BEA Tuxedo

BEA Tuxedo
COBOL リファレンス

BEA Tuxedo リリース 8.0J
8.0 版
2001 年 10 月

Copyright

Copyright © 2001 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, WebLogic, Tuxedo, and Jolt are registered trademarks of BEA Systems, Inc. How Business Becomes E-Business, BEA WebLogic E-Business Platform, BEA Builder, BEA Manager, BEA eLink, BEA WebLogic Commerce Server, BEA WebLogic Personalization Server, BEA WebLogic Process Integrator, BEA WebLogic Collaborate, BEA WebLogic Enterprise, and BEA WebLogic Server are trademarks of BEA Systems, Inc.

All other product names may be trademarks of the respective companies with which they are associated.

BEA Tuxedo COBOL リファレンス

Document Edition	Date	Software Version
8.0J	2001 年 10 月	BEA Tuxedo リリース 8.0J

目次

このマニュアルについて

このマニュアルは、BEA Tuxedo ATMI 環境で使用される COBOL バインディングについて説明します。リファレンス・ページは関数名のアルファベット順になっています。

対象読者

このマニュアルは、次のような読者を対象としています。

- BEA Tuxedo 環境でアプリケーションの作成と管理を担当する管理者
- BEA Tuxedo 環境でアプリケーションのプログラミングを行うアプリケーション開発者

このマニュアルは、BEA Tuxedo プラットフォームおよび COBOL プログラミングの知識があることを前提としています。

e-docs Web サイト

BEA 製品のマニュアルは BEA 社の Web サイト上で参照することができます。BEA ホーム・ページの [製品のドキュメント] をクリックするか、または http://edocs.beasys.co.jp/e-docs/index.html に直接アクセスしてください。

マニュアルの印刷方法

このマニュアルは、ご使用の Web ブラウザで一度に 1 ファイルずつ印刷できます。Web ブラウザの [ファイル] メニューにある [印刷] オプションを使用してください。

このマニュアルの PDF 版は、Web サイト上にあります。また、マニュアルの CD-ROM にも収められています。BEA Tuxedo この PDF を Adobe Acrobat Reader で開くと、マニュアル全体または一部をブック形式で印刷できます。PDF 形式を利用するには、BEA Tuxedo Documents ページの [PDF 版] ボタンをクリックして、印刷するマニュアルを選択します。

Adobe Acrobat Reader をお持ちではない場合は、Adobe Web サイト (http://www.adobe.co.jp/) から無償で入手できます。

関連情報

関連情報は各リファレンス・ページの「関連項目」に一覧表示されています。

サポート情報

皆様の BEA Tuxedo マニュアルに対するフィードバックをお待ちしています。ご意見やご質問がありましたら、電子メールで **docsupport-jp@bea.com** までお送りください。お寄せいただきましたご意見は、BEA Tuxedo マニュアルの作成および改訂を担当する BEA 社のスタッフが直接検討いたします。

電子メール メッセージには、BEA Tuxedo 8.0 リリースのマニュアルを使用していることを明記してください。

BEA Tuxedo に関するご質問、または BEA Tuxedo のインストールや使用に際して問題が発生した場合は、**www.bea.com** の BEA WebSUPPORT を通して BEA カスタマ・サポートにお問い合わせください。カスタマ・サポートへの問い合わせ方法は、製品パッケージに同梱されているカスタマ・サポート・カードにも記載されています。

カスタマ・サポートへお問い合わせの際には、以下の情報をご用意ください。

- お客様のお名前、電子メール・アドレス、電話番号、Fax 番号
- お客様の会社名と会社の住所

- ご使用のマシンの機種と認証コード
- ご使用の製品名とバージョン
- 問題の説明と関連するエラー・メッセージの内容

表記上の規則

このマニュアルでは、以下の表記規則が使用されています。

規則	項目
太字	用語集に定義されている用語を示します。
Ctrl + Tab	2 つ以上のキーを同時に押す操作を示します。
イタリック体	強調またはマニュアルのタイトルを示します。

等幅テキスト	コード・サンプル、コマンドとオプション、データ構造とメンバ、データ型、ディレクトリ、およびファイル名と拡張子を示します。また、キーボードから入力する文字も示します。
	例 :
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

等幅太字	コード内の重要な単語を示します。 例 : `void` **`commit ( )`**
等幅イタリック体	コード内の変数を示します。 例 : `String` *`expr`*

規則	項目
大文字	デバイス名、環境変数、および論理演算子を示します。 例 : LPT1 SIGNON OR
{ }	構文の行で選択肢を示します。かっこは入力しません。

[ ]	構文の行で省略可能な項目を示します。かっこは入力しません。
	例 :
	`buildobjclient [-v] [-o name ] [-f file-list]...`
	`[-l file-list]...`

|	構文の行で、相互に排他的な選択肢を分離します。記号は入力しません。

...	コマンド行で次のいずれかを意味します。
	■ コマンド行で同じ引数を繰り返し指定できること
	■ 省略可能な引数が文で省略されていること
	■ 追加のパラメータ、値、その他の情報を入力できること
	省略符号は入力しません。
	例 :
	`buildobjclient [-v] [-o name ] [-f file-list]...`
	`[-l file-list]...`

. . .	コード例または構文の行で、項目が省略されていることを示します。省略符号は入力しません。

セクション 3(cbl) - COBOL 関数

表 1 BEA Tuxedo ATMI COBOL **関数**

名前	機能説明
COBOL アプリケーション・トランザクション・モニタ・インターフェイスの紹介	COBOL ATMI について紹介する
FINIT、FINIT32(3cbl)	フィールド化バッファの初期化
FVFTOS、FVFTOS32(3cbl)	フィールド化バッファから COBOL 構造体にコピーする
FVSTOF(3cbl)	C 構造体からフィールド化バッファにコピーする
TPABORT(3cbl)	現在の BEA Tuxedo ATMI のトランザクションのアボート
TPACALL(3cbl)	サービスへのメッセージの非同期送信を行うルーチン
TPADVERTISE(3cbl)	サービス名の宣言を行うルーチン
TPBEGIN(3cbl)	BEA Tuxedo ATMI のトランザクションを開始するルーチン
TPBROADCAST(3cbl)	名前によって通知をブロードキャストする
TPCALL(3cbl)	サービスへのメッセージの同期送信を行うルーチン
TPCANCEL(3cbl)	未処理の応答の通信ハンドルを取り消す
TPCHKAUTH(3cbl)	BEA Tuxedo ATMI アプリケーションへの結合に認証が必要かどうかをチェックする
TPCHKUNSOL(3cbl)	任意通知型メッセージをチェックする
TPCLOSE(3cbl)	BEA Tuxedo ATMI のリソース・マネージャのクローズ
TPCOMMIT(3cbl)	現在の BEA Tuxedo ATMI のトランザクションのコミット
TPCONNECT(3cbl)	会話接続の確立

表 1 BEA Tuxedo ATMI COBOL 関数 (続き)

名前	機能説明
TPDEQUEUE(3cbl)	キューからメッセージを取り出すルーチン
TPDISCON(3cbl)	会話接続の切断
TPENQUEUE(3cbl)	メッセージをキューに登録するルーチン
TPFORWAR(3cbl)	BEA Tuxedo ATMI のサービス要求を別のルーチンに転送
TPGETCTXT(3cbl)	現在のアプリケーション関連のコンテキスト識別子を取り出す
TPGETLEV(3cbl)	BEA Tuxedo ATMI のトランザクションの進行状況のチェック
TPGETRPLY(3cbl)	非同期メッセージからの応答の獲得
TPGETUNSOL(3cbl)	任意通知型メッセージの獲得
TPGPRIO(3cbl)	サービス要求の優先順位を獲得
TPINITIALIZE(3cbl)	BEA Tuxedo ATMI アプリケーションに結合する
TPKEYCLOSE(3cbl)	以前にオープンされたキー・ハンドルのクローズ
TPKEYGETINFO(3cbl)	キー・ハンドルに関連付けられている情報の獲得
TPKEYOPEN(3cbl)	デジタル署名の生成、メッセージの暗号化または暗号解読のためのキー・ハンドルのオープン
TPKEYSETINFO(3cbl)	キー・ハンドルに関連付けるオプション属性パラメータを設定する
TPNOTIFY(3cbl)	クライアント識別子により通知を送信する
TPOPEN(3cbl)	BEA Tuxedo ATMI のリソース・マネージャのオープン
TPPOST(3cbl)	イベントを通知する
TPRECV(3cbl)	会話接続でメッセージを受信する
TPRESUME(3cbl)	グローバル・トランザクションを再開する
TPRETURN(3cbl)	BEA Tuxedo ATMI のサービス・ルーチンからのリターン
TPSCMT(3cbl)	TPCOMMIT の終了時期の設定
TPSEND(3cbl)	会話接続でメッセージを送信するルーチン

表 1 BEA Tuxedo ATMI COBOL **関数 (続き)**

名前	機能説明
TPSETCTXT(3cbl)	現在のアプリケーション関連のコンテキスト識別子を設定する
TPSETUNSOL(3cbl)	任意通知型メッセージの処理方法の設定
TPSPRIO(3cbl)	サービス要求の優先順位の設定
TPSUBSCRIBE(3cbl)	イベントのサブスクライブ
TPSUSPEND(3cbl)	グローバル・トランザクションの一時停止
TPSVCSTART(3cbl)	BEA Tuxedo ATMI のサービスの開始
TPSVRDONE(3cbl)	BEA Tuxedo ATMI のサーバを終了するルーチン
TPSVRINIT(3cbl)	BEA Tuxedo ATMI のサーバを初期化するルーチン
TPTERM(3cbl)	アプリケーションを抜け出る
TPUNADVERTISE(3cbl)	サービス名の宣言を取り消すルーチン
TPUNSUBSCRIBE(3cbl)	イベントのサブスクリプションの削除
TXBEGIN(3cbl)	グローバル・トランザクションの開始
TXCLOSE(3cbl)	リソース・マネージャ・セットのクローズ
TXCOMMIT(3cbl)	トランザクションのコミット
TXINFORM(3cbl)	グローバル・トランザクション情報を返す
TXOPEN(3cbl)	リソース・マネージャ・セットのオープン
TXROLLBACK(3cbl)	トランザクションのロールバック
TXSETCOMMITRET(3cbl)	*commit_return* 特性の設定
TXSETTRANCTL(3cbl)	*transaction_control* 特性の設定
TXSETTIMEOUT(3cbl)	*transaction_timeout* 特性の設定
USERLOG(3cbl)	BEA Tuxedo ATMI の中央イベント・ログへのメッセージの書き込み

COBOL アプリケーション・トランザクション・モニタ・インターフェイスの紹介

機能説明

アプリケーション・トランザクション・モニタ・インターフェイス (ATMI) は、COBOL アプリケーションとトランザクション処理システムとの間に介在し、ATMI インターフェイスと呼ばれています。このマニュアル・ページでは、ATMI インターフェイスの COBOL 言語バインディングについて説明します。このインターフェイスは、リソースのオープンとクローズ、トランザクションの管理、レコード・タイプの管理、要求 / 応答や会話サービス・コールの呼び出しなどを行う各種のルーチンで構成されています。

コミュニケーション・パラダイム

ATMI マニュアル・ページに記述されているルーチン群を読めば、これらがコミュニケーションのためのモデルであることが分かります。このモデルは、クライアント・プログラムとサーバ・プログラムが要求および応答の各メッセージを使用して如何にコミュニケートできるかという観点から表現されています。

コミュニケーションの基本的パラダイムとして、要求 / 応答と会話の 2 つがあります。要求 / 応答型サービスは、サービス要求とそれに関わるデータによって呼び出されます。要求 / 応答型サービスは、要求を 1 つだけ受け取ることができ (該当サービス・ルーチンに入った時点で)、かつ応答も 1 つだけ送信することができます (該当サービス・ルーチンから戻る時点で)。一方、会話サービスは接続要求によって呼び出されます。このとき、オープンされた接続を参照する手段が必要です (すなわち、以後の接続ルーチンを呼び出す際に使用されるハンドル)。接続が確立され、サービス・ルーチンが呼び出されると、接続元プログラムあるいは会話サービスは、その接続が切断されるまでアプリケーションが定義するようにデータを送受信することができます。

なお、プログラムは要求 / 応答と会話によるコミュニケーションのいずれをも行うことができますが、両方のサービス要求を受け付けることはできません。以下の節では、これら 2 種類のコミュニケーション・パラダイムについてその概要を説明します。

注記 BEA Tuxedo ATMI のマニュアルでは、スレッドという用語をよく使用します。BEA Tuxedo システムは COBOL でのマルチスレッド化をサポートしていないので、「スレッド」という用語は、状況に応じてプロセス全体またはコンテキストのいずれかを指しているものと考えてください。以下に例を示します。

- 3 つのコンテキストに関連付けられた 3 つのスレッドを持つマルチスレッド化されたマルチコンテキストの C クライアントは、3 つのコンテキストを持つマルチコンテキストの COBOL クライアントに対応します。

- 単一のコンテキストに関連付けられた 3 つのスレッドを持つマルチスレッド化されたシングルコンテキストの C クライアントは、スレッド化されていないシングルコンテキストの COBOL クライアントに対応します。

クライアントおよびサーバに対する BEA Tuxedo の要求 / 応答パラダイム

要求 / 応答によるコミュニケーションの場合、クライアントは要求を送り、応答を受け取ることができる１つのプロセスと定義されています。定義によれば、クライアントは要求を受け取ったり、応答を送ったりすることはできません。その代わり、クライアントはいくつでも要求を送ることができ、またそれと同時に応答を待ったり、あるいは適宜ある一定数の応答を受け取ったりすることができます。場合によっては、クライアントは応答を必要としない要求を送ることもできます。`TPINITIALIZE(3)` と `TPTERM(3)` を使用すれば、クライアントは BEA Tuxedo ATMI のアプリケーションと結合および分離することができます。

一方、要求 / 応答型サーバは一度に１つのサービス要求を受け取り、その要求に対して１つの応答を返すことができるプログラムと定義されています。サーバは特定の要求を処理しながら、一方で要求 / 応答の要求あるいは会話要求を出し、それらの応答を受け取ることにより、クライアントのように働くこともできます。サーバはこうした能力の故に、リクエスタと呼ばれることもあります。ただし、クライアント・プログラムとサーバ・プログラムはどちらもリクエスタになることができます (実際、クライアントはリクエスタ以外の何物でもありません)。

要求 / 応答型サーバは別の要求 / 応答型サーバに要求を送る (転送する) ことができます。この場合、最初のサーバは受け取った要求を別のサーバに渡すだけで、応答を受け取ることは期待しません。この連鎖の中の最後サーバがその要求に対する応答をもとのリクエスタに送ります。この転送ルーチンの利用によって、もとのリクエスタは最終的にその応答を受け取ることができるのです。

サーバとサービス・ルーチンの利用から、BEA Tuxedo ATMI アプリケーションの作成に構造化手法をとることが可能になります。サーバ側では、アプリケーション作成者は、要求の受信や応答の送信といったコミュニケーションの詳細ではなく、サービスによって実行する事柄に関する作業に専念すればよいのです。コミュニケーション上の詳細の多くは BEA Tuxedo システムが処理するため、サービス・ルーチンを作成するときにはある一定の規則に従う必要があります。サーバは、そのサービス・ルーチンを終了する時点で、`TPRETURN()` を使用して応答を送ったり、あるいは `TPFORWAR()` を使用して要求を転送したりできます。サービスはその他の作業を行ったり、この時点以後別のプログラムとコミュニケートすることは許されません。そのため、サーバが実行するサービスは、要求が受け取られたときに開始され、応答が送信あるいは要求が転送された時点で終了します。

要求メッセージと応答メッセージとの間には、本質的に異なる点があります。すなわち、要求にはそれが送信される以前には関連するコンテキストはありませんが、応答にはあるという点です。たとえば、ある要求を送る際に、呼び出し元はアドレッシング情報を与えなければなりませんが、応答は常にその要求を出したプログラムに返されます。つまり、応答の場合には、要求が出されるときに与えられたアドレッシング情報が維持されていて、応答の送信側はそのあて先になんら手を加えることはできません。この両者の相違点については、TPCALL() のルーチンのパラメータと説明で明らかにされています。

要求メッセージはその送信時に特定の優先順位が付与されます。この優先順位にしたがって、要求はキューから取り出されます。つまり、サーバはキューの中で最も優先順位の高い要求から順に取り出すのです。ただし、要求がいつまでもキューの中に残されてしまうのを防ぐために、優先順位に関係なく、最も長くキューに入っている要求が一定間隔で取り出されます。デフォルトの設定では、要求の優先順位はその要求が送られるサービス名に対応させて付けられます。サービス名にはシステムのコンフィギュレーション時に優先順位を与えることができます (UBBCONFIG(5) 参照)。特に定義されていない場合には、デフォルトの優先順位が使用されます。この優先順位は、TPCALL() に説明のあるルーチン (TPSPRIO()) を使用して実行時に設定することができます。呼び出し元はこの方法により、メッセージ送信時にコンフィギュレーションあるいはデフォルトの優先順位を変更できます。

クライアントおよびサーバに対する BEA Tuxedo システムの会話パラダイム

会話型のコミュニケーションの場合、クライアントは、会話接続を行うことはできるが、接続要求を受け付けることはできないプログラムと定義されています。

一方、会話サーバは、接続要求を受け取ることができるプログラムです。接続が確立され、サービス・ルーチンが呼び出されると、接続元プログラムあるいは会話サービスは、その接続が切断されるまでアプリケーションが定義するようにデータを送受信することができます。会話は半二重方式で行われます。つまり、接続の一方の側が制御権をもってデータを送信し、他方の側は制御権を渡されるまではデータを送信できません。接続が確立されている間、サーバは「予約状態」になり、ほかのプログラムがそのサーバとの間で接続を確立することはできません。

要求 / 応答型サーバの場合と同様、会話サーバは他の要求を出したり、他のサーバとの接続を行うことによりリクエスタとして機能できます。一方、要求 / 応答型サーバと異なり、会話サーバは要求を別のサーバに転送することはできません。このため、会話サーバによって実行される会話サービスは、要求を受け取った時点で開始され、TPRETURN() を介して最終的な応答が送信された時点で終了します。

このため、会話サーバによって実行される会話サービスは、要求を受け取った時点で開始され、TPRETURN() を介して最終的な応答が送信された時点で終了します。メッセージは、アプリケーション側の規定に従って送受信することができます。要求メッセージ応答メッセージとの間には本質的な相違はなく、またメッセージの優先順位に関する規定もありません。

SYSTEM/T のキュー式メッセージ・モデル

BEA Tuxedo ATMI のキュー・メッセージ・モデルは、要求メッセージの完了を待たず、そのメッセージが後で処理されるようにキューに登録し、またオプションとしてキューに入れられた応答メッセージを介して応答が得られるようにします。メッセージをキューに登録したり応答をキューから取り出すための ATMI 関数は、TPENQUEUE( ) と TPDEQUEUE( ) です。これらの関数は、BEA Tuxedo ATMI のアプリケーション・プロセスの全ての型：クライアント、サーバ、会話型のいずれのプロセスからも呼び出せます。

キュー式メッセージ機能は、XA 準拠のリソース・マネージャです。永続的なメッセージはトランザクション内でキューへの登録および取り出しが行われ、高い信頼性で一度に処理されます。

ATMI トランザクション

BEA Tuxedo システムは、トランザクションの定義および管理について、相互に排他的な 2 つの関数をサポートしています。BEA Tuxedo システムの ATMI トランザクション境界関数 (名前の先頭が TP) と X/Open の TX インターフェイス関数 (名前の先頭が TX_) です。X/Open では TX インターフェイスのベースとして ATMI のトランザクション境界関数が使用されるので、TX インターフェイスの構文およびセマンティクスは、ATMI とほとんど同じです。この項では、ATMI のトランザクション概念について概要を述べます。次の項では、TX インターフェイスについて補足説明します。

BEA Tuxedo システムにおけるトランザクションは、全体としてある結果を導く、あるいは何も結果を示さない 1 つの論理的な作業単位を定義するときに使用します。トランザクションにより、多くのプロセスによって (そして、おそらく様々な場所で) なされる作業を 1 つの作業単位として扱うことができます。トランザクションの開始者は TPBEGIN( ) および TPCOMMIT( ) または TPABORT( ) を使用してトランザクション内での操作内容を記述します。

イニシエータはまた、TPSUSPEND( ) を呼び出して現在のトランザクションでの作業を中断することもできます。他のプロセスが TPRESUME( ) を呼び出して、中断されているトランザクションのイニシエータの役割を引き継ぐこともできます。トランザクションのイニシエータとして、プログラムは TPSUSPEND( )、TPCOMMIT( )、または TPABORT( ) のいずれかを呼び出す必要があります。従って、あるプロセスがトランザクションを終了し、別のプロセスがトランザクションを開始することができます。

サービスを呼び出すプログラムがトランザクション・モードにあると、呼び出されたサービス・ルーチンも同じトランザクションのためにトランザクション・モードに入ります。このプロセスがトランザクション・モードでない場合、サービスがトランザクション・モードで呼び出されるかどうかは、コンフィギュレーション・ファイルにおいて該当サービスにどのようなオプションが指定されているかによって決まります。トランザクション・モードで呼び出されないサービスは、それが呼び出された時点から終了時点までの間に複数のトランザクションを定義できます。一方、トランザクション・モードで呼び出されたサービス・ルーチンは、１つのトランザクションにのみ関与し、終了するまでそのトランザクションでの作業を続けます。なお、接続をトランザクション・モードにアップグレードすることはできません。会話中に TPBEGIN( ) が呼び出されると、会話はそのトランザクションの外側で維持されます (TPCONNECT( ) が TPNOTRAN を設定して呼び出された場合と同様)。

別のプログラムによって起動されたトランザクションに加わるサービスを、パーティシパントと呼びます。トランザクションは常に、１つの起動元をもち、かついくつかのパーティシパントをもつことができます。同じトランザクションでの作業を行うためにサービスを２回以上呼び出すことができます。TPCOMMIT( ) または TPABORT( ) を呼び出すことができるのは、トランザクションのイニシエータ (つまり、TPBEGIN( ) または TPRESUME( ) のいずれかを呼び出すプログラム) だけです。パーティシパントは、TPRETURN( ) あるいは TPFORWAR( ) を使用することによりトランザクションの結果に影響を与えます。これらの２つの呼び出しはそれぞれ、サービス・ルーチンの終わり、およびそのルーチンがトランザクションの中でそれが担当する部分を終了したことを示すものです。

TX トランザクション

TX インターフェイスによって定義されるトランザクションは、ATMI 関数によって定義されるトランザクションと実質的に同じです。アプリケーションの開発者は、クライアント・ルーチンおよびサービス・ルーチンを作成するにあたり、どちらの関数でも使用できます。実際、BEA Tuxedo システムでは、単一のアプリケーションにおけるクライアント・プログラムおよびサーバ・プログラムすべてに、必ずしもどちらかの１つの関数を使用する必要はありません。ただし、単一のプログラム内で２つの関数を一緒に使用することはできません。たとえば、同じプログラム内で TPBEGIN( ) を呼び出した後に TXCOMMIT( ) を呼び出すことはできません。

TX インターフェイスには、移植性の高い方法でリソース・マネージャのオープンとクローズを行う２つのコール、TXOPEN( ) および TXCLOSE( ) があります。トランザクションは、TXBEGIN( ) で開始され、TXCOMMIT( ) または TXROLLBACK( ) のいずれかで完了します。TXINFORM( ) はトランザクション情報を取り出すために使用されます。また、トランザクションにオプションを設定する３つの呼び出し TXSETCOMMITRET( )、TXSETTRANCTL( )、および TXSETTIMEOUT( ) があります。TX インターフェイスには、ATMI の TPSUSPEND( ) と TPRESUME( ) に相当するものはありません。

TX インターフェイスには、ATMI トランザクションについて定義されているセマンティクスおよび規則の他にも、ここで説明しておくべきセマンティクスがいくつかあります。まず、TX インターフェイスを使用するサービス・ルーチン開発者は、TXOPEN() を呼び出す独自の TPSVRINIT() ルーチンを使用する必要があります。省略時に BEA Tuxedo システムが提供する TPSVRINIT() は、TPOPEN() を呼び出します。TPSVRDONE() についても同じことがあてはまります。TX インターフェイスを使用している場合は、サービス・ルーチン開発者は、TXCLOSE() を呼び出す独自の TPSVRDONE() を使用しなければなりません。

次に、TX インターフェイスには、ATMI にはない別のセマンティクスが 2 つあります。それは、連鎖および非連鎖トランザクションと、トランザクション特性です。

連鎖および非連鎖トランザクション

TX インターフェイスは、トランザクション実行の連鎖モードおよび非連鎖モードをサポートしています。省略時には、クライアント・ルーチンおよびサービス・ルーチンは、非連鎖モードで実行されます。この場合、アクティブなトランザクションが完了した際、新しいトランザクションは TXBEGIN() が呼び出されるまで実行されません。

連鎖モードでは、新しいトランザクションは、現在のトランザクションが完了すると、暗黙に開始されます。つまり、TXCOMMIT() または TXROLLBACK() が呼び出されると、BEA Tuxedo システムは、現在のトランザクションの完了を調整し、制御を呼び出し元に返す前に新しいトランザクションを開始します。(異常終了の条件によっては、新しいトランザクションの開始が妨げられることもあります)。

クライアント・ルーチンおよびサービス・ルーチンは、TXSETTRANCTL() を呼び出すことによって連鎖モードのオンとオフを切り替えます。連鎖モードと非連鎖モードの間の移行により、その次の TXCOMMIT() コールまたは TXROLLBACK() コールの動作が変わります。TXSETTRANCTL() の呼び出しでは、呼び出し元のトランザクション・モードのオンとオフの切り替えは行いません。

TXCLOSE(3) は、呼び出し元がトランザクション・モードにあるときには呼び出すことができないため、連鎖モードで実行中の呼び出し元が TXCLOSE(3) を呼び出すには、非連鎖モードに切り替えて、現在のトランザクションを完了してから呼び出さなければなりません。

トランザクション特性

クライアント・ルーチンまたはサービス・ルーチンは、TXINFORM() を呼び出すことによって、そのルーチンのトランザクション特性の現在の値を取得したり、そのルーチンがトランザクション・モードで実行中であるかどうかを判別したりすることができます。

アプリケーション・プログラムの状態には、いくつかのトランザクション特性があります。呼び出し元は、TXSET* 関数の呼び出しによってこれらの特性を指定します。クライアント・ルーチンまたはサービス・ルーチンが特性の値を設定している場合は、異なる値を呼び出し元が指定するまでは、その値が有効のままです。呼び出し元が TXINFORM() を使用して特性の値を取得しても、これによって値が変更されることはありません。

タイムアウト

BEA Tuxedo ATMI システムには 3 種類のタイムアウトがあります。1 つはトランザクションの開始から終了までの期間に関連するもの、2 つめはブロッキング・コールで呼び出し元が制御権を再度入手するまでブロック状態を維持する最大時間に関連するものです。3 つめはサービスのタイムアウトです。これは呼び出しの秒数がコンフィギュレーション・ファイルの SERVICES セクションにおける SVCTIMEOUT パラメータで指定された秒数を越えた時に発生します。

最初のタイムアウトは、TPBEGIN() を使用してトランザクションを起動するときに指定します (詳細については、TPBEGIN() を参照)。2 つめのタイムアウトは、TPCALL(3) に定義されている BEA Tuxedo ATMI のコミュニケーション・ルーチンを使用する際に発生することがあります。これらのルーチンの呼び出し元は一般に、まだ届いていない応答を待っている間はブロック状態になります。これらの呼び出し元はデータの送信を行うこともブロックされることがあります (たとえば、要求キューがいっぱいの場合など)。呼び出し元がブロック状態になる最大時間は、BEA Tuxedo ATMI のコンフィギュレーション・ファイルに記述されているパラメータによって決まります。詳細については、UBBCONFIG(5) の BLOCKTIME パラメータの項を参照してください。

ブロッキング・タイムアウトは呼び出し者がトランザクション・モードにないときにデフォルトの設定によって実行されます。クライアントあるいはサーバがトランザクション・モードにあると、そのトランザクションが起動されたときに指定されたタイムアウト値が働き、UBBCONFIG ファイルに指定されているブロッキング・タイムアウト値の影響は受けません。

トランザクション・タイムアウトが発生すると、トランザクション・モードで行われた非同期の要求に対する応答は失効状態になることがあります。つまり、あるプログラムがトランザクション・モードで送られた要求に対する特定の非同期応答の到着を待っているときに、トランザクション・タイムアウトが発生すると、その応答のハンドルが無効になります。同様に、トランザクション・タイムアウトが発生すると、トランザクションに関連する通信ハンドルに対してイベントが生成され、そのハンドルは無効になります。一方、ブロッキング・タイムアウトが発生した場合、該当するハンドルは無効にならず、応答を待機しているプログラムはその応答を待機するための呼び出しを再度出すことができます。

サービス・タイムアウト機構によって、未知の、または予期しないシステム・エラーが原因でフリーズする可能性のあるプロセスについて、システムが強制終了を行うことができます。要求 / 応答サービス時にサービス・タイムアウトが発生すると、BEA Tuxedo システムによって、フリーズしたサービスを実行中のサーバ・プロセスが強制終了され、エラーコード TPESVCERR が戻ります。サービス・タイムアウトが会話型サービスで発生した場合は、TPEV_SVCERR イベントが返ります。

トランザクションがタイムアウトになった場合、トランザクションがアボートされる前に有効な通信は、TPNOREPLY、TPNOTRAN、および TPNOBLOCK を設定した TPACALL() の呼び出しだけです。

動的サービス宣言

デフォルトの設定では、サーバのサービスは、それがブートされるときに宣言され、シャットダウンするときに宣言が解除されます。サーバは、それが提供するサービス・セットに対する制御を実行時に必要とする場合、TPADVERTISE() および TPUNADVERTISE() を使用します。これらのルーチンは、該当サーバが複数サーバ / 単一キュー・セットに属していないかぎり、呼び出し元サーバが提供するサービスだけに影響します。MSSQ セットのサーバはすべて同じサービス・セットを提供しなければならないため、これらのルーチンもまた呼び出し元の MSSQ セットを共有するすべてのサーバの宣言に影響します。

型付きレコード

別のアプリケーション・プログラムにデータを送信するために、アプリケーション・プログラムは、まずそのデータをレコードに入れます。インターフェイスは、型付きレコードの概念をサポートしています。型付きレコードは、実際には、COBOL の 2 つのレコードからなります。データ・レコードは、静的領域内に定義され、別のアプリケーション・プログラムに渡すアプリケーション・データが入ります。データ・レコードには、補助型レコードが付加され、異なるマシン・バウンダリ間にまたがってデータ・レコードを渡す際に、BEA Tuxedo システムに対し、データ・レコードの解釈と変換の規則を識別します。補助型レコードには、データ・レコードのタイプ、任意選択のサブタイプ、および任意選択のデータ長が入ります。レコード・タイプによっては、特定のレコード・レイアウトなど、サブタイプによる追加の指定が必要となる場合があり、可変長レコードには長さを指定する必要があります。

アプリケーション・プログラマは、サポートされる 6 つの型付きレコードの中から 1 つを選択できます。BEA Tuxedo システムでは、ユーザ固有の型付きレコードを追加することもできます。詳細については、『BEA Tuxedo C 言語リファレンス』の「C 言語アプリケーション・トランザクション・モニタ・インターフェイスについて」を参照してください。*TPTYPE-REC* 内の REC-TYPE によって、アプリケーションで送信または受信しようとするレコード・タイプを選択します。さらに細かい分類が必要な場合には、*TPTYPE-REC* 内の SUB-TYPE も指定しなければなりません (たとえば、VIEW レコードなど)。*TPTYPE-REC* 内の LEN によって、送信時には送信するバイト数を示し、受信時にはユーザのレコードに移動するバイト数を示します。サポートされる REC-TYPE は次の通りです。

CARRAY
: レコード・タイプ CARRAY では、任意の文字数を入れることが可能です。レコード内のどこかに LOW-VALUE 文字列が入ります。データ送信時には、LEN に転送バイト数が入っていなければなりません。

STRING
: レコード・タイプ STRING は、任意の文字数を入れることが可能です。LOW-VALUE 文字列はレコード内には含まれず、レコードの末尾に付けられます。データ送信時には、LEN に転送バイト数が入っていなければなりません。

VIEW
: このレコード・タイプは、viewc() コンパイラを使用して生成された COBOL レコードを記述します。VIEW を使用する際には、SUB-TYPE に VIEW の名前が入っていなければなりません。VIEW タイプの送信時には、LEN に転送バイト数を入れるか、そうでなければ NO-LENGTH を設定しなければなりません。NO-LENGTH を設定すると、VIEW の長さ分が送信されます。

上記のレコード・タイプの中の 2 つには、同等のタイプがあります。X_OCTET は CARRAY と同じであり、X_COMMON は VIEW と同じです。X_COMMON は、VIEW がサポートするデータ・タイプのサブセットをサポートします。それらは、longs (PIC S9(9) COMP-5)、shorts (PIC S9(4) COMP-5)、および characters (PIC X(n)) です。X_COMMON は、C と COBOL の両方のプログラムがやりとりする際に、使用してください。

どのタイプの場合でも、正常に転送が行われると、LEN には転送されたバイト数が入ります。受信時には、LEN には、データ領域に入る最大バイト数が入っていなければなりません。正常な呼び出しの後、LEN は、データ領域に移動されたバイト数が入ります。着信メッセージのサイズが LEN で指定されているサイズより大きいと、LEN で指定されたデータ長分のみがデータ領域に移動され、残りのデータは破棄されます。

バッファ・タイプ・スイッチ

BEA Tuxedo システムでは、ユーザ固有のレコード・タイプを追加できます。詳細については、「C 言語アプリケーション・トランザクション・モニタ・インターフェイスについて」の「バッファ・タイプ・スイッチ」を参照してください。

プロセスごとのシングルコンテキストとマルチコンテキスト

BEA Tuxedo システムでは、クライアント・プログラムはプロセスごとに 1 つまたは複数のアプリケーションとの関連を作成することができます。TPINFDEF-REC の CONTEXTS-FLAG に TP-MULTI-CONTEXTS を設定して TPINITIALIZE() が呼び出されると、複数のクライアント・コンテキストを利用できます。TPINITIALIZE() が暗黙的に呼び出された場合、または CONTEXTS-FLAG に TP-MULTI-CONTEXTS が設定されていない場合は、1 つのアプリケーションとの関連しか作成できません。

シングルコンテキスト・モードで TPINITIALIZE() が 2 回以上呼び出された場合 (つまり、クライアントがアプリケーションに参加した後で呼び出された場合) は、アクションは何も実行されず、成功を示す戻り値が返されます。

マルチコンテキスト・モードでは、TPINITIALIZE() の呼び出しのたびに新しいアプリケーションとの関連が作成されます。プログラムは TPGETCTXT() を呼び出すことによりこのアプリケーション関連を表すハンドルを取得し、TPSETCTXT() を呼び出してそのコンテキストを設定することができます。

アプリケーションでシングルコンテキスト・モードが選択されると、すべてのアプリケーション関連が終了するまで、すべての TPINITIALIZE() 呼び出しでシングルコンテキスト・モードを指定する必要があります。同様に、アプリケーションでマルチコンテキスト・モードが選択されると、すべてのアプリケーション関連が終了するまで、すべての TPINITIALIZE() 呼び出しでマルチコンテキスト・モードを指定する必要があります。

サーバ・プログラムは 1 つのアプリケーションとしか関連付けられないため、クライアントとして機能することはありません。

注記 BEA Tuxedo システムでは、1 プロセスで複数のアプリケーション・コンテキストを利用できるほか、1 プロセスで複数のアプリケーション・スレッドを使用できます。ただし、マルチスレッド機能は C 言語インターフェイスでしかサポートされません。

次の表は、クライアント・プロセス内で発生する、非初期化状態、シングルコンテキスト・モードで初期化された状態、およびマルチコンテキスト・モードで初期化された状態の遷移を示しています。

プロセスごとのコンテキスト・モード

	状態		
関数	**非初期化 S_0**	**シングルコンテキスト・モードでの初期化 S_1**	**マルチコンテキスト・モードでの初期化 S_2**
TP-MULTI-CONTEXTS なし TPINITIALIZE()	S_1	S_1	S_2 (エラー)
TP-MULTI-CONTEXTS 設定の TPINITIALIZE()	S_2	S_1 (エラー)	S_2
暗黙的 TPINITIALIZE()	S_1	S_1	S_2 (エラー)
TPTERM() - 最後の関連でない			S_2
TPTERM() - 最後の関連		S_0	S_0
TPTERM() - 関連なし	S_0		

任意通知

上記のように定義されたクライアント / サーバ間でのやりとりの境界外からメッセージをアプリケーション・クライアントに送る方法は２通りあります。第１の方法は、TPBROADCAST( ) によって実現されるブロードキャスト機構です。この関数により、アプリケーション・クライアント、サーバおよび管理者は割り当てられた名前に基づいて選択されるクライアントに型付きレコード・メッセージをブロードキャストすることができます。クライアントに割り当てられる名前は、一部はアプリケーションにより (特に、*TPINFDEF-REC* データ構造体に TPINITIALIZE 実行時に渡される情報により)、また一部は (クライアントがアプリケーションのアクセスに使用するプロセッサに基づいて) システムにより決められます。

もう１つの方法は、以前のあるいは現在のサービス要求から識別される特定クライアントによる通知方法です。各サービス要求には、そのサービス要求を出したクライアントを特定する一意のクライアント識別子が含まれています。サービス・ルーチンの中から出される TPCALL( ) および TPFORWAR( ) 関数の呼び出しは、そのサービス要求の連鎖に対応する元のクライアントを変更しません。クライアント識別子は保存しておき、アプリケーション・サービス間で受け渡すことができます。この方法で識別されたクライアントに対する通知は、TPNOTIFY( ) 関数を使用して行います。

COBOL 言語 ATMI の戻り値とその他の定義

ATMI ルーチンは、次に挙げる戻り値と設定の定義を使用します。

```
*
* TPSTATUS.cbl
*
05 TP-STATUS       PIC S9(9) COMP-5.
  88 TPOK            VALUE 0.
  88 TPEABORT        VALUE 1.
  88 TPEBADDESC      VALUE 2.
  88 TPEBLOCK        VALUE 3.
  88 TPEINVAL        VALUE 4.
  88 TPELIMIT        VALUE 5.
  88 TPENOENT        VALUE 6.
  88 TPEOS           VALUE 7.
  88 TPEPERM         VALUE 8.
  88 TPEPROTO        VALUE 9.
  88 TPESVCERR       VALUE 10.
  88 TPESVCFAIL      VALUE 11.
  88 TPESYSTEM       VALUE 12.
  88 TPETIME         VALUE 13.
  88 TPETRAN         VALUE 14.
  88 TPEGOTSIG       VALUE 15.
  88 TPERMERR        VALUE 16.
  88 TPEITYPE        VALUE 17.
  88 TPEOTYPE        VALUE 18.
  88 TPERELEASE      VALUE 19.
  88 TPEHAZARD       VALUE 20.
```

```
  88 TPEHEURISTIC    VALUE 21.
  88 TPEEVENT        VALUE 22.
  88 TPEMATCH        VALUE 23.
  88 TPEDIAGNOSTIC   VALUE 24.
  88 TPEMIB          VALUE 25.
  88 TPEMAXVAL       VALUE 26.
05 TPEVENT         PIC S9(9) COMP-5.
  88 TPEV-NOEVENT    VALUE 0.
  88 TPEV-DISCONIMM  VALUE 1.
  88 TPEV-SENDONLY   VALUE 2.
  88 TPEV-SVCERR     VALUE 3.
  88 TPEV-SVCFAIL    VALUE 4.
  88 TPEV-SVCSUCC    VALUE 5.
05 TPSVCTIMOUT     PIC S9(9) COMP-5.
  88 TPED-NOEVENT    VALUE 0.
  88 TPEV-SVCTIMEOUT VALUE 1.
  88 TPEV-TERM       VALUE 2.
05 APPL-RETURN-CODE PIC S9(9) COMP-5.
```

TPTYPE COBOL 構造体は、アプリケーション・データの送受信時に必ず使用されます。REC-TYPE は、送信されるデータ・レコードのタイプを表します。SUB-TYPE は、VIEW REC-TYPE が指定されている場合の VIEW の名前を表します。LEN は、送信するデータのサイズおよび受信されるデータのサイズを表します。

```
*
* TPTYPE.cbl
*
05 REC-TYPE      PIC X(8).
  88 X-OCTET       VALUE "X_OCTET".
  88 X-COMMON      VALUE "X_COMMON".
05 SUB-TYPE      PIC X(16).
05 LEN           PIC S9(9) COMP-5.
  88 NO-LENGTH     VALUE 0.
05 TPTYPE-STATUS PIC S9(9) COMP-5.
  88 TPTYPEOK      VALUE 0.
  88 TPTRUNCATE    VALUE 1.
```

TPSVCDEF データ構造体は、BEA Tuxedo システムと設定値をやりとりする関数によって使用されます。

```
*
* TPSVCDEF.cbl
*
05 COMM-HANDLE         PIC S9(9) COMP-5.
05 TPBLOCK-FLAG        PIC S9(9) COMP-5.
  88 TPBLOCK            VALUE 0.
  88 TPNOBLOCK          VALUE 1.
05 TPTRAN-FLAG         PIC S9(9) COMP-5.
  88 TPTRAN             VALUE 0.
  88 TPNOTRAN           VALUE 1.
```

```
05 TPREPLY-FLAG        PIC S9(9) COMP-5.
  88 TPREPLY             VALUE 0.
  88 TPNOREPLY           VALUE 1.
05 TPACK-FLAG          PIC S9(9) COMP-5 REDEFINES TPREPLY-FLAG.
  88 TPNOACK             VALUE 0.
  88 TPACK               VALUE 1.
05 TPTIME-FLAG         PIC S9(9) COMP-5.
  88 TPTIME              VALUE 0.
  88 TPNOTIME            VALUE 1.
05 TPSIGRSTRT-FLAG     PIC S9(9) COMP-5.
  88 TPNOSIGRSTRT        VALUE 0.
  88 TPSIGRSTRT          VALUE 1.
05 TPGETANY-FLAG       PIC S9(9) COMP-5.
  88 TPGETHANDLE         VALUE 0.
  88 TPGETANY            VALUE 1.
05 TPSENDRECV-FLAG     PIC S9(9) COMP-5.
  88 TPSENDONLY          VALUE 0.
  88 TPRECVONLY          VALUE 1.
05 TPNOCHANGE-FLAG     PIC S9(9) COMP-5.
  88 TPCHANGE            VALUE 0.
  88 TPNOCHANGE          VALUE 1.
05 TPSERVICETYPE-FLAG  PIC S9(9) COMP-5.
  88 TPREQRSP            VALUE IS 0.
  88 TPCONV              VALUE IS 1.
*
05 APPKEY              PIC S9(9) COMP-5.
05 CLIENTID OCCURS 4 TIMES PIC S9(9) COMP-5.
05 SERVICE-NAME        PIC X(15).
```

TPINFDEF データ構造体は、TPINITIALIZE() によってアプリケーションに結合する際に使用されます。

```
*
* TPINFDEF.cbl
*
05 USRNAME            PIC X(30).
05 CLTNAME            PIC X(30).
05 PASSWD             PIC X(30).
05 GRPNAME            PIC X(30).
05 NOTIFICATION-FLAG  PIC S9(9) COMP-5.
  88 TPU-SIG           VALUE 1.
  88 TPU-DIP           VALUE 2.
  88 TPU-IGN           VALUE 3.
05 ACCESS-FLAG        PIC S9(9) COMP-5.
  88 TPSA-FASTPATH     VALUE 1.
  88 TPSA-PROTECTED    VALUE 2.
05 CONTEXTS-FLAG      PIC S9(9) COMP-5.
  88 TP-SINGLE-CONTEXT VALUE 0.
  88 TP-MULTI-CONTEXTS VALUE 1.
05 DATALEN            PIC S9(9) COMP-5.
```

TPCONTEXTDEF データ構造体は、TPGETCTXT( ) および TPSETCTXT( ) がプログラムのコンテキストを操作する際に使用されます。

```
      *
      *  TPCONTEXTDEF.cbl
      *
       05 CONTEXT     PIC S9(9) COMP-5.
```

TPQUEDEF データ構造体は、キューへのメッセージ登録に関連する情報を渡したり、取り出したりする際に、使用します。

```
*
* TPQUEDEF.cbl
*
05 TPBLOCK-FLAG                 PIC S9(9) COMP-5.
  88 TPNOBLOCK                    VALUE 0.
  88 TPBLOCK                      VALUE 1.
05 TPTRAN-FLAG                  PIC S9(9) COMP-5.
  88 TPNOTRAN                     VALUE 0.
  88 TPTRAN                       VALUE 1.
05 TPTIME-FLAG                  PIC S9(9) COMP-5.
  88 TPNOTIME                     VALUE 0.
  88 TPTIME                       VALUE 1.
05 TPSIGRSTRT-FLAG              PIC S9(9) COMP-5.
  88 TPNOSIGRSTRT                 VALUE 0.
  88 TPSIGRSTRT                   VALUE 1.
05 TPNOCHANGE-FLAG              PIC S9(9) COMP-5.
  88 TPNOCHANGE                   VALUE 0.
  88 TPCHANGE                     VALUE 1.
05 TPQUE-ORDER-FLAG             PIC S9(9) COMP-5.
  88 TPQDEFAULT                   VALUE 0.
  88 TPQTOP                       VALUE 1.
  88 TPQBEFOREMSGID               VALUE 2.
05 TPQUE-TIME-FLAG              PIC S9(9) COMP-5.
  88 TPQNOTIME                    VALUE 0.
  88 TPQTIME-ABS                  VALUE 1.
  88 TPQTIME-REL                  VALUE 2.
05 TPQUE-PRIORITY-FLAG          PIC S9(9) COMP-5.
  88 TPQNOPRIORITY                VALUE 0.
  88 TPQPRIORITY                  VALUE 1.
05 TPQUE-CORRID-FLAG            PIC S9(9) COMP-5.
  88 TPQNOCORRID                  VALUE 0.
  88 TPQCORRID                    VALUE 1.
05 TPQUE-REPLYQ-FLAG            PIC S9(9) COMP-5.
  88 TPQNOREPLYQ                  VALUE 0.
  88 TPQREPLYQ                    VALUE 1.
05 TPQUE-FAILQ-FLAG             PIC S9(9) COMP-5.
  88 TPQNOFAILUREQ                VALUE 0.
```

```
  88 TPQFAILUREQ                   VALUE 1.
05 TPQUE-MSGID-FLAG              PIC S9(9) COMP-5.
  88 TPQNOMSGID                    VALUE 0.
  88 TPQMSGID                      VALUE 1.
05 TPQUE-GETBY-FLAG              PIC S9(9) COMP-5.
  88 TPQGETNEXT                    VALUE 0.
  88 TPQGETBYMSGIDOLD              VALUE 1.
  88 TPQGETBYCORRIDOLD             VALUE 2.
  88 TPQGETBYMSGID                 VALUE 3.
  88 TPQGETBYCORRID                VALUE 4.
05 TPQUE-WAIT-FLAG               PIC S9(9) COMP-5.
  88 TPQNOWAIT                     VALUE 0.
  88 TPQWAIT                       VALUE 1.
05 TPQUE-DELIVERY-FLAG           PIC S9(9) COMP-5.
  88 TPQNODELIVERYQOS              VALUE 0.
  88 TPQDELIVERYQOS                VALUE 1.
05 TPQUEQOS-DELIVERY-FLAG        PIC S9(9) COMP-5.
  88 TPQQOSDELIVERYDEFAULTPERSIST  VALUE 0.
  88 TPQQOSDELIVERYPERSISTENT      VALUE 1.
  88 TPQQOSDELIVERYNONPERSISTENT   VALUE 2.
05 TPQUE-REPLY-FLAG               PIC S9(9) COMP-5.
  88 TPQNOREPLYQOS                 VALUE 0.
  88 TPQREPLYQOS                   VALUE 1.
05 TPQUEQOS-REPLY-FLAG            PIC S9(9) COMP-5.
  88 TPQQOSREPLYDEFAULTPERSIST     VALUE 0.
  88 TPQQOSREPLYPERSISTENT         VALUE 1.
  88 TPQQOSREPLYNONPERSISTENT      VALUE 2.
05 TPQUE-EXPTIME-FLAG             PIC S9(9) COMP-5.
  88 TPQNOEXPTIME                  VALUE 0.
  88 TPQEXPTIME-ABS                VALUE 1.
  88 TPQEXPTIME-REL                VALUE 2.
  88 TPQEXPTIME-NONE               VALUE 3.
05 TPQUE-PEEK-FLAG                PIC S9(9) COMP-5.
  88 TPQNOPEEK                     VALUE 0.
  88 TPQPEEK                       VALUE 1.
05 DIAGNOSTIC                     PIC S9(9) COMP-5.
  88 QMEINVAL                      VALUE -1.
  88 QMEBADRMID                    VALUE -2.
  88 QMENOTOPEN                    VALUE -3.
  88 QMETRAN                       VALUE -4.
  88 QMEBADMSGID                   VALUE -5.
  88 QMESYSTEM                     VALUE -6.
  88 QMEOS                         VALUE -7.
  88 QMEABORTED                    VALUE -8.
  88 QMEPROTO                      VALUE -9.
  88 QMEBADQUEUE                   VALUE -10.
  88 QMENOMSG                      VALUE -11.
  88 QMEINUSE                      VALUE -12.
  88 QMENOSPACE                    VALUE -13.
```

```
  88 QMERELEASE                    VALUE -14.
  88 QMEINVHANDLE                     VALUE -15.
  88 QMESHARE                         VALUE -16.
05 DEQ-TIME                       PIC S9(9) COMP-5.
05 EXP-TIME                       PIC S9(9) COMP-5.
05 PRIORITY                       PIC S9(9) COMP-5.
05 MSGID                          PIC X(32).
05 CORRID                         PIC X(32).
05 QNAME                          PIC X(15).
05 QSPACE-NAME                    PIC X(15).
05 REPLYQUEUE                     PIC X(15).
05 FAILUREQUEUE                   PIC X(15).
05 CLIENTID OCCURS4 TIMES         PIC S9(9) COMP-5.
05 APPL-RETURN-CODE               PIC S9(9) COMP-5.
05 APPKEY                         PIC S9(9) COMP-5.
```

TPSVCRET データ構造体は、TPRETURN() がトランザクションの状態を表す際に、使用されます。

```
*
* TPSVCRET.cbl
*
05 TP-RETURN-VAL                  PIC S9(9) COMP-5.
  88 TPSUCCESS                      VALUE 0.
  88 TPFAIL                         VALUE 1.
  88 TPEXIT                         VALUE 2.
05 APPL-CODE                      PIC S9(9) COMP-5.
```

TPTRXDEF データ構造体はトランザクション・タイムアウトを設定するために TPBEGIN()、トランザクション識別子を獲得するために TPSUSPEND()、設定するために TPRESUME() でそれぞれ使用されます。

```
*
* TPTRXDEF.cbl
*
05 T-OUT                         PIC S9(9) COMP-5 VALUE IS 0.
05 TRANID       OCCURS 6 TIMES   PIC S9(9) COMP-5.
```

TPCMTDEF データ構造体は、TPSCMT(3) がコミット・レベル特性を設定する際に、使用されます。

```
*
* TPCMTDEF.cbl
*
05 CMT-FLAG                       PIC S9(9) COMP-5.
  88 TP-CMT-LOGGED                  VALUE 1.
  88 TP-CMT-COMPLETE                VALUE 2.
05 PREV-CMT-FLAG                  PIC S9(9) COMP-5.
  88 PREV-TP-CMT-LOGGED             VALUE 1.
  88 PREV-TP-CMT-COMPLETE           VALUE 2.
```

TPAUTDEF データ構造体は、TPCHKAUTH(3) が認証の必要性を調べる際に、使用されます。

```
* TPAUTDEF.cbl
*
05 AUTH-FLAG                      PIC S9(9) COMP-5.
  88 TPNOAUTH                       VALUE 0.
  88 TPSYSAUTH                      VALUE 1.
  88 TPAPPAUTH                      VALUE 2.
```

TPPRIDEF データ構造体は、TPSPRIO() および TPGPRIO() がメッセージの優先順位を操作する際に、使用されます。

```
*
* TPPRIDEF.cbl
*
05 PRIORITY                       PIC S9(9) COMP-5.
05 PRIO-FLAG                      PIC S9(9) COMP-5.
  88 TPABSOLUTE                     VALUE 0.
  88 TPRELATIVE                     VALUE 1.
```

TPTRXLEV データ構造体は、TPGETLEV(3) がトランザクション・レベル設定を取り出す際に使用されます。

```
*
* TPTRXLEV.cbl
*
05 TPTRXLEV-FLAG     PIC S9(9) COMP-5.
  88 TP-NOT-IN-TRAN    VALUE 0.
  88 TP-IN-TRAN        VALUE 1.
```

TPBCTDEF データ構造体は、TPNOTIFY(3) および TPBROADCAST(3) が通知を送る際に、使用されます。

```
*
* TPBCTDEF.cbl
*
05 TPBLOCK-FLAG      PIC S9(9) COMP-5.
  88 TPBLOCK           VALUE 0.
  88 TPNOBLOCK         VALUE 1.
05 TPTIME-FLAG       PIC S9(9) COMP-5.
  88 TPTIME            VALUE 0.
  88 TPNOTIME          VALUE 1.
05 TPSIGRSTRT-FLAG   PIC S9(9) COMP-5.
  88 TPNOSIGRSTRT      VALUE 0.
  88 TPSIGRSTRT        VALUE 1.
05 LMID              PIC X(30).
05 USERNAME          PIC X(30).
05 CLTNAME           PIC X(30).
```

FML-INFO データ構造体は、FINIT()、FVSTOF() および FVFTOS() が FML バッファを扱う際に、使用されます。

```
*
* FMLINFO.cbl
*
05 FML-STATUS        PIC S9(9) COMP-5.
  88 FOK               VALUE 0.
  88 FALIGNERR         VALUE 1.
  88 FNOTFLD           VALUE 2.
  88 FNOSPACE          VALUE 3.
  88 FNOTPRES          VALUE 4.
  88 FBADFLD           VALUE 5.
  88 FTYPERR           VALUE 6.
  88 FEUNIX            VALUE 7.
  88 FBADNAME          VALUE 8.
  88 FMALLOC           VALUE 9.
  88 FSYNTAX           VALUE 10.
  88 FFTOPEN           VALUE 11.
  88 FFTSYNTAX         VALUE 12.
  88 FEINVAL           VALUE 13.
  88 FBADTBL           VALUE 14.
  88 FBADVIEW          VALUE 15.
  88 FVFSYNTAX         VALUE 16.
  88 FVFOPEN           VALUE 17.
  88 FBADACM           VALUE 18.
  88 FNOCNAME          VALUE 19.
  88 FEBADOP           VALUE 20.
*
05 FML-LENGTH        PIC S9(9) COMP-5.
*
05 FML-MODE          PIC S9(9) COMP-5.
  88 FUPDATE           VALUE 1.
  88 FCONCAT           VALUE 2.
  88 FJOIN             VALUE 3.
  88 FOJOIN            VALUE 4.
*
05 VIEWNAME          PIC X(33).
```

TPEVTDEF データ構造は、イベントの通知とサブスクリプションを処理するために TPPOST()、TPSUBSCRIBE()、および TPUNSUBSCRIBE() が使用します。

```
*
* TPEVTDEF.cbl
*
05 TPBLOCK-FLAG        PIC S9(9) COMP-5.
  88 TPBLOCK             VALUE 0.
  88 TPNOBLOCK           VALUE 1.
05 TPTRAN-FLAG         PIC S9(9) COMP-5.
  88 TPTRAN              VALUE 0.
```

```
  88 TPNOTRAN            VALUE 1.
05 TPREPLY-FLAG        PIC S9(9) COMP-5.
  88 TPREPLY             VALUE 0.
  88 TPNOREPLY           VALUE 1.
05 TPTIME-FLAG         PIC S9(9) COMP-5.
  88 TPTIME              VALUE 0.
  88 TPNOTIME            VALUE 1.
05 TPSIGRSTRT-FLAG     PIC S9(9) COMP-5.
  88 TPNOSIGRSTRT        VALUE 0.
  88 TPSIGRSTRT          VALUE 1.
05 TPEV-METHOD-FLAG    PIC S9(9) COMP-5.
  88 TPEVNOTIFY          VALUE 0.
  88 TPEVSERVICE         VALUE 1.
  88 TPEVQUEUE           VALUE 2.
05 TPEV-PERSIST-FLAG   PIC S9(9) COMP-5.
  88 TPEVNOPERSIST       VALUE 0.
  88 TPEVPERSIST         VALUE 1.
05 TPEV-TRAN-FLAG      PIC S9(9) COMP-5.
  88 TPEVNOTRAN          VALUE 0.
  88 TPEVTRAN            VALUE 1.
*
05 EVENT-COUNT         PIC S9(9) COMP-5.
05 SUBSCRIPTION-HANDLE PIC S9(9) COMP-5.
05 NAME-1              PIC X(31).
05 NAME-2              PIC X(31).
05 EVENT-NAME          PIC X(31).
05 EVENT-EXPR          PIC X(255).
05 EVENT-FILTER        PIC X(255).
```

TPKEYDEF データ構造体は、メッセージ・ベースのデジタル署名と暗号化を行うために使用する秘密鍵を管理する場合に、TPKEYCLOSE()、TPKEYGETINFO()、TPKEYOPEN()、および TPKEYSETINFO() によって使用されます。

```
*
*     TPKEYDEF.cbl
*
      05     KEY-HANDLE           PIC S9(9) COMP-5.
      05     PRINCIPAL-NAME       PIC X(512).
      05     LOCATION             PIC X(1024).
      05     IDENTITY-PROOF       PIC X(2048).
      05     PROOF-LEN            PIC S9(9) COMP-5.
      05     CRYPTO-PROVIDER      PIC X(128).
      05     SIGNATURE-FLAG       PIC S9(9) COMP-5.
             88 TPKEY-NOSIGNATURE   VALUE 0.
             88 TPKEY-SIGNATURE     VALUE 1.
      05     DECRYPT-FLAG         PIC S9(9) COMP-5.
             88 TPKEY-NODECRYPT     VALUE 0.
             88 TPKEY-DECRYPT       VALUE 1.
      05     ENCRYPT-FLAG         PIC S9(9) COMP-5.
```

```
           88 TPKEY-NOENCRYPT     VALUE 0.
           88 TPKEY-ENCRYPT       VALUE 1.
       05  AUTOSIGN-FLAG        PIC S9(9) COMP-5.
           88 TPKEY-NOAUTOSIGN    VALUE 0.
           88 TPKEY-AUTOSIGN      VALUE 1.
       05  AUTOENCRYPT-FLAG     PIC S9(9) COMP-5.
           88 TPKEY-NOAUTOENCRYPT VALUE 0.
           88 TPKEY-AUTOENCRYPT   VALUE 1.
       05  ATTRIBUTE-NAME       PIC X(64).
       05  ATTRIBUTE-VALUE-LEN  PIC S9(9) COMP-5.
```

COBOL 言語 TX の戻り値その他の定義

TX ルーチンは、次に挙げる戻り値と設定の定義を使用します。

```
*
* TXSTATUS.cbl
*
05 TX-STATUS      PIC S9(9) COMP-5.
  88 TX-NOT-SUPPORTED      VALUE 1.
*  正常実行
  88 TX-OK                 VALUE 0.
*  正常実行
  88 TX-OUTSIDE            VALUE -1.
*  アプリケーションが RM ローカル・トランザクションにある。
  88 TX-ROLLBACK           VALUE -2.
*  トランザクションがロールバックされた。
  88 TX-MIXED              VALUE -3.
*  トランザクションが部分的にコミットされ、部分的に
*  ロールバックされた。
  88 TX-HAZARD             VALUE -4.
*  トランザクションが部分的にコミットされ、部分的に
*  ロールバックされた可能性がある。
  88 TX-PROTOCOL-ERROR     VALUE -5.
*  ルーチンが不適切なコンテキストで呼び出された。
  88 TX-ERROR              VALUE -6.
*  一時的なエラー
  88 TX-FAIL               VALUE -7.
*  致命的なエラー
  88 TX-EINVAL             VALUE -8.
*  無効な引数が指定された。
  88 TX-COMMITTED          VALUE -9.
*  トランザクションがヒューリスティックにコミットされた。
  88 TX-NO-BEGIN           VALUE -100.
*  トランザクションはコミットされ、新しいトランザクションは
*  開始できなかった。
  88 TX-ROLLBACK-NO-BEGIN  VALUE -102.
*  トランザクションはロールバックされ、新しいトランザクションは
*  開始できなかった。
```

```
  88 TX-MIXED-NO-BEGIN     VALUE -103.
*  混合条件が発生し、新しいトランザクションが開始できなかった。
  88 TX-HAZARD-NO-BEGIN    VALUE -104.
*  ハザードがあり、新しいトランザクションが開始できなかった。
  88 TX-COMMITTED-NO-BEGIN VALUE -109.
*  ヒューリスティックにコミットされ、トランザクションが
*  開始できなかった。
```

TXINFDEF レコードは、TXINFORM() コールの結果が格納されるデータ構造体を定義します。

```
*
* TXINFDEF.cbl
*
05 XID-REC.
*  XID レコード
10 FORMAT-ID            PIC S9(9) COMP-5.
*  FORMAT-ID に値 -1 が入っている場合、XID は NULL
10 GTRID-LENGTH         PIC S9(9) COMP-5.
10 BRANCH-LENGTH        PIC S9(9) COMP-5.
10 XID-DATA             PIC X(128).
05 TRANSACTION-MODE     PIC S9(9) COMP-5.
*  トランザクション・モード設定
  88 TX-NOT-IN-TRAN            VALUE 0.
  88 TX-IN-TRAN                VALUE 1.
05 COMMIT-RETURN        PIC S9(9) COMP-5.
*   Commit_return 設定
  88 TX-COMMIT-COMPLETED       VALUE 0.
  88 TX-COMMIT-DECISION-LOGGED VALUE 1.
05 TRANSACTION-CONTROL  PIC S9(9) COMP-5.
*  Transaction_control 設定
  88 TX-UNCHAINED              VALUE 0.
  88 TX-CHAINED                VALUE 1.
05 TRANSACTION-TIMEOUT  PIC S9(9) COMP-5.
*  Transaction_timeout 値
  88 NO-TIMEOUT                VALUE 0.
05 TRANSACTION-STATE    PIC S9(9) COMP-5.
*  Transaction_state 情報
  88 TX-ACTIVE                 VALUE 0.
  88 TX-TIMEOUT-ROLLBACK-ONLY  VALUE 1.
  88 TX-ROLLBACK-ONLY          VALUE 2.
```

ATMI の状態遷移

BEA Tuxedo システムは、各プログラムの状態を記録し、各種の関数呼び出しやオプションごとに正当な状態遷移が行われているかどうかを検証します。この状態情報には、プログラムのタイプ (要求 / 応答型サーバ、会話サーバ、またはクライアント)、初期化状態 (初期化済み、非初期化)、リソースの管理状態 (クローズまたはオープン)、プログラムのトランザクション状態およびすべての非同期要求 / 応答および接続ハンドルの状態などがあります。不当な状態遷移が行われようとすると、呼び出された関数は異常終了し、*TPSTATUS-REC* に TPEPROTO( ) が設定されます。この情報に関する正規の状態と遷移について、次の表に示します。

この表は、要求 / 応答型サーバ、会話サーバ、およびクライアントがどの関数を呼び出すことができるかを示しています。ただし、TPSVRINIT( ) と TPSVRDONE( ) はこの表には示してありません。これらの関数はアプリケーションからは呼び出されないためです。これらはアプリケーションが提供する関数ですが、BEA Tuxedo システムによって呼び出されます。

使用可能な関数

関数	プログラム・タイプ		
	要求 / 応答型サーバ	会話サーバ	クライアント
TPABORT( )	Y	Y	Y
TPACALL( )	Y	Y	Y
TPADVERTISE( )	Y	Y	N
TPBEGIN( )	Y	Y	Y
TPBROADCAST( )	Y	Y	Y
TPCALL( )	Y	Y	Y
TPCANCEL( )	Y	Y	Y
TPCHKAUTH( )	Y	Y	Y
TPCHKUNSOL( )	N	N	Y
TPCLOSE( )	Y	Y	Y
TPCOMMIT( )	Y	Y	Y
TPCONNECT( )	Y	Y	Y
TPDEQUE( )	Y	Y	Y

使用可能な関数 (続き)

関数	プログラム・タイプ		
	要求 / 応答型サーバ	会話サーバ	クライアント
TPDISCON()	Y	Y	Y
TPENQUEUE()	Y	Y	Y
TPFORWAR()	Y	N	N
TPGETCTXT()	Y	Y	Y
TPGETLEV()	Y	Y	Y
TPGETRPLY()	Y	Y	Y
TPGPRIO()	Y	Y	Y
TPINITIALIZE()	N	N	Y
TPNOTIFY()	Y	Y	Y
TPOPEN()	Y	Y	Y
TPPOST()	Y	Y	Y
TPRECV()	Y	Y	Y
TPRESUME()	Y	Y	Y
TPRETURN()	Y	Y	N
TPSCMT()	Y	Y	Y
TPSEND()	Y	Y	Y
TPSETCTXT()	N	N	Y
TPSETUNSOL()	N	N	Y
TPSPRIO()	Y	Y	Y
TPSUBSCRIBE()	Y	Y	Y
TPSUSPEND()	Y	Y	Y
TPTERM()	N	N	Y

使用可能な関数 (続き)

関数	プログラム・タイプ		
	要求 / 応答型サーバ	会話サーバ	クライアント
TPUNADVERTISE()	Y	Y	N
TPUNSUBSCRIBE()	Y	Y	Y

以下に示す表は、特に明記されていないかぎり、クライアントとサーバ両方に適用されます。なお、関数によってはクライアントとサーバの両方が呼び出せるわけではないので (たとえば TPINITIALIZE())、以下の状態遷移のなかには、両方のプログラム・タイプには適用できないものもあります。上記の表を参照して、目的のプログラムから特定の関数を呼び出すことができるかどうかを判断するようにしてください。

次の表は、クライアント・プログラムがトランザクション・マネージャで初期化され登録されているかどうかを示しています。この表では、TPINITIALIZE() を使用しているものとします。TPINITIALIZE() は、シングルコンテキスト・モードでは任意選択可能です。つまり、シングルコンテキストのクライアントは、多数の ATMI 関数のどれか (たとえば、TPACALL() または TPCALL()) を発行することによって、暗黙的にアプリケーションに参加することができます。次のいずれかに該当する場合、クライアントは TPINITIALIZE() を使用する必要があります。

- アプリケーション認証が必要な場合 (TPINITIALIZE() および UBBCONFIG(5) の SECURITY キーワードの説明を参照。)
- クライアントから XA 準拠のリソース・マネージャに直接アクセスする場合 (TPINITIALIZE(3cbl) を参照。)
- クライアントが複数のアプリケーションとの関連を作成する場合。

サーバは TPSVRINIT() 関数が呼び出される前に BEA Tuxedo ディスパッチャによって初期化状態になり、TPSVRDONE() 関数が返された後、BEA Tuxedo ディスパッチャによって非初期化状態になります。なお、下記のすべての表において、関数がエラーを起こした場合、特に明記されていないかぎり、プログラムの状態は変わりません。

初期化状態

関数	状態	
	非初期化 I_0	初期化 I_1
TPCHKAUTH()	I_0	I_1

初期化状態

関数	状態	
	非初期化 I_0	初期化 I_1
TPGETCTXT()	I_0	I_1
TPINITIALIZE()	I_1	I_1
TPSETCTXT() ヌル以外のコンテキストに設定	I_1	I_1
TPNULLCONTEXT 設定の TPSETCTXT()	I_0	I_0
TPSETUNSOL()	I_0	I_1
TPTERM()	I_0	I_0
ほかのすべての ATMI 関数		I_1

以降の表は、事前条件として状態が I1 であると想定しています (TPINITIALIZE()、TPSETCTXT()、または BEA Tuxedo の サービス・ディスパッチャを介してこの状態でプログラムが到着したかどうかに関わりなく)。

次の表は、クライアントまたはサーバに対応するリソース・マネージャが初期化されているかいないかに応じてクライアントまたはサーバの状態を示しています。

リソース・マネージャの状態

関数	状態	
	クローズ R_0	オープン R_1
TPOPEN()	R_1	R_1
TPCLOSE()	R_0	R_0
TPBEGIN()		R_1
TPCOMMIT()		R_1

リソース・マネージャの状態

関数	状態	
	クローズ R_0	オープン R_1
TPABORT( )		R_1
TPSUSPEND( )		R_1
TPRESUME( )		R_1
TPTRAN 設定の TPSVCSTART( )		R_1
ほかのすべての ATMI 関数	R_0	R_1

次の表は、プログラムがトランザクションに対応しているかどうかに関してそのプログラムの状態を示したものです。サーバの場合、状態 T_1 と T_2 への遷移は、事前条件として状態 R_1 を想定しています (たとえば、TPOPEN( ) はそれ以降 TPCLOSE( ) または TPTERM( ) への呼び出しがないものとして呼び出されています)。

アプリケーション関連のトランザクション状態

関数	状態		
	トランザクション内ではない T_0	起動元 T_1	パーティシパント T_2
TPBEGIN( )			
TPABORT( )		T_0	
TPCOMMIT( )		T_0	
SPSUSPEND( )		T_0	
TPRESUME( )		T_0	
TPTRAN 設定の TPSVCSTART( )	T_2		

アプリケーション関連のトランザクション状態

関数	状態		
	トランザクション内ではない T_0	起動元 T_1	パーティシパント T_2
TPSVCSTART( ) (トランザクションモードでない)	T_0		
TPRETURN( )	T_0		T_0
TPFORWAR( )	T_0		T_0
TPCLOSE( )	R_0		
TPTERM( )	I_0	T_0	
ほかのすべての ATMI 関数	T_0	T_1	T_2

次の表は、TPACALL( ) が返す 1 つの要求ハンドルの状態を示すものです。

非同期要求ハンドルの状態

関数	状態	
	ハンドルなし A_0	有効なハンドル A_1
TPACALL( )	A_1	
TPGETRPLY( )		A_0
TPCANCEL( )		A_0[a]
TPABORT( )	A_0	A_0[b]
TPCOMMIT( )	A_0	A_0[b]
TPSUSPEND( )	A_0	A[c]
TPRETURN( )	A_0	A_0
TPFORWAR( )	A_0	A_0

非同期要求ハンドルの状態

関数	状態	
	ハンドルなし A_0	有効なハンドル A_1
TPTERM()	I_0	I_0
ほかのすべての ATMI 関数	A_0	A_1

注記 [a] この状態の遷移は、記述子が呼び出し元のトランザクションに関連しない場合にのみ起こります。

[b] この状態の遷移は、記述子が呼び出し元のトランザクションに関連する場合にのみ起こります。

[c] 記述子が呼び出し元のトランザクションに関連する場合、TPSUSPEND() はプロトコル・エラーを返します。

次の表は、TPCONNECT() によって返される接続記述子または TPSVCINFO 構造体でサービス呼び出しを行うことによって提供される接続記述子の状態を示したものです。接続ハンドルをとらないプリミティブの場合、特に明記されていないかぎり、状態の変化はすべての接続ハンドルに適用されます。

状態には次のものがあります。

C_0 - ハンドルなし
C_1 - TPCONNECT ハンドル送信専用
C_2 - TPCONNECT ハンドル受信専用
C_3 - TPSVCDEF ハンドル送信専用
C_4 - TPSVCDEF ハンドル受信専用

接続要求ハンドルの状態

関数 / イベント	状態				
	C_0	C_1	C_2	C_3	C_4
TPSENDONLY 設定の TPCONNECT()	C_1[a]				
TPRECVONLY 設定の TPCONNECT()	C_2[a]				
TPSENDONLY 設定の TPSVCSTART()	C_3[b]				
TPRECVONLY 設定の TPSVCSTART()	C_4[b]				
TPRECV()/ イベントなし			C_2		C_4
TPRECV()/TPEV_SENDONLY			C_1		C_3
TPRECV()/TPEV_DISCONIMM			C_0		C_0
TPRECV()/TPEV_SVCERR			C_0		
TPRECV()/TPEV_SVCFAIL			C_0		
TPRECV()/TPEV_SVCSUCC			C_0		
TPSEND()/ イベントなし		C_1		C_3	
TPRECVONLY 設定の TPSEND()		C_2		C_4	
TPSEND()/TPEV_DISCONIMM		C_0		C_0	
TPSEND()/TPEV_SVCERR		C_0			
TPSEND()/TPEV_SVCFAIL		C_0			
TPTERM() (クライアントのみ)	C_0	C_0			
TPCOMMIT() (オリジネータのみ)	C_0	C_0[c]	C_0 [c]		
TPSUSPEND() (オリジネータのみ)	C_0	C_0[d]	C_0[d]		
TPABORT() (オリジネータのみ)	C_0	C_0 [c]	C_0[c]		
TPDISCON()		C_0	C_0		
TPRETURN() (CONV サーバ)		C_0	C_0	C_0	C_0

接続要求ハンドルの状態 (続き)

関数 / イベント	状態				
	C_0	C_1	C_2	C_3	C_4
TPFORWAR( ) (CONV サーバ)		C_0	C_0	C_0	C_0
ほかのすべての ATMI 関数	C_0	C_1	C_2	C_3	C_4

注記 [a] プログラムがトランザクション・モードで TPNOTRAN の指定がない場合、接続はトランザクション・モードになります。

[b] TPTRAN が設定されている場合、接続はトランザクション・モードになります。

[c] 接続がトランザクション・モードにない場合、状態は変化しません。

[d] 接続がトランザクション・モードの場合、TPSUSPEND( ) はプロトコル・エラーを返します。

TX の状態遷移

BEA Tuxedo では、プロセスが必ず TX 関数を正しい順序で呼び出すことを確認します。不正の状態遷移が試行されると (つまり、ブランクの遷移エントリの状態から呼び出し)、呼び出された関数は、TX_PROTOCOL_ERROR を返します。TX 関数の正当な状態および遷移を、次の表に示します。異常終了を返す呼び出しの場合、この表で特に明記されていないかぎり、状態遷移は行われません。BEA Tuxedo のクライアントまたはサーバは、TX 関数を使用できます。

状態は、次のように定義されています。

S_0

どの RM もオープンまたは初期化が行われていません。プロセスは、TXOPEN( ) を正常に呼び出すまで、グローバル・トランザクションを開始できません。

S_1

プログラムは、RM をオープンしましたが、トランザクションには入っていません。transaction_control 特性は、TX-UNCHAINED です。

S_2

プログラムは、RM をオープンしましたが、トランザクションには入っていません。transaction_control 特性は、TX_CHAINED です。

S_3

プログラムが RM をオープンし、トランザクションには入っています。transaction_control 特性は、TX_UNCHAINED です。

S_4

プログラムが RM をオープンし、トランザクションには入っています。transaction_control 特性は、TX_CHAINED です。

TX の状態遷移

関数	状態				
	S_0	S_1	S_2	S_3	S_4
TXBEGIN()		S_3	S_4		
TXCLOSE()	S_0	S_0	S_0		
TXCOMMIT() -> TX_SET1				S_1	S_4
TXCOMMIT() -> TX_SET2					S_2
TXINFORM()		S_1	S_2	S_3	S_4
TXOPEN()	S_1	S_1	S_2	S_3	S_4
TXROLLBACK() -> TX_SET1				S_1	S_4
TXROLLBACK() -> TX_SET2					S_2
TXSETCOMMITRET()		S_1	S_2	S_3	S_4
TXSETTRANCTL() control = TX-CHAINED		S_2	S_2	S_4	S_4
TXSETRRANCTL() control = TX-UNCHAINED		S_1	S_1	S_3	S_3
TXSETTIMEOUT(3c)		S_1	S_2	S_3	S_4

- TX_SET1 は、TX_OK、TX_ROLLBACK、TX_MIXED、TX_HAZARD、TX_COMMITTED のいずれかを表します。TX_ROLLBACK は tx_rollback() からは返されず、TX-COMMITTED は tx_commit() からは返されません。
- TX_SET2 は、TX_NO_BEGIN、TX_ROLLBACK_NO_BEGIN、TX_MIXED_NO_BEGIN、TX_HAZARD_NO_BEGIN、TX_COMMITTED_NO_BEGIN のいずれかを表します。TX_ROLLBACK_NO_BEGIN は tx_rollback() からは返されず、TX-COMMITTED-NO-BEGIN は tx_commit() からは返されません。

- TX_FAIL は、いずれかの呼び出しから返された場合には、アプリケーション・プログラムの状態は、上記の表では未定義の状態になります。
- tx_info() がトランザクション状態情報に TX_ROLLBACK_ONLY または TX_TIMEOUT_ROLLBACK_ONLY を返した場合、トランザクションは「ロールバックのみ」とマークされ、アプリケーション・プログラムが tx_commit() と tx_rollback() のどちらを呼び出したかに関係なく、ロールバックされます。

関連項目

buffer(3c)、TPINITIALIZE(3cbl)、TPADVERTISE(3cbl)、TPBEGIN(3cbl)、TPCALL(3cbl)、TPCONNECT(3cbl)、TPGETCTXT(3cbl)、TPKEYCLOSE(3cbl)、TPKEYGETINFO(3cbl)、TPKEYOPEN(3cbl)、TPKEYSETINFO(3cbl)、TPOPEN(3cbl)、TPSETCTXT(3cbl)、TPSVCSTART(3cbl)、tuxtypes(5)、typesw(5)

FINIT、FINIT32(3cbl)

名前

FINIT()、FINIT32() - フィールド化バッファの初期化

形式

```
01 FML-BUFFER.
 05 FML-ALIGN      PIC S9(9) USAGE IS COMP.
 05 FML-DATA       PIC X(applen).

01 FML-REC
 COPY FMLINFO.

CALL "FINIT" USING FML-BUFFER FML-REC.

CALL "FINIT32" USING FML-BUFFER FML-REC.
```

機能説明

FINIT() は、フィールド化バッファの初期化をする場合に呼び出すことができます。*FML-BUFFER* は、フィールド化バッファとして使用するレコードであり、FML16 および FML32 と合わせて使用するためには、4 バイトの境界にアライメントされている必要があります。これは、上の「形式」に示すようにレコードの要素を 2 つ定義することで可能になります。*FML-REC* 中の FML-LENGTH はレコードの長さです。フィールド化バッファに対応して、内部構造体がフィールドなしで設定されます。アプリケーション・プログラムはレコードを FINT()、FVFTOS()、または FVSTOF() に渡したり、型付きレコードが必要な ATMI 呼び出しに渡す (この場合、タイプは "FML" でサブタイプはありません) 場合を除いて、レコードを解釈してはなりません。

FINIT32() は 32 ビット FML で使用します。

戻り値

FINIT() は正常終了時には、*FML-REC* 内の FML-STATUS に FOK を設定します。

エラーが発生した場合は、FML-STATUS は 0 以外の値に設定されます。

エラー

次の条件が発生すると、FINIT() は異常終了し、*FML-REC* 内の FML-STATUS に次の値を設定します。

[FALIGNERR]
"fielded buffer not aligned"
バッファの先頭が正しい境界上にありません。

[FNOSPACE]
"no space in fielded buffer"
指定されたバッファのサイズは、フィールド化バッファには小さすぎます。

使用例　　バッファからフィールドを削除する再初期化のための正しい方法を以下に示します。

```
Finit(frfr, (FLDLEN)Fsizeof(fbfr));
```

関連項目　　「FML 関数の紹介」

FVFTOS、FVFTOS32(3cbl)

名前

FVFTOS()、FVFTOS32() - フィールド化バッファから COBOL 構造体にコピーする

形式

```
01 DATA-REC.
COPY User data.

01 FML-BUFFER.
 05 FML-ALIGN      PIC S9(9) USAGE IS COMP.
 05 FML-DATA       PIC X(applen).

01 FML-REC COPY FMLINFO.

CALL "FVFTOS" USING FML-BUFFER DATA-REC FML-REC.

CALL "FVFTOS32" USING FML-BUFFER DATA-REC FML-REC.
```

機能説明

FVFTOS() 関数はフィールド化バッファから COBOL のレコードにデータを転送します。*FML-BUFFER* は、FINIT() で初期化されたフィールド化バッファを指すポインタです。*DATA-REC* は、C の構造体へのポインタです。*FML-REC* 中の VIEWNAME は、COBOL のレコードを記述する VIEW の名前です。

各フィールドは、フィールド化バッファから、*VIEWNAME* での要素の記述に基づいた構造体にコピーされます。フィールド化バッファのフィールドに対応する要素が COBOL のレコードに存在しない場合は、そのフィールドは無視されます。COBOL のレコードで指定された要素に対応するフィールドがフィールド化バッファに存在しない場合は、その要素には NULL 値がコピーされます。使用する NULL の値は、各要素ごとに VIEW 記述で定義できます。

複数のオカレンスを COBOL のレコードに格納するには、レコードの要素を OCCURS で定義してください。バッファ中のフィールドのオカレンスが、要素のオカレンスよりも少ない場合は、余分な要素スロットには NULL の値が代入されます。反対に、バッファ中のフィールドのオカレンスが、要素のオカレンスよりも多い場合は、余分なオカレンスは無視されます。

FVFTOS32() は、より多くのフィールドを持つ大きな view 用の view32() 型付きバッファで定義された view のために使用されます。

戻り値

FVFTOS32() は正常終了時には、*FML-REC* 内の FML-STATUS に FOK を設定します。

エラーが発生した場合は、FML-STATUS は 0 以外の値に設定されます。

エラー

次の条件が発生すると、FVFTOS() は異常終了し、FML-STATUS に次の値を設定します。

[FALIGNERR]
"fielded buffer not aligned"
バッファの先頭が正しい境界上にありません。

[FNOTFLD]
"buffer not fielded"
バッファがフィールド化されていないか、または Finit() で初期化されていません。

[FEINVAL]
"invalid argument to function"
呼び出された関数の引数の 1 つが無効です。

[FBADACM]
"ACM contains negative value"
COBOL のレコードからフィールド化バッファにデータを移動させるときには、関連カウント・メンバ (ACM) は負の値であってはいけません。

[FBADVIEW]
"cannot find or get view"
VIEWNAME の VIEW 記述が、VIEWDIR または VIEWFILES で指定されたファイルに見つかりません。

関連項目

「FML 関数の紹介」、viewfile(5)

FVSTOF(3cbl)

名前　　FVSTOF() - C 構造体からフィールド化バッファにコピーする

形式

```
01 DATA-REC.
 COPY User data.

01 FML-BUFFER.
 05 FML-ALIGN      PIC S9(9) USAGE IS COMP.
 05 FML-DATA       PIC X(applen).

01 FML-REC
 COPY FMLINFO.

CALL "FVSTOF" USING FML-BUFFER DATA-REC FML-REC.

CALL "FVSTOF32" USING FML-BUFFER DATA-REC FML-REC.
```

機能説明

FVSTOF() は COBOL のレコードからフィールド化バッファにデータを転送します。*FML-BUFFER* は、フィールド化バッファを含むレコードです。*DATA-REC* は、COBOL のレコードです。*FML-REC* 中の VIEWNAME は、COBOL のレコードを記述する VIEW の名前です。*FML-REC* 中の FML-MODE には、データを転送する方式を指定します。FML-MODE には次の 4 種類の値のいずれかを設定できます。

```
FUPDATE
FOJOIN
FJOIN
FCONCAT
```

これらのモードの動作は、Fupdate、Fupdate32(3fml)、Fojoin、Fojoin32(3fml)、Fjoin、Fjoin32(3fml)、および Fconcat、Fconcat32(3fml) で説明したものと同じです。これらの関数ではソース・バッファを指定するのに対して、FVSTOF() では COBOL のレコードを指定することを除けば、FVSTOF() はこれらの関数と同じものと考えることもできます。FUPDATE は NULL の値を持つレコードの要素を移動させないこと憶えていてください。

FVSTOF32() は、より多くのフィールドを持つ大きな view 用の view32() 型付きバッファで定義された view のために使用されます。

戻り値　　FVSTOF32() は正常終了時には、*FML-REC* 内の FML-STATUS に FOK を設定します。

エラーが発生した場合は、FML-STATUS は 0 以外の値に設定されます。

エラー

次の条件が発生すると、FVSTOF() は異常終了し、FML-STATUS に次の値を設定します。

[FALIGNERR]
"fielded buffer not aligned"
バッファの先頭が正しい境界上にありません。

[FNOTFLD]
"buffer not fielded"
バッファがフィールド化されていないか、または Finit() で初期化されていません。

[FEINVAL]
"invalid argument to function"
呼び出された関数の引数の 1 つが無効です。

[FBADACM]
"ACM contains negative value"
COBOL のレコードからフィールド化バッファにデータを移動させるときには、関連カウント・メンバ (ACM) は負の値であってはいけません。

[FBADVIEW]
"cannot find or get view"
VIEWNAME の VIEW 記述が、VIEWDIR または VIEWFILES で指定されたファイルに見つかりません。

関連項目

「FML 関数の紹介」、viewfile(5)

TPABORT(3cbl)

名前

TPABORT( ) - 現在の BEA Tuxedo ATMI のトランザクションのアボート

形式

```
01 TPTRXDEF-REC.
  COPY TPTRXDEF.

01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPABORT" USING TPTRXDEF-REC TPSTATUS-REC.
```

機能説明

TPABORT( ) は、トランザクションの中途終了を指定します。このルーチンが終了すると、そのトランザクションでなされたリソースへの変更内容はすべて取り消されます。TPCOMMIT(3) と同様、このルーチンはトランザクションの実行元しか呼び出せません。パーティシパント (サービス・ルーチン) は、トランザクションをアボートさせたい場合、TPFAIL( ) を設定して TPRETURN( ) を呼び出します。

未処理の応答に対する通信ハンドルが存在するときに TPABORT( ) を呼び出すと、このルーチンの終了時にトランザクションはアボートし、呼び出し元のトランザクションに関連する通信ハンドルは以後無効になります。呼び出し元のトランザクションと無関係の通信ハンドルは有効なままです。

トランザクション・モードの会話サーバに対してオープン接続がある場合、TPABORT( ) は TPEV-DISCONIMM イベントをサーバに送ります (そのサーバが接続の制御権を有するかどうかに関係なく)。TPBEGIN( ) の前に、あるいは TPNOTRAN 設定を付けて (つまり、トランザクション・モードにない状態で) オープンした接続は、影響を受けません。

現時点では、TPABORT( ) の引数 *TPTRXDEF-REC* は将来使用するために予約されています。

戻り値

TPABORT( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPABORT( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。呼び出し元のトランザクションは影響を受けません。

[TPEHEURISTIC]
: ヒューリスティックな判断のため、トランザクションの一部としてなされた作業が一部はコミットされ、一部は中途終了しています。

[TPEHAZARD]
ある種の障害のため、トランザクションの一部としてなされた作業がヒューリスティックに完了している可能性があります。

[TPEPROTO]
tpabort() が不正なコンテキストで呼び出されました (パーティシパントによって呼び出されるなど)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

注意事項

TPBEGIN()、TPCOMMIT() および TPABORT() を使用して BEA Tuxedo ATMI のトランザクションを記述する際には、XA インターフェイスに準拠した (および呼び出し元に適切にリンクされている) リソース・マネージャが行う作業がトランザクションとしての特性を備えていることを忘れないようにすることが重要です。トランザクションで行われるその他の処理内容は、TPCOMMIT() や TPABORT の影響を受けません。

関連項目

TPBEGIN(3cbl)、TPCOMMIT(3cbl)、TPGETLEV(3cbl)

TPACALL(3cbl)

名前　　TPACALL() - サービスへのメッセージの非同期送信を行うルーチン

形式

```
01 TPSVCDEF-REC.
  COPY TPSVCDEF.

01 TPTYPE-REC.
  COPY TPTYPE.

01 DATA-REC.
  COPY User data.

01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPACALL" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　TPACALL() は、*TPSVCDEF-REC* 内の SERVICE-NAME で指定されているサービスに要求メッセージを送ります。この要求は、以前になされた TPSPRIO() の呼び出しで変更されていないかぎり、SERVICE-NAME に定義されている優先順位で送信されます。*DATA-REC* は、送信するメッセージであり、*TPTYPE-REC* 内の LEN は、*DATA-REC* に入るデータの大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES の場合は、*DATA-REC* および LEN は無視され、要求はデータ部なしで送信されます。REC-TYPE が STRING で、LEN が 0 の場合は、要求はデータ部なしで送信されます。*DATA-REC* の REC-TYPE および SUB-TYPE は、SERVICE-NAME が認識する REC-TYPE および SUB-TYPE のいずれかと一致しなければなりません。トランザクション・モードにあるときに送信される要求ごとに、最終的には対応する応答が受信されなければなりません。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、この設定を使用していると、SERVICE-NAME が呼び出されても、呼び出し元のトランザクションの一部として実行されません。トランザクションをサポートしないサーバに SERVICE-NAME が属しており、呼び出し元がトランザクション・モードにある場合は、この設定を指定しなければなりません。このフラグ設定を使用するトランザクション・モードの呼び出し元は、依然としてトランザクション・タイムアウトの対象となります (それ以外はなし)。この設定を使用した状態で呼び出されたサービスが正常に実行できない場合、呼び出し元のトランザクションは影響を受けません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、SERVICE-NAME が呼び出されたときに、このプログラムは呼び出し元のトランザクションのために実行されます。呼び出し元がトランザクション・モードにない場合、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPNOREPLY
: 応答を期待していないことを tpacall() に通知します。TPNOREPLY が設定されると、このルーチンは、正常終了時には [TPOK] を返し、*TPSVCDEF-REC* 内の COMM-HANDLE に 0 を設定します。0 は、無効な通信ハンドルです。呼び出し元がトランザクション・モードにある場合は、TPTRAN も設定するときには、この設定は使用できません。TPNOREPLY または TPREPLY が設定されていなければなりません。

TPREPLY
: 応答を期待していることを TPACALL() に通知します。TPREPLY が設定されると、このルーチンは、正常終了時には [TPOK] を返し、COMM-HANDLE に有効な通信ハンドルを設定します。呼び出し元がトランザクション・モードにあり、TPTRAN がセットされている場合は、このフラグを設定する必要があります。TPNOREPLY または TPREPLY が設定されていなければなりません。

TPNOBLOCK
: この要求は、ブロッキング条件が存在する場合 (たとえば、メッセージの送信先である内部バッファがいっぱいの場合など) には、送信されません。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

戻り値

TPACALL( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。また、*TPSVCDEF-REC* に TPREPLY が設定されている場合には、TPACALL( ) は、COMM-HANDLE に通信ハンドルを返します。このハンドルは、送信した要求に対する応答を受信する際に使用できる有効なハンドルです。

エラー

次の条件が発生すると、TPACALL( ) は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました (*TPSVCDEF-REC* の設定が無効など)。

[TPENOENT]
: 存在しないか、要求 / 応答型サービスでないため (すなわち、会話サービス)、SERVICE-NAME に送信できません。

[TPEITYPE]
: REC-TYPE および SUB-TYPE が、SERVICE-NAME が受け付けるタイプおよびサブタイプではありません。

[TPELIMIT]
未終了の非同期要求が、保持できる最大数に達したため、呼び出し元の要求が送信できませんでした。

[TPETRAN]
SERVICE-NAME は、トランザクションをサポートしていないサーバに属していて、TPTRAN が設定されていました。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションは「アボートのみ」とマークされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、新しいリクエストを送信したり、未処理の応答を受信しようとしても、[TPETIME] が発生して失敗します。

また、[TPETIME] は、現在 TX_ROLLBACKONLY 状態のトランザクション内でサービスが異常終了したことを示すこともあります。トランザクションが TX_ROLLBACKONLY 状態である限り、TPACALL( ) を呼び出すと必ず [TPETIME] が返されます。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT がセットされていました。

[TPEPROTO]
TPACALL( ) の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　TPCALL(3cbl)、TPCANCEL(3cbl)、TPGETRPLY(3cbl)、TPGPRIO(3cbl)、TPSPRIO(3cbl)

TPADVERTISE(3cbl)

名前　TPADVERTISE( ) - サービス名の宣言を行うルーチン

形式

```
01 SVC-NAME     PIC X(15).
01 PROGRAM-NAME   PIC X(32).
01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPADVERTISE" USING SVC-NAME PROGRAM-NAME TPSTATUS-REC.
```

機能説明

TPADVERTISE( ) は、サーバが提供するサービスを宣言するときに使用します。デフォルトの設定では、サーバのサービスは、サーバのブート時に宣言され、サーバのシャットダウン時にその宣言が解除されます。

複数サーバ単一キュー (MSSQ) セットに属するすべてのサーバは、同じサービス・セットを提供しなければなりません。これらのルーチンは、MSSQ セットを共有する全サーバを宣言することによってこの規則を適用します。

TPADVERTISE( ) は、サーバ (または、呼び出し元の MSSQ セットを共有するサーバ・セット) の *SVC-NAME* を宣言します。*SVC-NAME* の長さは 15 文字以内にしてください。これに、SPACES を指定することはできません (UBBCONFIG(5) の SERVICES セクションを参照)。これより長い名前は、15 文字までで切り捨てられてしまいます。このため、切り捨てられた名前が他のサービス名と同じにならないよう、注意が必要です。*PROGRAM-NAME* は BEA Tuxedo ATMI のサービス・プログラムの名前です。このプログラムは、*SVC-NAME* に対する要求をサーバが受け取ったときに起動されます。*PROGRAM-NAME* に SPACES を指定することはできません。

SVC-NAME が既にサーバに対して宣言されていて、*PROGRAM-NAME* がその現在のプログラムと一致する場合 (既に宣言されていた名前と切り捨てられた名前が一致する場合を含む)、TPADVERTISE( ) は正常終了します。ただし、*SVC-NAME* がすでにサーバに対して宣言されていても、*PROGRAM-NAME* が現在のプログラムと一致しない場合には (すでに宣言されていた名前と切り捨てられた名前が一致する場合も含む)、エラーが返されます。

戻り値　TPADVERTISE( ) TPADVERTISE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPADVERTISE() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
SVC-NAME または *PROGRAM-NAME* が SPACES である場合、または *PROGRAM-NAME* が有効なプログラムの名前でない場合。

[TPELIMIT]
十分な空き領域がないため *SVC-NAME* を宣言できない場合 (UBBCONFIG(5) の *RESOURCES セクションの MAXSERVICES を参照)。

[TPEMATCH]
SVC-NAME がサーバに対してすでに宣言されているが、*PROGRAM-NAME* 以外のプログラムで宣言されている場合。TPADVERTISE() が異常終了しても、*SVC-NAME* はその現在のプログラムで宣言されたまま変わりません。つまり、現在のプログラムは *PROGRAM-NAME* に置き換えられません。

[TPEPROTO]
TPADVERTISE() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

移植性

RS6000 上の AIX では、最初の COBOL オブジェクト・ファイルで提供されるどのサービスもシンボル・テーブルで利用できません。buildserver コマンドで -s オプションを指定してサービス名を特定し、サービス名が TPADVERTISE() により実行時に宣言されるようにする必要があります。

関連項目

TPUNADVERTISE(3cbl)

TPBEGIN(3cbl)

名前　TPBEGIN() - BEA Tuxedo ATMI のトランザクションを開始するルーチン

形式

```
01 TPTRXDEF-REC.
  COPY TPTRXDEF.

01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPBEGIN" USING TPTRXDEF-REC TPSTATUS-REC.
```

機能説明

BEA Tuxedo システムにおけるトランザクションは、完全に成功するか、あるいは何も影響を残さない 1 つの論理的な作業単位を定義するときに使用します。トランザクションにより、多くのプロセスで (そして、多分様々なサイトで) 行われる作業を 1 つの分割できない単位として扱うことができます。トランザクションのイニシエータは、TPBEGIN() と TPCOMMIT() または TPABORT() を使用して、トランザクション内の処理を記述します。TPBEGIN() が呼び出された後は、ほかのプログラムとの通信は、後のプログラム (つまり、サーバ) を「トランザクション・モード」にすることが可能です。つまり、サーバの作業はトランザクションの一部になります。トランザクションに参加した制御のスレッドをパーティシパントと呼びます。トランザクションには、必ず 1 つの実行元があり、いくつかのパーティシパントをもつことができます。TPCOMMIT() または TPABORT() を呼び出せるのは、トランザクションの実行元だけです。パーティシパントは、それらが TPRETURN() を呼び出したときに、*TPSVCDEF-REC* の設定によってトランザクションの結果に影響することがあります。トランザクション・モードに入ると、サーバに出されたサービス要求はすべて、トランザクションの一部として処理されます (明示的にリクエスタからのそれ以外の指定がない場合)。

また、会話サーバに対して確立されたオープン接続があるときにプログラムがトランザクションを起動しても、これらの接続はトランザクション・モードには変わりません。これは、TPCONNECT() の呼び出し時に TPNOTRAN 設定を指定したことと同じです。

T-OUT は、トランザクションのタイムアウトまでの時間を最低 T-OUT 秒にすることを指定します。トランザクションはタイムアウト時間を経過した後は、中途終了しなければなりません。T-OUT の値が 0 であると、トランザクションにはシステムが許す最大時間 (秒単位) のタイムアウト値が与えられます。つまり、このときのタイムアウト値は、システムが定義する符号なし long 型の最大値になります。

戻り値　TPBEGIN() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPBEGIN() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
無効な引数が指定されました。

[TPETRAN]
呼び出し元は、トランザクション起動時にエラーが発生したため、トランザクション・モードになれません。

[TPEPROTO]
tpbegin() が不正なコンテキストで呼び出されました (たとえば、呼び出し元がすでにトランザクション・モードにある)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

注意事項

TPBEGIN()、TPCOMMIT()、および TPABORT() を使用して BEA Tuxedo ATMI のトランザクションを記述する際には、XA インターフェイスに準拠した (および呼び出し元に適切にリンクされている) リソース・マネージャが行う作業がトランザクションとしての特性を備えていることを覚えておくことが重要です。トランザクションで行われるその他の処理内容は、TPCOMMIT() や TPABORT() の影響を受けません。そのリソース・マネージャが行った処理が BEA Tuxedo ATMI のトランザクションの一部となるよう、XA インターフェイスを満たすリソース・マネージャをサーバにリンクする方法については、buildserver(1) を参照してください。

関連項目

TPABORT(3cbl)、TPCOMMIT(3cbl)、TPGETLEV(3cbl)、TPSCMT(3cbl)

TPBROADCAST(3cbl)

名前　　TPBROADCAST() - 名前によって通知をブロードキャストする

形式

```
01 TPBCTDEF-REC.
  COPY TPBCTDEF.

01 TPTYPE-REC.
  COPY TPTYPE.

01 DATA-REC.
  COPY User data.

01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPBROADCAST" USING TPBCTDEF-REC TPTYPE-REC DATA-REC
TPSTATUS-REC.
```

機能説明

tpbroadcast() は、クライアント・プロセスやサーバ・プロセスがシステム内に登録されているクライアントに任意通知型メッセージを送ることができるようにします。ターゲット・クライアント・セットは、tpbroadcast() に渡されるワイルドカード以外の識別子すべてと一致するクライアントで構成されます。識別子の指定にワイルド・カードを使用できます。

LMID、USRNAME および CLTNAME は、すべて *TPBCTDEF-REC* に入り、ターゲット・クライアント・セットの選択に使用する論理識別子です。論理識別子の SPACES 値は、その引数のワイルドカードとなります。ワイルドカード引数は、そのフィールドの全クライアント識別子と一致します。各識別子は、システムが有効とみなすよう定義されたサイズの制約事項を満たさなければなりません。つまり、各識別子の長さは 0 から 30 文字までででなければなりません。

要求のデータ部は、*DATA-REC* によって識別され、*TPTYPE-REC* 内の LEN は、送信する *DATA-REC* の大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES の場合、*DATA-REC* および LEN は無視され、データ部なしで要求が送られます。

次に、*TPBCTDEF-REC* の有効な設定の一覧を示します。

TPNOBLOCK
: この要求は、ブロッキング条件が存在する場合 (たとえば、メッセージの送信先である内部バッファがいっぱいの場合など) には、送信されません。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: ブロッキング条件が存在する場合には要求は送られません (たとえば、メッセージの送信先である内部バッファがいっぱいのとき)。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。tpbroadcast() が正常終了した場合には、メッセージはシステムに渡され、選択されたクライアントに転送されます。tpbroadcast() は、選択された各クライアントにメッセージが送られるのを待機しません。TPNOSIGRSTRT または TPSIGRSTRT のいずれかを設定しなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

戻り値

TPBROADCAST() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPBROADCAST() はブロードキャスト・メッセージをアプリケーション・クライアントに送信せず、TP-STATUS に次の値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。不当な LMID を使用すると、TPBROADCAST() は異常終了し、TPEINVAL() が返されます。ただし、存在しないユーザやクライアントの名前の場合は、誰にもブロードキャストされないだけでこのルーチンは正常に終了します。

[TPETIME]
ブロッキング・タイムアウトが生じましたが、TPBLOCK も TPTIME も指定されていました。

[TPEBLOCK]
呼び出し時にブロッキング条件が検出されましたが、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPBROADCAST() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

移植性

TPNOTIFY() に記述されているインターフェイスはすべて、ネイティブ・サイトの UNIX システムベースのプロセッサ上で利用できます。さらに、ルーチン TPBROADCAST() と TPCHKUNSOL() およびルーチン TPSETUNSOL() は、UNIX および MS-DOS ワークステーション・プロセッサ上で利用できます。

使用法

シグナル・ベースの通知方法を選択したクライアントは、シグナルに関する制約から、システムがシグナルで通知を制御することはできません。このような状態で通知がなされた場合、システムは、選択されたクライアントに対する通知方法をディップインに切り替えることを示すログ・メッセージを生成し、以後、クライアントにはディップイン方式で通知が行われることになります。(通知方法の詳細については、UBBCONFIG(5) の中の RESOURCES セクションの NOTIFY パラメータの説明を参照してください)。

なお、クライアントのシグナル通知は常にシステムによって行われるので、元の通知呼び出しがどこで行われるかにかかわらず、通知の形態は一貫しています。したがって、シグナル・ベースの通知を使用するには、以下のことが必要です。

- ネイティブ・クライアントがアプリケーション管理者として実行している。
- ワークステーション・クライアントはアプリケーション管理者として実行している必要はない。

アプリケーション管理者の ID は、そのアプリケーションのコンフィギュレーション・ファイルで識別されます。

あるクライアントに対してシグナル・ベースの通知方法を選択すると、いくつかの ATMI 呼び出しは異常終了します。TPSIGRSTRT の指定がなければ、任意通知型メッセージを受け取るため、TPGOTSIG( ) を返します。通知方法の選択については、UBBCONFIG(5) および TPINITIALIZE(3cbl) を参照してください。

関連項目　TPINITIALIZE(3cbl)、TPNOTIFY(3cbl)、TPTERM(3cbl)、UBBCONFIG(5)

TPCALL(3cbl)

名前　TPCALL() - サービスへのメッセージの同期送信を行うルーチン

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 ITPTYPE-REC.
 COPY TPTYPE.

01 IDATA-REC.
 COPY User data.

01 OTPTYPE-REC.
 COPY TPTYPE.

01 ODATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPCALL" USING TPSVCDEF-REC ITPTYPE-REC IDATA-REC OTPTYPE-REC
ODATA-REC TPSTATUS-REC.
```

機能説明　TPCALL() は要求を送り、それと同期してその応答を待ちます。このルーチンへの呼び出しは、TPACALL() を呼び出した後即座に TPGETRPLY() を呼び出すのと同じことです。TPCALL() は、*TPSVCDEF-REC* 内の SERVICE-NAME で指定する要求 / 応答型サービスに要求を送ります。この要求は、以前になされた TPSPRIO() の呼び出しで変更されていないかぎり、SERVICE-NAME に定義されている優先順位で送信されます。要求のデータ部分は、*IDATA-REC* によって指定され、*ITPTYPE-REC* 内の LEN は、送信する *IDATA-REC* の大きさを指定します。なお、*IDATA-REC* は、長さの指定を必要としないタイプのレコードを指している場合、*ITPTYPE-REC* 内の LEN は無視されます (0 でかまいません)。*ITPTYPE-REC* 内の REC-TYPE が SPACES の場合には、*IDATA-REC* および *ITPTYPE-REC* 内の LEN は無視され、データ部なしで要求が送信されます。*ITPTYPE-REC* 内の REC-TYPE が STRING のときに *ITPTYPE-REC* 内の LEN が 0 の場合にも、データ部なしで要求が送信されます。*ITPTYPE-REC* 内の REC-TYPE および *ITPTYPE-REC* 内の SUB-TYPE は、SERVICE-NAME が認識する REC-TYPE および SUB-TYPE のいずれかと一致しなければなりません。

ODATA-REC は応答が読み込まれる場所を指定し、入力時の *OTYPE-REC* 内の LEN は *ODATA-REC* に移動されるべき最大バイト数を示します。同じレコードを送信と受信の両方に使用する場合には、*ODATA-REC* を *IDATA-REC* に REDEFINED してください。TPCALL() の正常終了時には、*OTPTYPE-REC* 内の LEN には、*ODATA-REC* に実際に移動されたバイト数が入ります。*OTPTYPE-REC* 内の REC-TYPE および *OTPTYPE-REC* 内の SUB-TYPE には、応答のタイプおよびサブタイプがそれぞれ入っています。応答が *ODATA-REC* より大きい場合は、*ODATA-REC* にはこのレコードに入るバイト数分のみが入ります。応答の残りは破棄され、TPCALL() は TPTRUNCATE() を設定します。

OTPTYPE-REC 内の LEN が正常終了時に 0 であると、応答にはデータ部がなく、*ODATA-REC* は変更されていません。*OTPTYPE-REC* 内の LEN が入力時に 0 であると、エラーになります。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、この設定を使用していると、SERVICE-NAME が呼び出されても、呼び出し元のトランザクションの一部として実行されません。トランザクションをサポートしないサーバに SERVICE-NAME が属しており、呼び出し元がトランザクション・モードにある場合は、この設定を指定しなければなりません。この値を真に設定する、トランザクション・モードの呼び出し元は、やはりトランザクション・タイムアウトの影響を受けます (それ以外はなし)。この設定を使用した状態で呼び出されたサービスが正常に実行できない場合、呼び出し元のトランザクションは影響を受けません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、SERVICE-NAME が呼び出されたときに、このプログラムは呼び出し元のトランザクションのために実行されます。呼び出し元がトランザクション・モードにない場合は、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPNOCHANGE
: この設定を使用すると、*ODATA-REC* のタイプは変更されません。つまり、応答されたレコードのタイプとサブタイプは、受け取り側が着信レコードのタイプを識別するかぎりは、それぞれ *OTPTYPE-REC* 内の REC-TYPE および *OTPTYPE-REC* 内の SUB-TYPE と一致しなければなりません。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPCHANGE
: 応答レコードのタイプとサブタイプの両方または片方は、受け取り側が着信レコードのタイプを識別するかぎりは、それぞれ *OTPTYPE-REC* 内の REC-TYPE および *OTPTYPE-REC* 内の SUB-TYPE と異なっていることが可能です。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPNOBLOCK
: この要求は、ブロッキング条件が存在する場合 (たとえば、メッセージの送信先である内部バッファがいっぱいの場合など) には、送信されません。ただし、この設定は TPCALL( ) の送信部分にしか適用されません。このルーチンは応答を待ってブロックすることがあります。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、ルーチンは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

戻り値

TPCALL( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。TP-STATUS に TPOK または TPESVCFAIL が設定されると、*TPSTATUS-REC* 内の APPL-RETURN-CODE に TPRETURN( ) の一部として送信されたアプリケーション定義の値が入ります。

着信メッセージの大きさが入力時に LEN に指定されたものより大きい場合は、TPTRUNCATE() が設定され、LEN 分のデータのみが *ODATA-REC* に移動されて残りのデータは破棄されます。

エラー

次の条件が発生すると、TPCALL() は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました (たとえば、SERVICE-NAME が SPACES である場合、または *TPSVCDEF-REC* の設定が無効である場合など)。

[TPENOENT]
: 存在しないか、要求 / 応答型サービスでないため (つまり、会話サービス)、SERVICE-NAME に送信できません。

[TPEITYPE]
: REC-TYPE および SUB-TYPE が、SERVICE-NAME が受け付けるタイプおよびサブタイプではありません。

[TPEOTYPE]
: 応答のタイプまたはサブタイプのいずれかは、呼び出し元が認識しているものでありません。あるいは、TPNOCHANGE が設定されていて、*ODATA-REC* の REC-TYPE と SUB-TYPE が、そのサービスから送られた応答のタイプおよびサブタイプと一致しません。*ODATA-REC* および *OTPTYPE-REC* 内の LEN はいずれも変更されません。サービス要求が呼び出し元の現在のトランザクションの一部として行われると、応答が破棄されるので、そのトランザクションに中途終了マークが付けられます。

[TPETRAN]
: SERVICE-NAME は、トランザクションをサポートしていないサーバに属していて、TPTRAN が設定されていました。

[TPETIME]
: タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションは「アボートのみ」とマークされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。いずれの場合でも、*ODATA-REC* も *OTPTYPE-REC* も変更されません。トランザクション・タイムアウトが発生した場合、１つの例外を除き、トランザクションが中途終了するまでは、新しい要求の送信や未終了の応答の受信を行おうとしてもできず、TPETIME が返されます。

[TPESVCFAIL]
: 呼び出し元の応答を送るサービス・ルーチンが、TPFAIL() を設定して TPRETURN() を呼び出しました。これは、アプリケーション・レベルの障害です。サービスの応答の内容 (送信された場合) は、*ODATA-REC* に入ります。呼び出し元のトランザクションの一部としてサービス要求が出された場合、トランザクションは中途終了マークが付けられます。トランザクションがタイムアウトになったかどうかにかかわらず、トランザクションがアボートされる前に有効な通信は、TPNOREPLY、TPNOTRAN、および TPNOBLOCK を設定した TPACALL() の呼び出しだけです。

[TPESVCERR]
: サービス・ルーチンの呼び出し中、またはそのルーチンが TPRETURN() で完了する際に、エラーを検出しました (たとえば、間違った引数が渡された場合など)。このエラーが発生すると、応答データは返されません。つまり、*ODATA-REC* も *OTPTYPE-REC* も変更されません。呼び出し元のトランザクションのためにサービス要求が出された場合 (つまり、TPNOTRAN が設定されていない場合)、このトランザクションは中途終了マークが付けられます。トランザクションがタイムアウトになったかどうかにかかわらず、トランザクションがアボートされる前に有効な通信は、TPNOREPLY、TPNOTRAN、および TPNOBLOCK を設定した TPACALL() の呼び出しだけです。

[TPEBLOCK]
: TPCALL() の送信部分でブロッキング状態が検出されましたが、TPNOBLOCK が指定されていました。

[TPGOTSIG]
: シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
: TPCALL() の呼び出し方法が不適切です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

関連項目　TPACALL(3cbl)、TPFORWAR(3cbl)、TPGPRIO(3cbl)、TPRETURN(3cbl)、TPSPRIO(3cbl)

TPCANCEL(3cbl)

名前　　TPCANCEL() - 未処理の応答の通信ハンドルを取り消す

形式

```
01 TPSVCDEF-REC.
  COPY TPSVCDEF.

01 TPSTATUS-REC.
  COPY TPSTATUS.

CALL "TPCANCEL" USING TPSVCDEF-REC TPSTATUS-REC.
```

機能説明　　TPCANCEL() は、TPACALL() が返す通信ハンドル、*TPSVCDEF-REC* 内の COMM-HANDLE を取り消します。トランザクションに対応する通信ハンドルを取り消そうとすると、エラーになります。

正常終了の場合、COMM-HANDLE は以後無効になり、COMM-HANDLE のために受信する応答はすべて、何の警告もなく捨てられてしまいます。

戻り値　　TPCANCEL() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー　　次の条件が発生すると、TPCANCEL() は異常終了し、TP-STATUS に次の値を設定します。

[TPEBADDESC]
: COMM-HANDLE が無効な通信ハンドルです。

[TPETRAN]
: COMM-HANDLE が呼び出し元のトランザクションに対応しています。COMM-HANDLE はそのまま有効で、呼び出し元の現在のトランザクションは影響を受けません。

[TPEPROTO]
: TPCANCEL() の呼び出し方法が不適切です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

関連項目　　TPACALL(3cbl)

TPCHKAUTH(3cbl)

名前

TPCHKAUTH( ) - BEA Tuxedo ATMI アプリケーションへの結合に認証が必要かどうかをチェックする

形式

```
01 TPAUTDEF-REC.
 COPY TPAUTDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

 CALL "TPCHKAUTH" USING TPAUTDEF-REC TPSTATUS-REC.
```

機能説明

TPCHKAUTH( ) は、アプリケーションのコンフィギュレーションが認証を必要としているかどうかを調べます。これは一般に、TPINITIALIZE( ) を呼び出す前にアプリケーション・クライアントが使用して、ユーザからのパスワードの入力を必要とするかどうかを判別します。

戻り値

TPCHKAUTH( ) は正常終了時には、TP-STATUS に [TPOK] を設定し、*TPAUTDEF-REC* に次のいずれかの値を設定します。

TPNOAUTH
: 認証が必要とされないことを示します。

TPSYSAUTH
: システムの認証のみが必要とされることを示します。

TPAPPAUTH
: システムの認証およびアプリケーション固有の認証の両方が必要であることを示します。

エラー

次の条件が発生すると、TPCHKAUTH( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

移植性

TPCHKAUTH( ) に記述されているインターフェイスは、UNIX システムおよび MS-DOS オペレーティング・システム上でサポートされています。

関連項目

TPINITIALIZE(3cbl)

TPCHKUNSOL(3cbl)

名前　　TPCHKUNSOL() - 任意通知型メッセージをチェックする

形式

```
01 MSG-NUM PIC S9(9) COMP-5.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPCHKUNSOL" USING MSG-NUM TPSTATUS-REC.
```

機能説明　　tpchkunsol() は、クライアントが任意通知型メッセージの検査を行うときに使用します。クライアントでシグナル・ベースの通知方法を使用してこのルーチンを呼び出しても、何も行われず、ただちに終了します。このルーチンを呼び出すと、アプリケーションで定義された任意通知型メッセージ処理ルーチンへの呼び出しが BEA Tuxedo ATMI のライブラリによって行われることがあります。

戻り値　　TPCHKUNSOL() は正常終了時には、TP-STATUS に [TPOK] を設定し、ディスパッチされた任意通知型メッセージの数を *MSG-NUM* に返します。

エラー　　次の条件が発生すると、TPCHKUNSOL() は異常終了し、TP-STATUS に次の値を設定します。

[TPEPROTO]
: tpchkunsol() が不正なコンテキストで呼び出されました (たとえば、サーバ内から)。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

移植性　　TPNOTIFY() に記述されているインターフェイスはすべて、ネイティブ・サイトの UNIX システムベースのプロセッサ上で利用できます。さらに、ルーチン TPBROADCAST() と TPCHKUNSOL() およびルーチン TPSETUNSOL() は、UNIX および MS-DOS ワークステーション・プロセッサ上で利用できます。

シグナル・ベースの通知方法を選択したクライアントは、シグナルに関する制約から、システムがシグナルで通知を制御することはできません。このような状態で通知がなされた場合、システムは、選択されたクライアントに対する通知方法をディップインに切り替えることを示すログ・メッセージを生成し、以後、クライアントにはディップイン方式で通知が行われることになります。(通知方法の詳細については、UBBCONFIG(5) の中の RESOURCES セクションの NOTIFY パラメータの説明を参照してください)。

なお、クライアントのシグナル通知は常にシステムによって行われるので、元の通知呼び出しがどこで行われるかにかかわらず、通知の形態は一貫しています。したがって、シグナル・ベースの通知を使用するには、以下が必要です。

- ネイティブ・クライアントがアプリケーション管理者として実行している必要があります。
- ワークステーション・クライアントはアプリケーション管理者として実行している必要はありません。

アプリケーション管理者の ID は、そのアプリケーションのコンフィギュレーションの一部として識別されます。

あるクライアントに対してシグナル・ベースの通知方法を選択すると、いくつかの ATMI 呼び出しは異常終了します。TPSIGRSTRT の指定がなければ、任意通知型メッセージを受け取るため、TPGOTSIG( ) を返します。通知方法の選択については、UBBCONFIG(5) および TPINITIALIZE(3cbl) を参照してください。

関連項目　TPBROADCAST(3cbl)、TPINITIALIZE(3cbl)、TPNOTIFY(3cbl)、TPSETUNSOL(3cbl)

TPCLOSE(3cbl)

名前　TPCLOSE( ) - BEA Tuxedo ATMI のリソース・マネージャのクローズ

形式

```
01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPCLOSE" USING TPSTATUS-REC.
```

機能説明　tpclose( ) は、呼び出し元とそれにリンクされたリソース・マネージャとの関係を絶ちます。リソース・マネージャはクローズの内容がそれぞれで異なるため、個々のリソース・マネージャをクローズするために必要な情報をコンフィギュレーション・ファイルに記述します。

リソース・マネージャがすでにクローズしている場合 (すなわち、tpclose( ) が 1 回以上呼び出された)、何も処理は行われず、正常終了を示すコードが返されます。

戻り値　TPCLOSE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー　次の条件が発生すると、TPCLOSE( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPERMERR]
: リソース・マネージャが正しくクローズできませんでした。より詳しい理由については、リソース・マネージャを独自の方法で調べることによって得ることができます。ただし、エラーの正確な性質を判別するための呼び出しを使用すると、移植性が損なわれます。

[TPEPROTO]
: tpclose( ) が不正なコンテキストで呼び出されました (たとえば、呼び出し元がトランザクション・モードにあるとき)。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

関連項目　TPOPEN(3cbl)

TPCOMMIT(3cbl)

名前　　TPCOMMIT( ) - 現在の BEA Tuxedo ATMI のトランザクションのコミット

形式

```
01 TPTRXDEF-REC.
 COPY TPTRXDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPCOMMIT" USING TPTRXDEF-REC TPSTATUS-REC
```

機能説明

tpcommit( ) はトランザクションの終了 (コミット) を示します。tpcommit( ) は 2 フェーズ・コミット・プロトコルを使用して、パーティシパント間の調整をとります。tpcommit( ) は、トランザクションのイニシエータからのみ呼び出されます。いずれかのパーティシパントがトランザクションをコミットできない場合 (たとえば、それらが TPFAIL( ) を設定して TPRETURN( ) を呼び出す場合など)、そのトランザクション全体がアボートされ、TPCOMMIT( ) は異常終了します。つまり、そのトランザクションに関連して各プロセスが行ったすべての作業は取り消されます。すべてのパーティシパントがトランザクションの中のそれぞれが担当する部分のコミットを決定した場合、この決定は安定記憶装置に記録された後、すべてのパーティシパントに対して作業のコミットが要求されます。

TP-COMMIT-CONTROL 特性の設定条件に従って (TPSCMT( ) 参照)、コミットの決定が記録された後、あるいは 2 フェーズ・コミット・プロトコルが完了した後、TPCOMMIT( ) は正常に終了することができます。コミットの決定が記録された後、第 2 フェーズが完了する前 (TP-CMT-LOGGED) に TPCOMMIT( ) が終了する場合、すべてのパーティシパントはトランザクションのために行った作業内容をコミットすることに同意していると見なし、第 2 フェーズでトランザクションをコミットする約束を果たす必要があります。ただし、TPCOMMIT( ) は第 2 フェーズが完了する前に終了してしまうので、パーティシパントの中には、このルーチンが正常終了した場合でも、トランザクションの担当部分をヒューリスティックに (コミットの決定とは矛盾するような方法で) 完了するといった状況が発生してしまいます。

TP-COMMIT-CONTROL 特性が 2 フェーズ・コミット・プロトコルの完了 (TP-CMT-COMPLETE) 後に TPCOMMIT( ) が終了するように設定されている場合、その戻り値にはトランザクションの正確な状態が反映されます (つまり、トランザクションがヒューリスティックに完了したかどうか)。

なお、トランザクションに1つのリソース・マネージャしか関与していない場合には、1フェーズ・コミットが行われます(つまり、リソース・マネージャには、コミットできるかどうかの確認はされず、単にコミットの指示が出されます)。そして、この場合、TP-COMMIT-CONTROL 特性はコミットには関係せず、TPCOMMIT() はヒューリスティックな結果(もしあれば)を返します。

未処理の応答の通信ハンドルがあるときに TPCOMMIT() が呼び出されると、TPCOMMIT() の終了時にトランザクションはアボートし、呼び出し元のトランザクションに対応する通信ハンドルがすべて無効になります。呼び出し元のトランザクションに対応していない通信ハンドルは有効なままです。

呼び出し元のトランザクションに対応する接続がすべてクローズしてから、TPCOMMIT() を呼び出さなければなりません。そうでないと、[TPEABORT] が返され、トランザクションはアボートし、TPEV-DISCONIMM イベントを示して各接続が不規則に切断されます。TPBEGIN() の前または TPNOTRAN を設定して(つまり、トランザクション・モードでない状態で)オープンされた接続は、TPCOMMIT() または TPABORT() の影響を受けません。

現時点では、TPCOMMIT() の引数 *TPTRXDEF-REC* は将来使用するために予約されています。

戻り値

TPCOMMIT() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPCOMMT() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: *TPTRXDEF-REC* が 0 でありません。呼び出し元のトランザクションは影響を受けません。

[TPETIME]
: トランザクションがタイムアウトし、トランザクションの状態が不明です(つまり、そのトランザクションはコミットされているかもしれないし、中途終了しているかもしれません)。ただし、トランザクションがタイムアウトし、その状態が中途終了であることが分かっている場合には、[TPEABORT] が返されます。

[TPEABORT]
: トランザクションの実行元あるいはそのパーティシパントが行った作業をコミットできなかったために、そのトランザクションをコミットできませんでした。また、このエラーは、tpcommit() が未終了の応答が残っているか、会話型接続をオープンしたまま呼び出された場合にも返されます。

[TPEHEURISTIC]
ヒューリスティックな判断のため、トランザクションの一部としてなされた作業が一部はコミットされ、一部は中途終了しています。

[TPEHAZARD]
ある種の障害のため、トランザクションの一部としてなされた作業がヒューリスティックに完了している可能性があります。

[TPEPROTO]
tpcommit() が不正なコンテキストで呼び出されました (たとえば、パーティシパントにより呼び出されるなど)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

注意事項

TPBEGIN()、TPCOMMIT() および TPABORT() を使用して BEA Tuxedo ATMI のトランザクションを記述する際には、XA インターフェイスに準拠した (および呼び出し元に適切にリンクされている) リソース・マネージャが行う作業がトランザクションとしての特性を備えていることを忘れないようにすることが重要です。トランザクションで行われるその他の処理内容は、TPCOMMIT() や TPABORT() の影響を受けません。そのリソース・マネージャが行った処理が BEA Tuxedo ATMI のトランザクションの一部となるよう、XA インターフェイスを満たすリソース・マネージャをサーバにリンクする方法については、buildserver(1) を参照してください。

関連項目

TPABORT(3cbl)、TPBEGIN(3cbl)、TPCONNECT(3cbl)、TPGETLEV(3cbl)、TPRETURN(3cbl)、TPSCMT(3cbl)

TPCONNECT(3cbl)

名前　　TPCONNECT()- 会話接続の確立

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPCONNECT" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明

TPCONNECT() により、プログラムは会話サービス、*TPSVDEF-REC* 内の SERVICE-NAME との半二重接続をセットアップできます。この名前は、会話サーバがポストした会話サービス名の１つでなければなりません。

呼び出し元は、接続セットアップ処理の一部として、アプリケーション定義データを受信サービス・ルーチンに渡すことができます。呼び出し元がデータを渡すことを選択した場合には、*DATA-REC* にはデータが入り、*TPTYPE-REC* 内の LEN は送信するレコードの大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES である場合、*DATA-REC* および LEN は無視されます (アプリケーション・データは会話サービスに渡されません)。REC-TYPE および *TPTYPE-REC* 内の SUB-TYPE は、SERVICE-NAME が認識するタイプおよびサブタイプと一致しなければなりません。

TPSVCSTART() の正常終了時には会話サービスは *DATA-REC* および LEN を受け取るので、TPCONNECT() が送信したデータを獲得するために、会話サービスが TPRECV() を呼び出すことはありません。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、この設定を使用していると、SERVICE-NAME が呼び出されても、呼び出し元のトランザクションの一部として実行されません。トランザクションをサポートしないサーバに SERVICE-NAME が属しており、呼び出し元がトランザクション・モードにある場合は、この設定を指定しなければなりません。このフラグ設定を使用するトランザクション・モードの呼び出し元は、依然としてトランザクション・タイムアウトの対象となります (それ以外はなし)。この設定を使用した状態で呼び出されたサービスが正常に実行できない場合、呼び出し元のトランザクションは影響を受けません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、SERVICE-NAME が呼び出されたときに、このプログラムは呼び出し元のトランザクションのために実行されます。呼び出し元がトランザクション・モードにない場合、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPSENDONLY
: 呼び出し元は、最初にそれがデータの送信のみを行え、呼び出されたサービスはデータの受信のみを行えるよう接続を設定します (つまり、呼び出し元が当初の接続の制御権を有するように)。TPSENDONLY または TPRECVONLY のいずれかを指定しなければなりません。

TPRECVONLY
: 呼び出し元は、それがデータの受信のみを行え、呼び出されたサービスがデータの送信のみを行えるように接続を設定します (つまり、呼び出されるサービスが当初、接続の制御権を有するように)。TPSENDONLY または TPRECVONLY のいずれかを指定しなければなりません。

TPNOBLOCK
: ブロッキング条件が存在する場合、接続が設定されず、データが送信されません (たとえば、メッセージが送信されるときに使用されるデータ・バッファがいっぱいである場合)。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続きプログラムに対して有効です。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: シグナルが関数内部のシステム・コールを中断させると、呼び出しが再度出されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: TPNOSIGRSTRT を指定しているときにシグナルが関数内部のシステム・コールを中断させると、その呼び出しは異常終了し、TP-STATUS に TPGOTSIG() が設定されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

戻り値

TPCONNECT() は正常終了時には、TP-STATUS に [TPOK] を設定し、*TPSVDEF-REC* 内の COMM-HANDLE に通信ハンドルを返します。このハンドルは、以降の呼び出しにおいて接続を参照するために使用されます。

エラー

次の条件が発生すると、TPCONNECT() は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました (*TPSVCDEF-REC* の設定が無効など)。

[TPENOENT]
: SERVICE-NAME が存在しないか、会話サーバでないため、SERVICE-NAME への接続を行えません。

[TPEITYPE]
: REC-TYPE および SUB-TYPE が、SERVICE-NAME が受け付けるタイプおよびサブタイプではありません。

[TPELIMIT]
: 未終了の接続の最大数に達したため、接続が送られませんでした。

[TPETRAN]
: SERVICE-NAME はトランザクションをサポートしないプログラムに属していますが、TPNOTRAN が設定されていませんでした。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションは「アボートのみ」とマークされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、接続を使ったメッセージの送受信や新しい接続の開始はできません。これらの操作を行おうとすると、[TPETIME] が発生して失敗します。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPCONNECT() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　TPDISCON(3cbl)、TPRECV(3cbl)、TPSEND(3cbl)

TPDEQUEUE(3cbl)

名前　　TPDEQUEUE( ) - キューからメッセージを取り出すルーチン

形式

```
01 TPQUEDEF-REC.
 COPY TPQUEDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY STATDEF.

CALL "TPDEQUEUE" USING TPQUEDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　TPDEQUEUE( ) は、キュー・スペース QSPACE-NAME 内の QNAME で指定されるキューから、処理するメッセージを取り出します。

省略時設定では、キューの先頭のメッセージが取り出されます。キュー上のメッセージの順序は、そのキューの作成時に定義されます。アプリケーションは、MSGID を使用してメッセージ識別子を指定するか、または CORRID を使用して相関識別子を指定することにより、特定のメッセージをキューから取り出すことを要求できます。メッセージを現在取り出せない場合には、アプリケーションはメッセージを待つことを示すために、*TPQUEDEF-REC* 設定を使用することもできます。また、*TPQUEDEF-REC* 構造体を使用して、メッセージを読み出してもそれがキューから削除されないようにしたり、キュー上の相対位置が変更されないようにすることが可能です。この後のこのレコードについての説明を参照してください。.

DATA-REC は、キューから取り出されたメッセージの読み込み先を指定します。入力時の LEN は、*DATA-REC* へ移動する最大バイト数を示します。正常終了時には、LEN には、*DATA-REC* へ移動された実際のバイト数が入ります。REC-TYPE および SUB-TYPE には、それぞれ応答のタイプおよびサブタイプが入ります。応答が *DATA-REC* より大きい場合は、*DATA-REC* にはこのレコードに入るバイト数分のみが入ります。応答の残りは破棄され、TPDEQUEUE( ) は異常終了して [TPTRUNCATE] を返します。

正常終了時に LEN が 0 である場合は、応答にはデータ部がなく、*DATA-REC* は変更されていません。入力時に LEN を 0 にすると、エラーになります。

呼び出し元がトランザクション・モードにあり、TPTRAN が設定されている場合は、メッセージは、トランザクション・モードでキューから取り出されます。この結果、*TPDEQUEUE* が正常終了して呼び出し元のトランザクションが正常にコミットされると、メッセージはキューから削除されます。呼び出し元のトランザクションが、明示的に、またはトランザクション・タイムアウトあるいはなんらかの通信エラーの結果としてロールバックされると、メッセージはキュー上に残されます。つまり、キューからのメッセージの削除もロールバックされます。同じトランザクション内で、同じメッセージの登録と取り出しを行うことはできません。

呼び出し元がトランザクション・モードにないか、または TPNOTRAN が設定されている場合は、メッセージはトランザクション・モードではキューから取り出されません。トランザクション・モードでない場合に通信エラーまたはタイムアウトが発生すると、アプリケーションには、メッセージが正しくキューから取り出されたかどうかがわからず、メッセージが失われることがあります。

次に、*TPQUEDEF-REC* の有効な設定の一覧を示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、メッセージは呼び出し元と同じトランザクション内ではキューから取り出されません。この値を真に設定する、トランザクション・モードの呼び出し元は、やはりトランザクション・タイムアウトの影響を受けます (それ以外はなし)。この設定を使用した状態で呼び出されたキューからのメッセージの取り出しに失敗した場合、呼び出し元のトランザクションは影響されません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、メッセージは呼び出し元と同じトランザクション内でキューから取り出されます。呼び出し元がトランザクション・モードにない場合は、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPNOBLOCK

ブロッキング状態が存在すると、メッセージはキューから取り出されません。TPNOBLOCK を設定した場合、メッセージの転送先である内部バッファがいっぱいであるなどのブロッキング状態が存在すると、呼び出しは異常終了し、TP-STATUS に TPEBLOCK が設定されます。TPNOBLOCK を設定し、ターゲット・キューが別のアプリケーションによって排他的にオープンされているというブロッキング状態が存在する場合、呼び出しは異常終了し、TP-STATUS に TPEDIAGNOSTIC が設定され、TPQUEDEF レコードの DIAGNOSTIC フィールドに QMESHARE が設定されます。後者の場合、BEA Tuxedo システム以外の BEA 製品ベースのアプリケーションが、キューイング・サービス API (QSAPI) を使用して読み取りおよび書き込みを排他的に行うため、キューをオープンしています。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK

TPBLOCK が設定されている場合にブロッキング状態が存在すると、その状態が解消されるかタイムアウト (トランザクション・タイムアウトまたはブロッキング・タイムアウト) が発生するまで、呼び出し元はブロックされます。TPQWAIT の設定が指定された場合は、このブロッキング条件には、キュー自体の上でのブロッキングは含まれません。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME

この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME

このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPNOCHANGE

この設定を使用すると、*DATA-REC* のタイプは変更できません。つまり、受信側が着信レコードのタイプを識別できる場合、応答レコードのタイプおよびサブタイプは、それぞれ *TPTYPE-REC* 内の REC-TYPE IN および *TPTYPE-REC* 内の SUB-TYPE IN と一致しなければなりません。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPCHANGE
: 受信側が着信レコードのタイプを識別するかぎり、キューから取り出されたメッセージのタイプとサブタイプの両方または片方は、それぞれ *TPTYPE-REC* 内の REC-TYPE および *TPTYPE-REC* 内の SUB-TYPE に指定されたタイプまたはサブタイプと異なっていることが可能です。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPSIGRSTRT
: シグナルが関数内部のシステム・コールを中断した場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、ルーチンは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPDEQUEUE() が正常終了すると、アプリケーションは *TPQUEDEF-REC* を使用してメッセージに関する追加情報を取得できます。この情報には、キューから取り出されたメッセージのメッセージ識別子、すべての応答または異常終了メッセージに付随し発信元がメッセージと元の要求を結び付けることができるようにする相関識別子、メッセージに対する配信サービスの品質、メッセージの応答に対する配信サービスの品質、応答が要求された場合は応答キューの名前、およびメッセージをキューから取り出すときの失敗に関する情報をアプリケーションが登録できる異常終了キューの名前が含まれます。これらについて以下に説明します。

制御構造体

TPQUEDEF-REC は、アプリケーション・プログラムが、キューからのメッセージの取り出しに関連する情報を渡したり、取得したりするために使用します。*TPQUEDEF-REC* の設定は、構造体の他のどの要素が有効であるかを示すために使用されます。

TPDEQUEUE() への入力時には、次の要素を *TPQUEDEF-REC* に設定できます。

```
05 MSGID    PIC X(32).
05 CORRID   PIC X(32).
```

TPDEQUEUE() の入力情報を制御する *TPQUEDEF-REC* の有効な設定の一覧を次に示します。

TPQGETNEXT
: この値を設定すると、デフォルトのキューの順序を使用して、キュー上の次のメッセージが取り出されることを要求します。TPQGETNEXT、TPQGETBYMSGID、または TPQGETBYCORRID のいずれかを設定しなければなりません。

TPQGETBYMSGID

この値を設定すると、MSGID で識別されるメッセージがキューから取り出されることを要求します。メッセージ識別子は、事前の TPENQUEUE( ) 呼び出しによって返されます。メッセージがあるキューから別のキューに移動した場合、メッセージ識別子は正しい値を示さないので注意が必要です。また、メッセージ識別子の値は 32 バイト全体が意味を持つので、MSGID に指定される値は、たとえば空白類を埋め込むなどして完全に初期化される必要があります。

TPQGETNEXT、TPQGETBYMSGID、または TPQGETBYCORRID のいずれかを設定しなければなりません。

TPQGETBYCORRID

この値を設定すると、CORRID で識別されるメッセージが取り出されることを要求します。相関識別子は、アプリケーションが TPENQUEUE( ) によってメッセージをキューに登録するときに指定されます。相関識別子の値は 32 バイト全体が意味を持つので、CORRID で識別される値は、たとえば空白類を埋め込むなどして完全に初期化される必要があります。

TPQGETNEXT、TPQGETBYMSGID、または TPQGETBYCORRID のいずれかを設定しなければなりません。

TPQWAIT

この値を設定すると、キューが空の場合にはエラーが返されません。代わりに、メッセージを取り出せるようになるまで、プロセスは待つ必要があります。メッセージが取り出せるまで待機しない場合は、TPQNOWAIT を設定します。TPQWAIT が TPQGETBYMSGID または TPQGETBYCORRID とともに設定されると、指定されたメッセージ識別子または相関識別子を持つメッセージがキューに存在しない場合、エラーは返されません。代わりに、基準を満たすメッセージを取り出せるようになるまで、プロセスは待つ必要があります。プロセスは呼び出し元のトランザクション・タイムアウトの影響を受けます。また、トランザクション・モードでない場合、プロセスは TMQUEUE プロセスで -t オプションによって指定されたタイムアウトの影響を受けます。

基準に一致するメッセージがすぐには取り出せず、設定されているアクション・リソースの限界に達すると、TPDEQUEUE は異常終了し、TP-STATUS に TPEDIAGNOSTIC が設定され、DIAGNOSTIC に QMESYSTEM が設定されます。

条件を満たすメッセージがすぐに取り出せない場合は、TPQWAIT 制御パラメータを指定するそれぞれの TPDEQUEUE( ) 要求で、キュー・マネージャ (TMQUEUE) のアクション・オブジェクトを利用できることが必要です。アクション・オブジェクトを利用できない場合、TPDEQUEUE( ) 要求は異常終了します。利用できるキュー・マネージャのアクション数は、キュー・スペースの作成時または変更時に指定されます。待機中のキューからの取り出し要求が完了すると、関連するアクション・オブジェクトは別の要求で使用できるようになります。

TPQPEEK

TPQPEEK を設定すると、指定されたメッセージを読み取ってもキューから削除されなくなります。この場合、TPNOTRAN フラグが設定されている必要があります。トランザクション内でキューに登録されたメッセージやキューから取り出されたメッセージは、そのトランザクションが完了するまで読み取ることができません。

あるスレッドで TPQPEEK を使用してメッセージをキューから非破壊的に取り出す場合、非破壊的な取り出し要求をシステムが処理する少しの間、ほかの非ブロッキング状態のメッセージ取り出し操作からそのメッセージが見えないことがあります。たとえば、メッセージ識別子や相関識別子などの特定の選択基準を使用してメッセージをキューから取り出す操作で、現在、非破壊的なキューから取り出し中のメッセージを検索する場合などです。

TPDEQUEUE() からの出力時には、次の要素が *TPQUEDEF-REC* に設定されます。

```
05 PRIORITY                PIC S9(9) COMP-5.
05 MSGID                   PIC X(32).
05 CORRID                  PIC X(32).
05 TPQUEQOS-DELIVERY-FLAG  PIC S9(9) COMP-5.
05 TPQUEQOS-REPLY-FLAG     PIC S9(9) COMP-5.
05 REPLYQUEUE              PIC X(15).
05 FAILUREQUEUE            PIC X(15).
05 DIAGNOSTIC              PIC S9(9) COMP-5.
05 CLIENTID OCCURS 4 TIMES PIC S9(9) COMP-5
05 APPL-RETURN-CODE        PIC S9(9) COMP-5.
05 APPKEY                  PIC S9(9) COMP-5.
```

TPDEQUEUE() からの出力情報を制御する *TPQUEDEF-REC* の有効な設定を次に示します。どの設定でも、TPDEQUEUE() の呼び出し時に真の場合、メッセージがキューに入れられたときに提供された値が、レコード内の対応する要素に格納され、その設定は真のままになります。値を利用できない、つまりメッセージがキューに入れられたときに値が提供されていない場合、または TPDEQUEUE() の呼び出し時に設定が真でない場合、その設定が真でない状態で TPDEQUEUE() は完了します。

TPQPRIORITY

この値を設定し、TPDEQUEUE() の呼び出しが正常終了し、メッセージが明示的な優先順位でキューに登録されていた場合は、その優先順位が PRIORITY に格納されます。優先順位は 1 から 100 の範囲 (1 と 100 を含みます) で、番号が大きいほど優先順位は高くなります (つまり大きい番号のメッセージが、番号の小さいメッセージより先にキューから取り出されます)。TPQNOPRIORITY が設定されていると、優先順位は利用できません。

メッセージがキューに入れられたときに明示的な優先順位が指定されていない場合、そのメッセージの優先順位は 50 になります。

TPQMSGID
: この値を設定し、TPDEQUEUE() の呼び出しが正常終了した場合は、メッセージ識別子が MSGID に格納されます。メッセージ識別子の値は 32 バイト全体が意味を持ちます。TPQNOMSGID が設定されていると、メッセージ識別子は利用できません。

TPQCORRID
: この値を設定し、TPDEQUEUE() の呼び出しが正常終了し、メッセージが相関識別子を付けてキューに登録された場合は、相関識別子が CORRID に格納されます。相関識別子の値は 32 バイト全体が意味を持ちます。BEA Tuxedo /Q からのメッセージに対するすべての応答には、元の要求メッセージの相関識別子が付きます。TPQNOCORRID が設定されていると、相関識別子は利用できません。

TPQDELIVERYQOS
: この値を設定し、TPDEQUEUE() の呼び出しが正常終了し、配信サービスの品質を設定してメッセージをキューに登録した場合、TPQUEQOS-DELIVERY-FLAG で指定されるフラグ TPQQOSDELIVERYDEFAULTPERSIST、TPQQOSDELIVERYPERSISTENT、または TPQQOSDELIVERYNONPERSISTENT が配信サービスの品質を示します。TPQNODELIVERYQOS を設定すると、配信サービスの品質は指定できなくなります。

  メッセージがキューに登録されたときに配信サービスの品質が明示的に指定されていない場合は、ターゲットキューのデフォルトの配信方針によってメッセージの配信サービスの品質が決まります。

TPQREPLYQOS
: この値を設定し、TPDEQUEUE() の呼び出しが正常終了し、応答サービスの品質を設定してメッセージをキューに登録した場合は、TPQUEQOS-REPLY-FLAG で指定されるフラグ TPQQOSREPLYDEFAULTPERSIST、TPQQOSREPLYPERSISTENT、または TPQQOSREPLYNONPERSISTENT が応答サービスの品質を示します。TPQNOREPLYQOS を設定すると、応答サービスの品質は指定できなくなります。

  メッセージがキューに登録されたときに応答サービスの品質が明示的に指定されていない場合は、REPLYQUEUE キューのデフォルトの配信方針によって応答の配信サービスの品質が決まります。デフォルトの配信方針は、メッセージに対する応答がキューに登録されるときに決まります。つまり、元のメッセージがキューに登録されてから応答が登録されるまでの間に、応答キューのデフォルトの配信方針が変更された場合は、最後に応答が登録された時点で有効な方針が使用されます。

TPQREPLYQ
この値を設定し、TPDEQUEUE( ) の呼び出しが正常終了し、メッセージが応答キューを使用してキューに入れられた場合は、その応答キューの名前が REPLYQUEUE に格納されます。メッセージへの応答はすべて、要求メッセージと同じキュー・スペース内の、指定された応答キューに入る必要があります。TPQNOREPLYQ が設定されていると、応答キューは利用できません。

TPQFAILUREQ
この値を設定し、TPDEQUEUE( ) の呼び出しが正常終了し、メッセージが異常終了キューを使用してキューに入れられた場合は、その異常終了キューの名前が FAILUREQUEUE に格納されます。すべての異常終了メッセージは、要求メッセージと同じキュー・スペース内の、指定された異常終了キューに入る必要があります。TPQNOFAILUREQ が設定されていると、異常終了キューは利用できません。

TPQUEDEF-REC の残りの設定は、TPDEQUEUE( ) が呼び出されると、次の値に設定されます。TPQNOTOP、TPQNOBEFOREMSGID、TPQNOTIME_ABS、TPQNOTIME_REL、TPQNOEXPTIME_ABS、TPQNOEXPTIME_REL、および TPQNOEXPTIME_NONE。

TPDEQUEUE( ) の呼び出しが異常終了し、TP-STATUS に TPEDIAGNOSTIC が設定された場合は、異常終了の原因を示す値が DIAGNOSTIC に返されます。返される可能性のある値は、この後の「診断」の項で定義しています。

また、TPDEQUEUE( ) 呼び出しが正常終了した場合の出力時には、APPKEY にアプリケーション認証キーが設定され、CLIENTID に要求の発信元であるクライアントの識別子が設定され、APPL-RETURN-CODE にメッセージ登録時に設定されたユーザ戻りコードが設定されます。

戻り値
TPDEQUEUE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー
次の条件が発生すると、TPDEQUEUE( ) は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
無効な引数が指定されました (たとえば、QSPACE-NAME が SPACES である場合、*TPQUEDEF-REC* の設定が無効である場合など)。

[TPENOENT]
QSPACE-NAME を利用できないので、これにアクセスできません (関連する TMQUEUE(5) サーバは利用できません)。または、グローバル・トランザクション・テーブル (GTT) にエントリがないので、グローバル・トランザクションを開始できません。

[TPEOTYPE]
キューから取り出されたメッセージの REC-TYPE および SUB-TYPE のいずれかは、呼び出し元が認識しているものではありません。あるいは、TPNOCHANGE が設定されていて、REC-TYPE と SUB-TYPE が、キューから取り出されたメッセージの REC-TYPE および SUB-TYPE と一致しません。*DATA-REC* も *TPTYPE-REC* も変更されません。呼び出しがトランザクション・モードで行われ、このエラーが発生すると、トランザクションはアボートのみになり、メッセージはキューに残ります。

[TPTRUNCATE]
着信メッセージのサイズが、LEN で指定されたサイズより大きいことを示します。LEN で指定されたデータ長分のみが *DATA-REC* へ移動され、残りのデータは破棄されます。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションはアボートのみになります。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。いずれの場合も、*DATA-REC* も *TPTYPE-REC* も変更されません。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、TPDEQUEUE() または TPENQUEUE() を呼び出そうとしても TPETIME が発生して失敗します。

[TPEBLOCK]
ブロッキング条件が存在し、TPBLOCK が設定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT が設定されていました。

[TPEPROTO]
TPDEQUEUE() の呼び出し方法が不適切です。キューまたはトランザクションには影響ありません。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。キューには影響ありません。

[TPEOS]
オペレーティング・システムのエラーが発生しました。キューには影響ありません。

[TPEDIAGNOSTIC]
指定されたキューからのメッセージの取り出しが異常終了しました。異常終了の原因は、*TPQUEDEF-REC* を介して返される診断値によって判別できます。

診断　次の診断値は、キューからのメッセージの取り出し中に返されます。

[QMEINVAL]
無効な設定が指定されました。

[QMEBADRMID]
無効なリソース・マネージャ識別子が指定されました。

[QMENOTOPEN]
このリソース・マネージャは現在オープンしていません。

[QMETRAN]
呼び出しがトランザクション・モードでない、または TPNOTRAN 設定で呼び出しが行われ、キューからメッセージを取り出すトランザクションの開始を試みた際に、エラーが発生しました。この診断は、BEA Tuxedo リリース 7.1 以降のキュー・マネージャでは返されません。

[QMEBADMSGID]
キューからの取り出し用に、無効なメッセージ識別子が指定されました。

[QMESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[QMEOS]
オペレーティング・システムのエラーが発生しました。

[QMEABORTED]
作業が中途終了しました。グローバル・トランザクション内で実行された場合、グローバル・トランザクションが中途終了のみになっています。それ以外の場合、キュー・マネージャは操作を中断しました。

[QMEPROTO]
トランザクション状態がアクティブでないときに、キューからの取り出しが行われました。

[QMEBADQUEUE]
無効な、または削除された、あるいは予約されているキューの名前が指定されました。

[QMENOMSG]
キューから取り出せるメッセージはありません。なお、メッセージがキュー上に存在し、別のアプリケーション・プロセスが、このメッセージをキューから読み取っていた可能性があることに注意してください。この場合は、その別のプロセスがトランザクションをロールバックしたときにメッセージはキューに戻されます。

[QMEINUSE]
: メッセージ識別子または相関識別子を使用してメッセージをキューから取り出す際に、指定したメッセージは別のトランザクションによって使用されています。それ以外の場合、現在キューにあるメッセージはすべて、ほかのトランザクションによって使用されています。この診断は、BEA Tuxedo リリース 7.1 以降のキュー・マネージャでは返されません。

[QMESHARE]
: 指定されたキューからメッセージを取り出す際に、そのキューが別のアプリケーションによって排他的にオープンされています。BEA Tuxedo システム以外の BEA 製品ベースのアプリケーションが、キューイング・サービス API (QSAPI) を使用して読み取りおよび書き込みを排他的に行うため、キューをオープンしています。

関連項目　qmadmin(1)、TPENQUEUE(3cbl)、TMQUEUE(5)

TPDISCON(3cbl)

名前　　TPDISCON() - 会話接続の切断

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPDISCON" USING TPSVCDEF-REC TPSTATUS-REC.
```

機能説明　　TPDISCON() は、通信ハンドル、*TPSVCDEF-REC* 内の COMM-HANDLE で指定された接続を即座に切断し、接続の他方の側で TPEV-DISCONIMM イベントを生成します。

TPDISCON() は会話のイニシエータからしか呼び出せません。TPDISCON() は、呼び出しに使用された通信ハンドルに対応する会話サービスの中からは呼び出せません。会話サービスは TPRETURN() を使用して、会話の該当部分が完了したことを通知しなければなりません。同様に、会話サービスとのやりとりを行うプログラムが TPDISCON() を発行できる場合でも、正しい結果を保証するためには、そのサービスに TPRETURN() で接続を切断させるようにしてください。接続の起動元がサーバである場合には、TPRETURN() を使用し正常に切断を行うこともできます。接続の起動元がトランザクション・モードにある場合、TPCOMMIT() または TPABORT() を使用して、接続を正常に終了させることができます。

TPDISCON() を使用すると、接続はただちに切断されます (すなわち、正常終了ではなく、中途終了)。したがって、あて先に届いていないデータは失われます。TPDISCON() は、接続の他方の側のプログラムが呼び出し元のトランザクションに参加している場合でも発行されます。この場合、このトランザクションは中途終了します。また、呼び出し元は、TPDISCON() が呼び出されるときにその接続の制御権をもっている必要はありません。

戻り値　　TPDISCON() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー　　次の条件が発生すると、TPDISCON() は異常終了し、TP-STATUS に次の値を設定します。

[TPEBADDESC]
: COMM-HANDLE が無効であるか、会話サービスに使用された通信ハンドルです。

[TPETIME]
: タイムアウトが発生しました。通信ハンドルは無効になります。

[TPEPROTO]
TPDISCON() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。通信ハンドルは無効になります。

[TPEOS]
オペレーティング・システムのエラーが発生しました。通信ハンドルは無効になります。

関連項目 TPABORT(3cbl)、TPCOMMIT(3cbl)、TPCONNECT(3cbl)、TPRECV(3cbl)、TPRETURN(3cbl)、TPSEND(3cbl)

TPENQUEUE(3cbl)

名前　　TPENQUEUE( ) - メッセージをキューに登録するルーチン

形式

```
01 TPQUEDEF-REC.
 COPY TPQUEDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPENQUEUE" USING TPQUEDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明

TPENQUEUE( ) は、キュー・スペース QSPACE-NAME 内の QNAME で指定されるキューにメッセージを格納します。キュー・スペースは、キューを集めたもので、そのうちの 1 つのキューが QNAME でなければなりません。

メッセージが BEA Tuxedo ATMI のサーバを対象としている場合、QNAME は、サーバによって提供されるサービスの名前と一致します。システムが提供するサーバである TMQFORWARD(5) は、メッセージをキューから取り出し、キューと同じ名前のサービスを提供するサーバにそのメッセージを転送するデフォルトの機構となります。発信元が応答を期待していた場合、特に指定されていなければ、転送されたサービス要求の応答は発信元のキューに格納されます。発信元は、後で応答メッセージをキューから取り出します。キューは、任意の 2 つの BEA Tuxedo ATMI のプロセス間 (クライアントやサーバ) における信頼性の高いメッセージ転送機構としても使用できます。この場合、キューの名前はサービス名とは一致せず、メッセージ転送のための名前に関連します。

メッセージのデータ部は、*DATA-REC* によって指定されます。*TPTYPE-REC* 内の LEN は、キューに登録する *DATA-REC* の大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES の場合には、*DATA-REC* および LEN は無視され、データ部なしでメッセージがキューに登録されます。*TPTYPE-REC* 内の REC-TYPE および SUB-TYPE は、QSPACE-NAME が認識する REC-TYPE および SUB-TYPE のいずれかと一致しなければなりません。

メッセージは、QSPACE-NAME 用に定義された優先順位が事前の TPSPRIO( ) の呼び出しによって無効化されていないかぎり、QSPACE-NAME 用の優先順位でキューに登録されます。

呼び出し元がトランザクションにあり、TPTRAN が設定されている場合は、メッセージは、トランザクション・モードでキューに登録されます。この結果、TPENQUEUE( ) が正常終了して呼び出し元のトランザクションが正常にコミットされると、メッセージは、トランザクションの完了後に処理されることが保証されます。呼び出し元のトランザクションが、明示的に、またはトランザクション・タイムアウトあるいはなんらかの通信エラーの結果としてロールバックされると、メッセージはキューから削除されます。つまり、キューへのメッセージの登録もロールバックされます。同じトランザクション内で同じメッセージの登録と取り出しを行うことはできません。

呼び出し元がトランザクション・モードにないか、または TPNOTRAN が設定されている場合は、メッセージはトランザクション・モードではキューに登録されません。TPENQUEUE( ) が正常終了した後は、サブミットされたメッセージがキューに登録されたことが保証されます。トランザクション・モードでない場合に通信エラーまたはタイムアウトが発生すると、アプリケーションには、メッセージが正しくキューに格納されたかどうかがわかりません。

メッセージが処理される順序は、以下に説明するように、アプリケーションによって *TPQUEDEF-REC* を介して制御されます。デフォルトのキューの順序付けは、キューの作成時に設定されます。

TPQUEDEF-REC の有効な設定の一覧を次に示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、メッセージは呼び出し元と同じトランザクション内ではキューに登録されません。この値を真に設定する、トランザクション・モードの呼び出し元は、やはりトランザクション・タイムアウトの影響を受けます (それ以外はなし)。この設定を使用した状態で呼び出されたキューへのメッセージの登録が失敗した場合、呼び出し元のトランザクションは影響されません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、この設定が使用されていると、メッセージは呼び出し元と同じトランザクション内でキューに登録されます。呼び出し元がトランザクション・モードにない場合は、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPNOBLOCK
: ブロッキング状態が存在すると、メッセージはキューに入りません。TPNOBLOCK を設定した場合、メッセージの転送先である内部バッファがいっぱいであるなどのブロッキング状態が存在すると、呼び出しは異常終了し、TP-STATUS に TPEBLOCK が設定されます。TPNOBLOCK を設定し、ターゲット・キューが別のアプリケーションによって排他的にオープンされているというブロッキング状態が存在する場合、呼び出しは異常終了し、TP-STATUS に TPEDIAGNOSTIC が設定され、TPQUEDEF レコードの DIAGNOSTIC フィールドに QMESHARE が設定されます。後者の場合、BEA Tuxedo システム以外の BEA 製品ベースのアプリケーションが、キューイング・サービス API (QSAPI) を使用して読み取りおよび書き込みを排他的に行うため、キューをオープンしています。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK が設定されている場合にブロッキング状態が存在すると、その状態が解消されるかタイムアウト (トランザクション・タイムアウトまたはブロッキング・タイムアウト) が発生するまで、呼び出し元はブロックされます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: シグナルが関数内部のシステム・コールを中断した場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、ルーチンは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

キューへのメッセージ登録に関する追加情報は、*TPQUEDEF-REC* を介して指定できます。この情報には、デフォルトのキューの順序付けを無効にしてキューの先頭または登録済みのメッセージの前にメッセージを入れるための値、キューに登録したメッセージを処理する絶対時間または相対時間、メッセージが期限切れになりキューから削除される絶対時間または相対時間、メッセージに対する配信サービスの品質、メッセージの応答に対する配信サービスの品質、応答または異常終了メッセージと元の要求を結び付けるときに役立つ相関識別子、応答を登録するキューの名前、およびすべての異常終了メッセージを登録するキューの名前が含まれます。

制御パラメータ

TPQUEDEF-REC は、アプリケーション・プログラムが、キューへのメッセージの登録に関連する情報を渡したり、取得したりするために使用します。設定は、レコード内のどの要素が有効であるかを示すために使用されます。

TPENQUEUE() への入力時には、次の要素を *TPQUEDEF-REC* に設定できます。

```
05 DEQ-TIME                 PIC S9(9) COMP-5.
05 PRIORITY                 PIC S9(9) COMP-5.
05 MSGID                    PIC X(32).
05 CORRID                   PIC X(32).
05 TPQUEQOS-DELIVERY-FLAG   PIC S9(9) COMP-5.
05 TPQUEQOS-REPLY-FLAG      PIC S9(9) COMP-5.
05 EXP-TIME                 PIC S9(9) COMP-5.
05 REPLYQUEUE               PIC X(15).
05 FAILUREQUEUE             PIC X(15).
05 APPL-RETURN-CODE         PIC S9(9) COMP-5.
```

次の値は、どの値が *TPQUEDEF-REC* に設定されるかを示します。

TPQTOP

この値を設定すると、キューの順序は無効化され、メッセージはキューの先頭に登録されます。この要求は、キューが順序を無効化できるように構成されているかどうかに応じて、受け入れられない場合があります。省略時設定のキューの順序を使用するには、TPQDEFAULT を設定してください。TPQTOP、TPQBEFOREMSGID、TPQDEFAULT のいずれかを設定していなければなりません。

TPQBEFOREMSGID

この値を設定すると、キューの順序は無効化され、メッセージは MSGID によって識別されるメッセージの前に登録されます。この要求は、キューが順序を無効化できるように構成されているかどうかに応じて、受け入れられない場合があります。省略時設定のキューの順序を使用するには、TPQDEFAULT を設定してください。TPQTOP、TPQBEFOREMSGID、TPQDEFAULT のいずれかを設定していなければなりません。

メッセージ識別子の値は 32 バイト全体が意味を持つので、MSGID で識別される値は、たとえば空白類を埋め込むなどして完全に初期化する必要があります。

TPQTIME-ABS
: この値を設定すると、メッセージは、DEQ-TIME によって指定された時間の後に処理されます。DEQ-TIME は、time(2) または mktime(3C) によって生成される絶対時間値です (協定世界時 (UTC) 1970 年 1 月 1 日 00:00:00 から経過した秒数)。絶対時間値も相対時間値も設定しない場合は、TPQNOTIME を設定してください。TPQTIME-ABS、TPQTIME-REL、TPQNOTIME のいずれかを設定していなければなりません。絶対時間は、キュー・マネージャ・プロセスが常駐するマシンの時計によって決定されます。

TPQTIME-REL
: この値を設定すると、メッセージは、キューへの登録が完了してからの相対時間経過後に処理されます。DEQ-TIME は、キューへの登録が完了した後から、サブミットされたメッセージが処理されるまでの遅延秒数を指定します。絶対時間値も相対時間値も設定しない場合は、TPQNOTIME を設定してください。
TPQTIME-ABS、TPQTIME-REL、TPQNOTIME のいずれかを設定していなければなりません。

TPQPRIORITY
: この値を設定すると、メッセージがキューに登録される際の優先順位が PRIORITY に格納されます。優先順位は、1 以上 100 以下の範囲内でなければなりません。数値は高いほど優先順位も高くなります (高い数値のメッセージが低い数値のメッセージよりも先にキューに登録される)。優先順位によって順序付けられていないキューでは、この値は参考のために提供されます。
TPQNOPRIORITY が設定されている場合、メッセージの優先順位はデフォルトで 50 です。

TPQCORRID
: この値を設定すると、メッセージが TPDEQUEUE( ) によってキューから取り出されるとき、CORRID に指定された相関識別子の値を使用することができます。この識別子は、キューに登録されるすべての応答または異常終了メッセージに付随するので、アプリケーションは応答を特定の要求と結び付けることができます。相関識別子を利用できない場合は、TPQNOCORRID を設定してください。

  相関識別子の値は 32 バイト全体が意味を持つので、CORRID に指定される値は、たとえば空白類を埋め込むなどして完全に初期化する必要があります。

TPQREPLYQ
: この値を設定すると、REPLYQUEUE に指定された応答キューが、キューに入れられるメッセージに関連付けられます。メッセージへの応答はすべて、要求メッセージと同じキュー・スペース内の、指定されたキューに登録されます。応答キューの名前を利用できない場合は、TPQNOREPLYQ を設定してください。

TPQFAILUREQ

これが設定されると、FAILUREQUEUE で指定された異常終了キューは、待機メッセージに関連付けられます。(1) キューに登録されたメッセージが TMQFORWARD( ) によって処理され、(2) TMQFORWARD が -d オプションで起動され、(3) サービスが異常終了してヌル以外の応答を返す場合は、応答とそれに関連付けられた TPSTATUS レコード内の APPL-RETURN-CODE で構成される異常終了メッセージが、元の要求メッセージと同じキュー・スペース内の指定されたキューに入ります。異常終了キューの名前を利用できない場合は、TPQNOFAILUREQ を設定してください。

TPQDELIVERYQOS
TPQREPLYQOS

TPQDELIVERYQOS フラグを設定すると、TPQUEQOS-DELIVERY-FLAG で指定されるフラグを使用して、メッセージの配信サービスの品質を制御できます。相互に排他的な次のフラグ、TPQQOSDELIVERYDEFAULTPERSIST、TPQQOSDELIVERYPERSISTENT、または TPQQOSDELIVERYNONPERSISTENT のいずれかを設定する必要があります。TPQDELIVERYQOS を設定しない場合、TPQNODELIVERYQOS を設定しなければなりません。TPQNODELIVERYQOS を設定すると、ターゲット・キューのデフォルトの配信方針によって、メッセージの配信サービスの品質が決まります。

TPQREPLYQOS フラグを設定すると、TPQUEQOS-REPLY-FLAG で指定されるフラグを使用して、応答メッセージの配信サービスの品質を制御できます。相互に排他的な次のフラグ、TPQQOSREPLYDEFAULTPERSIST、TPQQOSREPLYPERSISTENT、または TPQQOSREPLYNONPERSISTENT のいずれかを設定する必要があります。TPQREPLYQOS フラグは、TMQFORWARD で処理されるメッセージから応答が返されるときに使用されます。サービスを呼び出す際に TMQFORWARD を使用しないアプリケーションでは、応答メカニズムのヒントとして TPQREPLYQOS フラグを使用できます。

TPQREPLYQOS を設定しない場合、TPQNOREPLYQOS を設定しなければなりません。TPQNOREPLYQOS を設定すると、REPLYQUEUE キューのデフォルトの配信方針によって、応答の配信サービスの品質が決まります。デフォルトの配信方針は、メッセージへの応答がキューに登録されるときに決定される点に注意してください。つまり、元のメッセージがキューに登録されてから応答が登録されるまでの間に、応答キューのデフォルトの配信方針が変更された場合、応答が最後に入れられた時点で有効な方針が使用されます。

以下に、有効な TPQUEQOS-DELIVERY-FLAG フラグおよび TPQUEQOS-REPLY-FLAG フラグの一覧を示します。

TPQQOSDELIVERYDEFAULTPERSIST
TPQQOSREPLYDEFAULTPERSIST

これらのフラグは、ターゲット・キューまたは応答キューで指定されているデフォルトの配信方針を使用して、メッセージが配信されるよう指定します。

TPQQOSDELIVERYPERSISTENT
TPQQOSREPLYPERSISTENT

これらのフラグは、メッセージがディスク・ベースの配信方式を使った永続的な方法で配信されるよう指定します。これらのフラグが設定されると、ターゲット・キューまたは応答キューに指定されたデフォルトの配信方針は無効になります。

TPQQOSDELIVERYNONPERSISTENT
TPQQOSREPLYNONPERSISTENT

これらのフラグは、メッセージがメモリ・ベースの配信方式を使った非永続的な方法で配信されるよう指定します。メッセージは、キューから取り出されるまでメモリ内に登録されています。これらのフラグが設定されると、ターゲット・キューまたは応答キューに指定されたデフォルトの配信方針は無効になります。
呼び出し元がトランザクション・モードの場合、一時的メッセージは呼び出し元のトランザクション内でキューに登録されますが、システムがシャットダウンしたりクラッシュした場合、またはキュー・スペースとしての IPC 共用メモリが除去された場合は、一時的メッセージは失われます。

TPQEXPTIME-ABS

この値を設定すると、メッセージに絶対期限切れ時間が設定されます。これは、キューからメッセージが削除される絶対時間です。
絶対期限切れ時間は、キュー・マネージャ・プロセスが常駐するマシンのクロックによって決定されます。

絶対期限切れ時間は、EXP-TIME に格納された値で示されます。EXP-TIME は、time(2) または mktime(3C) によって生成された絶対時間に設定されなければなりません (協定世界時 (UTC) 1970 年 1 月 1 日 00:00:00 から経過した秒数)。

キューへの登録操作の時間より早い絶対時間が指定されると、操作は正常終了しますが、メッセージはしきい値の計算に含まれません。期限切れ時間がメッセージの使用可能時間より早い時間の場合、使用可能時間が期限切れ時間より早い時間になるようにいずれかの時間を変更しない限り、メッセージをキューから取り出すことはできません。また、これらのメッセージをキューからの取り出しで一度も使用したことがなくても、期限切れ時間になるとキューから削除されます。メッセージがトランザクション内にあるときに期限切れになった場合、それによってトランザクションが異常終了することはありません。トランザクション内でキューへの登録中、またはキューからの取り出し中に期限切れになったメッセージは、トランザクションが終了した時点でキューから削除されます。メッセージが期限切れになったという通知はありません。

TPQEXPTIME-ABS、TPQEXPTIME-REL、TPQEXPTIME-NONE、または TPQNOEXPTIME のいずれかを設定する必要があります。

TPQEXPTIME-REL

この値を設定すると、メッセージに相対期限切れ時間が設定されます。これは、キューにメッセージが到達したときから削除されるまでの秒数です。相対期限切れ時間は、EXP-TIME に格納された値で示されます。

期限切れ時間がメッセージの使用可能時間より早い時間の場合、使用可能時間が期限切れ時間より早い時間になるようにいずれかの時間を変更しない限り、メッセージをキューから取り出すことはできません。また、これらのメッセージをキューからの取り出しで一度も使用したことがなくても、期限切れ時間になるとキューから削除されます。メッセージがトランザクション内にあるときに期限切れになった場合、それによってトランザクションが異常終了することはありません。トランザクション内でキューへの登録中、またはキューからの取り出し中に期限切れになったメッセージは、トランザクションが終了した時点でキューから削除されます。メッセージが期限切れになったという通知はありません。

TPQEXPTIME-ABS、TPQEXPTIME-REL、TPQEXPTIME-NONE、または TPQNOEXPTIME のいずれかを設定する必要があります。

TPQEXPTIME-NONE

この値を設定すると、メッセージが期限切れになることはありません。このフラグは、ターゲット・キューに関連するデフォルトの期限切れ方針を無効にします。メッセージを削除するには、管理用のインターフェイスを使用して、キューからメッセージを取り出すか削除します。TPQEXPTIME-ABS、TPQEXPTIME-REL、TPQEXPTIME-NONE、または TPQNOEXPTIME のいずれかを設定する必要があります。

TPQNOEXPTIME
この値を設定すると、ターゲット・キューに関連するデフォルトの期限切れ時間がメッセージに使用されます。TPQEXPTIME-ABS、TPQEXPTIME-REL、TPQEXPTIME-NONE、または TPQNOEXPTIME のいずれかを設定する必要があります。

また、TPSTATUS-REC 内の APPL-RETURN-CODE にユーザ戻り値を設定することができます。この値は、メッセージをキューから取り出すアプリケーションに返されます。

TPENQUEUE() からの出力時には、次の要素が *TPQUEDEF-REC* に設定されます。

```
05 MSGID      PIC X(32).
05 DIAGNOSTIC PIC S9(9) COMP-5.
```

TPENQUEUE() からの出力情報を制御する *TPQUEDEF-REC* の有効な設定を次に示します。TPENQUEUE() の呼び出し時にこの設定が真の場合、BEA Tuxedo /Q サーバ TMQUEUE(5) は、レコード内の対応する要素にメッセージ識別子を格納します。TPENQUEUE() の呼び出し時にこの設定が真でない場合、TMQUEUE() は、レコード内の対応する要素にメッセージ識別子を格納しません。

TPQMSGID
この値が設定され、TPENQUEUE() の呼び出しが正常終了した場合は、メッセージ識別子が MSGID に格納されます。メッセージ識別子の値は 32 バイト全体が意味を持つので、MSGID に格納される値は、たとえばヌル文字を埋め込むなどして完全に初期化する必要があります。実際に初期化に使用される埋め込み文字は、BEA Tuxedo /Q コンポーネントのリリースごとに異なります。TPQNOMSGID が設定されていると、メッセージ識別子は利用できません。

制御構造体の残りのメンバーは、TPENQUEUE() への入力では使用しません。

TPENQUEUE() の呼び出しが異常終了し、TP-STATUS に TPEDIAGNOSTIC が設定された場合は、異常終了の原因を示す値が DIAGNOSTIC に返されます。返される可能性のある値は、この後の「診断」の項で定義しています。

戻り値

TPENQUEUE() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPENQUEUE() は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
無効な引数が指定されました (たとえば、QSPACE-NAME が SPACES である場合、*TPQUEDEF-REC* の設定が無効である場合など)。

[TPENOENT]
QSPACE-NAME を利用できないので、これにアクセスできません (関連する TMQUEUE(5) サーバは利用できません)。または、グローバル・トランザクション・テーブル (GTT) にエントリがないので、グローバル・トランザクションを開始できません。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションはアボートのみになります。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、TPDEQUEUE( ) または TPENQUEUE( ) を呼び出そうとしても TPETIME が発生して失敗します。

[TPEBLOCK]
ブロッキング条件が存在し、TPBLOCK が設定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT が設定されていました。

[TPEPROTO]
TPENQUEUE( ) の呼び出し方法が不適切です。キューまたはトランザクションには影響ありません。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。キューには影響ありません。

[TPEOS]
オペレーティング・システムのエラーが発生しました。キューには影響ありません。

[TPEDIAGNOSTIC]
指定されたキューへのメッセージの登録が異常終了しました。異常終了の原因は、*TPQUEDEF-REC* を介して返される診断値によって判別できます。

診断

次の診断値は、キューへのメッセージの登録中に返されます。

[QMEINVAL]
無効な設定が指定されました。

[QMEBADRMID]
無効なリソース・マネージャ識別子が指定されました。

[QMENOTOPEN]
このリソース・マネージャは現在オープンしていません。

[QMETRAN]
呼び出しがトランザクション・モードでない、または TPNOTRAN 設定で呼び出しが行われ、メッセージをキューに登録するトランザクションの開始を試みた際に、エラーが発生しました。この診断は、BEA Tuxedo リリース 7.1 以降のキュー・マネージャでは返されません。

[QMEBADMSGID]
無効なメッセージ識別子が指定されました。

[QMESYSTEM]
システム・エラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[QMEOS]
オペレーティング・システムのエラーが発生しました。

[QMEABORTED]
作業が中途終了しました。グローバル・トランザクション内で実行された場合、グローバル・トランザクションが中途終了のみになっています。それ以外の場合、キュー・マネージャは操作を中断しました。

[QMEPROTO]
トランザクション状態がアクティブでないときに、キューへの登録が行われました。

[QMEBADQUEUE]
無効な、または削除された、あるいは予約されているキューの名前が指定されました。

[QMENOSPACE]
キューにスペースがないなど、リソース不足のため、必要なサービスの内容 (永続ストレージまたは非永続ストレージ) を設定されたメッセージがキューに登録されませんでした。次のいずれかのリソース設定を超過すると、QMENOSPACE が返されます。(1) キュー・スペースに割り当てられたディスク (永続的) スペースの容量、(2) キュー・スペースに割り当てられたメモリ (非永続的) スペースの容量、(3) キュー・スペースで使用可能な、同時にアクティブ状態になるトランザクションの最大数、(4) キュー・スペースに一度に入れることができる最大メッセージ数、(5) キューイング・サービス・コンポーネントが処理できる並列アクションの最大数、または (6) キューイング・サービス・コンポーネントを同時に使用できる認証ユーザの最大数。

[QMERELEASE]
新機能をサポートしていないバージョンの BEA Tuxedo システムのキュー・マネージャに対して、メッセージをキューに登録するような処理が行われました。

[QMESHARE]
指定されたキューのメッセージを登録する際に、そのキューが別のアプリケーションによって排他的にオープンされています。BEA Tuxedo システム以外の BEA 製品ベースのアプリケーションが、キューイング・サービス API (QSAPI) を使用して読み取りおよび書き込みを排他的に行うため、キューをオープンしています。

関連項目 qmadmin(1)、TPDEQUEUE(3cbl)、TPSPRIO(3cbl)、TMQFORWARD(5)、TMQUEUE(5)

TPFORWAR(3cbl)

名前　TPFORWAR() - BEA Tuxedo ATMI のサービス要求を別のルーチンに転送

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

COPY TPFORWAR REPLACING TPSVCDEF-REC BY TPSVCDEF-REC
 TPTYPE-REC BY TPTYPE-REC
 DATA-REC BY DATA-REC
 TPSTATUS-REC BY TPSTAUS-REC
```

機能説明

TPFORWAR() を使用すると、サービス・ルーチンはクライアントの要求を別のサービス・ルーチンに転送して処理させることができます。TPFORWAR() には EXIT PROGRAM 文が含まれるので、BEA Tuxedo ATMI のディスパッチャに制御を確実に正しく返すために、TPFORWAR() はそれが起動されたルーチンと同じルーチンの中から呼び出されなければなりません。つまり、TPFORWAR() をサービス・ルーチンのサブ・プログラムから呼び出してはなりません。制御が BEA Tuxedo ATMI のディスパッチャに返されないからです。TPFORWAR() を会話サービスの中から呼び出すことはできません。

このルーチンは、*DATA-REC* に入っているデータを使用して *TPSVCDEF-REC* 内の SERVICE-NAME によって指定されるサービスに要求を転送します。要求を転送するサービス・ルーチンは応答を受け取りません。要求の転送が終わると、サービス・ルーチンは BEA Tuxedo ATMI のディスパッチャに戻り、他の作業を行える状態になります。なお、転送された要求からの応答は期待しないため、この要求は、要求を転送するサービスと同じ実行可能プログラム内の任意のサービス・ルーチンに、エラーなしで転送することができます。

サービス・ルーチンがトランザクション・モードであると、このルーチンはトランザクションの呼び出し元の部分を、そのトランザクションの起動元が TPCOMMIT( ) または TPABORT( ) のいずれかを出したときに完了できるような状態にします。トランザクションがサービス・ルーチンの実行中に TPBEGIN( ) により明示的に開始された場合、このトランザクションを TPCOMMIT( ) または TPABORT( ) で終了させてから、TPFORWAR( ) を呼び出さなければなりません。このため、転送チェインに関与するすべてのサービスは、トランザクション・モードで起動するか、あるいはどれもトランザクション・モードでは起動しないようにします。

転送チェインの最後のサーバは、TPRETURN( ) を使用して要求の発信元に応答を返します。要約して言えば、TPFORWAR( ) は待機している要求者に応答を返す役割を別のサーバに転送するわけです。

TPFORWAR( ) は、サービス・ルーチンが出したサービス要求から期待されるすべての応答を受け取った後、呼び出すようにしてください。受信されていない未終了の応答は、受信後、BEA Tuxedo ATMI のディスパッチャによって自動的に取り除かれます。さらに、以後、これらの応答の通信ハンドルは無効になり、要求は SERVICE-NAME に転送されません。

DATA-REC は送信されるレコードであり、*TPTYPE-REC* 内の LEN は、送信する *DATA-REC* の大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES の場合、*DATA-REC* および LEN は無視され、長さゼロのデータで要求がデータ長ゼロで送信されます。REC-TYPE が STRING で、LEN が 0 の場合は、要求はデータ部なしで送信されます。

サービス・ルーチンの作成者は TPFORWAR( ) の呼び出し後は制御を保持しないので、シグナル・リスタートによるブロッキング送信が使用されます (つまり、TPSIGRSTRT)。現時点では、*TPSVCDEF-REC* の設定は将来使用するために予約されており、指定しても無視されます。

戻り値

サービス・ルーチンは、呼び出し元、BEA Tuxedo ATMI のディスパッチャに何も値を返さないので、TP-STATUS は設定されません。

エラー

このルーチンに渡されたパラメータの処理中またはそのルーチン自体の処理中にエラーが発生した場合、異常終了メッセージが最初の要求元に返されます (応答が送信されない場合を除く)。未終了の応答または下位接続がある場合や呼び出し元のトランザクションに中途終了マークが付けられた場合には、異常終了メッセージの原因となる障害であるとみなされます。要求者は、エラーを示す TPESVCERR( ) によって、異常終了メッセージを検出します。このようなエラーが発生すると、呼び出し元のデータは送信されません。また、このエラーが原因で、呼び出し元の現在のトランザクションに中途終了のマークが付けられます。

サービス・ルーチンの処理中あるいは要求の転送中にトランザクション・タイムアウトになると、TPCALL() または TPGETRPLY() で応答を待つ要求元は TPETIME エラーを受け取ります。また、この要求元はデータを受け取ることはありません。ただし、サービス・ルーチンは TPRETURN() または TPFORWAR() のいずれかを使用して終了することになっています。会話サービス・ルーチンは、TPRETURN() を使用しなければならず、TPFORWAR() は使用できません。

サービス・ルーチンが TPRETURN() も TPFORWAR() も使用しない場合、または TPFORWAR() が会話サーバから呼び出された場合、そのサーバはログ・ファイルに警告メッセージを出力し、最初の要求者にサービス・エラーを返します。下位接続に対するオープン接続はすべて、ただちに切断され、未終了の非同期応答は無効とマークされます。サーバが異常終了時にトランザクション・モードにあった場合は、そのトランザクションには中途終了マークが付けられます。TPRETURN() または TPFORWAR() のいずれかがサービス・ルーチンとは別に (たとえば、クライアント、TPSVRINIT()、TPSVRDONE() などから) 使用されても、これらのルーチンは何も影響を及ぼさず終了するだけです。

関連項目　TPCONNECT(3cbl)、TPRETURN(3cbl)

TPGETCTXT(3cbl)

名前　TPGETCTXT() - 現在のアプリケーション関連のコンテキスト識別子を取り出す

形式

```
01 TPCONTEXTDEF-REC.
   COPY TPCONTEXTDEF.

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPGETCTXT" USING TPCONTEXTDEF-REC TPSTATUS-REC.
```

機能説明　TPGETCTXT() は、現在のアプリケーション関連のコンテキストを表す識別子を検索し、その識別子を TPCONTEXTDEF-REC 内の CONTEXT に入れます。一般的な COBOL アプリケーションの処理は次のとおりです。

1. TP-MULTI-CONTEXTS フラグを設定して TPINITIALIZE() を呼び出す。
2. TPGETCTXT() を呼び出し、TPCONTEXTDEF-REC を保存する。
3. 再度 TP-MULTI-CONTEXTS フラグを設定して、TPINITIALIZE() を呼び出す。
4. 再度 TPGETCTXT() を呼び出し、返されたコンテキストを保存する。
5. TPSETCTXT() を呼び出し、最初のコンテキストに戻る。

TPGETCTXT() は、シングルコンテキスト・アプリケーションおよびマルチコンテキスト・アプリケーションで呼び出すことができます。

戻り値　TPGETCTXT は正常終了時には、TP-STATUS に [TPOK] を設定し、プログラムのコンテキスト識別子を TPCONTEXTDEF-REC 内の CONTEXT に入れます。CONTEXT には現在のコンテキスト ID が設定されます。これは、次のいずれかで表されます。

- 実際のコンテキスト ID。
- TPNULLCONTEXT。この場合、プログラムは現在、コンテキストと関連していません。

注記 TPINVALIDCONTEXT が COBOL プログラムで返されることはありません。この値は、マルチスレッドのプログラムでしか返されないからです。

エラー　TPGETCTXT の異常終了時には、TP-STATUS に次のいずれかの値が設定されます。

[TPEINVAL]
　無効な引数が指定されました。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容は、ログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　「COBOL アプリケーション・トランザクション・モニタ・インターフェイスの紹介」、TPSETCTXT(3cbl)

TPGETLEV(3cbl)

名前　TPGETLEV( ) - BEA Tuxedo ATMI のトランザクションの進行状況のチェック

形式

```
01 TPTRXLEV-REC.
 COPY TPTRXLEV.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPGETLEV" USING TPTRXLEV-REC TPSTATUS-REC.
```

機能説明　TPGETLEV( ) は現在のトランザクション・レベルを呼び出し元に返します。現時点では、定義されているレベルは TP-NOT-IN-TRAN と TP-IN-TRAN だけです。

戻り値　TPGETLEV( ) は正常終了時には、TP-STATUS に [TPOK] を設定し、*TPTRXLEV-REC* に値を設定します。TPTRXLEV-REC に設定される値は、進行中のトランザクションがないことを示す TP-NOT-IN-TRAN 、トランザクションが進行中であることを示す TP-IN-TRAN のいずれかです。

エラー　次の条件が発生すると、TPGETLEV( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEPROTO]
: TPGETLEV( ) の呼び出し方法が不適切です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

注意事項　TPBEGIN( )、TPCOMMIT( )、および TPABORT( ) を使用して BEA Tuxedo ATMI のトランザクションを記述する際には、XA インターフェイスに準拠した (および呼び出し元に適切にリンクされている) リソース・マネージャが行う作業のみがトランザクションとしての特性を備えていることを忘れないようにすることが重要です。トランザクションで行われるその他の処理内容は、TPCOMMIT( ) や TPABORT( ) の影響を受けません。そのリソース・マネージャが行った処理が BEA Tuxedo ATMI のトランザクションの一部となるよう、XA インターフェイスを満たすリソース・マネージャをサーバにリンクする方法については、buildserver(1) を参照してください。

関連項目　TPABORT(3cbl)、TPBEGIN(3cbl)、TPCOMMIT(3cbl)、TPSCMT(3cbl)

TPGETRPLY(3cbl)

名前　TPGETRPLY( ) - 非同期メッセージからの応答の取得

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data

01 TPSTATUS-REC.
 COPY TPSTATUS

CALL "TPGETRPLY" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明

TPGETRPLY( ) は、事前に送られた要求の応答を返します。TPGETRPLY( ) は、特定の要求に対する応答、または取得可能なすべての応答を返します。選択可能なこの 2 つについて、ここで説明します。

DATA-REC は、応答を読み込む場所を指定し、*TPTYPE-REC* 内の LEN は、*DATA-REC* に移動すべき最大バイト数を示します。また、*TPTYPE-REC* の REC-TYPE を指定する必要があります。TPGETRPLY( ) の正常終了時には、実際に *DATA-REC* に移動されたバイト数が LEN に入り、*TPTYPE-REC* 内の REC-TYPE および SUB-TYPE には、それぞれデータのタイプおよびサブタイプが入ります。応答が *DATA-REC* より大きい場合は、*DATA-REC* にはこのレコードに入るバイト数分のみが入ります。応答の残りは破棄され、TPGETRPLY( ) は TPTRUNCATE( ) を設定します。

正常終了時に LEN が 0 である場合は、応答にはデータ部がなく、*DATA-REC* は変更されていません。入力時に LEN を 0 にすると、エラーになります。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPGETANY
: この値を設定すると、TPGETRPLY( ) は *TPSVCDEF-REC* 内の COMM-HANDLE によって示される通信ハンドルを無視し、存在する応答があればそれらを返し、返された応答の通信ハンドルを指すよう COMM-HANDLE を設定します。応答が存在しなければ、TPGETRPLY( ) は応答が届くまで待機します。TPGETANY と TPGETHANDLE のいずれかを設定しなければなりません。

TPGETHANDLE
: この値を設定すると、TPGETRPLY( ) は COMM-HANDLE によって示される通信ハンドルを使用し、その COMM-HANDLE に対応する応答があればそれを返します。応答が存在しなければ、TPGETRPLY( ) は応答が届くまで待機します。TPGETANY と TPGETHANDLE のいずれかを設定しなければなりません。

TPNOCHANGE
: この値を設定すると、*DATA-REC* のタイプは変更されなくなります。すなわち、応答レコードのタイプおよびサブタイプは、それぞれ REC-TYPE および SUB-TYPE と一致しなければなりません。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPCHANGE
: 応答レコードのタイプとサブタイプの両方または片方は、受け取り側が着信レコードのタイプを識別するかぎりは、それぞれ REC-TYPE および SUB-TYPE と異なっています。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPNOBLOCK
: TPGETRPLY( ) は、応答が送られてくるまで待機しません。応答が取り出せる状態であれば、TPGETRPLY( ) はその応答を取り込み、終了します。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK が設定されていて、データも取り出せる状態にない場合、呼び出し元は応答が到着するまで、あるいはタイムアウト (トランザクションまたはブロッキング) が発生するまでブロックされます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元をその応答に対して無期限にブロックでき、ブロッキング・タイムアウトの影響も受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

特に説明がなければ、COMM-HANDLE は対応する応答が受信された後は、無効になります。

戻り値

TPGETRPLY( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。TP-STATUS に TPOK( ) または TPESVCFAIL( ) が設定されると、*TPSTATUS-REC* 内の APPL-RETURN-CODE に TPRETURN( ) の一部として送信されたアプリケーション定義の値が入ります。着信メッセージの大きさが入力時に LEN に指定されたより大きい場合は、TPTRUNCATE( ) が設定され、LEN 分のデータのみが *DATA-REC* に移動されて残りのデータは破棄されます。

エラー

次の条件が発生すると、TPGETRPLY( ) は異常終了し、TP-STATUS を以下のように設定します。TPGETHANDLE が設定されている場合は、特に明記されないかぎり、COMM-HANDLE は無効になります。TPGETANY が設定されていると、COMM-HANDLE は障害を起こした応答の通信ハンドルを示します。応答を取り出せるようになる前にエラーが発生した場合には、COMM-HANDLE は 0 です。また、特に明記しないかぎり、この異常終了は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました (*TPSVCDEF-REC* の設定が無効など)。

[TPEOTYPE]
: 応答のタイプまたはサブタイプのいずれかは、呼び出し元が認識しているものでありません。あるいは、TPNOCHANGE が設定されていて、REC-TYPE と SUB-TYPE が、そのサービスから送られた応答のタイプおよびサブタイプと一致しません。*DATA-REC* も *TPTYPE-REC* も変更されません。呼び出し元の現在のトランザクションのために応答が受信された場合は、そのトランザクションには中途終了マークが付けられます。

[TPEBADDESC]
: COMM-HANDLE に無効な通信ハンドルが入っています。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合、トランザクション・タイムアウトが発生し、そのトランザクションはアボートのみになります。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。いずれの場合も、*DATA-REC* も *TPTYPE-REC* も変更されません。COMM-HANDLE は、呼び出し元がトランザクション・モードでなければ (そして、TPGETHANDLE が設定されている場合) そのまま有効です。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、新しいリクエストを送信したり、未処理の応答を受信しようとしても、[TPETIME] が発生して失敗します。

[TPESVCFAIL]
呼び出し元の応答を送るサービス・ルーチンが、TPFAIL( ) を設定して TPRETURN( ) を呼び出しました。これは、アプリケーション・レベルの障害です。サービスの応答の内容は (送信された場合)、*DATA-REC* に入ります。APPL-RETURN-CODE には、TPRETURN( ) の一部として送信されたアプリケーション定義の値が入ります。応答が呼び出し元のトランザクションのために受信された場合、トランザクションには中途終了マークが付けられます。トランザクションがタイムアウトになったかどうかにかかわらず、トランザクションがアボートされる前に有効な通信は、TPNOREPLY、TPNOTRAN、および TPNOBLOCK を設定した TPACALL( ) の呼び出しだけです。

[TPESVCERR]
サービス・ルーチンが TPRETURN( ) あるいは TPFORWAR( ) で完了する際にエラーを検出しました。(たとえば、誤った引数が渡された場合など)。このエラーが発生すると、応答データは返されません (つまり、*DATA-REC* も *TPTYPE-REC* も変更されません)。応答が呼び出し元のトランザクションのために受信された場合、トランザクションには中途終了マークが付けられます。トランザクションがタイムアウトになったかどうかにかかわらず、トランザクションがアボートされる前に有効な通信は、TPNOREPLY、TPNOTRAN、および TPNOBLOCK を設定した TPACALL( ) の呼び出しだけです。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。COMM-HANDLE はそのまま有効です。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPGETRPLY( ) の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　TPACALL(3cbl)、TPCANCEL(3cbl)、TPRETURN(3cbl)

TPGETUNSOL(3cbl)

名前　　TPGETUNSOL() - 任意通知型メッセージの獲得

形式

```
01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPGETUNSOL" USING TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　TPGETUNSOL() は、TPBROADCAST() または TPNOTIFY() によって送られた任意通知型メッセージを取得します。このルーチンは、任意通知型メッセージ・ハンドラからのみ呼び出し可能です。

正常終了時には、*DATA-REC* に移動された実際のバイト数が *TPTYPE_REC* 内の LEN に入ります。*TPTYPE-REC* 内の REC-TYPE および SUB-TYPE にはそれぞれデータのタイプおよびサブタイプが入ります。メッセージが *DATA-REC* より大きい場合は、*DATA-REC* にはこのレコードに入るバイト数分のみが入ります。メッセージの残りは破棄され、TPTRUNCATE() が設定されます。正常終了時に LEN が 0 である場合は、メッセージにはデータ部がなく、*DATA-REC* は変更されていません。

入力時に LEN を 0 にすると、エラーになります。

戻り値　　TPGETUNSOL() の正常終了時には、TP-STATUS に [TPOK] が設定されます。着信メッセージの大きさが入力時に LEN に指定されたより大きい場合、TPTRUNCATE() が設定され、LEN 分のデータのみが *DATA-REC* に移動されて残りのデータは破棄されます。

エラー　　次の条件が発生すると、TPGETUNSOL() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
　　無効な引数が指定されました。

[TPEPROTO]
　　TPGETUNSOL() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPSETUNSOL(3cbl)

TPGPRIO(3cbl)

名前　　TPGPRIO() - サービス要求の優先順位を獲得

形式

```
01 TPPRIDEF-REC.
 COPY TPPRIDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPGPRIO" USING TPPRIDEF-REC TPSTATUS-REC.
```

機能説明　　TPGPRIO() は、最後に送信あるいは受信した要求の優先順位を返します。優先順位の範囲は 1 から 100 までです。最高優先順位は 100 です。TPGPRIO() は TPCALL() または TPACALL() (また、キュー機能がインストールされている場合は TPENQUEUE() あるいは TPDEQUEUE()) の後で呼び出すことができ、返される優先順位は送信された要求のものです。また、TPGPRIO() はサービス・ルーチン内から呼び出して、呼び出されたサービスがどの優先順位で送られたかを明らかにします。TPGPRIO() は何回でも呼び出すことができ、次の要求が送られるまでは同じ値を返します。

会話プリミティブは優先順位と関連付けられていないので、TPSEND() や TPRECV() を発行しても、TPGPRIO() が返す優先順位には影響しません。また、会話サービス・ルーチンの場合も、TPCALL() や TPACALL() がそのサービス内から出されないかぎり、優先順位は関連付けられません。

戻り値　　TPGPRIO() の正常終了時には、TP-STATUS に [TPOK] が設定され、要求の優先順位が *TPPRIDEF-REC* 内の PRIORITY に返されます。

エラー　　次の条件が発生すると、TPGPRIO() は異常終了し、TP-STATUS に次の値を設定します。

[TPENOENT]
: TPGPRIO() が呼び出されましたが、(TPCALL() または TPACALL() を介して) 何の要求も送信されなかったか、要求が送られなかった会話サービスの中で TPGPRIO() が呼び出されました。

[TPEPROTO]
: TPGPRIO() の呼び出し方法が不適切です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPACALL(3cbl)、TPCALL(3cbl)、TPDEQUEUE(3cbl)、TPENQUEUE(3cbl)、TPSPRIO(3cbl)

TPINITIALIZE(3cbl)

名前　　TPINITIALIZE() - BEA Tuxedo ATMI アプリケーションに参加する

形式

```
01 TPINFDEF-REC.
   COPY TPINFDEF

01 USER-DATA-REC PIC X(any-length).

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPINITIALIZE" TPINFDEF-REC USER-DATA-REC TPSTATUS-REC.
```

機能説明

TPINITIALIZE() は、クライアントが BEA Tuxedo ATMI のアプリケーションに参加するときに使用します。クライアントが BEA Tuxedo システムの通信ルーチンまたはトランザクション・ルーチンを使用するには、まず BEA Tuxedo ATMI のアプリケーションに参加する必要があります。TPINITIALIZE() には、シングルコンテキスト・モードとマルチコンテキスト・モードの 2 つの動作モードがあります。これについては、後で詳しく説明します。シングルコンテキスト・モードでは TPINITIALIE() の呼び出しは任意選択なので、シングルコンテキストのクライアントは、TPINITIALIZE() を透過的に呼び出す多数の ATMI ルーチン (たとえば、TPACALL() や TPCALL()) を呼び出すことによりアプリケーションに参加することもできます。この場合、TPINITIALIZE() は、*TPINFDEF-REC* のメンバにデフォルト値を使用して呼び出されます。クライアントは TPINITIALIZE() を直接呼び出すこともでき、その場合は、ここで説明するパラメータを設定することができます。マルチコンテキスト・モードが必要な場合、またはアプリケーション認証が必要な場合には、TPINITIALIZE() を使用する必要があります (UBBCONFIG(5) の SECURITY キーワードの説明を参照)。TPINITIALIZE() が正常終了した後、クライアントはサービス要求を開始し、トランザクションを定義できます。

シングルコンテキスト・モードで TPINITIALIZE() が 2 回以上呼び出された場合、つまり、クライアントがアプリケーションに参加した後で呼び出された場合は、アクションは何も実行されず、正常終了を示す戻り値が返されます。

TPINFDEF-REC レコードの説明

TPINFDEF-REC レコードは、次のようなメンバーで構成されています。

```
05 USRNAME            PIC X(30).
05 CLTNAME            PIC X(30).
05 PASSWD             PIC X(30).
05 GRPNAME            PIC X(30).
05 NOTIFICATION-FLAG  PIC S9(9) COMP-5.
  88 TPU-SIG             VALUE 1.
  88 TPU-DIP             VALUE 2.
  88 TPU-IGN             VALUE 3.
05 ACCESS-FLAG        PIC S9(9) COMP-5.
  88 TPSA-FASTPATH        VALUE 1.
  88 TPSA-PROTECTED       VALUE 2.
05 CONTEXTS-FLAG      PIC S9(9) COMP-5.
  88 TP-SINGLE-CONTEXT    VALUE 0.
  88 TP-MULTI-CONTEXTS    VALUE 1.
05 DATALEN            PIC S9(9) COMP-5.
```

USRNAME は呼び出し元を表す名前です。CLTNAME は、その意味付けがアプリケーション側で定義されているクライアント名です。値 sysclient は、CLTNAME フィールド用にシステムによって予約されています。USRNAME および CLTNAME フィールドは TPINITIALIZE() 実行時にクライアントと関連付けられ、ブロードキャスト通知と管理統計情報の検索に使用されます。PASSWD は、アプリケーション・パスワードとの認証に使用される非暗号化形式のアプリケーション・パスワードです。PASSWD は 30 文字まで有効です。GRPNAME は、クライアントをリソース・マネージャ・グループ名と関連付けるときに使用します。GRPNAME が SPACES の場合、クライアントは、リソース・マネージャに関連付けられず、デフォルトのクライアント・グループになります。

シングルコンテキスト・モードとマルチコンテキスト・モード

TPINITIALIZE() には、シングルコンテキスト・モードとマルチコンテキスト・モードの 2 つの動作モードがあります。シングルコンテキスト・モードでは、プロセスは一度に 1 つのアプリケーションにしか参加できません。シングルコンテキスト・モードは、CONTEXTS-FLAG の TP-SINGLE-CONTEXT を設定して TPINITIALIZE() を呼び出すか、TPINITIALIZE() を暗黙的に呼び出す別の関数を呼び出すことによって指定されます。

シングルコンテキスト・モードで TPINITIALIZE() が 2 回以上呼び出された場合、つまり、クライアントがアプリケーションに参加した後で呼び出された場合は、アクションは何も実行されず、正常終了を示す戻り値が返されます。

マルチコンテキスト・モードは、CONTEXTS-FLAG の TP-MULTI-CONTEXTS を設定して TPINITIALIZE() を呼び出すことにより指定されます。マルチコンテキスト・モードでは、TPINITIALIZE() を呼び出すたびに別のアプリケーション関連が作成されます。

アプリケーション関連とは、プロセスと BEA Tuxedo アプリケーションを関連付けるコンテキストです。クライアントは、複数の BEA Tuxedo アプリケーションとの関連、または同じアプリケーションとの複数の関連を作成できます。クライアントのすべての関連は、同じリリースの BEA Tuxedo システムを実行するアプリケーションに対して作成します。また、すべての関連がネイティブ・クライアントであるか、すべての関連がワークステーション・クライアントでなければなりません。

ネイティブ・クライアントの場合、新しい関連の生成先になるアプリケーションは、TUXCONFIG 環境変数の値で決まります。ワークステーション・クライアントの場合、新しい関連の生成先になるアプリケーションは、WSNADDR または WSENVFILE 環境変数の値で決まります。現在の COBOL プロセスのコンテキストは、新しい関連に設定されます。

マルチコンテキスト・モードでは、アプリケーションは TPGETCTXT() を呼び出すことにより現在のコンテキストを表すハンドルを取得し、そのハンドルをパラメータとして TPSETCTXT() に渡して、特定の COBOL プロセスが動作するコンテキストを設定することができます。

シングルコンテキスト・モードとマルチコンテキスト・モードを一緒に使用することはできません。アプリケーションでいずれかのモードを選択したら、すべてのアプリケーション関連に対して TPTERM() を呼び出すまで、TPINITIALIZE() をほかのモードで呼び出すことはできません。

TPINFDEF-REC レコードの説明

TPINFDEF-REC の設定により、クライアント固有の通知機構とシステム・アクセスのモードの両方を示すことができます。これらの設定で、アプリケーションのデフォルトの設定を変更することができます。ただし、変更できない場合には、TPINITIALIZE() はログ・ファイルに警告メッセージを記録し、設定を無視して、TPINITIALIZE() の終了時にアプリケーションのデフォルト設定が *TPINFDEF-REC* に返されます。クライアントへの通知の場合、この設定として次のものがあります。

TPU-SIG
: シグナルによる任意通知を選択します。この設定は、CONTEXTS-FLAG の TP-MULTI-CONTEXTS とともに使用することはできません。

TPU-DIP
: ディップ・インによる任意通知を選択します。

TPU-IGN
　　任意通知を無視します。

一度にはこのいずれか 1 つのみ使用できます。クライアントが通知方法を選択しない場合、アプリケーションのデフォルトの方法が、TPINITIALIZE() の終了時に設定されます。

システム・アクセス・モードの設定の場合、設定として次のものがあります。

TPSA-FASTPATH
: システム・アクセス・モードを fastpath に設定します。

TPSA-PROTECTED
: システム・アクセス・モードを protected に設定します。

一度にはこのいずれか 1 つのみ使用できます。クライアントが通知方法またはシステム・アクセス・モードを選択しない場合、アプリケーションのデフォルトの方法が、TPINITIALIZE() の終了時に設定されます。クライアントの通知方法とシステム・アクセス・モードの詳細については、UBBCONFIG(5) を参照してください。

DATALEN は、サービスに送信されるアプリケーション固有のデータの長さです。アプリケーションが BEA Tuxedo システムのアプリケーション認証機能を使用しない場合、USRNAME および CLTNAME に SPACES 値を使用できます。現時点では、GRPNAME は SPACES でなければなりません。このオプションを使用するクライアントは、BEA Tuxedo システムにおいて次のように定義されます。USRNAME、CLTNAME、および GRPNAME はデフォルト値、デフォルト設定で、アプリケーション・データは得られません。

戻り値

TPINITIALIZE() は正常終了時には、TP-STATUS に [TPOK] を設定します。異常終了時には、TPINITIALIZE() は呼び出し元プロセスのコンテキストを変更せず、-1 を返して、TP-STATUS にエラー条件を示す値を設定します。

エラー

TPINITIALIZE() は異常終了時には、TP-STATUS に次の値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。

[TPENOENT]
: BEA Tuxedo システムの掲示板に領域が残っていないため、クライアントはアプリケーションに参加できません。

[TPEPERM]
: そのための許可をもっていないか、正しいアプリケーション・パスワードが与えられなかったため、クライアントはアプリケーションに結合できません。許可が与えられない理由として、アプリケーション・パスワードが無効であった場合、アプリケーション固有の認証検査にパスできない場合、あるいは名前の使用制限がある場合などがあります。

[TPEPROTO]
: TPINITIALIZE() の呼び出し方法が不適切です。たとえば、(a) 呼び出し元がサーバである場合、(b) シングルコンテキスト・モードで TP-MULTI-CONTEXTS 設定が指定された場合、または (c) マルチコンテキスト・モードで TP-MULTI-CONTEXTS 設定が指定されなかった場合などがあります。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

移植性

TPINITIALIZE() に記述されているインターフェイスは、UNIX システムおよび MS-DOS オペレーティング・システム上でサポートされています。ただし、シグナル・ベースの通知方法は、MS-DOS ではサポートされていません。TPINITIALIZE() の実行時にこの通知方法が選択されると、USERLOG() メッセージが生成され、通知方法は自動的にディップインに設定されます。

環境変数

TUXCONFIG
TPINITIALIZE() がワークステーション・クライアントでないネイティブ・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。この変数は、クライアントの接続先になるアプリケーションを示します。なお、この環境変数は、TPINITIALIZE() が呼び出されるときにのみ参照されます。それ以降の呼び出しでは、アプリケーション・コンテキストが使用されます。

WSENVFILE
TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。この変数には、環境変数の設定条件を収めたファイルを指定しますが、この設定は呼び出し元の環境で行うようにします。ワークステーション・クライアントに必要とされる環境変数の設定に関する詳細については、compilation(5) を参照してください。なお、このファイルは、TPINITIALIZE() が呼び出されるとき (その前ではなく) にのみ処理されます。

WSNADDR
TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。これは、アプリケーションをアクセスするときに使用されるワークステーション・リスナ・プロセスのネットワーク・アドレスを示します。この変数はワークステーション・クライアントでは必要ですが、ネイティブ・クライアントの場合は無視されます。

TCP/IP アドレスは次の形式で指定します。

"//host.name:port_number"
"//#.#.#.#:port_number"

最初の形式では、ドメインはローカル・ネーム解決機能 (通常 DNS) を使って、*hostname* のアドレスを見つけます。*hostname* はローカル・マシンでなければならず、ローカル・ネーム解決機能は、ローカル・マシンのアドレスへ *hostname* を解決しなければなりません。

2 番目の例の「*#.#.#.#*」はドットで区切った 10 進数の形式です。ドットで区切った 10 進数の形式では、各 # は 0 から 255 までの数でなければなりません。このドットで区切った 10 進数は、ローカル・マシンの IP アドレスを表現します。

上記両方の形式で、*port_number* はドメイン・プロセスが着信要求をリスンする TCP ポート番号です。*port_number* は 0 から 65535 の間の数字または名前です。*port_number* が名前の場合は、ローカル・マシンのネットワーク・サービス・データベースになければなりません。

アドレスは、先頭に "0x" がついている場合は、16 進形式で指定することもできます。"0x" の後の各文字は、0 から 9 までの数字か、A から F までの英字 (大文字・小文字に関係なく) です。16 進数の形式は、IPX/SPX や TCP/IP のような任意のバイナリ・ネットワーク・アドレスに使うことができます。

アドレスはまた、任意の文字列として指定することもできます。値は、コンフィギュレーション・ファイルの中の NETWORKS セクションの NLSADDR パラメータに指定された値と同じでなければなりません。

WSNADDR アドレス用のコンマで区切られたパス名のリストを指定すると、複数のアドレスを指定することができます。接続が確立するまで順番にアドレス指定が試みられます。アドレス・リストのメンバは、どれでもパイプで区切られたネットワーク・アドレスのかっこ付きのグループとして指定することができます。例えば、

```
WSNADDR="(//m1.acme.com:3050|//m2.acme.com:3050),//m3.acme.com:3050"
```

Windows 下で実行するには、アドレスは次のようになります。

```
set WSNADDR=(//m1.acme.com:3050^|//m2.acme.com:3050),//m3.acme.com:3050
```

パイプ記号 (|) は Windows では特殊文字とみなされるので、コマンド行で指定する際は、その前に Windows 環境での拡張文字のカレット (^) を付ける必要があります。ただし、WSNADDR が envfile で定義される場合、BEA Tuxedo システムは tuxgetenv(3c) 関数を通して、WSNADDR 定義の値を取得します。この場合、パイプ記号 (|) は特殊文字とはみなされないので、カレット (^) を使用する必要はありません。

BEA Tuxedo システムはかっこ付きアドレスを無作為に選択します。この方法は、一連のリスナ・プロセスに対してランダムに負荷分散します。接続が確立するまで順番にアドレス指定が試みられます。ワークステーション・リスナを呼び出すには、アプリケーションのコンフィギュレーション・ファイルの値を使用してください。値の先頭が "0x" のときは、16 進の文字列として解釈されます。それ以外の場合は、ASCII キャラクタとして解釈されます。

WSFADDR

TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。この変数は、ワークステーション・クライアントがワークステーション・リスナまたはワークステーション・ハンドラに接続するのに使用するネットワーク・アドレスを指定します。この変数は、WSFRANGE 変数とともに、ワークステーション・クライアントがアウトバウンド接続を行う前にバインドしようとする TCP/IP ポートの範囲を決定します。このアドレスには、TCP/IP アドレスを指定する必要があります。TCP/IP アドレスのポート部分は、ワークステーション・クライアントによってバインドされる一連の TCP/IP ポートのベース・アドレスを表します。WSFRANGE 変数は、範囲の大きさを指定します。たとえば、このアドレスが //mymachine.bea.com:30000 で WSFRANGE が 200 の場合、この *LMID* からアウトバウンド接続を試みるネイティブ・プロセスはすべて、mymachine.bea.com 30000 から 30200 の間のポートにバインドします。この変数を設定しないと、デフォルトで空の文字列が使用され、オペレーティング・システムはローカル・ポートをランダムに選択します。

WSFRANGE

TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。この変数は、ワークステーション・クライアント・プロセスがアウトバウンド接続を行う前にバインドしようとする TCP/IP ポートの範囲を指定します。WSFADDR パラメータは、範囲のベース・アドレスを指定します。たとえば、WSFADDR パラメータが //mymachine.bea.com:30000 に設定され、WSFRANGE が 200 に設定されると、この *LMID* からアウトバウンド接続を試みるネイティブ・プロセスはすべて、mymachine.bea.com 30000 から 30200 の間のポートにバインドします。有効な範囲は 1 から 65535 までです。デフォルト値は 1 です。

WSDEVICE

TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、TPINITIALIZE() 内で使用されます。これは、ネットワークのアクセス時に使用するデバイス名を示します。この変数はワークステーション・クライアントが使用し、ネイティブ・クライアントの場合は無視されます。なお、ソケットや NetBIOS などトランスポート・レベルのネットワーク・インターフェイスはデバイス名を必要としません。このようなインターフェイスによってサポートされているワークステーション・クライアントは、WSDEVICE を指定する必要はありません。

WSTYPE

TPINITIALIZE() がワークステーション・クライアントによって呼び出される際に、ネイティブ・サイトと符号化および復号化の義務について折衝するために TPINITIALIZE() 内で使用されます。この変数はワークステーション・クライアントの場合は省略可能で、ネイティブ・クライアントの場合は無視されます。

WSRPLYMAX

TPINITIALIZE() によって使用され、アプリケーション応答をファイルにダンプする前のバッファリングに使用するコア・メモリの最大容量を設定します。このパラメータのデフォルトの値は、実装方法に応じて異なります。詳細については、該当するプログラマ・ガイドを参照してください。

TMMINENCRYPTBITS

BEA Tuxedo システムに接続する際に必要な最小レベルの暗号化を確立するために使用されます。「0」は暗号化を行わないことを示します。「56」および「128」は、暗号化キーの長さをビット単位で指定します。この最小レベルの暗号化が一致しない時は、リンクの確立は失敗します。デフォルトの値は「0」です。

TMMAXENCRYPTBITS

BEA Tuxedo システムに接続する際の暗号化の最大レベルを調整するために使用されます。「0」は暗号化を行わないことを示します。「56」および「128」は、暗号化キーの長さをビット単位で指定します。デフォルト値は「128」です。

警告

シグナル・ベースの通知方法は、マルチコンテキスト・モードでは使用できません。また、シグナル・ベースの通知方法を選択するクライアントは、シグナルに関する制約から、システムからシグナルを受け取ることはできません。クライアントがシグナルを受け取ることができない場合、システムは、選択したクライアントへの通知方法をディップ・インに切り替えるログ・メッセージを生成し、クライアントはそれ以降ディップ・イン通知を介して通知を受けます。通知方法の詳細については、UBBCONFIG(5) の RESOURCES セクションの NOTIFY パラメータの説明を参照してください。

クライアントのシグナル通知は常にシステムによって行われるので、元の通知呼び出しがどこで行われるかにかかわらず、通知の形態は一貫しています。したがって、シグナル・ベースの通知を使用するには、以下が必要です。

- ネイティブ・クライアントがアプリケーション管理者として実行している必要があります。
- ワークステーション・クライアントはアプリケーション管理者として実行している必要はありません。

アプリケーション管理者の ID は、そのアプリケーションのコンフィギュレーション・ファイルで識別されます。

クライアントにシグナル・ベースの通知を選択すると、ある種の ATMI 呼び出しは正常に実行できないことがあります。このとき、TPSIGRSTRT の指定がない場合、任意通知型メッセージを受け取るため、TPGOTSIG が返されます。

関連項目　TPGETCTXT(3cbl)、TPSETCTXT(3cbl)、TPTERM(3cbl)

TPKEYCLOSE(3cbl)

名前　　TPKEYCLOSE() - 以前にオープンされたキー・ハンドルのクローズ

形式

```
01 TPKEYDEF-REC.
   COPY TPKEYDEF.

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPKEYCLOSE" USING TPKEYDEF-REC TPSTATUS-REC.
```

機能説明　　TPKEYCLOSE() は以前にオープンされたキー・ハンドル、およびそれに関連するすべてのリソースを解放します。プリンシパルの秘密鍵などの重要情報はすべて、メモリから消去されます。

呼び出し側プロセスは *TPKEYDEF-REC* 内の *KEY-HANDLE* を指定する必要があります。*KEY-HANDLE* は、事前の TPKEYOPEN() 呼び出しで返されるキー識別子です。

戻り値　　TPKEYCLOSE() は正常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に [TPOK] を設定します。

エラー　　TPKEYCLOSE() は異常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に次のいずれかの値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。たとえば、*TPKEYDEF-REC* 内の *KEY-HANDLE* が正しく設定されていません。

[TPESYSTEM]
: エラーが発生しました。詳細は、システム・エラーのログ・ファイルを調べてください。

関連項目　　TPKEYGETINFO(3cbl)、TPKEYOPEN(3cbl)、TPKEYSETINFO(3cbl)

TPKEYGETINFO(3cbl)

名前　TPKEYGETINFO() - キー・ハンドルに関連付けられている情報の獲得

形式

```
01 TPKEYDEF-REC.
   COPY TPKEYDEF.

01 ATTVALUE-REC.
   COPY user data

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPKEYGETINFO" USING TPKEYDEF-REC ATTVALUE-REC TPSTATUS-REC.
```

機能説明　TPKEYGETINFO() は、キー・ハンドルについての情報を報告します。キー・ハンドルは、特定のプリンシパルのキーおよびそれに関連付けられている情報を表します。

呼び出し側プロセスは *TPKEYDEF-REC* 内の *KEY-HANDLE* を設定する必要があります。KEY-HANDLE は、事前の TPKEYOPEN() 呼び出しで返されるキー識別子です。

情報が必要な属性は、*TPKEYDEF-REC* 内の *ATTRIBUTE-NAME* によって識別されます。属性名には、SPACES または LOW-VALUES を埋め込むことができます。一部の属性は暗号サービス・プロバイダ固有のものですが、以下に示す属性は、すべてのプロバイダによってサポートされています。

属性	Value
プリンシパル	キー (キー・ハンドル) と関連するプリンシパルを識別する名前です。ヌルで終了する文字列で表されます。
PKENCRYPT_ALG	公開鍵による暗号化のキーで使用される、公開鍵アルゴリズムの ASN.1 DER (Distinguished Encoding Rules) 形式のオブジェクト識別子です。 RSA のオブジェクト識別子は、次の表「アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング」で識別されます。
PKENCRYPT_BITS	公開鍵アルゴリズムのキー長 (RSA の絶対値) です。値は 512 から 2048 ビットまででなければなりません。

属性	Value
SIGNATURE_ALG	デジタル署名のキーで使用される、デジタル署名アルゴリズムの ASN.1 DER 形式のオブジェクト識別子です。 RSA および DSA のオブジェクト識別子は、次の表 「アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング」で識別されます。
SIGNATURE_BITS	デジタル署名アルゴリズムのキー長 (RSA の絶対値) です。これは、512 〜 2048 ビットの範囲内の値でなければなりません。
ENCRYPT_ALG	バルク・データの暗号化のキーで使用される、対称鍵アルゴリズムの ASN.1 DER 形式のオブジェクト識別子です。 DES、3DES、および RC2 のオブジェクト識別子は、次の表「アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング」で識別されます。
ENCRYPT_BITS	対称鍵アルゴリズムのキー長です。これは、40 〜 128 ビットの範囲内の値でなければなりません。 ENCRYPT_ALG で固定キー長によるアルゴリズムが設定されると、ENCRYPT_BITS の値は自動的にその固定キー長に調整されます。たとえば、ENCRYPT_ALG に DES が設定されると、ENCRYPT_BITS の値は自動的に 56 に設定されます。
DIGEST_ALG	デジタル署名のキーで使用される、メッセージ・ダイジェスト・アルゴリズムの ASN.1 DER 形式のオブジェクト識別子です。 MD5 および SHA-1 のオブジェクト識別子は、次の表 「アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング」で識別されます。
プロバイダ	暗号サービス・プロバイダの名前です。
バージョン	暗号サービス・プロバイダのソフトウェアのバージョンです。

次の表に、デフォルトの公開鍵インプリメンテーションでサポートされる ASN.1 DER アルゴリズム・オブジェクト識別子を示します。

アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング

ASN.1 DER **アルゴリズム・オブジェクト識別子**	**アルゴリズム**
{ 0x06, 0x08, 0x2a, 0x86, 0x48, 0x86, 0xf7, 0x0d, 0x02, 0x05 }	MD5
{ 0x06, 0x05, 0x2b, 0x0e, 0x03, 0x02, 0x1a }	SHA1

アルゴリズム・オブジェクト識別子のアルゴリズムへのマッピング (続き)

ASN.1 DER アルゴリズム・オブジェクト識別子	アルゴリズム
{ 0x06, 0x09, 0x2a, 0x86, 0x48, 0x86, 0xf7, 0x0d, 0x01, 0x01, 0x01 }	RSA
{ 0x06, 0x05, 0x2b, 0x0e, 0x03, 0x02, 0x0c }	DSA
{ 0x06, 0x05, 0x2b, 0x0e, 0x03, 0x02, 0x07 }	DES
{ 0x06, 0x08, 0x2a, 0x86, 0x48, 0x86, 0xf7, 0x0d, 0x03, 0x07 }	3DES
{ 0x06, 0x08, 0x2a, 0x86, 0x48, 0x86, 0xf7, 0x0d, 0x03, 0x02 }	RC2

指定された属性に関連付けられている情報は、ユーザ定義の ATTVALUE-REC に格納され、末尾には SPACES が埋められます。この位置に格納できる最大データ量は、*TPKEYDEF-REC* 内の *ATTRIBUTE-LEN* の呼び出しで指定します。

TPKEYGETINFO( ) が完了すると、*ATTRIBUTE-LEN* には実際に返されたデータの大きさが設定されます (埋め込み値を除く)。返されるべきバイト数が *ATTRIBUTE-VALUE-LEN* を超える場合、TPKEYGETINFO( ) は (TPELIMIT エラー・コードを示して) 異常終了し、必要なスペースを *ATTRIBUTE-VALUE-LEN* に設定します。

戻り値

TPKEYGETINFO( ) は正常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に [TPOK] を設定します。

エラー

TPKEYGETINFO( ) は異常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に次のいずれかの値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。たとえば、*KEY-HANDLE* が有効なキーではありません。

[TPESYSTEM]
: エラーが発生しました。詳細は、システム・エラーのログ・ファイルを調べてください。

[TPELIMIT]
: 要求された属性値を入れるためのスペースが不足しています。

[TPENOENT]
: 要求された属性はこのキーと関連していません。

関連項目

TPKEYCLOSE(3cbl)、TPKEYOPEN(3cbl)、TPKEYSETINFO(3cbl)

TPKEYOPEN(3cbl)

名前

TPKEYOPEN() - デジタル署名の生成、メッセージの暗号化または暗号解読のためのキー・ハンドルのオープン

形式

```
01 TPKEYDEF-REC.
   COPY TPKEYDEF.

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPKEYOPEN" USING TPKEYDEF-REC TPSTATUS-REC.
```

機能説明

TPKEYOPEN() によって、呼び出し元でキー・ハンドルを利用できるようになります。キー・ハンドルは、特定のプリンシパルのキーおよびそれに関連付けられた情報を表します。

キーは、以下に示す目的のうち、1 つまたは複数の目的に使用できます。

- デジタル署名を自動的に生成します。メッセージの内容を保護し、特定のプリンシパルがそのメッセージを発信したことを証明します。プリンシパルは、個人またはプロセスのいずれの場合もあります。このタイプのキーは秘密鍵であり、キーの所有者のみが利用できます。

  プリンシパル名と TPKEY-SIGNATURE および TPKEY-AUTOSIGN を設定して TPKEYOPEN() を呼び出すと、プリンシパルの公開鍵のハンドルが返され、AUTOSIGN モードで署名を生成することが可能になります。公開鍵ソフトウェアは、メッセージが送信される直前に、デジタル署名を生成してメッセージに添付します。
- デジタル署名を検証します。メッセージの内容が改ざんされていないこと、および特定のプリンシパルがそのメッセージを発信したことを証明します。

  署名の検証では TPKEYOPEN() を呼び出す必要はありません。検証プロセスは、署名を検証するためにデジタル署名付きメッセージに付属しているデジタル証明書で指定されている公開鍵を使用します。
- 特定のプリンシパルに宛てられたメッセージを自動的に暗号化します。このタイプのキーは、プリンシパルの公開鍵およびデジタル証明書にアクセスするすべてのプロセスが利用できます。

プリンシパル名と TPKEY-ENCRYPT および TPKEY-AUTOENCRYPT を設定して TPKEYOPEN( ) を呼び出すと、(プリンシパルのデジタル証明書を介して) プリンシパルの公開鍵のハンドルが返され、AUTOENCRYPT モードで暗号化することが可能になります。公開鍵ソフトウェアは、メッセージが送信される直前に、メッセージを暗号化し、メッセージに暗号化エンベロープを添付します。暗号化エンベロープにより、受信側プロセスはメッセージを解読することができます。

- 特定のプリンシパルに宛てられたメッセージを解読します。このタイプのキーは秘密鍵であり、キーの所有者のみが利用できます。

  プリンシパル名と TPKEY-DECRYPT を設定して TPKEYOPEN( ) を呼び出すと、プリンシパルの公開鍵およびデジタル証明書のハンドルが返されます。

TPKEYOPEN( ) によって返されるキー・ハンドルは、*TPKEYDEF-REC* 内の *KEY-HANDLE* に格納されます。

呼び出し側プロセスは *TPKEYDEF-REC* 内の *PRINCIPAL-NAME* を設定する必要があります。*PRINCIPAL-NAME* は、キーの所有者のアイデンティティを指定します。この名前の末尾には SPACES または LOW-VALUES を埋め込むことができます。*PRINCIPAL-NAME* がすべて SPACES か LOW-VALUES の場合、デフォルトのアイデンティティが使用されます。デフォルトのアイデンティティは、現在のログイン・セッション、現在のオペレーティング・システム・アカウント、またはローカル・ハードウェア・デバイスなどの属性に基づいて決定されます。

呼び出し側プロセスは、TPKEYDEF-REC 内の *LOCATION* を設定しなければならないことがあります。*LOCATION* はキー所有者のアイデンティティの位置を指定します。プロバイダがこのフィールドを要求していない場合は、このフィールドに SPACES か LOW-VALUES を入力できます。

PRINCIPAL-NAME のアイデンティティを認証するために、パスワードまたはパス・フレーズなどの証明材料が要求されることがあります。必要な場合は、証明材料を *TPKEYDEF-REC* 内の *IDENTITY-PROOF* に格納します。必要ない場合、このフィールドには SPACES か LOW-VALUES を入力できます。

証明材料の長さ (バイト単位) は、*TPKEYDEF-REC* 内の *PROOF-LEN* で指定されます。*PROOF-LEN* が 0 の場合、*IDENTITY-PROOF* は末尾に SPACES か LOW-VALUES を埋め込まれた文字列であるとみなされます。この場合、末尾の SPACES や LOW-VALUES は証明材料の一部とはみなされません。

ローカル・マシンのコンフィギュレーションおよび動作環境に基づいて、暗号サービス・プロバイダを選択しなければならないことがあります。暗号サービス・プロバイダを選択する場合は、必要なプロバイダ名を *TPKEYDEF-REC* 内の *CRYPTO-PROVIDER* に設定します。暗号サービス・プロバイダを選択しない場合、このフィールドに SPACES か LOW-VALUES を入力すると、デフォルトのプロバイダが選択されます。

キーの動作モードに必要なキー・アクセスのタイプは、*TPKEYDEF-REC* に次の設定を 1 つまたは複数指定することにより決定されます。

TPKEY-SIGNATURE:
この秘密鍵はデジタル署名を生成します。

TPKEY-AUTOSIGN:
このプロセスがメッセージを送信するときは常に、公開鍵ソフトウェアは署名者の秘密鍵を使用してデジタル署名を生成し、それをメッセージに添付します。

TPKEY-ENCRYPT:
この公開鍵は、暗号化メッセージの受信者を識別します。

TPKEY-AUTOENCRYPT:
このプロセスがメッセージを送信するときは常に、公開鍵ソフトウェアはメッセージを暗号化し、受信者の公開鍵を使用して暗号化エンベロープを生成します。次に、暗号化エンベロープをメッセージに添付します。

TPKEY-DECRYPT:
この秘密鍵は暗号解読に使用できます。

これらの設定を組み合わせて、指定することができます。キーが暗号化にのみ使用される場合 (*TPKEY-ENCRYPT* および *TPKEY-AUTOENCRYPT*)、*IDENTITY-PROOF* は不要です。

戻り値

TPKEYOPEN( ) は正常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に [TPOK] を設定します。また、このキーを表す値が *TPKEYDEF-REC* 内の *KEY-HANDLE* に設定されます。この値は、TPKEYGETINFO( ) などほかの関数で使用されます。

エラー

TPKEYOPEN( ) は異常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に次のいずれかの値を設定します。

[TPEINVAL]
無効な引数が指定されました。たとえば、設定 (フラグ) の値が正しくありません。

[TPEPERM]
パーミッション異常終了。暗号サービス・プロバイダは、与えられた証明情報と現在の環境で、このプリンシパルの秘密鍵にアクセスできませんでした。

[TPESYSTEM]
エラーが発生しました。詳細は、システム・エラーのログ・ファイルを調べてください。

関連項目

TPKEYCLOSE(3cbl)、TPKEYGETINFO(3cbl)、TPKEYSETINFO(3cbl)

TPKEYSETINFO(3cbl)

名前

TPKEYSETINFO() - キー・ハンドルに関連付けるオプション・パラメータを設定する

形式

```
01 TPKEYDEF-REC.
   COPY TPKEYDEF.

01 ATTVALUE-REC.
   COPY user data

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPKEYSETINFO" USING TPKEYDEF-REC TPSTATUS-REC.
```

機能説明

TPKEYSETINFO() は、キー・ハンドルのオプションの属性パラメータを設定します。キー・ハンドルは、特定のプリンシパルのキーおよびそれに関連付けられている情報を表します。

情報を変更するキーは、*TPKEYDEF-REC* 内の *KEY-HANDLE* で識別されます。*KEY-HANDLE* は、事前の TPKEYOPEN() 呼び出しで返されるキー識別子です。

情報を変更する対象の属性は、*TPKEYDEF-REC* 内の *ATTRIBUTE-NAME* で識別されます。属性名には、SPACES または LOW-VALUES を埋め込むことができます。一部の属性は暗号サービス・プロバイダ固有のものですが、TPKEYGETINFO(3cbl) リファレンス・ページで示す属性は、すべてのプロバイダでサポートされる必要があります。

ユーザ定義された ATTVALUE-REC の値は、*ATTRIBUTE-NAME* に関連付けられます。TPKEYSETINFO() が正常に終了すると、ATTVALUE-REC の情報が暗号サービス・プロバイダが定義する方法で格納または処理されます。ATTVALUE-REC のデータの内容が自己記述型の場合、*TPKEYDEF-REC* 内の *ATTRIBUTE-VALUE-LEN* は無視されます (0 でかまいません)。それ以外の場合、*ATTRIBUTE-VALUE-LEN* には、ATTVALUE-REC のデータ長を指定する必要があります。

戻り値

TPKEYSETINFO() は正常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に [TPOK] を設定します。

エラー

TPKEYSETINFO() は異常終了時には、*TPSTATUS-REC* 内の *TP-STATUS* に次のいずれかの値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。たとえば、*KEY-HANDLE* が正しく設定されていません。

[TPESYSTEM]
エラーが発生しました。詳細は、システム・エラーのログ・ファイルを調べてください。

[TPELIMIT]
指定された属性値が大きすぎます。

[TPENOENT]
要求された属性は、キーの暗号サービス・プロバイダによって認識されません。

関連項目 TPKEYCLOSE(3cbl)、TPKEYGETINFO(3cbl)、TPKEYOPEN(3cbl)

TPNOTIFY(3cbl)

名前　　TPNOTIFY() - クライアント識別子により通知を送信する

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPNOTIFY" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明

TPNOTIFY() は、サーバが個々のクライアントに任意通知型メッセージを送信できるようにします。

TPSVCDEF-REC 内の CLIENTID には、以前のあるいは現在のサービス呼び出しの *TPSVCDEF-REC* から保存されたクライアント識別子が入ります。

DATA-REC は、送信するメッセージであり、*TPTYPE-REC* 内の LEN は、送信する *DATA-REC* の大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードの場合は、LEN は無視されます (0 でかまいません)。*TPTYPE-REC* 内の REC-TYPE が SPACES の場合は、*DATA-REC* および LEN は無視され、要求はデータ部なしで送信されます。

TPNOTIFY() が正常終了した場合、メッセージはシステムに渡され、指定されたクライアントに送信されます。TPACK() が設定されていた場合、正常終了は、クライアントがメッセージを受信したことを意味します。さらに、クライアントが任意通知型のメッセージ・ハンドラに登録している場合は、ハンドラが呼び出されます。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPNOBLOCK
: この要求は、ブロッキング条件が存在する場合 (たとえば、メッセージの送信先である内部バッファがいっぱいの場合など) には、送信されません。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
ブロッキング条件が存在する場合、呼び出し元はその条件が解消されるか、タイムアウト (トランザクションまたはブロッキング) になるまでブロックされます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPACK
この設定は、呼び出し元がクライアントからの承認の待ち時間をブロックすることを示します。TPNOACK( ) または TPACK( ) のいずれかを設定しなければなりません。

TPNOACK
この設定は、呼び出し元がクライアントからの承認の待ち時間をブロックしないことを示します。TPNOACK( ) または TPACK( ) のいずれかを設定しなければなりません。

戻り値　TPNOTIFY( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー　次の条件が発生すると、TPNOTIFY( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
無効な引数が指定されました。

[TPENOENT]
ターゲット・クライアントが存在せず、TPACK( ) が設定されました。

[TPETIME]
ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されたか、TPACK( ) と TPTIME が設定されたが、応答が受信されず、TPTIME が指定されました。

[TPEBLOCK]
呼び出し時にブロッキング条件が検出され、TPNOBLOCK は指定されていませんでした。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPNOTIFY( ) が不正なコンテキストで呼び出されました (たとえば、クライアントから)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

[TPERELEASE]
TPACK( ) が指定され、ターゲットは応答プロトコルをサポートしない以前の BEA Tuxedo システムのリリースのクライアントです。

関連項目　TPBROADCAST(3cbl)、TPCHKUNSOL(3cbl)、TPINITIALIZE(3cbl)、TPSETUNSOL(3cbl)、TPTERM(3cbl)

TPOPEN(3cbl)

名前　　TPOPEN( ) - BEA Tuxedo ATMI のリソース・マネージャのオープン

形式

```
01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPOPEN" USING TPSTATUS-REC.
```

機能説明　　TPOPEN( ) は、呼び出し元がリンクされるリソース・マネージャをオープンします。呼び出し元には、多くとも 1 つのリソース・マネージャしかリンクできません。このルーチンはリソース・マネージャ固有の open( ) 呼び出しの代わりに使用するもので、これによりサービス・ルーチンから、移植性を損なう可能性のある呼び出しをなくすことができます。リソース・マネージャは初期化の内容がそれぞれで異なるため、個々のリソース・マネージャをオープンするために必要な情報をコンフィギュレーション・ファイルに記述します。

リソース・マネージャがすでにオープンされている場合 (すなわち、TPOPEN( ) を 2 回以上呼び出した場合)、何も処理は行われず、正常終了を示すコードが返されます。

戻り値　　TPOPEN( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。リソース・マネージャがオープンできなかった理由に関する詳しい情報は、そのリソース・マネージャに独自の方法で照会することで得ることができます。ただし、リソース・マネージャのエラーの正確な内容を判別するための呼び出しを使用すると、移植性が損なわれます。

エラー　　次の条件が発生すると、TPOPEN( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPERMERR]
: リソース・マネージャを正しくオープンできませんでした。より詳しい理由は、そのリソース・マネージャを独自の方法で調査することで得ることができます。ただし、エラーの正確な性質を判別するための呼び出しを使用すると、移植性が損なわれます。

[TPEPROTO]
: TPOPEN( ) が不正なコンテキストで呼び出されました (たとえば、BEA Tuxedo ATMI のサーバ・グループに参加していないクライアントにより)。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPCLOSE(3cbl)

TPPOST(3cbl)

名前　　TPPOST( ) - イベントを通知する

形式

```
01 TPEVTDEF-REC.
 COPY TPEVTDEF

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPPOST" USING TPEVTDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　呼び出し元は、イベントおよび関連データを通知するために TPPOST( ) を使用します。イベントの名前は *TPEVTDEF-REC* の EVENT-NAME で指定され、ポストされるデータは *DATA-REC* に含まれます。通知されるイベントおよびそのデータは、EVENT-NAME がそのサブスクリプション条件を満たし、*DATA-REC* がそのオプションのフィルタ規則に適合するすべてのサブスクライバに対し、BEA Tuxedo のイベント・ブローカによってディスパッチされます。

EVENT-NAME の文字長は最大 31 文字で、SPACES であってはなりません (スペースで埋めることはできません)。また、ドット (".") は BEA Tuxedo システムによって定義されるイベントの頭文字として使用されるため、EVENT-NAME の最初の文字にはドットを使用することはできません。

DATA-REC はポストされるレコードであり、そのタイプを指定することができます。イベントとともにポストされる *DATA-REC* のデータ長は、*TPTYPE-REC* の LEN で指定されます。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。*DATA-REC* がデータ長を指定する必要のあるタイプのレコードである場合は、LEN を 0 にセットすることはできません (0 にセットした場合はデータはポストされません)。*TPTYPE-REC* の REC-TYPE が SPACES の場合は、*DATA-REC* と LEN は無視され、データをともなわずにイベントだけがポストされます。

TPPOST( ) をトランザクション内で使用する場合、トランザクションの境界は、イベント・ブローカによって通知されるサーバおよび安定記憶域のメッセージ・キュー、またはそのいずれかを含むように拡張されます。トランザクショナル・ポストでは、ポストされたイベントを受け取ったことがポスト元のトランザクションに通知される場合 (サーバやキューなど) と、通知されない場合 (クライアントなど) があります。

ポスト元がトランザクション内にあり、TPTRAN が設定されると、通知されるイベントはトランザクション・モードでイベント・ブローカに渡されます。これは、イベント・ブローカがイベントをポスト元のトランザクションの一部としてディスパッチできるようにするためです。ブローカは、サブスクリプションの実行時に *TPEVTDEF-REC* の TPEVTRAN をセットしたサービス・ルーチンや安定記憶キューのサブスクリプションに対してのみトランザクション・イベント通知を送出します。クライアント通知、およびサブスクリプション時に *TPEVTDEF-REC* の TPEVNOTRAN を設定したサービス・ルーチンや安定記憶域キューのサブスクリプションも、イベント・ブローカによってディパッチされますが、これらはポスト側プロセスのトランザクションの一部としては処理されません。

次に、*TPEVTDEF-REC* の有効な設定の一覧を示します。

TPNOTRAN
: 呼び出し元がトランザクション・モードにあり、このフラグがセットされている場合は、イベントのポストは呼び出し元のトランザクションについては実行されません。このフラグ設定を使用するトランザクション・モードの呼び出し元は、依然としてトランザクション・タイムアウトの対象となります (それ以外はなし)。イベントのポストが正しく実行されなかった場合、呼び出し元のトランザクションには影響はありません。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPTRAN
: 呼び出し元がトランザクション・モードにあり、このフラグがセットされている場合は、イベントのポストは呼び出し元のトランザクションに関して実行されます。呼び出し元がトランザクション・モードにない場合、この設定は無視されます。TPNOTRAN または TPTRAN が設定されていなければなりません。

TPNOREPLY
: TPPOST( ) に対し、イベント・ブローカが EVENT-NAME に対するすべてのサブスクリプションを処理するまで待ってから戻る必要はないことを知らせます。TPNOREPLY を設定すると、TPPOST( ) が正常終了するかどうかにかかわらず、*TPEVTDEF-REC* の EVENT-COUNT は 0 に設定されます。呼び出し元がトランザクション・モードにある場合は、TPTRAN も設定するときには、この設定は使用できません。TPNOREPLY または TPREPLY が設定されていなければなりません。

TPREPLY
: TPPOST( ) に対し、すべてのサブスクリプションが処理されてから戻るよう指示します。TPREPLY を設定すると、ルーチンは正常終了時には [TPOK] を返し、EVENT-NAME に関してイベント・ブローカによってデイスパッチされたイベント通知の数が *TPEVTDEF-REC* の EVENT-COUNT に設定されます。呼び出し元がトランザクション・モードにあり、TPTRAN がセットされている場合は、このフラグを設定する必要があります。TPNOREPLY または TPREPLY が設定されていなければなりません。

TPNOBLOCK
: ブロッキング条件が存在する場合は、イベントはポストされません。このような条件が発生すると、呼び出しは失敗し、TP-STATUS は [TPEBLOCK] に設定されます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。 TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: 関数内部のシステム・コールが信号によって中断されると、そのシステム・コールは再度実行されることはなく、呼び出しは異常終了し、TP-STATUS は [TPGOTSIG] にセットされます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

戻り値

TPPOST( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。また、EVENT-COUNT にはイベント・ブローカが EVENT-NAME に関してディスパッチしたイベント通知 (つまり、そのイベント表現が EVENT-NAME に対して有効であると評価され、かつそのフィルタ規則が *DATA-REC* に対して有効であると評価されたサブスクリプションのためのポスト) の数が入ります。終了時に TP-STATUS が [TPESVCFAIL] に設定された場合、EVENT-COUNT には、イベント・ブローカが EVENT-NAME に関してディスパッチした非トランザクション・イベント通知の数が入ります。

エラー

次の条件が発生すると、TPPOST( ) は異常終了し、TP-STATUS に次のいずれかの値を設定します。(特に記述した場合を除いては、エラーが呼び出し元のトランザクションに影響を及ぼすことはありません)。

[TPEINVAL]
: 無効な引数が指定されました。(EVENT-NAME が SPACES である場合など)。

[TPENOENT]
: BEA Tuxedo のユーザ・イベント・ブローカにアクセスできません。

[TPETRAN]
: 呼び出し元がトランザクション・モードにあり、TPTRAN が設定され、TPPOST( ) がトランザクション伝達をサポートしないイベント・ブローカにコンタクトしました。つまり、TMUSREVT(5) が、トランザクションをサポートする BEA Tuxedo ATMI グループで実行されていません。

[TPETIME]
: タイムアウトが発生しました。呼び出し元がトランザクション・モードにある場合、トランザクション・タイムアウトが発生しており、トランザクションはアボートされます。トランザクション・モードでない場合は、ブロッキング・タイムアウトが発生しており、TPBLOCK と TPTIME が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、以降の作業はいずれも正しく実行されず、[TPETIME] が返されます。

[TPESVCFAIL]
イベント・ブローカが呼び出し元のトランザクションに関して、トランザクション・イベントをサービス・ルーチンまたは安定記憶域キューに通知する際に、エラーが発生しました。呼び出し元の現在のトランザクションは、「アボートのみ」にセットされます。このエラーが返されると、EVENT-COUNT には、イベント・ブローカが EVENT-NAME に関してディスパッチした非トランザクション・イベント通知の数が入ります。トランザクションのポスティングの結果はトランザクション終了時にアボートされるため、カウントされません。なお、トランザクションがタイムアウトしないかぎり、トランザクションが中途終了するまで通信は継続され、呼び出し元のトランザクションのためになされた作業はトランザクションが完了する時点で中途終了します (つまり、以降のやりとりで何らかの結果が得られる場合には、TPNOTRAN を設定しておく必要があります)。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT がセットされていました。

[TPEPROTO]
TPPOST() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　TPSUBSCRIBE(3cbl)、TPUNSUBSCRIBE(3cbl)、EVENTS(5)、TMSYSEVT(5)、TMUSREVT(5)

TPRECV(3cbl)

名前　　TPRECV() - 会話接続でメッセージを受信する

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPRECV" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明

TPRECV() は、別のプログラムからオープン接続を介してデータを受け取るときに使用します。COMM-HANDLE は、データを受け取るオープン接続を指定します。COMM-HANDLE は、TPCONNECT() または TPSVCSTART() から返される通信ハンドルです。*DATA-REC* は、メッセージが読み込まれる場所を指定し、入力時には LEN が *DATA-REC* に移動される最大バイト数を示します。

正常終了の場合、およびいくつかのイベント・タイプの場合、LEN には *DATA-REC* に移動された実際のバイト数が入ります。REC-TYPE および SUB-TYPE には、それぞれデータのタイプおよびサブタイプが入っています。メッセージが *DATA-REC* より大きい場合は、*DATA-REC* にはこのレコードに入るバイト数分のみが入ります。応答の残りは破棄され、TPRECV() は TPTRUNCATE を設定します。

正常終了時に LEN が 0 である場合は、応答にはデータ部がなく、*DATA-REC* は変更されていません。入力時に LEN を 0 にすると、エラーになります。

TPRECV() は、接続の制御をもたないプログラムしか出せません。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPNOCHANGE
: この設定を使用すると、*DATA-REC* のタイプは変更できません。すなわち、受信したメッセージのタイプとサブタイプは、それぞれ REC-TYPE および SUB-TYPE と一致しなければなりません。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPCHANGE
: 受信したメッセージのタイプとサブタイプの両方または片方は、受け取り側が着信レコードのタイプを識別するかぎりは、それぞれ REC-TYPE および SUB-TYPE で指定されているものと異なっていることが可能です。TPNOCHANGE または TPCHANGE が設定されていなければなりません。

TPNOBLOCK
: TPRECV( ) はデータが到着するまで待機します。すでにデータが受信できる状態であると、TPRECV( ) はデータを取り込んで終了します。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK が設定されていて、データが受信できる状態である場合は、呼び出し元は、データが到着するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続きプログラムに対して有効です。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: シグナルが関数内部の受信システム・コールを中断させると、呼び出しが再度出されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

通信ハンドル COMM-HANDLE に対してイベントが存在する場合、TPRECV( ) は終了し、TP-STATUS に TPEEVENT( ) を設定します。イベントのタイプは、TPEVENT( ) に返されます。TPEV-SVCSUCC、TPEV-SVCFAIL および TPEV-SENDONLY イベントとともに、データを受け取ることができます。TPRECV( ) の有効なイベントを次に示します。

TPEV-DISCONIMM
: 会話の従属側が受け取るこのイベントは、その会話の起動元が TPDISCON() により即時切断要求を出したこと、または、会話の起動元がまだオープンしている接続について TPRETURN() または TPCOMMIT() を出したときにエラーが発生したことを示します。このイベントは、通信エラー (サーバ、マシン、ネットワークの障害など) により接続が切断されたときにも起動元またはその従属側に返されます。これは即時切断通知 (つまり、正常ではなく中途の終了) であるため、処理途中のデータは失われます。2 つのプログラムが同じトランザクションに参加していた場合、そのトランザクションは中途終了マークが付けられます。COMM-HANDLE は無効になります。

TPEV-SENDONLY
: 接続の他方の側にあるプログラムは、接続の制御権を放棄しました。このイベントの受信側はデータを送信することはできますが、制御権を放棄するまではデータを受信することはできません。

TPEV-SVCERR
: このイベントは、会話の起動元が受け取るもので、会話の従属側が TPRETURN() を出したことを示します。TPRETURN() によって、サービスが正常に終了できないようなエラーが検出されました。たとえば、不正な引数が TPRETURN() に渡されていたり、TPRETURN() が、そのサービスが別の従属側にオープン接続を持っている最中に呼び出されている可能性があります。このイベントの性質上、アプリケーションが定義したデータや戻りコードは返されません。この接続は切断され、COMM-HANDLE は無効になります。このイベントが、受信側のトランザクションの一部として発生すると、そのトランザクションに中途終了マークが付けられます。

TPEV-SVCFAIL
: このイベントは、会話の起動元が受け取るもので、会話の他方の側の下位サービスがアプリケーションで定義されるとおり正常に終了しなかったことを示します。つまり、TPRETURN() が TPFAIL() または TPEXIT() を設定して呼び出されています。TPRETURN() が呼び出されたときに従属サービスがこの接続の制御権をもっている場合、アプリケーションで定義されている戻り値およびレコードを接続の起動元に渡すことができます。サービス・ルーチンの終了処理の一部として、サーバはこの接続を切断しています。したがって、COMM-HANDLE は無効になります。このイベントが、受信側のトランザクションの一部として発生すると、そのトランザクションに中途終了マークが付けられます。

TPEV-SVCSUCC
: このイベントは、会話の起動元が受け取るもので、会話の他方の側の下位サービスがアプリケーションで定義されるとおり正常に終了したことを示します。つまり、TPRETURN( ) が TPSUCCESS( ) を設定して呼び出されています。サービス・ルーチンの終了処理の一部として、サーバはこの接続を切断しています。このため、COMM-HANDLE は無効になります。受信側がトランザクション・モードにいる場合、そのトランザクションをコミット (それが起動元でもある場合) するか、中途終了して、サーバ (トランザクション・モードである場合) が行った作業内容をコミットあるいは中途終了させます。

戻り値

TPRECV( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。TP-STATUS に [TPEEVENT] が設定され、TPEVENT( ) が TPEV-SVCSUCC または TPEV-SVCFAIL の場合、APPL-RETURN-CODE には、TPRETURN( ) の一部として送信されたアプリケーション定義の値が入ります。着信メッセージの大きさが入力時に LEN に指定されたより大きい場合は、TPTRUNCATE( ) が設定され、LEN 分のデータのみが *DATA-REC* に移動されて残りのデータは破棄されます。

エラー

次の条件が発生すると、TPRECV( ) は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました (たとえば、*TPSVCDEF-REC* が無効である場合など)。

[TPEOTYPE]
: 着信メッセージのタイプとサブタイプが呼び出し元が認識しているものでありません。あるいは、TPNOCHANGE が設定されているが、REC-TYPE および SUB-TYPE が着信メッセージのタイプおよびサブタイプと一致しません。会話が呼び出し元のトランザクションの一部である場合には、着信メッセージが破棄されるので、そのトランザクションに中途終了マークが付けられます。

[TPEBADDESC]
: COMM-HANDLE に無効な通信ハンドルが入っています。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションは「アボートのみ」とマークされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPNOBLOCK と TPNOTIME がいずれも指定されていません。. いずれの場合も、*DATA-REC* は変更されません。トランザクション・タイムアウトが発生した場合、任意の接続上でメッセージの送信あるいは受信を行おうとしても、あるいは新しい接続を開始しようとしても、そのトランザクションが中途終了するまで、TPETIME で正常に実行できません。

[TPEEVENT]
イベントが発生し、そのタイプが TPEVENT() に記録されます。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPRECV() が不正なコンテキストで呼び出されました (たとえば、呼び出しプログラムがデータの送信のみを行えるよう接続が確立されている場合など)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

使用法　TPRETURN() が呼び出されたとき、サーバは、アプリケーション定義の戻り値および型付きレコードを渡すことができます。この戻り値は APPL-RETURN-CODE にあり、レコードは、*DATA-REC* にあります。

関連項目　TPCONNECT(3cbl)、TPDISCON(3cbl)、TPSEND(3cbl)

TPRESUME(3cbl)

名前　　TPRESUME() - グローバル・トランザクションを再開する

形式

```
01 TPTRXDEF-REC.
 COPY TPTRXDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPRESUME" USING TPTRXDEF-REC TPSTATUS-REC.
```

機能説明　　TPRESUME() を使用して、中断されているトランザクションでの作業を再開します。呼び出し元がトランザクションの作業を再開した場合、その作業は TPSUSPEND() で再度停止させるか、あるいはあとで TPCOMMIT() または TPABORT() を利用して完了させる必要があります。

トランザクションの作業を再開する際には、呼び出し元はリンクされたリソース・マネージャが (TPOPEN() を利用して) オープンされていることを確認する必要があります。

TPRESUME() は、TRANID() に含まれるグローバル・トランザクション識別子に関して、呼び出し元をトランザクション・モードに切り替えます。

戻り値　　TPRESUME() は正常終了時には、[TPOK] を設定します。

エラー　　次の条件が発生すると、TPRESUME() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: TRANID() に設定された識別子が、実在しないトランザクション (以前に終了したトランザクションやタイムアウトになったトランザクションを含む) のものであるか、または呼び出し元が再開することを認められていないトランザクションのものです。トランザクションに関する呼び出し元のステータスは変化しません。

[TPEMATCH]
: TRANID() に設定された識別子が示すトランザクションは、既に別のプログラムによって再開されています。トランザクションに関する呼び出し元のステータスは変化しません。

[TPETRAN]
呼び出し元が少なくとも一つ以上のリソース・マネージャでグローバル・トランザクションの外部の作業に関与しているため、BEA Tuxedo システムがグローバル・トランザクションを再開できません。グローバル・トランザクションを再開するためには、あらかじめこれらの作業をすべて終了させる必要があります。ローカル・トランザクションについての呼び出し元の状態は、変更されません。

[TPEPROTO]
TPRESUME( ) が不正なコンテキストで呼び出されました (たとえば、呼び出し元が既にトランザクション・モードになっている)。トランザクション・モードに関する呼び出し元のステータスは変更されません。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

注意事項

XA 準拠のリソース・マネージャがグローバル・トランザクションに含まれるようにするには、そのリソース・マネージャが正常にオープンされている必要があります (詳細は TPOPEN( ) を参照。)

停止されていたトランザクションを再開するプログラムは、そのトランザクションを停止させたプログラムと同じ論理マシン (LMID) 上に存在している必要があります ワークステーション・クライアントの場合は、そのワークステーションが接続されているワークステーション・ハンドラ (WSH) は、そのトランザクションを停止させたワークステーション・クライアントのハンドラと同じ論理マシン上に存在している必要があります。

関連項目

TPABORT(3cbl)、TPCOMMIT(3cbl)、TPOPEN(3cbl)、TPSUSPEND(3cbl)

TPRETURN(3cbl)

名前　　TPRETURN( ) - BEA Tuxedo ATMI のサービス・ルーチンからのリターン

形式

```
01 TPSVCRET-REC.
 COPY TPSVCRET.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

COPY TPRETURN REPLACING TPSVCRET-REC BY TPSVCRET-REC

 TPTYPE-REC BY TPTYPE-REC
 DATA-REC BY DATA-REC T
 PSTATUS-REC BY TPSTATUS-REC.
```

機能説明

TPRETURN( ) は、サービス・ルーチンが完了したことを示します。TPRETURN( ) には EXIT PROGRAM 文が含まれるので、BEA Tuxedo ATMI のディスパッチャに制御を確実に正しく返すために、TPRETURN( ) はそれが起動されたルーチンと同じルーチンの中から呼び出されなければなりません。つまり、TPRETURN( ) をサービス・ルーチンのサブ・プログラムから呼び出してはなりません。制御が BEA Tuxedo ATMI のディスパッチャに返されないからです。

TPRETURN( ) はサービスの応答メッセージを送るときに使用します。応答を受け取るサービスが TPCALL( )、TPGETRPLY( )、または TPRECV( ) で待機している場合、TPRETURN( ) の呼び出しが成功した時点で、受信側のレコードに応答が入ります。

会話サービスの場合、TPRETURN( ) は接続自体も切断します。したがって、サービス・ルーチンは、TPDISCON( ) を直接呼び出すことができません。正しい結果を確実に得るためには、会話サービスに接続しているプログラムは、TPDISCON( ) を呼び出してはなりません。その代わり、会話サービスが完了したことを示す通知を待ってください。つまり、TPEV-SVCSUCC や TPEV-SVCFAIL などのイベントが TPRETURN( ) から送られるのを待ってください。

サービス・ルーチンがトランザクション・モードであった場合、TPRETURN() はトランザクションのサービス部分を、そのトランザクションの完了時点でコミットあるいはアボートできる状態にします。サービスは同じトランザクションの一部として複数回呼び出すことができるので、TPCOMMIT() または TPABORT() がそのトランザクションの実行元によって呼び出されるまでは、完全にコミットあるいは中途終了させる必要は必ずしもありません。

TPRETURN() は、該当サービス・ルーチンが開始した要求 / 応答型サービス要求で期待されるすべての応答を受け取った後で、呼び出すようにしてください。そうでない場合は、サービスの性質に応じて、そのサービス・ルーチンで通信を起動したプログラムに、[TPESVCERR] 状態または TPEV-SVCERR イベントが返されます。受信されていない未終了の応答は、受信後、BEA Tuxedo ATMI のディスパッチャによって自動的に取り除かれます。また、これらの応答に対応する通信ハンドルは無効になります

TPRETURN() は、サービスが開始した接続をすべてクローズしてから呼び出すことも必要です。そうでない場合は、サービスの性質に応じて、そのサービス・ルーチンで通信を起動したプログラムに、[TPESVCERR] 状態または TPEV-SVCERR イベントが返されます。また、即時切断イベント (つまり、TPEV-DISCONIMM) が、オープンしているすべての接続を通して下位サービスに送信されます。

接続の制御に関しては、TPRETURN() 発行時に、サービス・ルーチンがそれが呼び出された接続の制御権を持っていない場合、2 とおりの結果が考えられます。まず、サービス・ルーチンが *TPSVCRET-REC* 内の TP-RETURN-VAL に TPFAIL() を、*TPTYPE-REC* 内の REC-TYPE に SPACES (つまり、送信データなし) を設定して TPRETURN() を呼び出す場合、TPEV-SVCFAIL イベントがこの会話の起動元に送信されます。あるいは、それ以外の設定で TPRETURN() 呼び出しが行われる場合、TPEV-SVCERR イベントが起動元に送信されます。

会話サービスは、自分が開始していないオープン接続を 1 つだけ持っているので、どの通信ハンドルについてデータ (およびイベント) を送信すればいいかをサーバは認識しています。このため、通信ハンドルは TPRETURN() には渡されません。

次に、TPRETURN( ) の引数について説明します。TP-RETURN-VAL は次のいずれかに設定できます。

TPSUCCESS

サービスは正常に終了しました。データが存在する場合、そのデータは送られます (戻り処理の失敗の場合を除き)。呼び出し元がトランザクション・モードの場合、TPRETURN( ) はそのトランザクションの呼び出し元の部分を、トランザクションが最終的にコミットする時点でコミットできるような状態にします。なお、TPRETURN( ) を呼び出しても、必ずしもトランザクション全体が終了することにはつながりません。また、呼び出し元が正常終了を示したとしても、未処理の応答またはオープンされたままの接続がある場合や、該当サービス内で行われた作業が原因でトランザクションがアボートのみになった場合は、異常終了メッセージが送られます。つまり、応答の受信者は TPESVCERR( ) 表示あるいは TPEV-SVCERR イベントを受け取ります。なお、サービス・ルーチンの処理中になんらかの理由でトランザクションがアボートのみになると、TP-RETURN-VAL は TPFAIL( ) に設定されます。TPSUCCESS( ) が会話サービスに対して指定されると、TPEV-SVCSUCC イベントが生成されます。

TPFAIL

アプリケーション上の問題でサービスが異常終了しました。応答を受け取ったプログラムにエラーが報告されます。つまり、応答を受け取る呼び出しは異常終了し、受信側は [TPSVCERR] 指示あるいは TPEV-SVCERR イベントを受け取ります。呼び出し元がトランザクション・モードの場合、TPRETURN( ) はそのトランザクションをアボートのみにします (ただし、トランザクションは既にアボートのみになっていることもあります)。戻り処理が失敗した場合を除き、呼び出し元のデータが送られます (もしあれば)。トランザクション・タイムアウトになって、呼び出し元のデータが送られない場合もあります。このケースでは、応答を待つプログラムは [TPETIME] エラーを受け取ることになります。

TPEXIT

この値は、サービスの完了という点では TPFAIL( ) と同じですが、トランザクションがアボートのみになり応答が要求者に返された後で、サーバは終了します。サーバが再開可能な場合は、自動的に再開します。

TP-RETURN-VAL がこれらの 3 つの値のいずれかに設定されない場合、デフォルトで TPFAIL( ) が使用されます。

アプリケーションが定義した戻り値 *TPSVCRET-REC* 内の APPL-CODE を、サービスの応答を受け取るプログラムに送ることもできます。このコードは、応答を正常に送ることができれば (つまり、受信の呼び出しが正常に行われるか [TPESVCFAIL] が返される、あるいはイベント TPEV-SVCSUCC または TPEV-SVCFAIL のいずれかが受信されれば)、TP-RETURN-VAL の設定に関係なく送られます。APPL-CODE の値は、受信側の変数 *TPSTATUS-REC* 内の APPL-RETURN-CODE に入ります。

DATA-REC は送信されるレコードであり、LEN は、送信する *DATA-REC* の大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプおよびサブタイプのレコードである場合、LEN は無視されます (0 でかまいません)。REC-TYPE が SPACES の場合、*DATA-REC* および LEN は無視されます。このとき、サービスを呼び出したプログラムが応答を期待している場合には、応答はデータ部なしで送信されます。応答を期待していない場合、TPRETURN( ) は渡されたデータをすべて無視し、応答を送信せずに終了します。REC-TYPE が STRING で、LEN が 0 の場合は、要求はデータ部なしで送信されます。

会話サービスの場合、アプリケーションの戻りコードとデータ部が送られないケースがいくつかあります。

- 呼び出しが行われたとき接続が既に終了していた場合 (呼び出し元がその接続で TPEV-DISCONIMM を受け取っていた場合)、この呼び出しは単にサービス・ルーチンを終了させ、現在のトランザクション (もしあれば) をロールバックします。この場合、呼び出し元のデータ・レコードは送信されません。
- 呼び出し元が接続の制御手段を持たない場合は、以下に示すように接続元に TPEV-SVCERR または TPEV-SVCFAIL が送出されます。接続元がどのイベントを受け取った場合でも、データ・レコードは送信されません。ただし、接続元が TPEV_SVCFAIL イベントを受け取った場合は、接続元の *TPSTATUS-REC* の APPL-RETURN-CODE に戻りコードが入ります。

戻り値

TPRETURN( ) には EXIT PROGRAM 文が含まれるので、呼び出し元には値が返されず、制御もサービス・ルーチンには返されません。サービス・ルーチンが TPRETURN( ) を使用せずに終了すると (つまり、EXIT PROGRAM 文を直接使用するか、単純に「サービス・ルーチンを終了する」と)、サーバはサービス・エラーをサービスの要求者に返します。また、従属側へのオープン接続はすべてただちに切断され、未終了の非同期応答は取り除かれます。サーバが異常終了時にトランザクション・モードにあった場合は、そのトランザクションには中途終了マークが付けられます。TPRETURN( ) がサービス・ルーチンとは別に使用される場合 (つまり、サービス・ルーチンでないルーチンによって使用される場合)、TPRETURN( ) は何の影響も与えません。

エラー　TPRETURN はサービス・ルーチンを終了させるので、引数の処理中またはそのルーチンの処理中に検出されたエラーは、そのルーチンの呼び出し元には示されません。このようなエラーが検出されると、TPCALL() または TPGETRPLY() でサービスからの結果を受信するプログラムについては、TP-STATUS に [TPESVCERR] が設定され、TPSEND() または TPRECV() を使用しているプログラムには、その会話について TPEV-SVCERR イベントが送信されます。

関連項目　TPCALL(3cbl)、TPCONNECT(3cbl)、TPFORWAR(3cbl)

TPSCMT(3cbl)

名前　　TPSCMT() - TPCOMMIT の終了時期の設定

形式

```
01 TPCMTDEF-REC.
 COPY TPCMTDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSCMT" USING TPCMTDEF-REC TPSTATUS-REC.
```

機能説明

TPSCMT() は TP-COMMIT-CONTROL 特性を *TPCMTDEF-REC* に指定されている値に設定します。この TP-COMMIT-CONTROL 特性は、制御をその呼び出し元に返す際の TPCOMMIT() の動作に影響します。プログラムがトランザクション・モードにあるかどうかに関係なく、プログラムから TPSCMT() を呼び出すことができます。他のプログラムがコミットしなければならないトランザクションに呼び出し元が参加している場合は、TPSCMT() を呼び出してもそのトランザクションに影響を与えないことに注意してください。むしろ、呼び出し元がコミットするその後のトランザクションに影響を与えます。

ほとんどの場合、トランザクションは、BEA Tuxedo ATMI のプログラムが TPCOMMIT() を呼び出したときにのみコミットされます。ただし、次の場合は例外です。すなわち、UBBCONFIG ファイルの SERVICES セクションで AUTOTRAN 変数が使用可能になっているために、サービスがトランザクション・モードでディスパッチされる場合は、そのトランザクションは TPRETURN() の呼び出し時点で完了します。TPFORWAR() が呼び出された場合、そのトランザクションは、サーバが最終的に TPRETURN() を呼び出すことによって終了します。このため、TPRETURN() を呼び出すサービスにおける TP-COMMIT-CONTROL 特性の設定内容により、TPCOMMIT() がサーバ内からいつ制御を返すかが決まります。TPCOMMIT() がヒューリスティックなエラー・コードを返した場合、サーバはメッセージをログ・ファイルに書き込みます。

クランアントが BEA Tuxedo ATMI アプリケーションに参加する場合は、この特性の初期設定はコンフィギュレーション・ファイルから取られます (UBBCONFIG(5) の RESOURCES セクションの CMTRET 変数を参照)。

次に、*TPCMTDEF-REC* の有効な設定の一覧を示します。

TP-CMT-LOGGED
: この設定は、コミット決定が 2 フェーズ・コミット・プロトコルの第 1 フェーズで記録された後、第 2 フェーズが完了する前に TPCOMMIT( ) を終了させることを示します。この設定により、TPCOMMIT( ) の呼び出し元へのより迅速な応答が得られるようになります。ただし、第 2 段階の処理が完了するまでに生じるタイミングのずれが原因で、トランザクションのパーティシパントが、その作業の完了 (つまり、中途終了) をヒューリスティックに判断してしまうというリスクが生じます。このような場合、TPCOMMIT( ) が終了していないため、呼び出し元にこの状況を知らせる方法がありません (ただし、リソース・マネージャがヒューリスティックな判断を行った場合には、BEA Tuxedo システムによってメッセージがログ・ファイルに書き込まれてはいます)。正常な状態では、第 1 フェーズの間にコミットすることを約束しているパーティシパントは、第 2 フェーズでコミットします。一般に、ネットワークやサイトの障害が原因で問題が生じると、第 2 フェーズでヒューリスティックな判断がなされます。

TP-CMT-COMPLETE
: この設定は、2 フェーズ・コミット・プロトコルが完全に終了した後、TPCOMMIT( ) を終了させることを示します。この設定により、TPCOMMIT( ) は第 2 フェーズのコミット中にヒューリスティックな判断がなされたことを示すことができます。

戻り値

TPSCMT( ) は正常終了時には、TP-STATUS に [TPOK] を設定し、TP-COMMIT-CONTROL 特性の以前の値を返します。

エラー

次の条件が発生すると、TPSCMT( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: *TPCMTDEF-REC* が TP-CMT-LOGGED または TP-CMT-COMPLETE でありません。

[TPEPROTO]
: TPSCMT( ) の呼び出し方法が不適切です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

注意事項

TPBEGIN()、TPCOMMIT() および TPABORT() を使用して BEA Tuxedo ATMI のトランザクションを記述する際には、XA インターフェイスに準拠した (および呼び出し元に適切にリンクされている) リソース・マネージャが行う作業のみがトランザクションとしての特性を備えていることを忘れないようにすることが重要です。トランザクションで行われるその他の処理内容は、TPCOMMIT() や TPABORT() の影響を受けません。そのリソース・マネージャが行った処理が BEA Tuxedo ATMI のトランザクションの一部となるよう、XA インターフェイスを満たすリソース・マネージャをサーバにリンクする方法については、buildserver(1) を参照してください。

関連項目

TPABORT(3cbl)、TPBEGIN(3cbl)、TPCOMMIT(3cbl)、TPGETLEV(3cbl)

TPSEND(3cbl)

名前　　TPSEND( ) - 会話接続でメッセージを送信するルーチン

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSEND" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　TPSEND( ) は、別のプログラムにオープン接続を介してデータを送信するときに使用します。このとき、呼び出し元がこの接続の制御権をもたなければなりません。COMM-HANDLE は、データを送信するオープン接続を指定するものです。COMM-HANDLE は、TPCONNECT( ) または TPSVCSTART( ) から返される通信ハンドルです。

DATA-REC は、送信するデータであり、LEN は、送信するデータの大きさを指定します。ただし、*DATA-REC* が長さの指定を必要としないタイプのレコードである場合 LEN は無視されます (0 でかまいません)。REC-TYPE が SPACES の場合は、*DATA-REC* および LEN は無視され、メッセージはデータなしで送信されます (これは、たとえば、データの送信なしに接続の制御権だけを与えるときなどに使用されます)。

次に、*TPSVCDEF-REC* の有効な設定の一覧を示します。

TPRECVONLY
: この設定は、呼び出し元のデータが送信された後、呼び出し元が接続の制御を放棄することを指示します。つまり、呼び出し元はそれ以降、TPSEND( ) 呼び出しを発行することはできなくなります。接続の他方の側のプログラムが TPSEND( ) によって送られたデータを受け取る場合、それが接続の制御を得たことを示すイベント (TPEV-SENDONLY) も受け取ります (そして、それ以降、TPRECV( ) 呼び出しは発行できません)。TPRECVONLY または TPSENDONLY のいずれかを指定しなければなりません。

TPSENDONLY
: この設定は、呼び出し元が接続の制御権を保持することを示します。TPRECVONLY または TPSENDONLY のいずれかを指定しなければなりません。

TPNOBLOCK
: ブロッキング条件が存在する場合、データおよびどのイベントも送信されません (たとえば、メッセージの送信に使用されるデータ・バッファがいっぱいのときなど)。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続きプログラムに対して有効です。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: シグナルがルーチン内部のシステム・コールを中断した場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

TPNOSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再開されず、この呼び出しは異常終了します。TPNOSIGRSTRT または TPSIGRSTRT が設定されていなければなりません。

COMM-HANDLE に対してイベントが存在する場合、TPSEND() は呼び出し元のデータを送信せずに終了します。イベントのタイプは、TPEVENT() に返されます。TPSEND() の有効なイベントを次に示します。

TPEV-DISCONIMM
: 会話の従属側が受け取るこのイベントは、その会話の起動元が TPDISCON( ) により即時切断要求を出したことを、あるいは接続の起動元が従属側のオープン接続を持って TPRETURN( ) を出したことを示します。このイベントは、通信エラー (サーバ、マシン、ネットワークの障害など) により接続が切断されたときにも起動元またはその従属側に返されます。

TPEV-SVCFAIL
: 会話の起動元が受け取るこのイベントは、その接続の従属側が会話の制御権をもたずに TPRETURN( ) を出したことを示します。さらに、TPRETURN( ) が TPFAIL( ) を設定した状態でデータなし (つまり、TPRETURN( ) に渡された REC-TYPE に SPACES が設定されている) で発行されています。

TPEV-SVCERR
: 会話の起動元が受け取るこのイベントは、その接続の従属側が会話の制御権をもたずに TPRETURN( ) を出したことを示します。さらに、TPEV-SVCFAIL に記述されている以外の形で、TPRETURN( ) が出されました。

これらのイベントはそれぞれ、即時切断通知 (すなわち、正常ではなく中途で終了) を示すので、処理途中のデータは失われます。この接続に使用された通信ハンドルは無効になります。2 つのプログラムが同じトランザクションに参加していた場合には、そのトランザクションに中途終了マークが付けられます。

戻り値

TPSEND( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。イベントが存在し、エラーが発生しなかった場合は、TPSEND( ) は TP-STATUS に [TPEEVENT] を設定します。TP-STATUS に [TPEEVENT] が設定され、TP-EVENT が TPEV-SVCSUCC または TPEV-SVCFAIL の場合、APPL-RETURN-CODE には、TPRETURN( ) の一部として送信されたアプリケーション定義の値が入ります。

エラー

次の条件が発生すると、TPSEND( ) は異常終了し、TP-STATUS に次の値を設定します。特に説明がなければ、この障害は呼び出し元のトランザクションには影響しません。

[TPEINVAL]
: 無効な引数が指定されました。

[TPEBADDESC]
: COMM-HANDLE に無効な通信ハンドルが入っています。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、そのトランザクションは「アボートのみ」とマークされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPNOBLOCK と TPNOTIME がいずれも指定されていません。いずれの場合も、*DATA-REC* も *TPTYPE-REC* も変更されません。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、接続を使ったメッセージの送受信や新しい接続の開始はできません。これらの操作を行おうとすると、[TPETIME] が発生して失敗します。

[TPEEVENT]
イベントが発生し、そのタイプが TPEVENT() に記録されます。このエラーが発生すると、*DATA-REC* は送られません。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルを受け取りましたが、TPSIGRSTRT が指定されていませんでした。

[TPEPROTO]
TPSEND() が不正なコンテキストで呼び出されました (たとえば、呼び出し元がデータの受信しかできないように接続が確立された場合など)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　TPCONNECT(3cbl)、TPDISCON(3cbl)、TPRECV(3cbl)

TPSETCTXT(3cbl)

名前

TPSETCTXT( ) - 現在のアプリケーション関連のコンテキスト識別子を設定する

形式

```
01 TPCONTEXTDEF-REC.
   COPY TPCONTEXTDEF.

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPSETCTXT" USING TPCONTEXTDEF-REC TPSTATUS-REC.
```

機能説明

TPSETCTXT( ) は、現在のプログラムが動作するコンテキストを定義します。(マルチスレッドの COBOL アプリケーションは現在サポートされていません。) これ以降の BEA Tuxedo の呼び出しは、TPCONTEXTDEF-REC の *CONTEXT* で示されるアプリケーションを参照します。TPCONTEXTDEF-REC の *CONTEXT* の値は、事前の TPGETCTXT( ) 呼び出しで提供されます。*CONTEXT* の値が TPNULLCONTEXT の場合、プログラムはどの BEA Tuxedo コンテキストとも関連しません。入力時に TPCONTEXTDEF-REC の *CONTEXT* に TPINVALIDCONTEXT を設定すると無効になります。

戻り値

TPSETCTXT( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

異常終了時には、TPSETCTXT( ) は呼び出し元プロセスのコンテキストを変更せず、TP-STATUS にエラー条件を示す値を設定します。

エラー

TPSETCTXT( ) は異常終了時には、TP-STATUS に次のいずれかの値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。

[TPENOENT]
: TPCONTEXTDEF-REC の *CONTEXT* の値が有効なコンテキストではありません。

[TPEPROTO]
: TPSETCTXT( ) の呼び出し方法が不適切です。たとえば、TPINITIALIZE( ) を呼び出していないプロセスや TP-MULTI-CONTEXTS を設定せずに TPINITIALIZE( ) が呼び出されたプロセスで TPSETCTXT( ) が呼び出された場合です。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容は、ログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

関連項目　「COBOL アプリケーション・トランザクション・モニタ・インターフェイスの紹介」、TPGETCTXT(3cbl)

TPSETUNSOL(3cbl)

名前　　TPSETUNSOL() - 任意通知型メッセージの処理方法の設定

形式

```
01 CURR-ROUTINE  PIC S9(9) COMP-5.

01 PREV-ROUTINE  PIC S9(9) COMP-5.

01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL "TPSETUNSOL" USING CURR-ROUTINE PREV-ROUTINE TPSTATUS-REC.
```

機能説明

TPSETUNSOL() は、任意通知型メッセージが BEA Tuxedo システムのライブラリによって受け取られる際に呼び出すルーチンをクライアントが指定できるようにします。TPSETUNSOL() の最初の呼び出しの前に、BEA Tuxedo ATMI ライブラリがクライアントのために受け取った任意通知型メッセージは記録されますが、無視されます。関数番号 *CURR-ROUTINE* に 0 を設定して TPSETUNSOL() を呼び出した場合も、同じ結果になります。システムが通知や検出のために使用する方法は、アプリケーションのデフォルトで決まります。このデフォルトは、クライアントごとに変更できます (TPINITIALIZE(3cbl) を参照)。

TPSETUNSOL() の呼び出し時に CURR-ROUTINE に渡されるルーチン番号には、定義済みの 16 のルーチンから 1 つを選択します。ルーチン名は、任意通知型メッセージを処理する C のルーチンの場合は、_tm_dispatch1 から _tm_dispatch8 まで、同じメッセージを処理する COBOL のルーチンの場合は、TMDISPATCH9 から TMDISPATCH16 までででなければなりません。C の関数 (_tm_dispatch1 から _tm_dispatch8 まで) は、tpsetunsol(3c) に記述されているパラメータ定義に準拠していなければなりません。COBOL のルーチン (TMDISPATCH9 から TMDISPATCH16 まで) は、データの受信に TPGETUNSOL() を使用しなければなりません。

C アプリケーションの任意通知型メッセージ処理ルーチン内での処理は、次の BEA Tuxedo 関数に限定されています。tpalloc()、tpfree()、tpgetctxt()、tpgetlev()、tprealloc()、および tptypes()。

COBOL アプリケーションの任意通知型メッセージ処理ルーチン内での処理は、次の BEA Tuxedo 関数に限定されています。TPGETLEV() および TPGETCTXT()。

戻り値

TPSETUNSOL( ) は正常終了時には、TP-STATUS に [TPOK] を設定し、任意通知型メッセージ処理ルーチンの以前の設定条件を返します。PREV-ROUTINE が 0 の場合、以前に任意通知型メッセージ処理ルーチンが設定されていないことを示し、正常終了です。

エラー

次の条件が発生すると、TPSETUNSOL( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
無効な引数が指定されました (たとえば、CURR-ROUTINE が有効なルーチンの値ではない場合など)。

[TPEPROTO]
TPSETUNSOL( ) が不正なコンテキストで呼び出されました (たとえば、サーバ内から)。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

移植性

TPNOTIFY( ) に記述されているインターフェイスはすべて、ネイティブ・サイトの UNIX システムベースのプロセッサ上で利用できます。さらに、ルーチン TPBROADCAST( ) と TPCHKUNSOL( ) およびルーチン TPSETUNSOL( ) は、UNIX および MS-DOS ワークステーション・プロセッサ上で利用できます。

TPSETUNSOL( ) は Windows、OS/2、および RS6000 では利用できません。ダイナミック・リンク・ライブラリと共用ライブラリの動作方法がこれらの環境では異なるためです。これらのプラットフォーム上で呼び出されると、TPEPROTO( ) が返されます。これらの環境では、C 言語のインターフェイス tpsetunsol( ) を使用してハンドラ関数を設定します。

関連項目

TPGETCTXT(3cbl)、TPGETUNSOL(3cbl)、TPINITIALIZE(3cbl)、TPTERM(3cbl)

TPSPRIO(3cbl)

名前

TPSPRIO() - サービス要求の優先順位の設定

形式

```
01 TPPRIDEF-REC.
 COPY TPPRIDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSPRIO" USING TPPRIDEF-REC TPSTATUS-REC.
```

機能説明

TPSPRIO() は、次に送信あるいは転送される要求の優先順位を設定します。設定された優先順位は、次に送信される要求に対してのみ有効です。キュー機能がインストールされている場合、優先順位は、TPENQUEUE() または TPDEQUEUE() によってキューに登録されるメッセージまたはキューから取り出されるメッセージについても設定できます。デフォルトの設定では、*TPPRIDEF-REC* 内の PRIORITY の設定が正か負かにより、サービスのデフォルトの優先順位が最大 100、あるいは最小 1 に上下します。100 が最も高い優先順位です。要求のデフォルトの優先順位は、その要求の送信先となるサービスによって決まります。このデフォルトの設定は、管理時に指定してもよいですし (UBBCONFIG(5) を参照)、システムの省略時値 50 を使用してもかまいません。TPSPRIO() は、TPCONNECT() または TPSEND() を介して送られるメッセージには影響しません。

次に、*TPPRIDEF-REC* の有効な設定の一覧を示します。

TPABSOLUTE
: 次の要求の優先順位は、PRIORITY の絶対値で送信されます。PRIORITY の絶対値は 1 から 100 までの範囲内の数値とします (最も高い優先順位は 100 です)。この範囲外の値を指定すると、デフォルトの値が使用されます。

TPRELATIVE
: 次の要求の優先順位は、PRIORITY の相対値で送信されます。

戻り値

TPSPRIO() は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー

次の条件が発生すると、TPSPRIO() は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: *TPPRIDEF-REC* の設定が無効です。

[TPEPROTO]
: TPSPRIO() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPACALL(3cbl)、TPCALL(3cbl)、TPDEQUEUE(3cbl)、TPENQUEUE(3cbl)、TPGPRIO(3cbl)

TPSUBSCRIBE(3cbl)

名前　　TPSUBSCRIBE( ) - イベントのサブスクライブ

形式

```
01 TPEVTDEF-REC.
 COPY TPEVTDEF.

01 TPQUEDEF-REC.
 COPY TPQUEDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSUBSCRIBE" USING TPEVTDEF-REC TPQUEDEF-REC TPSTATUS-REC.
```

機能説明　　呼び出し元は、*TPEVTDEF-REC* の EVENT-EXPR で指定されたイベントまたはイベント・セットをサブスクライブする際に TPSUBSCRIBE( ) を使用します。サブスクリプションは BEA Tuxedo のイベント・ブローカ、TMUSREVT( ) によって管理され、イベントが TPPOST( ) を介して通知されるときサブスクライバに知らせるために使用されます。それぞれのサブスクリプションには、通知メソッドを指定します。通知メソッドは、クライアント通知、サービス呼び出し、または安定記憶域内のキューへのメッセージ登録の 3 つの内いずれかの形式をとります。通知方法は、サブスクライバのプロセス・タイプおよび *TPEVTDEF-REC* の TPEV-METHOD-FLAG によって決定されます。

サブスクライブするイベントまたはイベント・セットの名前は、正規表現、すなわち *TPEVTDEF-REC* の EVENT-EXPR によって指定されますが、EVENT-EXPR フィールドが SPACES であってはなりません。正規表現は、tpsubscribe(3c) で指定される形式です。たとえば、EVENT-EXPR が "\e\e..*" のとき、呼び出し元はシステムが生成したすべてのイベントをサブスクライブし、EVENT-EXPR が "\e\e.SysServer.*" であれば、システムが生成したサーバ関連のイベントをサブスクライブします。また、EVENT-EXPR が "[A-Z].*" であれば、呼び出し元はアルファベットの A から Z の文字で始まるすべてのユーザ・イベントをサブスクライブし、EVENT-EXPR が ".*(ERR|err).*" であれば、"ERR" または "err" という文字列を含むすべてのユーザ・イベントをサブスクライブします。たとえば、"account_error" や "ERROR_STATE" というイベントはどちらもその対象となります。

TPEVTDEF-REC の EVENT-FILTER は、論理型フィルタ規則を含む文字列で、イベント・ブローカがイベントを通知する前に、有効であることが評価されなければなりません。ポストするイベントを受け取ると、イベント・ブローカはそのイベントのデータにフィルタ規則 (存在する場合) を適用します。データがフィルタ規則のチェックにパスした場合、イベント・ブローカは通知メソッドを呼び出します。データがフィルタ・ルールを通過しない場合は、イベント・ブローカは対応する通知メソッドを呼び出しません。呼び出し元は、異なるフィルタ・ルールを利用して同じイベントを何度でもサブスクライブすることができます。

フィルタ・ルールは、フィルタ・ルールの適用対象となる型付きレコードによって異なります。FML レコードや VIEW レコードの場合、フィルタ規則は文字列であり、それぞれの論理式コンパイラに渡すことができ (Fboolco、Fboolco32、Fvboolco、Fvboolco32(3fml) を参照)、通知されるレコードに対して評価されます (Fboolev、Fboolev32、Fvboolev、Fvboolev32(3fml) を参照)。STRING レコードの場合、フィルタ規則は tpsubscribe(3c) で指定される形式の正規表現です。その他のタイプのレコードについては、それぞれカスタマイズされた独自のフィルタ評価機構が必要になります (カスタマイズされたフィルタ評価機構の追加に関しては、buffer(3c) および typesw(5) を参照)。EVENT-EXPR に該当するフィルタ・ルールが存在しない場合は、EVENT-FILTER は SPACES にセットする必要があります。

サブスクライバが BEA Tuxedo ATMI のクライアント・プロセスであり、*TPEVTDEF-REC* の TPEVNOTIFY が設定されている場合、サブスクライブしたイベントが通知されると、イベント・ブローカはサブスクライバに任意通知型メッセージを送出します。つまり、EVENT-EXPR に対して有効であると評価されるイベント名が通知されると、イベント・ブローカは、EVENT-EXPR に関連するフィルタ規則を使用して、通知されるデータをチェックします。データがフィルタ・ルールによるチェックにパスした場合、あるいはそのイベントに対して適用すべきフィルタ・ルールが存在しない場合は、サブスクライバはポストされたイベントとそのデータ、および任意通知型メッセージを受け取ることになります。任意通知型メッセージを受け取るためには、クライアントは任意通知処理ルーチンを (TPSETUNSOL( ) を利用して) 登録しておく必要があります。BEA Tuxedo ATMI のサーバ・プロセスが TPEVNOTIFY を設定して TPSUBSCRIBE( ) を呼び出すと、TPSUBSCRIBE( ) は異常終了し、*TPSTATUS-REC* の TP-STATUS に [TPEPROTO] が設定されます。

任意通知型メッセージを介してイベント通知を受け取るクライアントは、終了する前に、イベント・ブローカのアクティブなサブスクリプションのリストから、それぞれのサブスクリプションを削除する必要があります (詳細は TPUNSUBSCRIBE( ) を参照)。クライアントは、TPUNSUBSCRIB( ) のワイルドカード・ハンドル、-1 を使用することにより、任意通知メソッドに関連するサブスクリプションを含むあらゆる「非永続的」サブスクリプションを削除することができます。プロセス終了後も存続するサブスクリプションおよびそれらに関連する通知メソッドについては、以下の TPEVPERSIST の説明を参照してください。クライアントがその非永続的サブスクリプションを削除せずに終了した場合、イベント・ブローカはクライアントにアクセスできなくなったことを検出すると、そのクライアントのサブスクリプションを削除します。

TPEVNOTIFY が設定される場合は、TPEVNOTRAN と TPEVNOPERSIST も設定される必要があります。そうでない場合、TPSUBSCRIBE( ) は異常終了し、TP-STATUS に [TPEINVAL] が設定されます。このことは、任意通知型の通知メソッドを持つクライアントによるサブスクリプションは、トランザクショナルでも、持続型でもあり得ないことを意味します。

サブスクライバ (プロセスのタイプとは無関係) が *TPEVTDEF-REC* の TPEVSERVICE( ) を設定すると、イベント通知は *TPEVTDEF-REC* の NAME-1 で指定された BEA Tuxedo ATMI のサービス・ルーチンに送信されます。つまり、EVENT-EXPR に対して有効であると評価されるイベント名が通知されると、イベント・ブローカは、EVENT-EXPR に関連するフィルタ規則を使用して、通知されるデータをチェックします。データがフィルタ・ルールによるチェックにパスした場合、あるいはそのイベントに適用すべきフィルタ・ルールが存在しない場合は、サービス要求はそのイベントに付随するデータとともに NAME-1 に渡されます。NAME-1 には BEA Tuxedo ATMI の有効なあらゆるサービス名をセットすることができ、また、サブスクリプションが行われた時点でアクティブであってもアクティブでなくてもかまいません。イベント・ブローカによって呼び出されたサービス・ルーチンは、応答データなしで戻ります。すなわち、これらのサービス・ルーチンは、TPRETURN( ) を呼び出す際には *TPTYPE-REC* の REC-TYPE を SPACES にセットする必要があります。TPRETURN( ) に渡されるデータは、すべて切り捨てられます。

TPEVTDEF-REC の TPEVTRAN も設定され、TPPOST( ) を呼び出すプロセスがトランザクション・モードの場合、イベント・ブローカは、サブスクライブされるサービス・ルーチンをポスト元のトランザクションの一部として呼び出します。イベント・ブローカ TMUSREVT( ) とサブスクライブされるサービス・ルーチンは両方とも、トランザクションをサポートするサーバ・グループに属している必要があります (詳細は UBBCONFIG(5) を参照)。TPEVNOTRAN が設定されている場合、イベント・ブローカは、サブスクライブされるサービス・ルーチンをポスト元のトランザクションとは別に呼び出します。

サブスクライバ (プロセスのタイプは無関係) が *TPEVTDEF-REC* の TPEVQUEUE( ) を設定すると、イベント通知は *TPEVTDEF-REC* の NAME-1 で指定されたキュー・スペースの、*TPEVTDEF-REC* の NAME-2 で指定されたキューに登録されます。つまり、EVENT-EXPR に対して有効であると評価されるイベント名が通知されると、イベント・ブローカは、EVENT-EXPR に関連するフィルタ規則を使用して、通知されるデータをチェックします。データがフィルタ規則によるチェックにパスした場合、あるいはそのイベントに適用すべきフィルタ規則が存在しない場合、イベント・ブローカは、イベントとともに通知されるあらゆるデータとメッセージを、NAME-1 で指定されたキュー・スペースの NAME-2 で指定されたキューに登録します。キュー・スペースとキューは、BEA Tuxedo ATMI で使用できるあらゆるキュー・スペースおよびキューを指定でき、どちらもサブスクリプションの時点で存在していても存在していなくてもかまいません。

TPQUEDEF-REC には、通知されるイベントをイベント・ブローカがキューに登録する際の手順をさらに詳しく指示するためのオプションを含めることができます。呼び出し元で指定すべきオプションがない場合は、*TPQUEDEF-REC* を "LOW-VALUE" にセットします。一方、指定すべきオプションがある場合は、TPENQUEUE( ) の項の「制御パラメータ」のセクションで説明した方法でオプションを指定します (とくに TPENQUEUE( ) の入力データ制御の有効なフラグ設定について説明した箇所を参照のこと)。

TPEVTDEF-REC の TPEVTRAN も設定され、TPPOST( ) を呼び出すプロセスがトランザクション・モードの場合、イベント・ブローカは、通知されるイベントとそのデータをポスト元のトランザクションの一部としてキューに登録します。イベント・ブローカ TMUSREVT( ) は、トランザクションをサポートするサーバ・グループに属している必要があります (詳細は UBBCONFIG(5) を参照)。TPEVNOTRAN が設定されている場合、イベント・ブローカは、通知されるイベントとそのデータをポスト元のトランザクションとは別にキューに登録します。

デフォルトでは、BEA Tuxedo のイベント・ブローカは、通知先のリソースが使用不可能である場合はサブスクリプションを削除します。たとえば、イベントのサブスクリプションに関連するサービス・ルーチン、またはキュー・スペースとキューにイベント・ブローカがアクセスできない場合などです。*TPEVTDEF-REC* の TPEVPERSIST は、サブスクライバがこのようなエラーに遭遇したあとも (将来リソースが再度使用可能になった場合に備えて) 引き続きサブスクリプションを維持したい場合にセットします。永続的なサブスクリプションは、TPEVSERVICE( ) および TPEVQUEUE( ) の通知メソッドでしか利用できません。TPEVNOTIFY が設定されるときに TPEVPERSIST を使用することはできません。使用した場合、関数は異常終了し、TP-STATUS に [TPEINVAL] が設定されます。TPEVNOPERSIST を設定した場合、サブスクリプションで指定されたクライアント、サービス名、またはキュー・スペースとキュー名にアクセスする際にエラーが発生すると、イベント・ブローカはこのサブスクリプションを削除します。

TPEVPERSIST と TPEVTRAN がともに設定されて、イベント通知時にリソースが使用できなかった場合、イベント・ブローカはポスト元に戻り、トランザクションがアボートされるようにします。すなわち、イベントのサブスクリプションが保持されている場合でも、リソースが使用できなければポスト元のトランザクションは異常終了します。

TPSUBSCRIBE( ) で要求されるサブスクリプションと一致するものがイベント・ブローカのアクティブなサブスクリプションのリストに既に存在する場合、関数は異常終了し、TP-STATUS に [TPEMATCH] が設定されます。サブスクリプションが既存のサブスクリプションと一致すると判断されるには、EVENT-EXPR と EVENT-FILTER の両方が、イベント・ブローカのアクティブなサブスクリプションのリストに既に存在するサブスクリプションのものと一致することが条件です。また、通知メソッドによっては、サブスクリプションの一致を判断する際にこれ以外の条件も使用されます。

TPEVNOTIFY がセットされていると、呼び出し元の CLIENTID (システムが定義したクライアント識別子) もサブスクリプションの一致を検出する際の判断材料に加えられます。つまり、EVENT-EXPR、EVENT-FILTER、および呼び出し元の CLIENTID がイベント・ブローカのリストに既に存在するサブスクリプションのデータと一致する場合、TPSUBSCRIBE( ) は異常終了します。

TPEVSERVICE( ) が設定されている場合は、EVENT-EXPR、EVENT-FILTER、および NAME-1 で指定されるサービス名がイベント・ブローカのリストに既に存在するサブスクリプションのそれと一致すると、TPSUBSCRIBE( ) は異常終了します。

TPEVQUEUE( ) が設定されている場合は、イベント・ブローカはサブスクリプションの一致を判断する際に、EVENT-EXPR や EVENT-FILTER のほか、キュー・スペース、キュー名、および相関識別子を使用します。相関識別子を利用することにより、同じ送信先の、同じイベント式やフィルタ・ルールに対する複数のサブスクリプションを区別することができます。したがって、呼び出し元が TPEVQUEUE( ) と TPQNOCOORID( ) の両方を設定した場合は、EVENT-EXPR、EVENT-FILTER、NAME-1 で指定されるキュー・スペースの名前、および NAME-2 で指定されるキューの名前が、イベント・ブローカのリストに既に存在するサブスクリプション (相関識別子は指定されていない) のデータと一致すると、TPSUBSCRIBE( ) は異常終了します。さらに、TPQCOORID( ) が設定されている場合、EVENT-EXPR、EVENT-FILTER、NAME-1、NAME-2、および *TPQUEDEF-REC* の CORRID がイベント・ブローカのリストに既に存在するサブスクリプション (同じ相関識別子を持つ) のデータと一致すると、TPSUBSCRIBE( ) は異常終了します。

次に、*TPEVTDEF-REC* の設定の一覧を示します。

TPNOBLOCK
: ブロッキング条件が存在すると、サブスクリプションは成立しません。このような条件が発生すると、呼び出しは失敗し、TP-STATUS は [TPEBLOCK] に設定されます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT のいずれかを設定しなければなりません。

TPNOSIGRSTRT
: 関数内部のシステム・コールが信号によって中断されると、そのシステム・コールは再度実行されることはなく、呼び出しは異常終了し、TP-STATUS は [TPGOTSIG] にセットされます。TPNOSIGRSTRT または TPSIGRSTRT のいずれかを設定しなければなりません。

戻り値

TPSUBSCRIBE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。また、TPSUBSCRIBE( ) は、*TPEVTDEF-REC* の SUBSCRIPTION-HANDLE にこのサブスクリプションのハンドルを設定します。SUBSCRIPTION-HANDLE は、イベント・ブローカのアクティブなサブスクリプションのリストからこのサブスクリプションを削除するために TPUNSUBSCRIBE( ) を呼び出す際に利用できます サブスクライバやその他のプロセスは、いずれも返されたハンドルを利用してこのサブスクリプションを削除することができます。

エラー

次の条件が発生すると、TPSUBSCRIBE( ) は異常終了し、TP-STATUS に次のいずれかの値を設定します。(特に記述した場合を除いては、エラーが呼び出し元のトランザクションに影響を及ぼすことはありません)。

[TPEINVAL]
無効な引数が渡されました (EVENT-EXPR が SPACES である場合など)。

[TPENOENT]
BEA Tuxedo のイベント・ブローカにアクセスできません。

[TPELIMIT]
イベント・ブローカの最大サブスクリプション数に達したため、サブスクリプションが異常終了しました。

[TPEMATCH]
イベント・ブローカのリストに既に存在するサブスクリプションと一致するため、サブスクリプションが異常終了しました。

[TPEPERM]
クライアントは、tpsysadm としてアタッチされず、サブスクリプションのアクションは、サービス呼び出しか、メッセージのキューへの登録になります。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、トランザクションはアボートされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、以降の作業はいずれも正しく実行されず、[TPETIME] が返されます。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT がセットされていました。

[TPEPROTO]
TPSUBSCRIBE( ) が不正なコンテキストで呼ばれました。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目　buffer(3c)、tpsubscribe(3c)、TPENQUEUE(3cbl)、TPPOST(3cbl)、TPSETUNSOL(3cbl)、TPUNSUBSCRIBE(3cbl)、Fboolco、Fboolco32、Fvboolco、Fvboolco32(3fml)、Fboolev、Fboolev32、Fvboolev、Fvboolev32(3fml)、EVENTS(5)、EVENT_MIB(5)、TMSYSEVT(5)、TMUSREVT(5)、tuxtypes(5)、typesw(5)、UBBCONFIG(5)

TPSUSPEND(3cbl)

名前　　TPSUSPEND() - グローバル・トランザクションの一時停止

形式

```
01 TPTRXDEF-REC.
 COPY TPTRXDEF.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSUSPEND" USING TPTRXDEF-REC TPSTATUS-REC.
```

機能説明　　TPSUSPEND() は、呼び出し元のプログラムでアクティブなトランザクションを一時停止するために使用されます。TPBEGIN() で始めたトランザクションは、TPSUSPEND() で一時停止できます。中断しているプログラム、あるいはその他のプログラムは、中断しているトランザクション上で TPRESUME() を利用して作業を再開することができます。TPSUSPEND() が復帰すると、呼び出し元はトランザクション・モードではなくなります。ただし、トランザクションが中断されている間は、そのトランザクションに関係のあるリソース (データベース・ロックなど) はすべてアクティブな状態に保たれます。アクティブなトランザクションと同様、中断されたトランザクションにもトランザクションを最初にスタートする際に割り当てたトランザクション・タイムアウトの値が適用されます。

トランザクションを別のプロセスで再開するには、TPSUSPEND() の呼び出し元が明示的に TPBEGIN() を呼び出すことによってトランザクションを起動している必要があります。TPSUSPEND() は、トランザクションの開始元以外のプロセスから呼び出すこともできます (たとえば、トランザクション・モードで要求を受信するサーバ)。後者の場合、TPSUSPEND() の呼び出し元のみ TPRESUME() を呼び出してトランザクションを再開することができます。このような方法が認められているのは、プロセスがトランザクションのスタートを一時中断し、そのトランザクションを終了する前に別のトランザクションをスタートさせ、作業を行えるようにするためです (エラーを記録するトランザクションを実行してから最初のトランザクションをロールバックする場合など)。

TPSUSPEND() は、一時停止するトランザクションの識別子を TRANID に設定します。

正常終了するためには、呼び出し元は TPSUSPEND( ) を実行する前に、サーバとの未終了のコミュニケーションを全て完了していなければなりません。すなわち、呼び出し元は、呼び出し元のトランザクションに関連のある TPACALL( ) で送出した要求に対する応答を、すべて受け取っていなければなりません。また、呼び出し元は、呼び出し元のトランザクションに関連のある会話サービスとの接続をすべてクローズしておくことも必要です (つまり、TPRECV( ) は TPEV-SVCSUCC イベントを返していなければ なりません)。いずれかの規則に従わない場合、TPSUSPEND( ) は異常終了し、呼び出し元の現在のトランザクションは一時停止せず、トランザクション通信ハンドルはすべて有効なままです。呼び出し元のトランザクションとは関連のない通信ハンドルは、TPSUSPEND( ) の結果に関係なく、有効なままです。

戻り値

TPSUSPEND( ) は正常終了時には、[TPOK] を設定します。

エラー

次の条件が発生すると、TPSUSPEND( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEABORT]
: 呼び出し元のアクティブなトランザクションがアボート (異常終了) されました。このトランザクションに関連のある通信ハンドルは、有効ではなくなりました。

[TPEPROTO]
: TPSUSPEND( ) が不正なコンテキストで呼ばれました (たとえば、呼び出し元がトランザクション・モードではない)。トランザクション・モードに関する呼び出し元のステータスは変更されません。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
: オペレーティング・システムのエラーが発生しました。

関連項目

TPACALL(3cbl)、TPBEGIN(3cbl)、TPRECV(3cbl)、TPRESUME(3cbl)

TPSVCSTART(3cbl)

名前　　TPSVCSTART( ) - BEA Tuxedo ATMI のサービスの開始

形式

```
01 TPSVCDEF-REC.
 COPY TPSVCDEF.

01 TPTYPE-REC.
 COPY TPTYPE.

01 DATA-REC.
 COPY User data.

01 TPSTATUS-REC.
 COPY TPSTATUS.

CALL "TPSVCSTART" USING TPSVCDEF-REC TPTYPE-REC DATA-REC TPSTATUS-REC.
```

機能説明　　TPSVCSTART( ) は、サービス・ルーチンをコーディングする際に最初に呼び出す BEA Tuxedo ATMI のルーチンです。実際、サービス・ルーチン内で TPSVCSTART( ) より先にほかの呼び出しを行うと、エラーになります。TPVCSTART( ) は、サービスのパラメータおよびデータを取り出すために使用されます。このルーチンは、ルーチン TPCALL( ) または TPACALL( ) を介して要求を受け取るサービス、およびルーチン TPCONNECT( )、TPSEND( ) および TPRECV( ) を介して通信を行う会話サービスの両方に使用できます。

ルーチン TPCALL( )、TPACALL( ) または TPFORWAR( ) による要求を処理するサービス・ルーチンは、最大 1 つの着信メッセージを受け取り (TPSVCSTART が正常に終了したとき)、最大 1 つの応答を送ります (TPRETURN( ) によってサービス・ルーチンを終了したとき)。

一方、会話サービスは、最大 1 つの着信メッセージとオープン接続の参照手段とともに、接続要求により呼び出されます。TPSVCSTART( ) が正常に終了すると、接続元プログラムまたは会話サービスは、アプリケーションで定義されるとおりにデータの送受信を行うことができます この接続は半二重方式で確立されます。つまり、接続の一方の側は、他方から明示的に制御を渡されるまで、会話を制御することはできません (データを送信できません)。

トランザクションとの関連で言えば、サービス・ルーチンはトランザクション・モードで呼び出されると、1 つのトランザクションにしか参加できません。サービス・ルーチン作成者側から見るかぎり、トランザクションはサービス・ルーチンから返った時点で終了します。サービス・ルーチンは、トランザクション・モードで呼び出されなかった場合、TPBEGIN()、TPCOMMIT() および TPABORT() を使用して必要な回数だけトランザクションを起動できます。ただし、トランザクションの終了には TPRETURN() を使用しません。したがって、サービス・ルーチン内から起動された未終了のトランザクションについて、TPRETURN() を呼び出すと、エラーになります。

DATA-REC は、サービスのデータが読み込まれる場所を指定し、*TPTYPE-REC* 内の LEN は、*DATA-REC* に移動される最大バイト数を示します。TPSVCSTART の正常終了時には、*DATA-REC* に移動された実際のバイト数が LEN に入ります。*TPTYPE-REC* 内の REC-TYPE および SUB-TYPE にはそれぞれデータのタイプおよびサブタイプが入ります。メッセージが *DATA-REC* より大きい場合は、*DATA-REC* にはこのレコードに入るバイト数分のみが入ります。メッセージの残りは破棄され、TPSVCSTART() は TPTRUNCATE() を設定します。

正常終了時に LEN が 0 である場合は、サービスには着信データがなく、*DATA-REC* は変更されていません。入力時に LEN を 0 にすると、エラーになります。

正常終了時には、*TPSVCDEF-REC* 内の SERVICE-NAME は、要求元のプログラムがサービスの起動に使用したサービス名が入ります。

TPSVCSTART() の終了時の *TPSVCDEF-REC* 設定は、次のようになります。

TPREQRSP
: サービスは、TPCALL() または TPACALL() を使用して起動されました。この設定は、TPCONV と相互に排他的です。

TPCONV
: サービスは、TPCONNECT() を使用して起動されました。この会話用の通信ハンドルは、*TPSVCDEF-REC* 内の COMM-HANDLE に入っています。この設定は、TPREQRSP と相互に排他的です。

TPNOTRAN
: サービス・ルーチンはトランザクション・モードにありません。この設定は、TPTRAN と相互に排他的です。

TPTRAN
: サービス・ルーチンはトランザクション・モードにあります。この設定は、TPNOTRAN と相互に排他的です。

TPNOREPLY
: サービス・ルーチンの呼び出し元プログラムは応答を期待していません。この設定は、TPREQRSP が設定されている場合のみ有効です。この設定は、TPREPLY と相互に排他的です。

TPREPLY
: サービス・ルーチンの呼び出し元プログラムは応答を期待しています。この設定は、TPREQRSP が設定されている場合のみ有効です。この設定は、TPNOREPLY と相互に排他的です。

TPSENDONLY
: サービスは、接続を介してデータの送信のみ可能で、接続の他方の側のプログラムはデータの受信しかできないよう呼び出されます。この設定は、TPCONV が設定されている場合のみ有効です。この設定は、TPRECVONLY と相互に排他的です。

TPRECVONLY
: サービスは、接続を介してデータの受信のみ可能で、接続の他方の側のプログラムはデータの送信しかできないよう呼び出されます。この設定は、TPCONV が設定されている場合のみ有効です。この設定は、TPSENDONLY と相互に排他的です。

TPSVCDEF-REC 内の APPKEY は、アプリケーション側で定義した認証サービスが要求クライアント・プログラムに割り当てるアプリケーション・キーに設定します。このキー値は、このサービス・ルーチンの今回の呼び出し中になされたあらゆるサービス要求とともに渡されます。アプリケーション認証サービスを通らないクライアントを起動する場合には、APPKEY の値は -1 になります。このようなクライアントには、保護アプリケーションとの相互運用性を有する以前のリリース・レベルのクライアントがあります。

戻り値

TPSVCSTART( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。着信メッセージの大きさが入力時に LEN に指定されたより大きい場合、TPTRUNCATE( ) が設定され、LEN 分のデータのみが *DATA-REC* に移動されて残りのデータは破棄されます。

エラー

次の条件が発生すると、TPSVCSTART( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: 無効な引数が指定されました。

[TPEPROTO]
: TPSVCSTART( ) が不正なコンテキストで呼ばれました。.

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 buildserver(1)、TPBEGIN(3cbl)、TPCALL(3cbl)、TPCONNECT(3cbl)、TPINITIALIZE(3cbl)、TPOPEN(3cbl)、TPSVRDONE(3cbl)、TPSVRINIT(3cbl)

TPSVRDONE(3cbl)

名前　　TPSVRDONE( ) - BEA Tuxedo ATMI のサーバ終了ルーチン

形式

```
01 TPSTATUS-REC.
 COPY TPSTATUS.
PROCEDURE DIVISION.
* User code
EXIT PROGRAM.
```

機能説明

BEA Tuxedo ATMI のサーバ用ルーチンは、サービス要求の処理完了後、サーバを終了する前に TPSVRDONE( ) を呼び出します。このルーチンを呼び出した時点では、サーバはまだシステムの一部のままですが、それ独自のサービスは宣言から外されています。このため、このルーチンで、BEA Tuxedo ATMI とのコミュニケーションとトランザクションの定義を行うことができます。ただし、オープンされたままの接続がある場合、保留中の非同期応答がある場合、あるいはまだトランザクション・モードにある場合に TPSVRDONE( ) が終了すると、BEA Tuxedo システムはその接続をクローズし、保留中の応答を無視し、トランザクションをロールバックします。その後で、サーバは終了します。

アプリケーションがサーバにこのルーチンを提供していない場合、BEA Tuxedo システムが提供するデフォルトのバージョンが代わりに呼び出されます。デフォルトの TPSVRDONE( ) は TPCLOSE( ) と USERLOG( ) を呼び出し、サーバが終了することを通知します。

使用法　　TPRETURN( ) または TPFORWAR( ) が TPSVRDONE( ) において呼び出されても、これらのルーチンは何も影響を及ぼさず終了するだけです。

関連項目　　TPCLOSE(3cbl)、TPSVRINIT(3cbl)

TPSVRINIT(3cbl)

名前　TPSVRINIT( ) - BEA TuxedoATMI のサーバ初期化ルーチン

形式

```
LINKAGE SECTION.

01 CMD-LINE.
 05 ARGC  PIC 9(4) COMP-5.
 05 ARGV.
  10 ARGS PIC X OCCURS 0 TO 9999 DEPENDING ON ARGC.
01 TPSTATUS-REC.
 COPY TPSTATUS.
PROCEDURE DIVISION USING CMD-LINE TPSTATUS-REC.
* User code
EXIT PROGRAM
```

機能説明

BEA Tuxedo ATMI のサーバ用ルーチンは、その初期化処理中に TPSVRINIT( ) を呼び出します。このルーチンは、プログラムがサーバになった後、サービス要求を処理する前に呼び出されます。このため、BEA Tuxedo ATMI とのコミュニケーションとトランザクションの定義をこのルーチンで行うことができます。ただし、接続がオープンされているとき、保留中の非同期応答があるとき、あるいはトランザクション・モードにあるときに TPSVRINIT( ) が戻った場合、BEA Tuxedo システムは接続をクローズし、保留中の応答を無視し、トランザクションを中途終了させます。そして、サーバは正常に終了します。

アプリケーションがサーバにこのルーチンを提供していない場合、BEA Tuxedo システムが提供するデフォルトのバージョンが代わりに呼び出されます。デフォルトの TPSVRINIT( ) は、TPOPEN( ) と USERLOG( ) を呼び出して、サーバが正常に起動されたことを通知します。

アプリケーション固有のオプションをサーバに渡し、TPSVRINIT( ) で処理させることができます (servopts(5) 参照)。このオプションは *ARGC* と *ARGV* を使用して渡します。*ARGC* には、渡された引数の数が入り、*ARGV* には引数が入ります (文字形式)。各引数は、1 つの SPACE 文字で区切られます。getopt( ) が BEA Tuxedo システムでは使用されます。

TPSVRINIT( ) が正常終了すると、TP-STATUS に [TPOK] が返され、サービスは要求の受け付けを開始できるようになります。TPSVRINIT でエラーが生じた場合、アプリケーションから TP-STATUS に [TPOK] 以外の値を返して正常にサーバを終了させることができます (サービス要求をとらずに)。

戻り値　　TPRETURN( ) または TPFORWAR( ) のいずれかがサービス・ルーチンとは別に (たとえば、クライアント、TPSVRINIT( )、TPSVRDONE( ) などから) 使用されても、これらのルーチンは何も影響を及ぼさず終了するだけです。

使用法　　TPRETURN( ) または TPFORWAR( ) が TPSVRINIT( ) において呼び出されても、これらのルーチンは何も影響を及ぼさず終了するだけです。

関連項目　　TPOPEN(3cbl)、TPSVRDONE(3cbl)

TPTERM(3cbl)

名前　TPTERM() - アプリケーションを抜け出る

形式

```
01 TPSTATUS-REC.
 COPY TPSTATUS.
CALL "TPTERM" USING TPSTATUS-REC.
```

機能説明　TPTERM() は、BEA Tuxedo ATMI アプリケーションからクライアントを削除します。クライアントがトランザクション・モードであると、トランザクションはロールバックされます。TPTERM() が正常終了すると、呼び出し元は BEA Tuxedo クライアントの操作を行えなくなります。未終了の会話はただちに切断されます。

TPTERM() を 2 回以上呼び出した場合 (つまり、呼び出し元がアプリケーションから抜け出た後で TPTERM() を呼び出した場合)、処理は何も行われず、正常終了を示す値が返されます。

マルチコンテキストに関する留意事項

TPTERM() が呼び出されると、プログラムは TPNULLCONTEXT コンテキストに置かれます。TPNULLCONTEXT コンテキストのプログラムによって呼び出される ATMI 関数のほとんどは、暗黙的 TPINITIALIZE() を実行します。TPINITIALIZE() の呼び出しが正常終了するかどうかは、通常の決定要因に依存し、コンテキスト固有の事柄とは無関係です。

戻り値　TPTERM() は正常終了時には、TP-STATUS に [TPOK] を設定します。マルチコンテキストのアプリケーションで正常終了すると、アプリケーションの現在のコンテキストは TPNULLCONTEXT に変更されます。その後、TPSETCTXT() を使用してコンテキストを要求に応じて変更するのは、ユーザの責任です。

異常終了時には、TPTERM() は -1 を返し、TP-STATUS にエラー条件を示す値を設定します。

エラー　TPTERM() は異常終了時には、TP-STATUS を次のいずれかの値に設定します。

[TPEPROTO]
: TPTERM() が不正なコンテキストで呼び出されました (たとえば、呼び出し元がサーバである場合)。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPINITIALIZE(3cbl)

TPUNADVERTISE(3cbl)

名前　TPUNADVERTISE( ) - サービス名の宣言を取り消すルーチン

形式

```
01 SVC-NAME PIC X(15).
01 TPSTATUS-REC.
 COPY TPSTATUS.
CALL "TPUNADVERTISE" USING SVC-NAME TPSTATUS-REC.
```

機能説明

TPUNADVERTISE( ) を使用すると、サーバは、宣言したサービスの宣言解除を行うことができます。デフォルトの設定では、サーバのサービスは、サーバのブート時に宣言され、サーバのシャットダウン時にその宣言が解除されます。

複数サーバ単一キュー (MSSQ) セットに属するすべてのサーバは、同じサービス・セットを提供しなければなりません。これらのルーチンは、MSSQ セットを共有する全サーバを宣言することによってこの規則を適用します。

TPUNADVERTISE( ) は、該当サーバ (または、呼び出し元の MSSQ セットを共有するサーバ・セット) に対して宣言されたサービスとして *SVC-NAME* を削除します。*SVC-NAME* に SPACES を使用することはできません。また、長さは 15 文字までとしてください。(UBBCONFIG(5) の SERVICES セクションを参照)。これ以上の長さの名前でも受け付けられますが、15 文字以降は切り捨てられてしまいます。このため、切り捨てられた名前が他のサービス名と同じにならないよう、注意が必要です。

戻り値　TPUNADVERTISE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。

エラー　次の条件が発生すると、TPUNADVERTISE( ) は異常終了し、TP-STATUS に次の値を設定します。

[TPEINVAL]
: 無効な引数が指定されました (たとえば、*SVC-NAME* が SPACES である場合など)。

[TPENOENT]
: *SVC-NAME* がサーバによって宣言されていない場合。

[TPEPROTO]
: TPUNADVERTISE( ) が不正なコンテキストで呼び出された場合 (たとえば、クライアントによって)。

[TPESYSTEM]
: BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPADVERTISE(3cbl)

TPUNSUBSCRIBE(3cbl)

名前　　TPUNSUBSCRIBE( ) - イベントのサブスクリプションの削除

形式

```
01 TPEVTDEF-REC.
 COPY TPEVTDEF.
01 TPSTATUS-REC.
 COPY TPSTATUS.
CALL "TPUNSUBSCRIBE" USING TPEVTDEF-REC TPSTATUS-REC.
```

機能説明

呼び出し元は TPUNSUBSCRIBE( ) を使用して、BEA Tuxedo のイベント・ブローカのアクティブなサブスクリプションのリストから、イベントのサブスクリプションまたはイベント・サブスクリプションのセットを削除します。*TPEVTDEF-REC* の SUBSCRIPTION-HANDLE は、TPSUBSCRIBE( ) から返されたイベントのサブスクリプション・ハンドルです。SUBSCRIPTION-HANDLE をワイルドカード値 -1 に設定すると、呼び出し元プロセスが以前に行った非永続的サブスクリプションをすべて削除するよう TPUNSUBSCRIBE( ) に指示されます。非持続タイプのサブスクリプションとは、TPSUBSCRIBE( ) を呼び出す際に TPEVNOPERSIST をセットした状態で行ったサブスクリプションを指します。持続タイプのサブスクリプションは、TPSUBSCRIBE( ) から返されたハンドルを使用することによってのみ削除できます。

"-1" ハンドルでは、呼び出し元の直前のインスタンスによってなされたサブスクリプションではなく、この機能を呼び出しているプロセスによってなされたサブスクリプションだけが削除される点に注意する必要があります (たとえば、いったんダウンして再起動したサーバは、最初のサーバによってなされたサブスクリプションをワイルドカードを利用して削除することはできません)。

次に、*TPEVTDEF-REC* の有効な設定の一覧を示します。

TPNOBLOCK
: ブロッキング条件が存在する場合は、サブスクリプションは削除されません。このような条件が発生すると、呼び出しは失敗し、TP-STATUS は [TPEBLOCK] に設定されます。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPBLOCK
: TPBLOCK がセットされ、ブロッキング条件が存在する場合は、呼び出し元はブロッキング条件が消失するか、またはタイムアウト (トランザクション・タイムアウト、またはブロッキング・タイムアウト) が発生するまでブロックします。TPNOBLOCK または TPBLOCK が設定されていなければなりません。

TPNOTIME
: この設定は、呼び出し元が無制限にブロックでき、ブロッキング・タイムアウトの影響を受けないようにすることを指定します。トランザクション・タイムアウトは引き続き発生する可能性があります。TPNOTIME または TPTIME が設定されていなければなりません。

TPTIME
: このフラグは、ブロッキング条件が存在し、ブロッキング時間に達すると、呼び出し元がブロッキング・タイムアウトを受け取ることを示します。TPNOTIME または TPTIME が設定されていなければなりません。

TPSIGRSTRT
: ルーチン内部のシステム・コールがシグナルによって中断された場合、中断されたシステム・コールは再発行されます。TPNOSIGRSTRT または TPSIGRSTRT のいずれかを設定しなければなりません。

TPNOSIGRSTRT
: 関数内部のシステム・コールが信号によって中断されると、そのシステム・コールは再度実行されることはなく、呼び出しは異常終了し、TP-STATUS は [TPGOTSIG] にセットされます。TPNOSIGRSTRT または TPSIGRSTRT のいずれかを設定しなければなりません。

戻り値

TPUNSUBSCRIBE( ) は正常終了時には、TP-STATUS に [TPOK] を設定します。また、TPUNSUBSCRIBE( ) は、*TPEVTDEF-REC* の EVENT-COUNT に、イベント・ブローカのアクティブなサブスクリプションのリストから削除したサブスクリプションの数を設定します (値は 0 以上)。EVENT-COUNT に 1 より大きい値がセットされるのは、ワイルドカードの "-1" というハンドルを使用した場合に限られます。また、EVENT-COUNT には、TPUNSUBSCRIBE( ) が異常終了した場合でも 0 より大きい値が設定されることがあります。つまり、ワイルドカード・ハンドルを使用した場合、イベント・ブローカがいくつかのサブスクリプションを正しく削除した後でエラーが発生した可能性があります。

エラー

次の条件が発生すると、TPUNSUBSCRIBE( ) は異常終了し、TP-STATUS に次のいずれかの値を設定します。(特に記述した場合を除いては、エラーが呼び出し元のトランザクションに影響を及ぼすことはありません)。

[TPEINVAL]
: 無効な引数が渡されました (SUBSCRIPTION-HANDLE が無効なサブスクリプション・ハンドルである場合など)。

[TPENOENT]
: BEA Tuxedo イベント・ブローカにアクセスできません。

[TPETIME]
タイムアウトが発生しました。呼び出し元がトランザクション・モードの場合は、トランザクション・タイムアウトが発生し、トランザクションはアボートされます。トランザクション・モードでない場合、ブロッキング・タイムアウトが発生し、TPBLOCK と TPTIME の両方が指定されていました。トランザクション・タイムアウトが発生すると、トランザクションがアボートされない限り、以降の作業はいずれも正しく実行されず、[TPETIME] が返されます。

[TPEBLOCK]
ブロッキング条件が存在し、TPNOBLOCK が指定されていました。

[TPGOTSIG]
シグナルが受信され、TPNOSIGRSTRT がセットされていました。

[TPEPROTO]
TPUNSUBSCRIBE() の呼び出し方法が不適切です。

[TPESYSTEM]
BEA Tuxedo システムのエラーが発生しました。エラーの正確な内容はログ・ファイルに書き込まれます。

[TPEOS]
オペレーティング・システムのエラーが発生しました。

関連項目 TPPOST(3cbl)、TPSUBSCRIBE(3cbl)、EVENTS(5)、EVENT_MIB(5)、TMSYSEVT(5)、TMUSREVT(5)

TXBEGIN(3cbl)

名前　TXBEGIN( ) - グローバル・トランザクションの開始

形式

```
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
CALL "TXBEGIN" USING TX-RETURN-STATUS.
```

機能説明　TXBEGIN( ) は、呼び出し元の制御スレッドをトランザクション・モードにする際に使用します。呼び出し元のスレッドは、トランザクションを開始する前に、リンクされているリソース・マネージャがオープンされている (TXOPEN( ) を介して) ことを、まず第 1 に確実にしなければなりません。呼び出し元が既にトランザクション・モードにある場合、または TXOPEN( ) が呼び出されていない場合、TXBEGIN は異常終了します (TX-STATUS に値 [TX-PROTOCOL-ERROR] が設定されます)。

トランザクション・モードに入ると、呼び出し元のスレッドは、現在のトランサクションを完了させるために、TXCOMMIT( ) または TXROLLBACK( ) を呼び出さなければなりません。トランザクションの連鎖に関連する条件によっては、トランザクションを開始する際に TXBEGIN( ) を明示的に呼び出す必要がないこともあります。詳細は TXCOMMIT( ) および TXROLLBACK( ) を参照してください。戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

選択可能なセットアップ　TXSETTIMEOUT( )

戻り値　TXBEGIN( ) は正常終了時には、負数でない値 TX-OK を返します。

エラー　次の条件が発生すると、TXBEGIN( ) は異常終了し、次のいずれかの負の値を返します。

[TX-OUTSIDE]
: 呼び出し元の制御スレッドが、1 つ以上のリソース・マネージャを利用して、グローバル・トランザクションの外部で現在作業中であるため、トランザクション・マネージャは、グローバル・トランザクションを開始できません。このような作業がすべて完了してからでなければ、グローバル・トランザクションは開始できません。ローカル・トランザクションについての呼び出し元の状態は、変更されません。

[TX-PROTOCOL-ERROR]
: TXBEGIN が不正なコンテキストで呼び出されました (たとえば、呼び出し元がすでにトランザクション・モードにある場合)。トランザクション・モードに関する呼び出し元のステータスは変更されません。

[TX-ERROR]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、新しいトランザクションの開始において一時的エラーを検出しました。このエラーが返される場合は、呼び出し元はトランザクション・モードにありません。エラーの正確な内容はログ・ファイルに書き込まれます。

[TX-FAIL]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、致命的エラーを検出しました。このエラーでは、トランザクション・マネージャまたはいずれかのリソース・マネージャ、あるいはその両方は、アプリケーションのために作業を行うことができなくなります。このエラーが返される場合は、呼び出し元はトランザクション・モードにありません。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目　TXCOMMIT(3cbl)、TXOPEN(3cbl)、TXROLLBACK(3cbl)、TXSETTIMEOUT(3cbl)

注意事項　XA 準拠のリソース・マネージャがグローバル・トランザクションに含まれるようにするには、そのリソース・マネージャが正常にオープンされている必要があります (詳細は、TXOPEN を参照)。

TXCLOSE(3cbl)

名前

TXCLOSE( ) - リソース・マネージャ・セットのクローズ

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
PROCEDURE DIVISION.
CALL "TXCLOSE" USING TX-RETURN-STATUS.
```

機能説明

TXCLOSE( ) は、移植生の高い方法でリソース・マネージャ・セットをクローズします。これにより、トランザクション・マネージャが呼び出されて、リソース・マネージャ固有の情報がトランザクション・マネージャ固有の方法で読み取られ、この情報が呼び出し元にリンクされたリソース・マネージャに渡されます。

TXCLOSE( ) は、呼び出し元にリンクされているリソース・マネージャをすべてクローズします。この関数は、リソース・マネージャ固有の「クローズ」呼び出しの代わりに使用されるので、アプリケーション・プログラムは、移植性を損なう可能性のある呼び出しを使用することがなくなります。リソース・マネージャは終了の内容がそれぞれで異なるため、個々のリソース・マネージャを「クローズ」するために必要な情報をリソース・マネージャごとに通知しなければなりません。

TXCLOSE( ) は、アプリケーションの制御スレッドがグローバル・トランザクションに関与する必要がなくなったときに呼び出してください。呼び出し元がトランザクション・モードにあると、TXCLOSE( ) は異常終了します ([TX-PROTOCOL-ERROR] が返されます)。したがって、現在のトランザクションに関与しない制御スレッドがあっても、リソース・マネージャは一切クローズされません。

TXCLOSE( ) が正常終了すると (TX-OK)、呼び出し元のスレッドにリンクしているリソース・マネージャはすべてクローズされます。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値

TXCLOSE( ) は正常終了時には、負数でない値 TX-OK を返します。

エラー

次の条件が発生すると、TXCLOSE( ) は異常終了し、次のいずれかの負の値を返します。

[TX-PROTOCOL-ERROR]
: この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がトランザクション・モードにある場合)。リソース・マネージャは一切クローズされません。

[TX-ERROR]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、一時的エラーを検出しました。エラーの正確な内容はログ・ファイルに書き込まれます。クローズ可能なリソース・マネージャはすべてクローズされます。

[TX-FAIL]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、致命的エラーを検出しました。このエラーでは、トランザクション・マネージャまたはいずれかのリソース・マネージャ、あるいはその両方は、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目　TXOPEN(3cbl)

TXCOMMIT(3cbl)

名前

TXCOMMIT() - トランザクションのコミット

形式

```
DATA DIVISION.
* Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
PROCEDURE DIVISION.
CALL "TXCOMMIT" USING TX-RETURN-STATUS.
```

機能説明

TXCOMMIT() は、呼び出し元の制御スレッドでアクティブなトランザクションの作業をコミットします。

transaction_control 特性 (TXSETTRANCTL() を参照) が TX-UNCHAINED の場合、TXCOMMIT() が終了すると、呼び出し元はトランザクション・モードでなくなります。一方、*transaction_control* 特性が TX-CHAINED の場合は、TXCOMMIT() が終了したとき、呼び出し元は新しいトランザクションのためにトランザクション・モードのままになります (以下の RETURN VALUE と ERRORS の項を参照してください)。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

選択可能なセットアップ

- TXSETCOMMITRET()
- TXSETTRANCTL()
- TXSETTIMEOUT()

戻り値

TXCOMMIT() は正常終了時には、負数でない値 TX-OK を返します。

エラー

次の条件が発生すると、TXCOMMIT() は異常終了し、次のいずれかの負の値を返します。

[TX-NO-BEGIN]
: 現在のトランザクションは、正常にコミットしました。ただし、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-ROLLBACK]
: 現在のトランザクションはコミットできず、ロールバックされました。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます。

[TX-ROLLBACK-NO-BEGIN]
現在のトランザクションはコミットできず、ロールバックされました。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-MIXED]
トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされました。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます。

[TX-MIXED-NO-BEGIN]
トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされました。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-HAZARD]
障害が原因で、トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされた可能性があります。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます

[TX-HAZARD-NO-BEGIN]
障害が原因で、トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされた可能性があります。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-PROTOCOL-ERROR]
この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がトランザクション・モードにない場合)。トランザクション・モードについての呼び出し元の状態は、変更されません。

[TX-FAIL]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、致命的エラーを検出しました。このエラーでは、トランザクション・マネージャまたはいずれかのリソース・マネージャ、あるいはその両方は、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。トランザクションについての呼び出し元の状態は、不明です。

関連項目　TXBEGIN(3cbl)、TXSETCOMMITRET(3cbl)、TXSETTIMEOUT(3cbl)、TXSETTRANCTL(3cbl)

TXINFORM(3cbl)

名前

TXINFORM() - グローバル・トランザクション情報を返す

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
01 TX-INFO-AREA.
 COPY TXINFDEF.
PROCEDURE DIVISION.
CALL "TXINFORM" USING TX-INFO-AREA, TX-RETURN-STATUS.
```

機能説明

TXINFORM() は、グローバル・トランザクション情報を *TX-INFO-AREA* に返します。また、この関数は、呼び出し元がトランザクション・モードにあるかどうかを示す値も返します。

TXINFORM() は、*TX-INFO-AREA* レコードにグローバル・トランザクション情報を入れます。*TX-INFO-AREA* レコードの内容は INTRO() で説明しています。

TXINFORM がトランザクション・モードにおいて呼び出されると、TX-IN-TRAN が設定され、*XID-REC* に現在のトランザクションのブランチ識別子が、TRANSACTION-STATE に現在のトランザクションの状態が入ります。呼び出し元がトランザクション・モードにない場合は、TX-NOT-IN-TRAN が設定され、*XID-REC* にヌル XID が (詳細は、TXINTRO を参照) 入ります。呼び出し元がトランザクション・モードにあるかどうかに関係なく、*COMMIT-RETURN*、*TRANSACTION-CONTROL* および *TRANSACTION-TIMEOUT* に *commit_return* および *transaction_control* 特性の現在の設定、および秒単位のトランザクション・タイムアウト値が入ります。

返されるトランザクション・タイムアウト値は、次のトランザクション開始時に使用される設定を反映しています。したがって、この値は、呼び出し元の現在のグローバル・トランザクションのタイムアウト値を反映しているわけではありません。現在のトランザクション開始後に行われた TXSETTIMEOUT() の呼び出しによって、この値が変更されていることがあるからです。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値

TXINFORM() は正常終了時には、負数でない値 TX-OK を返します。

エラー

次の条件が発生すると、TXINFORM() は異常終了し、次のいずれかの負の値を返します。

[TX-PROTOCOL-ERROR]
この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がまだ TXOPEN() を呼び出していない場合)。

[TX-FAIL]
トランザクション・マネージャが致命的エラーを検出しました。このエラーでは、トランザクション・マネージャは、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目

TXOPEN(3cbl)、TXSETCOMMITRET(3cbl)、TXSETTIMEOUT(3cbl)、TXSETTRANCTL(3cbl)

注意事項

同一のグローバル・トランザクション内では、TXINFORM の後の呼び出しは、XID に同一の *gtrid* 構成要素を指定することが保証されますが、同一の *bqual* 構成要素の指定は必ずしも保証されません。

TXOPEN(3cbl)

名前　TXOPEN() - リソース・マネージャ・セットのオープン

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
PROCEDURE DIVISION.
CALL "TXOPEN" USING TX-RETURN-STATUS.
```

機能説明

TXOPEN() は、移植生の高い方法でリソース・マネージャ・セットをオープンします。これにより、トランザクション・マネージャが呼び出されて、リソース・マネージャ固有の情報がトランザクション・マネージャ固有の方法で読み取られ、この情報が呼び出し元にリンクされたリソース・マネージャに渡されます。

TXOPEN() は、アプリケーションにリンクされたすべてのリソース・マネージャをオープンしようとします。この関数は、リソース・マネージャ固有の「オープン」呼び出しの代わりに使用されるので、アプリケーション・プログラムは、移植性を損なう可能性のある呼び出しを使用することがなくなります。リソース・マネージャは開始の内容がそれぞれで異なるため、個々のリソース・マネージャを「オープン」するために必要な情報をリソース・マネージャごとに通知しなければなりません。

TXOPEN() が TX-ERROR を返した場合、リソース・マネージャは一切オープンされません。TXOPEN() が TX-OK を返した場合は、リソース・マネージャの一部または全部がオープンされています。オープンされなかったリソース・マネージャは、アプリケーションによってアクセスされるときに、リソース・マネージャ固有のエラーを返します。TXOPEN() は、制御スレッドがグローバル・トランザクションに関与する前に、正常に終了しなければなりません。

TXOPEN() が正常終了した後で、(TXCLOSE() を呼び出す前に) TXOPEN を呼び出すことは可能です。このような後続の呼び出しは、正常に終了しますが、トランザクション・マネージャは、リソース・マネージャの再オープンは一切行いません。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値　TXOPEN() は正常終了時には、負数でない値 TX-OK を返します。

エラー

次の条件が発生すると、TXOPEN() は異常終了し、次のいずれかの負の値を返します。

[TX-ERROR]
: トランザクション・マネージャまたはいずれかのリソース・マネージャが、一時的エラーを検出しました。リソース・マネージャは一切オープンされません。エラーの正確な内容はログ・ファイルに書き込まれます。

[TX-FAIL]
: トランザクション・マネージャまたはいずれかのリソース・マネージャが、致命的エラーを検出しました。このエラーでは、トランザクション・マネージャまたはいずれかのリソース・マネージャ、あるいはその両方は、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目

TXCLOSE(3cbl)

TXROLLBACK(3cbl)

名前　TXROLLBACK() - トランザクションのロールバック

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
PROCEDURE DIVISION.
CALL "TXROLLBACK" USING TX-RETURN-STATUS.
```

機能説明　TXROLLBACK() は、呼び出し元の制御スレッドでアクティブなトランザクションの作業をロールバックします。

transaction_control 特性 (TXSETTRANCTL() を参照) が TX-UNCHAINED の場合、TXROLLBACK() が終了すると、呼び出し元はトランザクション・モードでなくなります。一方、*transaction_control* 特性が TX-CHAINED の場合は、TXROLLBACK() が終了したとき、呼び出し元は新しいトランザクションのためにトランザクション・モードのままになります (以下の RETURN VALUE と ERRORS の項を参照してください)。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

選択可能なセットアップ
- TXSETTRANCTL()
- TXSETTIMEOUT()

戻り値　TXROLLBACK() は正常終了時には、負数でない値 TX-OK を返します。

エラー　次の条件が発生すると、TXROLLBACK() は異常終了し、次のいずれかの負の値を返します。

[TX-NO-BEGIN]
: 現在のトランザクションはロールバックしました。ただし、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-MIXED]
: トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされました。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます。

[TX-MIXED-NO-BEGIN]
トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされました。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-HAZARD]
障害が原因で、トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされた可能性があります。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます。

[TX-HAZARD-NO-BEGIN]
障害が原因で、トランザクションのために行われた作業は、部分的にコミットされ、部分的にロールバックされた可能性があります。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-COMMITTED]
トランザクションのために行われた作業は、ヒューリスティックにコミットされました。また、*transaction_control* 特性が TX-CHAINED である場合には、新しいトランザクションが開始されます。

[TX-COMMITTED-NO-BEGIN]
トランザクションのために行われた作業は、ヒューリスティックにコミットされました。また、新しいトランザクションを開始できなかったので、呼び出し元はトランザクション・モードではなくなりました。この戻り値は、*transaction_control* 特性が TX-CHAINED である場合のみ発生します。

[TX-PROTOCOL-ERROR]
この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がトランザクション・モードにない場合)。

[TX-FAIL]
トランザクション・マネージャまたはいずれかのリソース・マネージャが、致命的エラーを検出しました。このエラーでは、トランザクション・マネージャまたはいずれかのリソース・マネージャ、あるいはその両方は、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。トランザクションについての呼び出し元の状態は、不明です。

関連項目　TXBEGIN(3cbl)、TXSETTIMEOUT(3cbl)、TXSETTRANCTL(3cbl)

TXSETCOMMITRET(3cbl)

名前　　TXSETCOMMITRET() - commit_return 特性の設定

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
*
01 TX-INFO-AREA.
 COPY TXINFDEF.
PROCEDURE DIVISION.
CALL "TXSETCOMMITRET" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能説明

TXSETCOMMITRET() は、*COMMIT-RETURN* で指定されている値を *commit_return* 特性に設定します。この特性は、TXCOMMIT() が呼び出し元に制御を返す際の動作方法に影響します。TXSETCOMMITRET() は、呼び出し元がトランザクション・モードにあるかどうかに関係なく、呼び出すことが可能です。この設定は、TXSETCOMMITRET() がこの後で呼び出されて変更されるまで、有効です。

この特性の初期設定は TX-COMMIT-COMPLETED です。

COMMIT-RETURN の有効な設定を次に示します。

TX-COMMIT-DECISION-LOGGED
: このフラグは、2 フェーズ・コミット・プロトコルの第 1 フェーズによってコミットの決定が記録された後、第 2 フェーズが終了する前に、TXCOMMIT() が終了することを示します。このフラグを設定すると、TXCOMMIT() の呼び出し元に高速で応答することができます。しかし、トランザクションがヒューリスティックな結果を得る危険があります。この場合、TXCOMMIT() の戻りコードから呼び出し元がこの状況を知ることはできません。正常な状態では、第 1 フェーズの間にコミットすることを約束しているパーティシパントは、第 1 フェーズでコミットします。ただし、ある特殊な条件下においては (たとえば、長時間継続しているネットワークやノードの障害など)、第 2 フェーズで行うことができずにヒューリスティックな結果が生じることがあります。

TX-COMMIT-COMPLETED
: このフラグは、2 フェーズ・コミット・プロトコルが完全に終了した後、TXCOMMIT() が終了することを示します。このフラグを設定すると、TXCOMMIT() の呼び出し元は、トランザクションがヒューリスティックな結果を得たこと、または得ていた可能性があることを戻りモードから知ることができます。

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値　TXSETCOMMITRET() は正常終了時には、負数でない値 TX-OK を返します。

エラー　次の条件が発生すると、TXSETCOMMITRET() は *commit_return* 特性の設定を変更せずに次のいずれかの負の値を返します。

[TX-EINVAL]
COMMIT-RETURN が、TX-COMMIT-DECISION-LOGGED と TX-COMMIT-COMPLETED のいずれでもありません。

[TX-PROTOCOL-ERROR]
この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がまだ TXOPEN() を呼び出していない場合)。

[TX-FAIL]
トランザクション・マネージャが致命的エラーを検出しました。このエラーでは、トランザクション・マネージャは、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目　TXBEGIN(3cbl)、TXCOMMIT(3cbl)、TXINFORM(3cbl)、TXOPEN(3cbl)、TXROLLBACK(3cbl)

TXSETTRANCTL(3cbl)

名前　　TXSETTRANCTL() - *transaction_control* 特性の設定

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
01 TX-INFO-AREA.
 COPY TXINFDEF.
PROCEDURE DIVISION.
CALL "TXSETTRANCTL" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能説明　　TXSETTRANCTL() は、*TRANSACTION-CONTROL* で指定されている値を *transaction_control* 特性に設定します。この特性は、TXCOMMIT() および TXROLLBACK() が、呼び出し元に制御を返す前に新しいトランザクションを開始するかどうかを決定します。TXSETTRANCTL() は、アプリケーション・プログラムがトランザクション・モードであるかどうかに関係なく、呼び出すことができます。この設定は、TXSETTRANCTL() がこの後で呼び出されて変更されるまで、有効です。

この特性の初期設定は TX-UNCHAINED です。

TRANSACTION-CONTROL の有効な設定を次に示します。

TX-UNCHAINED
: このフラグは、TXCOMMIT() および TXROLLBACK() が、呼び出し元に制御を返す前に新しいトランザクションを開始しないことを示します。呼び出し元は、新しいトランザクションを開始するには、TXBEGIN() を発行しなければなりません。

TX-CHAINED
: このフラグは、TXCOMMIT() および TXROLLBACK() が、呼び出し元に制御を返す前に新しいトランザクションを開始することを示します。
戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値　　TXSETTRANCTL() は正常終了時には、負数でない値 TX-OK を返します。

エラー　　次の条件が発生すると、TXSETTRANCTL() は *transaction_control* 特性の設定を変更せずに次のいずれかの負の値を返します。

[TX-EINVAL]
: *TRANSACTION-CONTROL* が、TX-UNCHAINED と TX-CHAINED のいずれでもありません。

[TX-PROTOCOL-ERROR]
この関数が不正なコンテキストで呼び出されました (たとえば、呼び出し元がまだ TXOPEN( ) を呼び出していない場合)。

[TX-FAIL]
トランザクション・マネージャが致命的エラーを検出しました。このエラーでは、トランザクション・マネージャは、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目 TXBEGIN(3cbl)、TXCOMMIT(3cbl)、TXOPEN(3cbl)、TXROLLBACK(3cbl)、TXINFORM(3cbl)

TXSETTIMEOUT(3cbl)

名前　TXSETTIMEOUT() - *transaction_timeout* 特性の設定

形式

```
DATA DIVISION.
 * Include TX definitions.
01 TX-RETURN-STATUS.
 COPY TXSTATUS.
*
01 TX-INFO-AREA.
 COPY TXINFDEF.
PROCEDURE DIVISION.
CALL "TXSETTIMEOUT" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能説明

TXSETTIMEOUT() は、*TRANSACTION-TIMEOUT* で指定されている値を *transaction_timeout* 特性に設定します。この値は、トランザクションがトランザクション・タイムアウトの対象になる前に完了しなければならない制限時間を指定します。この時間は、AP 呼び出し TXBEGIN() と TXCOMMIT() または TXROLLBACK() の間の時間になります。TXSETTIMEOUT() は、呼び出し元がトランザクション・モードであるかどうかに関係なく呼び出すことができます。TXSETTIMEOUT() がトランザクション・モードで呼び出された場合、新しいタイムアウト値は、次のトランザクションまでは無効です。

transaction_timeout の初期値は 0 (タイムアウトなし) です。

TRANSACTION-TIMEOUT は、トランザクションがトランザクション・タイムアウトの対象となるまでの秒数を指定します。この値には、S9(9) COMP-5 にシステムで定義されている最大値までの任意の値を設定できます。*TRANSACTION-TIMEOUT* 値ゼロは、タイムアウト機能をオフにします。.

戻り値に使用されるレコードは、*TX-RETURN-STATUS* です。

戻り値　TXSETTIMEOUT() は正常終了時には、負数でない値 TX-OK を返します。

エラー　次の条件が発生すると、TXSETTIMEOUT() は *transaction_timeout* 特性の設定を変更せずに次のいずれかの負の値を返します。

[TX-EINVAL]
　指定したタイムアウト値が無効です。

[TX-PROTOCOL-ERROR]
　関数の呼び出し方法が不適切です。たとえば、TXOPEN() より前に呼び出されました。

[TX-FAIL]
トランザクション・マネージャが致命的エラーを検出しました。このエラーでは、トランザクション・マネージャは、アプリケーションのために作業を行うことができなくなります。エラーの正確な内容はログ・ファイルに書き込まれます。

関連項目 TXBEGIN(3cbl)、TXCOMMIT(3cbl)、TXINFORM(3cbl)、TXOPEN(3cbl)、TXROLLBACK(3cbl)

USERLOG(3cbl)

名前　USERLOG( ) - BEA Tuxedo ATMI の中央イベント・ログへのメッセージの書き込み

形式

```
01 LOG-REC.
 COPY User data.
01 LOGREC-LEN PIC S9(9) COMP-5.
01 TPSTATUS-REC.
 COPY TPSTATUS.
CALL "USERLOG" USING LOG-REC LOGREC-LEN TPSTATUS-REC.
```

機能説明　USERLOG( ) は、固定出力ファイル (BEA Tuxedo ATMI の中央イベント・ログ) に LOG-REC を書き込みます。

中央のイベント・ログは通常の UNIX システムファイルで、そのパス名は次のように構成されています。

- シェル変数 ULOGPFX が設定されている場合、その値がファイル名の接頭辞として使用されます。ULOGPFX が設定されていない場合は、ULOG が使用されます。この接頭辞は最初に USERLOG( ) が呼び出されたときに判別されます。
- USERLOG( ) が呼び出されるたびに、日付が判別され、月、日、年が mmddyy の形式で接頭辞に連結されてファイルの名前が設定されます。
- プロセスが初めてユーザ・ログに書き込む場合、対応する BEA Tuxedo のバージョンを示す追加のメッセージを書き込みます。

このあと、メッセージがそのファイルに追加されます。この方法の場合、それ以降の日に USERLOG( ) をプロセスを呼び出すと、メッセージは別のファイルに書き込まれます。

- メッセージは、呼び出しプロセスの時刻 (hhmmss)、システム名、プロセス名、呼び出しプロセスのプロセス ID からなるタグと一緒にログ・ファイルに書き込まれます。このタグの末尾にはコロン (:) を付けます。
- BEA Tuxedo システムがログ・ファイルに出すエラー・メッセージの先頭には、一意の識別用文字列が次の形式で付けられます。
  catalog>:number>:
- この文字列は、メッセージ文字列を収めている国際版カタログの名前にメッセージ番号を加えたものです。規則によれば、BEA Tuxedo システムのエラー・メッセージは一度だけ使用されるようになっているため、この文字列はソース・コード内の位置を一意に識別します。
- *format* 指定の最後の文字が改行文字でない場合、USERLOG( ) は改行文字を 1 つ追加します。

- シェル変数 ULOGDEBUG の先頭文字が 1 または y であると、USERLOG() に送られるメッセージは、呼び出しプロセスの標準エラー出力にも書き出されます。
- USERLOG() は、BEA Tuxedo システムが各種のイベントを記録するために使用します。
- この USERLOG の機構は、トランザクション記録機構とは完全に独立しています。

移植性

USERLOG インターフェイスは、UNIX および MS-DOS オペレーティング・システムで利用できます。ログ・メッセージの一部として生成されるシステム名は、MS-DOS システムでは利用できません。このため、MS-DOS システムのシステム名としては、値 PC を使用します。

使用例

変数 ULOGPFX が /application/logs/log に設定されている場合、および USERLOG() の最初の呼び出しが 9/7/90 に行われた場合、作成されるログ・ファイルには /application/logs/log.090790 という名前が付けられます。たとえば、

```
01 LOG-REC PIC X(15) VALUE "UNKNOWN USER".
01 LOGREC-LEN PIC S9(9) VALUES IS 13.
CALL "USERLOG" USING LOG-REC LOGREC-LEN TPSTATUS-REC.
```

上記のような呼び出しが、プロセス ID が 23431 であるプログラムにより UNIX システムが指定した logsys 上で 4:22:14pm に出された場合は、ログ・ファイルには次のような行が書き込まれます。

```
162214.logsys!security.23431: UNKNOWN USER
```

プロセスがトランザクション・モードのときにメッセージが中央イベント・ログに送られた場合、ユーザ・ログ・エントリのタグに追加の要素が加わります。これらの要素は、リテラル gtrid と、それに続く 3 桁の PIC S9(9) COMP-5 の 16 進整数で構成されます。これらの値はグローバル・トランザクションを一意に示し、グローバル・トランザクション識別子と呼ばれます。この識別子は主に管理上の目的で使用されますが、中央イベント・ログでメッセージの先頭に付けられるタグの中に付けられます。前述のメッセージがトランザクション・モードで中央イベント・ログに書き出される場合、結果として得られるログ・エントリは次のようになります。

```
162214.logsys!security.23431: gtrid x2 x24e1b803 x239: UNKNOWN USER
```

シェル変数 ULOGDEBUG の値が y であると、ログ・メッセージはプログラム security の標準エラー出力にも書き出されます。

エラー

USERLOG() は、送信されるメッセージが stdio.h で定義されている BUFSIZ より大きい場合はハングします。

診断

USERLOG() が返す値には、現在のログ・ファイルのオープンや書き込みができないことを示す値も含まれます。ULOGDEBUG が設定されている場合、標準エラー出力への書き込みができない場合は、エラーとは見なされません。

注意事項

アプリケーションで USERLOG メッセージを使用する場合には、アプリケーション・エラーをデバッグするのに有用なものだけを使用するようにしてください。ログが情報であふれてしまうと、本来のエラーを検出するのが難しくなります。