

Tmax XAライブラリ及びXAゲートウェイガイド

Tmax v6.0



Copyright © 2016 TmaxSoft Co., Ltd. All Rights Reserved.

Copyright Notice

Copyright © 2016 TmaxSoft Co., Ltd. All Rights Reserved.

45, Jeongjail-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 13613, South Korea

Restricted Rights Legend

All TmaxSoft Software (Tmax®) and documents are protected by copyright laws and international convention. TmaxSoft software and documents are made available under the terms of the TmaxSoft License Agreement and this document may only be distributed or copied in accordance with the terms of this agreement. No part of this document may be transmitted, copied, deployed, or reproduced in any form or by any means, electronic, mechanical, or optical, without the prior written consent of TmaxSoft Co., Ltd.

Nothing in this software document and agreement constitutes a transfer of intellectual property rights regardless of whether or not such rights are registered) or any rights to TmaxSoft trademarks, logos, or any other brand features. This document is for information purposes only. The company assumes no direct or indirect responsibilities for the contents of this document, and does not guarantee that the information contained in this document satisfies certain legal or commercial conditions. The information contained in this document is subject to change without prior notice due to product upgrades or updates. The company assumes no liability for any errors in this document.

このソフトウェア(Tmax®)マニュアルの内容とプログラムは、日本国の著作権法および国際条約によって保護されています。マニュアルの内容とプログラムは、TmaxSoft Co., Ltd.との使用許諾契約書の下でのみ使用することができ、マニュアルは使用許諾契約で許可されている範囲を除いては、配布または複製することができません。TmaxSoftの書面による事前の承諾を得ることなく、このマニュアルの全部または一部を電子的または機械的な方法を問わず、転送、複製、配布したり、または二次的著作物を作成する等の行為を一切禁じます。

このソフトウェアのマニュアルとプログラムの使用許諾契約は、いかなる場合においても、マニュアル及びプログラムと関連する知的財産権(登録の有無を問わず)を譲渡するものと解釈されず、TmaxSoftのブランド、ロゴ、商標等の使用権限を与えるものではありません。マニュアルは、情報を提供する目的でのみ提供しており、これに伴う契約上の直接的ないしは間接的な責任を負わず、マニュアルの内容は法律上もしくは商業的な特定の条件が満たされることを保証しません。マニュアルの内容は、製品のアップグレード及び修正により、その内容が予告なく変更されることがあり、内容上の誤りがないことを保証しません。

Trademarks

Tmax®, Tmax WebtoB® and JEUS® are registered trademark of TmaxSoft Co., Ltd. Other products, titles or services may be registered trademarks of their respective companies.

Tmax®, Tmax WebtoB®, JEUS® は、TmaxSoft Co., Ltd.の登録商標です。その他、記載されている会社名、製品名などは、各社の商標または登録商標です。

Open Source Software Notice

Some modules or files of this product are subject to the terms of the following licenses. : openssl-0.9.7.m, zlib-1.1.4, expat-2.0.0, net-snmp, DCE1.0, pthread, google-diff-match-patch, libevent, getopt

Detailed Information related to the license can be found in the following directory :
\${INSTALL_PATH}/license/oss_licenses

この製品の一部ファイルまたはモジュールは、openssl-0.9.7.m、zlib-1.1.4、expat-2.0.0、net-snmp、DCE1.0、pthread、google-diff-match-patch、libevent、getoptライセンスを遵守します。

詳細情報については、製品ディレクトリーの\${INSTALL_PATH}/license/oss_licensesに記載されている事項を参照してください。

文書情報

文書名: Tmax XAライブラリ及びXAゲートウェイガイド

発行日: 2016年8月5日

ソフトウェアバージョン: Tmax v6.0

ガイドバージョン: v2.1.1

目次

このガイドについて	ix
第1章 XAライブラリーとゲートウェイ	1
1.1. 概念	1
1.1.1. XAライブラリー	1
1.1.2. XAゲートウェイ	1
1.2. 環境設定	2
1.2.1. XAライブラリー	2
1.2.2. XAゲートウェイ	3
第2章 API	7
2.1. 同期型呼び出し	7
2.1.1. tpcall	7
2.2. 非同期型呼び出し	12
2.2.1. tpacall	12
2.2.2. tpgetrply	16
第3章 サンプル	21
3.1. Tmax XAサービスの呼び出し例	21
3.2. メイクファイル	23
索引	25

図目次

[図 1.1] XAゲートウェイのサービス処理プロセス	1
-----------------------------------	---

このガイドについて

対象読者

本書は、Tmax[®](以下、Tmax)システムで、XAライブラリーとXAゲートウェイを使用してプログラムを開発するユーザーを対象としています。

前提知識

本書は、Tmaxシステムに関する全般的な内容と、Tmaxシステムが提供している各種機能および特性を習得するためのガイドです。

本書を理解するには、事前に下記の内容を把握しておく必要があります。

- ミドルウェアおよびUNIXシステムに関する知識
- Tmaxの基本概念に関する知識
- Java、Cプログラミングに関する知識

制限事項

本書では、XAライブラリーおよびXAゲートウェイに関する内容のみ記述しているため、本書を読む前にTmaxの基本概念について熟知している必要があります。実務における詳細な使用方法および管理、運用に関する内容については、各製品のガイドを参照してください。

参考

Tmaxシステム開発に関する基本的な内容は、『Tmax 運用ガイド』、または『Tmax アプリケーション開発ガイド』を参照してください。Tmaxが提供しているコマンドとC APIに関する説明については、『Tmax リファレンスガイド』を参照してください。

本書の構成

Tmax XAライブラリー及びXAゲートウェイガイドは、計3章で構成されています。

各章の主な内容は下記のとおりです。

- 第1章: XAライブラリーとゲートウェイ

XAライブラリーとXAゲートウェイの概念および環境設定について記述します。

- 第2章: API

XAライブラリーとXAゲートウェイを使用するための基本的なAPIの使用方法および環境設定方法について記述します。

- 第3章: サンプル

Tmax XAサービスを呼び出す簡単なサンプルを例示します。

表記上の規則

表記	意味
<AaBbCc123>	プログラム・ソースコードのファイル名、ディレクトリー
<Ctrl>+C	CtrlキーとCキーを同時に押す
[Button]	GUIのボタン、メニュー名
太字	強調
「」、『』（鍵カッコ）	関連文書、あるいはガイド内の他の章および節の表示
「入力項目」	画面UI上の入力項目
ハイパーリンク	メールアカウント、Webサイト
>	メニューの実行順
+----	下位ディレクトリー/ファイル有り
----	下位ディレクトリー/ファイル無し
<div>参考</div>	参照/注意事項
[図 1.1]	図の名称
[表 1.1]	表の名称
AaBbCc123	コマンド、コマンド実行結果の画面出力、サンプル・コード
[]	オプション・パラメータ値
	選択パラメータ値

システム要件

	要求事項
プラットフォーム	IBM AIX 5.x / 6.1 / 7.1
	HP-UX 11.xx
	SunOS 5.7~5.9 / SunOS 5.10 / SunOS 5.11
ハードウェア	1GB以上のハードディスク空き容量
	512MB以上のメモリー空き容量
データベース	Oracle 9~12
	Tibero 4~5
	DB2
	Informix

関連ガイド

ガイド	説明
Tmax 運用ガイド	Tmaxを利用するための環境設定ファイルとシステム運用方法について説明しています
Tmax アプリケーション開発ガイド	Tmaxアプリケーション・プログラムの開発で使用するAPIの概念と使用方法および例について説明しています
Tmax リファレンスガイド	Tmaxアプリケーションの開発に使用するコマンドおよびクライアントとサーバーの接続、通信に使用する関数の使用方法と例について説明しています

お問合せ先

Korea

TmaxSoft Co., Ltd.
45, Jeongjail-ro, Bundang-gu,
Seongnam-si, Gyeonggi-do, 13613
South Korea
Tel: +82-31-8018-1000
Fax: +82-31-8018-1115
Email: info@tmax.co.kr
Web (Korean): <http://www.tmaxsoft.com>
TechNet: <http://technet.tmaxsoft.com>

USA

TmaxSoft Inc.
101 North Wacker Drive, Suite 2014,
Chicago, IL 60606
U.S.A
Tel: +1-312-525-8330
Email: info@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/us_en/home

Japan

TmaxSoft Japan Co., Ltd.
5F Sanko Bldg, 3-12-16 Mita, Minato-Ku, Tokyo, 108-0073
Japan
Tel: +81-3-5765-2550
Fax: +81-3-5765-2567
Email: info@tmaxsoft.co.jp
Web (Japanese): <http://www.tmaxsoft.co.jp>

China

Beijing TmaxSoft System Software Co., Ltd.
Room103, No.2 Huizhong Building, Seven Street Shangdi,
Haidian District, Beijing, 100085
P.R.China
Tel: +86-10-6298-8827
Email: info@tmaxsoft.com.cn
Web (Chinese): http://www.tmaxsoft.com/cn_en/home_cn_en

Brazil

Tmax Brasil Sistemas e Serviços Ltda.
Av. Copacabana, 177, sala 32~35 Empresarial 18 do Fortel
Alphaville Barueri, Sao Paulo, 06472-001
Brazil
Tel: +55-11-4191-3100
Fax: +55(11) 4191-3705 (extension#112)
Email: info.bra@tmaxsoft.com
Web (Portuguese): http://www.tmaxsoft.com/br_en/home_br_en

Russia

Tmax Rus L.L.C.
Leninsky prospekt, 113/1 (Park Place Moscow),
Office 318e, Moscow, 117198
Russia
Tel: +7(495)970-01-35
Email: info.rus@tmaxsoft.com
Web (Russian): http://www.tmaxsoft.com/ru_ru/home_ru_ru

Singapore

Tmax Singapore Pte. Ltd.
430 Lorong 6, Toa Payoh #10-02,
OrangeTee Building, 319402
Singapore
Tel: +65-6259-7223
Fax: +65-6258-7112
Email: info.sg@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/sg_en/home_sg_en

United Kingdom

TmaxSoft UK Ltd.
215 Knyvett House, Watermans Business Park,
The Causeway, Staines TW18 3BAB
United Kingdom
Tel: +44-1784-895005
Email: info.uk@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/gb_en/home_gb_en

Canada

TmaxSoft Canada, Inc.
2425 Matheson Blvd East, 8th floor,
Unit 824 Mississauga, ON, L4W 5K4
Canada
Tel: +1-905-361-2888
Email: info.canada@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/ca_en/home_ca_en

Australia

TmaxSoft Proprietary Limited
L32, 101 Miller Street, North Sydney 2060
Australia
Tel: +91-9845-330-704
Email: info.aus@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/au_en/home_au_en

India

TmaxSoft Technologies Private Limited
Sobha Alexander Plaza, 3rd Floor,
16/2 Commissariat Road, Bangalore-560025
India
Tel: +91-9845-330-704
Email: info.india@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/in_en/home_in_en

Turkey

TmaxSoft Co., Ltd. Turkey Liaison Office
Windowist Tower. Eski Buyukdere Cad. No:26,
Maslak 34467 Istanbul
Turkey
Tel: +90-544-553-6045
Email: cslee@tmaxsoft.com
Web (English): http://www.tmaxsoft.com/tr_en/home_tr_en

第1章 XAライブラリーとゲートウェイ

本章では、XAライブラリーとXAゲートウェイの概念および環境設定方法について説明します。

1.1. 概念

X/Open XAとは、2PC(2 Phase Commit)を利用して分散トランザクション処理を行うため、X/Openが策定した標準規格です。XAは、DBMSのベンダー別に提供されており、この標準規格によって異機種間のトランザクションが保証されます。Tmaxでは、このようなXAライブラリーとXAゲートウェイを通じてXAを提供し、異機種との2PCを保証します。

1.1.1. XAライブラリー

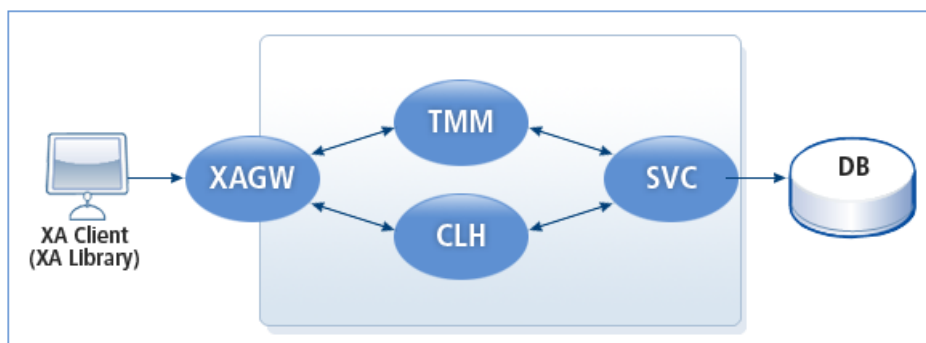
TmaxのXAを使用するには、XAライブラリーを使ってTmaxのサービスを処理する必要があります。ライブラリーは、Tmaxのクライアント・ライブラリーとは若干異なる形式で、XAゲートウェイと通信します。通常、XAインターフェースは、Tmaxのxa-switch名であるtmaxxaoswを使って処理しており、以外にもTmaxのサービスが処理できるように、tpcall/tpacall/tpgetrply APIを追加的にサポートしています。現ライブラリーはマルチスレッドをサポートしていません。

1.1.2. XAゲートウェイ

XAライブラリーからのデータを処理するためのモジュールです。Tmaxシステムでは、XAゲートウェイが追加的に起動されます。

下記の図は、XAゲートウェイがTmaxシステムでサービスを処理するプロセスです。

[図 1.1] XAゲートウェイのサービス処理プロセス



XAゲートウェイはTmaxの環境ファイルに設定した待ち受け(Listen)ポートで、XAライブラリーを利用してアプリケーションの要求を待ちます。ゲートウェイはトランザクション処理を行い、かつTmaxサービスを呼び出すことができます。

1.2. 環境設定

本節では、XAライブラリーとXAゲートウェイの環境設定方法について説明します。

1.2.1. XAライブラリー

XAライブラリー名は、<libtxa.so>です。

Tmaxシステムと連動してトランザクション処理を行うアプリケーションは、TmaxのXAライブラリーと一緒に接続して使用する必要があります。

さらに、アプリケーションがTmaxのバッファ型であるFIELD/STRUCT型を使用する場合は、環境変数としてSDLFILEまたはFDLFILEを下記のように設定します。

```
export SDLFILE=/usr/home/tmax/tmax.sdl
export FDLFILE=/usr/home/tmax/tmax.fdl
```

Tmaxが提供するXAライブラリーを利用してXA APIを使用するには、xa-switchを設定する必要があります。xa-switch名はtmaxxaoswであり、Tmaxと連動してトランザクション処理を行うアプリケーションで設定します。XAライブラリーの設定例と実際にアプリケーションでこのライブラリーと連動するメイクファイルについては、[「第3章 サンプル」](#)を参照してください。

参考

項目は区切り子(+)を使って区別して設定します。

必須項目

下記のフィールドは大文字と小文字を区別するため、必ず下記の文字列で設定します。

- host = ip:port
 - XAゲートウェイが待ち受けしているIPとポート情報は、区切り子のコロン(:)を使って区別して設定します。
 - IPv6プロトコルを使用する場合は、IPとポート番号を区別するためにIPアドレスに角括弧([])で囲みます。またipv6項目を「y」に設定します。

```
host = [2011::100:200]:9500+ipv6=y
```

選択項目

- TIMEOUT = int_value
 - XA APIを使用する際のTIMEOUT値がTMNOWAITでない場合は、この設定によってタイムアウトをチェックすることになります。tpcall、tpacall、tpgetrplyを使用する場合も同様に適用されます。
 - tpcall、tpacall、tpgetrplyについての詳細は、それぞれ[「2.1.1. tpcall」](#)、[「2.2.1. tpacall」](#)、[「2.2.2. tpgetrply」](#)を参照してください。
- ipv6 = [y|n]
 - 使用するプロトコルを設定します。
 - 下記は 設定値についての説明です。

設定値	説明
y	ホストがIPv6プロトコルを使用する場合に設定します
n	設定しないか、「N」に設定する場合、IPv4プロトコルを使用してホストに接続します

1.2.2. XAゲートウェイ

XAゲートウェイはTmaxのゲートウェイとして動作しており、他のゲートウェイと同様、Tmaxの環境ファイルに登録する必要があります。

XAゲートウェイは、Tmax環境ファイルのGATEWAYセクションに下記のとおり登録します。

```
*GATEWAY
Gateway Name      GWTYPE = { TMAX | TMAXNONX | SNACICS | OSITP | JEUS
              JEJW_ASYNC | TUXEDO | TUXEDO_ASYNC | WSGW | XAGW }
PORTNO = Listen_port
NODENAME = ノード名
[DIRECTION = (BIDIR) | IN | OUT,]
[TIMEOUT = second,]
[LOCAL_IPV6 = Y | N]
```

必須項目

項目は大文字と小文字を区別するため、必ず下記の文字列で設定します。

- Gateway Name = string
 - GATEWAYセクションには必ずゲートウェイ名が設定される必要があります。ゲートウェイ名は16字以内に設定し、アルファベットで始まる必要があります。

- GWTYPE = string
 - このフィールドは、Tmaxゲートウェイのタイプを設定するものです。XAゲートウェイを使用する際には、「XAGW」に設定する必要があります。
- PORTNO = Listen_port
 - XAゲートウェイが待ち受けするポートを設定します。このポートは、xa_open文字列のホスト設定と同じ値にします。
- NODENAME = ノード名
 - XAゲートウェイが起動するノード名を設定します。ノードのIPは、xa_open文字列のホスト設定と同じ値にします。

参考

各設定項目についての詳細は、『Tmax 運用ガイド』の「第3章 環境ファイルの設定」を参照してください。

選択項目

- DIRECTION = IN
 - XAゲートウェイは、INチャンネルのみサポートするゲートウェイなので、設定値は必ず「IN」を設定します。
- TIMEOUT = numeric
 - 範囲：1 ~ 128
 - デフォルト値：1
 - 単位：秒
 - ゲートウェイを利用する場合、両ドメインのCLHプロセス間の並列通信チャンネル数を指定します。
 - ゲートウェイ・プロセスが処理量が非常に多く、サービスの所要時間が長い場合、マルチチャンネルを設けて並列通信で処理して処理速度を増加させることができます。TMAXNONTXタイプのゲートウェイの場合、2~3個以下のCPCでも十分です。
- LOCAL_IPV6 = [Y | N]
 - XAゲートウェイが待ち受けソケットを作成する際にIPv6プロトコルを使用するかどうかを設定します。

– 下記は設定値についての説明です。

設定値	説明
Y	IPv6プロトコルを使用する場合に設定します
N	設定しないか、「N」に設定する場合、IPv4プロトコルを使用します

参考

各設定項目についての詳細は、『*Tmax 運用ガイド*』の「第3章 環境ファイルの設定」を参照してください。

設定例

下記は、XAゲートウェイの設定例です。

```
*GATEWAY
XAGW1          GWTYPE = XAGW,
                PORTNO = 10042,
                NODENAME = tmax1,
                DIRECTION = IN,
                TIMEOUT = 30,
                CPC=5,
                LOCAL_IPV6 = N
```


第2章 API

本章では、XAライブラリーとXAゲートウェイを使用するための基本的なAPIの使用方法および環境設定方法について説明します。

Tmaxのサービスを呼び出すためのAPIとして、XAライブラリーは同期型呼び出しと非同期型呼び出しを提供します。

2.1. 同期型呼び出し

同期型呼び出しとして、tpcallを提供します。

tpcallは、tmaxのクライアント・ライブラリーおよびサーバー・ライブラリーで使用するAPIです。同様な構造のXAライブラリーで 사용할 ことができます。

2.1.1. tpcall

サーバーとクライアントの同期型のサービス要求を送受信する関数です。同期型通信でsvcに指定されたサービスにサービス要求を送信し、その応答を受信します。tpcall()を呼び出した後、引き続きtpgetrply()を呼び出した場合と同一に処理されます。

tx_beginを呼び出してから、tx_timeが過ぎた後、tpcallはflagにTPNOTRAN | TPNOREPLYを設定した呼び出しだけ成功し、それ以外はすべてTPETIMEエラーを発生させます。

● プロトタイプ

```
# include <atmi.h>
int tpcall (char *svc, char *idata, long ilen, char **odata, long *olen,
            long flags)
```

● パラメータ

パラメータ	説明
svc	呼び出されるサービス名です。Tmax応用サーバー・プログラムで提供しているものである必要があります
idata	サービス要求のデータに対するポインターです。以前にtpalloc()によって割り当てられたバッファである必要があります。idataのタイプ(type)とサブタイプ(subtype)は、svcがサポートするタイプである必要があります

パラメータ	説明
ilen	<p>送信するデータ長です</p> <ul style="list-style-type: none"> – idataが指すバッファが、特に長さを指定する必要がないタイプ (STRING、STRUCT、X_COMMON、X_C_TYPE) の場合、ilenは無視され、デフォルトで0が使用されます – idataが指すバッファが、長さを指定する必要があるタイプ (X_OCTET、CARRAY、MULTI STRUCTURE) の場合、ilenは0にできません – idataがNULLの場合、ilenは無視されます
*odata	<p>*odataは、受信する応答バッファのポインターです。バッファは*olenで返された長さ分の応答を受信します。*odataは、必ず以前にtpalloc()によって割り当てられたバッファである必要があります。</p> <p>同一バッファが送信と受信の役割を両方共行う場合、*odataはidataのアドレスで設定される必要があります。応答バッファのサイズ変更可否を決定するには、tpcall()が完了する前に、*odataで割り当てられた応答バッファのサイズと受信した*olenを比較します。受信した*olenの方が大きい場合、割り当てられた応答バッファのサイズが増加します。そうでない場合、サイズは変更されません。</p> <p>idataと*odataで同一バッファが使用されてtpcall()が呼び出された場合、*odataが変更されたら、idataが指すアドレスは今後有効ではありません。*odataは受信データが大きくて変更されることがあり、その他の理由によっても変更されることがあります。</p> <p>*olenが0と返された場合、いかなるデータも受信されず、*odataと*odataが指すバッファも何の変化もありません。*odataやolenがNULLになるのはエラーです</p>
*olen	*odataに返される応答の長さです
flags	<p>呼び出し時に使用されるオプションです。通信方式を指定します。</p> <p>flagsで使用可能な値は以下のとおりです</p> <ul style="list-style-type: none"> – TPNOTRAN <p>tpcall()の呼び出し元がトランザクション・モードでこのフラグを設定してsvcサービスを要求した場合、svcサービスはトランザクション・モードから除外されて実行されます。トランザクション・モードで、svcがトランザクションをサポートしないサービスである場合、tpcall()を呼び出すときは、フラグを必ずTPNOTRANに設定する必要があります。トランザクション・モード内でtpcall()を呼び出したとき、TPNOTRANが設定されていても、トランザクション・タイムアウト(timeout)の影響を受けます。つまり、トランザクション・タイムアウトが過ぎた後のTPNOTRANを適用したtpcallも、サービスを呼び出さずに失敗処理するという意味です。TPNOTRANで呼び出されたサービスが失敗した場合、呼び出し元のトランザクションには影響を及ぼしません</p>

パラメータ	説明
	<p>– TPNOCHANGE</p> <p>TPNOBLOCKフラグを設定した状態で、内部バッファが送信メッセージでいっぱいになったときのようなブロッキング状況になった場合、サービス要求は失敗します。TPNOCHANGEは<code>tpcall()</code>のTx部分にのみ適用されます。TPNOBLOCKフラグを設定せずに<code>tpcall()</code>を呼び出したときにブロッキング状況が発生すると、関数の呼び出し元はブロッキング状況が解除されるか、タイムアウト(トランザクション・タイムアウト、またはブロック・タイムアウト)が発生するまで待機します</p> <p>– TPNOTIME</p> <p>関数の呼び出し元がブロック・タイムアウトを無視し、応答を受信するまで無限に待機します。トランザクション・タイムアウト内で<code>tpcall()</code>を実行した場合、トランザクション・タイムアウトが適用されます</p> <p>– TPSIGRSTRT</p> <p>シグナル割り込みを受け入れる場合に使用します。内部でシグナル割り込みが発生し、システム関数の呼び出しが妨害されたとき、システム関数の呼び出しが再実行されます。TPSIGRSTRTフラグが設定されていない状態でシグナル割り込みが発生した場合、関数は失敗し、<code>tperrno</code>にTPGOTSIGが設定されます</p>

● 戻り値

戻り値	説明
1	関数の呼び出しに成功した場合です
-1	関数の呼び出しに失敗した場合です。 <code>tperrno</code> にエラーコードが設定されます

● エラー

`tpcall()`が正常に実行されなかった場合、`tperrno`に以下の値のいずれかが設定されます。

エラーコード	説明
[TPEINVAL]	引数が有効ではないか、フラグが有効でない場合です。たとえば、 <code>svc</code> がNULLであるか、 <code>data</code> が <code>tpalloc()</code> で割り当てられていないバッファを指します
[TPENOENT]	<code>svc</code> というサービスが存在しないため、サービスを要求できない場合です
[TPEITYPE]	<code>data</code> のタイプおよびサブタイプが、 <code>svc</code> がサポートしていないタイプです
[TPEOTYPE]	受信された応答バッファのタイプあるいはサブタイプが、呼び出し元が認識できないタイプです。フラグがTPNOCHANGEと設定されているにもかかわらず、 <code>*odata</code> が指すバッファのタイプおよびサブタイプが、受信された応答バッファのタイプおよびサブタイプと一致しない場合、 <code>*odata</code> の内容と <code>*olen</code> はすべて変更されません。

エラーコード	説明
	呼び出し元がトランザクション・モードでサービスを要求した場合、そのトランザクションは応答が無視されるためロールバックされます
[TPETRAN]	トランザクション・サービスの呼び出し時に、データベースに問題が発生してxa_starが失敗した場合です
[TPETIME]	<p>タイムアウトが発生した場合です。関数の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ロールバックされます。</p> <p>トランザクション・モードではなく、TPNOTIMEとTPNOBLOCKのどちらも指定されていない場合、ブロック・タイムアウトが発生します。この2つの場合に、*odataの内容と*olenは変更されません。</p> <p>トランザクション・タイムアウトが発生した場合、新規サービス要求を送信したり応答を待機したりすることは、トランザクションがロールバックされるまで[TPETIME]エラーで失敗します</p>
[TPESVCFAIL]	<p>サービス要求に対する応答を送信するサービス・ルーチンでアプリケーション・エラーが発生し、TPFAILでtpreturn()を呼び出した場合です。サービス要求が受信されたら、その内容は*odataが指すバッファーを通じて使用できます。</p> <p>トランザクション・タイムアウトが発生し、トランザクションがロールバックされる前に他の通信が実行されることがあります。そういった通信は正常に処理される場合もあり、失敗する場合もあります。通信が正常に実行されるには、TPNOTRANが設定されている必要があります。呼び出し元のトランザクション・モードで実行された作業は、トランザクションの完了時にすべてロールバックされます</p>
[TPESVCERR]	<p>サービス・ルーチン実行中、あるいはtpreturn()(たとえば、誤った引数が送信された場合)実行中にエラーが発生した場合です。エラーが発生すると、いかなる応答データも返さず、*odataの内容または*olenもすべて変更されません。</p> <p>tpallocで生成されていないバッファーを使用した場合や、割り当てられたバッファーのTmaxヘッダーが正しくないポインター(memcpyなど)の影響を受けた場合、あるいはtpacallまたはtpconnectのcdで返した場合、Recvモードでサービスが有効でない対話型の内容のときに発生します。ただし、tpreturnを実行した場合、クライアントはTPESVCERRを受信します。</p> <p>クライアントが強制的に対話を解除し、サービス・プログラムがTPEV_DISCOMNを受信するのと同じTPEV_DISCOMNイベントが発生した場合、クライアント・サービスのtpreturnにTPESVCERR tperrnoが転送されます。</p> <p>関数の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する前に他の通信が実行されることがあります。そういった通信は正常に処理される場合もあり、または失敗する場合もあります。通信が正常に実行されるには、</p>

エラーコード	説明
	<p>TPNOTRANが設定されている必要があります。呼び出し元のトランザクション・モードで実行された作業は、トランザクションの完了時にすべてロールバックされます。</p> <p>Tmax環境ファイルにサービス別にSVCTIMEOUTを設定できます。サービスの実行時間が該当時間を超えた場合、サービスは実行を停止し、TPESVCERRを返します。SVCTIMEOUTが発生した場合、tpsvctimeout()を呼び出しますが、この関数内でバッファ解除やロギング作業など、業務別に適切な作業を行うことができます</p>
[TPEBLOCK]	TPNOBLOCKが設定された状態でブロッキング状況が発生した場合です
[TPGOTSIG]	TPSIGRSTRTが設定されていない状態でシグナルが受信された場合です
[TPEPROTO]	tpcall()が不適切な状況で呼び出された場合です
[TPESYSTEM]	Tmaxシステムにエラーが発生した場合です。詳細情報はログファイルに記録されます
[TPEOS]	運用システムにエラーが発生した場合です

- 例

```

#include <stdio.h>
#include <usrinc/atmi.h>

void main(int argc, char *argv[])
{
    int ret;
    char *sndbuf, *rcvbuf;
    long sndlen, rcvlen;

    ret=tpstart((TPSTART_T *)NULL);
    if (ret==-1) { error processing }

    sndbuf = (char *)tpalloc("CARRAY", NULL, 20);
    if (sndbuf==NULL) {error processing };

    rcvbuf = (char *)tpalloc("CARRAY", NULL, 20);
    if (rcvbuf==NULL) {error processing };

    data process....

    sndbuf=strlen(sndbuf);
    ret=tpcall("SERVICE", sndbuf, sndlen, &rcvbuf, &rcvlen, TPNOCHANGE);
    if (ret==-1) { error processing }

    data process....

```

```

    tpfree((char *)sndbuf);
    tpfree((char *)rcvbuf);
    tpend();
}

```

- 関連関数

tpalloc(), tpacall(), tpgetrply(), tpreturn()

2.2. 非同期型呼び出し

非同期型呼び出しとして、tpacallと応答を取得するためのtpgetrplyを提供します。tpacall、tpgetrplyは、tmaxのクライアント・ライブラリーおよびサーバー・ライブラリーで使用するAPIです。同様な構造のXAライブラリーで 사용할 ことができます。

2.2.1. tpacall

サーバーとクライアントで非同期サービスの要求を送信する関数です。**svc**で指定されたサービスに、サービス要求メッセージを送信します。非同期通信でメッセージを送信後、結果を受信するまで待機せず、すぐに返されます。**tpgetrply()**を使用して応答を受信することができます。あるいは、**tpcancel()**を使用して応答をキャンセルすることもできます。

tx_beginを呼び出してから、tx_timeが過ぎた後、tpacallはflagにTPNOTRAN | TPNOREPLYを設定した呼び出しだけ成功し、それ以外はすべてTPETIMEエラーを発生させます。

- プロトタイプ

```

#include <atmi.h>
int tpacall (char *svc, char *data, long len, long flags)

```

- パラメータ

パラメータ	説明
svc	呼び出されるサービス名です。Tmax応用サーバー・プログラムで提供されるものである必要があります
data	NULL値でない場合、tpalloc()で割り当てられたバッファのポインターである必要があります
len	送信されるデータ長です <ul style="list-style-type: none"> – dataが指すバッファが、長さを指定する必要がないタイプ (STRING、STRUCT、X_COMMON、X_C_TYPE) の場合、lenは無視され、一般的に0が使用されます – dataが指すバッファが、長さを指定する必要があるタイプ (X_OCTET、CARRAY、MULTI STRUCTURE) の場合、lenは0になれません

パラメータ	説明
	<ul style="list-style-type: none"> dataがNULLの場合、lenは無視され、データがない状態でサービス要求が送信されます。dataのタイプ(type)とサブタイプ(subtype)は、svcがサポートするタイプである必要があります。サービス要求がトランザクション・モードで送信された場合、該当応答は必ず受信されます
flags	<p>呼び出し時に使用されるオプションであり、呼び出しモードを指定します</p> <p>flagsに使用可能な値は以下のとおりです</p> <ul style="list-style-type: none"> TPBLOCK <p>フラグなしでtpacall()が使用されると、svcに呼び出されたサービスがない場合、もしくは正しくない結果が返された場合、正常な結果が返されます。tpgetrply()の呼び出し時にエラーが返されます。TPBLOCKフラグを使用してtpacall()を呼び出した場合、サービス状態が正常かどうかを確認できます</p> TPNOTRAN <p>トランザクション・モードで、svcがトランザクションをサポートしないサービスである場合は、tpacall()を呼び出すときに、flagsをTPNOTRANに設定する必要があります。tpacall()の呼び出し元が、トランザクション・モードでTPNOTRANを設定してsvcサービスを要求した場合、svcサービスはトランザクション・モードから除外されて実行されます。トランザクション・モードでtpacall()を呼び出すとき、TPNOTRANが設定されていても、トランザクション・タイムアウト(timeout)の影響を受けます。つまり、トランザクション・タイムアウトが過ぎた後のTPNOTRANを適用したtpacallも呼び出さずに失敗するという意味です。例外として、TPNOTRAN TPNOREPLYを適用したtpacallは、呼び出しを許容します。TPNOTRANで呼び出されたサービスが失敗した場合、呼び出し元のトランザクションには影響を及ぼしません</p> TPNOREPLY <p>tpacall()で送信したサービス要求は、応答を待たずに即時返します。結果は、後でtpacall()で返した記述子を利用してtpgetrply()で結果を受信します。flagsをTPNOREPLYに設定した場合、サービスに対する応答を受けないように設定されます。TPNOREPLYを設定した場合、tpacall()はサービスが正常に呼び出されたときに0を返します。関数の呼び出し元がトランザクション・モードにある場合、TPNOREPLYを使用するためには、TPNOTRANフラグと一緒に設定する必要があります。またサービス状態が正常かどうかをチェックするためには、TPBLOCKと一緒に設定する必要があります。TPBLOCKを設定していない場合は、サービスがNRDYの場合でもエラーを返しません</p> TPNOBLOCK <p>内部バッファが送信メッセージでいっぱいになったときのようなブロッキング(blocking)状況になった場合に、サービス要求が失敗するように設定するフラグで</p>

パラメータ	説明
	<p>す。TPNOBLOCKフラグが設定されていない状態でtpacall()が呼び出され、ブロッキング状況が発生すると、ブロッキング状況が解除されるか、タイムアウト(トランザクション・タイムアウト、またはブロック・タイムアウト)が発生するまで関数の呼び出し元は待機します</p> <p>– TPNOTIME</p> <p>関数の呼び出し元がブロック・タイムアウトを無視し、応答が受信されるまで無限に待機するように設定するフラグです。トランザクション・タイムアウト内でtpacall()を実行した場合、トランザクション・タイムアウトが適用されます</p> <p>– TPSIGRSTRT</p> <p>シグナル(signal)割り込みを受け入れる場合に使用します。内部でシグナル割り込みが発生し、システム関数の呼び出しが妨害されたとき、システム関数の呼び出しが再実行されます。TPSIGRSTRTが設定されていない状態でシグナル割り込みが発生した場合、関数は失敗し、tperrnoにTPGOTSIGが設定されます</p>

● 戻り値

戻り値	説明
記述子(descriptor)	関数の呼び出しに成功した場合です。返された記述子は、送信されたサービス要求に対する応答の受信に使用されます
-1	関数の呼び出しに失敗した場合です。tperrnoにエラーコードが設定されます

● エラー

tpacall()が正常に実行されなかった場合、tperrnoに以下の値のいずれかが設定されます。

エラーコード	説明
[TPEINVAL]	引数が有効でない場合です。たとえば、svcがNULLである場合や、データがtpalloc()で割り当てられていないバッファを指す場合、フラグが有効でない場合に発生します
[TPENOENT]	svcというサービスが存在しないため、サービスを要求できない場合です
[TPEITYPE]	dataのタイプおよびサブタイプが、svcがサポートしていないタイプです。構造体の場合、使用された構造体がSDLFILEファイルに宣言されていない場合に発生します
[TPELIMIT]	処理されていない非同期性サービス要求数が限界に達したため、呼び出し元のサービス要求が送信されなかった場合に発生します
[TPETIME]	タイムアウトが発生した場合です。関数の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たことを意味します。トランザクションはロールバックされます。関数の呼び出し元がトランザクション・モードでない場合、

エラーコード	説明
	TPNOTIMEやTPNOBLOCKがすべて設定されていない状態ではブロック・タイムアウトが発生します。トランザクション・タイムアウトが発生した場合、トランザクションがロールバックされるまで新規サービスの要求を送信したり、まだ受信していない応答を待機することはすべて[TPETIME]エラーとして処理されます
[TPEBLOCK]	TPNOBLOCKが設定された状態でブロッキング状況が発生した場合です
[TPGOTSIG]	TPSIGRSTRTが設定されていない状態でシグナルを受信した場合です
[TPEPROTO]	トランザクション・モードでTPNOREPLYサービスの呼び出し時に、TPNOTRANフラグを設定していないなどの不適切な状況で発生します
[TPESYSTEM]	Tmaxシステムにエラーが発生した場合です
[TPEOS]	運用システムにエラーが発生した場合です

- 例

```
#include <stdio.h>
#include <usrinc/atmi.h>
void main(int argc, char *argv[])
{
    char *buf;
    int ret,cd;
    long len;

    ret=tpstart((TPSTART_T *)NULL);
    if (ret<0) { error processing }

    buf = (char *)tpalloc("CARRAY", NULL, 20);
    if (buf==NULL) {error processing }
    data process....

    cd = tpacall("SERVICE", sndbuf, 20, TPNOTIME);
    if (cd<0) {error processing }
    data process...

    ret=tpgetrply(&cd, (char **)&buf, &len, TPNOTIME);
    if (ret<0) { error processing }
    data process....

    tpfree((char *)buf);
    tpend();
}
```

- 関連関数

tpalloc(), tpcall(), tpcancel(), tpgetrply()

2.2.2. tpgetrply

サーバーとクライアントで非同期的に要求したサービスに対する応答を受信する関数です。**tpacall()**で要求したサービスに対する応答を受信します。

- プロトタイプ

```
# include <atmi.h>
int tpgetrply(int *cd, char **data, long *len, long flags)
```

- パラメータ

パラメータ	説明
cd	tpacall()によって返された呼び出し記述子を指します。一般的に、cdと一致する応答が受信されるか、あるいはタイムアウトが発生するまで待機します。一般的に、cdは応答の受信後、無効になります
*data	以前にtpalloc()によって割り当てられたバッファーに対するポインターである必要があります
len	tpgetrply()が正常に受信したデータ長です。必要な場合、応答内容が指定されたバッファーに受信されるように、バッファーのサイズを増加させます。 *dataは、受信データが大きくて変更されることもあり、それ以外の理由によって変更されることもあります。lenが呼び出し前のバッファーの総サイズより大きい場合、lenが該当バッファーの新規サイズになります。lenが0と返された場合、いかなるデータも受信されず、*dataとlenが指示するバッファーすべて何も変化はありません。 *dataやlenがNULLになるのはエラーです
flags	flagsで使用可能な値は以下のとおりです – TPGETANY 入力値として設定したcdを無視し、これとは関係なく、受信可能な応答を返すようにします。cdは、返された応答に対する呼び出し記述子になります。応答が何もない場合、一般的にtpgetrply()は応答が到着するまで待機します。TPGETANYが設定されていない場合、特に言及がない限り、*cdは無効化される点に留意します。TPGETANYが設定されている場合、cdはエラーが発生した応答に対する記述子になります。応答が返される前にエラーが発生した場合、cdは0になります。特に言及がない場合、呼び出し元のトランザクションに影響を及ぼしません – TPNOCHANGE *dataが指すバッファーのタイプは変更できません。受信された応答バッファーと*dataが指すバッファーのタイプが異なる場合、*dataのバッファー・タイプの受信者が認識できる限度内で受信された応答バッファーのタイプに変更されます。TPNOCHANGEフラグが設定されている場合、変更はできません。受信された応

パラメータ	説明
	<p>答バッファのタイプおよびサブタイプは、*dataが指すバッファのタイプおよびサブタイプと一致する必要があります</p> <p>– TPNOBLOCK</p> <p>応答が到着するまで待機しません。受信可能な応答がある場合は返します。TPNOBLOCKフラグが指定されていない状態で受信可能な応答がない場合、応答が到着するか、またはタイムアウト(トランザクション・タイムアウトやブロック・タイムアウト)が発生するまで関数の呼び出し元は待機します</p> <p>– TPNOTIME</p> <p>関数の呼び出し元がブロック・タイムアウトを無視し、応答が受信されるまで無限に待機します。トランザクション・モードでtpgetrply()を実行した場合、トランザクション・タイムアウトが適用されます</p> <p>– TPSIGRSTRT</p> <p>シグナル割り込みを受け入れる場合に使用します。システム関数の呼び出しが妨害されたとき、システム関数の呼び出しが再実行されます。TPSIGRSTRTフラグが設定されていない状態でシグナル割り込みが発生した場合、関数は失敗し、tperrnoにTPGOTSIGが設定されます</p>

● 戻り値

戻り値	説明
1	関数の呼び出しに成功した場合です。tpreturn()で渡されるtpurcodeグローバル変数は、tpgetrply()が正常に返された場合、あるいはtperrnoが[TPESVCFAIL]の場合、アプリケーションで定義した値をもちます
-1	関数の呼び出しに失敗した場合です。tperrnoにエラーコードが設定されます

● エラー

tpgetrply()が正常に実行されなかった場合、tperrnoに以下の値のいずれかが設定されます。

エラーコード	説明
[TPEINVAL]	引数が有効でない場合です。たとえば、cdやdata、*data、lenなどがNULLである場合や、flagsが有効でない場合に発生します。cdがNULLでない場合は、エラー発生後もcdは有効であり、それに対する応答を待ち続けます
[TPEBADDESC]	cdが有効でない記述子である場合です
[TPEOTYPE]	受信された応答のタイプまたはサブタイプは、呼び出し元を認識できないタイプです。

エラーコード	説明
	flagsはTPNOCHANGEに設定されていますが、*dataのタイプおよびサブタイプが、サービスが送信した応答のタイプと一致しません。*dataの内容と*lenはすべて変更されません。応答が呼び出し元のトランザクション・モードで受信された場合、そのトランザクションは応答が無視されるため、ロールバックされます
[TPETIME]	タイムアウトが発生した場合です。関数の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ロールバックされます。関数の呼び出し元がトランザクション・モードではなく、TPNOTIMEとTPNOBLOCKのどちらも指定されていない場合は、ブロック・タイムアウトが発生します。この2つの場合、*dataの内容と*len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、新規サービスの要求を送信することや、応答を待機することは、トランザクションがロールバックされるまで[TPETIME]エラーとなります
[TPESVCFAIL]	サービス要求に対する応答を送信するサービス・ルーチンが、アプリケーション・エラーが発生したため、TPFAILでtpreturn()を呼び出した場合です。サービス応答が受信されたら、その内容は*dataが指すバッファにより使用できます。関数の呼び出し元がトランザクション・モードである場合、そのトランザクションはロールバックされます。 トランザクション・タイムアウトが発生するまでは、トランザクションがロールバックされる前に他の通信が実行されることがあります。そういった通信は正常に処理されることもあり、または失敗することもあります。通信が正常に実行されるには、TPNOTRANが設定されている必要があります。呼び出し元のトランザクション・モードで実行された作業は、トランザクションの完了時にすべてロールバックされます
[TPEBLOCK]	TPNOBLOCKが設定された状態でブロッキング状況が発生した場合です。記述子(cd)は有効です
[TPGOTSIG]	TPSIGRSTRTが設定されていない状態でシグナルが受信された場合です
[TPEPROTO]	tpgetrply()が不適切な状況で呼び出された場合です
[TPETRAN]	トランザクション・サービスの呼び出し時に、データベースに問題が発生してxa_startが失敗した場合です
[TPESYSTEM]	Tmaxシステムにエラーが発生した場合です
[TPEOS]	運用システムにエラーが発生した場合です

- 例

```
#include <stdio.h>
#include <usrinc/atmi.h>
void main(int argc, char *argv[])
{
    int ret;
    long len;
```

```

char *buf;

ret=tpstart((TPSTART_T *)NULL);
if (ret==-1) { error processing }

buf = (char *)tpalloc("STRING", NULL, 0);
if (buf==NULL) { error processing }
data process ...

cd = tpacall("SERVICE", buf, 0, TPNOFLAGS);
if (cd==-1) { error procesing }
data process....

ret=tpgetrply(&cd, &buf, &len, TPNOTIME);
if (ret==-1) { error processing }
data process....

tpfree(buf);
tpend();
}

```

- 関連関数

tpacall(), tpalloc(), tpreturn()

第3章 サンプル

本章では、Tmax XAサービスを呼び出す簡単なサンプルとメイクファイルについて説明します。

3.1. Tmax XAサービスの呼び出し例

```
/* Tmaxのxa switch openinfo */
#define OPENINFO
"host=192.168.1.43:7000+dbgldvl=1+timeout=10"
/* Tmaxのxa switch closeinfo */
#define CLOSEINFO      ""
extern struct xa_switch_t tmaxxaosw;
struct xa_switch_t _xasw;
int _xa_load(int xaooption)
{
    _xasw.name[0] = 0;
    _xasw = tmaxxaosw;
    if (_xasw.name[0] == 0)
        return -1;
    return 1;
}
int main(int argc, char **argv)
{
    ...
    XID xid;

    // XA情報のロード
    ret = _xa_load(0);
    if (ret < 0) {
        error processing...
    }

    // xa open
    ret = xa_open(OPENINFO, rmid, TMNOFLAGS);
    if (ret < 0) {
        error processing...
    }
    ...

    // xa start
    ret = xa_start(&xid, rmid, TMNOFLAGS);
```

```

    if (ret < 0) {
        error processing...
    }

    // Tmax XAサービスの呼び出し
    ret = call_tmax("SVCA", sndbuf, rcvbuf);
    if (ret < 0) {
        error processing...
    }

    // xa end
    ret = xa_end(&xid, rmid, TMSUSPEND);
    if (ret < 0) {
        error processing...
    }

    // xa start
    ret = xa_start(&xid2, rmid, TMRESUME | TMJOIN);
    if (ret < 0) {
        error processing...
    }
    /* tpcallを含む関数 */

    // Tmax XAサービスの呼び出し
    ret = call_tmax(svcname, sndbuf, rcvbuf);
    if (ret < 0) {
        error processing...
    }

    // xa end
    ret = xa_end(&xid2, rmid, TMSUCCESS);
    if (ret < 0) {
        error processing...
    }

    // xa prepare
    ret = xa_prepare(&xid, rmid, TMNOFLAGS);
    if (ret < 0) {
        error processing...
    }
    // xa prepare
    ret = xa_prepare(&xid2, rmid, TMNOFLAGS);
    if (ret < 0) {
        error processing...
    }
    else if (ret == XA_RDONLY) {
        // xa commit

```



```

        ret = xa_commit(&xid, rmid, TMNOFLAGS);
        if (ret < 0) {
            error processing...
        }
    }
    else {
        ret = xa_commit(&xid, rmid, TMNOFLAGS);
        if (ret < 0) {
            error processing...
        }
        ret = xa_commit(&xid2, rmid, TMNOFLAGS);
        if (ret < 0) {
            error processing...
        }
    }

    // xa close
    ret = xa_close(CLOSEINFO, rmid, TMNOFLAGS);
    if (ret < 0) {
        error processing...
    }
...
}

// Tmaxサービス呼び出しルーチン
int call_tmax(char *svcname, char * sndbuf, char * rcvbuf)
{
    ...
    ret = tpcall(svcname, (char *)sndbuf, strlen(sndbuf), (char **)&rcvbuf,
                  (long *)&rcvlen, TPNOFLAGS);
    if(ret < 0) {
        error processing...
    }
    ...
}

```

3.2. メイクファイル

```

# makefile
TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

TMAXLIBD= $(TMAXDIR)/lib64

TMAXLIBS= -ltxa

```

```

# hp-paの場合
CFLAGS = -g -Ae +DA2.0W +DD64 +DS2.0 -O -I$(TMAXDIR)

#
.SUFFIXES : .c

.c.o:
    $(CC) $(CFLAGS) -c $<

#
# client compile
#

$(TARGET): $(APOBJS)
#     proc iname=$(TARGET) include=$(TMAXDIR)
    $(CC) $(CFLAGS) -L$(TMAXLIBD) -o $(TARGET) $(APOBJS) $(TMAXLIBS)

$(APOBJS): $(TARGET).c
#     proc iname=$(TARGET) include=$(TMAXDIR)
    $(CC) $(CFLAGS) -c $(TARGET).c
#
clean:
    -rm -f *.o core $(TARGET)

```

索引

F

FDLFILE設定, 2

S

SDLFILE設定, 2

T

tpacall, 12

tpcall, 7

tpgetrply, 16

X

X/Open XA, 1

XAゲートウェイ, 1

XAゲートウェイ設定

 DIRECTION, 4

 GWTYPE, 4

 LOCAL_IPV6, 4

 NODENAME, 4

 PORTNO, 4

 TIMEOUT, 4

 選択項目, 4

 必須項目, 3

 ゲートウェイ名, 3

XAライブラリー, 1

XAライブラリー設定

 host, 2

 ipv6, 3

 timeout, 3

 選択項目, 3

 必須項目, 2

