがアプリケーションに戻されます。

ユーザー定義

ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成し、出口を呼び出す原因となったエラーがアプリケーションに戻されます。

CICS_EpiTranFailedExit

CICS_EpiTranFailedExit	Anchor TermIndex Wait Event

目的

トランザクションが異常終了したとき、または端末に障害が起きたときのデータを収集します。

呼び出し地点

イベントが CICS_EPI_EVENT_END_TRAN で、 **AbendCode** フィールドがブランクではない場合に、 **CICS_EpiGetEvent** が呼び出し側に制御を戻す直前に呼び出されます (**GetEventExit** がある場合や、ない場合もあります)。

リモート・システムで障害があり、3270 データ・ストリームにエラー・メッセージはあるが、 CICS_EPI_EVENT_END_TRAN に異常終了コードがないことがあります。このエラー・メッセージは、 CICS_EPI_EVENT_END_TRAN と同じイベントで出されないこともあります。この状態を出口で処理する必要がある場合には、これを **CICS_EpiGetEventExit** でモニターして、該当する 3270 データ・ストリームを走査します。

パラメーター

Anchor 入力パラメーターです。**CICS_EpiInitializeExit** で設定されたポインターです。

TermIndex 入力パラメーターです。この出口が関連した **CICS_EpiGetEvent** 呼び出しの **TermIndex** パラメーターのアプリケーションに戻される値です。

Wait 入力パラメーターです。この出口に関連した **CICS_EpiGetEvent** 呼び出しの **Wait** パラメーターに渡された値です。

Event 入力パラメーターです。この出口が関連した **CICS_EpiGetEvent** 呼び出しの **Event** パラメーターのアプリケーションに戻される値です。

戻りコード

CICS_EXIT_OK
CICS_EpiGetEvent 呼び出しの処理を継続します。

CICS_EXIT_BAD_ANCHOR
CICS は、無効なアンカー・フィールドを検出しました。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

CICS_EXIT_BAD_PARM
CICS は、無効なパラメーターを検出しました。EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

ユーザー定義
ユーザー定義の戻りコードには、CICS_EXIT_USER_BASE 以上の値が入っていなければなりません。 EPI は、クライアントのトレース・レコードを作成してから、**CICS_EpiGetEvent** 呼び出しの処理を続行します。

診断情報

CICS クライアントは、入力パラメーターが呼び出される前に、即時に出口までこのパラメーターをトレースし、出口が戻されるとこの出口からの出力をトレースします。CICS トレースは、出口の中で使用することはできません。

CICSTERM、CICSPRNT、および EPI 出口

CICS ユニバーサル・クライアントのバージョン 3 では、 CICSTERM および CICSPRNT コマンドを使用して EPI ユーザー出口を起動することができます。これにより、CICS ユニバーサル・クライアント (Windows 版) の Load Manager に対して類似したロード・バランシング機能を提供することができます (「*CICS ユニバーサル・クライアント (Windows 版) 管理の手引き*」を参照してください)。

EPI 出口を使用するには、cicsepix.dll と呼ばれる DLL (UNIX プラットフォームでは cicsepix.a) に CICS_EPIEXITINIT 関数を提供します。CICSTERM または CICSPRNT セッションを開始すると、 CICS ユニバーサル・クライアントは、その bin ディレクトリーで cicsepix.dll (cicsepix.a) を検索します。 DLL が検出されない場合は、出口処理は行われません。

CICS_EPIEXITINIT 関数は、ExitList 構造を設定して、全出口関数 (cicsepix.dll にも含まれます) のアドレスを指します。サンプルの CICS_EPIEXITINIT を以下に示します。

```
void CICSEXIT CICS_EPIEXITINIT(CICS_EpiExitList_t *ExitList)

{
    ExitList->InitializeExit    = &CICS_EpiInitializeExit;
    ExitList->TerminateExit     = &CICS_EpiTerminateExit;
    ExitList->AddTerminalExit   = &CICS_EpiAddTerminalExit;
    ExitList->StartTranExit     = &CICS_EpiStartTranExit;
    ExitList->ReplyExit         = &CICS_EpiReplyExit;
    ExitList->DelTerminalExit   = &CICS_EpiDelTerminalExit;
    ExitList->GetEventExit      = &CICS_EpiGetEventExit;
    ExitList->TranFailedExit    = &CICS_EpiTranFailedExit;
    ExitList->SystemIdExit      = &CICS_EpiSystemIdExit;
    ExitList->TermIdExit        = &CICS_EpiTermIdExit;
    ExitList->TermIdInfoExit    = &CICS_EpiTermIdInfoExit;
}
```

提供されたアドレスを使用して出口に入るので、関数のシグニチャーが CICS_Epi* 関数と正確に同じである限り、任意の名前を指定することができます。したがって、ExitList-> 出口は、汎用端末制御出口と見なすことができます。

Initializeexit には、CICSTERM または CICSPRNT によって起動されるときにバージョン番号 X'FF000000' が渡されます。これによって、ユーザー・プログラムは、必要に応じて CICSTERM と CICSPRNT ユーザー出口、および EPI ユーザー出口を区別することができます。

CICSTERM および CICSPRNT は、EPI トレース点も起動します。

以下では、CICSTERM および CICSPRNT に使用することができる EPI 出口のサブセット、およびこれらの実現方法について説明します。これらが CICS クライアントの端末エミュレーターの動作にどのように影響を与えるかについても説明します。

EPI: CICS_EpiInitializeExit

この EPI 出口は、EPI プログラムの呼び出しの実行には影響を及ぼしませんが、ユーザーは、EPI プログラムを呼び出すプロセスに対してユーザー出口をオンまたはオフに切り替えることができます。これは、EPI を使用するプロセスごとに 1 回ずつ呼び出されます。これは、他の EPI が呼び出される前、かつ **CICS_EpiInitialize** が正常に終了するときに呼び出されます。

CICSTERM: InitializeExit

この出口は、各 CICSTERM が別個のプロセスで実行されるので、作成される CICSTERM セッションごとに 1 回しか呼び出されません。渡されるバージョン番号は X'FF000000' です。

EPI: CICS_EpiTerminateExit

CICS_EpiTerminate によって呼び出される場合は、これは、常に特定のプロセスの最後の EPI 呼び出しです。これは、EPI プログラムの呼び出しの実行には影響を及ぼしません。これは、EPI が初期化されたこととアクティブな通知スレッドがないことを検査した後、および EPI が実際に終了する前に呼び出されます。 EPI 出口 DLL は、ユーザー出口が呼び出されるとすぐにアンロードされます。

CICSTERM: TerminateExit

CICSTERM 終了処理中に一度しか呼び出されません。

EPI: CICS_EpiAddTerminalExit

サーバーを選択して、EPI 呼び出しに渡されるサーバー・パラメーターを変更して、サーバーへの端末の追加を拒否することができます。これは、すべて EPI 呼び出しから起こります。その後、EPI プログラムは索引番号によってサーバーを参照するため、プログラムが実際の接続先サーバーを認識している必要はありません。ユーザー出口がサーバーの接続を拒否する場合は、**CICS_EpiSystemIDExit** は呼び出されません (詳細については、以下を参照してください)。

CICS_EpiAddTerminalExit は、EPI が正常に初期化されたことと空きセッションがあることを **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** が検査した後に呼び出されます。これは、**CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しが実際に端末定義をサーバーに送信する前に呼び出されます。

CICSTERM: AddTerminalExit

CICSTERM の /s または /r パラメーターによって、ユーザーは、CICS ユニバーサル・クライアントが以下と接続できることを指定することができます。

- クライアント構成ファイルで定義された最初のサーバー。
- 使用可能なサーバーのリストからユーザーが選択したサーバー。
- /s または /r パラメーターによって指定されたサーバー。

AddTerminalExit は、パラメーターとしてシステム名を受信して、必要な場合に異なるサーバーを指定したり、サーバーを拒否して端末エミュレーターを終了させたりすることができます。 **AddTerminalExit** がインストールを拒否する場合は、 CICSTERM は、サーバーが使用不能であることを示すエラーを表示します。

EPI: CICS_EpiSystemIdExit

CICS_EpiAddTerminal または **CICS_EpiAddExTerminal** が失敗した場合に、ユーザーがサーバーを再選択することができます。このユーザー出口は、**CICS_EpiAddTerminal** が失敗した場合に呼び出されます

(**CICS_EpiAddTerminalExit** が障害を起こした場合は呼び出されません)。これが CICS_EXIT_OK を戻すと、 **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** は端末を再度追加しようとします。再試行の間に呼び出されるこの出口によって、このサーバー・パラメーターを変更することができます。

CICS_EpiSystemIdExit を EPI プログラムによって非同期的または同期的に呼び出すことができます。**CICS_EpiSystemIdExit** は、以下によって表すことができます。

- CICS_EPI_ERR_SYSTEM エラー。サーバーが認識されていないことを意味します。
- CICS_EPI_ERR_SERVER_DOWN エラー。サーバーに障害が起こったことを意味します。
- CICS_EPI_ERR_SECURITY エラー。セキュリティーの障害です。
- CICS_EPI_ERR_FAILED エラー。その他のタイプの障害です。

これには、 **cics_syserr_t** データ構造の **cause** フィールドと同じパラメーターが渡されます。この値は、エラーも指定して、操作環境に特有の値です。

CICSTERM: SystemIdExit

クライアント・リクエスターに (クライアント構成ファイルの **Maximum Requests** パラメーターによって制御される) 十分なセッションが空いていないために、CICSTERM **CICS_EpiAddTerminal** または **CICS_EpiAddExTerminal** 呼び出しが失敗する場合は、**SystemIdExit** が 1 次理由コード CICS_EPI_ERR_FAILED および 2 次理由コード 7046 とともに呼び出されます。2 次理由コードの数値は、リソース不足に対するクライアントのトレースの番号 (CCL7046E) と同じです。CICS_EPI_ERR_FAILED のその他の場合では、いずれも CICSTERM は 2 次理由コード 0 を渡します。

ユーザー出口がアクティブでないときに、使用可能なセッションが不足するために失敗する場合は、 CICSTERM は端末のインストールを再試行します。(これによって、インストール前に端末が空きセッションを待機することができます。)使用可能なユーザー出口がある場合は、再試行の動作は、出口によって完全に制御されます。

CICS_EpiSystemId は、常に端末スレッドから同期的に呼び出され (つまり、端末インストール応答が送信され)、応答を待機して、応答が正常でない場合は **SystemIdExit** を起動します。

EPI: CICS_EpiTermIdExit

追加された端末に指定された EPI Termid を確認することができます。

これは、端末がサーバーに正常にインストールされた後でしか呼び出されません。これは、EPI プログラムの実行に影響を及ぼしません。EPI Termid の番号は、EPI プログラムが稼働する各プロセスにローカルです。

CICSTERM: TermIdExit

CICSTERM は各プロセスで 1 つしか稼働しないので、 Termid の番号は 1 にハードコーディングされます。

EPI: CICS_EpiTermIdInfoExit

端末に関する詳細を知ることができます。これは、端末がサーバーに正常にインストールされた後で呼び出されます。

CICSTERM: TermIdInfoExit

CICSTERM は各プロセスで 1 つしか稼働しないので、 Termid の番号は 1 にハードコーディングされます。

EPI: CICS_EpiDelTerminalExit

端末コードを終結処理することができます。これは、**CICS_EpiDelTerminal** を出すと呼び出されます。これは、EPI プログラムの実行に影響を及ぼしません。

CICSTERM: DelTerminalExit

CICSTERM は各プロセスで 1 つしか稼働しないので、 Termid の番号は 1 にハードコーディングされます。これは、端末を終了するときに **TerminateExit** の直前に呼び出されます。サーバーに障害が起こると、サーバーの再起動時に **AddTerminalExit** が再度呼び出されます。ただし、**DelTerminalExit** は、サーバーに障害が起こっても呼び出されません。

EPI: CICS_EpiStartTranExit

トランザクションが開始されたこと、および Transid および 3270 データがそのトランザクションに送信されたことを確認することができます。これは、EPI プログラムの実行に影響を及ぼしません。**CICS_EpiStartTranExit** は、 EPI 状態が検査された後、および **CICS_EpiStartTran** が呼び出される直前に呼び出されます。

この場合は EPI プログラムがトランザクションを再度開始しなければならなくなるので、疑似会話型トランザクションによって **CICS_EpiStartTranExit** が呼び出されることに注意してください。

CICSTERM: StartTranExit:

非 ATI トランザクションが開始されている場合は、 **StartTranExit** が呼び出され、 Transid フィールドおよび Data フィールドの TIOA (端末入出力域) にブランクが送信されます。 Transid は、TIOA データの

最初の 4 文字であるか、あるいは (X'11' を開始する) 3270 バッファー・アドレス設定 (SBA) コマンドの後に置かれます。後者の場合は、(SBA コマンドは合計 3 バイトを取るので) これは TIOA の第 4 バイトで開始します。

StartTranExit は、ATI トランザクションに対しては起動されません。これは、出口は通常 **CICS_EpiStartTran** API 呼び出しによって起動され、 ATI トランザクションを開始するために呼び出されることはないためです。ただし、疑似会話型トランザクションは **StartTranExit** を起動します。これは、EPI プログラムでは、ユーザーが **CICS_EpiStartTran** を呼び出して疑似会話型トランザクションを開始しなければならないためです。疑似会話型トランザクションの場合は、トランザクション ID は transid パラメーター・ブロックに置かれ、データ・ブロックで渡される TIOA にはトランザクション ID は含まれません。

StartTranExit は、アプリケーションによって CICSTERM セッションから発行された EXEC CICS RETURN TRANSID*name* IMMEDIATE コマンドによって呼び出されることはありません。

EPI: CICS_EpiReplyExit

アプリケーションが応答を受けた時点を確認することができます。これは、EPI プログラムの実行に影響を及ぼしません。

CICSTERM ReplyExit

クライアントがデータを送信しており、トランザクションが現在アクティブである場合に活動化されます。 Termid の番号は 1 にハードコーディングされます。端末 TIOA は、**ReplyExit** に渡されます。

ReplyExit は、アプリケーションによって CICSTERM セッションから発行された EXEC CICS RETURN TRANSID*name* IMMEDIATE コマンドによって呼び出されることはありません。

GetEventExit および **TranFailedExit** 出口は、 CICSTERM および CICSPRNT に対しては実現されません。

付録D. 特記事項

本書はアメリカ合衆国で提供されている製品およびサービス用に作成されたものであり、本書に記載の製品、サービス、またはフィーチャー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、およびフィーチャーについては、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。これらに代えて、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す。ただし、IBM 以外の製品、プログラムまたはサービスの操作性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。実施権の許諾については、下記の宛先に書面にてご照会ください。

〒106-0032 東京都港区六本木 3 丁目 2-31
AP事業所
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

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で提供し、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。

本書は定期的に見直され、必要な変更 (たとえば、技術的に不適確な表現や誤植など) は、本書の次版に組み込まれます。IBM は、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、お客様の責任でご使用ください。

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 United Kingdom Laboratories, MP151,
Hursley Park, Winchester, Hampshire,
England, SO21 2JN

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。また、IBM 以外の製品に関するパフォーマンスの正確性、互換性、またはその他の要求は確証できません。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にお願いします。

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、IBM 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のめかしたり、保証することはできません。

プログラミング・インターフェース情報

本書は、CICS ファミリーの各種プロダクトの機能を使用するアプリケーションの作成を支援するためのものです。CICS ファミリーの各種プロダクトで使用する汎用プログラミング・インターフェースおよび関連ガイダンス情報が説明されています。

汎用プログラミング・インターフェースを使用すれば、さまざまな CICS ファミリーの各種プロダクトのサービスを得るためのプログラムの作成が利用可能になります。

本書では、CICS ファミリーの各種プロダクトで使用するプロダクト・センシティブ・プログラミング・インターフェースおよび関連ガイダンス情報についても説明しています。

プロダクト・センシティブ・プログラミング・インターフェースを使用すれば、システム導入部署は CICS の診断、修正、監視、修理、調整、またはチューニングなどの作業を実行することができます。そのようなインターフェースを使用すると、IBM ソフトウェア・プロダクトの詳細設計または実施への依存が生じます。プロダクト・センシティブ・プログラミング・インターフェースは、これらの特定な目的に限って使用されます。詳細設計または実施に依存するために、そのようなインターフェースへ書き込まれたプログラムは、新規プロダクト・リリースやバージョンを実行するために、または修正サービスの結果として変更しなければならない場合があります。

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関連ガイダンス情報が現れる場所は、章またはセクションの序文で識別されます。

商標

以下は、IBM Corporation の商標です。

AIX	CICS	CICS/ESA
CICS/MVS	CICS OS/2	CICS/VSE
CICS/400	IBM	OS/2
OS/390	OS/400	Presentation Manager
VisualAge	System/390	

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 がライセンスしている米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標または登録商標です。

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。

[カ行]

[サ行]

[タ行]

[ハ行]

[ヤ行]

[ラ行]

[ワ行]

A

B

C

D

E

L

N

O

P

R

S

T

U

V

W

Printed in Japan

SC88-8998-00

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12