

Tmax ゲートウェイガイド (シリアル)

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 ゲートウェイガイド (シリアル)

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 紹介 | 1 |
| 1.1. 概要 | 1 |
| 1.2. サービスのタイプ | 3 |
| 1.2.1. Tmaxからのサービス要求 | 3 |
| 1.2.2. リモートノードからのサービス要求 | 5 |
| 第2章 環境設定 | 7 |
| 2.1. 概要 | 7 |
| 2.2. Tmax環境構成 | 7 |
| 2.2.1. ACK/NAK応答のSRLGW | 9 |
| 2.2.2. 遅延処理のSRLGW | 10 |
| 2.3. SRLGWの環境ファイル | 10 |
| 2.3.1. アドレス情報の環境ファイル | 10 |
| 第3章 COMライブラリー | 13 |
| 3.1. 概要 | 13 |
| 3.2. ComOpen | 13 |
| 3.3. ComDetach | 14 |
| 3.4. ComClose | 14 |
| 3.5. ComSend | 15 |
| 3.6. ComRecv | 16 |
| 第4章 例 | 17 |
| 4.1. NOREPLYサービス呼び出し | 17 |
| 4.1.1. 環境ファイル | 17 |
| 4.1.2. リモートノード | 18 |
| 4.2. REPLYサービス呼び出し | 21 |
| 4.2.1. 環境ファイル | 21 |
| 4.2.2. リモートノード | 22 |
| 4.3. リモートノード同期型呼び出し | 25 |
| 4.3.1. 環境ファイル | 26 |
| 4.3.2. リモートノード | 27 |
| 4.4. ACK/NAKの通信方式 | 28 |
| 4.4.1. 環境ファイル | 29 |
| 4.4.2. リモートノード | 30 |
| 付録 A. ヘッダー・ファイル | 33 |
| A.1. COMMHEADファイル | 33 |
| 索引 | 35 |

図目次

| | | |
|---------|----------------------------------|---|
| [図 1.1] | SRLGWの動作構造 | 1 |
| [図 1.2] | NOREPLYサービス呼び出しのSRLGWの動作構造 | 3 |
| [図 1.3] | REPLYサービス呼び出しのSRLGWの動作構造 | 4 |
| [図 1.4] | 同期型呼び出しのSRLGW | 5 |
| [図 1.5] | 非同期型呼び出しのSRLGW | 6 |

このガイドについて

対象読者

本書は、Tmax[®](以下、Tmax)シリアル・ゲートウェイを使用して開発するユーザーを対象としています。同書では、Tmaxサーバーと非Tmaxサーバーがシリアル通信を行う場合、インターフェースを担当するTmaxが提供するSERIAL(RS232)ゲートウェイ(以下、SRLGW)について説明しています。

前提知識

本書は、Tmaxシステムの概要とTmaxシステムが提供する各種機能や特性などを習得するための手引書です。

本書を理解するためには、以下の事項についての知識が必要です。

- ミドルウェアおよびUNIXシステムについての知識
- Tmaxの基本概念
- Java, Cプログラミングについての知識

制限事項

本書を読む前にTmaxの基本概念を熟知している必要があります。実務上の具体的な使用方法や管理・運用についての内容は、各製品ガイドを参照してください。

参考

Tmaxシステムの開発についての基本的な内容は、『Tmax 運用ガイド』および『Tmax アプリケーション開発ガイド』を、Tmaxが提供するコマンドとC API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

本書は、計4章と付録で構成されています。

各章の主な内容は以下のとおりです。

- 第1章: 紹介

システムの概要および動作構造、サービスのタイプについて説明します。

- 第2章: 環境設定

SRLGWの環境構成および環境ファイルの構成について説明します。

- 第3章: COMライブラリー

SRLGWで使用するAPI関数について説明します。

- 第4章: 例

SRLGWの各役割に関する例について説明します。

- 付録A: ヘッダー・ファイル

COMMHEADファイルについて説明します。

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 |
| <Ctrl>+C | CtrlとC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 |
| 参考 | 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、コマンド実行後の画面に出力された結果物、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 |

関連文書

| 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 紹介

本章では、システムの概要および動作構造とサービスのタイプについて説明します。

1.1. 概要

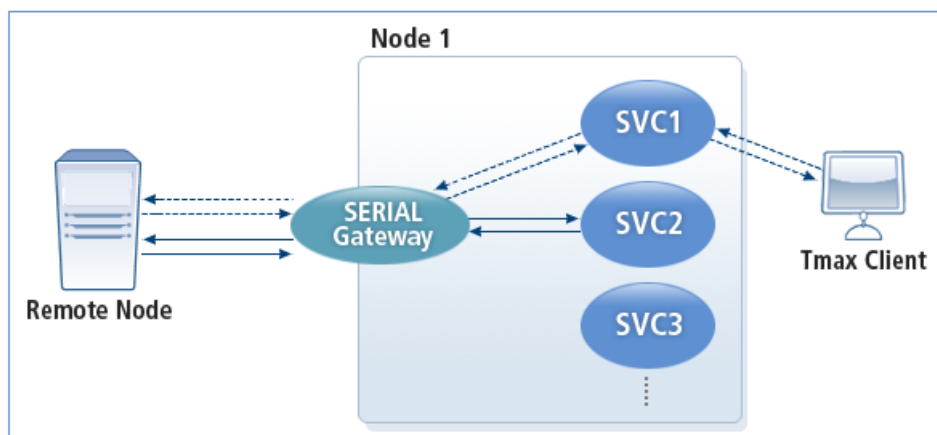
SERIAL(RS232)ゲートウェイ(以下、SRLGW)は、Tmaxサーバーと非Tmaxサーバー(以下、リモートノード)間のシリアル通信におけるインターフェースを担当する、Tmaxが提供するゲートウェイです。

SRLGWは、大きく2つのモジュールで分けられます。1つは、Tmaxとリモートノードがシリアルで送受信する部分です。もう1つは、リモートノードでTmaxのSRLGWとの通信を担当するCOMライブラリーです。COMライブラリーは、SRLGWユーザーがシリアル通信に関連するプログラムを作成しなくても、SRLGWとシリアル通信ができるようにするシリアル通信ライブラリーです。ユーザーはこのライブラリーを利用して、SRLGWからサービスを受信したり、サービスを要求したりすることができます。

TmaxのSRLGWとリモートノードのCOMライブラリーを使用して、ユーザーは双方向サービスを利用することができます。さらに、他のシステムとシリアル通信を行うために複雑な作業を行う必要はありません。例えば、ポートを開きメッセージを送受信する作業は、すべてSRLGWとCOMライブラリーで処理するため、開発者は業務ロジックのみ作成して簡単に他のシステムと接続することができます。

SRLGWはTCP/IPのようにサーバーとクライアント・モードで区分されないため、リモートノードとシリアルで接続さえできれば、どちらからでも先にサービスを要求することができます。SRLGWが動作する方式は、Tmaxのサービスまたはクライアントでリモートノードにサービスを要求する方式と、リモートノードでTmaxのサービスを呼び出す方式で分けられます。下記は、SRLGWの動作構造です。

[図 1.1] SRLGWの動作構造



- Tmaxからのサービス要求

[図 1.1]の点線で表示されている部分です。Tmaxクライアントまたはサービスからサービス要求を受け、リモートノードにサービスを要求します。Tmaxクライアントからリモート・ノードにサービスを要求する際、SRLGWに登録されているサービス名を使用します。このようなサービスをインバウンド・サービスといいます。

TmaxサービスでSRLGWにtpcallまたは他の方式でサービスを要求すると、SRLGWはリモートノードに要求メッセージを送信した後、応答が受信されたら自身を呼び出したサービスにtpreturnします。

SRLGWは、Tmaxからリモートノードへの要求を非同期型方式で処理します。すなわち、サービスまたはクライアントがリモートノードにtpcallでサービスを要求すると、SRLGWはリモートにサービス要求を送信して、サービスまたはクライアントに直ちに正常応答を返します。その後、リモートノードではサービス要求に対する応答であっても新しいサービス要求と見なします。したがって、Tmaxで要求するすべてのサービスは非同期型方式で動作します。

- リモートノードからのサービス要求

[図 1.1]の実線で表示されている部分です。リモートノードからサービス要求を受けて処理します。リモートノードからサービス要求を受ける際にはTmaxのサービス名を利用します。このようなサービスをアウトバウンド・サービスといいます。

SRLGWは、リモートノードが送信したメッセージを受け、ユーザーが要求したサービスにtpacallします。サービス結果は、最初に要求したリモートノードに転送します。

リモートノードでTmaxのサービスを要求する場合は、非同期型、同期型を使用することができます。

非同期型は、COMライブラリーを利用してSRLGWにサービスを要求する際、flagsにCOMMNOREPLYを設定すると、SRLGWはTmaxにユーザーが要求したサービス呼び出しのみ行い、応答は受けません。

同期型方式は、COMライブラリーを利用してSRLGWにサービスを要求する際、flagsにCOMMNOREPLYを設定せずに要求すると、SRLGWはTmaxにユーザーが要求したサービスを呼び出し、応答が受信されたらサービスを要求したリモートノードに転送します。

リモートノードでTmaxにサービスを要求する際、遅延処理方式をサポートします。flagsにCOMMDELAYを設定すると、SRLGWは以後ユーザーがRQのデータが一括処理できるようにRQに要求データを保存します。遅延処理は、常に非同期型方式で処理します。

TmaxにインストールされるSRLGWは、Tmaxサーバーの一種です。SRLGWを使用するには、Tmax環境ファイルにサーバーとして登録する必要があります。一般的には、TCSまたはUCS方式のサーバー・ライブラリーを利用してサーバーを作成しますが、SRLGWはユーザーが直接使用できるように実行可能なファイルで提供します。そのため、ユーザーはSRLGWをTmax環境ファイルにサーバーとして登録する作業のみ行います。

SRLGWは、指定されているリモートノードへのみサービスが要求できるため、ユーザーはサービスを要求したいリモートノードを事前に把握しておく必要があります。そうするためには、Tmax環境ファイルにサービス名を登録し、当該サービス名をSRLGW環境ファイルのリモートノード別の情報に登録します。SRLGWは

サービス要求を受けると、要求したサービス名を使ってリモートノードを見つけ、該当のリモートノードにデータを転送します。Tmax環境設定についての詳細内容は、「[第2章 環境設定](#)」を参照してください。

1.2. サービスのタイプ

1.2.1. Tmaxからのサービス要求

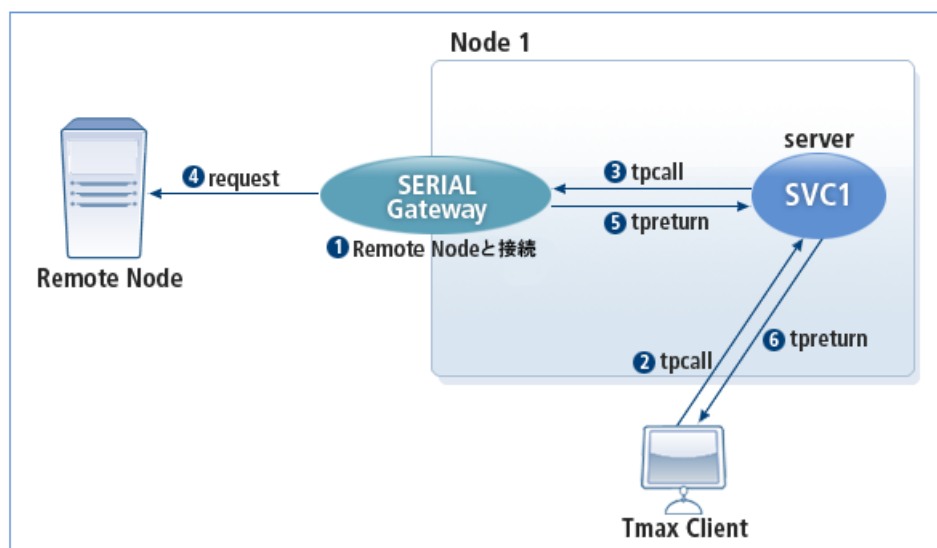
Tmaxクライアントまたはサービスからサービス要求を受け、リモートノードにサービスを要求することができます。Tmaxでサービスを要求する際、リモートノードにサービス要求を送信し、応答は受信しない場合(NOREPLYサービス呼び出し)と、サービス要求を送信し、当該応答を新しい要求として処理(REPLYサービス呼び出し)する、2つの方式で分けることができます。

NOREPLYサービス呼び出し

Tmaxクライアントの要求を受けたサービスでSRLGWにtpcallすると、SRLGWはリモートノードにサービスを要求し、リモートノードの結果を問わず直ちに返す方式です。このような方式でSRLGWを動作させると、Tmaxクライアントで呼び出したTmaxサービスはSRLGWを呼び出した後、結果を問わず継続して処理することになります。

下記は、NOREPLY方式でサービスを呼び出す場合のSRLGWの動作構造です。

[図 1.2] NOREPLYサービス呼び出しのSRLGWの動作構造



1. SRLGWとリモートノードが接続されています。
2. TmaxクライアントはTmaxサービスをtpcallします。

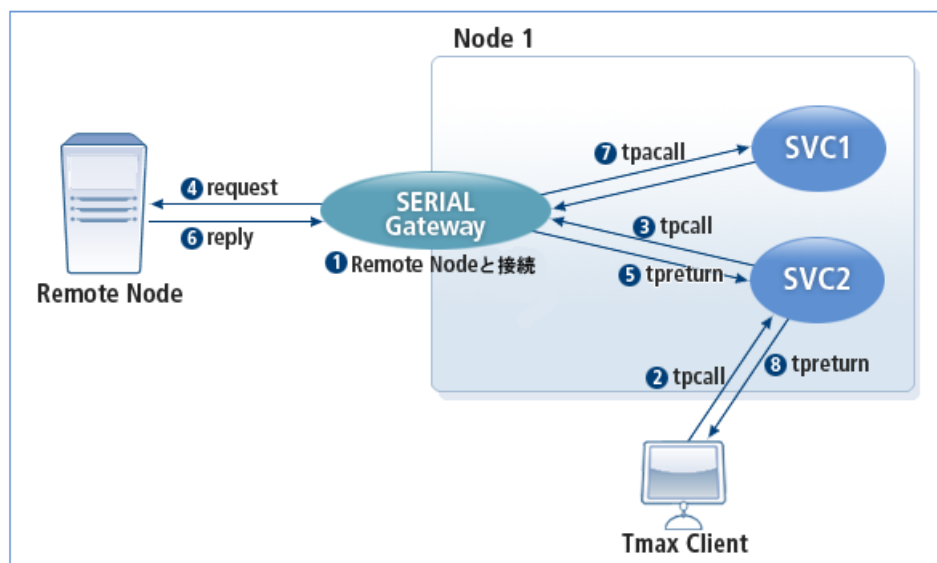
3. Tmaxサービスではクライアントの要求を受け、SRLGWにサービスをtpcallします。SRLGWには多数のサービスが登録されているため、ユーザーが要求しようとするリモートノードと接続されているシリアル・ポートのサービス名を指定する必要があります。
4. SRLGWは、ユーザーが指定したサービスと接続されているリモートノードにサービスを要求します。
5. SRLGWは、サービスを要求したサービスに正常応答をtpreturnします。

REPLYサービス呼び出し

REPLYサービス呼び出しは、リモートノードが要求サービス进行处理し、その結果をTmaxの新しいサービスとして呼び出す方式です。新しいサービスを呼び出す場合は、ComSend関数のflagsに設定する値によって、Tmaxのサービス処理結果を受けることも受けないこともできます。このような方式は、最初にサービスを呼び出すサービスと結果を処理するサービスを区分するために使用します。

下記は、REPLY方式でサービスを呼び出す場合のSRLGWの動作構造です。Tmaxはサービス処理結果を受けないように設定されています。

【図 1.3】 REPLYサービス呼び出しのSRLGWの動作構造



1. SRLGWとリモートノードが接続されています。
2. TmaxクライアントはTmaxサービスをtpcallします。
3. Tmaxサービスではクライアントの要求を受け、SRLGWにサービスをtpcallします。SRLGWには多数のサービスが登録されているため、ユーザーが要求しようとするリモートノードと接続されているシリアル・ポートのサービス名を指定する必要があります。
4. SRLGWは、ユーザーが指定したサービスと接続されているリモートノードにサービスを要求します。

5. SRLGWは、サービスを要求したサービスに正常応答をtpreturnします。
6. リモートノードで結果に対する新しいサービス要求を受けます。
7. SRLGWは、ユーザーがTmaxに指定したサービスを呼び出します。

1.2.2. リモートノードからのサービス要求

リモートノードでTmaxのサービスを呼び出します。この方式は、同期型方式と非同期型方式で分けられます。同期型は、サービスを要求して応答を待つ方式であり、非同期型は、サービスを要求するが応答は待たない方式です。同期型方式を利用する際、サービスを要求して応答を受ける前に、他のサービス要求が当該リモートノードに転送されることもあるため、これに対する処理は適切に行う必要があります。

同期型呼び出し方式

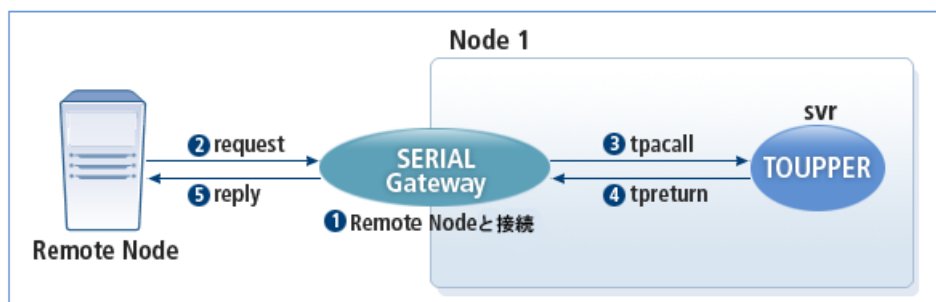
同期型呼び出しは、リモートノードでSRLGWにサービスを要求し、応答を待つ方式です。SRLGWはリモートノードで要求したサービスを呼び出し、その結果を受けて当該サービスを要求したチャンネルに転送します。リモートノードはSRLGWに、Tmax環境ファイルに定義されているMAXSACALL数を超過して呼び出すことができません。

リモートノードのユーザーは、COMライブラリーを利用してSRLGWにサービスを呼び出し、応答を受けます。さらに、ComSend時は、flagsにいかなる設定も不可です。

このような動作方式は、リモートノードでTmaxのサービスを呼び出す最も一般的な方式です。SRLGWは、リモートノードのチャンネル情報を保存しておき、サービスから結果が受信されたら保有中のチャンネルから該当のチャンネルを見つけ結果を転送します。このとき、該当のチャンネルに結果を返す前に他のサービスを要求することができます。すなわち、リモートノードで要求したチャンネルはブロックされないため、要求に対する応答がリモートノードに転送される前に他のサービス要求がリモートノードに転送される場合があるので、SRLGWはこれも処理しなければなりません。

下記は、同期型呼び出し方式でのSRLGWの動作方式です。

[図 1.4] 同期型呼び出しのSRLGW



1. SRLGWとリモートノードが接続されています。

2. リモートノードは、SRLGWと接続されているチャンネルにメッセージを転送します。
3. SRLGWは、tpacallでTmaxサービスを呼び出します。
4. SRLGWはサービス処理結果を受け、メッセージを要求したチャンネルを検索します。
5. 該当のチャンネルが正常に接続されているなら結果を転送します。

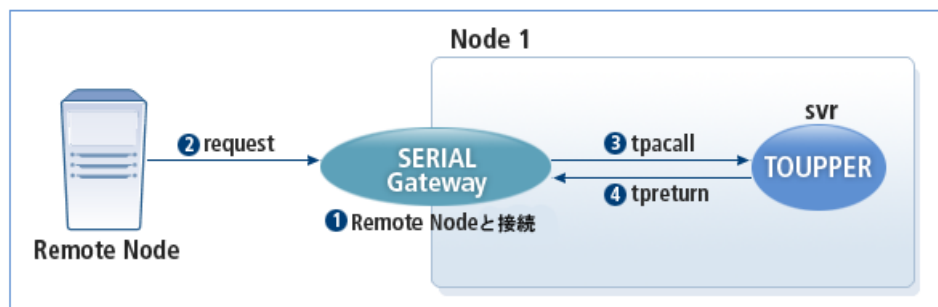
非同期型呼び出し方式

リモート非同期型呼び出しは、リモートノードから先にSRLGWにサービスを要求するが応答は待たない方式です。SRLGWはリモートノードで要求したサービスを呼び出し、その結果は受けません。

この方式で処理するためにリモートノードでは、ComSend時にflagsにCOMMNOREPLYを設定する必要があります。COMMNOREPLYが設定されていると、SRLGWはTmaxにサービスを要求するが応答は受けません。

下記は、非同期型呼び出し方式でのSRLGWの動作方式です。

[図 1.5] 非同期型呼び出しのSRLGW



1. SRLGWとリモートノードが接続されています。
2. リモートノードは、SRLGWと接続されているチャンネルにメッセージを転送します。
3. SRLGWは、tpacallでTmaxサービスを呼び出します。
4. 処理結果を返します。

第2章 環境設定

本章では、SRLGWの環境構成および環境ファイルの構成について説明します。

2.1. 概要

SRLGWサーバーを構成するには、下記のようなファイルが必要となります。ファイル(SRLGWとライブラリー)は、Tmax SRLGWのインストール時に各ディレクトリーの下位に作成されます。

以下は、OSに応じて使用される環境ファイルです。

- UNIX

| ディレクトリー | ファイル名 |
|---------|----------------------|
| Lib | tmaxcom.a、tmaxcom.so |
| Lib64 | tmaxcom.a、tmaxcom.so |
| appbin | serialgw |

- Windows

| ディレクトリー | ファイル名 |
|---------|-------------------------|
| Lib | tmaxcom.lib、tmaxcom.dll |
| bin | serialgw.exe |

参考

現在はWindowsシステムに対してのみSRLGWが実行されており、UNIXシステムは次期バージョンでサポートする予定です。

2.2. Tmax環境構成

SRLGWを使用するには、Tmax環境ファイルにSRLGWをサーバーとして登録する必要があります。登録方法は、TmaxサーバーのUCS方式と類似しています。ただし、UCSでのSVRTYPEがCUSTOM_GATEWAYという点が異なります。SRLGWを使用するためにTmax環境ファイルを変更するときは、SERVERとSERVICEセクションのみ設定します。

下記は、Tmax環境ファイルの例です。

```

*DOMAIN
tmax      SHMKEY = 88000, MINCLH = 1, MAXCLH = 1, TPORTNO = 8800

*NODE
tmax1     TMAXDIR = "/home/tmax",
          APPDIR = "/home/tmax/appbin"

*SVRGROUP
svg1      NODENAME = tmax1

*SERVER
serialgw  SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
          SVRTYPE = CUSTOM_GATEWAY,
          CLOPT = "-- -F /home/tmax/config/serlal.cfg -w 5050"

*SERVICE
COM1      SVRNAME = serialgw
COM2      SVRNAME = serialgw

```

| 項目 | 説明 |
|---------|--|
| MIN | SRLGWのプロセス数を指定する項目です。値は常に1に指定します |
| CPC | TmaxエンジンとSRLGW間のチャンネル数を指定します。Tmaxのクライアントまたはサービスからリモートノードに要求する場合は、同時に要求する数の分だけCPC数を指定します。逆の場合、SRLGWはTmaxエンジンにtpacallで要求するため、多くのチャンネルを使用する必要はありません |
| CLOPT | CLOPT項目(SRLGWオプション) の内容を参照してください |
| SVCTIME | SRLGWは、リモートノードに要求したサービスに対してサービス・タイムアウトを適用しません。リモートノードにはサービスを要求するのみで、応答は別のサービスとして処理します |

CLOPT項目(SRLGWオプション)

SRLGWはTmax環境ファイルに登録できる項目が制限されているため、CLOPT項目にいくつかのオプションを設定する必要があります。

下記のオプションによってSRLGWの動作方式が異なるため、以下の説明を十分に理解する必要があります。

| オプション | 説明 |
|-------|---|
| [-F] | SRLGWで使用するシリアル・ポートの情報およびリモートノードを見つけるサービス名が登録されているファイルを指定します。SRLGWはこのオプションに指定したファイルの情報をロードしてリモートノードと接続します。また、Tmaxのサービスまたはクライアントでリモートノードにサービスを要求する際、要求したサービス名でリモートノードを検索する場合にも使 |

| オプション | 説明 |
|-------|---|
| | 用します。ファイルの登録方法については、「 2.3. SRLGWの環境ファイル 」を参照してください |
| [-w] | SRLGWは、内部的に1つのポートに対して1つのスレッドが処理する構造になっています。それぞれのポートを担当するスレッドから、メイン・スレッドにデータを転送するために1つのソケットを使用しますが、このとき使用するポート番号を指定するオプションです(デフォルト値のポート番号: 8880) |
| [-q] | リモートノードでTmaxのサービスを要求するとき、遅延処理方式で 사용할 ことができます。すなわち、Tmaxが提供するRQに一時データを保存しておき、以後一括処理する方式でSQLGWが使用できますが、このとき使用するRQ名を登録するオプションです |
| [-A] | SRLGWとリモートノード間の送受信データに対して、ACK/NAKメッセージの送受信有無を指定するオプションです。このオプションを指定すると、送受信データに対してACK/NAKをやり取りします。TmaxのSRLGWは、内部的にACK/NAKメッセージを処理しますが、リモートノードのCOMライブラリーではACK/NAKを処理しないため、ユーザー側のプログラムでこれが処理できる部分が必要となります(デフォルト値: ACK/NAKメッセージを送受信しない) |
| [-t] | リモートからACKメッセージが受信される待機時間を指定するオプションです。この時間内にACK/NAKメッセージが受信されない場合は、ACKメッセージが受信されたと判断します(デフォルト値: 無限) |

2.2.1. ACK/NAK応答のSRLGW

下記は、ACK/NAKメッセージを受ける場合のSRLGW環境設定に関する例です。

```
*DOMAIN
...
*NODE
...
*SVRGROUP
...
*SERVER
serialgw          SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
                  SVRTYPE = CUSTOM_GATEWAY,
                  CLOPT = "-- -F /home/tmax/config/serlal.cfg -w 5555 -A"
*SERVICE
COM1              SVRNAME = serialgw
```

上記のように環境ファイルを作成した場合、serialgwという名前のSRLGWの1つがTmaxのブート時に起動されます。serialgwのサービス名はCOM1です。SRLGWはリモートノードとのデータ送受信時にACK/NAKメッセージをやり取りするため、リモートノードのプログラムの場合はACK/NAKまで考慮する必要があります。

2.2.2. 遅延処理のSRLGW

下記は、RQを利用して遅延処理方式でSRLGWを使用する場合の環境設定に関する例です。

```
*DOMAIN
...
*NODE
...
*SVRGROUP
...
*SERVER
serialgw          SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
                  SVRTYPE = CUSTOM_GATEWAY,
                  CLOPT = "-- -F /home/tmax/config/serlal.cfg -w 5555 -q testq"
*SERVICE
COM1              SVRNAME = serialgw
```

上記のように環境ファイルを作成し、リモートノードでCOMライブラリーを利用して(flagsIにCOMMDELAYを設定)Tmaxにサービスを要求すると、SRLGWは要求データをtestqに保存します。

SRLGWはリモートノードから受信したデータをRQに保存するのみでいかなる処理もしないため、ユーザーはRQに保存されているデータを処理する必要があります。

2.3. SRLGWの環境ファイル

Tmax環境ファイルのCLOPTセクションに[-F]オプションで指定したファイルは、フォーマット情報の環境ファイルに登録される必要があります。

2.3.1. アドレス情報の環境ファイル

下記は、リモートノードとシリアルで接続するポートの情報を登録するファイルです。ファイルに登録したポート情報とリモートノードで使用するポート情報は一致する必要があります。

SRLGWは、環境ファイルに登録されている数の分だけ内部的にスレッドを割り当てするため、リモートノードとのシリアル通信が必要なすべてのポートを登録します。下記のファイルに登録されていないリモートノードは、SRLGWと通信することができません。

```
#####
# gwno baudrate stopbit Parity Byte Port      Tmaxsvc
# 0          9600      1      ODD      8  COM1      SVC
#####
# line start with "#" is comment line
# gwno must start at 0 and be increased by 1
# stopbit: 1, 1.5, 2
# parity : NONE, ODD, EVEN
```

```
# byte: 5, 6, 7, 8
# port: serial port name
# svc: tmax service name
#####
0      9600    1      NONE    8      COM1    SVC1
0      9600    1      NONE    8      COM2    SVC2
```

下記は、アドレス情報の環境ファイルに関する説明です。

| 項目 | 説明 |
|----------------|--|
| gwno(ゲートウェイ番号) | Tmax環境ファイルのMIN値が2以上の場合に使用します。ただし、MIN値は常に1なので、この項目はゼロに指定する必要があります |
| baudrate | リモートノードとSRLGW間の通信速度を指定します |
| stopbit | シリアル通信時に使用するストップ・ビット数を指定します。ストップ・ビットは、1ビット、1.5ビット、2ビットのいずれかを指定します |
| Parity | パリティを指定します。NONE、ODD、EVENタイプのいずれかを指定します |
| Byte(ビット数) | 1つのバイトが何ビットで構成されるのかを指定します。5、6、7、8のいずれかを指定します |
| Port(ポート名) | リモートノードと接続されているポート名を指定します。SRLGWを指定したポートを利用してリモートノードと通信するため、リモートノードと物理的に接続されているポート名を登録する必要があります |
| Tmaxsvc(サービス名) | <p>Tmaxのクライアントまたはサービスで特定のリモートノードにサービスを要求するためには、SRLGWに送信しようとするリモートノードを知らせる必要があります。このような場合に使用する項目で、指定されたサービス名でSRLGWを呼び出すと、当該サービス名を使ってリモートノードを見つけ、該当のリモートノードにデータを転送します。</p> <p>項目に指定したサービス名は、Tmax環境ファイルのSERVICEセクションに登録されているサービス名と一致する必要があります。SRLGWがリモートノードを見つけるには、登録したサービス名を利用する方法と、以後リモートノードからSRLGWに接続を設定する際にリモートノード名を登録する方法があります。このように登録するリモート名は、Tmax環境ファイルのSERVICEセクションに登録されている名前と一致する必要があります</p> |

第3章 COMライブラリー

本章では、SRLGWの観点でCOMライブラリーが提供するAPIの使用方法について説明します。COMライブラリーのプロトタイプは、comhead.hヘッダー・ファイルに定義されています。

3.1. 概要

COMライブラリーは、リモートノードでTmaxのSRLGWとRS232で通信できるように、Tmaxが提供するライブラリーです。SRLGWとの通信を希望するリモートノードは、必ずCOMライブラリーを使用して通信します。リモートノードのユーザーは、COMライブラリーを利用してTmaxにシリアルでサービスを要求するか、またはサービス要求を受けることができます。さらに、Tmaxと通信する際に通信関連部分はCOMライブラリーで処理するため、簡単にTmaxのサービスを利用することができます。

下記は、COMライブラリーが提供するAPIの一覧です。

| API | 説明 |
|---------------------------|---|
| ComOpen | TmaxのSRLGWとシリアルで接続を確立し、リモートノードをSRLGWに登録する関数です |
| ComDetach | TmaxのSRLGWとの登録を解除するために使用する関数です |
| ComClose | TmaxのSRLGWとの登録を解除し、接続を切断する関数です |
| ComSend | リモートノードからTmaxのSRLGWにデータを転送する関数です |
| ComRecv | TmaxのSRLGWからデータを受信する関数です |

3.2. ComOpen

TmaxのSRLGWとシリアルで接続を確立し、リモートノードをSRLGWに登録する関数です。

TmaxのSRLGW環境ファイルのサービス名項目に、この関数を利用してサービス名を登録する必要があります。サービス名が登録されていない場合、リモートノードからTmaxにサービスを要求することは可能ですが、Tmaxからリモートノードにサービスを要求することはできません。ユーザーは他の関数を使用する前に、必ずこの関数を使用してTmaxのSRLGWと接続します。

- プロトタイプ

```
int ComOpen(int type, char *id, char *port)
```

- パラメータ

| パラメータ | 説明 |
|-------|---|
| type | Tmaxと通信が使用するプロトコル(TCPIP/SERIAL)を示します。 各プロトコルは、comhead.hファイルに登録されています |
| id | リモートノードを識別するコードです。このコードは上述したSRLGWの環境ファイルと関連しています。環境ファイルにサービス名を登録するか、またはコードを指定してリモートノードを識別するコードが登録できます。登録するコードは、Tmax環境ファイルのSERVICEセクションに登録されているサービス名である必要があります |
| port | TmaxのSRLGWと接続する際に使用するポートを指定します。シリアルで通信する場合、上述したSRLGWの環境設定に登録した各項目を指定します。 ポートに、情報をストリングで連続して指定します(例:“COM1:9600:8:ODD:1”) |

- 戻り値

| 戻り値 | 説明 |
|-----|-----------------|
| 0 | 関数呼び出しに成功した場合です |
| -1 | 関数呼び出しに失敗した場合です |

3.3. ComDetach

TmaxのSRLGWとの登録を解除するために使用する関数です。登録のみ解除して実際の接続は切断しません。ComDetach関数を使用する場合、TmaxのSRLGWとのデータ送受信ができません。

- プロトタイプ

```
int ComDetach(void)
```

- 戻り値

| 戻り値 | 説明 |
|-----|-----------------|
| 0 | 関数呼び出しに成功した場合です |
| -1 | 関数呼び出しに失敗した場合です |

3.4. ComClose

TmaxのSRLGWとの登録を解除し、接続を切断する関数です。TmaxのSRLGWとの通信を切断する場合はこの関数を呼び出します。シリアル通信の場合、SRLGWでリモートノードのプログラムのダウン有無が把握できないため、関数を利用してSRLGWとの接続を解除する必要があります。

- プロトタイプ

```
int ComClose(void)
```

- 戻り値

| 戻り値 | 説明 |
|-----|-----------------|
| 0 | 関数呼び出しに成功した場合です |
| -1 | 関数呼び出しに失敗した場合です |

3.5. ComSend

リモートノードからTmaxのSRLGWにデータを転送する関数です。この関数を利用してTmaxにサービスを要求するか応答を転送することができます。Tmaxのサービス要求に対する応答を転送する場合も、必ずTmaxの新しいサービス名を指定します。

- プロトタイプ

```
Int ComSend(char *svc, int msgtype, char *data, long len, long flags)
```

- パラメータ

| パラメータ | 説明 |
|---------|--|
| svc | Tmaxに要求するサービス名を指定します |
| msgtype | TmaxのSRLGWに転送するデータ型です。各メッセージのタイプについては、commhead.hファイルを参照してください |
| data | TmaxのSRLGWに転送するデータが保存されているバッファのポインターです |
| len | 転送するデータ長です |
| flags | サービス要求時に、SRLGWが当該要求をどのように処理するかを示します。 flagsにCOMMDELAYが設定されている場合、SRLGWは当該データを遅延処理として認識してRQに保存します。一方、COMMNOREPLYが設定されている場合は、サービスを要求するだけで、応答は受けないという意味で認識されます。 現在flagsに指定できる値は2つです。commhead.hファイルを参照してください |

- 戻り値

| 戻り値 | 説明 |
|-----|-----------------|
| 0 | 関数呼び出しに成功した場合です |
| -1 | 関数呼び出しに失敗した場合です |

3.6. ComRecv

TmaxのSRLGWからデータを受信する関数です。データはサービス要求か、または応答データです。

- プロトタイプ

```
Int ComRecv(char *svc, int *msgtype, char *data, long *len, long flags)
```

- パラメータ

| パラメータ | 説明 |
|---------|--|
| svc | Tmaxに要求したサービス名が保存されます |
| msgtype | TmaxのSRLGWが転送するデータ型を示します。各メッセージのタイプについては、commhead.hファイルを参照してください |
| data | TmaxのSRLGWから受信したデータが保存されるバッファのポインターです。このバッファは受信データの中で、最も大きいデータのバッファ・サイズを有する必要があります |
| len | 受信したデータ長です |
| flags | 使用しません |

- 戻り値

| 戻り値 | 説明 |
|-----|-----------------|
| 0 | 関数呼び出しに成功した場合です |
| -1 | 関数呼び出しに失敗した場合です |

第4章 例

本章では、SRLGWの各役割に関する例について説明します。

4.1. NOREPLYサービス呼び出し

TmaxのサービスまたはクライアントからSRLGWにtpacallでサービスを要求し、その結果を問わず処理する例です。

プログラムの構成は下記のとおりです。

| 区分 | ファイル名 |
|---------|-------------------|
| 環境ファイル | tmax.m、serial.cfg |
| リモートノード | remote.c |

4.1.1. 環境ファイル

< tmax.m >

```
*DOMAIN
res          SHMKEY = 88000,
              MINCLH = 1,
              MAXCLH = 1,
              TPORTNO = 8888

*NODE
node1        TMAXDIR = "/home/tmax",
              APPDIR = "/home/tmax/appbin"

*SVRGROUP
svg1         NODENAME = node1

*SERVER
serialgw     SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
              SVRTYPE = CUSTOM_GATEWAY,
              CLOPT = "-- -F /home/tmax/config/serial.cfg -w 5050"

svr          SVGNAME = svg1, MIN = 1, MAX = 1,
```

```
*SERVICE
COM1      SVRNAME = serialgw
TOUPPER   SVRNAME = svr
```

< serial.cfg >

```
#####
# gwno baudrate stopbit Parity Byte Port      Tmaxsvc
# 0          9600      1      ODD      8  COM1      SVC
#####
# line start with "#" is comment line
# gwno must start at 0 and be increased by 1
# stopbit: 1, 1.5, 2
# parity : NONE, ODD, EVEN
# byte: 5, 6, 7, 8
# port: serial port name
# svc:  tmax service name
#####
0          9600      1      NONE      8      COM1      COM1
```

4.1.2. リモートノード

< toupper.c >

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <usrinc/atmi.h>

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

    if (argc != 2) {
        printf("Usage: toupper string\n");
        exit(1);
    }

    if (tpstart((TPSTART_T *)NULL) == -1){
        printf("Tpstart failed\n");
        exit(1);
    }
}
```

```

        if ((sndbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
            printf("Sndbuf alloc failed !\n");
            tpend();
            exit(1);
        }

        if ((rcvbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
            printf("Rcvbuf alloc failed !\n");
            tpfree((char *)sndbuf);
            tpend();
            exit(1);
        }

        strcpy(sndbuf, argv[1]);

        if(tpcall("TOUPPER", sndbuf, 0, &rcvbuf, &rcvlen, 0)==-1){
            printf("Can't send request to service TOUPPER\n");
            tpfree((char *)sndbuf);
            tpfree((char *)rcvbuf);
            tpend();
            exit(1);
        }

        printf("Sndbuf data: %s\n", sndbuf);
        printf("Rcvbuf data: %s\n", rcvbuf);

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

```

< SVR.C >

```

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

TOUPPER(TPSVCINFO *msg)
{
    char          *sndbuf;
    long          sndlen;

    if ((sndbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
        printf("sndbuf alloc failed !\n");
        tpreturn(TPFAIL, -1, NULL, 0, 0);
    }

    strcpy(sndbuf, msg->data);
}

```

```

        sndlen = msg->len;

        if(tpacall("COM1", sndbuf, sndlen, TPNOREPLY)==-1){
            printf("Can't send request to service TCPSVC1\n");
            tpfree((char *)sndbuf);
            tpreturn(TPFAIL, -1, NULL, 0, 0 );
        }

        tpfree((char *)sndbuf);
        tpreturn(TPSUCCESS,0,(char *)msg->data, msg->len,0);
    }
}

```

< remote.c >

```

#include <stdio.h>
#include <winsock2.h>
#include <usrinc/commhead.h>

#define COM_PORT    "COM1:9600:8:NONE:1"

int main(int argc, char *argv[])
{
    int    n, msgtype;
    long   ilen, olen;
    char   buffer[1024];
    char   svcname[24];

    /* ゲートウェイにリモートノードを登録します。 */
    printf("Remote Service Start...\n\n");
    n = ComOpen(SERIAL, "COM1", COM_PORT);
    if (n < 0) {
        perror("Gateway register error:");
        return -1;
    }

    while(1) {
        memset(svcname, 0x00, sizeof(svcname));
        memset(buffer, 0x00, sizeof(buffer));

        /* データを受信します。 */
        n = ComRecv(svcname, &msgtype, buffer, &olen, COMMNOFLAG);
        if (n < 0) {
            perror("Data Receive error:");
            ComClose();
            return -1;
        }

        printf("REMOTE RECV : svcname = [%s]\n", svcname);
    }
}

```

```

        printf("REMOTE RECV : len = %d\n", olen);
        printf("REMOTE RECV : data= [%s]\n\n", buffer);
    }

    /* ゲートウェイとの接続を解除します。 */
    ComClose( );
}

```

4.2. REPLYサービス呼び出し

TmaxのサービスまたはクライアントからSRLGWにtpacallでサービスを要求し、結果を問わず処理します。リモートノードで、サービス要求に対する処理結果をTmaxの別のサービスで処理できるようにTOLOWERを呼び出す例です。

プログラムの構成は下記のとおりです。

| 区分 | ファイル名 |
|---------|-------------------|
| 環境ファイル | tmax.m、serial.cfg |
| リモートノード | remote.c |

4.2.1. 環境ファイル

< tmax.m >

```

*DOMAIN
res          SHMKEY = 88000,
              MINCLH = 1,
              MAXCLH = 1,
              TPORTNO = 8888

*NODE
node1        TMAXDIR = "/home/tmax",
              APPDIR = "/home/tmax/appbin"

*SVRGROUP
svg1         NODENAME = node1

*SERVER
serialgw     SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
              SVRTYPE = CUSTOM_GATEWAY,
              CLOPT = "-- -F /home/tmax/config/serial.cfg -w 5050"
svr          SVGNAME = svg1, MIN = 1, MAX = 1,

```

```
*SERVICE
COM1          SVRNAME = serialgw
TOUPPE        SVRNAME = svr
TOLOWER       SVRNAME = svr
```

< serial.cfg >

```
#####
# gwno baudrate stopbit Parity Byte Port      Tmaxsvc
# 0          9600      1      ODD      8  COM1      SVC
#####
# line start with "#" is comment line
# gwno must start at 0 and be increased by 1
# stopbit: 1, 1.5, 2
# parity : NONE, ODD, EVEN
# byte: 5, 6, 7, 8
# port: serial port name
# svc:  tmax service name
#####
0          9600      1      NONE      8      COM1      COM1
```

4.2.2. リモートノード

< toupper.c >

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <usrinc/atmi.h>

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

    if (argc != 2) {
        printf("Usage: toupper string\n");
        exit(1);
    }

    if (tpstart((TPSTART_T *)NULL) == -1){
        printf("Tpstart failed\n");
        exit(1);
    }
}
```



```

        if ((sndbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
            printf("Sendbuf alloc failed !\n");
            tpend();
            exit(1);
        }

        if ((rcvbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
            printf("Recvbuf alloc failed !\n");
            tpfree((char *)sndbuf);
            tpend();
            exit(1);
        }

        strcpy(sndbuf, argv[1]);

        if(tpcall("TOUPPER", sndbuf, 0, &rcvbuf, &rcvlen, 0)==-1){
            printf("Can't send request to service TOUPPER\n");
            tpfree((char *)sndbuf);
            tpfree((char *)rcvbuf);
            tpend();
            exit(1);
        }

        printf("Sendbuf data: %s\n", sndbuf);
        printf("Recvbuf data: %s\n", rcvbuf);

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

```

< SVR.C >

```

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

TOUPPER(TPSVCINFO *msg)
{
    char          *sndbuf;
    long          sndlen;

    if ((sndbuf = (char *)tpalloc("STRING", NULL, 0)) == NULL) {
        printf("sendbuf alloc failed !\n");
        tpreturn(TPFAIL, -1, NULL, 0, 0);
    }
}

```

```

    }

    strcpy(sndbuf, msg->data);
    sndlen = msg->len;

    if(tpacall("COM1", sndbuf, sndlen, TPNOREPLY)==-1){
        printf("Can't send request to service TCPSVC1\n");
        tpfree((char *)sndbuf);
        tpreturn(TPFAIL, -1, NULL, 0, 0 );
    }

    tpfree((char *)sndbuf);
    tpreturn(TPSUCCESS,0,(char *)msg->data, msg->len,0);
}

TOLOWER(TPSVCINFO *msg)
{
    int                i;

    printf("TOLOWER service is started!\n");
    printf("INPUT : data=%s\n", msg->data);

    for (i = 0; i < msg->len; i++)
        msg->data[i] = tolower(msg->data[i]);

    printf("OUTPUT: data=%s\n\n", msg->data);

    tpreturn(TPSUCCESS,0,(char *)msg->data, 0,0);
}

```

< remote.c >

```

#include <stdio.h>
#include <winsock2.h>
#include <usrinc/commhead.h>

#define COM_PORT    "COM1:9600:8:NONE:1"

int main(int argc, char *argv[])
{
    int    n, msgtype;
    long   ilen, olen;
    char   buffer[1024];
    char   svcname[24];

```

```

/* ゲートウェイにリモートノードを登録します。 */
printf("Remote Service Start...\n\n");
n = ComOpen(SERIAL, "COM1", COM_PORT);
if (n < 0) {
    perror("Gateway register error:");
    return -1;
}

while(1) {
    memset(svcname, 0x00, sizeof(svcname));
    memset(buffer, 0x00, sizeof(buffer));

    /* データを受信します。 */
    n = ComRecv(svcname, &msgtype, buffer, &olen, COMMNOFLAG);
    if (n < 0) {
        perror("Data Receive error:");
        ComClose();
        return -1;
    }
    printf("REMOTE RECV : svcname = [%s]\n", svcname);
    printf("REMOTE RECV : len = %d\n", olen);
    printf("REMOTE RECV : data= [%s]\n\n", buffer);

    strcpy(buffer, "I'M REMOTE SERVICE.");
    ilen = strlen(buffer) + 1;

    /* 応答データを転送します。 */
    n = ComSend("TOLOWER", COMMDATA, buffer, ilen, COMMNOREPLY);
    if (n < 0) {
        perror("Data send error:");
        ComClose();
        return -1;
    }
}

/* ゲートウェイとの接続を解除します。 */
ComClose( );
}

```

4.3. リモートノード同期型呼び出し

Tmaxのブート時にSRLGWが起動されており、リモートノードからの要求が受信されたら、ユーザーが指定したサービスを呼び出した後、再びリモートノードに処理結果を転送する例です。

プログラムの構成は下記のとおりです。

| 区分 | ファイル名 |
|---------|-------------------|
| 環境ファイル | tmax.m、serial.cfg |
| リモートノード | remote.c |

4.3.1. 環境ファイル

< tmax.m >

```
*DOMAIN
res          SHMKEY = 88000,
              MINCLH = 1,
              MAXCLH = 1,
              TPORTNO = 8888

*NODE
node1        TMAXDIR = "/home/tmax",
              APPDIR = "/home/tmax/appbin

*SVRGROUP
svg1         NODENAME = node1

*SERVER
serialgw     SVGNAME = svg1, MIN = 1, MAX = 1, CPC = 10,
              SVRTYPE = CUSTOM_GATEWAY,
              CLOPT = "-- -F /home/tmax/config/serial.cfg -w 5050"
svr          SVGNAME = svg1, MIN = 1, MAX = 1

*SERVICE
COM1         SVRNAME = serialgw
TOUPPER      SVRNAME = svr
```

< serial.cfg >

```
#####
# gwno baudrate stopbit Parity Byte Port    Tmaxsvc
# 0      9600      1      ODD      8  COM1      SVC
#####
# line start with "#" is comment line
# gwno must start at 0 and be increased by 1
# stopbit: 1, 1.5, 2
# parity : NONE, ODD, EVEN
# byte: 5, 6, 7, 8
# port: serial port name
```

```
# svc:  tmax service name
#####
0      9600    1      NONE    8      COM1    COM1
```

4.3.2. リモートノード

< SVR.C >

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

TOUPPER(TPSVCINFO *msg)
{
    int      i;
    char      *sndbuf;

    printf("TOUPPER service is started!\n");
    printf("INPUT : data=%s, len = %d\n", msg->data, msg->len);

    for (i = 0; i < msg->len; i++)
        msg->data[i] = toupper(msg->data[i]);
    printf("OUTPUT: data=%s, len = %d\n", msg->data, msg->len);
    tpreturn(TPSUCCESS, 0, (char *)msg->data, msg->len, 0);
}
```

< remote.c >

```
#include <stdio.h>
#include <winsock2.h>
#include <usrinc/commhead.h>

#define COM_PORT    "COM1:9600:8:NONE:1"

int main(int argc, char *argv[])
{
    int      n, msgtype;
    long     ilen, olen;
    char     buffer[1024];
    char     svcname[20];
    char     *ptr;

    /* ゲートウェイにリモートノードを登録します。*/
```

```

printf("Remote Service Start...\n\n");
n = ComOpen(SERIAL, "COM1", COM_PORT);
if (n < 0) {
    perror("Gateway register error:");
    return -1;
}

ptr = argv[1];
ilen = strlen(argv[1]) + 1;
printf("Send Data = [%s], len = %d\n",ptr,ilen);

n = ComSend("TOUPPER", COMMDATA, ptr, ilen, COMMNOFLAG);
if (n < 0) {
    perror("Data send error:");
    ComClose();
    return -1;
}

memset(svcname, 0x00, sizeof(svcname));
memset(buffer, 0x00, sizeof(buffer));

/* 応答データを受信します。 */
n = ComRecv(svcname, &msgtype, buffer, &olen, COMMNOFLAG);
if (n < 0) {
    perror("Data Receive error:");
    ComClose();
    return -1;
}

printf("REMOTE RECV : svcname = [%s]\n", svcname);
printf("REMOTE RECV : len = %d\n",olen);
printf("REMOTE RECV : data= [%s]\n\n", buffer);

/* ゲートウェイとの接続を解除します。 */
ComClose( );
}

```

4.4. ACK/NAKの通信方式

Tmaxのブート時にSRLGWが起動されており、リモートノードからの要求が受信されたら、ユーザーが指定したサービスを呼び出した後、再びリモートノードに処理結果を転送する例です。この場合、送受信データに対してACK/NAKを処理する例です。

プログラムの構成は下記のとおりです。

| 区分 | ファイル名 |
|---------|-------------------|
| 環境ファイル | tmax.m、serial.cfg |
| リモートノード | remote.c |

4.4.1. 環境ファイル

< tmax.m >

```
*DOMAIN
res          SHMKEY = 88000,
              MINCLH = 1,
              MAXCLH = 1,
              TPORTNO = 8888

*NODE
node1        TMAXDIR = "/home/tmax",
              APPDIR = "/home/tmax/appbin"

*SVRGROUP
svg1         NODENAME = node1

*SERVER
Serialgw     SVGNAME = svg1, MIN = 1,MAX = 1,CPC = 10,
              SVRTYPE = CUSTOM_GATEWAY,
              CLOPT = "-- -F /home/tmax/config/serial.cfg -w 5050 -A"
avr          SVGNAME = svg1, MIN = 1, MAX = 1

*SERVICE
COM1         SVRNAME = serialgw
TOUPPER      SVRNAME = svr
```

< serial.cfg >

```
#####
# gwno baudrate stopbit Parity Byte Port    Tmaxsvc
# 0      9600      1      ODD      8  COM1      SVC
#####
# line start with "#" is comment line
# gwno must start at 0 and be increased by 1
# stopbit: 1, 1.5, 2
# parity : NONE, ODD, EVEN
# byte: 5, 6, 7, 8
# port: serial port name
# svc:  tmax service name
```

```
#####  
0          9600      1          NONE      8          COM1      COM1
```

4.4.2. リモートノード

< SVR.C >

```
#include <stdio.h>  
#include <usrinc/atmi.h>  
  
TOUPPER(TPSVCINFO *msg)  
{  
    int      i;  
    char     *sndbuf;  
  
    printf("TOUPPER service is started!\n");  
    printf("INPUT : data=%s, len = %d\n", msg->data, msg->len);  
  
    for (i = 0; i < msg->len; i++)  
        msg->data[i] = toupper(msg->data[i]);  
    printf("OUTPUT: data=%s, len = %d\n", msg->data, msg->len);  
    tpreturn(TPSUCCESS, 0, (char *)msg->data, msg->len, 0);  
}
```

< remote.c >

```
#include <stdio.h>  
#include <winsock2.h>  
#include <usrinc/commhead.h>  
  
#define COM_PORT    "COM1:9600:8:NONE:1"  
  
int main(int argc, char *argv[])  
{  
  
    int    n, msgtype;  
    long   ilen, olen;  
    char   buffer[1024];  
    char   svcname[20];  
    char   *ptr;  
  
    /* ゲートウェイにリモートノードを登録します。 */  
    printf("Remote Service Start...\n\n");
```



```

n = ComOpen(SERIAL, "COM1", COM_PORT);
if (n < 0) {
    perror("Gateway register error:");
    return -1;
}

while(1) {
    ptr = argv[1];
    ilen = strlen(argv[1]) + 1;
    printf("Send Data = [%s], len = %d\n",ptr,ilen);

    n = ComSend("TOUPPER", COMMDATA, ptr, ilen, COMMNOFLAG);
    if (n < 0) {
        perror("Data send error:");
        ComClose();
        return -1;
    }

    memset(svcname, 0x00, sizeof(svcname));
    memset(buffer, 0x00, sizeof(buffer));

    /* ACK/NAK受信 */
    n = ComRecv(svcname, &msgtype, buffer, &olen, COMMNOFLAG);
    if (n < 0) {
        perror("Data Receive error:");
        printf("RT = %d\n",n);
        ComClose();
        return -1;
    }
    if (msgtype == COMMACK)
        break;
}
printf("REMOTE RECV OK! : ACK msgtype = [%d]\n", msgtype);

/* ACK/NAK送信 */
n = ComSend("TOUPPER", COMMACK, NULL, 0, COMMNOFLAG);
if (n < 0) {
    perror("Data send error:");
    ComClose();
    return -1;
}
printf("REMOTE SEND OK! : Return = [%d]\n", n);

/*ゲートウェイとの接続を解除します。 */
ComClose( );
}

```


付録 A. ヘッダー・ファイル

A.1. COMMHEADファイル

```
#ifndef _COMM_H_
#define _COMM_H_

#ifndef _WIN32
#define __cdecl
#endif

#ifndef _LOBYTE
#if (defined(_SOCK1) || defined(_SOCK11))
#define _LOBYTE 1
#define _HIBYTE 1
#else
#define _LOBYTE 2
#define _HIBYTE 0
#endif
#endif

/*  type */
#define TCPIP          1
#define SERIAL         2

/*  flags */
#define COMMNOFLAG      0x00
#define COMMDELAY      0x01
#define COMMNOREPLY    0x10

/*  msgtype */
#define COMMDATA        1
#define COMMACK        2
#define COMMNAK        3
#define COMMREGISTER   4
#define COMMUNREGISTER 5

define COMM_MSG_SIZE    1024
#define COMM_DATAP(cp) ((char *)((struct comm_header_t *)cp+1))
#define COMM_HEADER_SIZE sizeof(struct comm_header_t)
#define COMM_LEN_SIZE    4
#define COMM_UID_SIZE    5
```

```

#define COMM_SVC_SIZE          16
#define COMM_ERRCODE_SIZE     4

struct comm_header_t {
    char  len[COMM_LEN_SIZE]; /* data length: データ長 */
    char  msgtype;             /* Message type */
    char  uid[COMM_UID_SIZE]; /* Unique Value: ACK, NAKで使用*/
    char  svc[COMM_SVC_SIZE]; /* Destination Id. */
    char  errcode[COMM_ERRCODE_SIZE]; /* 未使用: 次回使用する予定 */
    char  flags;               /* 上記で指定したflagのうち1つ */
    unsigned char crc;         /* Check Byte */
};

typedef struct comm_header_t  COMM_HEADER;

#ifdef __cplusplus
extern "C" {
#endif

/* ----- function prototypes ----- */
int __cdecl ComOpen(int type, char *id, char *port);
int __cdecl ComDetach(void);
int __cdecl ComClose(void);
int __cdecl ComSend(char *svc, int msgtype, char *data, long len, long flags);
int __cdecl ComRecv(char *svc, int *msgtype, char *data, long *len, long flags);

#ifdef __cplusplus
}
#endif
#endif /* _COMM_H_ */

```

索引

A

ACK/NAKの通信方式の例, 28
ACK/NAK応答のSRLGW, 9

C

COMライブラリー, 1
 ComClose(), 14
 ComDetach(), 14
 ComOpen(), 13
 ComRecv(), 16
 ComSend(), 15

N

NOREPLYサービス呼び出し, 3
NOREPLYサービス呼び出しの例, 17

R

REPLYサービス呼び出し, 4
REPLYサービス呼び出しの例, 21

S

SERIAL Gateway, 1
SRLGW, 1
SRLGWの動作構造, 1
SRLGWの環境ファイル, 10
 アドレス情報の環境ファイル, 10

T

Tmaxからのサービス要求
 NOREPLYサービス呼び出し, 3
 REPLYサービス呼び出し, 4
Tmax環境ファイル, 7
Tmax環境構成
 ACK/NAK応答のSRLGW, 9
 CLOPT項目(SRLGWオプション), 8

遅延処理のSRLGW, 10

Tmax環境構成(CLOPT項目)

[-A], 9
[-F], 8
[-q], 9
[-t], 9
[-w], 9

あ

アウトバウンド・サービス, 2
インバウンド。サービス, 2

た

同期型呼び出し方式, 5

は

非同期型呼び出し方式, 6

ら

例

 ACK/NAKの通信方式, 28
 NOREPLYサービス呼び出し, 17
 REPLYサービス呼び出し, 21
 リモートノード同期型呼び出し, 25
リモートノードからのサービス要求
 同期型呼び出し方式, 5
 非同期型呼び出し方式, 6
リモートノード同期型呼び出しの例, 25

