

# Tmax 運用ガイド

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 運用ガイド

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 .....	xi
<b>第1章 紹介 .....</b>	<b>1</b>
1.1. 概要 .....	1
1.2. 内部構造 .....	2
1.3. システム構成 .....	4
1.4. システム管理 .....	6
1.4.1. 静的管理 .....	6
1.4.2. 動的管理 .....	7
1.5. ディレクトリーの構成 .....	7
<b>第2章 環境変数の設定 .....</b>	<b>11</b>
2.1. 概要 .....	11
2.2. 環境変数 .....	11
2.2.1. サーバーの環境変数 .....	11
2.2.2. クライアントの環境変数 .....	15
2.3. 環境変数の設定方法 .....	18
2.3.1. サーバー環境変数の設定 .....	18
2.3.2. クライアント環境変数の設定 .....	19
2.4. 多数のサーバー情報の登録 .....	20
<b>第3章 環境ファイルの設定 .....</b>	<b>23</b>
3.1. 概要 .....	23
3.1.1. 環境ファイルの形式 .....	24
3.1.2. 環境ファイルの作成 .....	26
3.2. 基本環境設定 .....	27
3.2.1. DOMAINセクション .....	27
3.2.2. NODEセクション .....	50
3.2.3. SVRGROUPセクション .....	87
3.2.4. SERVERセクション .....	103
3.2.5. SERVICEセクション .....	123
3.2.6. GATEWAYセクション .....	128
3.2.7. ROUTINGセクション .....	146
3.2.8. RQセクション .....	146
3.2.9. HMSセクション .....	146
3.2.10. 基本環境設定の例 .....	146
3.3. データベースの環境設定 .....	147
3.3.1. SVRGROUPセクション .....	148
3.3.2. データベース環境設定の例 .....	150
3.4. 分散トランザクションの環境設定 .....	151
3.4.1. DOMAINセクション .....	152
3.4.2. NODEセクション .....	154

3.4.3.	SVRGROUPセクション .....	155
3.5.	負荷調整の環境設定 .....	157
3.5.1.	システム性能による負荷調整 .....	158
3.5.2.	データ値による負荷調整 .....	161
3.6.	信頼性キューの環境設定 .....	168
3.7.	HMS環境設定 .....	172
3.7.1.	DOMAINセクション .....	173
3.7.2.	NODEセクション .....	173
3.7.3.	SVRGROUPセッション .....	174
3.7.4.	HMSセクション .....	177
3.8.	障害対策の環境設定 .....	179
3.8.1.	ハードウェア的な障害 .....	180
3.8.2.	ソフトウェア的な障害 .....	180
3.9.	セキュリティの環境設定 .....	183
3.9.1.	段階別のセキュリティ設定 .....	183
3.9.2.	その他のセキュリティ設定 .....	185
3.9.3.	サードパーティー・セキュリティ設定 .....	187
3.10.	マルチドメインの環境設定 .....	188
3.10.1.	DOMAINセクション .....	188
3.10.2.	NODEセクション .....	189
3.10.3.	SVRGROUPセクション .....	189
3.10.4.	SERVERセクション .....	189
3.10.5.	SERVICEセクション .....	189
3.10.6.	GATEWAYセクション .....	190
3.10.7.	ROUTINGセクション .....	190
3.11.	Tmax環境ファイルのコンパイル .....	193
3.12.	サービス・テーブルの作成 .....	195
3.13.	サービス・タイムアウトの監視 .....	196
3.13.1.	tmapm .....	196
<b>第4章</b>	<b>起動と終了 .....</b>	<b>199</b>
4.1.	Tmaxの起動 .....	199
4.1.1.	racd .....	199
4.1.2.	tmboot .....	202
4.2.	Tmaxの終了 .....	208
4.2.1.	tmdown .....	209
<b>第5章</b>	<b>管理ツール .....</b>	<b>215</b>
5.1.	概要 .....	215
5.2.	tadmin .....	215
5.3.	環境情報の照会 .....	219
5.3.1.	tmaxinfo(ti) .....	219
5.3.2.	history .....	220
5.3.3.	config(cfg) .....	220

5.3.4.	configopt(cfgopt)	229
5.4.	動作状態情報	230
5.4.1.	stat(st)	230
5.4.2.	gwinfo	247
5.4.3.	txgwinfo / nontxgwinfo	248
5.4.4.	jgwinfo / ajgwinfo	250
5.4.5.	wsgwinfo	251
5.4.6.	smtrc	252
5.4.7.	clhsinfo	254
5.4.8.	tmmsinfo	255
5.4.9.	repeat(r)	256
5.4.10.	clientinfo	257
5.4.11.	svrinfo(si)	258
5.4.12.	txquery(txq)	259
5.4.13.	rqstat(rqs)	262
5.5.	運用管理	264
5.5.1.	suspend(sp)	264
5.5.2.	resume(rs)	266
5.5.3.	advertise/unadvertise	267
5.5.4.	restat	269
5.5.5.	rebootsvr(rbs)	270
5.5.6.	cfgadd(ca)	274
5.5.7.	set	281
5.5.8.	setopt	282
5.5.9.	qpurge(qp)	283
5.5.10.	discon(ds)	284
5.5.11.	logstart/logend	284
5.5.12.	chtrc	285
5.5.13.	chlog	286
5.5.14.	chlog2	288
5.5.15.	txcommit/txrollback	290
5.5.16.	wsgwreload	291
5.5.17.	restart	291
5.5.18.	notify_reconnect_clh(nrc)	292
5.5.19.	admnoti(an)	293
<b>第6章</b>	<b>IPv6の設定</b>	<b>295</b>
6.1.	概要	295
6.1.1.	基本概念	295
6.1.2.	移行技術	295
6.2.	TmaxのIPv6対応	296
6.2.1.	対応機能	296
6.2.2.	付加機能	299

<b>付録 A. ゲートウェイのCLOPTセクション .....</b>	<b>301</b>
A.1. Tmax .....	301
A.1.1. Tmaxトランザクション・ドメイン・ゲートウェイ .....	301
A.1.2. Tmax非トランザクション・ドメイン・ゲートウェイ .....	302
A.2. Java .....	303
A.2.1. JEUSゲートウェイ .....	303
A.2.2. JEUS Asyncゲートウェイ .....	303
A.3. Tuxedo .....	304
A.3.1. Tuxedoゲートウェイ .....	304
A.3.2. Tuxedo Asyncゲートウェイ .....	306
<b>付録 B. 使用時の参考事項 .....</b>	<b>307</b>
B.1. 複数のCLHの使用 .....	307
B.1.1. ASQCOUNT .....	307
B.1.2. 同時スケジューリング .....	307
B.2. ドメインゲートウェイCOUSINの設定方法 .....	307
B.2.1. SVRGROUPセクション .....	308
B.2.2. GATEWAYセクション .....	308
B.3. ドメイン・ゲートウェイの知能型ルーティング .....	312
B.3.1. 従来のドメイン・ゲートウェイのルーティング .....	312
B.3.2. 現在のドメイン・ゲートウェイのルーティング .....	313
B.3.3. 知能型ルーティング機能に対応するゲートウェイ .....	314
B.4. 各バージョンにおけるFDの計算方法 .....	314
<b>索引 .....</b>	<b>315</b>



# 図目次

[図 1.1]	Tmaxの役割 .....	1
[図 1.2]	Tmaxの構造 .....	2
[図 1.3]	クライアントとTmaxシステム間の関係 .....	5
[図 1.4]	Tmaxシステム構成図 .....	5
[図 3.1]	環境ファイルの構成要素間の相関関係 .....	24
[図 3.2]	MINCLH = 2の場合 .....	31
[図 3.3]	TRB .....	70
[図 3.4]	LOAD項目が-1の場合の動作原理 .....	90
[図 3.5]	LOAD項目が-2の場合の動作原理 .....	91
[図 3.6]	LOAD項目が正数の場合の動作原理 .....	93
[図 3.7]	データベース管理のためのTMSプロセス .....	148
[図 3.8]	分散トランザクション処理環境 .....	152
[図 3.9]	2PCの処理プロセス .....	153
[図 3.10]	システム・ロードによる負荷調整 .....	158
[図 B.1]	従来のドメイン・ゲートウェイのルーティング動作 .....	312
[図 B.2]	現在のドメイン・ゲートウェイのルーティング動作(1) .....	313
[図 B.3]	現在のドメイン・ゲートウェイのルーティング動作(2) .....	313



# このガイドについて

## 対象読者

本書は、Tmax<sup>®</sup>(以下、Tmax)を使用してプログラムを開発するユーザーと管理者を対象としています。

## 前提知識

本書は、Tmaxシステムについての全般的な理解とTmaxシステムが提供する各種機能および特性を習得するための基本ガイドです。

同書を理解するには、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
- Tmaxの基本概念
- JavaおよびCプログラミング

## 制限事項

本書ではTmax全体の内容を取り上げてはおりません。

---

### 参考

Tmaxシステムの開発に関する基本的な内容は、『Tmax スタートガイド』および『Tmax アプリケーション開発ガイド』を参照してください。Tmaxが提供するコマンドとC APIについての説明は、『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6章と2つの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: 概要

Tmaxの内部構造、システム構成、管理方法、ディレクトリー環境について説明します。

- 第2章: 環境変数の設定

Tmaxシステムで使用する環境変数の種類と役割、および設定方法について説明します。

- 第3章: 環境ファイルの設定

Tmaxシステムの環境ファイルの種類と役割、および設定方法について説明します。

- 第4章: 起動と終了

Tmaxを起動および終了する方法について説明します。

- 第5章: 管理ツール

tmaxadminツールの使用方法について説明します。

- 第6章: IPv6の設定

Tmaxが提供するIPv6の概念とサポート範囲について説明します。

- 付録A.: ゲートウェイのCLOPTセクション

Tmaxドメイン・ゲートウェイ、Javaゲートウェイ、TuxedoゲートウェイのCLOPTセクションの設定オプションについて説明します。

- 付録B.: 使用時の参考事項

Tmaxを使用する際に参考になる事項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、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
<b>参考</b>	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

## 関連文書

ガイド	説明
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 HMSユーザガイド	TmaxのHMS(Hybrid Messaging System)の基本概念と環境設定、APIの使用方法および使用例について説明しています
Tmax プログラミングガイド(ダイナミックライブラリ)	TmaxのTDL(Tmax Dynamic Library)を使用してプログラムを開発するユーザー向けに、TDLを使用するための環境設定と提供されるAPIの使用方法について説明しています
Tmax プログラミングガイド(4GL)	4GL言語を使用してTmaxアプリケーションを開発するユーザー向けに、言語別インターフェースと関連関数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
Tmax WebTAsyncユーザガイド	TmaxのAsync Java Gatewayとインバウンドおよびアウトバウンド通信を非同期で行うためのJavaライブラリーであるWebTAsync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](mailto: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](mailto:info@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/us\\_en/home](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](mailto: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](mailto:info@tmaxsoft.com.cn)  
Web (Chinese): [http://www.tmaxsoft.com/cn\\_en/home\\_cn\\_en](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](mailto:info.bra@tmaxsoft.com)  
Web (Portuguese): [http://www.tmaxsoft.com/br\\_en/home\\_br\\_en](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](mailto:info.rus@tmaxsoft.com)  
Web (Russian): [http://www.tmaxsoft.com/ru\\_ru/home\\_ru\\_ru](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](mailto:info.sg@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/sg\\_en/home\\_sg\\_en](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](mailto:info.uk@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/gb\\_en/home\\_gb\\_en](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](mailto:info.canada@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/ca\\_en/home\\_ca\\_en](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](mailto:info.aus@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/au\\_en/home\\_au\\_en](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](mailto:info.india@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/in\\_en/home\\_in\\_en](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](mailto:cslee@tmaxsoft.com)  
Web (English): [http://www.tmaxsoft.com/tr\\_en/home\\_tr\\_en](http://www.tmaxsoft.com/tr_en/home_tr_en)



# 第1章 紹介

本章ではTmaxの概要、内部構造、システム構成などについて説明します。

## 1.1. 概要

クライアント/サーバー(Client/Server)中心のコンピューティング環境下で、クライアント数の増加によるマシン(Machine)の多様化、オペレーティングシステムとデータベースの混在、データ互換性の欠如、サーバー性能の低下などの問題が発生しています。

Tmaxはこのような問題を解決し、プロセス管理やトランザクション管理、負荷調整、異機種間のリソース管理などの機能を提供する、トランザクション処理用のミドルウェアです。

[図 1.1] Tmaxの役割



- プロセス管理

サーバー・プロセスの起動、業務の分散、プロセスの自動生成などの実行をモニタリングします。

- トランザクション管理

ACID特性を保証するよう、トランザクションを処理します。

- 負荷調整

システム性能を最適化して一定の応答時間を誘導します。

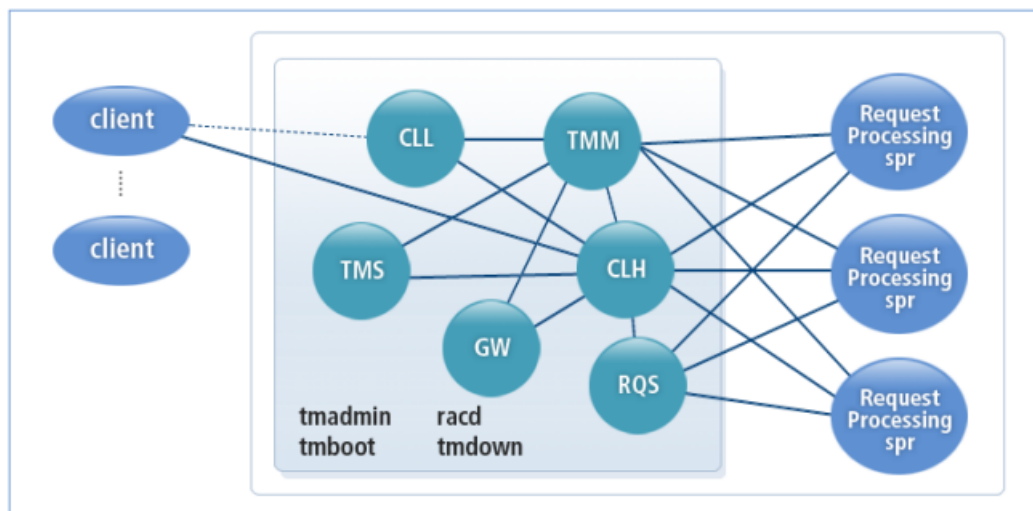
- 異種リソース管理

異機種間で発生する可能性のある様々な問題を解決します。

## 1.2. 内部構造

Tmax内部には、9つの運用プロセス(TMM、TMS、CLL、CLH、RQS、GW、CAS、TLM、HMS)と4つの管理コマンド(tmadmin、racd、tmboot、tmdown)が存在します。

[図 1.2] Tmaxの構造



- 運用プロセス

- TMM (Tmax Manager)

Tmaxの全体システムを運用するプロセスです。システムの運用情報を管理し、CLL、CLH、TMS、RQSプロセス、および業務処理サーバー・プロセスを管理します。TMMはTmaxシステムの起動時に最初にメモリにロードされ、システム終了時に最後に終了されます。

Tmaxシステムがマルチノードで構成された場合、他ノードのTMMとセッションを維持しながら、ハートビートをチェックしサーバー・グループ単位のフェイルオーバー対策をサポートします。また、すべての共有情報の更新と各プロセスのログ記録を担当します。ディスクフルなどの原因によってログを書き込みできない場合、ログの追加記録を中断し、問題が解決するまで正常な運用を可能にします。

- CLL (Client Listener)

クライアントとTmax間の接続を管理するリスナープロセスです。クライアントが最初にTmaxに接続するとCLLに接続されて通信が行われます。クライアントからサービス要求がある場合、内部でCLHと接続されすべてのサービスを処理します。

- CLH (Client Handler)

クライアント・マネージャともいい、実際にクライアントとサーバーの業務処理プロセスを中継するプロセスです。CLHは業務処理サーバー・プロセスとゲートウェイ、RQプロセス、Tmaxプロセスに接続してデータフローを管理します。クライアントのサービス要求を受信して業務を処理し、その結果をクライアントに再び転送します。

– TMS (Transaction Management Server)

Tmaxシステムの運用に必須なTMM、CLL、CLHプロセスとは違い、TMSはデータベース管理および分散トランザクション処理を担当するプロセスです。該当データベースのライブラリーを利用して作成されます。

– RQS (Reliable Queue Server)

Tmaxシステムのディスクキューを管理するプロセスです。一般的にデータはメモリーに保存され処理されますが、予想外のシステム障害が発生した場合、メモリーにあるデータは削除されます。RQSはデータの信頼度を高めるためにサービス要求前後のデータをディスクに保存します。システムの再起動後、障害が起きる前に保存されたデータを使用して要求されたサービスを再処理することができます。

サービスの実行速度を低下させる可能性があるため、基本的には動作しません。ユーザーが必要に応じて環境ファイルに定義したシステムでのみ動作します。

– GW (Gateway)

マルチドメインでシステムを構築した場合にドメイン間の通信を担当するプロセスです。Tmaxシステムだけでなく、TCP/IP、SNA/LU0、SNA/LU6、X.25など、様々なシステムと連動できます。

実際のプロセス名はユーザーが環境ファイルに定義した内容によって異なり、ゲートウェイの種類によってはさらに追加設定が必要な場合もあります。このプロセスもTmaxシステムで基本的に動作するプロセスではなく、環境ファイルのGATEWAYセクションに定義されたシステムでのみ動作します。

– CAS (Client Authentication Server)

セキュリティが必要なTmaxシステムを運用する場合に使われるプロセスです。第1段階および第2段階のセキュリティを担当し、ユーザー権限を確認する機能をします。

– TLM (Transaction Log Manager)

従来のTMM(4.0以前)からトランザクションのロギング機能を分離してTLMが担当するようにしました。従来のロギング機能に加えて、待機トランザクション(Pending Transaction)に対する監視機能も行います。

– HMS (Hybrid Messaging System)

送信者(Sender)と受信者(Receiver)間の緩結合(loosely coupled)を可能にする通信媒体です。メッセージングシステムを使用して、送信者と受信者はお互いの情報を知らなくても仮想チャンネルのデスティネーション(Destination)の情報だけで相互通信ができます。また、送信時点と受信時点を分離することができ、データに対する遅延処理が可能です。そのために、メッセージング・システムはメッセージの送受信の信頼性(Reliability)を保証します。

- 管理コマンド

- tmadmin (Tmax Administration)

tmadminは管理ツールとして、運用中のTmaxのシステム環境を確認し動的に変更する機能を提供します。現在のサーバー・プロセスの動作状態と統計情報、サービスやキューの状態情報も確認できます。

- racd (Remote Access Control Daemon)

racdは、複数のノードを1つのドメインにしてTmaxシステムを構築した場合、1つのノードで他のシステムを管理できるようにそれぞれのノードに起動させるデーモン・プロセスです。racdを使用するとドメイン内の1ノードでtmadminを介して全体ノードをモニタリングできます。ファイルをいちいちコピーしなくても環境ファイルをすべてのノードに適用できます。

- tmboot (Tmax System Boot)

tmbootは環境ファイルに定義されたとおりTmaxシステムを初期化して開始させます。まず、一般的にTmaxシステムプロセス(TMM、CLL、CLH)が先に起動し、続いて追加環境設定のためのRQ、TMS、GWプロセスが起動します。そして最後に、サーバー・プロセスが起動します。tmbootは多様なオプションを提供し、多様な方法でTmaxシステムを起動できます。

- tmdown (Tmax System Down)

tmdownは環境ファイルに基づいてtmbootで起動させたシステムを終了させるコマンドです。終了手順はtmbootと逆であり、まずすべてのサーバー・プロセスを終了させた後、Tmaxシステム管理プロセスを終了させます。

## 1.3. システム構成

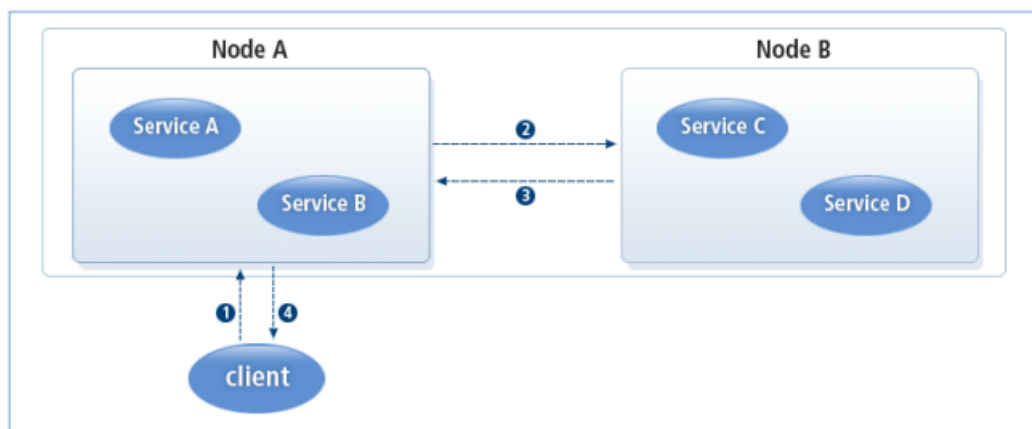
**ドメイン(Domain)**は1つの同一の環境ファイルを共有するTmaxシステムの構成単位です。ドメインは1つ以上のノード(machine)で構成できます。これらノードはpeer-to-peer方式で接続され、一定の時間間隔で他のノードと通信し、環境ファイルに構成された情報を共有します。このような方法でそれぞれのノードはドメイン全体の状態とドメイン内の他のノードの情報を共有しています。そのため、クライアントはドメイン内のどのノードに接続してもドメインが提供するすべてのサービスを利用できます。

クライアントはドメインを構成する特定ノードに接続できます。接続されたノードは当該クライアントのサービスを担当します。クライアントが要求したサービスが接続中のノードから提供するものであれば、接続ノードがサービスを処理します。一方、他のノードが提供するサービスであれば、クライアントに接続中のノードがサービスを提供するノードに要求し処理結果をクライアントに返します。

他のノードで提供されるサービスをクライアントが要求する場合は、以下のフローで処理されます。



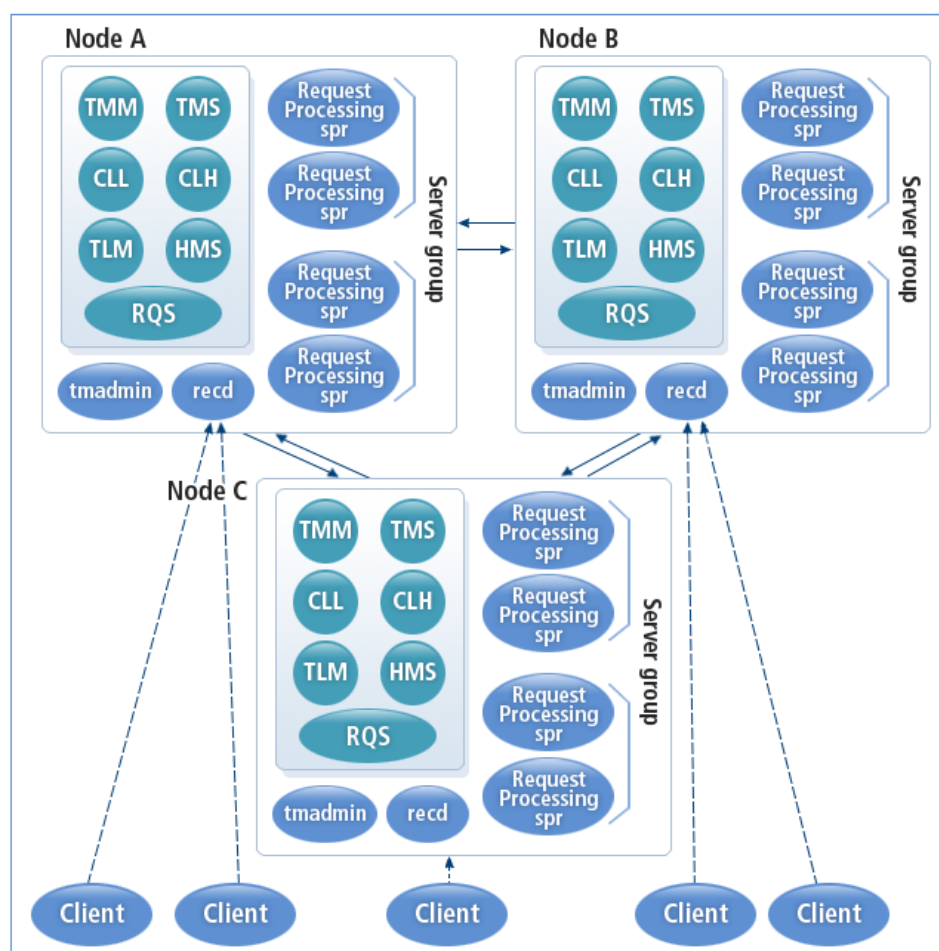
【図 1.3】クライアントとTmaxシステム間の関係



クライアントと最初に接続(1)されたNode Aがサービスを提供するノードのNode Bと通信して、当該サービス処理を要求し(2)、応答を受けて(3)クライアントに返します(4)。

以下は、Tmaxシステムの詳細構成です。

【図 1.4】Tmaxシステム構成図



## 1.4. システム管理

Tmaxシステム管理は大きく静的管理と動的管理に区分できます。

- 静的管理

システムの起動前に実行される管理、または動作中のシステムに影響を与えない管理のことで、**システム環境設定(configuration)**作業を意味します。

- 動的管理

システムが動作している状況でシステムの状態や性能に影響を与え得る管理のことをいい、**Tmaxシステムの起動と終了(tmboot/tmdown)、システム動作管理(Tmax administration)およびシステム環境変数の変更**作業を意味します。

### 1.4.1. 静的管理

システムの静的管理は、Tmaxシステムの環境設定によって行われます。

- Tmax環境ファイルの設定

Tmaxシステムの環境設定のために環境ファイルを作成する必要があります。この環境ファイルには、1つの独立したシステムを構成するドメイン、およびドメイン間の接続やレガシー・システムと通信するゲートウェイ、システムを構成するノード、各ノードのサービス、サービスを提供するサーバー・プロセス、関連するサーバー群であるサーバー・グループなどの全般的なシステム環境情報が含まれています。

環境ファイルの作成は、Tmax管理者が担当します。環境ファイルの作成に関する詳細は、[「第3章 環境ファイルの設定」](#)を参照してください。

- Tmax環境ファイルのコンパイル

Tmax環境ファイルを作成した後は、コンパイルを行い環境ファイルが正しく作成されたことを検証します。環境ファイルが正しく作成されたら、コンパイル・プロセスによりバイナリ形式に変換されたTmax環境ファイルが生成されます。生成されたバイナリ形式のTmax環境ファイルは、Tmaxシステムの起動と終了時に参照されます。

- サービス・テーブルの作成

Tmaxシステムにサービスの種類と位置を知らせるために、作成した環境ファイルを使ってサービス・テーブルを作成する必要があります。サービス・テーブルは環境ファイルの内容に基づいて各サーバーのプロセスごとにテーブルの形で作成されます。また、開発者が作成したサーバー用のソース・プログラムと一緒にコンパイルされ、サーバーの実行プログラムを作成するために使われます。サービス・テーブルをリンクしていない実行プログラムはTmaxシステムが認識できないため、実行できません。

## 1.4.2. 動的管理

以下では、動的管理と関連する起動と終了、システムの動作管理について説明します。

- 起動と終了

環境設定作業が完了すると管理者はTmaxを起動できます。システムが起動するとき、Tmaxの機能プロセスであるTMM、CLL、CLH(データベースが関連する場合はTMSプロセスが含まれ、RQが関連する場合はRQプロセスが含まれます)だけでなく、実際の業務処理サーバー・プロセスも一緒に起動します。業務処理サーバー・プロセスは、別途に動作するのではなく、機能プロセスとともに起動および終了されます。そして、その管理はTmaxシステムが担当します。

- システムの動作管理

Tmaxシステムが動作中であるとき、管理者は各構成要素の状態を確認し、必要な場合は適切な措置を取ります。Tmaxシステムでは**tmadmin**というコンソール形式の会話型管理ツールを使用して動作状態の変更などの動的な作業ができます。

- 各種情報の出力

tmadminは現在の環境設定情報を確認するコマンドを提供します。Tmax管理者はtmadminでコマンドとオプションを使ってクライアント(名前、位置)、サーバー(位置、グループ、プロセス状態、実行回数)、サービス(位置、状態、処理時間、実行回数、蓄積されたデータ量)、ネットワーク(接続ノード、パーティションノード)、その他に実行中のトランザクション、アプリケーション・パラメータなどの情報を確認できます。

- 動的な環境設定の変更

管理者は環境設定によってシステムの設定を修正できます。応答時間の超過(timeout)、処理の優先順位(priority)、負荷調整(load)の値、動作不能状態のサーバー・プロセスの再起動(restart)、サーバー・プロセスの終了、終了されたプロセスの再起動などを設定することができます。また運用中のサーバー・プロセスに新規サービスを動的に追加する機能も提供します。

---

### 注

動的に変更されたTmaxの環境変数は、動作中には変更された内容が適用されますが、システムを終了(tmddown)してから再起動(tmboot)すると、Tmax環境ファイルに設定されている内容が適用されます。

---

## 1.5. ディレクトリーの構成

一般的にTmaxシステムがインストールされるファイルシステムのディレクトリー構成は以下のとおりです。ディレクトリーはシステム環境変数やTmaxシステムの環境設定ファイルの項目で参照されます。

```
$Tmax HOME
+---- appbin
+---- bin
+---- config
+---- lib
+---- lib64
+---- license
|---- log
      +---- slog
      +---- tlog
      +---- ulog
+---- mod
+---- path
+---- run
|---- sample
      +---- client
      +---- tdl
      +---- fdl
      +---- sdl
      +---- server
+---- svct
+---- usrinc
+---- topinc
+---- cobinc
+---- tuxinc
+---- tcpgw
+---- tcpgwthr
+---- x25gw
+---- UninstallerData
+---- bk_appbin
```

\$Tmax HOME

Tmaxシステムのホーム・ディレクトリーです。(システム変数: TMAXDIR、環境ファイル項目: TMAXDIR)

appbin

Tmaxを利用して開発されたアプリケーション・サーバー・プログラムのディレクトリーです。(環境ファイル項目: APPDIR)

bin

Tmaxのコマンドとユーティリティーが存在するディレクトリーです。

config

Tmaxシステム環境ファイルが存在するディレクトリーです。

lib

Tmaxライブラリー(64ビットの場合、「lib64」に表示)のディレクトリーです。

license

ライセンス・ファイルが存在するディレクトリーです。

log

ログファイルが存在するディレクトリーです。

サブディレクトリー	説明
slog	システム・ログファイルを保存するディレクトリーです (環境ファイル項目: SLOGDIR)
ulog	ユーザー・ログファイルを保存するディレクトリーです (環境ファイル項目: ULOGDIR)
tlog	トランザクション情報を保存するディレクトリーです (環境ファイル項目: TLOGDIR)

mod

TDLを使用する場合、アップデートするライブラリーのディレクトリーです。

path

プロセス間の通信に必要なパイプの役割をするディレクトリーです。(環境ファイル項目: PATHDIR)

run

TDLを使用する場合、tdlupdateの後バージョンが付けられたライブラリーのディレクトリーです。

sample

サブディレクトリー	説明
client	クライアントのサンプル・プログラム・ファイルが存在するディレクトリーです
tdl	TDLサンプル・プログラム・ファイルが存在するディレクトリーです
fdl	サンプル・フィールド・キー定義ファイル(demo.f)が存在するディレクトリーです(システム変数: FDLFILE)
sdl	サンプル構造体定義ファイル(demo.s)が存在するディレクトリーです(システム変数: SDLFILE)
server	サーバーのサンプル・プログラム・ファイルが存在するディレクトリーです

svct

アプリケーション・サーバー・プログラムをコンパイル時に必要なサービス・テーブル・ファイルが存在するディレクトリーです。

usrinc

Tmax ヘッダー・ファイルが存在するディレクトリーです。

demo.fとdemo.sは、FDL(Field Definition Language)とSDL(Structure Definition Language)形式のフィールドを定義するファイルです。ユーザーは実際のプロジェクトで使用するFDLおよびSDL形式のフィールドを必要によって直接作成して使用できます。

#### topinc

TOP ENDからTmaxへの移行時に、当該するTuxedo関数をTmax関数に変換するヘッダー・ファイルが存在するディレクトリーです。

#### cobinc

COBOLヘッダー・ファイルが存在するディレクトリーです。

#### tuxinc

TuxedoからTmaxへの移行時に必要なヘッダー・ファイルが存在するディレクトリーです。

#### tcpgw

TCPGWヘッダー・ファイルが存在するディレクトリーです。

#### tcpgwthr

TCPGW THRヘッダー・ファイルが存在するディレクトリーです。

#### x25gw

x25gwヘッダー・ファイルが存在するディレクトリーです。

#### UninstallerData

Tmaxのアンインストールに必要なファイルが存在するディレクトリーです。

#### bk\_appbin (optional)

現在使用しているサーバー・プロセスを新しいプロセスに変更しようとする場合、新規プロセスがインストールされるディレクトリーです。ユーザーが作成します。(システム変数: TMAX\_BKAPPDIR)

## 第2章 環境変数の設定

本章では、サーバー、クライアントに設定する環境変数とその設定方法、多数のサーバー情報を登録する方法について説明します。

### 2.1. 概要

Tmaxシステムで使用する環境変数は大きく2つに分けられます。1つはTmaxシステムで使用する環境変数であり、もう1つはサーバー・プロセスで使われる環境変数です。Tmaxシステムで使用する環境変数はTmaxシステムにアクセスするための変数がほとんどであり、本章で設定する環境変数は次章で説明する環境ファイルの項目と同じ値を入力する必要があります。また、環境変数は当該システムで実行されるすべてのサーバー・プロセスに同様に適用されます。

Tmaxシステムがインストールされたサーバーは該当するシェルのプロファイルに環境変数を必ず登録する必要がありますが、クライアントの場合は2つの方法で設定して使用することができます。

- サーバーと同じくプロファイルに設定して使用する方法
- 特定のファイルに一定のフォーマットで設定して使用する方法

Tmaxシステムと一緒に起動されるサーバー・プロセスで使われる環境変数は特定のファイルに指定されたフォーマットで設定します。環境ファイルのENVFILE項目に当該ファイルを登録して使用できます。

### 2.2. 環境変数

Tmaxシステムがインストールされたサーバーとクライアントはすべて同じ環境変数を使用しますが、Tmaxシステムがインストールされたノードのすべての環境変数をクライアントに設定する必要はありません。

#### 2.2.1. サーバーの環境変数

以下はサーバーに設定する環境変数の一覧です。

環境変数	内容
TMAXDIR	Tmaxシステムのホーム・ディレクトリーです (環境ファイルの項目: TMAXDIR)
TMAX_BKAPPDIR	既存のサーバー・プロセスを変更する場合、新しいプロセスが位置するディレクトリーです

環境変数	内容
TMAX_HOST_ADDR	Tmaxシステムが実行されるサーバーのIPアドレスです。クライアントがランダムにサーバーを選択して接続するため負荷を分散できます。詳細は、 <a href="#">「2.4. 多数のサーバー情報の登録」</a> を参照してください
TMAX_HOST_PORT	Tmaxシステムに接続するポート番号です (環境ファイルの項目: TPORTNO)
TMAX_BACKUP_ADDR	Tmaxバックアップ・サーバーのIPアドレスです
TMAX_BACKUP_PORT	Tmaxバックアップ・サーバーのポート番号です
TMAX_RAC_PORT	分散ノードで構成されたシステムを統合管理するためのポート番号であり、racdが使用します (環境ファイルの項目: RACPORT)
TMAX_ERR_MSG	サーバーのコンソール画面にエラーメッセージを表示する場合は「Y」に、そうでない場合は「N」に設定します
SDLFILE	構造体情報が保存されるファイルを指定します
FDLFILE	フィールドキー情報が保存されたファイルを指定します
TMAX_PATHDIR	tmdown時に参照する環境ファイルのパスです (環境ファイルの項目: PATHDIR)
TMAXHOME	インストール・ディレクトリーと作業ディレクトリーを分離して使用する場合、以下のように設定します(環境ファイルの項目: TMAXHOME) <ul style="list-style-type: none"> <li>– TMAXHOMEのインストール・ディレクトリー : bin, lib, usrinc, tuxinc, topinc, cobinc, license</li> <li>– TMAXDIRの作業ディレクトリー : config, path, log, svct</li> </ul>
TMAX_SEMANTICS	FMLバッファー中に、データを読み込む関数(fbget、fbget_tu、fbget_tut、fbgetf、fbgetlast_tu、fbnext_tu)でバッファーの長さに応じた動作を指定するための環境変数です。関数の最後のフィールドにデータ長を明示し、ユーザーが指定したバッファーを検証します。ユーザーが指定したバッファーが実際のデータより小さい場合に発生する問題を解決できます
TMAX_TRACE	Tmaxアプリケーションの実行に対するランタイム・トレース(Runtime tracing)が行われるようにする環境変数です
TMAX_DEBUG	この環境変数を設定するとき、ファイル名の後ろに該当するクライアントのPID(Process ID)が自動で設定されます(ファイル名.pid)
TMAX_DEBUG	クライアントのデバッグおよびエラー・ログを画面ではなく、ユーザーが指定したファイルに記録できます。  ログを記録するファイルを設定します。filename.pidのように設定すると、実際に記録されるファイル名は、ファイル名の後ろに当該クライアントのPID(Process ID)が付けられた形式で自動で設定されます。  (例: log.22672)



環境変数	内容
TMAX_STRING_NULL	<p>tpalloc/tprealloc APIを使ってSTRING型のバッファを割り当てます。割り当てられたバッファに対し、ユーザーがNULLのための最後の1バイトまでデータを入力してサービスを要求すると、TPEINVAL(クライアント)、TPESVCERR(サーバー)エラーが発生します。</p> <p>この環境変数を「Y」に設定すると、上記の状況が発生した場合、1バイトのバッファが追加で割り当てられ、そのスペースに自動でNULL文字が設定されます</p>
TMAX_HOST_IPV6	<p>TMAX_HOST_ADDR環境変数に設定したアドレスがIPv6プロトコルを使用する場合は、「Y」または「IPV6」に設定します。</p> <p>InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</p>
TMAX_BACKUP_IPV6	<p>TMAX_BACKUP_ADDR環境変数に設定したアドレスがIPv6プロトコルを使用する場合は「Y」または「IPV6」に設定します。</p> <p>InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</p>
TMAX_WEBADM_IPV6	<p>TmaxのWeb管理エージェント・プログラムであるtwagentでTMAX_WEBADM_PORTを介してリッスンする際、IPv6プロトコルを使用するかどうかを設定します。IPv6プロトコルを使用する場合は「Y」または「IPV6」に設定します。</p> <p>InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</p>
TMAX_RAC_IPV6	<p>分散ノードで構成されたシステムの統合管理のために使われるracdデーモン・プロセスで接続やリッスンするとき、IPv6プロトコルを使用するかどうかを設定します。</p> <ul style="list-style-type: none"> <li>– 「Y」または「IPV6」に設定すると、IPv6プロトコルを使用します。InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</li> <li>– 「N」に設定するか、あるいは環境変数を設定しない場合は、IPv4プロトコルを使用します。</li> </ul> <p>racdを起動する際、[-k]オプションを使用する場合にのみ環境変数が適用され、バイナリのTmax環境ファイルにより起動される場合は、SYSTEM_IPV6の設定値から影響を受けます</p>
TMAX_IPV6_LINK_IF	<p>IPv6プロトコルを使用する環境において、Tmaxのプロセスがリッスンまたは接続を実行する際、ネットワーク・インターフェース名に対するインターフェース・イ</p>

環境変数	内容
	ンデックスを正常に認識できない場合、ネットワーク・インターフェース名を指定します
TMAX_APPLY_IPCPERM	<p>サーバーが最初に起動するとき、CLOPTセクションの-e、-oオプションを指定して記録するログファイルの権限が、環境設定ファイルのDOMAINまたはNODEセクションに設定されるIPCPERM値に従って指定されるように設定します</p> <ul style="list-style-type: none"> <li>– 「Y」または「y」に設定すると、サーバーが-e、-oオプションで記録するログファイルがIPCPERMに従って作成されます</li> <li>– 「N」または「n」に設定するか、設定しない場合は、最初に作成されるログファイルの権限がIPCPERMに従うよう保証されません</li> </ul>
TMAX_LOGLVL	<p>tmmに接続しないモジュールやtmmに接続する前のモジュール、およびtmmについて出力するデバッグ・ログのレベルを決定します。設定値は環境設定ファイルのDOMAINセクションのTMMLOGLVLと同じです。</p> <p>TMAX_DEBUGと一緒に使用してログの出力位置を決定できます。TMAX_DEBUG以上のログは、stderrがデフォルトの位置になります。</p> <p>一般ユーティリティーは、TMAX_ERR_MSG環境変数がNまたはnの場合は、メッセージを出力しません。</p> <p>tmmに接続するモジュールは、tmmに接続するまでのみTMAX_ERR_MSG環境変数によってログの出力可否が決定されます。tmmへの接続以降はすべてのログを出力します。DEBUG4レベルは、一部のゲートウェイでダンプ・メッセージを出力できます</p>
TMAX_TENC_CIPHER	<p>Tmax環境設定ファイルで使用可能なSEEDベースの暗号化・復号化ユーティリティーに関連する設定であり、暗号化方式を設定します。</p> <p>SEED以外の値を設定するか、または何も指定しない場合は、既存の暗号化方式で処理されます</p>
TMAX_TENC_KEYFILE	秘密鍵ファイルの場所を指定します。暗号化方式がSEEDの場合は、必ず指定します

#### 注

TMAX\_STRING\_NULLは、tpalloc/tpreallocで割り当てられたバッファーに限ってサポートされ、サーバーとクライアントが同様に動作します。環境変数を必ず「Y」に設定してください。

以下は、TMAX\_SEMANTICS環境変数の設定値についての例です。ユーザーが指定したバッファー・サイズが実際のデータサイズより小さい場合、ユーザーが指定したバッファー・サイズのみだけ切り捨てて(truncate)バッファーに格納します。

```
TMAX_SEMANTICS = "FDL_LEN=TRU"
```

以下のように設定すると、ユーザーが指定したバッファ・サイズが実際のデータ・サイズより小さい場合、FBENOSPACEエラーが発生します。

```
TMAX_SEMANTICS = "FDL_LEN=ERR"
```

以下のように設定すると、ユーザーが入力したlengthが無視され、ユーザーが十分なサイズのバッファを割り当てたと仮定して動作します。この環境変数はfbget関数の最後のフィールドにデータ長を明示した場合のみ有効です。最後のフィールドが0またはNULLであれば、ユーザーが十分なサイズのバッファを割り当てたと仮定して動作します。

```
TMAX_SEMANTICS = "FDL_LEN=ERR"
```

## 2.2.2. クライアントの環境変数

以下は、クライアントに登録する環境変数の一覧です。

環境変数	内容
TMAX_HOST_ADDR	Tmaxサービス・サーバーのIPアドレスです
TMAX_HOST_PORT	Tmaxサービス・サーバーのポート番号です
TMAX_CONNECT_TIMEOUT	Tmaxシステムに接続する場合、タイムアウト時間をマイクロ秒単位で指定できます(x.xxx秒)
SDLFILE	構造体情報が保存されたファイルを指定する環境変数です
FDLFILE	フィールドキー情報が保存されたファイルを指定する環境変数です
ULOG(またはULOGPFX)	クライアントにログを残したいパス名を設定します。設定しないとTmax環境ファイルに設定されたULOGDIRパスに記録が残ります
TMAX_TRACE	クライアント・アプリケーションの実行に対してランタイム・トレース(Runtime tracing)を行います
TMAX_SEMANTICS	FMLバッファ中に、データを読み込む関数(fbget、fbget_tu、fbget_tut、fbgetf、fbgetlast_tu、fbnext_tu)でバッファの長さに応じた動作を指定するための環境変数であり、サーバーの環境変数を参照します。  暗号化機能を使用する場合、Tmaxはクライアントとサーバー間の送受信データをより安定的にやり取りできるように暗号化機能を提供します
TMAX_CRYPT_ALGORITHM	暗号化機能を使用する場合に暗号化方式を指定します  – 指定しない場合、デフォルト値として「3DES」方式が設定されます

環境変数	内容
	<ul style="list-style-type: none"> <li>「AES」方式に変更するには、TMAX_CRYPT_ALGORITHM="AES"を指定します</li> </ul>
TMAX_DEBUG	<p>クライアントのデバッグおよびエラー・ログを画面ではなくユーザーが指定したファイルに記録することができます。ログを記録するファイルを設定します。</p> <p>filename.pidのように設定すると、実際に記録されるファイル名は、ファイル名の後ろに当該クライアントのPID(Process ID)が付けられた形式で自動で設定されます(例: log.22672)</p>
TMAX_DEBUG	<p>「directory/filename」の形式でファイルを設定し、設定時にファイル名の後ろに「ファイル名.pid」の形式で該当するクライアントのPID(Process ID)が自動で設定されます</p>
TMAX_HOST_IPV6	<p>TMAX_HOST_ADDR環境変数に設定したアドレスがIPv6プロトコルを使用する場合は「Y」または「IPV6」に設定します。</p> <p>InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</p>
TMAX_BACKUP_IPV6	<p>TMAX_BACKUP_ADDR環境変数に設定したアドレスがIPv6プロトコルを使用する場合は「Y」または「IPV6」に設定します。</p> <p>InfiniBand SDP(Socket Direct Protocol)を使用する場合は「SDP」に設定します</p>
TMAX_IPV6_LINK_IF	<p>IPv6プロトコルを使用する環境において、Tmaxのプロセスがリッスンまたは接続を実行する際、ネットワーク・インターフェース名に対するインターフェース・インデックスを正常に認識できない場合、ネットワーク・インターフェース名を指定します</p>
TMAX_ACTIVATE_AUTO_TPSTART	<p>「Y」または「N」を設定します。設定しない場合のデフォルト値は「Y」になります。</p> <ul style="list-style-type: none"> <li>Y: クライアント・プログラムでtpstartを明示的に呼び出さずに、tpcallやtpacallなどのAPIを呼び出すと、自動的にtpstart(NULL)を呼び出してTmaxシステムと接続します</li> <li>N: tpstart()が明示的に呼び出されなかった場合、TPEPROTOエラーが発生します</li> </ul> <p>この環境変数の影響を受けるAPIは以下のとおりです</p> <ul style="list-style-type: none"> <li>tpcall(), tpacall(), tpcallsvg(), tpacallsvg(), tpconnect(), tpchkunsol(), tpsetunsol(), tpgetunsol(), tpsubscribe(), tpsubscribe2(), tpunsubscribe(), tpunsubscribe(), tpbroadcast(), tpgetcliaddr(),</li> </ul>

環境変数	内容
	<p>tpgetpeername()、tpgetsockname()、tpsleep()、tpgetactivesvr()、tpqsvccstat()、tpqstat()、tpenq()、tpdeq()、tpreissue()、tpmcall()、tpsvgcall()、tpenq_ctl()、tpdeq_ctl()、tx_begin()</p>
PB_TPCALL_RETRY	<p>PowerBuilderでtpcall要求がネットワーク・エラーによって失敗した場合、再試行する最大回数を設定します。</p> <p>この環境変数の影響を受けるAPIは以下のとおりです</p> <ul style="list-style-type: none"> <li>– pb_tpcall()、pb_tpfcall()、pb_tpcallw()、pb_btpcall()、pb_etpcall()、tuxcall()、tuxfcall32()、tp_call()、tp_call32()、tp_fcall()、tp_fcall32()</li> </ul>
PB_TPCALL_RETRY_TIME OUT	<p>PowerBuilderでtpcallを要求した場合、各要求に対するブロック時間を秒単位で指定します。この環境変数の影響を受けるAPIについては、上記のPB_TPCALL_RETRY環境変数を参照してください</p>
PB_TUXCOMPAT	<p>PowerBuilderで一部APIの動作に対してTuxedoライブラリーとの互換性を提供します。詳細内容については、『Tmax プログラミングガイド(4GL)』を参照してください。</p> <p>この環境変数の影響を受けるAPIは以下のとおりです</p> <ul style="list-style-type: none"> <li>– tuxcall()、tuxfcall32()、tp_call()、tp_call32()、pb_etpcall()、tp_fcall()、tp_fcall32()</li> </ul>
FDLVERSION	<p>FDLバージョンを指定することができます。(デフォルト値：1)</p> <ul style="list-style-type: none"> <li>– 2 : 検索パフォーマンスが向上されたバージョンで動作しますが、既存のTmaxとの互換性はありません。この環境変数の影響を受けるのは、サーバー・アプリケーション、ROUTINGセクションの設定に伴うCLHのスケジューリングです。クライアントでは環境設定のオプションが適用されず、次の環境変数を設定する必要があります。</li> </ul> <p>export FDLVERSION=2またはtmax.env環境変数ファイルに「FDLVERSION =2」と設定し、tmaxreadenv()を呼び出します。fb* APIを使用する際、FDLVERSION設定が自身の環境設定と一致しない場合は、FBESETVERエラーを返します</p>
TMAX_LOGGLVL	<p>tmmに接続しないモジュールやtmmに接続する前のモジュール、およびtmmについて出力するデバッグ・ログのレベルを決定します。設定値は環境設定ファイルのDOMAINセクションのTMMLOGGLVLと同じです</p>

TMAX\_SEMANTICSを設定すると**暗号化機能**を使用できます。暗号化機能を使用するには、クライアントとサーバーの両方に暗号化機能が設定されている必要があります。クライアントの暗号化設定は、以下のとお

り「.profile」に設定します。サーバーの暗号化の設定については、「[3.2.1 DOMAINセクション](#)」のCRYPT項目を参照してください。

<.profile>

```
export TMAX_SEMANTICS = "CRYPT_SW=Y"
```

tmaxreadenvでTMAX\_SEMANTICS環境変数を使用する場合、二重引用符(" ")内に等号(=)の代わりにコロン(:)を使用します。

<tmax.env>

```
TMAX_SEMANTICS = "FDL_LEN:ERR"  
TMAX_SEMANTICS = "CRYPT_SW:Y"
```

## 2.3. 環境変数の設定方法

以下では、サーバーとクライアントの環境変数の設定方法について説明します。

### 2.3.1. サーバー環境変数の設定

サーバー環境変数の設定は、使用するシェルに応じて設定する対象が異なります。kornシェルおよびbashシェルの場合は<.profile>と<.bash\_profile>、cシェルの場合は<.cshrc>にそれぞれ設定します。

---

#### 注

環境変数を設定するとき、環境変数名と等号(=)の間に空白を入れないようにします。

---

- kornシェルおよびbashシェル

<.profile / .bash\_profile>

```
export TMAXDIR = /home/tmax  
export TMAX_BKAPPDIR = /home/tmax/bkappbin  
export TMAX_HOST_ADDR = 192.168.0.1  
export TMAX_HOST_PORT = 8888  
export TMAX_RAC_PORT = 3333  
export SDLFILE = /home/tmax/sample/sdl/tmax.sdl  
export FDLFILE = /home/tmax/sample/sdl/tmax.fdl  
export TMAX_PATHDIR = /home/tmax/path_new  
export TMAX_DEBUG = /home/tmax/sample/client/clidebug
```

- cシェル

<.cshrc>

```

setenv TMAXDIR = /home/tmax
setenv TMAX_BKAPPDIR = /home/tmax/bkappbin
setenv TMAX_HOST_ADDR = 192.168.0.1
setenv TMAX_HOST_PORT = 8888
setenv TMAX_RAC_PORT = 3333
setenv SDLFILE = /home/tmax/sample/sdl/tmax.sdl
setenv FDLFILE = /home/tmax/sample/sdl/tmax.fdl
setenv TMAX_PATHDIR = /home/tmax/path_new
setenv TMAX_DEBUG = /home/tmax/sample/client/clidebug

```

Tmaxインストーラーを利用してTmaxをインストールします。基本的な環境変数はシェルのプロファイルの最後に自動で設定されます。Tmaxシステムのサーバーで使用する環境変数はテキスト・ファイルで作成してTmax環境ファイルのNODEセクションおよびSVRGROUPセクションのENVFILE項目にそのファイル名を登録すると、サーバー・プロセスで参照できます。

<サーバー環境のテキスト・ファイル>

```

LOGDIR = /tmp
USER_VARIABLE = test1
USER_VARIABLE = test2
USER_VARIABLE = test3
USER_VARIABLE = test4
USER_VARIABLE = test5

```

## 2.3.2. クライアント環境変数の設定

クライアントの環境変数を設定する方法は、OSに応じて異なります。

- UNIXの場合

サーバーと同様にプロファイルに設定します。

- DOSまたはWindows98、Windows NT/2000の場合

<autoexec.bat>ファイルに登録するか、**[スタート] > [コントロールパネル] > [システム] > [詳細設定] > [環境変数]**で設定します。

<autoexe.bat>

```

set TMAX_HOST_ADDR = 192.168.0.1
set TMAX_HOST_PORT = 8888
set TMAX_CONNECT_TIMEOUT = 3
set SDLFILE = /home/tmax/sample/sdl/tmax.sdl
set FDLFILE = /home/tmax/sample/sdl/tmax.fdl
set TMAX_DEBUG = /home/tmax/sample/client/clidebug

```

クライアントの環境変数は以下のようにテキストファイルで登録すると、必要に応じて当該ファイルを参照して使用できます。

<tmax.env>

```
[TEST]
TMAX_HOST_ADDR = 192.168.0.1
TMAX_HOST_PORT = 8888
TMAX_CONNECT_TIMEOUT = 3
SDLFILE = /home/tmax/sample/sdl/tmax.sdl
FDLFILE = /home/tmax/sample/sdl/tmax.fdl
ULOGPFX = /home/tmax/testlog
TMAX_DEBUG = /home/tmax/sample/client/clidebug

[REAL]
TMAX_HOST_ADDR = 192.168.0.2
TMAX_HOST_PORT = 1234
TMAX_CONNECT_TIMEOUT = 3
SDLFILE = /home/tmax/sample/sdl/tmax.sdl
FDLFILE = /home/tmax/sample/sdl/tmax.fdl
ULOGPFX = /home/tmax/reallog
TMAX_DEBUG = /home/tmax/sample/client/clidebug
```

---

#### 参考

クライアントの環境変数の設定についての詳細は、『Tmax リファレンスガイド』の「3.1.20. tmaxreadenv」を参照してください。

---

## 2.4. 多数のサーバー情報の登録

以前のバージョンでは1つのクライアントは1つのサーバーにだけアクセスできましたが、Tmax 3.12.2バージョンからは、クライアントがランダムにサーバーを選択してアクセスすることで負荷を分散できる機能が追加されました。多数のサーバーを使用する場合は、**TMAX\_HOST\_ADDR**環境変数を設定します。

以下は、多数のサーバー情報を登録する方法です。

- IPv4

```
TMAX_HOST_ADDR = (host_address:portno|host_address2:portno2),
                  host_address3:portno3,host_address4:portno4
```

- IPv6

```
TMAX_HOST_ADDR = ([host_address]:portno|[host_address2]:portno2),
                  [host_address3]:portno3,[host_address4]:portno4
```



コンマ(,)による区切りは、前から順にバックアップ関係を示します。つまり、「host\_address:portno|host\_address2:portno2」への接続に失敗すると「host\_address3:portno3」に接続を試み、そのサーバーへの接続も失敗したら「host\_address4:portno4」に接続を試みます。

括弧内のパイプ記号(|)で連結されたものは負荷分散の関係にあります。負荷分散の関係にあるサーバーはクライアントからランダムに1つのサーバーを選択して接続を試み、接続に失敗すると負荷分散の関係にある別のサーバーに順次に接続を試みた後、バックアップに指定されたサーバーに接続を試みます。

以下は、TMAX\_HOST\_ADDRの設定についての説明です。

- TMAX\_HOST\_ADDRに設定する環境変数の最大長は255バイトです。
- TMAX\_HOST\_ADDRに指定されたアドレスへの接続がすべて失敗すると、TMAX\_BACKUP\_ADDR環境変数に指定されたアドレスに接続を試みます。
- TMAX\_HOST\_ADDR環境変数の値には空白を入れないようにします。
- シェルでexportやsetenvで環境変数を指定する場合には括弧やパイプ記号(|)をシェルが別途処理するため二重引用符(" ")を使用します。

## 使用例

以下は、サーバー情報の登録方法を比較する例です。

- 既存の方式で動作させる場合

```
TMAX_HOST_ADDR = 192.1.1.1
TMAX_HOST_PORT = 9000
```

- クライアントをランダムに接続させる場合

```
TMAX_HOST_ADDR = (192.1.1.1:9000|192.1.1.2:9001),192.1.1.3:9003,192.1.1.4:9004
```

- IPv6プロトコル環境でクライアントをランダムに接続させる場合

IPv6プロトコル環境では、ホスト・アドレスとポート番号(portno)を区別するために、大括弧([ ])を使ってアドレスを囲みます。

```
TMAX_HOST_ADDR =
([2011::10]:9000|[2011::20]:9001),[2011::100:30]:9003,[2011::100:40]:9004
```



# 第3章 環境ファイルの設定

本章では各環境ファイルを設定する方法について説明します。

## 3.1. 概要

Tmax環境ファイルには1つのTmaxドメインの環境設定を行います。ドメイン(Domain)とは、複数のノード(Machine)で構成できる、独立した1つのTmaxシステムをいいます。ドメイン情報にはシステムに接続できるユーザーの最大数やトランザクションの処理制限時間、クライアントのサービス処理時間などの内容が含まれます。

ドメインを構成する各ノード別に環境設定を行うことができます。ノードに接続できるユーザーの最大数、ノードに存在するTmaxプログラムの位置、ジョブ処理サーバー・プロセスの位置などの内容を含みます。

さらに、サーバー・グループの環境設定も行えます。サーバー・グループはサービスを提供する業務処理サーバー・プロセスの集合であり、複数のサーバー・プロセスを1つのサーバー・グループにまとめて管理できます。サーバー・グループにはデータベースのアクセス情報、負荷調整および障害対策などの環境設定を行います。

実際にサービスを提供する業務処理サーバー・プロセス(コマンドラインのオプションやサーバー数)とサービス(優先順位と処理制限時間)、ルーティング(データ型と範囲に合わせてサーバー・グループごとに分散処理)の環境設定も必要です。ドメイン間のサービス进行处理できるゲートウェイ環境も設定できます。

環境ファイルは以下の9つのセクションで構成されます。

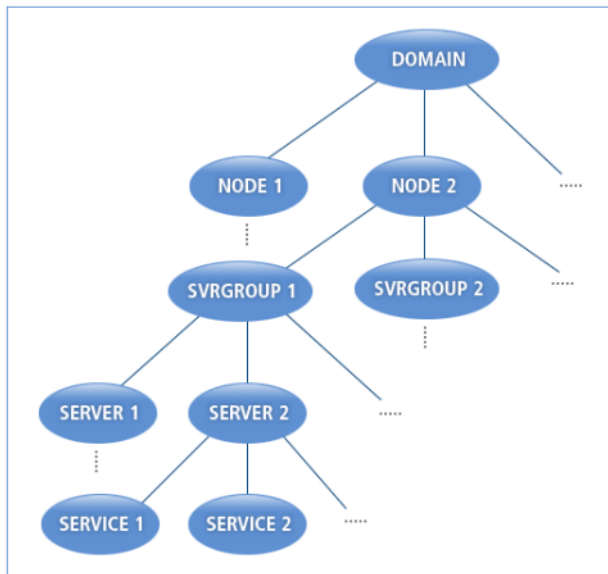
- DOMAIN
- NODE
- SVRGROUP
- SERVER
- SERVICE
- GATEWAY
- ROUTING
- RQ

- HMS

各セクション同士はツリー(tree)形態の相関関係にあります。Tmaxシステムの全体環境に当たるドメインが存在しており、ドメインには1つ以上のノード、各ノードには1つのサーバー・グループ、サーバー・グループには1つ以上の業務処理サーバー・プロセス、各業務処理サーバー・プロセスには1つ以上のサービスが含まれる関係にあります。

以下は、環境ファイルを構成する各セクションの相関関係をツリー形態で示しています。

**[図 3.1] 環境ファイルの構成要素間の相関関係**



### 3.1.1. 環境ファイルの形式

Tmax環境ファイルはDOMAIN、NODE、SVRGROUP、SERVER、SERVICE、GATEWAY、ROUTINGセクションで構成されます。このうち、GATEWAYセクションとROUTINGセクションを除いた5つのセクションは必ず定義される必要があります。

以下は、Tmax環境ファイルの設定例です。

```
*DOMAIN
res1    SHMKEY = 77990, MINCLH = 1, MAXCLH = 3,
        MAXUSER = 100, TPORTNO = 8888, BLOCKTIME = 100

*NODE
tmax1    TMAXDIR = "/home/tmax",
        APPDIR = "/home/tmax/appbin",
        PATHDIR = "/home/tmax/path",
        SLOGDIR = "/home/tmax/log/slog",
```

```

TLOGDIR = "/home/tmax/log/tlog",
ULOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1    NODENAME = tmax1, DBNAME = ORACLE,
        OPENINFO = "Oracle_XA+Acc = P/tmax/tmax+SesTm = 60"
svg2    NODENAME = tmax1

*SERVER
svr1    SVGNAME = svg1, CLOPT = "-- -f aa -x bb"
svr2    SVGNAME = svg2

*SERVICE
SVC1    SVRNAME = svr1
SVC2    SVRNAME = svr2

```

## 詳細形式

以下は、各セクションと項目の形式です。

### ● セクション

セクションは以下の形式で定義します。

```

*セクション
名前          項目1 = .... , 項目2 = .... , 項目3 = ....

```

### ● 項目

各セクションの項目は以下の形式で定義します。項目は「項目名 = 値」の形式で定義します。

```

項目 = タイプ (デフォルト値)
        範囲またはサイズ
        内容

```

各項目は、以下のデータ型のうち1つで定義できます。項目名は値として使用できません。

項目値	説明
numeric	数字の範囲を定義します
string	abc形式の文字列を意味します
literal	"abc" 形式の文字列を意味します
Y   N	YesまたはNOの形式です
Y   B   N	YesまたはBothまたはNoの形式です

環境ファイルの各セクションと項目は以下の規則に従って作成します。

- セクションはアスタリスク(\*)で書き始め、大文字を使います。
- セクション名と項目名は必ず最初の列から書きます。
- セクション名と項目名以外は最初の列から書くことができません。
- セクションの各項目はコンマ(,)で区切ります。コンマ(,)がないと該当するセクションの定義が終わったものと見なします。
- セクションを定義する順序は決まっていないため、どのセクションを先に定義しても構いません。1つのセクションを複数回に分けて一度以上定義することができます。たとえば、ノードごとにサーバー・グループ、業務処理サーバー・プロセス、サービスなどを分離して定義できます。ただし、同じ内容を繰り返し定義することはできません。
- 項目同士の間空白は意味がありません。
- 項目を定義しないとデフォルト値で設定されます。

### 3.1.2. 環境ファイルの作成

Tmax環境ファイルの作成はTmaxシステム管理者の主要業務です。Tmaxシステムは環境ファイルに基づいて起動または終了します。そのため、環境ファイルが存在しないか正しく作成されていないとTmaxシステムは動作しません。

Tmax環境ファイルは大きく9つの部分に分けられます。

- **基本環境設定**

Tmaxシステムの構成に基本的に必要な環境設定をします。Tmax環境ファイルに必須に定義される必要のあるDOMAIN、NODE、SVRGROUP、SERVER、SERVICEセクションと、ドメイン間のサービスを処理するGATEWAYセクション、ルーティング環境設定のために追加できるROUTINGセクションなど、Tmax環境ファイルの全般的な内容が扱われます。

データベース、トランザクション、負荷調整、障害対策、セキュリティ関連の環境設定はこの基本環境設定に基づいて行われます。

- **データベースの環境設定**

データベース情報を登録し、データベースの管理に必要なデータベース関連の環境設定を行います。

- **分散トランザクションの環境設定**

Tmaxシステムの分散トランザクション処理(Distributed Transaction Processing)機能を使用する場合に必要なトランザクション環境設定を行います。

- **負荷調整の環境設定**

ノード間の負荷調整のために必要な環境設定を行います。選択的に定義できるROUTINGセクションの詳細が取り扱われます。

- **信頼性キューの環境設定**

障害によってシステムが異常終了したとき、終了前に要求されたサービスをシステムが正常起動した後も引き続き処理できるように環境設定を行います。

- **HMS環境設定**

メッセージの送受信時点を分離してデータを遅延処理することができ、転送の信頼性を保証するための環境設定を行います。

- **障害対策の環境設定**

ノードに障害が発生すると障害ノードのサービスが他のノードで提供されるように環境設定を行います。

- **セキュリティの環境設定**

Tmaxシステムのセキュリティ機能の環境設定を行います。

- **マルチドメイン環境設定**

Tmaxシステムのマルチドメイン環境でドメイン間のルーティング関連の環境設定を行います。

## 3.2. 基本環境設定

環境ファイルに必須に定義される5つのセクション(DOMAIN、NODE、SVRGROUP、SERVER、SERVICE)とマルチドメインを処理するためのGATEWAYセクション、およびROUTINGセクションについて説明します。

### 3.2.1. DOMAINセクション

独立した単一のTmaxシステム全体に対する環境設定を行います。マルチドメインでシステムを定義した場合にはGATEWAYセクションを利用してドメイン間のサービスを処理できます。

DOMAINセクションでは基本的に以下の内容を設定できます。

- 共有メモリーのキー値
- Tmaxプロセス(CLH)の実行数

- 接続クライアントの最大数
- ポート番号
- クライアントがサービスを要求した後の最大の待ち時間

DOMAINセクションで定義された内容は1つのTmaxシステムを構成するすべてのノードに共通して適用されます。ただし、DOMAINセクションに定義されていても、残りの4つの必須セクションで再定義できる項目もあります。基本定義内容と再定義内容が重複する場合は後者が優先します。

DOMAINセクションの基本環境設定の形式は以下のとおりです。

```
*DOMAIN
[DEFAULT : ]
Domain Name      SHMKEY = shared-memory-segment-key,
                  [MAXUSER = 1 ~ MAX_INT,]
                  [MINCLH = 1 ~ 10,]
                  [MAXCLH = 1 ~ 10,]
                  [TPORTNO = port-number,]
                  [RACPORT = port-number,]
                  [BLOCKTIME = timeout-value,]
                  [CPC = channel-number]
                  [MAXFUNC = max-function-number,]
                  [LOGOUTSVC = logout-service-name,]
                  [CLICKINT = interval-time-value,]
                  [IDLETIME = idle-time,]
                  [MAXSACALL = numeric,]
                  [MAXCACALL = numeric,]
                  [TXTIME = transaction-timeout-value,]
                  [NLIVEINQ = alive-check-interval,]
                  [NCLHCHKTIME = interval-time-value,]
                  [MAXCONV_NODE = numeric,]
                  [MAXCONV_SERVER = numeric,]
                  [SECURITY = ("NO_SECURITY") | "DOMAIN_SEC" | "USER_AUTH" |
                               "ACL" | "MANDATORY" | "OTHER_AUTH" ,]
                  [OWNER = name,]
                  [IPCPERM = mask,]
                  [DOMAINID = numeric,]
                  [CMTRET = literal,]
                  [MAXNODE = numeric,]
                  [MAXSVG = numeric,]
                  [MAXSPR = numeric,]
                  [MAXSVR = numeric,]
                  [MAXSVC = numeric,]
                  [MAXCPC = numeric,]
                  [MAXTMS = numeric,]
```



```

[MAXROUT = numeric,]
[MAXROUTSVG = numeric,]
[MAXRQ = numeric,]
[MAXGW = numeric,]
[MAXCOUSIN = numeric,]
[MAXCOUSINSVG = numeric,]
[MAXBACKUP = numeric,]
[MAXBACKUPSVG = numeric,]
[MAXTOTALSVG = numeric,]
[MAXPROD = numeric,]
[GWCHKINT = interval-time-value,]
[GWCONNECT_TIMEOUT = interval-time-value,]
[TMMLOGLVL = tmm-log-level,]
[CLHLOGLVL = clh-log-level,]
[TMSLOGLVL = tms-log-level,]
[LOGLVL = server-log-level,]
[CRYPT=Y|(N),]
[CRYPT_ALGORITHM=("3DES")|"AES",]
[MAXTHREAD = numeric,]
[TDL = Y|(N),]
[MAXSESSION = numeric,]
[TXPENDINGTIME = pending-transaction-timeout,]
[CLIENT_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[SYSTEM_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[EXTSVR_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[CLIENT_IPV6 = Y|(N),]
[SYSTEM_IPV6 = Y|(N),]
[EXTSVR_IPV6 = Y|(N),]
[CLL_BIND_IP = (Y)|N,]
[FDL_VERSION = 1 | 2,]
[MSGSIZEWARN = numeric,]
[MSGSIZEMAX = numeric,]
[CASCPC = numeric]

```

## 必須項目

- *Domain Name* = string
  - 範囲 : 63文字以内
  - DOMAINセクションはTmaxシステムの全般的な環境を定義します。
- SHMKEY = numeric
  - 範囲 : 32768 ~ 262143

- 共有メモリー・セグメントを示す値です。

Tmaxには、機能別に大きく4つのプロセスがあります。

区分	説明
TMM	Tmaxの管理ツールとして他のプロセスを管理します
TMS	データベースおよびリソースを管理します
CLL	クライアントの接続を管理します
CLH	クライアントとサーバー・プロセス間で要求と応答を処理します

SHMKEYでは、上記の4つのプロセスが共有する情報を管理するための共有メモリーのキー値を定義します。Tmaxシステムは内部で4つの共有メモリーが使われ、定義された値から順番に4つのキー値 (SHMKEY、SHMKEY+1、SHMKEY+2、SHMKEY+3)を使用します。

---

#### 注

共有メモリーのキー値を定義する前にそのキー値が他の業務で使われていないか必ず確認します。衝突を防ぐには、使用されていない値で定義する必要があります。

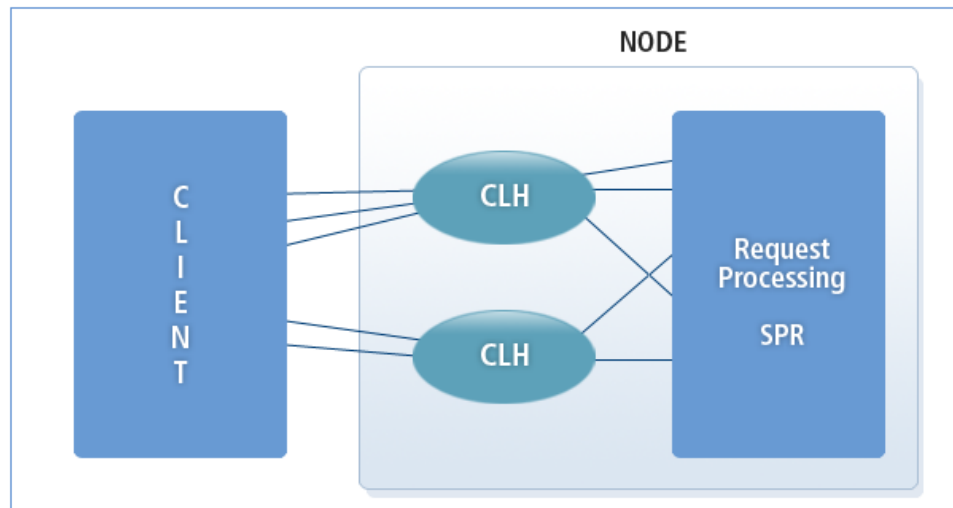
---

## 選択項目

- DEFAULT
  - DOMAINセクションを含むTmax環境ファイルのすべてのセクションで利用できる項目です。詳細は、「[3.2.2 NODEセクション](#)」のDEFAULT項目を参照してください。
- MAXUSER = numeric
  - 範囲：1 ~ MAX\_INT
  - ドメインに同時に接続できるクライアントの最大数を定義します。「MAXUSERに設定された値+1」の分だけクライアントの接続を許します。
- MINCLH = numeric
  - 範囲：1 ~ 10
  - デフォルト値：1
  - Tmaxシステムが起動するとき作動するCLHの最小数を定義します。

CLHはクライアントの要求を処理するプロセスであり、クライアントのサービス要求を受けて業務処理サーバー・プロセスにサービスを要求し、その結果を受けて再度クライアントに転送する機能をします。CLHはすべてのクライアントと接続されるプロセスで、システムによって1つのCLHに接続できるクライアント数には制限があります。そのため、多くのクライアントが接続を試みる大型ドメインの場合、1つ以上のCLHプロセスを動作させる必要があります。このためには適切なMINCLH値の定義が必要です。

[図 3.2] MINCLH = 2の場合



- MAXCLH = numeric
  - 範囲 : 1 ~ 10
  - デフォルト値 : 10
  - CLHの最大数を定義します。Tmaxシステムが初めて起動するとき、CLHはMINCLHに定義した数だけ動作します。MINCLHで定義したCLHプロセス数では処理できないほどにクライアントの要求が増加すると、CLHプロセスを追加動作させて業務が円滑に処理できるようにします。
- 追加されるCLHプロセスの数に、すでに動作しているCLHプロセスの数を合計した値はMAXCLH数を超えることができません。現在は、クライアントの要求が増加するにつれてMAXCLHプロセス数が増える機能は提供していません。
- TPORTNO = numeric
  - デフォルト値 : 8888
  - クライアントとサーバーの接続に使用するTmaxシステムのポート番号を定義します。定義された値(TPORTNO)とその次の値(TPORTNO+1)がポート番号に使われます。

最初のTPORTNO値はクライアントがサーバーに接続するときに使用し、次の値(TPORTNO+1)はノード間通信のために使われます。クライアント/サーバー環境で、クライアントはサーバーのアドレスとTmaxシステムのポート番号を把握する必要があります。

- Tmaxクライアント環境では、アドレス(TMAX\_HOST\_ADDR)とポート番号(TMAX\_HOST\_PORT)の設定が必要です。この場合、設定するポート番号はTPORTNOです。しかし、クライアントとサーバーが同じノードに存在する場合にはTCP/IPソケットを使用するのではなく、ドメイン・ソケットを使用してより効果的にサービスを処理できます。

サーバーは2タイプのクライアントを両方とも受信できるように設定されています。TCP/IPソケットに対してはTPORTNOで待ち受け(listen)し、ドメイン・ソケットに対してはPATHDIRのcllrcadというストリームパイプで待ち受けます。

- クライアントとサーバーが同じノードに存在する場合には環境変数のTMAX\_HOST\_PORT項目にTPORTNOで使用したポート番号の代わりにPATHDIRを指定します。たとえば、「.profile」またはtmax.envに以下のように指定できます。

```
TMAX_HOST_PORT=/home/tmax/path
```

「TPORTNO+1」はノード間通信のためのもう1つのポート番号に使われます。

- Tmaxシステムのノード間の構成方式はpeer-to-peer形態です。TPORTNO+1値を使用して環境ファイルに登録された順にノードを確認します。システム領域では1024未満のポート番号が使われているため、管理者はこの範囲のポート番号は使わないようにします。
- クライアントとTmaxシステムまたはノード間で通信するとき、TPORTNOとTPORTNO+1値に定義されたポート番号が使われるため、この値が他のところで使用されているかを確認する必要があります。また、マルチノードを設定するときCLHが追加ポートを使用するため、CLH数の分だけポート値が有効であるかも確認する必要があります。
- IPv6プロトコルを使用する場合は、**CLIENT\_PROTOCOL**項目を「IPV6」に設定します。

- RACPORT = numeric

- デフォルト値 : 3333
- 1ドメイン内に複数のノードが構成されている場合、複数ノードを中央で管理するため、ノード間通信のためのポート番号を定義します。管理プロセスの1つであるracdで使用するポート番号です。当該ポート番号は環境変数であるTMAX\_RAC\_PORT番号に定義された値と同一に設定します。システム領域では1024未満のポート番号が使われているため管理者はこの範囲を避けて指定する必要があります。
- 環境変数で値が異なる場合はTMAX\_RAC\_PORTを使用します。RACPORTはTPORTNOで定義したポート番号より小さいポート値を使用することを推奨します。また、RACPORTのポート番号も他のところで使用されているかを必ず確認します。

- BLOCKTIME = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 0 (単位: 秒)
- 小数点単位で設定が可能です。
- 値を設定しないとデフォルト値が設定され、無限待ちとなります。
- サービス要求に対するタイムアウト(timeout)時間を決めます。クライアントがAPI関数を使ってサービスを要求した後、最大のBLOCKTIME時間が過ぎるまで応答がないとクライアント内部でタイムアウト処理します。しかし、サーバー・プロセスは引き続きサービス进行处理します。
- サービスも終了させたいときは**SERVICEセクション**の**SVCTIME**項目にサービスタイムを指定します。BLOCKTIMEはSVCTIMEと関連付けられています。BLOCKTIMEがSVCTIMEより長いと、クライアントのtpcallをタイムアウトするときSVCTIMEが適用されます。反対にSVCTIMEがBLOCKTIMEより長いと、BLOCKTIMEが適用されます。BLOCKTIMEはクライアントのtpcallのタイムアウトであり、SVCTIMEはサーバー・プロセスのタイムアウトであるため、SVCTIMEのタイムアウト時間までtpcall()されたサービスが実行されます。一般にBLOCKTIMEはSVCTIMEより大きい値を持ちます。

- CPC = numeric

- 範囲 : 1 ~ 128
- デフォルト値 : 1
- 複数のノードで構成されたTmaxシステムでノードのCLHプロセスのチャンネル数を決める項目です。両ノード間でサービス要求が多かったり、分散トランザクションが多かったりする場合、ノード間でチャンネル数が少なすぎるとサービス処理が遅延され、全体的にシステム性能が低下する恐れがあります。そのため全体システムの性能を向上させるには、各ノードのCLHプロセス間で適切な並列通信のチャンネル数を指定する必要があります。

- MAXFUNC = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 0
- 値を設定しないとデフォルト値が設定され、同機能は使用されません。
- TopEndシステムをTmaxシステムに転換するとき、TopEndシステムで使用する関数をTmaxシステムで提供できるように関数テーブルのサイズを決める項目です。

- LOGOUTSVC = service-name

- 正常、異常状態で実行されるサービスをTmaxサーバーのクライアント・ログアウト・プロセスに登録します。
- LOGOUTSVCはシステムでクライアントの正常なログアウトのためにサービスにログインやログアウトするときに使われます。LOGOUTSVCは設定された時間(CLICKINT、IDLETIME)が過ぎてクライアントが自動的にログアウトされる場合その処理を行います。
- LOGOUTSVC項目を使用時に注意すべき点は、必ずTmaxクライアントである場合のみLOGOUTSVCが実行されることです。一般ソケットの接続が発生する状況では実行されません。service nameのサービスをするプログラムには内部的にtpgetclid()を呼び出してクライアントIDを確認し、ログアウトの原因についてはtpsvinfoの構造体項目のCDを参照します。

設定値	説明
TPNOAUTH(0)	クライアントの正常なログアウト
TPSYSAUTH(1)	Tmaxが認識したログアウト(ネットワークなど)
TPAPPAUTH(2)	異常なログアウト(データ受信の障害など)

- CLICKINT = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 0
- 値を設定しないとデフォルト値が設定され、同機能は使用されません。
- クライアントが正常に起動したかどうかを与えられた監視インターバルでチェックします。単独で使用するより、下記の項目で説明するIDLETIME項目と組み合わせて使用することで、クライアントの正確な状態をTmaxエンジンで監視し、浪費されるリソースを最小化できます。
- この項目をDOMAINセクションに定義すると当該ドメインに接続したすべてのクライアントを監視し、NODEセクションに登録すると当該ノードに接続したクライアントのみ監視します。

- IDLETIME = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 0
- 値を設定しないとデフォルト値が設定され、同機能は使用されません。

- クライアントの遊休時間(idle time)を定義します。CLICKINTを5に設定し、IDLETIMEを30に設定した場合、5秒間隔でクライアントの状態を確認し、30秒以内に要求を受信しないとTmaxエンジンでそのクライアントを削除します。
  - IDLETIMEはクライアントからtpstart()で接続を確立するか、tpcall()、tpacall()の要求があってから有効です。この値はBLOCKTIMEやSVCTIMEより大きい必要があります。IDLETIME値がBLOCKTIMEとSVCTIMEより小さく設定されるとクライアントが応答を受信する前に自動でログアウトされます。
- MAXSACALL = numeric
    - 範囲 : 1 ~ 1024
    - デフォルト値 : 8
    - サーバー・プロセスがtpgetrply無しに要求できるtpacallの数を定義します。サーバー・ライブラリーはそれぞれのtpacallに従うため、テーブル・サイズを設定する初期化MAXSACALLにテーブルを作成します。
- MAXCACALL = numeric
    - 範囲 : 1 ~ 1024
    - デフォルト値 : 16
    - クライアント・プロセスがtpgetrply無しに要求できるtpacallの数を定義します。クライアント・ライブラリーはそれぞれのtpacallに従うため、テーブル・サイズを設定する初期化MAXCACALLにテーブルを作成します。
- TXTIME = numeric
    - 範囲 : 1 ~ MAX\_INT
    - デフォルト値 : 0 (単位: 秒)
    - 小数点単位で設定が可能です。
    - 値を設定しないとデフォルト値が設定され、無限待ちとなります。
    - トランザクション・モードで要求したサービスの処理が与えられた時間内に実行されないと当該サービスの処理を取り消します。
    - サービス実行の前後、XAの処理中にデータベースやネットワーク障害によって無限ブロックが発生するのを防ぐために内部のタイムアウトを適用します。

- NLIVEINQ = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 30 (単位: 秒)
  - Tmaxシステムが複数のノードで構成された場合、ノード間の監視インターバル(秒単位)を決める項目です。Tmaxシステムは隣接する両側のノードをpeer-to-peerで接続し、環境ファイルに登録された順序どおり次のノードを監視することでノード全体が円滑に通信できるように構成されています。
  - NLIVEINQ時間が短いと両ノード間に監視が頻繁に行われ、ノードの異常状態を即刻チェックできますが、通信量が嵩み、通信の負荷が発生する恐れがあります。反対に、NLIVEINQ時間が長すぎると通信量は少ないものの、ノードの異常状態を即時監視することはできません。そのため、システム性能を最適化するには、ネットワーク負荷量、業務の優先順位などを考慮して適切なNLIVEINQ値を決める必要があります。
- NCLHCHKTIME = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 0 (単位: 秒)
  - 各ノードのCLHプロセスが一定時間の間データフローが生じずに指定した時間に達した場合、自動的に各ノードのCLHプロセス間で監視信号を送信します。
  - ファイアウォールは特定時間の間データフローがないと使用プロセスを削除する機能がありますが、この際Tmaxの内部CLHプロセスも影響を受けます。したがって、この機能を使用するときは細心の注意が必要です。
- MAXCONV\_NODE = numeric
  - 範囲 : 1 ~ 2048
  - デフォルト値 : 16
  - CLH別に同時に実行可能な会話型サービスの数です。つまり、すべてのクライアントおよびサーバー・プロセスで同時に会話型サービスを100個まで呼び出すようにするなら、この値に100以上を指定する必要があります。
  - MAXを超えると、TPELIMITエラーが発生します。
- MAXCONV\_SERVER = numeric



- 範囲 : 1 ~ 256
- デフォルト値 : 8
- 各クライアント、サーバー・プロセス別に同時に実行可能な会話型サービスの数であり、MAXCACALL、MAXSACALLと同じ概念です。
- MAXCONV\_NODEと同様、最大値を超える場合、TPELIMITエラーが発生します。
- SECURITY = "NO\_SECURITY" | "DOMAIN\_SEC" | "USER\_AUTH" | "ACL" | "MANDATORY" | "OTHER\_AUTH"
  - デフォルト値 : "NO\_SECURITY"
  - "NO\_SECURITY" | "DOMAIN\_SEC" | "USER\_AUTH" | "OTHER\_AUTH"の中で1つを設定してセキュリティ設定を行います。Tmaxが提供するセキュリティ設定に関する詳しい説明は、[「3.9. セキュリティの環境設定」](#)を参照してください。
- OWNER = user\_name
  - 範囲 : 63字以内のstring
  - 詳細は[「3.9. セキュリティの環境設定」](#)を参照してください。
- IPCPERM = numeric
  - 範囲 : 0600 ~ 0777
  - デフォルト値 : 0600 (IPCPERMがない場合、ファイル接続制御のデフォルト値は600です)
  - IPCPERM(inter-process communication permission mask)は共有メモリーセグメントと\${TMAXDIR}/pathに存在するパイプ・ファイルの制御単位の設定に使われます。また、システム・ログファイル(\${TMAXDIR}/log/slog)の制御単位も設定します。
  - UNIXシステム環境では管理者が個人やグループまたはその他に対してそれぞれファイル接続制御(読み取り、書き込み、実行)を指定することができます。ユーザー(個人)は読み取りと書き込みは可能ですが、グループとその他はできません。
  - IPCPERMフィールドは他のユーザーに開始と終了の権限を与えるための制御の設定に使われます。IPCEPERMがないと他のユーザーはサーバー・プロセスの起動と終了を制御できません。

- TMM、CLL、CLHによって生成されるログファイルの制御単位も設定します。svcllog、ulogログファイルまたはASQCOUNT、PODの設定により起動されるサーバー・ログファイルなど、エンジンの影響から生成されるログファイルはIPCPERMの影響を受けます。
- DOMAINID = numeric
  - 範囲：0 ~ 255
  - デフォルト値：0
  - 複数のドメインで業務を実行するとき特定ドメインを表すためのキー値が必要です。このフィールドはキー値を定義するために使われます。マルチドメイン環境では必ず設定する必要があります。
- CMTRET = Y | N
  - デフォルト値：Y
  - 詳細は、「[3.4. 分散トランザクションの環境設定](#)」を参照してください。
- MAXNODE = numeric
  - 範囲：1 ~ MAX\_INT
  - デフォルト値：32
  - ドメイン内のノードの最大数を設定します。
- MAXSVG = numeric
  - 範囲：1 ~ MAX\_INT
  - デフォルト値：32
  - ノードで利用できるサーバー・グループの最大数を設定します。
- MAXSPR = numeric
  - 範囲：1 ~ MAX\_INT
  - デフォルト値：64
  - 1つのノードで起動できるサーバー・プロセスの最大数であり、各サーバー・プロセスの最大値の合計で計算します。

- MAXSVR = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 64
  - 1つのノードに登録できるサーバーの最大数を設定します。
  - 1つのノードにsvr1、svr2、svr3という3つのサーバーが登録され、各サーバーの最大値が50であるとした場合、MAXSPR=150、MAXSVR=3になります。
  
- MAXSVC = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 512
  - 1つのドメインに登録できるサービスの最大数を設定します。
  - tadminでcfg -dを使って確認したとき、指定した値と異なって表示される理由は、以降動的サービスおよび動的ノードの追加時に必要な予備領域まで含まれているからです。
  
- MAXCPC = numeric
  - 範囲 : 0 ~ 65535
  - デフォルト値 : 32
  - 1つのノード当たりのCPCの最大値を設定します。
  
- MAXTMS = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 32
  - ノードで起動できるTMSの最大数を設定します。
  
- MAXROUT = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 16

- サービスの中でROUTING項目を指定できるサービスの最大数を設定します。

たとえば、30個のサービスがあるとした場合、そのうち10のサービスだけがROUTING項目を指定できるとMAXROUTは10になります。

- MAXROUTSVG = numeric

- 範囲：1 ~ MAX\_INT

- デフォルト値：32

- SERVICEセクションのROUTING項目に指定したルーティングするグループ数(サーバー・グループ、またはドメイン・ゲートウェイ)の合計の最大値を設定します。

たとえば、サービスAのROUTING項目がRT1、サービスBのROUTING項目がRT2であり、RT1のRANGES項目のグループ数が2で、RT2のRANGES項目のグループ数が4であるとしたら、ROUTSVGs数は6になります。したがって、MAXROUTSVGは6以上を設定する必要があります。

```
*SERVICE
A          ROUTING = RT1
B          ROUTING = RT2

*ROUTING
RT1        RANGES = "'a':svg1,*:DGW1"
RT2        RANGES = "'a':svg1,'b':svg2,'c':DGW2,*:DGW1"
```

- MAXRQ = numeric

- 範囲：1 ~ MAX\_INT

- デフォルト値：2

- ノードで利用できるRQの最大数を設定します。

- MAXGW = numeric

- 範囲：1 ~ MAX\_INT

- デフォルト値：2

- ノードで利用できるゲートウェイの最大数を設定します。

- MAXCOUSIN = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 16
- COUSIN項目を指定できるサーバー・グループの最大数を意味します。たとえば、30のサーバー・グループがあるとした場合、そのうち10のサーバー・グループだけがCOUSIN項目を指定できる場合、MAXCOUSINIは10になります。
- MAXCOUSINSVG = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 32
  - SVRGROUPセクションのCOUSIN項目に指定したサーバー・グループの合計の最大値を意味します。たとえば、MAXCOUSINの値が10であり、COUSIN項目に2つのサーバー・グループを指定できる場合、MAXCOUSINSVGの値は20になります。
- MAXBACKUP = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 16
  - BACKUP項目を指定できるサーバー・グループの最大数を意味します。たとえば、30のサーバー・グループがあると仮定し、そのうち10のサーバー・グループのみBACKUP項目を指定できる場合、MAXBACKUPIは10になります。
- MAXBACKUPSVG = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 32
  - SVRGROUPセクションのBACKUP項目に指定したサーバー・グループの合計の最大値を意味します。たとえば、MAXBACKUPの値が10であり、それぞれBACKUP項目に2つのサーバー・グループを指定できる場合、MAXBACKUPSVGの値は20になります。
- MAXTOTALSVG = numeric
  - 範囲 : 1 ~ MAX\_INT

- デフォルト値 : 64
  - 指定できるサーバー・グループの合計の最大値を意味します。
- MAXPROD = numeric
    - 範囲 : 1 ~ MAX\_INT
    - デフォルト値 : 0
    - 値を設定しないとデフォルト値が設定され、同機能は使用されません。
    - TopEndシステムをTmaxシステムに転換するときに使用する項目であり、指定できる製品名(product name)の最大値を意味します。(TopEnd用)
- GWCHKINT = numeric
    - 範囲 : 1 ~ MAX\_INT (単位: 秒)
    - マルチドメイン環境でバックアップ・ノード・ゲートウェイが存在する場合、遠隔のメイン・ノード・ゲートウェイに障害が生じると、バックアップ・ノード・ゲートウェイに接続してサービスを要求します。こうした状況が発生すると、メイン・ノードが復旧された後もバックアップ・ノードにサービスを要求します。メイン・ノードと再接続するにはバックアップ・ノード・ゲートウェイを任意にダウンさせる必要があります。ただし、メイン・ノードの復旧後に、メイン・ノードとのトランザクションを実行する場合は、Tmax環境ファイルにこの項目を設定する必要があります。
    - 以下は、この項目を設定時に考慮すべき事項です。
      - GWCHKINT項目が設定されていない場合、従来の方式で動作します。つまり、メイン・ノードが再起動されてもメイン・ノードとの再接続を試みません。
      - メイン・ノードと接続された後、バックアップ・ノード・ゲートウェイとの接続は任意に解除する必要があります。そうしない場合、メイン・ノードともバックアップ・ノードとも接続を維持します。ただし、トランザクションはメイン・ノードにのみ送信されます。
- GWCONNECT\_TIMEOUT = numeric
    - 範囲 : 1 ~ MAX\_INT (単位: 秒)
    - ドメイン・ゲートウェイは初期システムの起動時にリモートゲートウェイとの接続を試みます。

環境ファイルのDOMAINセクションにGWCHKINT項目が設定されていない状態で、リモート・ゲートウェイがインストールされたマシンが起動されていない場合、要求側は接続に失敗したとのメッセージが出力されるまで、長時間待機することになります。

また、こうした状況でトランザクションがないと再接続を試みず、サービスが要求される時点で再接続を試みますが、この場合にもリモート・ゲートウェイが用意されていない状況であれば長時間待機することになります。これを解決するには、接続タイムアウトを指定します。

- GWCONNECT\_TIMEOUTが設定されていないと従来の方式で動作します。

- TMMLOGLVL = string

- デフォルト値 : DETAIL

- TMMのログレベルを設定します。COMPACT、BASIC、(DETAIL)、DEBUG1、DEBUG2、DEBUG3、DEBUG4のうち1つを選択して設定します。

レベル	説明
COMPACT	E + F
BASIC	E + F + W
DETAIL	E + F + W + I
DEBUG1 ~ DEBUG4	内部デバッグ・ログが最も詳しいのはDEBUG4です。デバッグ用のバイナリおよびライブラリー(clh.dbg、tmm.dbg、libsvrd、libtmsd、libtmsthld)でのみ有効です

- CLHLOGLVL = string

- 範囲 : TMMと同じ

- CLHのログレベルを設定します。

- TMSLOGLVL = string

- 範囲 : TMMと同じ

- TMSのログレベルを設定します。

- LOGLVL = string

- サーバーのログレベルを設定します。

- CRYPT = Y | N

- デフォルト値 : N

- Tmax v4.0からはクライアントとサーバー間の送受信データをより安定的に転送できるように暗号化機能を提供しています。暗号化機能を使用するには、クライアントとサーバーの両方に暗号化機能が設定されている必要があります。

- サーバーで暗号化機能を使用するには、CRYPT項目を「Y」に設定し、使用しない場合には「N」に設定します。クライアントの暗号化機能の設定については、[2.2.2. クライアントの環境変数](#)のTMAX\_SEMANTICS環境変数を参照してください。

- 以下は、この項目を設定時に考慮すべき事項です。

- 最初にシステムにアクセスする際に暗号化キーの交換が行われ、サービスの呼び出しごとに暗号化/復号化(encryption/decryption)が行われるため、オーバーヘッドによる性能低下が発生する恐れがあります。
    - 暗号化機能を使用するには、使用するシステムに'/dev/random'がインストールされている必要があります。暗号化機能を使用すると、最初にシステムにアクセスして暗号化キーの交換が行われる際、カーネルで提供する'/dev/random'という文字装置を使用してランダムキーを生成します。そのため、現在使用中のシステムに'/dev/random'がインストールされているか否かを確認する必要があります。インストール可否はlsコマンドで確認します。インストールされていない場合は、各システムのマニュアルを参照してインストールします。
    - ノード間通信では暗号化をサポートしません。
    - クライアントが\$PATHDIRを使ってアクセスする場合は暗号化しません。
    - 暗号化が設定されていないクライアントが、暗号化の設定されているエンジンにアクセスする場合、TPECLOSEエラーが発生します。暗号化の設定されたクライアントが、暗号化が設定されていないエンジンにアクセスする場合、クライアントはブロックされます。

- CRYPT\_ALGORITHM = "3DES" | "AES"

- デフォルト値 : 3DES

- Tmax v5.0 SP3から暗号化機能を使用する際に暗号化方式を指定できます。

- 3DES方式とAES方式をサポートします。

- サーバー環境設定ファイルのCRYPT\_ALGORITHMとクライアント側の環境変数のTMAX\_CRYPT\_ALGORITHMが同じ方式に指定されているかを確認します。



- この設定を適用するためには、CRYPT=Yに設定する必要があります。
- MAXTHREAD = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 0
  - 値を設定しないとデフォルト値が設定され、当該機能は使用されません。
  - Tmax v4.0以降バージョンでは、マルチスレッド方式のトランザクション・マネージャーを提供し、少ないリソースを効率的にトランザクション処理に利用できます。マルチスレッドTMSを使用するには、この項目にマルチスレッドTMSの最大スレッド数を指定します。また、SVRGROUPセクションにTMSTYPE、TMSTHREAD、TMSOPT項目も設定します。各項目の詳細については、「[3.2.3 SVRGROUPセクション](#)」のTMSTYPE、TMSTHREAD、TMSOPTを参照してください。
  - 以下は、この項目を設定時に考慮すべき事項です。
    - DBMSがマルチスレッドに対応する必要があります。(現在Oracleが対応中)
    - Oracleの場合、環境ファイルのSVRGROUPセクションのOPENINFO項目に「Thread=true」を追加します。

```
OPENINFO="Oracle_XA+Acc=P/scott/tiger+SesTm=60+Thread=true"
```
    - マルチスレッドTMSを生成時は、必ずlibtmsの代わりにlibtmsthr、libpthreadライブラリーをリンク付けてコンパイルします。
- TDL = Y | N
  - デフォルト値 : N
  - TDL(Tmax Dynamic Library)の暗示的なバージョン保護機能を使用するかどうか(Y|N)を設定します。

---

#### 参考

TDLと関連する詳細内容は、『Tmax プログラミングガイド(ダイナミックライブラリ)』を参照してください。

---

- MAXSESSION = numeric
  - 範囲 : 1 ~ 65535

- デフォルト値 : 1024
- 1つのドメイン内のHMSで生成できるセッション数の最大値を設定します。詳細は、「[3.7. HMS環境設定](#)」を参照してください。
- TXPENDINGTIME = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 0
  - ペンディングされたトランザクションのタイムアウトを設定します。タイムアウト時間の間ペンディングされたトランザクションの応答がなければロールバックさせます。設定しない場合、検査を行いません。
- CLIENT\_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - デフォルト値 : "IPV4"
  - Tmaxシステムでクライアントの接続ポートを作成時に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。この場合、クライアントはIPv4でのみ接続できます。ノードごとに独立した設定が可能です
IPV6	IPv6プロトコルを使用して接続要求を待ちます。クライアントはIPv4またはIPv6環境でTmaxシステムに接続できます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPV4またはIPV6と一緒に使用できます

- SYSTEM\_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - デフォルト値 : "IPV4"
  - Tmaxシステムが複数のノードで構成されている場合に、ノード間の通信のためポートを作成するときに使用するプロトコルを決定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です

設定値	説明
IPV6	他のノードへの接続要求のためポートを作成するとき、IPv6プロトコルを使用して接続要求を待ちます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- EXTSVR\_PROTOCOL = "IPv4" | "IPv6" | "SDP"

– デフォルト値 : "IPv4"

– Tmaxシステムから外部サーバーに接続するためポートを作成する際に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です
IPV6	IPv6プロトコルを使用して接続要求を待ちます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- CLIENT\_IPV6 = Y | N

– デフォルト値 : N

– Tmaxシステムでクライアントの接続のためのポートを作成する際に、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	IPv6プロトコルを使用して接続要求を待ちます。クライアントはIPv4環境またはIPv6環境でTmaxシステムに接続することができます
N	IPv4プロトコルを使用して接続要求を待ちます。この場合、クライアントはIPv4でのみ接続できます。ノードごとに独立した設定が可能です

## 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにCLIENT\_PROTOCOL項目を使用することをお勧めします。

- SYSTEM\_IPV6 = Y | N

– デフォルト値：N

– Tmaxシステムがマルチノードで構成されており、ノード間通信のためのポートを作成する際に、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	他のノードの接続要求のためにポートを作成する際、IPv6プロトコルを使用して接続要求を待ちます
N	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です

---

#### 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにSYSTEM\_PROTOCOL項目を使用することをお勧めします。

---

#### ● EXTSVR\_IPV6 = Y | N

– デフォルト値：N

– Tmaxシステムから外部サーバーに接続するためポートを作成する際に、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	IPv6プロトコルを使用して接続要求を待ちます
N	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です

---

#### 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにEXTSVR\_PROTOCOL項目を使用することをお勧めします。

---

#### ● CLL\_BIND\_IP = Y | N

– デフォルト値：Y

– クライアント接続のためのポートを作成する際、NODEセクションのIPに設定されたアドレスのインターフェースを使用するかどうかを指定します。

設定値	説明
N	「N」に指定するか、あるいはIPを設定しないと、すべてのインターフェースからの接続を許可します

- FDL\_VERSION = numeric

- デフォルト値 : 1

- 範囲 : 1 ~ 2

- 2に指定すると、検索パフォーマンスが向上されたバージョンで動作しますが、既存のTmaxとの互換性はありません。

同オプションの影響を受けるのは、サーバー・アプリケーション、ROUTINGセクションの設定に伴うCLHのスケジューリングです。fb\* APIの使用時に、FDL\_VERSION設定が自身の環境と一致しない場合はFBSETVERエラーを返します。

- MSGSIZEWARN = numeric

- デフォルト値 : 1073741824

- 範囲 : 1024 ~ 1073741824

- クライアント/サーバーから受信するメッセージのサイズが指定した値を超えると、slogに警告メッセージが出力され、正常に処理されます。

- MSGSIZEMAX = numeric

- デフォルト値 : 1073741824

- 範囲 : 1024 ~ 1073741824

- クライアント/サーバーから受信するメッセージのサイズが指定した値を超えると、slogにエラー・メッセージが出力され、当該接続を終了します。

ROC = Nに指定する場合、接続が終了されたサーバーはCLHと再接続できないので、ROC = Yに指定します。

- CASCPC = numeric

- デフォルト値 : 1

- 範囲 : 1 ~ 65535

- CASプロセスと自ノードのCLHプロセス間のCPC数を設定します。
- 現在、Tmax6.0では**CASCPCは1のみ**サポートします。今後、Fix1以降複数のCPC数を設定できるようにサポートする予定です。

## 使用例

以下は、DOMAINセクションの使用例です。

```
*DOMAIN
res1      SHMKEY = 77990, MAXUSER = 300,
          MINCLH = 2, MAXCLH = 3,
          TPORTNO = 8889, RACPORT = 3334,
          BLOCKTIME = 60, CLICKINT = 5,
          IDLETIME = 30, LOGOUTSVC = logout,
          CLHLOGLVL = DEBUG2
```

### 3.2.2. NODEセクション

DOMAINセクションでTmaxシステム全体に対する環境を定義した後、Tmaxシステムを構成している各ノードの環境設定を行います。

NODEセクションには以下の内容を定義できます。

- Tmaxシステムのホームディレクトリーと作業ディレクトリー
- アプリケーション・サーバー・プログラムの実行ファイルを含むディレクトリー
- Tmaxプロセス間の通信のためのディレクトリー
- システム・メッセージを保存するディレクトリー
- ユーザー・メッセージを保存するディレクトリー
- トランザクション情報を保存するディレクトリー
- アプリケーション・サーバー・プログラムの起動時に必要な環境変数を定義した環境ファイル

NODEセクションの一部項目はDOMAINセクションで定義した場合もノードによって別途定義することができます。再定義可能な項目は3.2.2節「[再定義できる項目](#)」の説明を参照してください。

Tmaxシステムを起動させるとノードごとに、TMM、CLL、CLH(データベース関連システムの場合TMSを、マルチドメインの場合ゲートウェイ・プロセスを含む)プロセスと実際のサービスを実行するサーバー・プロセスが起動されます。このようなプロセスを起動するには実行ファイルが存在するディレクトリーを把握する必要

があります。また、Tmaxプロセスの通信に必要なディレクトリーとTmaxシステムやユーザーからの各種エラーと警告メッセージを保存するディレクトリーも指定できます。

NODEセクションの基本環境設定の形式は以下のとおりです。

```
*NODE
[DEFAULT: ]
Node Name      TMAXDIR = tmax-home-path ,
                APPDIR = application-path,
                [HOSTNAME = HOST_NAME,]
                [TMAXHOME = tmax-home-path,]
                [NODETYPE = SHM_RACD/(SHM_USER),]
                [PATHDIR = stream-pipe-path,]
                [SLOGDIR = system-log-path,]
                [TLOGDIR = transaction-log-path,]
                [ULOGDIR = user-log-path,]
                [DOMAINNAME = domain-name,]
                [CLHQTIMEOUT = 1 - MAX_INT,]
                [ENVFILE = environment-file-name,]
                [SHMKEY = shared-memory-segment-key,]
                [MAXUSER = 1 ~ ,]
                [MINCLH = 1 ~ 10,]
                [MAXCLH = 1 ~ 10,]
                [TPORTNO = port-number,]
                [TPORTNO2 = port-number,]
                [TPORTNO3 = port-number,]
                [TPORTNO4 = port-number,]
                [TPORTNO5 = port-number,]
                [RACPORT = port-number,]
                [IPCPERM = mask,]
                [IP = IP address,]
                [CRYPT_ALGORITHM=("3DES")|"AES",]
                [CLCHKINT = interval-time-value,]
                [IDLETIME = idle-time,]
                [TMMOPT = TMM-log-path,]
                [TLMOPT = TLM-transaction-log-file-size,]
                [CLHOPT = CLH-log=path,]
                [REALSVR = server-name,]
                [RSCPC = 1 ~ 128,]
                [MAXSVG = 1 ~ MAX_INT,]
                [MAXSPR = 1 ~ MAX_INT,]
                [MAXSVR = 1 ~ MAX_INT,]
                [MAXTMS = 1 ~ MAX_INT,]
                [MAXCPC = 0 ~ 128,]
                [MAXGWSVR = 1 ~ MAX_INT,]
                [MAXRQSVR = 1 ~ MAX_INT,]
```

```

[MAXGWCPC = 1 ~ MAX_INT,]
[AUTOBACKUP = (Y)|N,]
[LOGOUTSVC = logout-service-name,]
[TMAXPORT = port num1, ..., port number5,]
[COMPRESSPORT = port num1, ..., port number5,]
[COMPRESSSIZE = compress_size,]
[RESTART = (Y)|N,]
[MAXRSTART = numeric,]
[GPERIOD = numeric,]
[TMMLOGLVL = tmm-log-level,]
[CLHLOGLVL = clh-log-level,]
[TMSLOGLVL = tms-log-level,]
[LOGLVL = server-log-level,]
[EXTPORT = port number,]
[EXTCLHPORT = clh port number,]
[SMSUPPORT = Y | (N),]
[SMTBLSIZE = num,]
[CLLBLOCK = Y|(N),]
[CLLUNBLKPORT = Portno1, Portno2, ...,]
[CLLBLOCKTIME = num]
[CLLCONNLB=(RR)|LC,]
[CRYPTPORT=literal,]
[TRB = nodename,]
[MAXSESSION = 1 ~ MAX_INT,]
[SQKEY = 32768~262143 ,]
[SQSIZE = 4 ~ 4000000,]
[SQMAX = 2~MAX_INT-1,]
[SQKEYMAX = 1 ~ MAX_INT,]
[SQTIMEOUT =1 ~ MAX_INT,]
[CLIENT_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[SYSTEM_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[EXTSVR_PROTOCOL = ("IPV4")|"",IPV6"|"",SDP",]
[CLIENT_IPV6 = Y|(N),]
[SYSTEM_IPV6 = Y|(N),]
[EXTSVR_IPV6 = Y|(N),]
[CLL_BIND_IP = (Y)|N,]
[SVCLOG_FORMAT = svclog-format-string,]
[CASTHREAD = 1 ~ 65535,]
[CASOPT = literal,]
[SECURITY_LIB = literal,]
[CRYPT_LIB = literal,]
[TGOPT = literal]

```

## 必須項目

- Node Name = string



- サイズ：63字以内
- ノードの物理名で、UNIXの「uname -n」コマンドで確認した名前を定義します。  
 設定したノード名は必ず「/etc/hosts」ファイルに登録されている必要があります。1つのドメインは1つ以上のノードで構成されるため、NODEセクションには少なくとも1つ以上のノード名の定義が必要です。
- TMAXDIR = literal
  - サイズ：255字以内
  - Tmaxプログラムがインストールされているホームディレクトリーの絶対パスです。パス名には環境変数のTMAXDIRと同じ値を定義します。TMAXHOMEが定義されていない場合、Tmax関連作業はすべてTMAXDIRディレクトリーで実行されます。
- APPDIR = literal
  - サイズ：255字以内
  - Tmaxを利用するアプリケーションの実行ファイルが存在するディレクトリーの絶対パスであり、サーバーの作業ディレクトリーの絶対パスでもあります。
  - TmaxシステムにはTmax自体プロセス以外にもクライアントからの要求を処理するアプリケーションがあります。アプリケーションはTmaxシステムが起動するとき、一緒に起動し、終了するときも一緒に終了します。
  - アプリケーションの実行ファイルのインストール先ディレクトリーを定義する項目は**APPDIR**です。また、APPDIRはサーバーの作業ディレクトリーでもあり、サーバープログラムで発生するcoreファイルがAPPDIRで保存されます。

## 選択項目

- DEFAULT: 項目 = 値, ...
  - NODEセクションのみならず、Tmax環境ファイルのすべてのセクションで定義できるラベルです。  
 NODEセクションのこのラベルは、複数ノードで構成された場合、各ノードに同一値をもつ項目を定義するときに有効に使用できます。ラベルに「DEFAULT:」が定義されると、その定義は当該セクションが終わるか他の「DEFAULT:」によってオーバーライドされるまで有効です。  
 「DEFAULT:」によって定義された項目がNODEセクションに別途定義されていない場合は、「DEFAULT:」によって定義された値が全ノードに共通に適用されます。
  - 再定義されると、該当ノードにはNODEセクションに定義した新しい値が適用されます。

- HOSTNAME = literal

- サイズ : 255字以内

- 実際のホスト名と環境ファイルに定義するノード名が異なる場合にも動作できるようにするための項目であり、実際のホスト名を指定します。ホスト名を指定した場合、1つのマシンで論理的なマルチノードを使用することができます。1つのホスト名に複数のノード名を使用できます。

- NODEセクションにはTMAXHOME、NODETYPE、RACPORT、TPORTNO、SHMKEY項目が一緒に定義される必要があります。定義されていない場合、ノード名がホスト名になります。詳細については、下記のNODEセクションの[3.2.2節「使用例」](#)を参照してください。

- TMAXHOME = literal

- サイズ : 255字以内

- Tmaxのインストール先ディレクトリーと作業ディレクトリーを分離して使用できるようにするための項目であり、Tmaxプログラムがインストールされているホームディレクトリーの絶対パスを定義します。このパスは環境変数のTMAXHOMEと同一値で定義します。TMAXHOME項目が定義されている場合はTMAXDIRは作業ディレクトリーになり、定義されていない場合はTMAXDIRがインストール先ディレクトリーと作業ディレクトリーになります。

Tmaxのインストール先ディレクトリーと作業ディレクトリーは下記の表のように区分できます。

区分	説明
TMAXHOME	インストール・ディレクトリーであり、bin、lib、usrinc、tuxinc、topinc、cblinc、licenseなどの下位ディレクトリーがあります
TMAXDIR	作業ディレクトリーであり、config、path、log、svctなどの下位ディレクトリーがあります

HOSTNAME項目と一緒に設定された場合はそれぞれのTMAXDIRが1つの論理ノードになり、TMAXHOMEは各論理ノードが共有するTmaxのインストール先ディレクトリーになります。したがって、同じノード内の論理ノードのTMAXDIRとSHMKEY、TPORTNOの設定は互いに異なる必要があります。

- NODETYPE = SHM\_RACD | SHM\_USER

- デフォルト値 : SHM\_RACD

- 1つのマシン内に複数のローカル・ノードを定義した場合、それぞれのローカル・ノードを管理する方法を決定する項目です。1マシンのローカル・ノードのRACPORTは必ず異なるように設定する必要があります。したがって、2つ以上のローカル・ノードを使う場合は、RACPORTを必ず再定義します。

重要なことはNODETYPE項目は「tmboot/tmdown/tmadmin/cfl」を実行する論理ノードと同じホストにあるノードだけを管理するときに使用するということです。他のホストにある論理ノードを管理する場合にはNODETYPEに関係なくRACPORTを使用します。

- 「tmdown/tmadmin/cfl/tmboot」を開始した論理ノード(node A)と同じホストに他の論理ノード(node B)が存在する状態で、その他ノード(node B)の「tmdown/tmadmin/cfl/tmboot command」を実行するとき、racdを介して実行するか直接実行するかを決めるときに使います。
- 以下は、NODETYPEの設定値についての説明です。

設定値	説明
SHM_RACD	RACPORTを利用して各論理ノードを管理します
SHM_USER	RACPORTを利用せずに各論理ノードを直接管理します

- PATHDIR = literal

- サイズ : 255字以内
- デフォルト値 : \${TMAXDIR}/path
- Tmaxシステムのプロセスが通信するための名前付きストリーム・パイプ(Named stream pipe)のパスです。プロセス間の通信には、パイプ(Unnamed pipe, Named pipe)、メッセージ・キュー(Message queue)、共有メモリー(Shared memory)を使用する方法があります。

Tmaxではプロセス間で情報を送受信するため、共有メモリー(「[3.2.1 DOMAINセクション](#)」のSHMKEY項目を参照)と共にストリーム・パイプ(Stream pipe)方式を使います。ストリーム・パイプを使うためにはパス名(pathname)が必要です。

- PATHDIRはTmax関連プロセスのストリーム・パイプ経由の通信に必要なパス名が生成されるディレクトリー名です。プロセス通信のため、このディレクトリーに名前付きパイプ(named pipe)が生成されます。この項目が定義されていない場合は、TMAXDIRで指定したディレクトリー下のpathディレクトリーにパイプが生成されます。
- PATHDIRを「\$TMAXDIR/path」以外のディレクトリーに指定すると、Tmaxシステム終了時にシステムが異常終了する場合があります。それは、Tmaxシステム終了時に、システムが「\$TMAXDIR/path」ディレクトリーにあるストリーム・パイプ(named pipe)を使用するためです。これを解決するには、環境変数のTMAX\_PATHDIRに、ユーザーが環境ファイルに設定したPATHDIRを指定する必要があります。指定後にTmaxシステムを終了すると、Tmaxシステムは環境変数に指定したTMAX\_PATHDIRを参照して終了するため、正常終了できます。

```
export TMAX_PATHDIR = /home/tmax/path_new
```

- Tmaxシステムを起動時に、設定(config)ディレクトリーにあるバイナリ形式の環境ファイルが環境ファイルに設定したPATHDIRにコピーされます。これはTmaxシステムが起動した後に環境ファイルが修正された場合に、正常終了できない問題を解決するためのものです。やはり環境変数のTMAX\_PATHDIRに、変更前(Tmaxシステムが起動するときに参照したPATHDIR)のPATHDIRを設定すると、このディレクトリー下にバイナリ形式の環境ファイルが生成されます。そしてその環境ファイルはTmaxシステムが起動した後に環境ファイルが修正されても変更されません。

この方法で、Tmaxシステムの起動後にユーザーが環境ファイルを修正してもTmaxシステムが異常終了しないように防止できます。

- SLOGDIR = literal

- サイズ : 255字以内
- デフォルト値 : \${TMAXDIR}/log/slog
- システム・メッセージが記録(logging)されるディレクトリーの絶対パスを指定します。システム・メッセージは、TMM、TMS、CLL、CLHなどが出力するメッセージと、システム内部で発生したメッセージをいいます。指定されたSLOGDIRディレクトリーには日付別に「slog.日付」ファイルが生成され、各ファイルに日付別に発生したメッセージがロギングされます。
- この項目を定義していない場合はTMAXDIRで指定したディレクトリー下の「log/slog」ディレクトリーに記録されます。

---

#### 注

指定するパスは使用中のシステムに存在することが前提です。

---

- TLOGDIR = literal

- サイズ : 255字以内
- デフォルト値 : \${TMAXDIR}/log/tlog
- XAを使ったトランザクションが発生したとき、関連情報をTXLOGファイルに記録します。TXLOGファイルが生成される絶対パスを指定します。バイナリ・ファイルであるため解読はできず、Tmaxエンジンでのトランザクション情報の管理に必要です。TXLOGファイルはファイルシステムで生成し、Tmaxではrawファイルによってこのファイルの記録をサポートしません。
- この項目を定義していない場合は、TMAXDIRで指定したディレクトリー下の「log/tlog」ディレクトリーに記録されます。

---

## 注

指定するパスは使用中のシステムに存在することが前提です。

---

- 起動時に以前起動したときのログが残っている場合は既存のファイルを.oldにバックアップし、新しいトランザクション・ログファイルが生成されます。基本的に最大16MBで、この値はTLMOPTの[-l]オプションを使って修正できます。
  - txlogに残る1つのxidのサイズは約32Byteです。(txlogは無限に大きくなりません)
  - ULOGDIR = literal
    - サイズ : 255字以内
    - デフォルト値 : \${TMAXDIR}/log/ulog
    - ユーザー・メッセージが記録(logging)されるディレクトリーの絶対パス名を指定します。
    - ユーザー、つまりTmaxアプリケーション・プログラマーはuserlog()関数を利用して、デバック用のメッセージ、各種エラー及び警告メッセージなどを簡単にロギングできます。
    - ユーザーがuserlog()関数を使用して送信するメッセージを保存するディレクトリーがULOGDIRです。指定したULOGDIRディレクトリーには日付別に「ulog\_日付」ファイルが生成され、各ファイルには日付別に発生したメッセージが記録されます。userlog()関数についての詳細はTmaxガイドのうち、『Tmaxアプリケーション開発ガイド』を参照してください。
    - この項目を定義していない場合は、TMAXDIRで指定したディレクトリー下の「log/ulog」ディレクトリーに記録されます。
- 

## 注

指定するパスは使用中のシステムに存在することが前提です。

---

- DOMAINNAME = string
  - サイズ : 63字以内
  - 1つの環境ファイルに複数のドメインを定義できるため、該当ノードが属したドメイン名を指定する必要があります。ドメイン名は必ずDOMAINセクションで定義した名前である必要があります。
- CLHQTIMEOUT = numeric

- 範囲：1 ~ MAX\_INT
  
- メッセージがキューに指定された時間以上待機状態にある場合、時間が経過したメッセージをキューから削除する機能です。ホストやネットワーク問題などによってメッセージが溜まる場合、無限に溜めるのではなく、クライアントにその状況を知らせて適切な措置を取るようにする設定値です。
  
- ENVFILE = literal
  - サイズ：255字以内
  
  - ENVFILEは、アプリケーション・サーバー・プログラムの実行に必要な特定環境を設定するためにUNIX環境変数が定義されている環境ファイルを指定します。したがって、サーバープログラムはENVFILEに指定された環境で実行されます。
  
- TPORTNO[2-5] = numeric
  - クライアントがサーバーに接続時に使用するTmaxシステムのポート番号です。TPORTNO項目で定義したポート以外の他ポートでTmaxシステムに接続するために指定します。
  
  - TPORTNO2からTPORTNO5までをすべて定義するとTmaxシステムに接続できるポートは計5つです。一般的にシステム領域では1024以下のポート番号を使うため、管理者はこの範囲を避けて指定する必要があります。
  
  - TPORTNOで定義したポート番号とその次のポート番号(TPORTNO+1)はクライアントとTmaxシステム、または他ノード間の通信に使われるため、他のポート番号を指定する必要があります。また、他のところで使われているかどうかの確認も必要です。TPORTNOで指定したポート番号は既に指定したポート番号と異なる必要があります。
  
- TMMOPT = literal
  - サイズ：255字以内
  
  - TMMの起動時にTMMプロセスに送信されるコマンドオプションを定義します。定義されたオプションのうち「--」以前に指定されたオプションはシステムで使用し、その後に指定されたオプションはユーザーが自由に使用できます。
  
  - 以下はユーザー・オプションについての説明です。

オプション	説明
-e ファイル名	TMMの動作中に発生する標準エラー(standard error)をファイルに記録します

オプション	説明
-o ファイル名	TMMの動作中に発生する標準出力(standard output)をファイルに記録します
-i   w   e   f	<p>サーバー・プロセスのタイプ(SVRTYPE)がEVT_SVRである場合、SLOGが発生時に呼び出されるコールバック関数のログレベルを指定するときに使われます。</p> <ul style="list-style-type: none"> <li>- i : fatal、error、warn、info (-iを指定すると、infoレベル以下に対してすべてコールバック関数が呼び出されます)</li> <li>- w : fatal、error、warn</li> <li>- e : fatal、error (デフォルト値)</li> <li>- f : fatal (-fを指定すると、slogでfatalエラーの場合のみコールバック関数が呼び出されます)</li> </ul>
-t 秒	<p>ASQCOUNTで起動されたサーバー・プロセスの接続タイムアウト時間を変更できるオプションです。運用中にtmadminのsetoptコマンドを使って動的に変更することができます(デフォルト値: 10秒、単位: 秒)</p> <p>(例: TMMOPT = "-t 20" )</p>
-B backlog値	<p>サーバー・プロセスの接続を処理するListenポートのbacklog値を指定します。設定可能な値は、1からSOMAXCONN値です。(デフォルト値: 1023)</p> <p>設定値が小さすぎると、多数のサーバー・プロセスが同時にTMMに接続する際、接続に失敗することがあります</p>
-A 処理回数	<p>サーバー・プロセスの接続要求に対してACCEPT処理を行うとき、一度に処理する数を指定します。(デフォルト値: 100)</p> <p>運用中にtmadminのsetoptコマンドを使って動的に変更することができます</p>
-F 同時処理の最大数	<p>TMMでサーバー・プロセスの追加起動や再起動などにより、新しいプロセスを作成するとき、同時に処理できる最大数を指定します。システムのパフォーマンスや負荷状態に応じて、新しいプロセスの起動に影響を与える場合、同オプションを使って調整することができます。設定可能な範囲は、0から現在設定されているbacklog値までです。(デフォルト値: backlog設定値の0.75倍)</p> <p>0に設定すると、無制限で新しいプロセスをすぐに起動させます。運用中にtmadminのsetoptコマンドを使って動的に変更することができます。0</p>

オプション	説明
	<p>に変更すると、変更時点ではまだ起動していない待機中のプロセスをすべて起動させます。</p> <p>プロセスの起動数は、以下のような動作により計算されます。起動要求によりプロセスを作成するとき、現在起動中のプロセス数を1つ増やします。起動されたプロセスがTMMIに接続した後、Registeredされるたびに1つ減らします。起動されたプロセスでexec()が失敗するか、あるいは登録される前に失敗する場合は、[-t]オプションに定義されている起動時間(boottime)を確認して1つ減らします。特定のプロセスに実行ファイルがないか、または依存ライブラリー(dependent library)エラーなどで正常に起動できない場合は、boottimeの間、起動数を維持します。</p> <p>現在起動しているプロセスの数が[-F]オプションの設定値に達している状態で、追加的な起動要求が発生すると、その起動要求は現在起動中のプロセスがレジスターを完了して余裕が生じるまで待機します</p>
-r 間隔	秒単位(-kオプションを参照)
-k 同時再接続要求プロセス数	<p>CLHが再起動すると、TMMIはすべてのサーバー・プロセスに新しく起動するCLHと接続を確立するよう通知します。この際、多数のプロセスがCLHに同時に接続しようすると、正常に動作していたサーバーがブロック状態になる可能性があるため、それを防止するために-r、-kオプションを使用して、-r時間の周期で-k個の同時接続プロセス数を制限します。</p> <p>このオプションはCLHに再接続するために同時に要求を送るプロセス数を指定します。-rと-kオプションを一緒に使用した場合に動作します。外部サーバーは直ちに接続するよう通知し、オプションが適用されません。tmadminのnrcコマンドにより、特定のサーバーがCLHに接続を確立するようにすることができます</p>
-g	<p>ファイル出力のための別途のスレッドを作成して処理します。設定しない場合は、従来どおりメイン・スレッドで処理します。</p> <p>TMMIはslog、trace log、svclogをファイルに出力します。TMAX_TRACEログを使用し、サーバー数が多くてサービス時間が大変短く、負荷のある状況では、TMMIがトレース・ログを記録するためにファイルに出力するのに遅延処理が発生して結果的にサービスの実行時間が遅くなる原因になることがあります。このオプションを使用すると、ファイル出力を別途のスレッドで行うようにしてこの問題を解決することができます。</p> <p>-mオプションと関連があります</p>
-m	<p>ファイル出力が遅い場合、保管する要求数を指定します。何も指定しない場合、10000個に固定します。1つの要求を保管するのに約8192バイトが消費されます。</p>



オプション	説明
	<p>-gオプションと一緒に使う必要があります。</p> <p>TMMOPT = "-g -m 10000"設定がある場合やslogまたはulogに記録するサーバー要求がある場合、受信した要求TPSがファイル書き込みが可能な件数より多い場合は、-mに設定された数までキューイングして処理します。処理ができなくなった場合は、要求を捨てます。</p> <p>-mを無限に増やす場合、TMMのメモリー・サイズが大きくなり過ぎることがあり、TMMでasqcountや異常終了時の再開始などの機能を実行(fork)時に、メモリー・サイズのため問題が発生することがあります</p>

## 参考

オプションについての詳細内容は『*Tmax* リファレンスガイド』の「3.2.2. `_tmax_event_handler`」を参照してください。

### ● TLMOPT = literal

- サイズ : 255字以内
- TLMの起動時にTLMプロセスに送信されるコマンド・オプションを定義します。
- 使用されるユーザー・オプションは以下のとおりです。

オプション	説明
-l ログファイルのサイズ	設定しない場合、デフォルトで16MBのトランザクション・ログファイルを作成します(単位 : MB、最大値 : 2048MB)

### ● CLHOPT = literal

- CLHの起動時にCLHプロセスに送信されるコマンド・オプションを定義します。詳細オプションはTMMOPTと同様です。

オプション	説明
-e ファイル名	CLHの動作中に発生する標準エラー(standard error)をファイルに記録します
-o ファイル名	CLHの動作中に発生する標準出力(standard output)をファイルに記録します
-h   w   e   f	サーバー・プロセスのタイプ(SVRTYPE)がEVT_SVRである場合、SLOGが発生時に呼び出されるコールバック関数のログレベルを設定するときに使われます

オプション	説明
	<ul style="list-style-type: none"> <li>- i : fatal、error、warn、info (-iを設定する場合、infoレベル以下に対してすべてコールバック関数が呼び出されます)</li> <li>- w : fatal、error、warn</li> <li>- e : fatal、error (デフォルト値)</li> <li>- f : fatal (-fを設定する場合、slogでfatalエラーの場合のみコールバック関数が呼び出されます)</li> </ul>
-L サービス名	<p>サービス呼び出しをしたクライアントやサービスが終了または再開始されると、CLHで呼び出されたサービスからの応答メッセージを廃棄(Discard)します。応答メッセージが廃棄されたら、ユーザーが指定したサービス(Loss Service)を呼び出し(tpacacall with TPNOREADY   TPNOTRAN)できます。</p> <p>サービス名に応答メッセージが廃棄されるときに呼び出されるLoss Serviceを指定します。Loss Serviceに指定されたサービスはTPSVCINFOのcltidを通じて追加情報の送信を受けます。</p> <ul style="list-style-type: none"> <li>- cltid.clientdata[1]: 廃棄(Discard)された応答のtperrno</li> <li>- cltid.clientdata[2]: 廃棄(Discard)された応答のtpurcode</li> <li>- cltid.clientdata[3]: 廃棄(Discard)された応答のサービス・インデックス(tpgetsvcname()関数のパラメータとして使用)</li> </ul>
-r カウント	<p>CLHでサーバやクライアントと非ブロッキングI/Oを実行中、読み取りの最大のリトライ回数をカウント値で設定します。(デフォルト値 : 100)</p> <p>カウント値はデフォルト値以上1000以下の値を使用することをお勧めします</p>
-w カウント	<p>CLHでサーバやクライアントと非ブロッキングI/Oを実行中、書き込みの最大のリトライ回数をカウント値で設定します。(デフォルト値 : 100)</p> <p>カウント値はデフォルト値以上1000以下の値を使用することをお勧めします</p>
-x 1	<p>TPENOENTがエラーを発生させる場合、SLOGに当該エラーを記録します。設定しない場合は、SLOGに記録せずに、クライアントでのみエラーを確認することができます</p>
-s 秒	<p>クライアントが接続した後、TPSTARTメッセージに待機する時間を設定します(デフォルト値: 60秒)</p>
-f	<p>NCLHCHKTIMEが設定されている場合、CLHのチャンネルが終了した際に、チェック周期まで待たずに直ちに再接続を行います</p>

オプション	説明
-c	cousinサーバーの状態情報を他のノードに通知する役割はTMMが担当していましたが、CLHOPTに[-c]オプションを指定すると、CLHが担当することになります。TMM、CLH、サーバー間の時間差起動により、サーバーの状態情報が衝突するのを防ぎます。多くの負荷が生じます
-k	Tmax v5.0以上のバージョンと4.0 sp3 fix9以下のバージョンで構成されたマルチドメイン環境において、ドメイン同士がtmaxgwntで接続されており、かつCOUSINで設定されたサービスが存在する場合に設定します。  遠隔ドメインでゲートウェイとCOUSINに設定されたサービスが再起動される場合、正常にスケジューリングされるようにします

- CLHで指定されたサービスにメッセージを送信するとき、以下の条件を満たす必要があります。
  1. tpccall() または tpacall() への応答である必要があります。
  2. 正常応答またはエラー応答に関係なく、データが存在する必要があります。
  3. CLHが異常終了する場合には送信されないこともあります。
- REALSVR = literal
  - RDPタイプのサーバー・プロセスを運用したいとき、サーバー・プロセス名を設定します。
  - リアルサーバーは1つのノードに唯一である必要があります。該当ノードに存在するすべてのサービスはRDPサーバー・プロセスによってクライアントに直接送信されます。そのため、他サーバー・プロセスが処理した結果を取得するためのチャンネル数のRSCPC値を設定する必要があります。また、CLHとRDPサーバー・プロセスの数は一定に維持します(つまり、MIN値とMAX値を同一にします)。ただし、RDPサーバー・プロセスはCLHより常に多く設定します。通常は2倍数に設定します。
- RSCPC = numeric
  - 範囲 : 1 ~ 128
  - RDPサーバー・プロセスと他サーバー間の通信に使われるチャンネル数です。
- MAXSVG = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 32
  - 当該ノードに最大に設定できるサーバー・グループ数を意味します。

- MAXTMS = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 32
  - 当該ノードで起動できるTMSの最大数を設定します。
  
- MAXGWSVR = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 2
  - MAXGWが1つのノードで使えるゲートウェイの最大数であれば、MAXGWSVRは該当ノードで起動できるゲートウェイ・サーバー・プロセスの最大数です。デフォルト値は2です。つまり、2つのゲートウェイがあり、各ゲートウェイは1つのゲートウェイ・サーバーを持つことができます。
  
- MAXRQSVR = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 2
  - 当該ノードで起動できるRQサーバー・プロセスの最大数です。デフォルト値は2です。つまり、2つのRQがあり、各RQは1つのRQサーバーを持つことができます。
  
- MAXGWCPC = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 8
  - ゲートウェイ・サーバーとCLH間の接続数を指定します。
  - サーバーとのCPCは基本的にPAIR(input channel/output channel)で構成されるため、CPC=2の環境でゲートウェイ・サーバー数が2であれば8の接続が必要になるため、MAXGWCPCは8以上の値を設定する必要があります。また、MAXGWCPC値の合計はMAXCPCの値より小さく設定します。

MAXGWCPCの値を増やす場合にはMAXCPC値も一緒に増加させる必要があります。
  
- AUTOBACKUP = Y | N
  - デフォルト値 : Y

- バックアップ・サーバーを指定した場合、メイン・サーバーに問題が発生したとき自動的にバックアップ・サーバーを起動するかどうかを設定します。

- TMAXPORT = "port num1, ... , port num5"

- TPORTNO、TPORTNO[2-5]を代替するための項目であり、複数のポートを使う場合、TPORTNO[2-5]よりはTMAXPORTを使うことを推奨します。

- COMPRESSPORT = "port num1, ... , port num5"

- Tmaxではエンジンとクライアント間のデータを圧縮することで、通信を可能にする機能を提供します。クライアントが圧縮機能を使用して通信しようとするときに接続するポート番号を指定します。指定しないと圧縮を実行しません。重要なことは、ポート番号を現在Tmaxが管理しているポートの中から指定することです。

NODEセクションのTMAXPORT項目が定義されている場合はTMAXPORT項目に指定されているポート番号の中から指定し、TMAXPORT項目が定義されていない場合はDOMAINセクションのTPORTNO項目に指定します。このように指定すると、当該ポートに接続したクライアントのみが圧縮機能を使用できます。

- 一般的なネットワーク環境であればネットワーク転送の遅延より圧縮するオーバーヘッドが大きいため、圧縮機能を使用しないことを推奨します。圧縮は、クライアントとサーバー間のネットワーク条件が良くない環境で比較的大容量のデータを送受信するときに必要です。

たとえば、モデムを介して接続するクライアントが数十KB以上のデータの送受信をしようとする場合に設定します。

- 以下の表はクライアントが100MbpsローカルLANと56KBモデムを使って接続した環境で1.5MBのデータを受信する時間を測定したものです。テスト結果から分かるように、モデムで接続したクライアントがテキスト形式の大容量データの送受信をする場合に圧縮機能を使用することが最も効果的です。

- ローカル・ネットワーク・クライアント

区分	圧縮した場合	圧縮しなかった場合
mp3データを受信する場合	10	3
	10	3
	10	4
	10	3
テキスト・データを受信する場合	8	1
	8	0(0.85)
	8	0(0.85)
	8	0(0.85)

• モデム・クライアント

区分	圧縮した場合	圧縮しなかった場合
mp3データを受信する場合	352	350
	309	307
	328	331
	349	351
テキスト・データを受信する場合	80	137
	89	153
	77	154
	77	154

<転送するデータサイズ>

区分	実際サイズ	圧縮したサイズ
mp3	1,459,095バイト	1,442,869バイト
txt	1,445,184バイト	313,057バイト

注

1. 圧縮機能を使うためには、クライアント・プログラムをコンパイルするときにlibz.a(so)と一緒にリンク付ける必要があります。
2. 現在、リアル・サーバーとクライアント、ノード間のデータ転送、ドメイン間のデータ転送には圧縮機能をサポートしません。

● COMPRESSSIZE = numeric

- 圧縮機能を使うとき、指定したサイズ以上のデータだけを圧縮できるようにする項目です。(単位: バイト)
- CompressSizeは必ずCompressPortと一緒に指定します。
- 下記の例のように設定した場合、Tmaxエンジンはポート番号9999で接続したクライアントと圧縮機能を使用して通信します。通信の際、データサイズが10,000バイト以上のメッセージである場合のみ圧縮し、データサイズが10,000バイト以下であるか、またはクライアントがポート番号8888で接続した場合は圧縮機能を使わずに通信します。

```
TMAXPORT = "8888, 9999", CompressPort = "9999" CompressSize = 10000
```

- RESTART = Y | N

- デフォルト値 : Y

- NODEセクションに指定した場合、Tmaxエンジンプロセスのうち、CLHとCAS、CLL、TLMの再起動の可否を決定し、異常終了時に再起動させます。

- MAXRSTART、GPERIOD

- 障害対策に使用するフィールドです。詳細については、「[3.8. 障害対策の環境設定](#)」を参照してください。

- EXTPORT = numeric

- TmaxはTmax環境ファイルに登録されたサーバー機能をTmax以外のプロセスで処理できるように、Externサーバー機能をサポートします。

Externサーバー機能を使用するには、サーバー・プロセスのタイプをEXTSVRに指定し、NODEセクションにEXTPORTおよびEXTCLHPORT項目を設定します。EXTPORT項目にTMMの待ち受け(Listen)ポート番号を設定します。

- EXTCLHPORT = literal

- CLHの待ち受けポート番号を設定します。指定しない場合、システムで自動に割り当てます。

- 以下はExternサーバーの使用例です。

```
*NODE
tmaxh3      ...
            EXTPORT = 9000, EXTCLHPORT = 9010

*SERVER
alinkjmapp  SVGNAME = svg1,
            MIN = 1, MAX = 1,
            CPC = 10,
            SVRTYPE=EXTSVR

*SERVICE
JMAPPING    SVRNAME = alinkjmapp,
            SVCTIME = 30
```

- SMSUPPORT = Y | N

- デフォルト値 : N

- SysMasterのトレース機能を使用するかどうかを設定するオプションです。

設定値	説明
Y	トレース機能をサポートします
N	トレース機能をサポートしません

- SysMasterのトレース機能は、実行中のサービスのGIDを取得し、システム単位で業務を追跡します。全タイプのサーバーに対応するのではなく、TCS、UCSに限ってGIDを発行します。追加で、TCP\_GATEWAY(CUSTOM\_GW)からの要求に対してもGIDが発行されます。

以下は、GIDの構造です。(12バイト)

区分	サイズ	説明
GID0	4バイト	製品内のクライアント別の固有番号(WebtoBの場合、cli id)のdomain id、node id、hth #、slot idなどであり、製品に接続したクライアントを区別するための番号です
GID1	4バイト	上位の3バイトはseq #であり、下位の1バイトは製品の固有IDです
SEQNO	4バイト	上位の2バイトは非同期呼び出し時のbranch #に使用し、下位の2バイトはすべての呼び出し時のseq #に使用します

- SMTBLSIZE = numeric

- 範囲：1024~MAX\_INT

- デフォルト値: 50000

- CLH別のSysMasterトレースの最大保存件数であり、**SMSUPPORT**項目が「Y」である場合のみ設定します。

- CLLBLOCK = Y | N

- デフォルト値：N

- CLLブロックの設定の可否を決定します。

設定値	説明
Y	クライアントがTmaxシステムに接続するときにCLLがブロックされます

- Tmaxサービスが完全に起動していない状態でクライアントの要求があった場合、CLLが起動された状態であれば、クライアントはTmaxシステムに接続が許可され、TPENOREADYエラーが発生します。



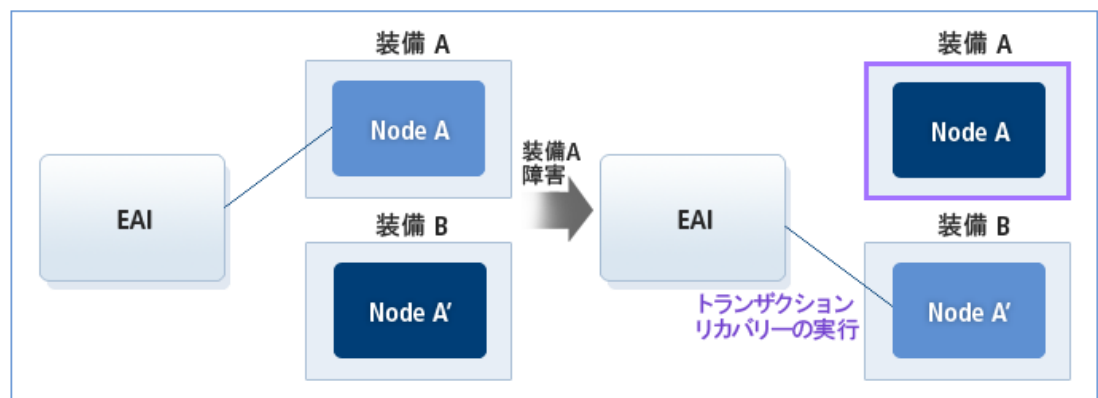
こうした問題を解決するために管理者がクライアントの要求をCLLでブロック処理します。これにより、サービスが完全に準備される前にサービス要求が送信されるのを防止できます。

- サービスがすべて起動された場合、tmadminコマンドを利用してCLLブロック機能を解除できます。
- `CLLUNBLKPORT = "Portno1, Portno2 ..."`
  - CLLブロックの例外ポート番号を設定します。TMAXPORTに設定されたポート番号の中から指定できます。
  - CLLブロックが設定された場合、Tmax環境ファイルにCLLブロックの例外ポートを指定できます。ブロック例外ポートを設定するとCLLブロック機能が設定されていても、当該ポートに接続したクライアントはブロックされません。
- `CLLBLOCKTIME = numeric`
  - デフォルト値：0
  - `CLLBLOCK=Y`の場合、CLLBLOCKTIMEを使用してCLLのブロック時間(秒)を指定することができます。ブロック時間が終わると、CLLのブロックが解除されます。
  - ブロック時間が0以下の値であるかCLLBLOCK=Yの場合、CLLBLOCKTIMEを指定しなかった場合はtmadminを使用してCLLブロック機能を解除する必要があります。
- `CLLCONNLB = RR | LC`
  - デフォルト値：RR
  - CLLがCLHIにクライアント接続を渡す方式を設定します。
  - `CLLCONNLB = LC`に設定すると、リース接続(Lease Connection)方式で、CLLが最も少ない接続を持つCLHIに次の接続を確立します。
  - CLLCONNLBを指定しないか、それ以外の値を指定すると、デフォルト値のラウンドロビン方式で次の接続を確立します。
- `CRYPTPORT= Portnumber list`
  - NODEセクションにCRYPTPORT項目を指定する場合、CRYPTPORTに指定されたポートに接続したクライアントのデータのみ暗号化されます。暗号化機能をより柔軟に活用できるように、セッション別に暗号化を選別して指定できるようにします。
  - 以下は、この項目を設定時に考慮すべき事項です。

- CRYPTPORT項目は必ずTMAXPORT項目の中から1つを指定します。
  - DOMAINセクションのCRYPT項目を「Y」に指定した場合にのみ暗号化できます。
  - DOMAINセクションのCRYPT項目を「Y」に指定した後、NODEセクションのCRYPTPORT項目値を指定しないと、TMAXPORTに設定したすべての値がCRYPTPORTに指定されます。
  - CRYPTPORTに複数ポートを指定できます。
- TRB = string
- マルチノードおよびマルチドメイン環境において、装備の障害などによって特定ノードが一時的に使用できない場合、ペンディング・トランザクションによる可用性の低下を防ぐために、トランザクションの復旧を代行するバックアップ・ノード(TRBノード)を構成します。
  - 下図のように、ノードAに対しTRB Node A'を事前に構成しておきます。

装備Aに障害が生じると、TRB Node A'を起動させてペンディングされたトランザクションを自動復旧させます。

**[図 3.3] TRB**



- TRB項目は、トランザクション・リカバリー・バックアップ・ノード(TRBノード)に指定し、1次(Primary)ノード名を設定します。
- TRBノードは1つのサーバー・グループを保持します。そしてこのサーバー・グループに、使用予定のデータベース情報を指定し、1つのTMSを設定します。ゲートウェイを使用する場合、GATEWAYセクションにもTRB項目を指定します。
- 以下は、この項目を設定時に考慮すべき事項です。
  - TRBノードを起動させる前に障害が生じた1次(Primary)ノードのtlog(TXLOG、GWTXLOG)をTRBノードにコピーします。

- TXLOG、GWTXLOGは現行の装備に依存するバイナリ形式であり、同機種でのみ互換できます。CPUが異なる場合(Little endian / Big Endian)や、バイナリモードが異なる場合(32bit / 64bit)は互換されません。
- 障害が発生したノードがアクセスしたデータベースをTRBノードでもアクセスできる必要があります。
- 障害が発生したノードとTRBノードのIP/PORTが異なるため、これをEAIで受け入れる必要があります。
- TRBノードではトランザクション復旧機能のみ実行し、サービスは実行しません。
- 基本的にTRBノードの起動および終了は手動で管理し、APMソリューションなどを利用してノードの障害の検知やTRBノードの起動・終了を自動化できます。

#### – TRBノードの起動と終了

TRBノードは一般ノードとは異なる方式で起動または終了させるため、-Bオプションを使用します。

```
$tmboot -B <nodename>
```

TRBと関連する環境設定は以下のとおりです。

```
*NODE
#Primary node
NODE1      TMAXDIR = /data/tmax

#TRB node
NODE2      TMAXDIR = /data2/tmax, TRB = NODE1
```

#### ● MAXSESSION = numeric

- 範囲：1 ~ 65535
- デフォルト値：1024
- ノード内のHMSで生成できるセッション数の最大値を設定します。詳細は、「[3.7. HMS環境設定](#)」を参照してください。

#### ● SQKEY=numeric

- 範囲: 32768~262143
- SQリポジトリとして使用される共有メモリー・セグメント(shared memory segment)を指すキー値を定義します。

- SQSIZE=numeric
  - 範囲: 4 ~ 4000000 (単位 : KB)
  - SQリポジトリとして使用される共有メモリー・セグメントのサイズを定義します。
- SQMAX=numeric
  - 範囲: 2~MAX\_INT-1 (単位 : KB)
  - ノードに生成できる最大SQ数であり、偶数で設定します。奇数で設定する場合、1つ小さくなります。
- SQKEYMAX=numeric
  - デフォルト値: 1~MAX\_INT
  - 各SQで保存可能な最大キー数です。
- SQTIMEOUT=numeric
  - デフォルト値: 1~MAX\_INT (単位: 秒)
  - SQデータ保管タイムアウトです。タイムアウトの場合、データが自動削除されます。
- IP = literal
  - サイズ : 255字以内
  - ノードのIPアドレスを指定します。
  - 他のノードとの接続を試みるとき、ホスト名からIPアドレスが把握できない場合に必ず指定します。
  - IPv6アドレスを使用する場合は、SYSTEM\_PROTOCOL設定を「IPV6」に設定します。
  - クライアント接続要求のためのポートを作成するとき、バインドされるインターフェースのIPアドレスとして使用されます。関連設定には、CLL\_BIND\_IPがあります。
- CRYPT\_ALGORITHM = "3DES" | "AES"
  - デフォルト値 : 3DES
  - Tmax v5.0 SP3から暗号化機能を使用する際に暗号化方式を指定できます。

- 3DES方式とAES方式をサポートします。
  - サーバー環境設定ファイルのCRYPT\_ALGORITHMとクライアント側の環境変数のTMAX\_CRYPT\_ALGORITHMが同じ方式に指定されているかを確認します。
  - この設定を適用するためには、DOMAINセクションでCRYPT=Yに設定する必要があります。
- CLIENT\_PROTOCOL = "IPV4" | "IPV6" | "SDP"
    - デフォルト値 : "IPV4"
    - Tmaxシステムでクライアントの接続ポートを作成時に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。この場合、クライアントはIPv4でのみ接続できます。ノードごとに独立した設定が可能です
IPV6	IPv6プロトコルを使用して接続要求を待ちます。クライアントはIPv4またはIPv6環境でTmaxシステムに接続できます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- SYSTEM\_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - デフォルト値 : "IPV4"
  - Tmaxシステムが複数のノードで構成されている場合に、ノード間の通信のためポートを作成するとき使用するプロトコルを決定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です
IPV6	他のノードへの接続要求のためポートを作成するとき、IPv6プロトコルを使用して接続要求を待ちます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- EXTSVR\_PROTOCOL = "IPV4" | "IPV6" | "SDP"
  - デフォルト値 : "IPV4"

- Tmaxシステムから外部サーバーに接続するためポートを作成する際に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます。ノードごとに独立した設定が可能です
IPV6	IPv6プロトコルを使用して接続要求を待ちます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- CLIENT\_IPV6 = Y | N

- デフォルト値 : N

- Tmaxシステムでクライアントの接続のためのポートを作成する際、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	IPv6プロトコルを使用して接続要求を待機します。クライアントはIPv4環境またはIPv6環境でTmaxシステムに接続することができます
N	IPv4プロトコルを使用して接続要求を待機します。この場合、クライアントはIPv4環境でのみ接続が可能です

## 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにCLIENT\_PROTOCOL項目を使用することをお勧めします。

- SYSTEM\_IPV6 = Y | N

- デフォルト値 : N

- Tmaxシステムが複数のノードで構成されており、ノード同士の通信のためにポートを作成する際、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	他のノードからの接続要求のためにポートを作成する際、IPv6プロトコルを使用して接続要求を待機します
N	IPv4プロトコルを使用して接続要求を待機します

---

## 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにSYSTEM\_PROTOCOL項目を使用することをお勧めします。

---

- EXTSVR\_IPV6 = Y | N

- デフォルト値 : N

- Tmaxシステムで外部サーバーの接続のためにポートを作成する際、IPv6プロトコルを使用するかどうかを指定します。

設定値	説明
Y	IPv6プロトコルを使用して接続要求を待機します
N	IPv4プロトコルを使用して接続要求を待機します

---

## 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにEXTSVR\_PROTOCOL項目を使用することをお勧めします。

---

- CLL\_BIND\_IP = Y | N

- デフォルト値 : Y

- クライアント接続のためのポートを作成する際、NODEセクションのIPに設定されたアドレスのインターフェースを使用するかどうかを指定します。

設定値	説明
N	IPを設定しないと、すべてのインターフェースからの接続を許可します

---

- SVCLOG\_FORMAT = literal

- デフォルト値 : N

- svclogの項目をユーザーが調整できるようにします。「\$(CLIENTIP) \$(RESULT)」のような形式で文字列を指定すると、当該項目を追加することができます。

文字列	svcrpt項目
\$(CLIENTIP)	クライアントIP

---

文字列	svcrpt項目
\$(SBUFTYPE)	要求バッファ・タイプ
\$(RBUFTYPE)	リターン・バッファ・タイプ
\$(RESULT)	処理結果
\$(MEM)	メモリー使用量
\$(SCPU)	システムCPU使用量
\$(UCPU)	ユーザーCPU使用量
\$(TID)	tid

- CASTHREAD = 1 ~ 65535
  - TmaxのCASプロセスのワーカー・スレッド数を指定します。Tmaxが提供するセキュリティー設定についての詳細は「[3.9.3. サードパーティー・セキュリティー設定](#)」を参照してください。
  
- CASOPT = literal
  - サイズ : 255字以内
  - CASが起動時にCASプロセスに渡されるコマンド・オプションを定義します。Tmaxが提供するセキュリティー設定についての詳細は「[3.9.3. サードパーティー・セキュリティー設定](#)」を参照してください。
  
- SECURITY\_LIB = literal
  - Tmaxがデフォルトで提供する認証ライブラリー以外の他のプラグイン・ライブラリーを使用する場合、そのライブラリーのパスを設定します。Tmaxが提供するセキュリティー設定についての詳細は「[3.9.3. サードパーティー・セキュリティー設定](#)」を参照してください。
  
- CRYPT\_LIB = literal
  - Tmaxがデフォルトで提供する暗号化ライブラリー以外の他のプラグイン・ライブラリーを使用する場合、そのライブラリーのパスを設定します。Tmaxが提供するセキュリティー設定についての詳細は「[3.9.3. サードパーティー・セキュリティー設定](#)」を参照してください。
  
- TGOPT = literal
  - サイズ : 255字以内
  - TmaxGridが起動時にTmaxGridプロセスに渡されるコマンド・オプションを定義します。

定義されたオプションのうち、「--」の前に指定されたオプションはシステムで使用し、その後に指定されたオプションはユーザーが自由に使用することができます。



– 以下は、ユーザー・オプションについての説明です。

オプション	説明
-eファイル名	TmaxGridの動作中に発生する標準エラー(standard error)をファイルに記録します
-oファイル名	TmaxGridの動作中に発生する標準出力(standard output)をファイルに記録します

## 再定義できる項目

DOMAINセクションに定義されていながら、各ノードに再定義できる項目です。この項目はNODEセクションに定義されない場合、DOMAINセクションで定義した値が全ノードに共通して適用されます。再定義された場合、そのノードに対してはNODEセクションで定義した新規値が適用されます。形式と内容はDOMAINセクションと同じです。

以下は、NODEセクションの表現形式と項目を示した表です。

項目名 = 項目値

項目名	項目値	デフォルト値
SHMKEY	numeric	
MINCLH	numeric	1
MAXCLH	numeric	10
TPORTNO	numeric	8888
RACPORT	numeric	3333
MAXUSER	numeric	
IPCPERM	numeric	0600
MAXSPR	numeric	64
MAXSVR	numeric	64
MAXCPC	numeric	32
LOGOUTSVC	string	
TMMLOGLVL	string	DETAIL
CLHLOGLVL	string	DETAIL
TMSLOGLVL	string	DETAIL
LOGLVL	string	DETAIL
TDL	Y N	N
CLIENT_PROTOCOL	"IPV4"   "IPV6"   "SDP"	"IPV4"

項目名	項目値	デフォルト値
SYSTEM_PROTOCOL	"IPV4"   "IPV6"   "SDP"	"IPV4"
EXTSVR_PROTOCOL	"IPV4"   "IPV6"   "SDP"	"IPV4"
CLIENT_IPV6	Y N	N
SYSTEM_IPV6	Y N	N
EXTSVR_IPV6	Y N	N
CLL_BIND_IP	Y N	Y
MSGSIZEWARN	numeric	1073741824
MSGSIZEMAX	numeric	1073741824

## 使用例

以下は、NODEセクションの使用例です。

- NODEセクションの使用例1 (物理ノード)

```
*NODE
DEFAULT:
    MINCLH = 2, MAXCLH = 3
tmax1 TMAXDIR = "/home/tmax",
      APPDIR = "/home/tmax/appbin",
      PATHDIR = "/home/tmax/path",
      SLOGDIR = "/home/tmax/log/slog",
      TLOGDIR = "/home/tmax/log/tlog",
      ULOGDIR = "/home/tmax/log/ulog",
      TMMOPT = "-o /home/tmax/log/slog/tmmo.log -e /home/tmax/log/slog/tmme.log",

      CLHOPT = "-o /home/tmax/log/slog/clho.log -e /home/tmax/log/slog/clhe.log",

      ENVFILE = "/home/tmax/server/start",
      TMAXPORT = "8850, 9000, 9001, 9002, 9003",
      CompressPort = "9000, 9001",
      CompressSize = 10240,
      RESTART = Y,
      MAXRSTART = -1,
      GPERIOD = 100
```

- NODEセクションの使用例2 (1つのマシンにおける論理ノード)

```
*NODE
DEFAULT:
    HOSTNAME = "tmaxs1",
    TMAXHOME = "/user/QA/tmax",
    NODETYPE = SHM_USER
```

```

tmaxs1  TMAXDIR = "/user/QA/tmax",
        APPDIR  = "/user/QA/tmax/appbin",
        PATHDIR = "/user/QA/tmax/path",
        TLOGDIR = "/user/QA/tmax/log/tlog",
        ULOGDIR = "/user/QA/tmax/log/ulog",
        SLOGDIR = "/user/QA/tmax/log/slog"

NODE1   TMAXDIR = "/user/QA/proj1",
        APPDIR  = "/user/QA/proj1/appbin",
        PATHDIR = "/user/QA/proj1/path",
        TLOGDIR = "/user/QA/proj1/log/tlog",
        ULOGDIR = "/user/QA/proj1/log/ulog",
        SLOGDIR = "/user/QA/proj1/log/slog",
        TPORTNO = 8952, SHMKEY = 76995,
        RACPORT  = 3334

NODE2   TMAXDIR = "/user/QA/proj2",
        APPDIR  = "/user/QA/proj2/appbin",
        PATHDIR = "/user/QA/proj2/path",
        TLOGDIR = "/user/QA/proj2/log/tlog",
        ULOGDIR = "/user/QA/proj2/log/ulog",
        SLOGDIR = "/user/QA/proj2/log/slog",
        TPORTNO = 8993, SHMKEY = 76999,
        RACPORT  = 3335, NODETYPE = RACD

```

実際ホスト名はtmaxs1で、3つの論理ノード(tmaxs1、NODE1、NODE2)が存在し、マルチノードを構成します。TMAXHOMEは3つの論理ノードが共有するTmaxインストール先ディレクトリー(bin、lib、usrinc、tuxinc、topinc、cblinc)であり、TMAXDIRは論理ノードの作業ディレクトリー(config、path、license、log、svct)になります。tmaxs1論理ノードでtmbootを行うと、NODE1のNODETYPEがSHM\_USERであるため、NODE1にはracdを介さずに直接「tmboot/tmdown/tmadmin」を実行します。NODE2のNODETYPEがSHM\_RACDであるため、NODE2にはracdを介してtmbootなどを実行します。

各論理ノードは現在の環境ファイルを読み込み、TMAXDIRを利用して自分のノード番号を把握した後、該当TMMIに接続して「tmboot/tmdown/tmadmin/cfl/gst」を実行するか、目的のファイルの読み取り/書き込み作業を行います。

cflを利用して環境ファイルをコンパイルする場合、現在の環境変数\${TMAXDI}Rディレクトリーのconfig(\${TMAXDIR}/config)を参照し、gstを使用する場合にも「\${TMAXDIR}/svct」を参照してローカル・ノードに該当するサービス・テーブルを生成します。

- NODEセクションの使用例3 (複数マシンにおける論理ノード)

```

*NODE
DEFAULT:
    HOSTNAME = "tmaxs1",
    TMAXHOME = "/user2/QA/tmax32",

```

```

NODETYPE = SHM_USER

tmaxs1  TMAXDIR = "/user2/QA/tmax32",
        APPDIR  = "/user2/QA/tmax32/appbin",
        PATHDIR = "/user2/QA/tmax32/path",
        TLOGDIR = "/user2/QA/tmax32/log/tlog",
        ULOGDIR = "/user2/QA/tmax32/log/ulog",
        SLOGDIR = "/user2/QA/tmax32/log/slog"

NODE1   TMAXDIR = "/user2/QA/proj1",
        APPDIR  = "/user2/QA/proj1/appbin",
        PATHDIR = "/user2/QA/proj1/path",
        TLOGDIR = "/user2/QA/proj1/log/tlog",
        ULOGDIR = "/user2/QA/proj1/log/ulog",
        SLOGDIR = "/user2/QA/proj1/log/slog",
        TPORTNO = 8952, SHMKEY = 76995,
        RACPORT = 4335

DEFAULT:
        HOSTNAME = tmaxc1,
        TMAXHOME = "oracle/QA/tmax",
        NODETYPE = SHM_RACD

tmaxc1  TMAXDIR = "/oracle/QA/tmax",
        APPDIR  = "/oracle/QA/tmax/appbin",
        PATHDIR = "/oracle/QA/tmax/path",
        TLOGDIR = "/oracle/QA/tmax/log/tlog",
        ULOGDIR = "/oracle/QA/tmax/log/ulog",
        SLOGDIR = "/oracle/QA/tmax/log/slog",
        TPORTNO = 8893, SHMKEY = 76980,
        RACPORT = 4335

NODE3   TMAXDIR = "/oracle/QA/proj1",
        APPDIR  = "/oracle/QA/proj1/appbin",
        PATHDIR = "/oracle/QA/proj1/path",
        TLOGDIR = "/oracle/QA/proj1/log/tlog",
        ULOGDIR = "/oracle/QA/proj1/log/ulog",
        SLOGDIR = "/oracle/QA/proj1/log/slog",
        TPORTNO = 9984, SHMKEY = 76999,
        RACPORT = 4336

```

実際ホスト名はtmaxs1とtmaxc1です。4つの論理ノード(tmaxs1ホストのtmaxs1とNODE1、tmaxc1ホストのtmaxc1とNODE3)が存在し、マルチノードを構成します。

tmaxc1、NODE3のNODETYPEがSHM\_RACDであるため、tmaxc1ホストの中でtmaxc1とNODE3論理ノード間にracdを介して「tmboot/tmdown/tmadmin」などを実行します。tmbootを実行するノードがtmaxs1ホストのNODE1であれば、tmaxc1ホスト内のtmaxc1とNODE3ノードのNODETYPEは意味がなくなり、無条件にracdを利用して通信します。

racdはTMAXDIRとRACPORTを利用することで起動しますが、ユーザーが各論理ノードのracdを起動するたびに2つの環境変数を毎回換えることは不便です。racdに[-i]オプションと[-l]オプションを使用してどのノードのracdであるかを簡単に区別できます。NODETYPEがSHM\_RACDの場合、論理ノードごとにracdを1つずつ起動させる必要があるためです。

以下は、TRBノードの使用例です。

- マルチノード使用例

```
*DOMAIN
dom1      SHMKEY =78755,
          RACPORT = 3366, . . .

*NODE
tmaxh4     TMAXDIR = "/data/tmaxqas/tmax",
          APPDIR  = "/data/tmaxqas/tmax/appbin",

tmaxil     TMAXDIR = "/data/tmaxqas/tmax",
          APPDIR  = "/data/tmaxqas/tmax/appbin",

tmaxh4b     TMAXDIR = "/data/QA/tmax",
          APPDIR  = "/data/QA/tmax/appbin",
          HOSTNAME = "tmaxil", SHMKEY = 78630, TPORTNO = 8630,
          TRB = tmaxh4, RACPORT = 3155

tmaxilb     TMAXDIR = "/EMC01/QA/tmax",
          APPDIR  = "/EMC01/QA/tmax/appbin",
          HOSTNAME = tmaxh4, SHMKEY = 78950, TPORTNO = 8520
          TRB = tmaxil,
          RACPORT = 3355

*SVRGROUP
svg1       NODENAME = tmaxil, DBNAME = ORACLE,
          OPENINFO = "...",
          TMSNAME  = tms_oral

svg5       NODENAME = tmaxh4, DBNAME = ORACLE,
          OPENINFO = "...",
          TMSNAME  = tms_oras

#TRB NODE
svgb1     NODENAME = tmaxh4b, DBNAME = ORACLE,
          OPENINFO = "...",
          TMSNAME  = tms_orab1,
          MINTMS = 1, MAXTMS = 1  #only 1 tms needed

svgb2     NODENAME = tmaxilb, DBNAME = ORACLE,
```

```

OPENINFO = "...
TMSNAME  = tms_orab2,
MINTMS   = 1, MAXTMS = 1

*SERVER
svr2301TX SVGNAME = svg1,
          MIN = 5, MAX = 5, MAXRSTART = -1
svr2305TX SVGNAME = svg5,
          MIN = 5, MAX = 5, MAXRSTART = -1

```

- マルチドメイン使用例1 (2つのドメイン)

<マルチドメイン Dom1>

```

*DOMAIN
dom1    SHMKEY =78351,
        DOMAINID = 10, ...

*NODE
tmaxh4   TMAXDIR = "/data1/tmaxqas/tmax",
        APPDIR  = "/data1/tmaxqas/tmax/appbin",

tmaxh4b  TMAXDIR = "/data/QA/tmax",
        APPDIR  = "/data/QA/tmax/appbin",
        TRB     = tmaxh4, HOSTNAME = tmax11, ...

*SVRGROUP
svg5     NODENAME = tmaxh4, DBNAME = ORACLE,
        OPENINFO = "...",
        TMSNAME  = tms_ora5

#TRB NODE
svgb     NODENAME = tmaxh4b, DBNAME = ORACLE,
        OPENINFO = MINTMS=1, MAXTMS=1, # only 1 tms needed
        TMSNAME  = tms_orab1

*SERVER
svr2305TX SVGNAME = svg5, MIN = 5, MAX = 5, MAXRSTART = -1

*SERVICE
SVC2301TX_1  SVRNAME = gw2301X
SVC2305TX_1  SVRNAME = svr2305TX

*GATEWAY
gw2301X     GWTYPE = TMAX, PORTNO = 4789,
        NODENAME = tmaxh4

```

```

        # Domain Bのgw2301TX
        RGWADDR = "192.168.1.13",
        RGWPORTNO = 4789,

        # Domain Bのgw2302TX (TRB)
        BACKUP_RGWADDR = "192.168.1.43",
        BACKUP_RGWPORTNO = 5789,

gw2302X      GWTYPE = TMAX, PORTNO = 5789,
              NODENAME = tmaxh4b,

        # Domain Bのgw2301TX
        RGWADDR = "192.168.1.13",
        RGWPORTNO = 4789,

        # Domain Bのgw2302TX (TRB)
        BACKUP_RGWADDR = "192.168.1.43",
        BACKUP_RGWPORTNO = 5789,

        TRB = gw2301X

```

## <マルチドメイン Dom2>

```

# Nodeno 1
*DOMAIN
domB      SHMKEY = 78651,
          DOMAINID = 20, ...

*NODE
tmaxil    TMAXDIR = "/data/tmaxqas/tmax",
          APPDIR = "/data/tmaxqas/tmax/appbin",

tmaxilb   TMAXDIR = "/EMC01/QA/tmax",
          APPDIR = "/EMC01/QA/tmax/appbin",
          TRB = tmaxil, HOSTNAME = tmaxh4

*SVRGROUP
svg1      NODENAME = tmaxil, DBNAME = ORACLE,
          OPENINFO = "...",
          TMSNAME = tms_oral

# TRB NODE
svgb1     NODENAME = tmaxilb, DBNAME = ORACLE,
          OPENINFO = "...",
          TMSNAME = tms_orab1, MINTMS = 1, MAXTMS = 1

```

```

*SERVER
svr2301TX      SVGNAME = svg1, MIN = 5, MAX = 5, MAXRSTART = -1

*SERVICE
SVC2301TX_1    SVRNAME = svr2301TX
SVC2305TX_1    SVRNAME = gw2301X

*GATEWAY
gw2301X        GWTYPE = TMAX, PORTNO = 4789,
               NODENAME = tmaxi1,

               # Domain Aのgw2301X
               RGWADDR = "192.168.1.43",
               RGWPORTNO = 4789,

               # Domain Aのgw2302X (TRB)
               BACKUP_RGWADDR = "192.168.1.13",
               BACKUP_RGWPORTNO = 5789,

gw2302X        GWTYPE = TMAX, PORTNO = 5789,
               NODENAME = tmaxi1b,

               # Domain Aのgw2301X
               RGWADDR = "192.168.1.43",
               RGWPORTNO = 4789,

               # Domain Bのgw2302X (TRB)
               BACKUP_RGWADDR = "192.168.1.13",
               BACKUP_RGWPORTNO = 5789,
               TRB = gw2301X

```

- マルチドメイン使用例2 (3つのドメイン)

<マルチドメイン Dom1>

```

*DOMAIN
dom1           SHMKEY = 78351,
               RACPORT = 3155,
               DOMAINID = 10,

#-----
*NODE

tmaxh4         TMAXDIR = "/data1/tmaxqas/tmax",
               APPDIR = "/data1/tmaxqas/tmax/appbin",

tmaxh4b1       TMAXDIR = "/data/QA/tmax",

```



```

APPDIR = "/data/QA/tmax/appbin",
TRB = tmaxh4, SHMKEY = 86655, TPORTNO = 8450,
HOSTNAME = tmaxi1

tmaxh4b2    TMAXDIR = "/user1/tmaxqam/tmax",
APPDIR = "/user1/tmaxqam/tmax/appbin",
TRB = tmaxh4, SHMKEY = 86655, TPORTNO = 8450,
HOSTNAME = ibm51

*SVRGROUP
svg1        NODENAME = tmaxh4, DBNAME = ORACLE,
OPENINFO = "...",
TMSNAME = tms_oral

# TRB NODE
svgb1       NODENAME = tmaxh4b1, DBNAME = ORACLE,
OPENINFO = "...",
TMSNAME = tms_orab1, MINTMS = 1, MAXTMS = 1

svgb2       NODENAME = tmaxh4b2, DBNAME = ORACLE,
OPENINFO = "...",
TMSNAME = tms_orab2, MINTMS = 1, MAXTMS = 1

*SERVER
svr2301TX   SVGNAME = svg1, MIN = 5, MAX = 5, MAXRSTART = -1

*SERVICE
SVC2301TX_1 SVRNAME = svr2301TX
SVC2305TX_1 SVRNAME = gw2301X
SVC2309TX_1 SVRNAME = gw2302X

*GATEWAY
# Domain Bに対するゲートウェイ
gw2301X     GWTYPE = TMAX, PORTNO = 4010,
NODENAME = tmaxh4,

#Domain Aのgw2301X
RGWADDR = "192.168.1.13",
RGWPORTNO = 4010,

# Domain Bのgw2301Xb (TRB)
BACKUP_RGWADDR = "192.168.1.31",
BACKUP_RGWPORTNO = 5010,

# Domain Bのgw2302Xb (TRB)
BACKUP_RGWADDR2 = "192.168.1.43",
BACKUP_RGWPORTNO2 = 6010,

```

```

gw2301Xb      GWTYPE = TMAX, PORTNO = 5010,
               NODENAME = tmaxh4b1, #192.168.1.13
               RGWADDR = "192.168.1.13",
               RGWPORTNO = 4010,
               BACKUP_RGWADDR = "192.168.1.31",
               BACKUP_RGWPORTNO = 5010,
               BACKUP_RGWADDR2 = "192.168.1.43",
               BACKUP_RGWPORTNO2 = 6010,
               TRB = gw2301X

gw2301Xb2     GWTYPE = TMAX, PORTNO = 6010,
               RGWADDR = "192.168.1.13",
               RGWPORTNO = 4010,
               BACKUP_RGWADDR = "192.168.1.31",
               BACKUP_RGWPORTNO = 5010,
               BACKUP_RGWADDR2 = "192.168.1.43",
               BACKUP_RGWPORTNO2 = 6010,
               NODENAME = tmaxh4b2, #192.168.1.31
               TRB = gw2301X

# Domain Cに対するゲートウェイ
gw2302X       GWTYPE = TMAX, PORTNO = 4020,

               #Domain Cのgw2302X
               RGWADDR = "192.168.1.31",
               RGWPORTNO = 4020,

               # Domain Cのgw2302Xb (TRB)
               BACKUP_RGWADDR = "192.168.1.43",
               BACKUP_RGWPORTNO = 5020,

               # Domain Cのgw2302Xb2 (TRB)
               BACKUP_RGWADDR2 = "192.168.1.13",
               BACKUP_RGWPORTNO2 = 6020,
               NODENAME = tmaxh4,

gw2302Xb      GWTYPE = TMAX, PORTNO = 5020,
               RGWADDR = "192.168.1.31",
               RGWPORTNO = 4020,
               BACKUP_RGWADDR = "192.168.1.43"の
               BACKUP_RGWPORTNO = 5020,
               BACKUP_RGWADDR2 = "192.168.1.13",
               BACKUP_RGWPORTNO2 = 6020,
               NODENAME = tmaxh4b1, #192.168.1.13
               TRB = gw2301X

gw2302Xb2     GWTYPE = TMAX, PORTNO = 6020,

```

```

RGWADDR = "192.168.1.31",
RGWPORTNO = 4020,
BACKUP_RGWADDR = "192.168.1.43",
BACKUP_RGWPORTNO = 5020,
BACKUP_RGWADDR2 = "192.168.1.13",
BACKUP_RGWPORTNO2 = 6020,
NODENAME = tmaxh4b2, #192.168.1.31
TRB = gw2301X

```

### 3.2.3. SVRGROUPセクション

Tmaxはアプリケーション・サーバー・プロセスをグループ単位でまとめて管理します。サーバのグループ化はそのグループがどのノードに存在するか、どのデータベースを使用するか、またはどういう論理関係にあるかによって行われます。サーバー・グループはトランザクション処理、負荷調整、障害対策、ルーティングが行われる基本単位です。詳細については、関連節で説明します。

SVRGROUPセクションには以下の内容が定義されます。

- 各サーバー・グループが属するノード
- データベース関連情報
- 分散トランザクション関連情報

SVRGROUPセクションの一部項目はNODEセクションで定義した項目でも、ノードによって新しく定義できます。再定義可能な項目は、[3.2.3節「再定義できる項目」](#)の説明を参照してください。

SVRGROUPセクションの基本環境設定の形式は以下のとおりです。

```

*SVRGROUP
[DEFAULT :]
SVRGROUP Name      NODENAME = node-name
                   [APPDIR = path,]
                   [ULOGDIR = path,]
                   [SVGTYPE = TMAX|EXTSVG ,]
                   [COUSIN = group-name,]
                   [BACKUP = group-name,]
                   [LOAD = numeric,]
                   [DBNAME = name-of-database,]
                   [OPENINFO = string,]
                   [CLOSEINFO = string,]
                   [MINTMS = numeric,]
                   [MAXTMS = numeric,]
                   [TMSNAME = name-of-tms,]

```

```

[ENVFILE = path,]
[XAOPTION = xaoption,]
[CPC = numeric,]
[RESTART = (Y)|N,]
[MAXRSTART = numeric,]
[GPERIOD = numeric,]
[TMSLOGLVL = tms-log-level,]
[LOGLVL = server-log-level,]
[TMSRECOVERY = (Y)|N,]
[TMSDEP = tms_name,...,]
[TMSRANGE = (DOMAIN)|NODE,]
[TMSTYPE = (STD) | STD_MT,]
[TMSTHREAD = numeric,]
[TMSOPT = arguments,]
[TMSXATIME = numeric,]
[DUMMY = Y | (N),]
[HMSNAME = string,]
[HMSINDEX = numeric,]
[HMSMAXTHR = numeric,]
[HMSMAXDBTHR = numeric,]
[HMSMAXBULKTHR = numeric,]
[HMSMAXBULKSIZE = numeric,]
[HMSOPT = literal,]
[HMSUBSCFG = string,]
[HMSMSGLIVE = numeric,]
[HMSPORT = numeric,]
[HMSHEARTBEAT = numeric,]
[HMSGQINT = numeric]
[SVCLOG_FORMAT = svclog-format-string,]
[RQSOPT = literal]

```

## 必須項目

- SVRGROUP Name = string
  - サイズ : 63字以内
  - サーバー・グループの論理名です。SVRGROUPセクションで一意の名前で定義する必要があります。
  - SVRGROUPセクションの名前はSERVERセクションのSVGNAME項目で使われます。
- NODENAME = string
  - サイズ : 63字以内

- サーバー・グループが存在するノードを定義します。使われるノード名はNODEセクションで定義したノード名である必要があります。ノード名にハイフン(-)が含まれる場合、必ず二重引用符(")で囲みます。

## 選択項目

- DEFAULT: 項目 = 値, ...
  - 「[3.2.2 NODEセクション](#)」のDEFAULT項目を参照してください。
- SVGTYPE = string
  - デフォルト値 : TMAX
  - サーバー・グループのタイプを指定する項目です。タイプにはTMAX、EXTSVG、MTMAX、STMAXがあり、デフォルト値はTMAXです。MTMAX、STMAX項目についての説明は、『Tmax プログラミングガイド(MultipleRM)を参照してください。
  - EXTSVGはExternサーバーと同じタイプであり、Webアプリケーションサーバー(WAS)と2相コミットをサポートするためにEXTERN TMSを使用するときに指定します。EXTSVGを使用する場合、OPENINFO項目も一緒に指定する必要があります。OPENINFOセクションは事実上大した意味はないですが、二重引用符(" ")で設定します。

<EXTSVGサーバー・グループの使用例>

```
*NODE
tmaxh3          ...
                EXTPORT = 9000, EXTCLHPORT = 9010

*SVRGROUP
Extsvg          NODENAME = "tmaxh4",
                SVGTYPE = "EXTSVG", TMSNAME = TMSWAS
                OPENINFO = " "

*SERVER
alinkjmapp      SVGNAME = Extsvg1,
                MIN = 1, MAX = 1, CPC = 20,
                SVRTYPE = EXTSVR

*SERVICE
JMAPP           SVRNAME = alinkjmapp,
                SVCTIME = 30
```

- COUSIN = literal



- 設定例

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2", LOAD = -1
svg2      NODENAME = tmaxh2, LOAD = -1

*GATEWAY
gw1       NODENAME = tmaxh4, LOAD = -1
gw2       NODENAME = tmaxh2, LOAD = -1
```

- LOAD = -2

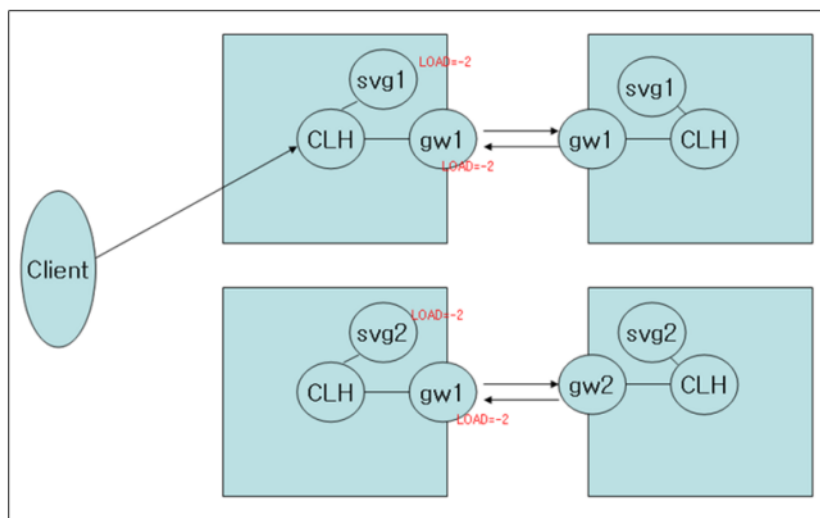
マルチノード環境で、それぞれゲートウェイを設定し、各ノードにクライアントを分離接続させてトランザクションを要求する場合、1次的に接続されたノードで当該サービス进行处理することで業務負荷を軽減できます。

ローカル優先処理機能を使用する場合、COUSINでまとまっているマルチノード環境ではクライアントが接続したノードに属するサーバー・グループが1:1でサービス进行处理します。また、ゲートウェイを介して他ドメインにトランザクションを要求する場合には、クライアントが接続したノードに属しているゲートウェイがそのトランザクション进行处理します。

特定サーバー・グループを指定して呼び出し(tpccallsvg、tpacallsvg)する場合、ローカル・ノードの優先処理機能(LOAD=-2)を設定したとしてもこの機能が適用されず、すべてのサーバー・グループで処理します。またこの場合、そのサーバー・グループが終了すると、他サーバー・グループが代わりに処理を行わず、TPENOREADYエラーが発生します。

トランザクション・ゲートウェイの場合、COUSIN構成のための一般的なサーバー・グループを(dummyサーバー・グループ)設定するときにも必ずXAサーバー・グループに設定します。COUSINでまとめられたすべてのサーバー・グループとゲートウェイのLOAD値をすべて-2に設定する必要があります。

### [図 3.5] LOAD項目が-2の場合の動作原理



- 非トランザクション・ゲートウェイの設定例

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2", LOAD = -2
svg2      NODENAME = tmaxh2, LOAD = -2

*GATEWAY
gw1       NODENAME = tmaxh4, GWTYPE = TMAXNONTX, LOAD = -2
gw2       NODENAME = tmaxh2, GWTYPE = TMAXNONTX, LOAD = -2
```

- トランザクション・ゲートウェイの設定例

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          DBNAME = ORACLE,
          OPENINFO = "Oracle_Xa+Acc=P/scott/tiger+SesTm=60+LogDir=
                    /data1/tmaxqam/tmax/log/tlog/xalog",
          TMSNAME = tms_ora,
          COUSIN = "svg2, gw1, gw2", LOAD = -2
svg2      NODENAME = tmaxh2,
          DBNAME = ORACLE,
          OPENINFO = "Oracle_Xa+Acc=P/scott/tiger+SesTm=60+LogDir=
                    /data1/tmaxqam/tmax/log/tlog/xalog",
          TMSNAME = tms_ora,
          LOAD = -2

*GATEWAY
gw1       NODENAME = tmaxh4, GWTYPE = TMAX, LOAD = -2
gw2       NODENAME = tmaxh2, GWTYPE = TMAX, LOAD = -2
```

- LOAD = -3

COUSINに設定したサーバー・グループ間のバックアップが必要な場合に使用できます。

SVRGROUPセクションのBACKUP項目は、COUSINに設定したサーバー・グループのバックアップではなく、設定されたサーバー・グループに対するバックアップです。たとえば、サーバー・グループA COUSIN="B,C"、BACKUP="D"で構成された場合、A、B、Cが属しているノードがすべて終了してからDにスケジューリングするのではなく、Aが属しているノードが異常な場合、Aスケジューリングの順になったらDにスケジューリングします。A、B、Cに属するサーバーがすべて無効の場合、Dへ移動するための設定値がLOAD=-3です。

- 設定例

```
A COUSIN = "B,C,D", LOAD = -2
B LOAD = -2
```



```
C LOAD = -2
D LOAD = -3
```

上記のように設定されると、通常はA、B、Cにのみスケジューリングを行い、A、B、Cのすべてに問題がある場合、Dにスケジューリングします。すなわち、通常は-3ではなくLOADを持つサーバー・グループだけを使用してスケジューリングし、すべてに問題がある場合には-3を持つサーバー・グループにのみスケジューリングします。そのときスケジューリングは、-2のようなローカル優先の負荷分散に従います。

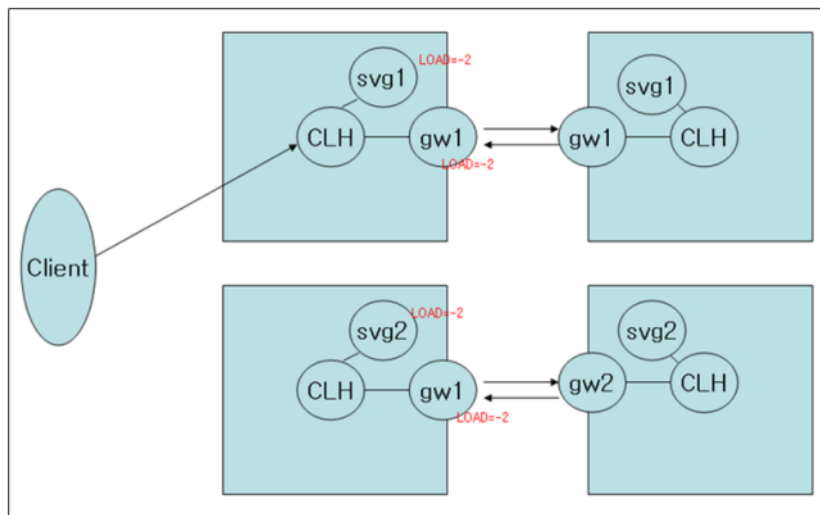
#### – LOAD = 0

動的ルーティング(Dynamic routing)が実行されます。Tmaxエンジンが判断し、負荷が最も少ないグループが処理するようにすることで負荷を分散します。

#### – LOAD > 0

ルールベース・ルーティング(Rule Based Routing)を意味します。分散したいグループにそれぞれ指定した比率で負荷を分散できます。COUSINフィールドで定義されたグループでのみ実行されます。Tmaxサーバー・グループのロードバランシングはLOAD項目とROUTING項目に設定できますが、2つをそれぞれ別々の値で設定して使用することはできません。ROUTINGセクションにサービスが設定されている場合は、当該サービスはルーティングによってのみスケジューリングされ、2次的にLOAD値が影響を与えられません。したがって、COUSIN内でROUTING別にグループをまとめてスケジューリングはできません。

[図 3.6] LOAD項目が正数の場合の動作原理



#### • 設定例

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2", LOAD = 1
```

```
svg2      NODENAME = tmaxh2, LOAD = 1

*GATEWAY
gw1       NODENAME = tmaxh4, LOAD = 1
gw2       NODENAME = tmaxh2, LOAD = 1
```

- DBNAME = string

- サイズ : 63字以内
- TMSに設定される場合、指定します。それ以外の場合は指定しなくても構いません。データベース固有の名前を指定します。

- OPENINFO = literal

- サイズ : 255字以内
- データベースとの接続にTMSが必要な場合に定義します。データベースへの接続を初期化し、各データベースが提供する文法で定義します。注意事項 : @@は暗号化された文章として認識されるため、使用してはなりません。
- OPENINFOセクションの内容を暗号化したい場合は、暗号化しようとする部分を\*\*\*\*\* (アスタリスク5個) で表示します。暗号化が設定された部分に対してはcflを使用すると暗号が入力されます。

- 部分暗号化

```
OPENINFO = "ORACLE_XA+Acc=P/scott/*****+SesTm=100"
```

- 全体暗号化

```
OPENINFO = "*****"
```

- 暗号化された文章

tencryptを使用すると、事前に暗号化した文章を入力することができます。

- CLOSEINFO = literal

- サイズ : 255字以内
- データベースと連動するグループであるTMSが設定される場合、指定します。データベースとの接続を終了するために使用し、各データベースが提供する文法で定義します。ほとんどのデータベースではこの項目を指定する必要がありません。しかし、Informixの場合は必ず指定してください。

- MINTMS = numeric
  - 範囲 : 1~16
  - デフォルト値 : 2
  - データベースと連動するグループであるTMSが設定される場合、データベース起動時に起動されるTMSプロセスの数を指定します。
- MAXTMS = numeric
  - 範囲 : 1~16
  - デフォルト値 : 3
  - データベースと連動するグループであるTMSが設定される場合、最大に起動できるTMSプロセスの数を指定します。
- TMSNAME = string
  - サイズ : 63字以内
  - データベースと連動するグループであるTMSが設定される場合、該当するサーバー・グループのデータベース管理を担当するTMSプロセス名を定義します。
- ENVFILE = literal
  - サイズ : 255字以内
  - このグループに属したサーバーに環境変数で値を送信しようとするとき、または、同じノードに複数個の同種データベースとの連動が必要な場合に指定します。
- XAOPTION = literal
  - ほとんどのデータベースではこの項目を指定する必要がありません。ただし、特定のデータベースではこの項目の設定が必要です。たとえば、DB2のように動的レジスター方式のみをサポートするDBMSの場合、「DYNAMIC」と指定します。
  - SybaseやInformixの旧バージョンのように、64ビットエンジンであるにも関わらず、xaswitch構造体のフィールドがlong型ではなくint型を使用するデータベースの場合は、「XASWITCH32」を指定します。
- CPC = numeric

- 範囲：1 ~ 128
  - SVGTYPEがRQMGRであるときのみ有効です。CLHプロセスを持つチャンネルの数を定義します。詳細については、「[3.6. 信頼性キューの環境設定](#)」を参照してください。
- RESTART = Y | N
    - デフォルト値：Y
    - SVRGROUPセクションに指定した場合、TMSの再起動の可否を決定する項目であり、異常終了時に再起動させます。
- MAXRSTART, GPERIOD
    - 障害対策に使用されるフィールドです。詳細については、「[3.8. 障害対策の環境設定](#)」を参照してください。
- TMSRECOVERY = Y | N
    - デフォルト値：Y
    - トランザクション・リカバリー機能を使用するかどうかを決定します。
    - Tmax v4.0以上ではX/Open DTPに定義されているトランザクション・リカバリー機能をサポートします。リカバリー機能を使用すると、TMSが再起動するときにRMからペンディングされているトランザクションのXIDリストを取得し、該当するトランザクションを復旧します。Tmaxは単一ノード、マルチノード、Tmaxドメイン・ゲートウェイを使用したマルチドメイン環境でのトランザクション・リカバリーをサポートします。
    - グループ全体を起動・終了する場合、リカバリーが実行されるため、TMSグループ単位で復旧できます。tmboot/tmdownコマンドに特定の名前を持つTMS全体を起動・終了させるオプションが追加されました。コマンドについての詳細は『Tmaxリファレンスガイド』を参照してください。
    - この機能を使用するには、RMでペンディング・リストを照会するための権限を与える必要があります。
  - TMSDEP = "tmsname1,..."
    - 同じノードで異なるサーバー・グループがデータベースを共有する場合、TMSの運用中にはリカバリー機能を実行しないように、依存関係を生成することができます。

たとえば、以下の例のように設定し、3つのサーバー・グループが1つのデータベースを共有する場合、tms1がtms2とtms3のリカバリーをすべて実行することでリカバリーが重複して実行されることを事前に防止できます。また、このように依存関係が設定された場合、tms2やtms3の運用中にはリカバリーが実行されなくなり、より安定的な復旧をサポートします。

```

svg1 TMSNAME = tms1,
      TMSDEP = "tms2, tms3"

svg2 TMSNAME = tms2,
      TMSRECOVERY = N

svg3 TMSNAME = tms3,
      TMSRECOVERY = N

```

- TMSRANGE = DOMAIN | NODE

- デフォルト値 : DOMAIN
- 異なるノードにあるサーバー・グループがデータベースを共有する場合、TMSRANGEを「NODE」に設定するとそれぞれ自分のノードのリカバリーを実行するため、ノード間のリカバリー・メッセージ・トラフィックを減らせる上、より安定的なリカバリーを実行できます。

- TMSTYPE = STD | STD\_MT

- デフォルト値 : STD
- TMSTYPEの設定値は以下のとおりです。

設定値	説明
STD	一般的なTMSを使用するためには、TMSTYPEをSTDに設定します。libtms.soライブラリーを使用する必要があります
STD_MT	マルチスレッドTMSを使用するには、TMSTYPEをSTD_MTに設定します。libtmsthr.soライブラリーを使用する必要があります

- TMSTHREAD = numeric

- 1つのTMSプロセス当たりの作業スレッド(Working Thread)数を設定します。

- TMSOPT = "arguments"

- TMSログが保存されるディレクトリーとファイル名を指定できます。使用方法はTMMOPT、CLHOPTと同じです。追加でSERVERセクションのCLOPTでの -Jオプションをサポートします。

- TMSXATIME = numeric

- 単位 : 秒
- TMSXATIMEより長くなるXA処理に対してロギング機能を提供します。

- TMSでxa\_prepare、xa\_commit、xa\_rollbackなどのXA処理時間が異常に長引く場合、データベースの障害が疑われます。そうした障害の原因を分析するため、TMSXATIMEに設定された値よりXA処理が長くなる場合のロギング機能を提供します。

- ログは以下のとおりです。

```
CLH2160 XA関数名(xid) processing was delayed (経過時間) by tms(サーバーグループ番号)
      : XA関数のリターン値
CLH2161 XA関数名(xid) processing may be stopped due to closed tms(サーバーグループ番号)
```

- DUMMY = Y | N

- デフォルト値 : N

- サーバー・グループ名を一時的に「\_\_dummy」に使用する方式を代わりに使用します。

- SVRGROUPセクションでDUMMY=Yに設定する場合、tmbootするときそのグループのTMSが起動されません。

SVRGROUPセクションのDUMMY設定はSERVERセクションに継承されないため、SERVERセクションにも指定する必要があります。

- Domain Gateway Cousin設定などdummyサーバー・グループの設定が必要な場合有用に使用できます。詳細については、[「B.2. ドメインゲートウェイCOUSINの設定方法」](#)を参照してください。

- DUMMYオプションがない場合はtmbootが実行されるとき、以下のようなエラーが発生します。

```
(F) BOOT0014 exec error : /data1/tmaxkjh/tmax/appbin/tms_tbr [BOOT0029]: No such file or directory
```

- HMSNAME = string

- サイズ: 63字以内

- HMSプロセス名を定義します。HMSを使用するには、SVGTYPE=HMSに設定する必要があります。HMS設定についての詳細内容は、『Tmax HMSユーザガイド』を参照してください。

- HMSINDEX = numeric

- 範囲 : 0 ~ 65535

- HMSのメッセージ処理に必要な設定値で、ドメイン内で必ず一意な値で設定します。

- HMSMAXTHR = numeric

- 範囲 : 0 ~ 65535

- HMSメッセージに対するストレージ処理を実行しないスレッドの数を設定します。

- HMSMAXDBTHR = numeric

- 範囲 : 0 ~ 65535

- HMSメッセージに対するストレージ処理を実行するスレッドの数を設定します。

- HMSMAXBULKTHR = numeric

- 範囲 : 1 ~ 65535

- HMSメッセージに対するストレージ処理を一括で実行するスレッドの数を設定します。

- HMSMAXBULKSIZE = numeric

- 範囲 : 2 ~ 65535

- HMSメッセージに対するストレージ一括処理を実行時に、一度に処理できるメッセージの最大数を設定します。

- HMSOPT = literal

- サイズ : 255字以内

- HMSの起動時にHMSプロセスに渡されるコマンド・オプションを定義します。

オプション	説明
-eファイル名	HMSの動作中に発生する標準エラー(standard error)をファイルに記録します
-oファイル名	HMSの動作中に発生する標準出力(standard output)をファイルに記録します

- HMSSUBSCFG = string

- サイズ : 255字以内

- 永続サブスクライバー(Durable Subscriber)を設定した環境ファイルのパスと名前を設定します。環境ファイルに定義された永続サブスクライバーはHMSが起動されると自動で登録されます。

- HMSMSGLIVE = numeric
  - 範囲 : 0 ~ 65535 (単位: 時間)
  - HMSは永続メッセージをストレージに保存します。ストレージにたまったメッセージを周期的に削除する時間を設定します。
- HMSPORT = numeric
  - 多数のHMSをクラスタリングする場合、各HMSは相互通信するために直接チャンネルを生成します。使用するポート番号を設定します。
- HMSHEARTBEAT = numeric
  - 範囲 : 0 ~ 65535 (単位: 秒)
  - 多数のHMSをクラスタリングする場合、HMS間のネットワーク障害を検知するために周期的にハートビートメッセージを送受信します。このメッセージを送信する周期を設定します。
- HMSGQINT = numeric
  - 範囲 : 0 ~ 65535 (単位: ミリ秒)
  - HMSのキューのクラスタリングに必要な状態情報の更新周期を設定します。
  - クラスタリングされたキューを使用するためには必ず設定します。周期が短いほど正確な分配が可能ですが、それだけネットワークの負荷は大きいので適当なサイズの値を設定する必要があります。
- RQSOPT = literal
  - サイズ : 255字以内
  - RQSが起動時にRQSプロセスに渡されるコマンド・オプションを定義します。

オプション	説明
-eファイル名	RQSの動作中に発生する標準エラー(standard error)をファイルに記録します
-oファイル名	RQSの動作中に発生する標準出力(standard output)をファイルに記録します

## 再定義できる項目

- APPDIR = literal



- サイズ : 255字以内
  - サーバグループに属するアプリケーションの実行ファイルがインストールされているディレクトリーの絶対パスです。
  - NODEセクションに登録されたAPPDIRは該当ノードで実行されるすべてのアプリケーションの位置を指定する項目です。APPDIRをサーバグループに再指定すると、グループ別にアプリケーションの実行ファイルの位置を指定できます。
- ULOGDIR = literal
    - サイズ : 255字以内
    - NODEセクションでユーザー・メッセージが記録(ロギング)されるディレクトリーの絶対パスをサーバ・グループ別に再指定するための項目です。ユーザー・メッセージの記録方式はNODEセクションで説明したULOGDIRの内容と同じです。

---

#### 注

指定されたパスは使用中のシステムに存在する必要があります。

---

- TMSLOGLVL = string
  - TMSのログレベルを設定します。
- LOGLVL = string
  - サーバーのログレベルを設定します。
- SVCLOG\_FORMAT = literal
  - svclogの項目をユーザーが調整できるようにします。

## 使用例

以下は、SVRGROUPセクションの使用例です。

```
*SVRGROUP
svg1      NODENAME = tmax,
          APPDIR = "/home/tmax/appbin",
          ULOGDIR = "/home/tmax/appbin/svg1_log"
```

```

svgora      NODENAME = tmax, DBNAME = ORACLE,
            OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm = 60,
            DbgFl = 0x01 + LogDir=/tmp",
            TMSNAME = svg1_tms

svginfo     NODENAME = tmax, DBNAME = INFORMIX,
            OPENINFO = "stores7; USER = {userid};
            PASSWD = {password}",
            TMSNAME = info_tms

```

svgoraでLogDir項目はXAログを保存するパスを指定します。Informixの場合、Tmaxシステムを起動させるアカウントがInformixにログインするアカウントと同じときはOPENINFO="stores7"のようにUSER、PASSWDを省略できます。

### COUSINに設定されたサーバー・グループ設定の検証

COUSINサーバー・グループに指定されたサーバー・グループが以下の形で構成されないように、cfl段階で予めエラーを出力します。

- 他サーバー・グループのバックアップに再指定(複合設定)

```

*SVRGROUP
svg1      NODENAME = "tmaxh4",
          COUSIN = "svg2, svg3"
svg2      NODENAME = "tmaxh4",
          BACKUP = "svg3"
svg3      NODENAME = "tmaxh2"

```

以下のようなコンパイルエラーが発生します。

```
(E) CFL3008 server group svg3 is defined as both COUSIN and BACKUP [CFL0309]
```

- 他サーバー・グループのCOUSINに再指定(多重継承)

```

*SVRGROUP
svg1      NODENAME = "tmaxh4",
          COUSIN = "svg3"
svg2      NODENAME = "tmaxh4",
          COUSIN = "svg3"
svg3      NODENAME = "tmaxh2"

```

以下のようなコンパイルエラーが発生します。

```
(E) CFL3008 server group svg3 is defined as duplicate COUSIN [CFL0310]
```

### 多重バックアップの設定

障害対策の1つとして提供されるバックアップ機能を拡張して多重バックアップ機能をサポートします。

```
svg1 NODENAME = @HOSTNAME@, BACKUP = "svg2,svg3,svg4"  
svg2 NODENAME = @RMTNAME@  
svg3 NODENAME = @RMTNAME2@  
svg4 NODENAME = @RMTNAME3@
```

多重バックアップ機能はより安定的な障害対策を提供するための機能です。バックアップ・サーバーに障害が発生したとき、他のバックアップ・サーバーに切り替えられるようにします。

BACKUP項目には1つ以上のサーバー・グループを指定できます。2つ以上のバックアップ・サーバー・グループを指定する場合、コンマ(,)で区切って表記します。このように設定すると、本来のサーバー・ノードに障害が発生したとき、1番目に指定されたバックアップ・ノードが動作し、そのバックアップ・ノードにも異常が生じると、2番目に指定したバックアップ・ノードが動作する多重の障害対策をサポートします。

バックアップ・ノードの処理中にメイン・ノードが復旧されると再度メイン・ノードが処理するようにするフェイルバック (Fail Back)機能もサポートします。2番目のバックアップ・ノードの動作中に1番目のバックアップ・ノードが復旧されると1番目のバックアップ・ノードが動作し、メイン・ノードが復旧されるとメイン・ノードが動作します。

### 3.2.4. SERVERセクション

DOMAIN、NODE、SVRGROUPセクションはTmaxシステム関連の環境設定です。SERVERセクションには実際に提供するサービスを登録します。作成されたアプリケーションが提供するサービスの種類を登録して、提供されるサービスとサービスを提供するアプリケーション・サーバー・プロセスの情報を提供します。

Tmaxは登録されたサービスのみ処理するため、新しいサーバー・プログラムやサービスが追加された場合には、Tmax環境ファイルのSERVERセクションやSERVICEセクションに追加内容を必ず登録して適用させる必要があります。

SERVERセクションにはTmaxが管理するすべてのアプリケーション・サーバー・プロセスが登録されます。基本的にTmaxシステムが起動するとき、登録されたサーバー・プロセスも一緒に起動され、Tmaxが終了するとき、同じくサーバー・プロセスも一緒に終了されます。

SERVERセクションには以下の内容を定義します。

- 各サーバー・プロセスが属するサーバー・グループ
- サーバー・プロセスが起動するときに必要なコマンドラインのオプション
- サーバー・プロセスの最小数と最大数
- 動的サーバー・プロセスを起動させるためのキュー数
- 会話型モードの如何

- 再起動可能の可否と可能回数(「[3.8. 障害対策の環境設定](#)」を参照)
- マルチスレッドとマルチコンテキスト・サーバー・プロセスの最小スレッドと最大スレッドの数およびスレッドのスタック・サイズ

SERVERセクションの一部項目はSVRGROUPセクションで定義しても、ノードによって新たに定義できます。再定義可能な項目は、[3.2.4節「再定義できる項目」](#)の説明を参照してください。

SERVERセクションの基本形式は以下のとおりです。

```
*SERVER
[DEFAULT : ]
Server Name          SVGNAME = server-group-name
                     [CLOPT = literal,]
                     [MIN = numeric,]
                     [MAX = numeric,]
                     [CONV = Y | (N) | B ,]
                     [ASQCOUNT = numeric,]
                     [MAXQCOUNT = max-queue-count,]
                     [SVRTYPE = (STD) | UCS | CUSTOM_GATEWAY | STD_DYN |
                        UCS_DYN | REALSVR | REALSVR_MT | EXTSVR,]
                     [CPC = channel-number,]
                     [MINTHR = numeric,]
                     [MAXTHR = numeric,]
                     [STACKSIZE = numeric,]
                     [ULOGDIR = user-log-path,]
                     [RESTART = (Y) | N,]
                     [MAXRSTART = numeric,]
                     [GPERIOD = numeric,]
                     [TARGET = string,]
                     [AUTOTRAN = Y | (N) | B ,]
                     [SCHEDULE = (FA)/RR,]
                     [LIFESPAN = (IDLE_DOWN) | IDLE_0 | IDLE_sec,]
                     [LOGLVL = server-log-level,]
                     [DUMMY = Y | (N) ,]
                     [MAX_USE_COUNT = numeric,]
                     [CTX_EREPY = (Y) | N ,]
                     [MULTICLH = (Y) | N ,]
                     [AUS = Y | (N) ,]
                     [TMAPM = (Y) | N ,]
                     [MAC = Y | (N) ,]
                     [ROC = Y | (N) ,]
                     [SVRQTIMEOUT = numeric,]
                     [INBOUND CPC = numeric]
                     [SVCLOG_FORMAT = svclog-format-string]
```

## 必須項目

- Server Name = string
  - サイズ : 63字以内
  - サーバーの実行ファイル名です。一般的にサーバー名は一意である必要があります。つまり、1つのサーバー名はSERVERセクションに1回だけ定義します。ただし、マルチドメインで構成する場合は、1つのドメイン内で一意に指定します。
- SVGNAME = string
  - サイズ : 63字以内
  - サーバーが属するサーバー・グループを定義します。
  - 設定された値は必ずSVRGROUPセクションで定義したサーバー・グループ名である必要があります。サーバーとSVRGROUPセクションを関連付けて、サーバーがどのノードで動作するか、どのリソースマネージャ(データベース)を使用するかを確認でき、そのリソースマネージャを開くときに必要なパラメータを渡すことができます。
  - XAモードではTmaxがすべてのデータベーストランザクションを管理し、2相コミット(two phase commit)を保証します。SVRGROUPセクションはOPENINFOフィールドを含めて定義する必要があります。

XAモードのTmaxユーザーは各サーバー・プロセスのデータベースへの接続に関わる必要はありません。一方、XAモードでない場合は、OPENINFOを含まないSVRGROUPセクションを定義し、ユーザーはデータベース接続を管理する必要があります。

## 選択項目

- DEFAULT: 項目 = 値、...
  - 「[3.2.2 NODEセクション](#)」のDEFAULT項目を参照してください。
- CLOPT = literal
  - サイズ : 255字以内
  - サーバー・プロセスが起動するとき、そのプロセスに渡されるコマンド・オプションを定義します。定義されたオプションのうち、'-'以前に指定されたオプションはシステムで使用し、その後指定したオプションはユーザーが自由に使用できます。

– 以下はユーザー・オプションについての説明です。

オプション	説明
-eファイル名	「サーバー・プロセス名_ファイル名」ファイルが生成され、サーバー・プロセスの動作中に発生する標準エラーをファイルに記録します。標準エラーはfprintf(stderr, format, args)関数を使用します
-oファイル名	「サーバー・プロセス名_ファイル名」ファイルが生成され、サーバー・プロセスの動作中に発生する標準出力をファイルに記録します。標準出力は一般的なprintf()関数や標準エラー出力と同様、fprintf(stdout, format, args)関数を使用します
-u uid	ユーザーファイルに登録されているuidを指定します。  3段階のセキュリティ設定(サービスアクセスの制御)になっている環境でUCSのusermainやゲートウェイなど、Tmaxクライアントから開始されていないところよりACLサービスに要求が入った場合、Tmax環境ファイルのSERVERセクションのCLOPTオプションにuser_idを設定することで、ACLサービスにアクセスできます。たとえば、svr_ucsのusermain内でACLサービス(SVC_ACL)を呼び出す場合、Tmax環境ファイルのSERVERセクションのsvr_ucsのCLOPT項目にCLOPT="-u user_id"を設定できます
-r	サービスのタイムアウトによりサーバが再起動された場合、該当サーバーのリスタート回数の増加を制御するオプションです。  RESTART=Yの場合にのみ意味があり、UNIXプラットフォームでは、tpsvctimeout()でtpreturn()のパラメータがTPEXITの場合に適用されます
-l -L value	[-L]オプションと一緒にvalueに設定可能な値については、表の下の説明を参照してください。  [-l]オプションと関係なく、[-L]valueを設定できますが、[-l]オプションがない場合はsvcclogを残しません
-l	サーバーのサービス実行結果を1つのログで残します。ログは「\$ULOGDIR svclog.mmddyyyy」という名前のファイルに5分間隔で保存されます。Tmaxはログ内容を分析するためのsvcrptを使用します
-B	Tmax 4.0 SP3 Fix2からは、マルチCLH環境で1つのサーバー・プロセスに要求が同時にスケジューリングされる場合、CLHキュータイムアウト(Queue Timeout)が適用されず、処理が遅延する問題を改善しました。ただし、バッチ業務の場合は例外が必要で、SERVERセクションのCLOPT項目に[-B]オプションを適用すると、そのサーバー・プロセスにスケジューリングされた場合は例外的にキュータイムアウトを無視して実行されます
-X	Oracleですでにロールバックされたトランザクションに対して問合せが実行される場合、最初はORA-24761エラーを発生させますが、ユーザーがエラーを無視して継続して次の問合せを実行すると、ローカル・トランザクションとして処理され、整合性問題が発生することがあります。この場合はユーザーコードの問題

オプション	説明
	<p>ですが、これを防ぐために[-X]オプションを使用するとxa_end()の結果がXA_OKでない場合、サーバー・プロセスを再開始して整合性問題を解消できます。</p> <p>XAサーバーの場合、tpreturn()内でxa_end()が失敗処理されると基本的にXAチャンネルをリセットします。当該オプションを使用する場合、xa_end()が失敗処理されるとFatalエラーメッセージ(サービスコード : CSC5608)を出力した後サーバー・プロセスを終了します</p>
-c	サーバーが実行されている間に日付が変更された場合、サーバー動作のuserlogによる標準出力ファイルが、削除コマンドを実行しても削除されない動作を修正するためのオプションです。日付が変更されるとロギングファイルが閉じられなかった点を、[-c]オプションで日付変更をチェックして正常に削除されるようにします
-q	RDPMTサーバー(SVRTYPE=REALSVR_MT)の場合に使用するオプションであり、tpflush()を実行する場合、WRITEキューに送信されないデータがあってもTPSENDTOCLIキューでWRITEキューにデータを無条件に移動できるようにします。このオプションはRDPMTでのみ適用され、メモリーが増加すると制限がなくなるのでお勧め事項ではありません
-m	<p>ユーザー・サーバー・プログラムのメモリーが[-m]オプションに設定されたサイズよりも多く使われると、サーバー・プログラムを終了して再起動します。(単位: バイト)</p> <p>このオプションによりサーバーが再起動された場合、TMMIは「sever closed due to MAX_USE_COUNT or MEM_USAGE(CLOPT:-m)」を出力します</p>
-x	TARGETオプションと一緒に使用され、「-x既存のサービス名:置換するサービス名」のように設定し、置換されたサービス名でサービスを要求することができます。複数のサービスを置換する場合は、区切り子のコンマ(,)を使用します。宣言されていないサービス名を設定するか、あるいは重複して設定すると、サーバーの起動時にエラーが発生するので注意が必要です
-J 秒	<p>時間を指定した場合、xa_openが初めて失敗してから指定した秒数が過ぎた後、再起動されます。</p> <p>MAXRSTARTの数には含まれません</p>
-i	<p>UCSタイプのサーバーのusermain()関数の中で、tp_sleep()、tp_usleep()が呼び出される場合、[-i]オプションに応じて動作が異なります。基本的には、このオプションを指定するとTMMイベントを検知しますが、指定しない場合は検知しません。</p> <p>usermain()関数では、サービス要求やtmdownなどのイベント処理のためにtpschedule()を使用します。</p>

オプション	説明
	<p>tp_sleep()関数はCLHイベントが発生するまで待機する関数ですが、usermain()内でこの関数を使用すると、待機によりtpschedule()呼び出しが遅延される場合があります。ただし、tp_sleep()で待機する間、tmdown要求などが受信されると、TMMイベントを検知できず継続して待機することになりますが、明示的に[-i]オプションを指定した場合のみ、TMMイベントを検知するようにし、tpschedule()を利用してTMMイベントが処理できるようにします。</p> <p>APロジックでtp_sleep()の戻り値が2以上のとき、tpschedule()を呼び出さないと、以降、tp_sleep()は継続してTMMイベントが発生するので、必ずtpschedule()を呼び出す必要があります</p>

- 以下は、[-x]オプションを設定した例です。

```
*SERVER
svr2  SVGNAME = svg1, MIN = 1
svr2_1  SVGNAME = svg1, MIN = 1, TARGET = "svr2",
        CLOPT="-x  TOUPPER:TOUPPER1,TOLOWER:TOLOWER1"

*SERVICE
TOUPPER  SVRNAME = svr2
TOLOWER  SVRNAME = svr2
TOUPPER1 SVRNAME = svr2_1
TOLOWER1 SVRNAME = svr2_1
```

- 以下は、[-e]、[-o]オプションを設定した例です。

```
CLOPT = "-e err1 -o out1 -- abc"
```

オプション	説明
-e err1	サーバー・プロセス(tpcals)の動作中に発生した標準エラーをULOGDIRディレクトリーのtpcals_err1ファイルに保存します。
-o out1	サーバー・プロセス(tpcals)の動作中に発生した標準出力をULOGDIRディレクトリーのtpcals_out1ファイルに保存します。

- 以下は、CLOPT項目に-Lオプションと一緒にvalueに設定可能な値に関する説明と設定例です。

```
*SERVER
svr1      CLOPT = "-l -L SL_DFLT", ULOGDIR = "/EMC01/tmax/log/ulog"
svr2      CLOPT = "-l -L SL_NODE", ULOGDIR = "/EMC01/tmax/log/nlog"
svr3      CLOPT = "-l -L SL_SVG", ULOGDIR = "/EMC01/tmax/log/glog"
svr4      CLOPT = "-l -L SL_SVR", ULOGDIR = "/EMC01/tmax/log/slog"
```



設定値	説明
SL_DFLT	従来と同じです
SL_NODE	NODEセクションのULOGDIR項目にsvcclogが生成されます
SL_SVG	SVRGROUPセクションのULOGDIR項目にsvcclogが生成されます。ULOGDIR設定がなければNODEセクションのULOGDIR項目に生成されます
SL_SVR	SERVERセクションのULOGDIR項目にsvcclogが生成されます。ULOGDIR設定がなければNODEセクションのULOGDIR項目に生成されます

- [-r]オプションを設定した場合、SLOGに以下のようなログが記録されます。サーバー再起動の回数には影響を与えません。

```
(W) SVR3032 service timeout error : SVC25 [SVR0403]
(I) SVR3022 SVR (svr25) is down due to tpreturn(TPEXIT) at svc timeout
handler. [SVR0305]
(I) TMM0211 General Infomation : server closed due to TIMEOUT : SVR, pid =
15040 [TMM0161]
(I) TMM3004 SVR (svr25) is restarted the 5th time [TMM0149]
```

#### – MACRO Commands1

以下のマクロ(MACRO)を指定した場合、ファイル名はサーバーの起動時に決まります。

MACRO	説明
\$(SVR)	サーバー名です
\$(SVRI)	サーバー・インデックスです
\$(SPRI)	サーバー・プロセスのインデックスです
\$(SPRMIN)	SERVERセクションのMIN項目です
\$(SPRMAX)	SERVERセクションのMAX項目です
\$(SPRN)	サーバー・プロセスの順次番号です ( 0 - \$(SPRMAX))
\$(DATE)	MMDDYYYYフォーマットの日付です
\$(TIME)	HHMMSSフォーマットの時間です
\$(PID)	プロセスIDです

#### – MACRO Commands2

以下のマクロ(MACRO)を指定した場合、TCSサーバーは、tpsvrinit()、tpsvrdone()サービス関数が呼び出される直前に、UCSサーバーは、tpsvrinit()、tpsvrdone()関数が呼び出される直前とtpschedule()関数が返す直前に時間をチェックしてそのファイル名を決定します。

MACRO	説明
\$(CDATE)	MMDDYYYY形式の日付です
\$(CTIME)	HHMMSS形式の時間です
\$(CYEAR)	YYYY形式の年です
\$(CMONTH)	MM形式の月です (01 ~ 12)
\$(CMONTHS)	「Jan, Feb, ... Dec」形式の月です
\$(CDAY)	DD形式の日付です (01 ~ 31)
\$(CWDAY)	D形式の日付です (1 ~ 7)
\$(CWDAYS)	「Mon, Tue, ... Sun」形式の日付です
\$(CYDAY)	DDD形式の日付です (001 ~ 366)
\$(CHOUR)	HH形式の現在時間です (00 ~ 23)
\$(CMINUTE)	MM形式の現在分です (00 ~ 59)
\$(CSECOND)	SS形式の秒です (00 ~ 59)

- 上記で説明したマクロは多様なタイプのログファイル(例: 日付別)の作成に使用します。

以下は。マクロの設定例です。

```
CLOPT="-o $(SVR).$(DATE).out -e $(SVR).$(DATE).err"
```

'[server name].03312001.out, [server name].03312001.err'のULOGDIRフィールドに定義されたディレクトリーにログファイルを作成します。標準エラーと標準出力のファイル名を他の名前に変更したい場合には「-e /usr/tmax/log/ulog/test.log」の形で絶対パスを指定します。「--」以後の「abc」オプションはユーザー が使用できるようにサーバー・プログラムのtpsvrinit()関数にargv[1]で渡されます。

- MIN = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 1
- 基本的に起動されるサーバー・プロセスの数を決定します。
- 1クライアント当たりサーバー・プロセスが1つずつ起動される一般的なクライアントやサーバー・モデルとは違い、Tmaxシステムではサーバー・プロセスの数を調節し、一定に維持します。CLHプロセスはクライアントの要求があるたびにサーバーの状態を確認し、アイドル(idle)状態のサーバー・プロセスに作業を処理させることでシステムにとって最大の性能を発揮できるようにします。
- MIN項目にはサーバー・プロセスの数を指定します。運用経験により適切な個数を調整する必要があります。多量のサービス要求があるサービスに対して小数のサーバー・プロセスを動作させると、動作

中のサーバー・プロセスだけでは直ちにサービスを処理することが困難であるため、待機時間が増加するなどシステム性能が低下します。

逆に必要以上にサーバー・プロセスを起動した場合はわずかな数のプロセスだけがサービスを処理し、残りのプロセスはアイドル状態になるので、システムリソースの浪費と共に性能低下を招いてしまいます。従って、システム性能を効率的に活用するために、MIN項目の値は慎重に設定する必要があります。

- MAX = numeric

- 範囲 : 1 ~ MAX\_INT

- デフォルト値 : 1

- MIN項目とともにサーバー・プロセスの数を決定する項目です。

MIN項目は基本的に起動されるサーバー・プロセスの数であり、MAX項目はMINの値を含めて追加起動できるプロセスの最大数です。MAX項目は既に動作中の数を含めてさらに起動できるサーバー・プロセスの最大数を意味します。

サーバー・プロセスは基本的にMINの個数の分だけ起動され、突然クライアント要求が集中するなど負荷が多くなると、MAX数までサーバー・プロセスが起動されます。MAX項目もMIN項目と同様にシステムの性能に影響を与えるため、慎重に決定する必要があります。フォークタイプのサーバー・プロセスが使用された場合、MAX値を0に設定します。

- CONV = Y | N | B

- デフォルト値 : N

- 会話型サーバーの如何を指定します。

- Tmaxでは同期型、非同期型、会話型の3つの通信型を提供します。会話型通信を提供するサービスは該当サーバーを必ず会話型サーバーに登録します。会話型サーバーでない場合、同期型と非同期型の通信だけ提供します。CONV項目を「Y」に定義することで会話型サーバーに設定されます。

- CONV=Bに設定すると、該当するサーバー・プロセスはスケジューリング・プロセスになります。B=BOTHの省略形であり、会話型(Y)と非会話型(N)の両方とも対応するという意味です。また、ドメイン・ゲートウェイを介したドメイン同士の通信にも使用できます。

- ASQCOUNT = numeric

- 範囲 : 1 ~ MAX\_INT

- デフォルト値 : 0

- 自動にサーバー・プロセスが追加、起動される条件として、キューにたまった数を定義します。

サービス要求件数がキューにASQCOUNT数以上たまると、MAX数の限度内でサーバー・プロセスが自動で起動されます。指定されていない場合、キューにたまっている数に係わらずサーバー・プロセスは自動起動しません。

- ASQCOUNTで増加したサーバー・プロセスはサービス処理量を検査し、一定の時間間隔で使わないプロセスを終了してリソース使用量を軽減します。この項目はサーバー・プロセスのタイプがPODの場合には意味を持たないため、使用しません。

- MAXQCOUNT = numeric

- 範囲：1 ~ MAX\_INT
- デフォルト値：-1
- 値を設定しないとデフォルト値が設定され、MAXQCOUNTを使用しません。
- トラフィック殺到でクライアント要求がキューにたまり、正常なサービス実行ができない場合、継続する業務要求を無視するために設定する値です。
- キューに設定値以上の業務が要求されると、該当業務はキューに蓄積されず、クライアントに即時にエラー (tperrno = 26)を返します。この機能を使うと、銀行や官公庁など、特定時間帯に集中して多量の業務が要求されるところで負荷を解消できます。

- SVRQTIMEOUT = numeric

- 範囲：-1, 0 ~ MAX\_INT
- デフォルト値：-1
- メッセージがキューで指定した時間以上待機状態にある場合に、メッセージをキューから削除する機能です。サーバーごとに設定することができます。ホストやネットワーク問題などのため、メッセージがキューに溜まり続ける現象が発生する場合、メッセージをキューに無限に溜めずに、クライアントに現在の状況を知らせて、適切に対応するように誘導するための設定値です。

- 設定値による動作

設定値	説明
-1	NODEセクションのCLHQTIMEOUT設定と同一に適用します
0	メッセージをキューから削除しません
1 ~ MAX_INT	指定した時間以上待機状態にあるメッセージをキューから削除します

- SVRTYPE = STD | UCS | CUSTOM\_GATEWAY | STD\_DYN | UCS\_DYN | REALSVR | REALSVR\_MT | EXTSVR | STD\_MT

– デフォルト値 : STD

– サービス・プロセスのタイプを指定する項目です。以下は、各設定値についての説明です。

設定値	説明
UCS	一般的なサービスプロセスは、サービス要求が発生したとき、関連サービスプロセスが実行されて結果を転送するタイプであるに対し、UCSタイプのサーバー・プロセスは基本的にプロセスの内部にMain()部分が存在し、サービス要求が発生していない場合にも一方的に各クライアントに適したデータを送信する場合に有効です。株式相場の予測もUCSタイプのサーバー・プロセスの一例です
REALSVR	少量のデータを多数のクライアントに頻繁に送る必要があるとき(1秒当たり10回以上)、プロセス占有率や処理速度の面でUCSに比べて卓越な性能を見せます。1ノードには1つのRDPタイプのサーバーだけ使用できます
REALSVR_MT	マルチスレディングをサポートするREALSVRです
CUSTOM_GATEWAY	HostlinkやTN3270ゲートウェイなど、外部連動用ゲートウェイのための特殊な形式のサーバーです
EXTSVR	Tmax環境ファイルに登録されたサーバーの機能をTmax以外のプロセスで処理するための機能です。環境ファイルの詳細は <a href="#">「3.2.2 NODEセクション」のEXTPORT、EXTCLHPORT</a> を参照します
STD_DYN, UCS_DYN	<p>CFL当時に設定したMAX値を増やせるサーバーです。SERVERセクションに設定したMAX値を動的に増やせるように当該サーバータイプを設定します。STD_DYNはTCSIに設定し、UCS_DYNはUCSに設定します。該当タイプに設定したサーバーだけtmadminのsetコマンドにより動的にMAX値を変更できます。</p> <p>SERVERセクションのSVRTYPEにSTD_DYNまたはUCS_DYNに設定します。</p> <p>spriの変化は、MAX値の調整後、tmadminでsiコマンドを使って見るとspriが連続しません。すでに一度他のサーバーで使用したspriがMAX値の調整により他のサーバーで使用できます。</p> <p>&lt;注意&gt;</p> <p>FD_SET = 16384バージョンではサポートしません。MAX値を減少させる場合、MIN値より小さく減らせません。MAX値を減少させる場合、減少する値に該当するspr index以後のsprはダウン状態である必要があります。</p> <p>(例: 初期に設定されたspriが8193、8194、8195、8192の順序である場合、MAXを2に減らすために8195、8192はダウン状態である必要があります)</p>

設定値	説明
STD_MT	<p>STD_MTは、STDタイプのサーバーでマルチスレッドとマルチコンテキストの機能が追加されたサーバー・プロセス・タイプです。</p> <p>サーバーは内部的にスレッド・プールを構成し、それぞれのスレッドでサービスを実行します。したがって、1つのサーバー・プロセスで複数のサービス要求を同時に処理することができます。また、サーバー・プロセスはライブラリーが提供するスレッド・プール以外に、アプリケーション・レベルでユーザーがスレッドを追加作成することが可能であり、作成されたスレッドでサービス・スレッドのコンテキストを共有することができます。</p> <p>STD_MTタイプに設定する場合、必ずMINTHR、MAXTHR、STACKSIZE、CPCを一緒に設定します</p>

- CPC = numeric

- 範囲：1 ~ 128

- デフォルト値：1

- UCSサーバー・プロセスとCLHプロセス間の並列通信チャンネル数を決定する項目です。サーバー・プロセスの処理量が多量であり、CLHプロセスと1つのチャンネルでは処理速度が低下するとき、マルチチャンネルを使って並列通信で処理することで、処理速度を向上させることができます。

- CPCはUCS、UCS\_DYN、CUSTOM\_GATEWAY、REALSVR、REALSVR\_MT、STD\_MTのSVRTYPEでのみ有効です。

- サーバー・タイプがSTD\_MTの場合は、CPCの値が必ず設定されている必要があります。

CPC設定値はMAXCLH項目およびMAXTHR項目と関連があります。CPC値を指定する際の注意点は、MAXCLH x CPC値が最低MAXTHRと同じであるか、それ以上である必要があるということです。

たとえば、MAXCLHが1で、MAXTHRが8である場合、CPCは最低8以上である必要があり、MAXCLHが2でMAXTHRが8の場合、CPCは最低4以上である必要があります。これは1つのサーバー・プロセスがサービス要求を同時処理できる最大個数がCPC x MAXCLH値で決定され、MAXTHR項目の値をこの値以上に設定することは不要なためです。CPCを指定しない場合、MAXTHR/MAXCLHの値が適用されます。

- INBOUND CPC = numeric

- 範囲：1 ~ 128

- デフォルト値：CPCが設定されていない場合は1、CPCが設定されている場合はCPCと同じ値

- UCSサーバー・プロセスとCLHプロセス間の並列通信チャンネル数のうち、CLHプロセスからUCSサーバー・プロセスへの通信チャンネル数を決定する項目です。
  - INBOUND CPCはUCS、UCS\_DYNでのみ有効です。
  - INBOUND CPCを別途設定しない場合、CLH → UCSチャンネル数とUCS → CLHチャンネル数が同一になります。性能に直接及ぼす影響が比較的に少ないINBOUND CPC数を設定することで、リソースの無駄を削減できます。
- MINTHR = numeric
    - 範囲 : 0 ~ MAXTHR
    - デフォルト値 : 0
    - STD\_MTタイプのサーバー・プロセスで最初起動時に生成されるサービス・スレッドの最小数を設定します。STD\_MTタイプのサーバーであれば必ず設定します。
    - 設定値が0の場合、CLHからサービス要求が受信されたら、スレッドが作成された後に処理します。
- MAXTHR = numeric
    - 範囲 : 1 ~ 1000
    - デフォルト値 : 1
    - STD\_MTタイプのサーバー・プロセスにおいて、スレッド・プールで管理するサービス・スレッドの最大数を設定します。STD\_MTタイプのサーバーであれば必ず設定します。
    - サービスが要求されるたびにスレッドを無制限で追加して作成すると、性能低下を招くことになるため、スレッドの作成は最大数までに制限する必要があります。システムの性能に影響を与えるため、適切な値で設定する必要があります。
    - スレッドは、サービス・スレッドの最大数にメイン・スレッドが加わるため、MAXTHR+1のスレッドが動作することになります。メイン・スレッドはサービス要求を直接実行せずに、サービス・スレッドとサービス要求を管理します。設定値が1の場合、サービス・スレッドが1つのみ作成されるため、一般的なサーバー・ライブラリーを使用することをお勧めします。
    - MAXTHR設定値はCPC項目およびMAXCLH項目と関連があります。詳細情報は、「CPC」項目を参照してください。
- STACKSIZE = numeric

- 範囲 : 0 ~ MAX\_INT
  - デフォルト値 : 0 (単位: KB)
  - STD\_MTタイプのサーバー・プロセスでサービス・スレッドのスタック・サイズを設定します。設定しない場合、OSで基本的に適用されるサイズで設定されます。
- RESTART = Y | N
    - デフォルト値 : Y
    - サーバー・プロセスの再起動の可否を決定する項目であり、異常終了時に再起動させます。
- MAXRSTART、GPERIOD
    - 障害対策に使われるフィールドです。詳細については、「[3.8. 障害対策の環境設定](#)」を参照してください。
- TARGET = literal
    - サーバー名はサーバー実行ファイル名であるのに対し、TARGET項目はサーバー・プロセス名を再定義する項目です。ここでいうサーバー・プロセスとはAPPPDIRに存在するサーバー実行ファイルを実行させたプロセスを意味します。
    - 以下のようにTARGET項目を設定した場合、tmadminは実際のサーバー名を確認し、psを使ってサーバー・プロセス名を確認できます。

(例: SVGNAME = svrgrp, MIN = 1, MAX = 1, TARGET = "svr2")

```
/user1/starbj/tmax/sample/server> tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---
$$1 tmaxs1 (tmadm): st -p
CLH 0:
-----
svr_name  svgrname  spr_no  status  count  avg  svc
-----
SVR        svrgrp      32      RDY      0    0 .00  -1

/user2/starbj/tmax/sample/server> ps
starbj 28702 1 0 11:13:03 pts/8 0:00 cl1 -b 28700
starbj 28704 1 0 11:13:03 pts/8 0:00 svr2
        -b 28700 -S SVR -s SVR -d -1
starbj 28701 1 0 11:13:03 pts/8 0:00 tmm -b 28700
starbj 28703 1 0 11:13:03 pts/8 0:00 clh -b 28700
```



- 以下は、TARGETの使用環境ファイルの例です。

```
*SERVER
original      SVGNAME = svg0,
copied1       SVGNAME = svg0,
               TARGET  = "original",
               CLOPT   = "-e $(SVR).err -o $(SVR).out -x ORIGINAL:COPIED1"
copied2       SVGNAME = svg0,
               TARGET  = "original",
               CLOPT   = "-e $(SVR).err -o $(SVR).out -x ORIGINAL:COPIED2"

*SERVICE
ORIGINAL      SVRNAME = original
COPIED1       SVRNAME = copied1
COPIED2       SVRNAME = copied2
```

copied1、copied2サーバーはoriginalという名前でサーバー・プロセスが起動します。一般的にtmaxでサーバー名とサーバー・プロセス名は同じですが、TARGETを使用する場合、サーバー名とサーバー・プロセス名は同じではありません。Tmaxが起動するとき、originalというプロセスが3つ起動されます。

また、サービス要求がCOPIED1、COPIED2に入ると、CLOPTセクションの[-x]によってoriginalに代替されます。クライアントでCOPIED1、COPIED2にそれぞれサービス要求ができます。tmadminでcountを確認する場合もサーバー、サービスが増加します。

TARGET項目の設定は同じプログラムを使って行いますが、tpcallする主体によってCOUNTなどを別途確認したい場合に有用に使えます。

---

## 注

本来のサーバーを修正するとき、tmddown-Sコマンドを使用すると、ソース・サーバーだけがダウンするので、コピーされたサーバーとバージョンが合わないことがあります。したがって、tmddownを実行するときは、targetを使用したサーバーをすべてダウンさせることをお勧めします。

---

### ● AUTOTRAN = Y | N | B

- デフォルト値 : N

- mksvrコマンドを使ってサービスを動的に追加した場合、動的に追加されたサービスのAUTOTRANはデフォルト値の「N」で設定されます。

- SERVERセクションのAUTOTRAN項目は動的に追加されたサービスのトランザクションを自動で起動する項目であり、サーバーのAUTOTRANの可否によって動的に追加されるサービスの設定値に影響を与えます。AUTOTRAN項目は[SERVICEセクションのAUTOTRAN項目](#)の説明で詳細に説明します。

- SERVERセクションとSERVICEセクションの両方にAUTOTRANが設定されていれば、SERVICEセクションのAUTOTRANに設定した値が適用されます。

- Bは旧バージョン(~3.8.8)のAUTOTRANの互換設定であり、Yと同一ですが、当該トランザクションはグローバルではなくローカル・トランザクションとして動作します。

- 旧バージョンの動作方式：3.8.9 以前バージョン

XAサーバーがtx\_beginなしで呼び出される場合、自動でトランザクションが開始されます(デフォルト値: AUTOTRAN=Y)。このときのトランザクションは、ローカルトランザクションのように実行されます。

外部呼び出しの場合は、トランザクションが一緒にまとめられなく、結果的に現在のTPNOTRANで呼び出すのと同様に動作します。

- 現バージョンの動作方式：3.8.9 バージョン (以降バージョンを含む)

AUTOTRAN=Yの場合、グローバルトランザクションが自動開始され、外部からの呼び出しがある場合、自動でトランザクションでまとめられます。デフォルト値が「N」に変更されたため、アップグレードの際に当該値を明示的に設定する必要があります。

- SCHEDULE = FA | RR

- デフォルト値：FA

- SCHEDULE項目はMIN値が2以上に指定された場合、サーバー・プロセス間の処理順序を指定するための項目です。

- 以下はSCHEDULE項目の設定値です。

設定値	説明
FA(First Available)	番号が前のプロセスに優先的に作業を割り当てます
RR(Round-Robin)	すべてのプロセスがバランスよく作業を処理します

- LIFESPAN = IDLE\_DOWN | IDLE\_0 | IDLE\_sec

- デフォルト値：IDLE\_DOWN

- MIN個以上追加起動されたサーバー・プロセスの終了時点を決定する項目で、以下の場合に適用されます。

- PODにより起動されたサーバー・プロセス
- ASQCOUNTにより追加起動されたサーバー・プロセス
- tmboot -sコマンドにより手動で追加起動されたサーバー・プロセス

- PODにより起動されたサーバー・プロセス (MIN=0の場合)
  - MIN=0、MAX=1以上、LIFESPAN=IDLE\_DOWNの場合
 

最初にTmaxを起動時にMIN=0なので、サーバー・プロセスの数は0になります。クライアントから要求がある場合、サーバー・プロセスの数が1になり、ユーザーがサーバー・プロセスを終了するまで終了されません。
  - MIN=0、MAX=1以上、LIFESPAN=IDLE\_0の場合
 

クライアントから要求があれば、サーバー・プロセスの数が1になり、サービスの実行後、次の要求がない場合はすぐ終了されます。
  - MIN=0、MAX=1以上、LIFESPAN=IDLE\_secの場合
 

クライアントから要求があれば、サーバー・プロセスの数が1になり、サービスの実行後、secに設定された時間の間要求がなければ終了されます。
- ASQCOUNTにより追加起動されたサーバー・プロセス
  - MIN=1、MAX=2以上、ASQCOUNT=2, LIFESPAN=IDLE\_DOWNの場合
 

最初にTmaxを起動時にMIN=1なので、サーバー・プロセスの数は1になります。クライアントからの要求が2つ以上キューにたまっている場合、サーバー・プロセスの数は2以上になり、ユーザーがサーバー・プロセスを終了するまでサーバー・プロセスの数は2以上になります。
  - MIN=1、MAX=2以上、ASQCOUNT=2, LIFESPAN=IDLE\_0の場合
 

クライアントからの要求が2つ以上キューにたまっている場合、サーバー・プロセスの数が2以上になります。サービスの実行後、次の要求がない場合はすぐ終了され、再びサーバー・プロセスの数が1になります。
  - MIN=1、MAX=2以上、ASQCOUNT=2, LIFESPAN=IDLE\_secの場合
 

クライアントからの要求が2つ以上キューにたまっている場合、サーバー・プロセスの数が1になります。サービスの実行後、secに設定された時間の間要求がなければ終了され、再びサーバー・プロセスの数が1になります。
- DUMMY = Y | N
  - デフォルト値 : N
  - 一時的にサーバー名を「\_\_dummy」に使用する方式の代わりに使用します。
  - SERVERセクションでDUMMY=Yに設定する場合、tmbootの実行時に当該サーバー・プロセスが起動されません。

- Domain Gateway Cousinの設定など、dummyサーバー設定が必要な場合有効に使用できます。詳細については、「[B.2. ドメインゲートウェイCOUSINの設定方法](#)」を参照してください。
- DUMMYオプションがない場合にはtmbootを実行すると以下のようなエラーが発生します。

```
(F) BOOT0014 exec error : /data1/tmaxkjh/tmax/appbin/tbrtest [BOOT0029]: No such file or directory
```

- AUS = Y | N

- デフォルト値 : N
- Allowing Unregisterd Services
- 環境設定ではサービスを削除し、アプリケーションのサービスは削除(rebuild)していない状態で起動すると、MAXRSTARTの回数だけサーバーの再起動を試みます。
- SERVERセクションでAUS=Yに設定したら、これを無視して起動します。

- CTX\_EREPY = Y | N

- デフォルト値 : Y
- UCSサーバー・タイプでのみ設定できます。他のタイプではこのオプションを使用できません。
- UCSタイプのサーバーでtpsavctx()を呼び出してクライアントの情報を保存した後、tprelay()を呼び出す前にtmtdown -s、tmtdown -s -i、異常終了などによってサーバー・プロセスが終了されると、サービスの呼び出し元はそのサービス要求に対する応答を受信できなくなります。このオプションを「Y」に設定すると、サーバー・プロセスが終了されるとき、まだtprelay()として処理できなかった要求に対して、呼び出し元にエラー応答を返します。
- CTX\_EREPYオプションをデフォルト値として設定すると、サーバー・プロセスが終了する際、当該プロセスで処理されていた要求に対し、呼び出し元にエラー応答を返します。すなわち、同じサーバーの別のプロセスは影響を受けません。tpsavctx()呼び出しにより生成されたクライアント情報(CTX\_T)を、サーバー・プロセス内でのみ管理する場合は正常に動作します。

複数のプロセスがIPCなどを通じてクライアントの情報を共有し、特定のプロセスが異常終了した場合でも別のプロセスによってtprelay()を処理できるように構成された環境では、正常な処理が可能であるに関わらず、エラー応答が返される場合があります。したがって、この場合は設定値を「N」にして、エラー応答が返されないようにします。

- MULTICLH = Y | N

– デフォルト値：Y

– デフォルト値に設定すると、MINCLHが2以上の環境で1つのサーバー・プロセスが複数のCLHからのサービス要求を同時に受けることができます。サーバー・プロセスは、サービス要求を同時に受信したら、先に受信されたサービス要求を処理し、そのサービスがtpreturnで終わったときに別のCLHのサービス要求を処理します。

– 設定値を「N」に設定すると、それぞれのサーバー・プロセスは1つのCLHからのサービス要求を受信して処理します。したがって、複数のCLHから1つのサーバー・プロセスにサービス要求を同時に送信することを防ぐことができます。ただし、各サーバー・プロセスは1つのCLHからのみサービス要求を受けられるので、他のCLHで負荷が発生してもそれとは関係なく、自身に割り振られたCLHの要求のみ処理するため、性能の面では非効率的になる場合があります。

– 設定値をNに設定した場合の注意事項

- サーバーのMIN値を設定する場合、必ずMINCLH以上の値を設定します。
- サーバーのMAX値を設定する場合、必ずMAXCLH以上の値を設定します。MAX設定値がMAXCLH設定値の倍数である必要はありません。
- 各サーバー・プロセスは、自身が対応するCLHをサーバー・プロセス番号(spri)の値が低い順でclh番号に割り当てます。つまり、最初のサーバー・プロセスは最初のCLHが対応し、2番目のサーバー・プロセスは2番目のCLH、サーバー・プロセスの順がMAXCLHを超える場合は、再び最初のCLHから順に対応することになります。

MINCLHとMAXCLHの設定値が異なり、かつCLHがMAXCLHまで起動されていない場合、起動されていないCLHに対応するサーバー・プロセスは、いかなるサービス要求も受けることができません。したがって、tmbootまたはtmmによってサーバー・プロセスが起動されるとき、起動されていないCLHに対応するspriを持つサーバー・プロセスは起動されず、CLHが起動したら、そのCLHに対応するspriのサーバー・プロセスも起動することができます。

- MINCLHとMAXCLHの設定値が異なる環境でCLHを追加で起動した場合、当該CLHに対応するサーバー・プロセスは起動していない状態なので、直ちにtmbootを使って追加起動する必要があります。そうでなければ、そのCLHに接続されたクライアントがサービスを要求したとき、TPENOREADYエラーが返されます。
- MULTICLHオプションがデフォルト値として設定されたサーバーを「N」に変更するとき、MINとMAXの設定値を調整する必要があるかどうかをチェックします。MINとMAXの設定値を変更せずにMULTICLHを「N」に設定した場合、以前よりも各CLHがスケジューリングできるサーバー・プロセスの数が少なくなるため、特定のCLHに負荷が発生したら、要求がキューに蓄積されることがあります。

- $TMAPM = Y \mid N$

- デフォルト値：Y
  - tmapmを使用するかどうかをサービスごとに設定します。
  - ユーザー関数でタイムアウトを再調整する場合、(tpsetsvctimeout()) tmapmを使用しないことをお勧めします。
- MAC = Y | N
    - デフォルト値：N
    - MAC(Master Alive Check)は、TMMの終了時にゾンビ・プロセスとして残ることを防ぐために、場合に応じてサーバーを終了させる必要があります。
    - MAC=Yに設定した場合、TMMが異常終了するとサーバーも一緒に終了します。
- ROC = Y | N
    - デフォルト値：N
    - ROC(Reconnct Outbound CPC)は、サーバーとCLHの接続が切断されたとき、cpcを再接続する必要があります。
    - ROC=Yに設定すると、CLHがアライブの状態でソケットの接続が切断された場合、再接続を行います。
- MAX\_USE\_COUNT = numeric
    - デフォルト値：0
    - 該当するサーバーのサービス呼び出し回数がMAX\_USE\_COUNTよりも大きければ、終了して再起動します。
    - MAX\_USE\_COUNTによりサーバーが再起動されると、TMMはsever closed due to MAX\_USE\_COUNT or MEM\_USAGE(CLOPT:-m)を出力します。

## 再定義できる項目

- LOGLVL = string
  - サーバーのログレベルを設定します。
- ULOGDIR = literal

- ユーザー・メッセージが記録(ロギング)されるディレクトリーの絶対パスを指定します。
  - ユーザー、つまりTmax アプリケーション・プログラマーはuserlog()関数を使用してデバッグ用メッセージ、各種エラー及び警告メッセージなどを簡単にロギングできます。このようにプログラマーがuserlog()関数を使って送信するメッセージを保存するディレクトリーがULOGDIRです。指定されたULOGDIRディレクトリーには日付別に「ulog\_日付」ファイルが作成され、各ファイルには日付別に発生したメッセージが記録されます。
- 項目が定義されていない場合には、TMAXDIRで指定したディレクトリー下のlog/ulogディレクトリーに記録されます。指定したパスには使用中のシステムが存在する必要があります。

---

## 参考

userlog関数についての説明は、『Tmax アプリケーション開発ガイド』を参照してください。

---

- SVCLOG\_FORMAT = literal
  - svclogの項目をユーザーが調整できるようにします。

## 使用例

以下は、SERVERセクションの使用例です。

```
*SERVER
svr1      SVGNAME = svg1,
          CLOPT = "-e err1 -o out1 -- svr1 1",
          MIN = 1,MAX = 10, CONV = N

svr2      SVGNAME = svg1,
          CLOPT = "-e err1 -o out1 -- svr1 2",
          MIN = 1, MAX = 10,
          CONV = N,
          SVRTYPE = UCS,
          CPC = 5
```

### 3.2.5. SERVICEセクション

Tmaxシステムで提供するサービスを登録するセクションです。サービスを提供するサーバー・プロセスが動作中であっても、サービスが環境ファイルに登録されていないとそのサービスは処理されません。そのため、使用するサービスは必ずSERVICEセクションに定義する必要があります。

各サービスには以下の内容を定義できます。

- サービスが提供されるサーバー・プロセス
- サービスの優先順位
- サービス処理の制限時間
- ルーティング情報

SERVICEセクションは以下の形式で定義します。

```
[DEFAULT : ]
Service Name      SVRNAME = server-process-name or gateway-name
                  [FUNCNAME = module-name,]
                  [PRIO = priority-value,]
                  [ROUTING = rout-name,]
                  [SVCTIME = timeout-value,]
                  [EXPORT = (Y)|N,]
                  [AUTOTRAN = Y|(N)|B,]
                  [TXTIME = transaction-timeout-value]
```

## 必須項目

- *Service Name* = string
  - サイズ : 63字以内
  - サーバープログラム内の関数名(サービス・ルーチン名)を使用します。必ずSERVICEセクションで一意的な名前である必要があります。さらに、マルチドメイン環境ではすべてのドメインで一意的である必要があります。
- SVRNAME = string / literal
  - サイズ : 63字以内
  - 当該サービスを提供するサーバー・プロセスを指定します。
  - 当該サービスルーチンを持つサーバープログラムの実行ファイル名が定義されます。サーバー・プロセスはSERVERセクションに定義されている必要があります。サーバー・プロセスに特殊文字がある場合は「サーバー・プロセス名」の形式で定義します。SERVICEセクションで同じサーバー・プロセス名は数回にわたって出現することもあります。1つのサーバー・プロセスに複数のサービスを提供する場合、各サービスに同じサーバー・プロセス名を指定するときです。



## 選択項目

- DEFAULT: 項目 = 値、...
  - 「[3.2.2 NODEセクション](#)」のDEFAULT項目を参照してください。
- FUNCNAME = string
  - サービス名と実際に動作させるモジュールのファイル名が異なる場合は、モジュールの実際ファイル名を記述します。
- PRIO = numeric
  - 範囲 : 1 ~ 100
  - デフォルト値 : 50
  - クライアントの要求を処理する優先順位の値です。各サービスの優先順位を表し、高い優先順位のサービスを低い優先順位のサービスより先に処理します。数字が大きいほど高い優先順位を持ちます。つまり、優先順位の値が100のサービスを一番先に処理します。
- ROUTING = string
  - データ関連のルーティングに使われる項目です。詳細については、「[3.5. 負荷調整の環境設定](#)」を参照してください。
- SVCTIME = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 0
  - 小数点単位で設定が可能です。
  - 値を設定しないとデフォルト値が設定され、無限待ちとなります。
  - サービス処理の制限時間です。各サービスはサービス処理が始まる時点から終わるまで指定されたSVCTIME時間内に処理される必要があります。  
  
SVCTIMEを超過すると、サーバー・プロセスはサービスを中止し、クライアントにエラーを返します。DOMAINセクションのBLOCKTIME項目がクライアントがサービスを要求する時点 (tpcall()またはtpacall() が呼び出される時点) からであるのに対し、SVCTIMEはサーバー・プロセスがCLHプロセスからサービ

ス要求を取得してサービスルーチンを開始する時点からです。この値が0の場合、サービス処理に時間制限がないものとみなします。

- データベースセッションを使用中にサービス・タイムアウトが発生すると、データベースに接続されているセッションを強制的に切断できないため次のサービスを実行できなくなります。Tmaxはこの問題を次の方法で解決します。サービスのタイムアウトが発生した場合、ユーザーはtpsvctimeoutルーチンでTPRETURN(TPEXIT,...)を利用してサーバーを再起動させることができ、これはSERVERセクションに定義したMAXRSTART回数に含まれます。

また、WindowsでXAモードでアプリケーションを開発することを推奨し、Non-XAモードを使う場合はリスタート回数を十分に指定する必要があります。SVCTIMEはサーバーが動作しているときのみ適用されます。そのため、クライアントが要求したサービスがキューに保存されている間にはSVCTIMEが適用されません。

---

## 参考

tpsvctimeoutの詳細については、『Tmaxリファレンスガイド』を参照してください。

---

- EXPORT = Y | N

- デフォルト値 : Y
- マルチドメイン環境でゲートウェイを介してサービス要求を受信した場合、サービスを実行するかどうかを定義します。
- EXPORT=Nに設定する場合、他のドメインでサービスを要求するとTPESECURITYエラーを発生させます。

- AUTOTRAN= Y | N | B

- デフォルト値 : N
- SERVICEセクションに属する項目であり、トランザクション・モードでない状態でサービスが要求メッセージを受信した場合に、トランザクションを自動で起動するかどうかを定義します。
- Tmax 3.8.8以前バージョンのAUTOTRAN項目とは違いがあります。

以前はAUTOTRANを設定しなくても基本的にAUTOTRAN=Yに設定され、tx\_beginを実行しなくてもトランザクションを起動でき、このトランザクションはローカルト・ランザクションでした。一方、Tmax3.8.8で追加されたAUTOTRANは、この項目を設定しないと、基本的に「N」が設定され、トランザクションを自動で起動しません。

そのため、この項目を設定していない場合は、tx\_beginを実行してトランザクションを明示的に起動させる必要があります。この項目を「Y」に設定した場合、当該サービスが他のサービスを要求すると、これ

らのサービスはすべて1つのトランザクション(グローバル・トランザクション)として扱われます。「Y」に設定した場合は、tx\_beginでトランザクションを明示的に起動させると、エラー(TPESVCERR)が発生します。

- 以下は、環境ファイルとtmadminツールの例です。

< config.m>

```
*SVRGROUP
svg2          NODENAME = "tmax1"

### tms for Oracle ###
svg3          NODENAME = "tmax1",    DBNAME = ORACLE,
              OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60",
              TMSNAME  = tms_ora

*SERVER
svr2          SVGNAME = svg2
### servers for Oracle sample program###
fdltest      SVGNAME = svg3

*SERVICE
### services for fdltest ###
FDLINS       SVRNAME = fdltest
FDLSEL       SVRNAME = fdltest
FDLUPT       SVRNAME = fdltest
FDLDEL       SVRNAME = fdltest
TOUPPER      SVRNAME = svr2
```

<tmadminの例>

```
tmax@tmax1:/user/tmax/tmax/config>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmax1 (tmadm): cfg -s
-----
svc_name    prio(pr)  autotran  svctime(st)  routno  svrname    svgno
-----
FDLINS      50        YES       0             -1      fdltest    2
FDLUPT      50        YES       0             -1      fdltest    2
FDLSEL      50        YES       0             -1      fdltest    2
FDLDEL      50        YES       0             -1      fdltest    2
TOUPPER     50        NO        0             -1      svr2       0

$$2 tmax1 (tmadm): q
ADM quit for node (tmax1)
```

- TXTIME = numeric
  - 範囲 : 1 ~ MAX\_INT
  - デフォルト値 : 0
  - 「[3.2.1 DOMAINセクション](#)」のTXTIME項目を参照してください。

## 使用例

以下は、SERVICEセクションの使用例です。

```
* SERVICE
TOUPPER      SVRNAME = svr1,
              PRIO = 1,
              SVCTIME = 60
TOLOWER      SVRNAME = svr1,
              PRIO = 2
```

### 3.2.6. GATEWAYセクション

DOMAIN、NODE、SVRGROUP、SERVER、SERVICEセクションは必須セクションであるのに対し、GATEWAYセクションは選択的なセクションです。

システムが多数のノードで構成されている場合、すべてのノードを1つのドメインで連結すると、システム全体の管理が難しく、ノード間の通信トラフィックが殺到してシステム性能を低下させます。こうした問題を解決するためには適正な数のノードを1つのドメインに割り当て、ドメイン間のゲートウェイを介して連結することで、システム性能を向上させることができます。

GATEWAYセクションは、クライアントのサービスをどのシステムでも処理できるように登録するために使用されます。GATEWAYセクションは1つのドメインに複数を指定することができます。しかし、1つのノードには1つのゲートウェイのみ指定できます。GATEWAYセクションは必須セクションではなく、必要によって選択的に登録できます。当該セクションが登録されている場合は、ゲートウェイの種類によって適切なプロセスが実行されます。

GATEWAYセクションには以下の内容が定義されます。

- ゲートウェイ・プロセスが実行されるノード
- ゲートウェイ間通信のためのTCP/IP情報
- ゲートウェイのタイプ

GATEWAYセクションの基本環境設定の形式は以下のとおりです。

```
[DEFAULT : ]
Gateway Name      NODENAME = node-name,
                  PORTNO = port-number,
                  RGWADDR = remote-ip-addr,
                  RGWPORTNO = remote-port-number,
                  GWTYPE = {TMAX | TMAXNONTX | SNACICS | OSITP | JEUS
                           | JEUS_ASYNC | TUXEDO | TUXEDO_ASYNC | WSGW | XAGW }
                  [MAXINRGW = numeric(32),]
                  [CPC = channel-number,]
                  [COUSIN = gateway-name,]
                  [BACKUP = gateway-name,]
                  [BACKUP_RGWADDR = remote-ip-addr,]
                  [BACKUP_RGWPORTNO = remote-port-number,]
                  [BACKUP_RGWADDR2 = Backup-Tuxedo-ipaddr2,]
                  [BACKUP_RGWPORTNO2 = Backup-Tuxedo-domaingw-portno2,]
                  [BACKUP_RGWADDR3 = Backup-Tuxedo-ipaddr3,]
                  [BACKUP_RGWPORTNO3 = Backup-Tuxedo-domaingw-portno3,]
                  [LOCAL_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                  [RGW_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                  [RGW_B1_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                  [RGW_B2_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                  [RGW_B3_PROTOCOL = "IPV4" | "IPV6" | ",SDP",]
                  [LOCAL_IPV6 = Y|(N),]
                  [RGW_IPV6 = Y|(N),]
                  [RGW_B1_IPV6 = Y|(N),]
                  [RGW_B2_IPV6 = Y|(N),]
                  [RGW_B3_IPV6 = Y|(N),]
                  [TIMEOUT = second,]
                  [DIRECTION = (BIDIR) | IN | OUT,]
                  [CLOPT = string,]
                  [RESTART = (Y)|N,]
                  [MAXRSTART = numeric,]
                  [GPERIOD = numeric,]
                  [LOAD = numeric,]
                  [PTIMEOUT = 1~MAXINT/2,]
                  [PTIMEINT = 1~MAXINT/2,]
                  [GWCHKINT = numeric,]
                  [GWCONNECT_TIMEOUT = numeric,]
                  [NLIVEINQ = numeric,]
                  [TRB = nodename]
```

## 必須項目

- *Gateway Name* = string
  - サイズ : 63字以内
  - ゲートウェイの論理名であり、GATEWAYセクション内で一意な値である必要があります。
- *NODENAME* = literal
  - サイズ : 255字以内
  - ドメイン内でゲートウェイ・プロセスが実行されるノード名を指定する項目です。
  - 使用されるノード名はNODEセクションで定義したノード名である必要があります。
- *PORTNO* = numeric
  - ローカル・ゲートウェイ・プロセスが使う待ち受け(listen)ポート番号です。このポート番号はリモート・ゲートウェイ・プロセスがローカル・ゲートウェイ・プロセスと接続するときに使います。そのためにポート番号をRGWPORTNO項目に登録する必要があります。一般的にシステム領域では1024以内のポート番号を使うため、管理者はこの範囲を避けて使用する必要があります。
  - ポート番号はゲートウェイ間の通信に使われるため、この値が他のところで使われているかどうかを確認する必要があります。
- *RGWADDR* = literal
  - サイズ : 255字以内
  - ローカル・ゲートウェイ・プロセスが接続しようとするリモート・ゲートウェイ・プロセスが実行されるノードのIPアドレスやノード名を登録する項目です。
  - ノード名を指定する場合、そのノード名は「/etc/hosts」ファイルに登録されている必要があります。ローカル・ゲートウェイ・プロセスはRGWADDR項目のIPアドレスとRGWPORTNO項目のポート番号を利用してリモート・ゲートウェイ・プロセスに接続します。
- *RGWPORTNO* = numeric
  - ローカル・ゲートウェイ・プロセスがリモート・ゲートウェイ・プロセスに接続するためのポート番号です。

- このポート番号はリモート・ゲートウェイ・プロセスの待ち受け(listen)ポート番号です。一般的にシステム領域では1024以内のポート番号を使用するため、管理者はこの範囲を避けて使用する必要があります。

---

#### 注

RGWPORTNOはゲートウェイ間の通信に使用されるため、この値の使用の可否を必ず確認する必要があります。

---

- GWTYPE = string

- ローカル・ゲートウェイ・プロセスが接続するリモートゲートウェイのタイプを指定する項目です。
- 以下の値を設定します。

設定値	説明
TMAX	遠隔システムをTmaxシステムに指定します
TMAXNONTX	遠隔システムがTmaxシステムであり、ドメイン間のトランザクションを使用しない場合、ドメイン間のチャンネルをマルチフレックス(multiflexing)してCPCの効率を増加させます。1つの要求が結果を受信するまでチャンネルを占有しないため、応答時間の向上とリソース 使用量の減少効果を得ることができます
TUXEDO	遠隔システムをTuxedoシステムに指定します。Tuxedoのドメイン・ゲートウェイと同期型のチャンネルで通信します
TUXEDO_ASYNC	遠隔システムをTuxedoシステムに指定します。Tuxedoのドメイン・ゲートウェイと非同期型のチャンネルで通信します
JEUS	遠隔システムをJEUSシステムに指定します
JEUS_ASYNC	Webアプリケーションサーバー(WAS)のAsync WebTと入出力(In/Out)通信を非同期(asynchronous)型で行うためのTmaxのゲートウェイです。  GWTYPE = JEUSの場合、遠隔システムはJEUSである必要がありますが、JEUS_ASYNCは他のWASとも連携できます。詳細については、『Tmax WebTAsync ユーザガイド』を参照してください
WSGW	Tmaxサービスを特に変更せずにWebサービスとして使用できるようにするために提供されるゲートウェイで、双方向通信が可能です
XAGW	XAライブラリ(libtxa.so)からのデータを処理するためのゲートウェイで、異機種間の2相コミットを保証します
SNACICS	SNA LU6.2でのみ使用できる、トランザクション通信用のホスト・リンクとのゲートウェイです

## 選択項目

- DEFAULT:項目 = 値、...
  - 「3.2.2 NODEセクション」のDEFAULT項目を参照してください。
- MAXINRGW = numeric
  - 範囲 : 1 ~ 128
  - デフォルト値 : 32
  - マルチドメイン環境で1つのゲートウェイで受信可能なチャンネル数を指定します。
  - TMAXNONTXタイプである場合のみ使用できます。この項目を定義しない場合、1つのゲートウェイは32のドメインに接続してサービス要求を受信できます。
- LOCAL\_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - デフォルト値 : "IPV4"
  - ローカル・ゲートウェイがリスンするために使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続要求を待ちます
IPV6	IPv6プロトコルを使用して接続要求を待ちます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPV4またはIPV6と一緒に使用できます

- RGW\_PROTOCOL = "IPV4" | "IPV6" | ",SDP"
  - デフォルト値 : "IPV4"
  - ローカル・ゲートウェイがリモート・ゲートウェイに接続する際に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続を試みます
IPV6	IPv6プロトコルを使用して接続を試みます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPV4またはIPV6と一緒に使用できます



- LOCAL\_IPV6 = Y | N

- デフォルト値 : N
- ローカル・ゲートウェイがリスンのためにIPv6を使用するかどうかを指定します。

---

#### 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにLOCAL\_PROTOCOL項目を使用することをお勧めします。

---

- RGW\_IPV6 = Y | N

- デフォルト値 : N
- ローカル・ゲートウェイがリモート・ゲートウェイに接続するとき、IPv6を使用するかどうかを指定します。

---

#### 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにRGW\_PROTOCOL項目を使用することをお勧めします。

---

- CPC = numeric

- 範囲 : 1 ~ 128
- デフォルト値 : 1
- 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する項目です。
- ゲートウェイプロセスの処理量が多く、サービスの使用時間が長い場合、マルチチャンネルで並列通信処理することで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つから3つ以下のCPCでも十分です。

- TIMEOUT = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 0
- 値を設定しないとデフォルト値が設定され、無限待ちとなります。

- ドメイン間のtpcall / tpacall時に使われるタイムアウト値を指定します。

0はタイムアウトに制限がないことを意味します。ゲートウェイは他のサーバーと違って30秒間隔でタイムアウトをチェックするため、遅延時間が30秒より小さいとTIMEOUT項目が適用されません。

- COUSIN = literal

- サイズ : 8000字以内
- 同じサービスを複数のゲートウェイを介して負荷分散する必要がある場合、関連したゲートウェイの名前を指定します。ノードまたは他のノードに属しているゲートウェイの場合にも名前を指定することができます。COUSINとBACKUPを一緒に指定できますが、この場合COUSINとBACKUPはそれぞれ異なるゲートウェイに設定する必要があります。
- ルーティングはCOUSINに指定したゲートウェイが対応し、障害が発生した場合、バックアップに指定したゲートウェイが対応します。
- COUSIN項目にはGATEWAYセクションに登録されたゲートウェイ名のみ指定できます。サーバー・グループは指定できません。
- 以下は、環境ファイルの例です。

ゲートウェイgw1とgw2をCOUSINに設定した状態で、GWSVC1サービス要求を受信すると、LOAD設定によって分散処理されます。

< config.m>

```
*SERVICE
GWSVC1      SVRNAME = gw1
GWSVC2      SVRNAME = gw1
GWSVC3      SVRNAME = gw1

*GATEWAY
gw1         NODENAME = node1, GWTYPE = TMAXNONTX, PORTNO = 10001,
            RGWADDR = "192.168.1.1",   RGWPORTNO = 10002,
            LOAD = 1, CPC = 5, CLOPT = "-i",
            COUSIN = "gw1"

gw2         NODENAME = node2, GWTYPE = TMAXNONTX, PORTNO = 10001,
            RGWADDR = "192.168.1.2",   RGWPORTNO = 10002,
            LOAD = 1, CPC = 5, CLOPT = "-i"
```

- LOAD = numeric

- 『3.2.3. SVRGROUPセクション』のLOAD項目を参照してください。

- `BACKUP = gateway name`
  - サイズ : 7999字以内
  - 主ゲートウェイの障害時に使用するバックアップ・ゲートウェイの名前を定義します。このフィールドに定義された値は、Tmax環境ファイルのGATEWAYセクションに定義された値と同じである必要があります。
- `BACKUP_RGWADDR = remote address`
  - サイズ: 255字以内
  - バックアップのためにローカル・ゲートウェイの接続に使われたリモート・ゲートウェイのノード名やIPアドレスを定義します。
  - IPアドレスの代わりにノード名が定義されていれば、「/etc/hosts」ファイルにも定義されている必要があります。
- `BACKUP_RGWPORTNO = remote-port-no`
  - バックアップのためのローカル・ゲートウェイ接続に使われたリモート・ゲートウェイのポート番号を定義します。
  - 待ち受け(listening)しているリモートゲートウェイのポート番号です。一般的にシステム領域では1024以内のポート番号を使用するため、管理者はこの範囲を避けて使用する必要があります。
  - `BACKUP`、`BACKUP_RGWADDR`、`BACKUP_RGWPORTNO`フィールドは障害対策に使われます。

---

#### 注

`BACKUP_RGWPORTNO`はゲートウェイ間の通信に使われるため、それを使用するかどうかをまず確認する必要があります。

---

- `RGW_B1_PROTOCOL = "IPV4" | "IPV6" | ",SDP"`
  - デフォルト値 : "IPV4"
  - ローカル・ゲートウェイがバックアップ・リモート・ゲートウェイに接続時に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。
  - アドレスが`BACKUP_RGWADDR`に、ポートが`BACKUP_RGWPORTNO`に指定されたりリモート・ゲートウェイに接続するときに適用されます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続を試みます
IPV6	IPv6プロトコルを使用して接続を試みます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- RGW\_B1\_IPV6 = Y

- デフォルト値 : N
- ローカル・ゲートウェイがバックアップ・リモート・ゲートウェイに接続する際、IPv6を使用するかどうかを指定します。
- アドレスがBACKUP\_RGWADDRに、ポートがBACKUP\_RGWPORTNOに指定されたりリモート・ゲートウェイに接続するときに適用されます。

---

## 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにRGW\_B1\_PROTOCOL 項目を使用することをお勧めします。

---

- BACKUP\_RGWADDR2、BACKUP\_RGWPORTNO2 = *remote address*

- サイズ : 255字以内
- Tuxedoゲートウェイ(GWTYPE = TUXEDO | TUXEDO\_ASYNC)でのみ設定できます。バックアップ Tuxedoゲートウェイは3つまで指定でき、BACKUP\_RGWADDR2-BACKUP\_RGWPORTNO2、BACKUP\_RGWPORT3-BACKUP\_RGWPORTNO3で指定します。

- RGW\_B2\_PROTOCOL, RGW\_B3\_PROTOCOL = "IPV4" | "IPV6" | ",SDP"

- デフォルト値 : "IPV4"
- Tuxedoゲートウェイ(GWTYPE = TUXEDO | TUXEDO\_ASYNC)でのみ使用します。2番目及び3番目のバックアップ・リモート・ゲートウェイに接続時に使用するプロトコルを指定します。重複設定が可能な場合は、区切り子のコンマ(,)を使用して複数の項目を設定することができます。
- アドレスがBACKUP\_RGWADDR2に、ポートがBACKUP\_RGWPORTNO2に指定されたりリモート・ゲートウェイと、アドレスがBACKUP\_RGWADDR3に、ポートがBACKUP\_RGWPORTNO3に指定されたりリモート・ゲートウェイにそれぞれ接続するときに適用されます。

設定値	説明
IPV4	IPv4プロトコルを使用して接続を試みます
IPV6	IPv6プロトコルを使用して接続を試みます
SDP	InfiniBandのSDP(Socket Direct Protocol)を使用します。IPv4またはIPv6と一緒に使用できます

- RGW\_B2\_IPV6, RGW\_B3\_IPV6 = Y | N

- デフォルト値 : N
- Tuxedoゲートウェイ(GWTYPE = TUXEDO | TUXEDO\_ASYNC)でのみ使用します。2番目および3番目のバックアップ・リモート・ゲートウェイに接続する際、IPv6を使用するかどうかを指定します。
- アドレスがBACKUP\_RGWADDR2で、ポートがBACKUP\_RGWPORTNO2に指定されたリモート・ゲートウェイ、およびアドレスがBACKUP\_RGWADDR3で、ポートがBACKUP\_RGWPORTNO3に指定されたリモート・ゲートウェイに接続するときに適用されます。

---

#### 参考

Tmax v5.0 SP3バージョンからは、この項目の代わりにRGW\_B2\_PROTOCOL、RGW\_B3\_PROTOCOL項目を使用することをお勧めします。

---

- DIRECTION = BIDIR | IN | OUT

- デフォルト値 : BIDIR
- DIRECTIONはGWTYPEがTMAXとTMAXNONTXの場合にのみ有効な項目です。
- 以下は設定値についての説明です。

設定値	説明
BIDIR	双方向の要求処理が可能です
IN	相手側のドメインから受信した要求のみ処理できます。自分のドメインから相手側のトメインへ送信される要求に対してはTPESECURITYエラーが発生します
OUT	相手側のドメインに送信する要求のみ処理できます。相手側のトメインから自分のドメインへ送信される要求に対してはTPESECURITYエラーが発生します

- CLOPT = string

- サイズ : 255字以内

- ゲートウェイが起動するとき、そのゲートウェイに転送されるコマンド・オプションを定義します。定義されたオプションのうち、「--」以前に指定されたオプションはシステムで使い、それ以後に指定されたオプションはユーザーが自由に使えます。

- 以下は、主に使われるユーザー・オプションです。

- [ -h Tmax Version Number ]

Tmax v4.0以降バージョンと4.0以前バージョンのTmaxシステムで構成されたドメインを連動させるためのオプションです。Tmax v4.0以降バージョンに該当するドメインの環境ファイルに相手側のドメインに該当するTmaxシステムのエンジンバージョンを「xyyzz」の6桁形式で指定します。

ゲートウェイは設定されたバージョンに基づいて相手側のドメインと通信します。ただし、相手側ドメインのバージョンは下位バージョンまで明確に明示する必要があります。そうしない場合、サービス・ブロックなど様々な問題が発生することもあります。

TYPE=JEUSである場合(JEUSGWの場合)、「付録A. ゲートウェイのCLOPTセクション」を参照してください。

以下は、Tmax 3.14.4の場合の例です。

```
CLOPT = "-h 031404"
```

- [ -c checktime ]

項目に-cオプションを設定してchecktimeの周期でチェックを行い、CLHとの接続が切断されていれば再接続を試みます。-cオプションでGWのchecktimeを15秒に設定したとしても、BLOCKTIMEが5秒であれば、5秒ごとに(CLHまたはRGWから何の要求もない場合)CLHとのチャンネルが切断しているかをチェックします。

- [ -i ]

サービス要求と独立してチャンネルを接続します。オプションを設定しないと従来と同様に動作します。遠隔ドメインのネットワークやシステム障害によって遠隔ドメインとの接続が切断された場合、再接続を試みます。

従来はサービス要求があれば再接続を試みてGWCONNECT\_TIMEOUTの間待機しましたが、追加された機能では、再接続の実行をサービス要求と分離してサービス要求時に再接続を実行するのではなく、現在の状態に合わせて応答を転送します。つまり、遠隔ドメインとの接続が切断された状況で発生するサービス要求に、より迅速にエラー応答を転送するために使用するオプションです。

サービスを要求時に遠隔ドメインとの接続が切断された状態であれば、即刻TPENOREADYエラーを転送し、遠隔ドメインとの接続が切断されたことを確認した後、すでに遠隔ドメインに要求したサービスに対してはTPESYSTEMエラーを転送します。この場合、旧バージョンではTPETIMEエラーを転送しました。接続されている状態では、サービスを要求します。再接続のサービス要求とは別途に試み、その時間はNLIVEINQまたはBLOCKTIMEのうち、小さい値を使用します。GWCHKINTが設定されていれば、設定された間隔で再接続を試みます。

NLIVEINQによってBLOCTIME値を適切に設定しないとサービス要求に対して迅速な応答を転送することができません。BLOCKTIMEが短く設定された場合にはチャンネルが切断されたことを検知する前にTPETIMEエラーを受信する確率が高まります。また、NLIVEINQが短く設定された場合にはゲートウェイとの接続を頻繁に試みるため、ネットワーク障害時にほとんどの要求に対してTPENOREADYやTPESYSTEMではなく、TPETIMEエラーが発生する確率が高まります。

一方、NLIVEINQが短く設定された場合にはネットワーク障害を検知した後のほとんどの要求に対してTPENOREADYエラーを返しますが、ネットワーク障害の検知に時間が長くなります。IRTゲートウェイを使用する場合、当該オプションを使用して独立的に接続を試みることができるように設定します。その理由は、IRT COUSINを使用する場合には問題のあるゲートウェイにはサービス要求そのものが発生しないため、再接続を試みることができません。したがって、サービス要求とは別に接続を試みる必要があります。

IRT COUSINIは、TMAXNONTX、TMAX、JEUS、JEUS\_ASYNC、TUXEDO、TUXEDO\_ASYNCゲートウェイに対してのみサポートします。

#### • [-R]

Tmax v3.xバージョンとJTmaxはトランザクション・リカバリー機能をサポートしないため、ペンディング・トランザクションに対する処理が実行されないことがあります。したがって、Tmax v5.0バージョンでペンディング・トランザクションに対する処理方法を事前に定義することでペンディング・トランザクションを処理できるようにします。

このオプションに使用できる値は以下のとおりです。

設定値	説明
IGN	ペンディング・トランザクションをコミットまたはロールバックしません。そのため、データベースでコミットまたはロールバックを行う必要があります
COM	ペンディング中のトランザクションはすべてコミットされます
RBK	ペンディング中のトランザクションはすべてロールバックされます

#### • [-r]

トランザクション・リカバリー機能をサポートするオプションです。

区分	説明
GWTYPE=JEUS または JEUS_ASYNC	WebT-JEUSGW間のトランザクション・リカバリー機能を使用します。JEUSGW-JTmax間のトランザクション・リカバリー機能も使用できます。トランザクション・リカバリー機能に対応するWebTバージョンでのみ使用できます
GWTYPE=TUXEDO または TUXEDO_ASYNC	TuxedoでTmaxに接続するときに転送されるTuxedoのドメインID値を設定します。このオプションはTuxedoのドメインIDをチェックして認証を行うときに使用します。当該値に設定されていないドメインIDを持つゲートウェイが接続を試みる際にはエラーが発生し、接続を拒否します

区分	説明
GWTYPE = TMAX	<p>互いに異なるバージョンのTmaxシステムで構成されたドメイン間の連携時にトランザクション復旧を可能にするために使用するオプションです。相手側ドメインに該当するTmaxシステムのエンジン・バージョンを「xyyzz」の6桁形式で指定します。(例: Tmax 3.8.15 : 030815)</p> <p>このオプションを指定すると相手側ドメインに該当するTmaxシステムのエンジンバージョンによってゲートウェイの機能が異なります。現在、次の分類別(Tmaxバージョン)にXA関連動作が異なりますが、相手側ドメインに該当するTmaxシステムのエンジンバージョンが現在のバージョンと異なる分類に属する場合には、必ず[-r]オプションを指定します</p> <ul style="list-style-type: none"> <li>– 分類1 : 2.0から3.8.7</li> <li>– 分類2 : 3.8.から3.8.13</li> <li>– 分類3 : 3.8.14以降(Tmax4.x、Tmax5.x、Tmax6.xを含む)</li> </ul>

• [-a]

Tuxedoに接続するときに転送されるドメインID値を設定します。この名前はTuxedoの環境設定ファイルに定義された名前である必要があります。

• [-l]

ゲートウェイ接続を頻繁に確立したり切断したりする場合、大量のログが記録され、運用上不便が生じます。

このオプションを設定すると、接続時にログイン・ログを記録しないようにするので、ログ量を削減できます。

• [-p configuration file path]

ドメイン・ゲートウェイの要求に対して処理された結果およびメッセージを、指定した環境設定ファイルに定義されたそれぞれのケースにログとして記録します。同機能を使用するには、該当するゲートウェイのLOGLVL項目をDEBUG4に設定します。tmadminのchlogコマンドを使用して動的にLOGLVLを変更することができます。また、stdoutでログが出力されるので、[-o]オプションを使ってログファイル名を指定する必要があります。

このオプションに指定した環境設定ファイルは、各行をtype [on | off]形式で作成します。タイプは0から7までの数字であり、それぞれの意味は以下の表に示します。各タイプに該当する時点におけるメッセージ処理ログを記録する場合は、そのタイプをon(ON)に設定します。

タイプ	説明
0	CLHからの要求(call/acall)メッセージを受信



タイプ	説明
1	CLHからの応答(call/acall)メッセージを受信
2	リモート・ゲートウェイからの要求(call/acall)メッセージを受信
3	リモート・ゲートウェイからの応答(call/acall)メッセージを受信
4	CLHから要求(call/acall)を送信
5	CLHから応答(call/acall)を送信
6	リモート・ゲートウェイから要求(call/acall)を送信
7	リモート・ゲートウェイから応答(call/acall)を送信

以下は、環境ファイルの作成例です。すべてのケースにログを記録する設定です。

```
0 on
1 on
2 on
3 on
4 on
5 on
6 on
7 on
```

同機能が有効になると、以下のような形式でログが記録されます。

「success」は、成功の場合は0、失敗の場合は1に表示されます。それ以外の項目には、適切な値が設定され、次の行に16進数形式でメッセージを出力します。

```
時間 [サービス名] call type(0~7):success(0 or 1):data type(xx):data
size(xx):errno(xx):urcode(xx)
メッセージ
```

#### • [-k]

5.0以上のバージョンと4.0 sp3 fix9以下のバージョンで構成されたマルチドメイン環境において、ドメイン同士がtmaxgwntで接続されており、COUSINに設定されたサービスが存在する場合に設定します。遠隔ドメインでgwとCOUSINに設定されたサービスが再起動するとき、正常にスケジューリングされるようにします。

(注意：5.0以上のバージョン同士で構成されたゲートウェイで設定すると異常動作します。)

– 以下は、項目の設定時に考慮すべき事項です。

- TMAXGW -hオプションにTmax 3.x (例:031404)が設定されている場合、-Rオプションを別途設定しないとデフォルト値のRBKに設定されます。
  - JEUSGW -h1オプションが設定され、-Rオプションを別途設定しないとデフォルト値のRBKに設定されます。
  - JEUSGW -rオプション(リカバリーに対応)と-Rオプションの重複設定はサポートしません。
- 
- RESTART = Y | N
    - デフォルト値：Y
    - GATEWAYセクションに指定した場合、該当ゲートウェイ・プロセスを再起動するかどうかを設定します。異常終了時に再起動します。
- 
- MAXRSTART、GPERIOD
    - 障害対策に使われるフィールドです。詳細については、「[3.8. 障害対策の環境設定](#)」を参照してください。
- 
- PTIMEOUT = numeric
    - 範囲：1 ~ MAX\_INT/2
    - デフォルト値：-1
    - ゲートウェイでペンディング・リスト(pending list)を保持する最大時間を指定します。
    - 最大時間はTIMEOUT項目の値を2に分けた値より小さいか等しく設定します。(TIMEOUT >=PTIMEOUT)
    - ドメイン・ゲートウェイでペンディング(Prepare-done, commit, rollback)を再転送して判断(Decision)を受信できなかった場合でもペンディングが発生しないようにします。

Gateway1、Gateway2が2相コミットで接続された場合、Gateway2が準備(Prepare)を受信した後、コミットまたはロールバックなどを受信できなかった場合は、Gateway2がGateway1にPrepare-doneなどを送信します。ゲートウェイ間のネットワーク問題(または異常終了)などで判断(Decision)を受信できなかった状況でペンディングが発生した場合、PTIMEOUT、PTIMEINTを設定してペンディングを削除することができます。
- 
- PTIMEINT = numeric
    - 範囲：1 ~ MAX\_INT/2

- デフォルト値：-1
  - ゲートウェイでペンディング(Prepare-done, commit, rollback)を再転送する時間間隔を指定します。
  - 最大時間はPTIMEOUT値より小さいか等しく設定します。
- GWCHKINT = numeric
    - 範囲：-1 ~ MAX\_INT
    - デフォルト値：-1
    - ゲートウェイでバックアップに接続された状態でフェイルバックしてメインに接続を試みる周期です。

設定値	説明
-1	DOMAINセクションの設定を参照してください
0	フェイルバック機能を使用しません

- GWCONNECT\_TIMEOUT = numeric
  - 範囲：-1 ~ MAX\_INT
  - デフォルト値：-1
  - ゲートウェイでリモート・ゲートウェイにソケット接続を試みる際、接続が確立するまで待機する時間です。

設定値	説明
-1	DOMAINセクションの設定を参照してください
0	デフォルトで5秒を待機します

- NLIVEINQ = numeric
  - 範囲：-1 ~ MAX\_INT
  - デフォルト値：-1
  - ゲートウェイでリモート・ゲートウェイに接続したチャンネルが有効なのかを確認するため、定期的にチェック・メッセージを送信する周期です。

設定値	説明
-1	DOMAINセクションの設定を参照してください
0	この機能を使用しません

- TRB = string

– この項目についての詳細は、[「3.2.2 NODEセクション」のTRB項目](#)を参照してください。

## 使用例

以下は、GATEWAYセクションの使用例です。

```
*GATEWAY
gw1          NODENAME = "tmax1",
              PORTNO  = 2222,
              RGWADDR = "192.168.23.1",
              RGWPORTNO = 2225,
              GWTYPE  = TMAX,
              CPC     = 10,
              CLOPT   = "-r 030815"
```

以下は、TuxedoのVIEWタイプの使用例です。

```
*DOMAIN
...
*NODE
ibm51      TMAXDIR = "...",
           APPDIR  = "...",

*SVRGROUP
svg1       NODENAME = "ibm51"

*SERVER
svr_sdl    SVGNAME = svg1

*SERVICE
TOUPPER_SDL  SVRNAME = svr_sdl

#svc for tuxedo
TUX_TOUPPERSDL  SVRNAME = TUXGW

*GATEWAY
TUXGW      GWTYPE = TUXEDO,
           PORTNO = 9521,
```

```

RGWADDR = "192.168.1.43",
RGWPORTNO = 9511,
CLOPT = "-a TUXGW1 -v",
NODENAME = ibm51,
CPC = 20,
TIMEOUT = 30

```

以下は、ゲートウェイを利用してローカル・ノードを優先処理する環境設定の例です。

```

*DOMAIN
tmax1      SHMKEY = @SHMEMKY@,
           MINCLH = 1, MAXCLH = 3,
           TPORTNO = @TPORTNO@,
           BLOCKTIME = 25, RACPORT = @TRACPORT@

*NODE
@HOSTNAME@ TMAXDIR = "@TMAXDIR@",
           . . .
@RMTNAME@  TMAXDIR = "@RMTDIR@",
           . . .

*SVRGROUP
svg1      NODENAME = "@HOSTNAME@",
           COUSIN = "svg2, svg3, svg4, gw1, gw2",
           LOAD = -2
svg2      NODENAME = "@HOSTNAME@", LOAD = -2
svg3      NODENAME = "@RMTNAME@",  LOAD = -2
svg4      NODENAME = "@RMTNAME@",  LOAD = -2

*SERVER
svr2      SVGNAME = svg1

*SERVICE
TOUPPER   SVRNAME = svr2

*GATEWAY
gw1       GWTYPE = TMAXNONTX, PORTNO = 7788,
           RGWADDR = "@GATENAME@",
           RGWPORTNO = 6688,
           NODENAME = @HOSTNAME@,
           CPC = 2, LOAD = -2
gw2       GWTYPE = TMAXNONTX, PORTNO = 7788,
           RGWADDR = "@GATENAME2@",
           RGWPORTNO = 6688,
           NODENAME = @RMTNAME@,
           CPC = 2, LOAD = -2

```

上記例では、クライアントがHOSTNAMEで接続してTOUPPERサービスを呼び出した場合、svg2、gw1、svg1に属するサーバーが1:1:1で処理します。svg1に属するサーバーに障害が発生した場合は、svg2、gw1が1:1で処理し、svg2とgw1の両方に障害が発生した場合は、COUSINIに属する他ノードのサーバー・グループが処理します。RMTNAMEで接続した場合、svg3、svg4、gw2に属したサーバーが1:1:1で処理します。

---

#### 注

ドメイン・ゲートウェイのローカル優先処理の負荷分散機能は、非トランザクション・ゲートウェイに限りサポートします。

---

### 3.2.7. ROUTINGセクション

ROUTINGセクションは1つのドメインではノード間、マルチドメイン環境ではドメイン間のデータのルーティング環境を設定するためのセクションです。詳細については、「[3.5. 負荷調整の環境設定](#)」の「データによる負荷調整」を参照してください。

### 3.2.8. RQセクション

RQセクションもROUTINGセクションと同様に、選択的に定義できます。詳細については、「[3.6. 信頼性キューの環境設定](#)」を参照してください。

### 3.2.9. HMSセクション

HMSセクションもROUTINGセクションと同様に、選択的に定義できます。詳細については、「[3.7. HMS環境設定](#)」を参照してください。

### 3.2.10. 基本環境設定の例

以下は、基本的なTmax環境ファイルの例です。

```
*DOMAIN
res          SHMKEY = 77990, MAXUSER = 300, MINCLH = 3,
              MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60

*NODE
tmax1        TMAXDIR = "/home/tmax",
              APPDIR  = "/home/tmax/appbin",
              PATHDIR = "/home/tmax/path",
              SLOGDIR = "/home/tmax/slog",
              ULOGDIR = "/home/tmax/ulog",
              ENVFILE  = svr_env
tmax2        TMAXDIR = "/system/tmax",
```

```

APPDIR = "/system/tmax/server",
SLOGDIR = "/system/tmax/slog",
ULOGDIR = "/system/tmax/ulog",
ENVFILE = app_env

*SVRGROUP
svg1      NODENAME = tmax1
svg2      NODENAME = tmax2
svg3      NODENAME = tmax2, cousin = svg4
svg4      NODENAME = tmax2
*SERVER
svr1      SVGNAME = svg1, CLOPT="-e err1 -- apple", MIN = 3, MAX = 5
svr2      SVGNAME = svg2, MIN = 4, MAX = 5
svr3      SVGNAME = svg3,
          CLOPT = "-e $(SVR).$(DATE).err -o
          $(SVR).$(DATE).out"
*SERVICE
svc1      SVRNAME = svr1, PRIO = 100, SVCTIME = 40
svc2      SVRNAME = svr2
svc3      SVRNAME = svr3

```

### 3.3. データベースの環境設定

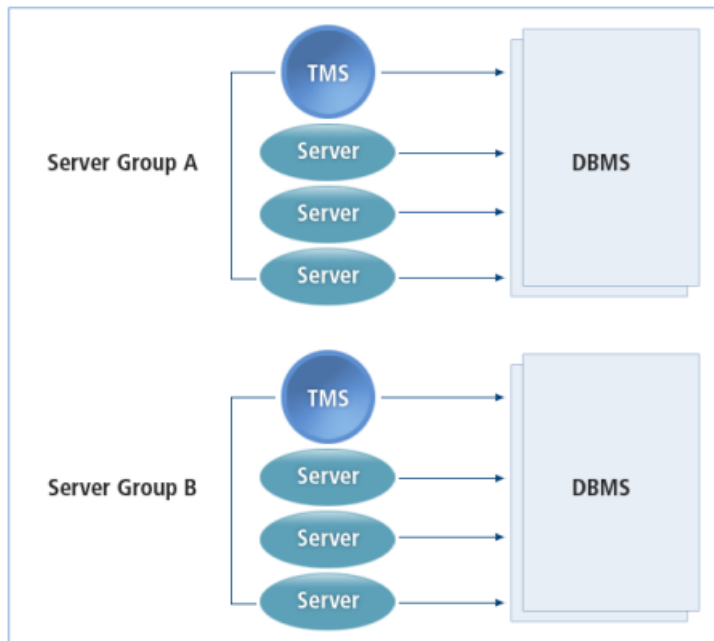
データベースに関連付けられたサービスを提供する場合、環境ファイルにデータベース情報を登録できます。データベース情報を登録することで、アプリケーションで処理するデータベースの起動・終了など、リソースマネージャとの接続および解除がTmaxシステムによって管理され、分散トランザクション処理も可能になります。

データベース環境設定においてサーバー・グループは重要な意味を持ちます。サーバー・プロセスが存在するノードもサーバー・グループを形成する要素になりますが、同一ノードに存在するサーバー・プロセスであっても使用するデータベースの種類によってサーバー・グループが区分されます。

1つのサーバー・グループ内のサーバー・プロセスは必ず同一なデータベース(DBMS)を使用する必要があります。使用するデータベース(DBMS)が異なると同じサーバー・グループになれません。つまり、サーバー・グループはデータベース管理の基本単位です。

こうしたデータベース管理のためにはTMS(Transaction Management Server)というTmax機能プロセスが必要です。TMSプロセスはデータベース管理が必要な場合、サーバー・グループ別に起動され、各サーバー・グループのデータベース管理と次章で説明するトランザクション処理を担当するサーバー・プロセスです。トランザクション処理を担当するTMSプロセスの機能については、「[3.4. 分散トランザクションの環境設定](#)」を参照してください。

【図 3.7】データベース管理のためのTMSプロセス



データベース情報を登録するための環境設定はすべてSVRGROUPセクションで扱われます。

### 3.3.1. SVRGROUPセクション

Tmaxのデータベース管理機能を使用するには、サーバー・グループが使用するデータベースの起動・終了情報とそのサーバー・グループに関連するTMSプロセスについての情報が必要です。

データベース環境設定についてSVRGROUPセクションに定義する項目は以下のとおりです。

```
*SVRGROUP
SVRGROUP Name      [DBNAME = DBMS-name, ]
                   [OPENINFO = DBMS-open-information, ]
                   [CLOSEINFO = DBMS-close-information, ]
                   [TMSNAME = TMS-process-name, ]
                   [MINTMS = TMS-process-min-number, ]
                   [MAXTMS = TMS-process-max-number]
```

- DBNAME = string
  - サイズ : 63字以内
  - 使用するデータベース名を記述します。使われる値はOracle、Informix、Sybaseなどです。
- OPENINFO = string
  - サイズ : 255字以内



- サーバー・グループに関連するデータベース情報を登録する項目です。
- 同じリソース・マネージャー、つまり同じデータベースを使う場合、1つのサーバー・グループで管理することで、リソースの効率を高め、分散トランザクション処理ができます。

OPENINFO形式はリソース・マネージャーを提供するベンダー、つまり、OracleまたはInformixなどのデータベースの種類によって異なります。その他のデータベースは当該データベースのマニュアルを参照して登録します。

- Oracleデータベース

```
OPENINFO="Oracle_XA+Acc=P/scott/tiger+SesTm=60"
```

「Oracle\_XA」はOracleデータベースXAインターフェースを意味しており、「Acc=P/scott/tiger」はOracleデータベースにアクセスするためのアカウントとパスワードです。

- Informixデータベース

InformixデータベースのOPENINFO値は単にデータベース名のみを要求するため、Oracleのように複雑ではありません。

```
OPENINFO= Database Name
```

- CLOSEINFO = literal (NULL)

- サイズ : 255字以内
- デフォルト値 : NULL
- CLOSEINFO項目は、リソース・マネージャーがデータベースをクローズするときに必要な情報を指定します。
- 通常、省略されるかNULLで定義されます。Informixデータベースでは、NULL文字列で指定する必要があり、Oracleデータベースでは省略可能です。

<Oracleを使用するサーバー・グループの例>

```
*SVRGROUP
svg1          NODENAME = tmax1,
              DBNAME = ORACLE,
              OPENINFO = "Oracle_XA+Acc = P/scott/tiger+SesTm= 60"
```

<Informixを使用するサーバー・グループの例>

```
*SVRGROUP
svg2          NODENAME = tmax2,
```

```
DBNAME = INFORMIX,  
OPENINFO = "test",  
CLOSEINFO = ' ' ' '
```

- TMSNAME = string

- サイズ：63字以内
- TMSNAMEは当該サーバー・グループのデータベース管理を担当するTMSプロセス名を定義します。
- データベースの起動情報(OPENINFO項目)を登録した場合は必ずTMSNAMEに対する定義が必要です。TMSプロセスは、データベースと関連するシステムで該当サーバー・グループのデータベースの管理を担当するために必ず必要です。そのため、Tmaxシステムでデータベースを管理できるようにするにはデータベースの起動と終了情報を登録し、サーバー・グループごとに必ずTMSNAMEを定義してTMSプロセスを起動させる必要があります。TMSプロセスはデータベースと連動するXAでトランザクションを処理するトランザクション・マネージャーです。このプロセスは「\$TMAXDIR/lib」にある<libtms.a>とSVRGROUPセクションにあるDBNAMEのデータベース・ライブラリーを関連付けして生成されます。

- MINTMS = numeric

- 範囲：1 ~ 32
- デフォルト値：2
- 該当するサーバー・グループのTMSプロセスの起動数を定義します。TMS数はデータベース管理よりはトランザクションの管理に影響を与える要素です。デフォルトでは1つのサーバー・グループ当たり2つが起動されます。データベースだけを管理する場合は、1つか2つのTMSで十分です。

- MAXTMS = numeric

- 範囲：1 ~ 32
- デフォルト値：3
- 動的にさらに起動できるTMSプロセス数を定義します。詳細については、[「3.4. 分散トランザクションの環境設定」](#)を参照してください。

### 3.3.2. データベース環境設定の例

以下は、基本的な環境設定にデータベース環境設定の内容を追加したTmax環境ファイルの例です。

```

*DOMAIN
res      SHMKEY = 77990,
        MAXUSER = 300 , MINCLH = 3 , MAXCLH = 5 ,
        TPORTNO = 8899, BLOCKTIME = 60

*NODE
tmax1    TMAXDIR = "/home/tmax" ,
        APPDIR = "/home/tmax/appbin",
        PATHDIR = "/home/tmax/path",
        SLOGDIR = "/home/tmax/slog",
        ULOGDIR = "/home/tmax/ulog",
        TLOGDIR = "/home/tmax/tlog"
tmax2    TMAXDIR = "/system/tmax",
        APPDIR = "/system/tmax/server",
        SLOGDIR = "/system/tmax/slog",
        ULOGDIR = "/system/tmax/ulog",
        TLOGDIR = "/system/tmax/tlog",
        ENVFILE = app_env

*SVRGROUP
svg1     NODENAME = tmax1, DBNAME = ORACLE,
        OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60",
        TMSNAME = ora_ tms, MINTMS = 3, MAXTMS = 5
svg2     NODENAME = tmax2, DBNAME = INFORMIX,
        OPENINFO = "infodb",
        TMSNAME = info_tms, CLOSEINFO = ""

*SERVER
svr1     SVGNAME = svg1 ,
        CLOPT="-e err1 -- apple", MIN = 3,
        MAX = 5
svr2     SVGNAME = svg2 , MIN = 4 , MAX = 5
svr3     SVGNAME = svg2,
        CLOPT = "-e $(SVR).$(DATE).err -o
        $(SVR).$(DATE).out"

*SERVICE
svc1     SVRNAME = svr1 ,
        PRIO=100, SVCTIME = 40
svc2     SVRNAME = svr2
svc3     SVRNAME = svr3

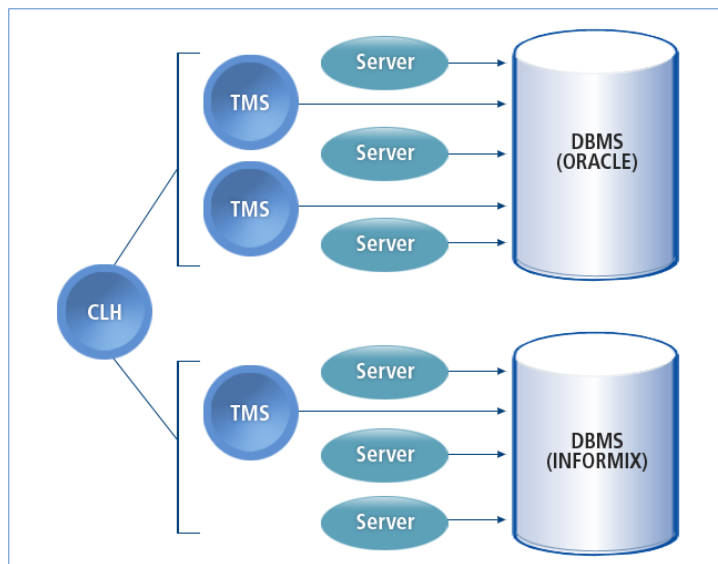
```

## 3.4. 分散トランザクションの環境設定

分散トランザクション処理はTP(Transaction Processing)モニターの主要機能の1つです。分散トランザクション処理とは、1つ以上の異種データベース・システムに関連する複数のトランザクションを1つのグローバ

ル・トランザクション(global transaction)にまとめて処理することです。こうした分散トランザクション機能を使用したい場合は基本的にデータベース環境設定が必要です。分散トランザクション処理のためには、各サーバー・グループ別にTMSプロセスが起動され、以下の構造を形成します。

**[図 3.8] 分散トランザクション処理環境**



CLHは分散トランザクションと関連するサービス要求を受信する場合、サービス別に下位トランザクションをコミットまたはロールバックするかどうかを決めます。つまり、1つの分散トランザクションに関連する下位トランザクションがすべてコミットできるとき、その分散トランザクションをコミット処理し、1つでもロールバック処理されると全体をロールバック処理することで分散トランザクションをサポートします。

実際、データベースを更新(update)するコミットまたはロールバック処理は各サーバー・グループのTMSプロセスが管理します。分散トランザクションのための環境設定はDOMAIN、NODE、SVRGROUPセクションにより行なわれます。

### 3.4.1. DOMAINセクション

DOMAINセクションには分散トランザクションに関連し、2相コミット(Two Phase Commit)の可能設定とタイムアウト時間の設定ができます。

データベース環境設定に関連し、DOMAINセクションに定義する項目は以下のとおりです。

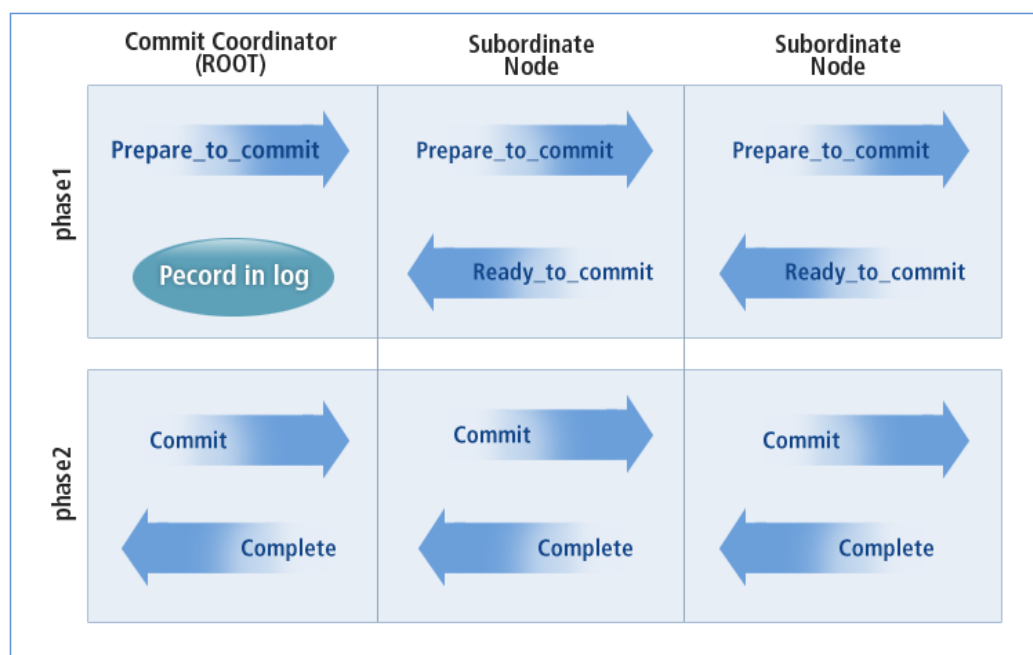
```

*DOMAIN
Domain Name      [CMTRET = (Y) | N,]
                  [TXTIME = transaction-timeout-value]
  
```

- CMTRET = Y | N
  - デフォルト値 : Y

- 2PC(Two Phase Commit)の如何を決定します。2相コミットとは、準備(Prepare)と実行(コミットまたはロールバック)の2段階トランザクション処理のことをいい、Tmaxでは分散トランザクションを実装するための2PCをサポートします。
- 1段階(準備):1つのトランザクションに関連して、各データの管理サーバーに当該トランザクションの正常処理(コミットまたはロールバック)を確認します。
- 2段階(コミットまたはロールバック):1段階の結果に基づいて、関連するすべてのトランザクションが正常処理されるとコミットし、1つでも正常処理が不可能であればロールバックします。

[図 3.9] 2PCの処理プロセス



Tmaxでは2PCに対して2つの方法をサポートします。1段階から2段階を経てコミットし、ロールバックに対する完全なロギングが実行された後、サービスを返します(complete return)。より迅速な処理のためには、第2段階でコミットまたはロールバック・コマンドのみを指示してサービスを返します。システム環境に応じて選択的に指定できます。

- CMTRET項目はcomplete returnの内容を決定します。

設定値	説明
Y	complete returnを意味します
N	迅速な処理のため、コミットまたはロールバック命令後に即刻サービスを返します

- TXTIME = numeric
  - 範囲 : 0 ~ MAX\_INT

- デフォルト値 : 0
- トランザクション処理の制限時間を決めます。tx\_begin()のトランザクション開始からtx\_commit()やtx\_rollback()でトランザクションが終了するまでの時間であり、TXTIMEが過ぎると自動でロールバック処理されます。

### 3.4.2. NODEセクション

NODEセクションでは分散トランザクションの環境設定と関連し、各ノードのトランザクション処理記録に使えるファイルを定義できます。

データベースの環境設定に関連し、NODEセクションに定義する項目は以下のとおりです。

```
*NODE
Node Name      [TLOGDIR = transaction-log-path]
```

- TLOGDIR = literal
  - サイズ : 255字以内
  - デフォルト値 : TMAXDIR
  - トランザクションを処理中、障害が発生すると復旧のためにトランザクション処理をロギングする必要があります。TLOGDIRはこうしたトランザクション情報を保存するファイルを指定します。

ログファイルは一般ファイルで保存されます。Tmaxはログファイルを保存する際、rawファイルをサポートしません。トランザクション処理が多くない場合にはTLOGDIRに定義された一般ファイルを使用し、任意のファイル名を定義します。逆にトランザクション処理が多い場合には応答時間(response time)を短縮させるため、UNIXファイル・システムを介さず、定義されたrawファイルを使用してTLOGDIRに装置(device)名を直接ディスクに記録します。ここに定義された装置はトランザクション・ロギングだけのために使用します。

---

#### 注

定義されたパスは使用中のシステムに存在することが前提です。

---

### 使用例

TLOGDIRパスのログファイルは一般ユーザーが読み込めないバイナリ形式で保存されています。トランザクションに関連するデータベースログはSVRGROUPセクションのOPENINFOにある追跡フラグ(trace flag)を定義して保存できます。

```
TLOGDIR = "/home/tmax/log/tlog"
```

/tmpディレクトリーのxa\_NULLmdd.trcファイルに以下のように作成します。

```
OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm = 60, DbgFl = 0x01 + LogDir=/tmp"
```

### 3.4.3. SVRGROUPセクション

SVRGROUPセクションではデータベースの環境設定を行います。設定項目のうち、MINTMSとMAXTMSはトランザクション処理に関連するため注意して設定してください。設定項目についての詳細は、「[3.3. データベースの環境設定](#)」を参照してください。

SVRGROUPセクションの基本環境設定形式は以下のように定義します。

```
*SVRGROUP
SVRGROUP Name      [DBNAME = name-of-database,]
                   [OPENINFO = string,]
                   [CLOSEINFO = string,]
                   [TMSNAME = name-of-tms,]
                   [MINTMS = numeric,]
                   [MAXRSTART = numeric,]
                   [MAXTMS = numeric]
```

- DBNAME = string
  - 使用するデータベース名を設定します。
- OPENINFO = literal
  - データベースの起動情報を登録します。
- CLOSEINFO = literal
  - デフォルト値：NULL
  - データベースの終了情報を登録します。
- TMSNAME = string
  - TMSプロセス名を設定します。
- MINTMS = numeric
  - TMSプロセスの最小数です。

- データベース管理だけの目的ならTMSプロセスの個数は重要ではないですが、トランザクション処理の際は重要になります。TMSプロセス数が1、2個に固定されていれば、トランザクション処理量が多いサーバー・グループの場合、コミットまたはロールバック処理にかなりの負荷を招くこともあり得ます。そのため、処理速度とリソース管理のためには適切な個数のTMSを起動させることが重要です。
- MAXRSTART = numeric
  - 障害対策に使用されるフィールドで、詳細については、「[3.8. 障害対策の環境設定](#)」を参照してください。
  - SVRGROUPセクションに設定された場合、TMSまたはRQのMAXRSTARTを意味します。
- MAXTMS = numeric
  - TMSプロセスの最大数で、動的にさらに起動できるTMSプロセス数を定義します。トランザクション処理量が増加すると、MAXTMS数を超えない範囲内でTMSプロセスをさらに起動できます。

## 使用例

以下は分散トランザクションの環境設定に関連するTmax環境ファイルの例です。

```
*DOMAIN
res      SHMKEY = 77990, MAXUSER = 300 ,
          MINCLH = 3 , MAXCLH = 5 ,
          TPORTNO = 8899 ,
          BLOCKTIME = 60 ,
          CMTRET = N , TXTIME = 50

*NODE
tmax1    TMAXDIR = "/home/tmax",
          APPDIR = "/home/tmax/appbin",
          PATHDIR = "/home/tmax/path",
          SLOGDIR = "/home/tmax/log/slog",
          ULOGDIR = "/home/tmax/log/ulog",
          TLOGDIR = "/home/tmax/log/tlog "
tmax2    TMAXDIR = "/system/tmax",
          APPDIR = "/system/tmax/server",
          SLOGDIR = "/system/tmax/slog",
          ULOGDIR = "/system/tmax/ulog",
          TLOGDIR = "/system/tmax/log/tlog"

*SVRGROUP
svg1     NODENAME = tmax1, DBNAME = ORACLE ,
          OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=60,
          DbgFl=0x01 + LogDir=/tmp",
```



```

        TMSNAME = ora_tms,
        MINTMS = 3, MAXTMS = 5
svg2      NODENAME = tmax2, DBNAME = INFORMIX,
        OPENINFO = "infodb", CLOSEINFO = "",
        TMSNAME = info_tms,
        MINTMS = 3, MAXTMS = 5
svg3      NODENAME = tmax2

*SERVER
svr1      SVGNAME = svg1,
        CLOPT = "-e err1 -- apple", MIN = 3, MAX = 5
svr2      SVGNAME = svg2, MIN = 4, MAX = 5,
        CLOPT = "-e $(SVR).$(PID).err -o $(SVR).$(PID).out"
svr3      SVGNAME = svg3
        CLOPT = "-e $(SVR).$(DATE).err -o $(SVR).$(DATE).out"

*SERVICE
svc1      SVRNAME = svr1, PRIO = 100, SVCTIME = 40
svc2      SVRNAME = svr2
svc3      SVRNAME = svr3

```

## 3.5. 負荷調整の環境設定

クライアントのサービス要求が集中する場合に、最適のシステム性能とリソースの活用を保証できます。

負荷調整には以下の3つの方法があります。

- システム性能による負荷調整(System Load Management : SLM)

ノードの性能やシステムの状態によってノード別にサービス処理量を調整します。

- データによる負荷調整(Data Dependent Routing : DDR)

データの範囲に応じてノード別にサービスを分散して処理します。

- 動的負荷調整(Dynamic Load Balancing : DLM)

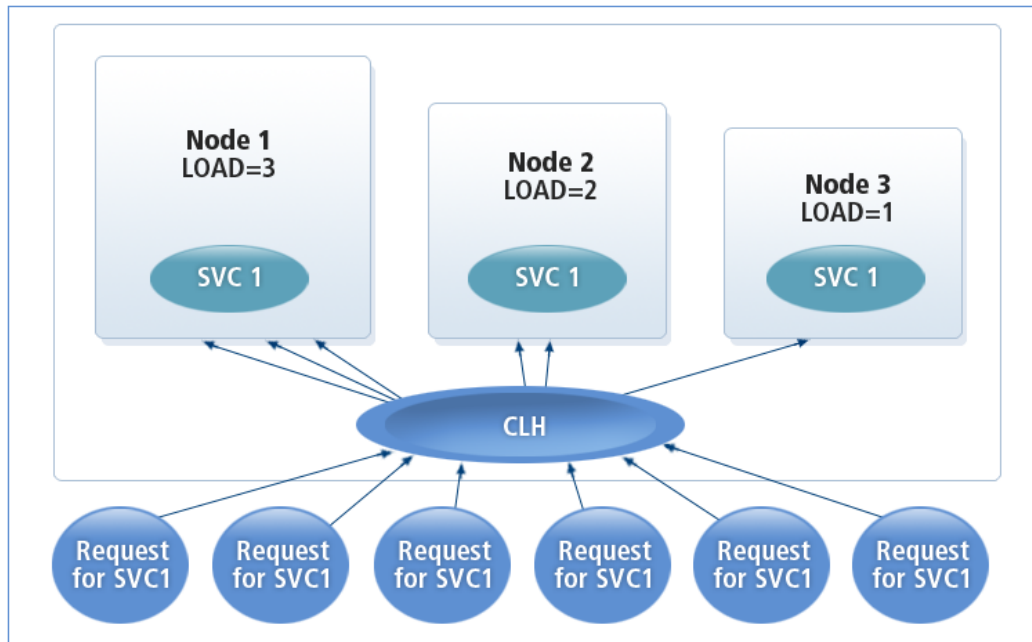
特定ノードに負荷が集中する場合、負荷を分散してシステム全体の処理量を最大化します。

各環境設定ファイルの項目を説明時に、本節で言及していない項目については、[「3.2. 基本環境設定」](#)を参照してください。

### 3.5.1. システム性能による負荷調整

ノードの性能やシステムの状態によってノード別にサービス処理量を異なるように指定します。このような負荷調整ができるには1つのサービスが複数のノードでサポートされている必要があります。システム・ロードはサーバー・グループ別に設定されます。

[図 3.10] システム・ロードによる負荷調整



## SVRGROUPセクション

システム性能による負荷調整の環境設定に関連し、SVRGROUPセクションに定義する項目は以下のとおりです。

```
*SVRGROUP
SVRGROUP Name          [COUSIN = group-name,]
                        [LOAD = load-value]
```

- COUSIN = literal
  - サイズ : 255字以内
  - 2つ以上のノードで負荷調整機能を使用する場合、サーバー・グループ名を指定する項目です。
  - 各サーバー・グループ名は、SVRGROUPセクションに登録します。基本的にTmaxシステムの負荷調整機能はサーバー・グループ単位で行われます。ただし、負荷調整が多数のノードにわたって行なわれる場合、当該ノード名は各サーバー・グループに異なる名前で設定する必要があるため、この際、ノード名は環境ファイルのNODEセクションに登録されているものである必要があります。

多数のサーバー・グループを定義する場合はコンマ(,)で区切って定義します。各ノードのAPPDIRには必要なサーバー・プログラムが存在している必要があります。ノードの性能によって負荷調整をする必要がある場合には各サーバー・グループに負荷調整のためのロード値を設定します。

- LOAD = numeric

- 範囲：-1 ~ MAX\_INT

- デフォルト値：1

- サーバー・グループのサービス処理能力を意味します。値が大きいほど処理能力が高いことを意味します。

- サービス処理能力に優れたノードに対してはLOAD値を大きく設定し、ノードの性能が低下すればLOAD値を小さく設定します。

システム・ロードによる負荷を調整したい場合、COUSIN項目設定にサーバー・グループをコピーした後、各サーバー・グループに適切なLOAD値を設定します。ノード別に分散処理したくない場合は、LOAD値を-1に設定します。LOAD値を設定しないか0に設定すると自動負荷分散が行われ、システム負荷によって自動でサービス要求が分散されます。

## 使用例

以下はSVRGROUPコピーのためのTmax環境ファイル設定の例です。

- SVRGROUPコピーのためのマルチノードのTmax環境ファイル

```
*DOMAIN
site1      SHMKEY = 77990, MAXUSER = 400,
           MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60,
           CMTRET = N, TXTIME = 50

*NODE
NODE1      TMAXDIR = "/home/tmax",
           APPDIR = "/home/tmax/appbin",
           PATHDIR = "/home/tmax/path",
           SLOGDIR = "/home/tmax/log/slog",
           ULOGDIR = "/home/tmax/log/ulog",
           TLOGDIR = "/home/tmax/log/tlog "
NODE2      TMAXDIR = "/system/tmax",
           APPDIR = "/system/tmax/appbin",
           SLOGDIR = "/system/tmax/log/slog",
           ULOGDIR = "/system/tmax/log/ulog",
           TLOGDIR = "/system/tmax/log/tlog"
```

```

NODE3      TMAXDIR = "/system/tmax",
           APPDIR = "/system/tmax/appbin",
           SLOGDIR = "/system/tmax/log/slog",
           ULOGDIR = "/system/tmax/log/ulog",
           TLOGDIR = "/system/tmax/log/tlog"

*SVRGROUP
SVG1       NODENAME = NODE1, COUSIN = "SVG2,SVG3".....(1)
           LOAD = 3 , DBNAME = ORACLE,
           OPENINFO = "ORACLE_XA+Acc=P/scott/tiger+SesTm=60",
           TMSNAME = tms1, MINTMS = 3
SVG2       NODENAME = NODE2, .....(2)
           LOAD = 2
SVG3       NODENAME = NODE3, .....(3)
           LOAD = 1, DBNAME = INFORMIX ,
           OPENINFO = "infodb", CLOSEINFO = "",
           TMSNAME = tms2
           .....

*SERVER
SVR1       SVGNAME = SVG1, MIN = 5 , MAX = 10
SVR2       SVGNAME = SVG1
           .....

*SERVICE
SVC_A      SVRNAME = SVR1
SVC_B      SVRNAME = SVR1
SVC_C      SVRNMAE = SVR2

```

- SVRGROUPコピーのための単一ノードのTmax環境ファイル

```

*DOMAIN
sitel      SHMKEY = 77990, MAXUSER = 400,
           MAXCLH = 5, TPORTNO = 8899, BLOCKTIME = 60,
           CMTRET = N, TXTIME = 50

*NODE
NODE1      TMAXDIR = "/home/tmax",
           APPDIR = "/home/tmax/appbin",
           PATHDIR = "/home/tmax/path",
           SLOGDIR = "/home/tmax/log/slog",
           ULOGDIR = "/home/tmax/log/ulog",
           TLOGDIR = "/home/tmax/log/tlog "

*SVRGROUP
SVG1       NODENAME = NODE1, COUSIN = "SVG2"
           LOAD = 3 , DBNAME = ORACLE,

```

```

OPENINFO = "ORACLE_XA+Acc=P/scott/tiger+SesTm=60",
TMSNAME = tms1, MINTMS = 3
SVG2      NODENAME = NODE1, LOAD = 2
.....

*SERVER
SVR1      SVGNAME = SVG1, MIN = 5 , MAX = 10
SVR2      SVGNAME = SVG1
.....

*SERVICE
SVC_A     SVRNAME = SVR1
SVC_B     SVRNAME = SVR1
SVC_C     SVRNMAE = SVR2

```

### 3.5.2. データ値による負荷調整

Tmaxシステムではデータ値に応じて負荷調整機能を提供します。データ値による負荷調整(Data Dependent Routing: 以下DDR)とは、データの範囲によってサービスが処理されるサーバー・グループを別々に指定し、分散処理及び負荷調整が行われるようにすることです。

ルーティングは、STRUCT、STRING、CARRAY型バッファやFIELD型バッファを使用するサービスに対して指定できます。データ依存型ルーティングをするためには、SVRGROUPセクションにサーバー・グループ環境のコピーに関する環境設定とSERVICEセクションとROUTINGセクションのルーティングに関する環境設定が必要です。

## SVRGROUPセクション

DDR環境設定に関連し、SVRGROUPセクションに定義する項目は以下のとおりです。

```

* SVRGROUP
SVRGROUP Name          [COUSIN = group-name]

```

- COUSIN = literal
  - サイズ : 255字以内
  - DDRは、ルーティングを必要とするサービスが多数のサーバー・グループで提供される必要があります。したがって、COUSIN項目の設定によって本来のSVRGROUPが他のノードにコピーされる必要があります。詳細については、[「3.2.3. SVRGROUPセクション」](#)の「COUSIN」項目を参照してください。

## SERVICEセクション

DDR環境設定に関連し、SERVICEセクションに定義する項目は以下のとおりです。

```
* SERVICE
SERVICE Name          [ROUTING = routing-name]
```

- ROUTING = string
  - サイズ : 63字以内
  - ルーティングの論理名を定義します。
  - SERVICEセクション内で一意である必要があります。この名前は以下に説明されたROUTINGセクションにも使われるものです。DDRを特定サービスに応用するためにはルーティング名が定義される必要があります、型付きバッファの名前とルーティングを必要とするフィールドおよびその範囲をサーバー・グループ名と一緒に定義します。

## ROUTINGセクション

1つのドメイン内でノード間、またはマルチドメイン環境下でドメイン間のDDRはROUTINGセクションに定義された型付きバッファの特定フィールド値またはメンバー値によって実行されます。

1つのドメイン内のノード間のDDRは、SERVICEセクションで定義した「ROUTING Name」で実行され、ROUTINGセクションにはユーザーが必要とする条件を記述します。ROUTINGセクションには「ROUTING Name」と共にSTRUCT、STRING、CARRAY型バッファやFIELD型バッファ名を定義し、そのバッファでルーティングを適用するメンバーもしくはフィールドとその値の範囲を、それを処理するサーバー・グループ名と共に定義します。マルチドメイン環境のDDRは、サーバー・グループ名の代わりにドメインのゲートウェイ名を指定します。

ROUTINGセクションの形式は以下のとおりです。

```
* ROUTING
ROUTING Name          FIELD = BufferType/field-name,
                      RANGES = " 1: Group1,  2: Group 2,  ....",
                      MATCH = (FIRST) | MULTIPLE
```

- ROUTING Name = string
  - サイズ : 63字以内
  - SERVICEセクションのROUTING項目に定義した名前です。データによる負荷調整はサービス別に行うため、SERVICEセクションに登録された名前である必要があります。

- FIELD = string

- サイズ : 2000字以内

- FIELDには、データ型別に設定する方法が異なります。データ依存型ルーティングを適用するサービスに対し、そのサービスが使用するSTRUCT、FIELD、STRING、CARRAYのタイプ情報を設定します。

- 各データ型の形式は以下のとおりです。

- STRUCT型のバッファ

STRUCT型バッファはユーザーが定義するC構造体であり、X/Openに定義されたX\_C\_TYPE、X\_COMMONバッファと同じです。ユーザーは該当する構造体の名前と内容を予め宣言して、手続きに従ってTmaxシステムが必要とするファイルを提供し、以下のように設定します。Tmaxシステムは環境変数のSDLFILEを参照して構造体の内容を識別します。

```
STRUCT/subtype/fieldname
```

項目	説明
subtype	構造体の名前を指定します
fieldname	構造体のメンバーのうち、実際にルーティングを適用するメンバーを選択して指定します

- FIELD型バッファ

ユーザーは予めFIELD型バッファのフィールド項目とタイプを定義し、手続きに従ってTmaxシステムが必要とするファイルを提供する必要があります。Tmaxシステムは環境変数のFDLFILEを参照してフィールド・バッファの内容を識別します。

FIELD型バッファが使われている場合には以下のように設定します。

```
FIELD/fieldname
```

項目	説明
fieldname	ルーティングを適用するフィールド名を指定します

- STRING型、CARRAY型バッファ

STRING型とCARRAY型のデータに対しても、データ依存型ルーティング(DDR)を適用できるようにサポートしています。CARRAY型は「CARRAY」または「CARRAY/offset/length」に設定し、STRING型は「STRING」または「STRING/offset/length」の形式に設定すると、STRING型とCARRAY型に対するDDRにみなされます。データ依存型ルーティング(DDR)は一致タイプと範囲タイプの2方式に区分できます。一致タイプと範囲タイプの違いは[3.5.2節「使用例」](#)で説明されています。CARRAY型には画像タイプのデータが入力できますが、その場合のデータは、印刷可能なタイプである必要があります。

```
STRING or STRING/offset/length  
CARRAY or CARRAY/offset/length
```

offsetとlengthについての説明は、次に説明する「STRING/CARRAYのoffset」、「STRING/CARRAYのlength」項目を参照してください。

- STRING / CARRAYのoffset = numeric(0)

- 全体データのうち一部のみをDDRに使用するときを設定する項目であり、入力データのうちDDRに使用する部分の最初の開始位置を示します。

offsetはFLDLEN項目と一緒に指定します。入力データの当該offsetからFLDLEN分のデータだけDDR時に使用されます。たとえば、STRING/0/4と設定した場合、offsetは0、lengthは4になり、"Tmaxsoft"という入力データを処理するとすると、0番目の位置から4Byte分の"Tmax"文字列だけがDDRの比較基準になります。この項目はCARRAY/STRING型のDDR時にのみ有効です。

- offsetとlengthを使用時の注意事項は、一致タイプのときは使用できず、範囲タイプのときのみ使用できるといことです。これはoffsetとlengthが入力データのみを対象に指定する項目であり、RANGES項目に指定するデータとは関連していないためです。

以下のように設定されている場合、"aaaaaaa" の文字列の0 offsetから4Byte分の"aaaa"文字列だけがDDR時の比較基準になります。入力データの比較基準の"aaaa"とRANGESに指定した"aaaaaaa"は一致しないためSVG2で処理します。この場合、どのデータが入力されても無条件にSVG2で処理するため注意が必要です。

```
ROUTE_1    FIELD = "STRING/0/4",  
           RANGES = "aaaaaaa":SVG1, *:SVG2
```

- 範囲タイプで設定されている場合、offsetとlengthが有効です。

下記のように設定されている環境で入力データが"aaaaaaa"の場合、入力データの中の0 offsetから4Byte分の"aaaa"だけが比較基準になり、"aaaa"という文字列はSVG1で指定した範囲より小さくなるため、SVG2で処理します。入力データが"bbbb"の場合、SVG1で指定した範囲内であるためSVG1で処理されます。

```
ROUTE_1    FIELD = "STRING/0/4",  
           RANGES = "aaaaaaa"-"ccccccc":SVG1, *:SVG2
```

- STRING / CARRAYの length = numeric

- offsetが入力データの中でDDR時に比較基準として使おうとする部分の先頭の開始位置を示すのであれば、lengthはoffsetの位置から指定した長さを比較基準として使うときに指定する項目です。そのため、lengthは必ずoffsetと一緒に使用します。
- 指定されていない場合、NULLが出現するまで、またはメッセージの最後まで比較します。



- RANGES = "範囲：グループ1、範囲：グループ2、...."

- サイズ：2047字以内

- RANGES項目はFIELDで定義した項目の値を基準に、ルーティング先の情報を設定します。

- RANGES項目の値を指定する形式は以下のとおりです。

- ルーティング・フィールドにはint、short、long、float、double、string、char型を指定できます。
    - 整数や実数の場合、1つの値または下限値や上限値で構成された範囲で表現できます。
    - 文字列が一致タイプの場合、一致するか否かのみを処理します。範囲タイプであれば、下限値と上限値の範囲で表現できます。
    - 符号(負数、正数)を表示できます。
    - 最小値を示すMIN(該当するタイプの最小値)と最大値を示すMAX(該当するタイプの最大値)文字を使用できます。
    - "A - B"形式の範囲では、AがBより小さい必要があります
    - アスタリスク(\*)は前の他の範囲に含まれていない任意の値を代表するワイルド文字として使われます。前に定義された範囲以外の値が出現することを考慮し、必ずアスタリスク(\*)のルーティング・サービスを定義します。ただし、MINとMAXを両方とも定義した場合は、アスタリスク(\*)の定義は省略できます。
    - サーバー・グループ名は範囲の後のコロン(:)の次に指定します。使用するサーバー・グループ名はSVRGROUPセクションのCOUSIN項目に定義されたサーバー・グループ名と同じである必要があります。
    - 文字列全体を二重引用符(" ")で囲みます。
    - ドメイン間のルーティングはサーバー・グループ名の代わりにドメイン・ゲートウェイ名を指定します。
    - 文字列でルーティングする場合、当該値を引用符(' ')で囲みます。

- MATCH = (FIRST)/MULTIPLE

- MATCH項目を使用しない場合

```
*ROUTING
rout1    FIELD = "STRING, CARRAY, FIELD/INPUT,
```

```
STRUCT/test/a",
RANGES = "min-'100':svg1,'101'-'200':svg1_2,min-'100':svg1_3,*:svg1_4"
```

上記のように環境ファイルが構成されている場合、以前のバージョンではmin-100までのデータを処理するために、svg1のサーバーをまず検査し、svg1に属したノードが正常に起動していないとバックアップを検査し、バックアップが起動されていないか存在しないとクライアントにエラーを送信しました。一方、現在のバージョンでROUTINGセクションにMATCH項目を指定すると新しい方式で動作します。

#### – MATCH項目を使用する場合

```
*ROUTING
rout1    FIELD = "STRING, CARRAY, FIELD/INPUT,
          STRUCT/test/a",
          RANGES ="min-'100':svg1,'101'-'200':svg1_2, min-'100':svg1_3,*:svg1_4"
          ,
          MATCH=MULTIPLE
```

上記のように環境ファイルが構成されている場合、min-100までのデータを処理するために、svg1のサーバーをまず検査し、svg1が属したノードが起動していないとバックアップを検査します。バックアップが起動していないか存在しないとRANGESで次に一致するサーバー・グループを検索します。したがって、svg1\_3がその対象になります。svg1\_3が属するノードが有効であるかを検査し、svg1\_3のサーバー・グループが起動していない場合、svg1\_3サーバー・グループのバックアップを検査し、バックアップが存在しないか正常に起動していないと、その次に一致するサーバー・グループはsvg1\_4になります。

「MATCH=FIRST」に指定すると従来と同じ方式で動作します。

## 使用例

以下はDDRの使用例です。

#### ● 整数の場合

以下はstudent構造体のnumberメンバーに対し、そのデータ値が負数から3まではサーバー・グループのSVG1に、4から6はSVG2に、7はすべてSVG3にルーティングする例です。

```
ROUTE_1    FIELD = "STRUCT/student/number",
          RANGES = "MIN - 3 : SVG1, 4 - 6 : SVG2, 7 - MAX : 1SVG3"
```

以下はstudent構造体のnumberメンバーに対し、偶数はサーバー・グループのSVG1で処理され、負数はSVG2で処理される例です。MOD演算子によって範囲が決まるため、ユーザーの幅を広げることができます。

```
ROUTE_1    FIELD = "STRUCT/student/number",
          RANGES = "MOD2 = 0 : SVG1, MOD2 = 1 : SVG2"
```

#### ● FIELD型バッファの場合

以下はフィールド型バッファのINPUTフィールド値がSTRINGの場合、サーバー・グループSVG1で処理され、その他の場合にはサーバー・グループSVG2で処理される例です。ドメイン間にルーティングする場合は、サーバー・グループ名の代わりにドメイン・ゲートウェイとして指定されたゲートウェイ名を指定します。文字列の場合には必ず引用符(' ')で囲んで表現します。

```
ROUTE_1      FIELD = "FIELD/INPUT",
              RANGES = "'STRING' : SVG1, *: SVG2"
```

- STRING型、CARRAY型バッファの場合(一致タイプ)

以下はSTRINGまたはCARRAYの値が'aaa'である場合のみグループSVG1で処理され、その他の場合にはサーバー・グループのSVG2で処理される例です。一致タイプの場合、注意すべき点は、有効なデータ長は指定した単語と完全に一致するときのみ当該範囲にみなされるということです。SVG1に指定された文字列が'aaa'だとaaa 以外の文字、たとえばaaab、aaaccなどの文字はすべてSVG2で処理します。

```
ROUTE_1      FIELD = "STRING",
              RANGES = "'aaa' : SVG1, *: SVG2"
ROUTE_2      FIELD = "CARRAY",
              RANGES = "'aaa' : SVG1, *: SVG2"
```

- STRING型、CARRAY型バッファの場合(範囲タイプ)

以下はSTRINGまたはCARRAYの値が'aaa'から'ccc'までの場合、サーバー・グループのSVG1で処理され、その他の場合にはサーバー・グループのSVG2で処理される例です。範囲タイプのデータ依存型ルーティング(DDR)は一致タイプと違って、指定された範囲の長さ限定されず、文字列の最後まで検査します。そのため、指定したデータ長と関係なく、当該文字列が当該範囲内に属するかを検査します。

```
ROUTE_1      FIELD = "STRING",
              RANGES = "'aaa'-'ccc' : SVG1, *: SVG2"
ROUTE_2      FIELD = "CARRAY",
              RANGES = "'aaa'-'ccc' : SVG1, *: SVG2"
```

- STRING型、CARRAY型の場合(範囲タイプ、DDRの比較長の設定)

以下は入力データのうち、0 offsetから4Byte分の“aaaa”文字列のみをDDR時の比較基準として使用する例です。たとえば、入力データが“aaaaaa”という文字列であれば、“aaaa”だけが比較基準として使われます。“aaaa”はSVG1で指定した範囲より小さいためSVG2で処理します。入力データが“bbbbbb”であれば“bbbb”という文字列はSVG1で指定した範囲に属するため、SVG1で処理します。

```
ROUTE_1      FIELD = "STRING/0/4",
              RANGES = "'aaaaaa'-'cccccc':SVG1,*:SVG2",
```

- 複数のデータ型を同時に使用する場合

以下はSTRING、CARRAY、FIELDを同時にDDRの比較基準として使用する例です。入力データが1000 以下の場合はsvg1で、1001以上2000以下の場合はsvg2で、その他の場合はsvg3で処理します。

```
ROUTE_1      FIELD = "STRING/0/4, CARRAY, FIELD/INPUT",
              RANGES = "min-'1000' : svg1, '1001'-'2000':
              svg2, *:svg3"
```

## 3.6. 信頼性キューの環境設定

クライアントのサービス要求が集中して直ちに処理できず、内部キューに蓄積された状態でシステム・エラーが発生し異常終了すると、すべてのサービス要求データが削除されます。このような問題を改善するために、サービス・データをメモリーではなくディスク・ファイルに溜めて、システムが正常に開始された後に処理を続けられるように環境設定を行います。このとき使用するキューが信頼性キュー(RQ)です。

RQを動作するために基本環境設定(DOMAIN、NODE、SVRGROUP、SERVER、SERVICEセクション)のほか、RQセクションを別途設定する必要があります。

RQを使用する場合、メモリーではなくディスクを利用するため、サービスを要求したデータは安全に保管されますが、RQを使用しないときより速度は低下する可能性があります。

### 1. SVRGROUPセクション

RQシステムを使用するためには、RQが属するサーバー・グループを定義する必要があります。そのためにSVRGROUPセクションの基本必須項目以外に、SVGTYPE項目を「RQMGR」に指定します。その他にもCPCなどを設定してシステムの運用効率を高めることができます。

SVRGROUPセクションの環境設定フォーマットは以下のように定義します。

```
*SVRGROUP
SVRGROUP Name      NODENAME = node-name,
                   CPC = number,
                   SVGTYPE = [RQMGR|TXRQMGR],
                   TMSNAME= "POD"
```

- SVRGROUP Name = string
  - サイズ: 63字以内
  - サーバー・グループの名前を設定します。SVRGROUPセクション内で一意な名前である必要があります。
- NODENAME = string
  - サーバー・グループが属するノード名を設定します。NODEセクションで定義したノード名である必要があります。
- CPC = numeric
  - 範囲: 4~128

- RQSプロセスとCLHプロセス間の並列通信チャンネル数を指定します。RQを使用しない場合は無視されます。RQを頻繁に使用する場合、マルチチャンネルを維持することで処理速度を向上することができます。

- SVGTYPE = [RQMGR|TXRQMGR]

- サーバー・グループのタイプを定義します。

設定値	説明
RQMGR	RQを使用する場合に設定します
TXRQMGR	<p>トランザクション環境でRQを使用する場合に設定します。</p> <p>以下はトランザクション環境設定に関する注意事項です。</p> <ul style="list-style-type: none"> <li>- RQサーバー・グループのタイプ(SVGTYPE)を「RQMGR」の代わりに「TXRQMGR」に設定します。</li> <li>- 従来のtpenq/tpdeq APIはトランザクションをサポートしません。</li> <li>- tpenq_ctlとtpdeq_ctlは1つのトランザクションとして扱うことができません。</li> <li>- tpenq_ctlをトランザクションとして扱う場合、RQに保存するプロセスまでをトランザクションとして処理します。svc callについてはトランザクションをサポートしません。</li> <li>- deq_timeを設定した場合、設定時間の後にsvc callをします。「現在時間 + delay_time」に設定します。</li> </ul> <p>delay_timeを10に与えたい場合、以下のように設定します。</p> <pre>time(&amp;t); ctl-&gt;deq_time = t + 10;</pre> <ul style="list-style-type: none"> <li>- deq_timeはトランザクションと一緒に使用できません。</li> </ul>

- TMSNAME = "POD"

- RQがPODサーバーのように動作します。TMSNAMEフィールドに"POD"を入力した場合、tpenq要求に対してのみ起動を行います。

## 使用例

```
*SVRGROUP
svg1      NODENAME = tmax1,
          CPC = 4,
          SVGTYPE=RQMGR
```

以下は、トランザクション環境でRQを使用する場合の環境設定の例です。

```
*SVRGROUP
Txrqsvg    NODENAME = tmaxh4,
          CPC = 4 ,
          SVGTYPE = TXRQMGR
```

## 2. RQセクション

各サーバー・グループは1つ以上のRQを保持して使用できます。RQセクションにはRQの管理方法を設定します。RQセクションの項目には、QSIZE、FILEPATH、BOOT、FSYNC、BUFFERINGなどがあります。このうち、**SVGNAME**は必ず設定する必要があります。

RQセクションの環境設定フォーマットは以下のように定義します。

```
*RQ
RQ Name    SVGNAME = server-group-name,
          BOOT = COLD | WARM,
          QSIZE = number,
          BUFFERING = Y | N ,
          FSYNC = Y | N ,
          FILEPATH = file-path,
          MAX_MEMQCOUNT = number
```

- RQ Name = string
  - サイズ: 14字以内
  - RQセクションの論理的な名前です。RQセクション内で複数のRQ定義が可能であり、それぞれのRQ名は一意的な値である必要があります。また、1つのサーバー・グループに1つ以上のRQを定義できます。
- SVGNAME = string
  - RQを使用するサーバー・グループ名を設定します
  - SVRGROUPセクションで定義したサーバー・グループ名である必要があります。
- BOOT = COLD | WARM

- デフォルト値: COLD
- BOOTフィールドはTmaxシステムが再起動されるときRQに保存されているデータを調整します。
- 以下は設定可能な値です。

設定値	説明
WARM	システム障害復旧の後、キューに溜まっているデータの復旧が可能になります
COLD	Tmaxシステムが再起動されるとき、ディスクに保存されているデータが削除されます

● QSIZE = numeric

- 範囲: 1~2047MB
- デフォルト値: 16MB
- RQのサイズを指定して使用できます。RQは基本的にTMAXDIRのpathディレクトリーに生成され、サービス処理を継続した結果、指定されたキューのサイズを超えた場合には、tpeq()時にTPEQFULLエラーが設定されます。RQのサイズは自動で増加しないため、RQが定義したサイズを越えてエラーが発生することを防ぐためには、適切なサイズを設定する必要があります。RQファイルを作成した後もRQデータが保存されない場合は、Tmaxが終了時にファイルが自動で削除されます。

● BUFFERING = Y | N

- デフォルト値: Y
- RQファイル内容をメモリーにキャッシュするかどうかを指定します。
- 以下は設定可能な値です。

設定値	意味
N	RQ処理が相対的に遅くなりますが、要求されるメモリーは小さくなります

● FSYNC = Y | N

- デフォルト値: Y
- BUFFERINGを「N」に設定しても、オペレーティング・システムはデータを即時ディスクに保存するわけではなく、最長30秒間メモリーに保管します。この待機時間を省いて即時メモリーのデータをディスクに保存させる場合に使用します。
- 以下は設定可能な値です。

設定値	意味
N	システム障害が発生した場合にデータを失う可能性があります。RQ処理速度は速くなります
Y	RQデータがメモリーを使用せずに常にディスクに安全に保存されます。ただし、Nを設定したときより処理速度は遅くなります

- FILEPATH = literal

- デフォルト値: \$(TMAXDIR)/path/RQ名.data

- RQで使用するファイルの場所を指定します。このフィールドはRQデータ・ファイルを作成し、このファイルは全体ファイル名とディレクトリー名で定義します。

---

#### 参考

TMAXDIRはTmaxをインストールしたホーム・ディレクトリーです。以降、Tmaxホーム・ディレクトリーをTMAXDIRと記述します。

---

- MAX\_MEMQCOUNT = numeric

- デフォルト値: 1024

- RQでTPRQS\_NON\_PERSISTENTタイプを指定した場合、つまりファイルにデータを記録せずにメモリーにのみデータを保存する場合に保存できるメッセージの最大数です。

## 使用例

```
*RQ
rqtest  SVGNAME = svg1,
        BOOT = COLD,
        QSIZE = 1024,
        BUFFERING = Y,
        FSYNC = Y,
        FILEPATH = "/user1/tmax/myrq.data"
```

## 3.7. HMS環境設定

メッセージング・システム(Messaging System)とは、送信者(Sender)と受信者(Receiver)間の緩結合(loosely coupled)を可能にする通信媒体です。メッセージング・システムを使用して送信者と受信者は相互に対する情報を知る必要なく仮想チャンネル、つまり、デスティネーション(Destination)の情報だけで相互通信できます。また送信時点と受信時点を分離できるため、データの遅延処理が可能です。メッセージング・システムはそのためのメッセージの送受信の信頼性(Reliability)を保証します。



HMS(Hybrid Messaging System)を使用するには、DOMAIN、NODE、SVRGROUP、HMSセクションの設定が必要です。HMSについての詳細は、『Tmax HMSユーザガイド』を参照してください。

### 3.7.1. DOMAINセクション

以下は、DOMAINセクションでHMSを使用するための環境設定形式と項目についての説明です。

```
*DOMAIN
Domain1      [MAXSESSION = numeric]
```

#### 選択項目

- MAXSESSION = numeric
  - サイズ : 1 ~ 65535
  - デフォルト値 : 1024
  - ドメイン内のHMSで作成できるセッション数の最大値を設定します。

### 3.7.2. NODEセクション

以下は、NODEセクションの環境設定形式と項目についての説明です。

```
*NODE
Node1        [MAXSESSION = numeric]
```

#### 選択項目

- MAXSESSION = numeric
  - サイズ : 1 ~ 65535
  - デフォルト値 : 1024
  - ノード内のHMSで作成できるセッションの最大数を設定します。

### 3.7.3. SVRGROUPセクション

以下は、SVRGROUPセクションの環境設定形式と項目についての説明です。

```
*SVRGROUP
ServergroupName      NODENAME = node-name,
                     SVGTYPE = HMS,
                     HMSNAME = string,
                     OPENINFO = literal,
                     HMSINDEX = numeric,
                     HMSMAXTHR = numeric,
                     HMSMAXDBTHR = numeric,
                     [HMSMAXBULKTHR = numeric,]
                     [HMSMAXBULKSIZE = numeric,]
                     [HMSOPT = literal,]
                     [HMSSUBSCFG = string,]
                     [HMSMSGLIVE = numeric,]
                     [HMSPORT = numeric,]
                     [HMSHEARTBEAT = numeric,]
                     [HMSGQINT = numeric]
```

#### 必須項目

- *ServergroupName* = string
  - サイズ: 63文字以内
  - HMSサーバー・グループの論理的名前として、SVRGROUPセクション内で一意の名前である必要があります。
- *NODENAME* = string
  - サイズ: 63文字以内
  - HMSサーバー・グループが存在するノードを定義します。使用される*NODENAME*は*NODE*セクションで定義したノード名である必要があります。
- *SVGTYPE* = string
  - 該当するサーバー・グループのタイプを定義します。HMSを使用するためには、必ず*SVGTYPE*=HMSと設定します。
- *HMSNAME* = string

- サイズ: 63文字以内
- ユーザーがビルドしたHMSプロセス名を定義します。
- OPENINFO = literal
  - HMSは基本的にデータベースをストレージとして使用します。したがって、HMSで接続するDBMSのOPENINFOを設定する必要があります。
  - データベースへの接続を初期化し、各データベースで提供する文法で定義します。
- HMSINDEX = numeric
  - サイズ: 0~65535
  - HMSのメッセージを処理するのに必要な設定値として、必ずドメイン内で一意の値で設定します。
- HMSMAXTHR = numeric
  - サイズ: 0~65535
  - ストレージ処理をしないスレッドの数を設定します。
  - 非永続的メッセージのみ送受信する場合、この項目の値を大きくします。非永続的メッセージ以外にもHMSの基本的な動作のために一定数以上設定します。
- HMSMAXDBTHR = numeric
  - サイズ: 0~65535
  - ストレージ処理をするスレッドの数を設定します。
  - 永続的メッセージが多い場合、この値を大きくします。永続的メッセージ処理以外にもHMSの基本的な動作のために一定数以上設定します。

## 選択項目

- HMSMAXBULKTHR = numeric
  - 範囲: 1~65535
  - HMSメッセージに対するストレージを一括処理するスレッドの数を設定します。

- HMSMAXBULKSIZE = numeric

- 範囲：2~65535

- HMSメッセージに対するストレージを一括処理する際、一度に処理できるメッセージの最大数を設定します。

- HMSOPT = literal

- サイズ：255文字以内

- HMSが起動される際、HMSプロセスに渡されるコマンド・オプションを定義します。

オプション	説明
-eファイル名	HMSの動作中に発生する標準エラー(standard error)をファイルに記録します
-oファイル名	HMSの動作中に発生する標準出力(standard output)をファイルに記録します

- HMSSUBSCFG = string

- サイズ：255文字以内

- 永続的サブスクライバーを設定した環境ファイルのパスと名前を設定します。ファイル内の内容は、Labelを[TopicName]と設定した後、各行ごとにコロン(:)で区切ってClientName:Listener:Selectorの順に設定します。

- 環境ファイルに定義された永続的サブスクライバーはHMSが起動すると自動的に登録されます。

- HMSMSGGLIVE = numeric

- サイズ：0~65535(単位：時間)

- HMSは永続的メッセージをストレージに格納します。ストレージに蓄積されたメッセージを周期的に削除する時間を設定します。

- すべてのサブスクライバーが受信したトピック・メッセージ、消費が完了したキュー・メッセージは、設定時間が経過するとストレージから削除されます。まだ受信していない永続的レシーバーが存在するメッセージは、設定時間が経過してもストレージから削除されません。

- HMSPORT = numeric

- 多数のHMSをクラスタ化する場合、各HMSは互いに通信するために直接チャンネルを作成します。この際に使用するポート番号を設定します。

- GLOBAL属性を設定したデスティネーションが存在する場合、HMSは設定されたポート番号で待機します。したがって、すでに使用しているポートを設定しないように注意します。
- HMSHEARTBEAT = numeric
  - サイズ : 0~65535(単位 : 秒)
  - 多数のHMSをクラスタ化する場合、HMS間のネットワーク障害を感知するために、周期的にHeartbeatメッセージを送受信します。そのメッセージを送信する周期を設定します。
  - 周期内のHeartbeatメッセージが他のHMSから来なかった場合は障害と感知し、該当HMSとの接続を切断します。
- HMSGQINT = numeric
  - サイズ : 0~65535(単位 : ミリ秒)
  - GLOBAL属性を設定したキュー間のメッセージを送受信するためには、各キューの状態情報を送受信する必要があります。この情報を送受信する周期を設定します。
  - クラスタ化されたキューを使用するためには必ず設定します。周期が短いほど正確な分配が可能です、その分ネットワークの負荷は大きいため、適切なサイズの値を設定する必要があります。

### 3.7.4. HMSセクション

以下は、HMSセクションの環境設定形式と項目についての説明です。

```
*HMS
DestinationName    SVGNAME = string,
                   TYPE = TOPIC | QUEUE,
                   [BOOT = WARM | (COLD),]
                   [GLOBAL = Y | (N),]
                   [GQTHR = numeric,]
                   [GQMAXREQ = numeric,]
                   [GQFULL = numeric]
```

#### 必須項目

- *DestinationName* = string
  - サイズ : 63文字以内

- デスティネーションの名前を設定します。プロデューサーとコンシューマはこの名前を使用して作成されます。
- SVGNAME = string
  - サイズ : 63文字以内
  - デスティネーションが存在するHMSサーバー・グループ名を定義します。
  - SVGNAMEはSVRGROUPセクションで定義したHMSサーバー・グループ名(SVRGROP名)である必要があります。
- TYPE = TOPIC | QUEUE
  - デスティネーションのタイプを設定します。現在はTOPICとQUEUEのみサポートします。

## 選択項目

- BOOT = WARM | COLD
  - デフォルト値 : COLD
  - 該当するデスティネーションに対して起動する際に復旧するメッセージがある場合の復旧可否を設定します。
  - RQと同様、WARMとCOLDのみ設定可能であり、デフォルト値はCOLDです。

区分	説明
WARM	ストレージにある未処理のメッセージが復旧します
COLD	起動時に該当するメッセージをすべて削除してから起動します

- GLOBAL = Y | N
  - デフォルト値 : N
  - クラスタ化されたHMSの同じ名前を持つデスティネーションを論理的に1つのデスティネーションで動作するように設定します。
  - GLOBAL設定をするためには、SVRGROUPセクションのHMS設定にHMSPORTが設定されている必要があります。

- キューのGLOBAL設定をするためには、次の項目で紹介するGQTHR、QMAXREQ、QFULLの追加設定が必要です。

- GQTHR = numeric

- サイズ : 0~65535

- クラスタ化されたキューが維持するバッファ・サイズを決定する係数値を設定します。

バッファサイズ = (コンシューマ数) X (GQTHRの値)

- GQTHR値です。現在のキューの長さがバッファのサイズより小さい場合、割り当てるメッセージが存在するクラスタ化された他のキューからメッセージを取得します。バッファのサイズより少ないメッセージを持っている場合は、メッセージの割り当て要求を受けてもメッセージを割り当てません。

- QMAXREQ = numeric

- サイズ : 1~65535

- クラスタ化されたキューが他のキューにメッセージを要求する場合に、1度に要求できるメッセージの数を設定します。設定された値が大きい場合は少ない回数で必要なメッセージを埋めることができますが、大き過ぎる場合はノード間の不要なメッセージの移動が発生することがあるため、設定値を大きくし過ぎないことを推奨します。

- QFULL = numeric

- サイズ : 0~65535

- クラスタ化されたキューのメッセージ消費が行われない場合やメッセージが不均衡に非常に多く蓄積される場合に備えて、設定した数以上のメッセージが蓄積した場合はその超過分に限って他のノードのクラスタリング・キューにメッセージを均等に分けます。

## 3.8. 障害対策の環境設定

Tmaxでは以下の障害対策を提供します。

- ハードウェア的な障害(ノードまたはネットワーク障害): サービス・バックアップによる障害対策
- ソフトウェア的な障害(プロセスのダウン): プロセスの再起動による障害対策

### 3.8.1. ハードウェア的な障害

障害によりサービスができない場合、バックアップ・サービスを利用してサービスを引き続き提供できます。サービスのバックアップはSVRGROUPセクションのBACKUP項目に設定します。

#### SVRGROUPセクション

```
* SVRGROUP
SVRGROUP Name          [BACKUP = group-name]
```

- BACKUP = literal
    - サイズ : 255字以内
    - バックアップ・サーバー・グループを指定します。
    - 1つ以上のバックアップ・サーバー・グループを指定できます。2つ以上のバックアップ・サーバー・グループを指定する場合は、コンマ(,)で区切って記述します。複数のバックアップ・サーバーを指定した場合、本来のサーバーノードに障害が発生すると、1番目に指定されたバックアップノードが動作し、そのバックアップ・ノードにも異常が発生すると、2番目に指定されたバックアップノードが動作する多重障害対策をサポートします。
    - バックアップノードのサーバー・グループ名は必ずSVRGROUPセクションに登録します。また、サーバー・グループとサーバーの環境オプションは基本的に本来のサーバーの環境に従います。環境オプションを変更したいときは、サーバー・グループまたはサーバー別に関連項目を再定義できます。ただし、サービスの環境オプションは二重コピーされたグループ別に再定義できず、本来のサーバー・グループと同じ環境を持ちます。
- これに対する規則は、負荷調整のために二重コピーされたサーバー・グループと同じです。バックアップノードは負荷調整のためにコピーされたサーバー・グループとは違って、本来のサーバー・グループのようにTmaxの起動時に動作するのではなく、障害が発生した場合のみ、本来のサーバー・グループに代わって動作します。

### 3.8.2. ソフトウェア的な障害

CLH、CASなどのTmaxの内部プロセスが異常終了した場合、プロセスが自動で再起動されて引き続きサービスを提供します。

#### NODEセクション

```
* NODE
NODE Name              [RESTART = (Y) | N,]
```



```
[MAXRSTART = numeric,]  
[GPERIOD = time-value]
```

- RESTART = Y | N

- デフォルト値 : Y
- TmaxエンジンプロセスのうちCLH、CAS、CLL、TLMを再起動させるかどうかを決めます。
- CLHとCAS、CLL、TLMに問題が発生して異常終了(down)した場合、再起動するにはRESTARTを「Y」に設定します。

- MAXRSTART = numeric

- 範囲 : -1 ~ MAX\_INT
- デフォルト値 : -1
- Tmaxエンジンプロセスのうち、CLHとCASプロセスの最大の再起動回数を決めます。
- GPERIOD項目と一緒に使用されます。

RESTARTが「Y」の場合、プロセスはGPERIODの時間内にMAXRSTARTの最大回数まで再起動されます。「N」に設定すると、システムはこの値を無視します。MAXRSTARTが-1の場合、プロセスの最大の再起動回数は無限になります。

- GPERIOD = numeric

- 範囲 : 1 ~ MAX\_INT
- デフォルト値 : 86400 (24時間) (単位 : 秒)
- MAXRSTART回数が有効な時間の周期です。

GPERIOD時間の間、MAXRSTART回数までプロセスが再起動されます。その時間が経過するとGPERIOD時間の間、再度MAXRSTART回数の分だけ再起動されます。

## SVRGROUPセクション

```
* SVRGROUP  
SVRGROUP Name [RESTART = (Y) | N,]
```

```
[MAXRSTART = numeric,]  
[GPERIOD = time-value]
```

- RESTART、MAXRSTART、GPERIOD

- SVGTYPEがRQMGRの場合は当該RQの再起動を、XA svgの場合は当該TMSの再起動を調整します。
- RQとTMSプロセスが異常終了した場合、プロセスが自動に再起動され、引き続きサービスを提供します。
- 各項目についての詳細は、前述した[3.8.2節「NODEセクション」](#)を参照してください。

## SERVERセクション

```
* SERVER  
SERVER Name      [RESTART = (Y) | N,]  
                  [MAXRSTART = numeric,]  
                  [GPERIOD = time-value]
```

- RESTART、MAXRSTART、GPERIOD

- サーバー・プロセスを再起動するかどうかを指定します。
- ゲートウェイ・プロセスが異常終了した場合、プロセスが自動で再起動され、引き続きサービスを提供します。
- MAXRSTARTのデフォルト値は5です。
- 各項目についての詳細は、前述した[3.8.2節「NODEセクション」](#)を参照してください。

## GATEWAYセクション

```
* GATEWAY  
GATEWAY Name     [RESTART = (Y) | N,]  
                  [MAXRSTART = numeric,]  
                  [GPERIOD = time-value]
```

- RESTART、MAXRSTART、GPERIOD

- ゲートウェイを再起動するかどうかを指定します。

- ゲートウェイ・プロセスが異常終了した場合、プロセスが自動で再起動され、引き続きサービスを提供します。
- 各項目についての詳細は、前述した[3.8.2節「NODEセクション」](#)を参照してください。

## 3.9. セキュリティーの環境設定

### 3.9.1. 段階別のセキュリティ設定

Tmaxシステムは内部で以下の3段階のセキュリティをサポートします。

- 第1段階: システム接続制御

Tmaxシステムへの接続を制限します。

- 第2段階: ユーザー認証

認証されたユーザーのみサービスを要求できます。

- 第3段階: サービス・アクセス制御

サービス別にユーザーのアクセスを制御できます。

セキュリティの各段階は以前の段階を含みます。つまり、第2段階のユーザー認証のセキュリティが設定された場合、第1段階のシステム接続制御のセキュリティも適用されます。セキュリティを設定する段階は、DOMAINセクションのSECURITY項目で決めます。SECURITY項目に定義できる値は以下のとおりです。

## DOMAINセクション

```
* DOMAIN
DOMAIN Name      [SECURITY = ( "NO_SECURITY" ) | "DOMAIN_SEC" | "USER_AUTH" | "OTHER_AUTH", ]

                  [OWNER = user-name]
```

- SECURITY = "NO\_SECURITY" | "DOMAIN\_SEC" | "USER\_AUTH" | "ACL" | "MANDATORY"
  - デフォルト値: "NO\_SECURITY"
  - 以下は設定値についての説明です。

設定値	内容
NO_SECURITY	いかなるセキュリティ設定も行いません
DOMAIN_SEC	<p>第1段階のセキュリティとしてシステム接続制御を定義します。DOMAINセクションのSECURITY項目に「DOMAIN_SEC」を定義します。</p> <p>mkpwを利用してTmaxシステムに対する単一パスワードを設定し、このパスワードを知っているユーザーに限って接続を許可します。このセキュリティを設定したい場合、DOMAINセクションにOWNER項目を定義する必要があります</p>
USER_AUTH	<p>第2段階のセキュリティであるユーザー認証を定義します。DOMAINセクションのSECURITY項目に「USER_AUTH」を定義します。ユーザー・アカウントとパスワードを管理し、認証されたユーザーにのみTmaxサービスを提供します。ユーザーとパスワードはmkpwで作成します。ユーザーはTmaxシステムに接続するときTPSTART_T構造体の「username」と「usrpwd」項目に認証できるユーザー・アカウントとパスワードを登録して接続を要求します。認証されていないユーザーはTmaxシステムに接続できません</p>
ACL, MANDATORY	<p>第3段階セキュリティであるサービス・アクセス制御を定義します。</p> <p>Tmaxシステムに接続したユーザーの中でサービス別にユーザーのアクセスを制限できる機能です。サービス・アクセス権制限はユーザー・グループ別に行われます。1つのサービスはそのサービスに対してアクセス権限が許可されたグループに該当するユーザーのみアクセスできます。そのため、この機能を使用するにはグループ・ファイルを作成する必要があり、当該グループに属するユーザー・ファイルが必要です。また、サービス別にアクセスできるユーザー・グループを指定するaclファイルも必要です</p>
OTHER_AUTH	<p>Tmaxが提供する認証ライブラリー以外の他のプラグイン・ライブラリーを使用する場合は、DOMAINセクションのSECURITY項目を「OTHER_AUTH」に指定し、NODEセクションのSECURITY_LIB項目に当該ライブラリーのパスを指定します</p>

各ファイルを作成するには、それぞれmkgrp、mkpw、mkac1を使用します。ACLとMANDATORYは同じ3段階セキュリティ設定に該当しますが、以下のような違いがあります。

設定値	内容
ACL	aclファイルに指定されたサービスは指定された1つのグループに属するユーザーのみアクセスでき、aclに指定されていないサービスはTmaxシステムに接続しているすべてのユーザーがアクセスできます
MANDATORY	aclファイルに指定されたサービスは指定された1つのグループに属するユーザーのみアクセスできる点はACLと同じです。ただし、aclファイルに指定されていないサービスは、Tmaxシステムに接続しているすべてのユーザーに対してアクセスを許可しません

- OWNER = string

- サイズ : 63字以内

- ユーザーはTmaxシステムに接続するとき、TPSTART\_T構造体のdompwd項目にこのアカウントのパスワードを入力して接続を申請する必要があります。パスワードが登録されていないか間違っている場合には、Tmaxシステムへの接続に失敗します。このフィールドで定義されたユーザーとパスワードはmkpwユーティリティーで作成できます。

セキュリティの環境設定をした後、パスワード・ファイルはmkpwで作成する必要があります。このファイルは各ユーザーのパスワードを含み、暗号化されているため、パスワードの露出が防止されています。Tmaxセキュリティ・システムを正常に作動するには、Tmaxシステムが設定される前にパスワード・ファイルが作成されている必要があります。

- DOMAIN\_SECの場合、パスワードがtpstart()内にあるTPSTART\_T構造体のdompwdフィールドの値と異なるとそのクライアントのプログラムはサービスを要求できません。tperrnoはTPESECURITY(25)に設定されます。USER\_AUTHの場合、TPSTART\_T構造体のusrpwdフィールドとusernameフィールドはmkpwで作成したパスワード・ファイルに登録されたアカウントとパスワードが同じである必要があります。
  - 第2段階(ユーザー認証)は第1段階(システムアクセス制御)を含むため、dompwdも設定されてドメイン・パスワードと一致する必要があります。

---

#### 注

Tmaxシステムが起動する前にパスワード・ファイルが作成されるか更新される必要があります。

---

- クライアント設定

- 認証関連機能を使用するには、クライアントの環境変数をTMAX\_SEMANTICS="AUTH\_SW:Y"に設定します。

## 3.9.2. その他のセキュリティ設定

TmaxではIPを基準にしてアクセスを許可するクライアントと許可しないクライアントを設定できます。接続を許可または拒否するクライアントの設定ファイルを、それぞれtmax.allow、tmax.denyで作成します。当該ファイルを使ってCLLでTCP/IPクライアントのアクセスに対する許可および拒否を選択します。

## IPベースのアクセス制限機能

- 接続許可クライアントの設定

「\$TMAXDIR/path」ディレクトリーにアクセス許可ファイル(tmax.allow)を作成してアクセスを許可するクライアントのIPを設定します。

– IPv4

```
192.168.1.43
192.168.1.48
```

– IPv6

```
fe80::213:77ff:fe4d:c57d
fe80::213:77ff:fe4d:c570
```

● 接続拒否クライアントの設定

「\$TMAXDIR/path」ディレクトリーにアクセス拒否ファイル(tmax.deny)を作成してアクセスを拒否するクライアントのIPを設定します。

– IPv4

```
192.168.1.35
192.168.1.45
```

– IPv6

```
fe80::213:77ff:fe4d:c50d
fe80::213:77ff:fe4d:c500
```

● ACL(Access Control List)の適用規則

- tmax.allowを先に検索した後、一致するACLが存在するとアクセスを許可します。
- tmax.denyを検索した後、一致するACLが存在したら接続を拒否します。当該ACLに属するクライアントのアクセス試行(TPSTART)時にTPECLOSEエラーが発生します。
- tmax.allowとtmax.denyファイルの両方に存在しなければアクセスを許可します。

● 使用文法

- 最初の文字が「#」の場合、コメントとして処理します。
  - IPまたはNETWORK/NETMASK方式のみ許可します。
- 例)

```
(IPv4)192.168.1.1 または 192.168.1.0/24,  
(IPv6)fe80::213:77ff:fe4d:c57d または fe80::213:77ff:fe4d:c57d/64
```

- 1行当たり1つのACCESS CONTROL LISTのみ許可し、空白やタブは許可されません。
- ALLはすべてのIPを意味する予約語です。

---

#### 注

tmax.allow、tmax.denyファイルはCLL起動時に適用されます。したがって、運用中に変更されたファイルは適用されません。

---

### 3.9.3. サードパーティー・セキュリティ設定

Tmaxではライブラリーをロードすることにより、Tmaxが提供するセキュリティ・システム以外のシステムを使用できるようにサポートします。

ユーザーはTmaxが提供する認証および暗号化関連APIを実装する必要があります。関連説明については『Tmax リファレンスガイド』のセキュリティ関連関数を参照してください。

認証セキュリティと暗号化セキュリティを区分して設定する必要があります。関連するDOMAINセクションおよびNODEセクションの設定は以下のとおりです。

#### DOMAINセクション

```
* DOMAIN  
DOMAIN Name      [SECURITY = "OTHER_AUTH", ]  
                  [CRYPT = Y]
```

- SECURITY = "OTHER\_AUTH"

- サードパーティー認証システムを使用するには、DOMAINセクションのSECURITY項目を"OTHER\_AUTH"に指定します。

またNODEセクションでSECURITY\_LIB項目を設定する必要があります。

- CRYPT = Y

- CRYPT = Yに設定し、NODEセクションのCRYPT\_LIB項目を設定します。

## NODEセクション

```
* NODE
NODE Name      [SECURITY_LIB = literal,]
                [CRYPT_LIB = literal]
```

- SECURITY\_LIB = literal
  - サードパーティー認証システムを使用するには、当該ライブラリーのパスおよびファイル名を SECURITY\_LIBに指定します。DOMAINセクションのSECURITY項目を"OTHER\_AUTH"に設定していない場合は、環境設定ファイルのcflは正常に実行されません。
- CRYPT\_LIB = literal
  - サードパーティー暗号化システムを使用するには、当該ライブラリーのパスおよびファイル名を CRYPT\_LIBに指定します。DOMAINセクションのCRYPT項目をYに設定していない場合は、環境設定ファイルのcflは正常に実行されません。

## 3.10. マルチドメインの環境設定

Tmaxではマルチドメイン環境でドメイン間のトランザクション処理機能を提供します。グローバル・トランザクション処理がさらに拡張された形で提供され、ドメイン間で同じアプリケーションを使ってデータ値によるルーティングをサポートし、開発への負担を軽減します。

マルチドメイン環境のためにはドメインのGATEWAYセクションにゲートウェイ情報を登録し、ROUTINGセクションとSERVICEセクションにはルーティング情報を登録する必要があります。各セクションの項目についての詳細は、[「3.2. 基本環境設定」](#)を参照してください。

### 3.10.1. DOMAINセクション

DOMAINセクションには基本的な内容(Shared Memory Key、Port number、MINCLH、MAXUSER)が登録されます。分散トランザクションと関連して2相コミット(Two Phase Commit)ができるかどうかとトランザクションのタイムアウト時間を設定できます。

以下は、マルチドメイン環境設定のためのDOMAINセクションの例です。

```
*DOMAIN
site1      SHMKEY = 79990, MAXUSER = 100, MINCLH = 1,
            MAXCLH = 3, TPORTNO = 8888,
            CMTRET = Y, BLOCKTIME = 60,
            DOMAINID = 1
```



## 3.10.2. NODEセクション

NODEセクションには各ドメインに属する同じノードの情報をすべて登録します。

以下は、マルチドメイン環境設定のためのNODEセクションの例です。

```
*NODE
tmax1      TMAXDIR = "/user3/tmax",
           APPDIR  = "/user3/tmax/appbin",
           PATHDIR = "/user3/tmax/path",
           TLOGDIR = "/user3/tmax/log/tlog",
           ULOGDIR = "/user3/tmax/log/ulog",
           SLOGDIR = "/user3/tmax/log/slog"
```

## 3.10.3. SVRGROUPセクション

マルチドメイン環境で各SVRGROUPセクションには、データベースを利用できるかどうかによってグループを区分して登録します。ドメイン間のグローバルトランザクション処理のためには、必ずXAグループでサーバーとサービスがまとめられている必要があります。

以下は、マルチドメイン環境設定のためのSVRGROUPセクションの例です。

```
*SVRGROUP
NXAGRP     NODENAME = tmax1
XAGRP      NODENAME = tmax1, DBNAME = ORACLE,
           OPENINFO = "Oracle_XA+Acc=P/scott/tiger+SesTm=80+logdir=.",
           TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5
```

## 3.10.4. SERVERセクション

マルチドメイン環境で各SERVERセクションには基本的な内容を登録します。

以下は、マルチドメイン環境設定のためのSERVERセクションの例です。

```
*SERVER
SVG_X      SVGNAME = XAGRP, MIN = 1, MAX = 2,
           CLOPT = "-o $(SVR).$(DATE).log -- -u scott -p tiger"
SVG_NX     SVGNAME = NXAGRP, MIN = 1, MAX = 2,
           CLOPT = "-o $(SVR).$(DATE).log"
```

## 3.10.5. SERVICEセクション

マルチドメイン環境で各SERVICEセクションには基本的な内容とルーティングが必要な場合、ROUTINGセクションのルーティング名を指定します。

以下は、マルチドメイン環境設定のためのSERVICEセクションの例です。

```
*SERVICE
SVC_X          SVRNAME = SVG_X, ROUTING = XRID
SVC_NX         SVRNAME = SVG_NX, ROUTING = NXRID
```

### 3.10.6. GATEWAYセクション

マルチドメイン環境でルーティングが必要な場合、各GATEWAYセクションを必ず登録し、必要な情報を記録します。RGWADDRとRGWPORTNOは必ず相手側のドメインサーバーのIPアドレスとポート番号である必要があります。ノード名はゲートウェイプロセスが実行されるサーバー名です。

以下は、マルチドメイン環境設定のためのGATEWAYセクションの例です。

```
*GATEWAY
DOMAIN_GW1     GWTYPE = TMAX, PORTNO = 5000,
               RGWADDR = "192.168.63.133",
               RGWPORTNO = 5000,
               NODENAME = tmax1
```

### 3.10.7. ROUTINGセクション

マルチドメイン環境でルーティングが必要な場合、必ず登録します。ルーティング名、フィールド名、SUBTYPE(バッファータイプ)、およびルーティング範囲(RANGES)の登録が必須です。

## マルチドメインのルーティング環境ファイルの例

<Domain Siteの環境ファイルの例>

```
*DOMAIN
sitel          SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 3,
               TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 60,
               DOMAINID = 1

*NODE
tmax1          TMAXDIR = "/user3/tmax",
               APPDIR = "/user3/tmax/appbin",
               PATHDIR = "/user3/tmax/path",
               TLOGDIR = "/user3/tmax/log/tlog",
               ULOGDIR = "/user3/tmax/log/ulog",
               SLOGDIR = "/user3/tmax/log/slog"

*SVRGROUP
NXAGRP         NODENAME = tmax1
```

```

XAGRP          NODENAME = tmax1, DBNAME = ORACLE,
                OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 80 + logdir = .
" ,

                TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5

*SERVER
SVG_X          SVGNAME = XAGRP, MIN = 1, MAX = 2,
                CLOPT = "-o $(SVR).$(DATE).log -- -u scott -p tiger"

SVG_NX         SVGNAME = NXAGRP, MIN = 1, MAX = 2,
                CLOPT = "-o $(SVR).$(DATE).log"

*SERVICE
SVC_X          SVRNAME = SVG_X, ROUTING = XRID
SVC_NX         SVRNAME = SVG_NX, ROUTING = NXRID

*GATEWAY
DOMAIN_GW1     GWTYPE = TMAX, PORTNO = 5000, RGWADDR = "192.168.63.133"
                RGWPORTNO = 5000, NODENAME = tmax1

*ROUTING
XRID           FIELD = FIELD/ROUTING_ID,
                RANGES = "'6400':XAGRP, '6471':DOMAIN_GW1"

NXRID          FIELD = FIELD/ROUTING_ID,
                RANGES = "'6400':NXAGRP, '6471':DOMAIN_GW1"

*DOMAIN
site2          SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 3,
                TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 60,
                DOMAINID = 2

*NODE
tmax2          TMAXDIR = "/user3/tmax",
                APPDIR = "/user3/tmax/appbin",
                PATHDIR = "/user3/tmax/path",
                TLOGDIR = "/user3/tmax/log/tlog",
                ULOGDIR = "/user3/tmax/log/ulog",
                SLOGDIR = "/user3/tmax/log/slog"

*SVRGROUP
NXAGRP         NODENAME = tmax2

XAGRP          NODENAME = tmax2, DBNAME = ORACLE,
                OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 80 + logdir = . "
,

                TMSNAME = XAGRP_tms, MINTMS = 2, MAXTMS = 5

*SERVER

```

```

SVG_X          SVGNAME = XAGRP, MIN = 1, MAX = 2,
                CLOPT = "-o $(SVR).$(DATE).log -- -u scott -p tiger"

SVG_NX         SVGNAME = NXAGRP, MIN = 1, MAX = 2,
                CLOPT = "-o $(SVR).$(DATE).log"

*SERVICE
SVC_X          SVRNAME = SVG_X, ROUTING = XRID
SVC_NX         SVRNAME = SVG_NX, ROUTING = NXRID

*GATEWAY
DOMAIN_GW2     GWTYPE = TMAX, PORTNO = 5000, RGWADDR = 192.168.63.132",
                RGWPORTNO = 5000, NODENAME = tmax2

*ROUTING
XRID           FIELD = FIELD/ROUTING_ID,
                RANGES = "'6471':XAGRP, '6400':DOMAIN_GW2"
NXRID          FIELD = FIELD/ROUTING_ID,
                RANGES = "'6471':NXAGRP, '6400':DOMAIN_GW2"

```

#### <Tmaxの複合機能を利用した環境ファイルの例>

```

*DOMAIN
sitel          SHMKEY = 79990, MAXUSER = 100, MINCLH = 1, MAXCLH = 5,
                TPORTNO = 8888, CMTRET = Y, BLOCKTIME = 30

*NODE
tmax1          TMAXDIR = "/tmax/tmax",
                APPDIR = "/tmax/tmax/appbin",
                PATHDIR = "/tmax/tmax/path",
                TLOGDIR = "/tmax/tmax/log/tlog",
                ULOGDIR = "/tmax/tmax/log/ulog",
                SLOGDIR = "/tmax/tmax/log/slog"

tmax2          TMAXDIR = "/tmax/tmax",
                APPDIR = "/tmax/tmax/appbin",
                PATHDIR = "/tmax/tmax/path",
                TLOGDIR = "/tmax/tmax/log/tlog",
                ULOGDIR = "/tmax/tmax/log/ulog",
                SLOGDIR = "/tmax/tmax/log/slog"

*SVRGROUP
SVGtmax1       NODENAME = tmax2, DBNAME = ORACLE,
                COUSIN = "SVGtmax1",
                OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 60+DbgFl=0x01",
                TMSNAME = svg1_tms, MINTMS = 2, MAXTMS = 5

```

```

SVGtmax2      NODENAME = tmax1, DBNAME = ORACLE,
              OPENINFO = "Oracle_XA+Acc=P/scott/tiger + SesTm = 60+DbgFl=0x01",
              TMSNAME = svg1_tms, MINTMS = 2, MAXTMS = 5

SVGtmax2NX    NODENAME = tmax1
              SVGRQ1      NODENAME = tmax1,
              SVGTYPE = RQMGR, CPC = 8,
              COUSIN = "SVGRQ1", LOAD = -1

*RQ
rq1           SVGNAME = SVGRQ, BOOT = COLD, FILEPATH = "/tmp/rq1"
              QSIZE = 24, FSYNC = Y, BUFFERING = N

*SERVER
svr1          SVGNAME = SVGtmax1, MIN = 1, MAX = 5

svr2          SVGNAME = SVGtmax2, MIN = 1, MAX = 5
svr3          SVGNAME = SVGtmax2NX, MIN = 1, MAX = 5

*SERVICE
svc1          SVRNAME = svr1, ROUTING = rout1
svc2          SVRNAME = svr2
svc3          SVRNAME = svr3
svc4          SVRNAME = GW2
              #Gateway is directly defined on the server name.

*GATEWAY
GW1           GWTYPE = TMAX, PORTNO = 5001,NODENAME = "tmax1",
              RGWADDR = "GW1_MAIN", RGWPORTNO = 5001
GW2           GWTYPE = TMAX, PORTNO = 5002,NODENAME = "tmax2",
              RGWADDR = "GW2_MAIN", RGWPORTNO = 5001
GW3           GWTYPE = TMAX, PORTNO = 5003,NODENAME = "tmax2",
              RGWADDR = "GW3_MAIN", RGWPORTNO = 5001

*ROUTING
rout          FIELD = FIELD/PLACE_CD,
              RANGES = "'00000':SVGtmax1, 'A0001'-'A0006':GW2,
              'A0007'-'A4000':GW3"

```

## 3.11. Tmax環境ファイルのコンパイル

一般的に、1つのプログラムを作成するプロセスは複数の段階を必要とします。まず、ソース・プログラムを作成してコンパイルし、エラーを修正した後、正しい実行ファイルを作成します。Tmax環境ファイルの作成方法も実際のプログラムのそれと同じです。Tmax環境ファイルをコンパイルし、エラーの発生しないTmax環境ファイルを正しく作成できます。

Tmaxシステムは起動するとき、Tmax環境ファイルに基づいて環境を設定するため、エラーが存在する環境ファイルであればTmaxシステム動作中に予期しないエラーが発生することがあります。Tmaxシステムは正しい環境ファイル無しには起動できません。そのため、Tmax環境ファイルのコンパイル作業によってTmaxが正常に動作できるようにTmax環境ファイルを正しく作成します。

コンパイルはcflコマンドによって実行されます。詳細については『*Tmax リファレンスガイド*』の「2.1. cfl」を参照してください。

```
$ cfl [-i テキストTmax環境ファイル名] [-o バイナリTmax環境ファイル名]
      [-h] [-V] [-n node_name] [-A] [-v num] [-I] [-r]
```

オプション	説明
[-i テキストTmax環境ファイル名]	コンパイル対象となるソース設定ファイルであるテキスト形式のTmax環境ファイル名を設定します。ディレクトリはユーザーが指定することが可能であり、指定しない場合のデフォルト環境設定ディレクトリは、 <b>\$TMAXDIR/config</b> です。ソース設定ファイルが見つからなかった場合は、警告メッセージを出力します
[-o バイナリTmax環境ファイル名]	コンパイルの結果であるバイナリTmax環境ファイル名を設定します。パスと一緒に指定可能であり、パスが指定されていない場合は、 <b>\$TMAXDIR/config</b> にバイナリTmax環境ファイルが作成されます。このオプションが省略されると、ファイル名は <b>tmconfig</b> と作成されます
[-h]	コマンドのヘルプ・オプションです
[-V]	実行ファイルのバージョンを確認することができます
[-n node_name]	マルチノード環境で、特定ノードの環境ファイルをコンパイルする場合に使用します。マルチノード環境で他のノードを管理するためには、 <b>racd</b> を設定する必要があります
[-A]	サービス・アクセス制御(第3段階セキュリティ)を使用する場合のみ有効です。  1つのドメイン内に存在するACLサービスへのアクセス権限は、すべてのノードで同様に適用される必要があるため、現行ノードの <b>\$TMAXDIR/config/group</b> 、 <b>acl</b> 、 <b>user</b> ファイルを、同じドメインに属している別のノードの <b>\$TMAXDIR/config</b> にデプロイするためのオプションです。このオプションを使用して環境ファイルをコンパイルする場合は、事前に <b>mkgrp</b> 、 <b>mkpw</b> 、 <b>mkacl</b> を使用して <b>group</b> 、 <b>acl</b> 、 <b>user</b> ファイルを作成しておく必要があります。  作成されたファイルは、1つのドメイン内に存在するすべてのノードが使用することになるため、ファイルの作成時にはドメインに属しているノードの全ユーザーを考慮しなければなりません。 <b>group_name</b> 、 <b>group_id</b> 、 <b>user_name</b> 、 <b>user_id</b> は、1つのドメイン内で一意である必要があります。パスワード・ファイルはコピーできないため、ノード別に作成するか、別途コピーする必要があります
[-v num]	numには0と1を指定できます(デフォルト値: 1)

オプション	説明
	<ul style="list-style-type: none"> <li>- 0: マルチノードを構成しているノードがすべて同じタイプのマシンであり、管理者がノード別に環境ファイルを管理する場合は0に設定します。コンパイル済みのバイナリ・ファイルは、それぞれのマシンにコピーして管理します</li> <li>- 1: 環境ファイルはracdによって自動的にコンパイルされます</li> </ul>
[-l]	cflの実行時に、環境ファイルDOMAINセクションのSHMKEY項目に設定した値が現在使用中の場合、その値のUIDを比較して一致しなければエラーメッセージが出力されます。このオプションは、共有メモリーの使用の有無やUIDが一致するかどうかをチェックしない場合に使用します
[-r]	<p>cflの実行段階で、ulimit -nを使って照会する際に出力される現行システムの使用可能なFDの最大数を事前にチェックしてユーザーに通知します。CLH 1つ当たりオープンできるFDの最大数を予め計算してチェックします。</p> <p>Tmaxシステムで使用するFD値がシステムで使用可能なFDより大きく設定されると、以下のようなエラーが発生します</p> <pre>(E) CFL9990 Current Tmax configuration contains more servers or nodes than current system can support[CFL5056]</pre>
[-a Tmax環境ファイル名]	<p>動的に追加するTmax環境ファイル名を指定します。サーバーを動的に追加する際、cflを使用してバイナリ環境ファイルを作成するときは、必ず[-a]オプションを使用します。</p> <p>このオプションを使わずに動的に追加した場合は、(E) ADM2048エラーが発生します。詳細については「<a href="#">5.5.6. cfgadd(ca)</a>」を参照してください</p>
[-Z]	<p>現在のcflでは、MAXSACALL/MAXCACALL 1024以下で制限されていますが、この制限を解除するオプションです</p> <p>例)</p> <pre>- cfl -Z -i sample.m</pre>

## 3.12. サービス・テーブルの作成

Tmaxシステムでは環境ファイルのSERVICEセクションに登録されたサービスのみ処理できます。ただし、1つのサーバー・プロセスが複数のサービスを提供することができるため、ユーザーはサーバー・プロセスで提供するサービスの種類をTmaxシステムに提供する必要があります。そのため、環境ファイルからサービス・テーブルを作成する必要であり、これはサーバー・プログラムがコンパイルされるとき、必ず一緒にコンパイルされなければなりません。

サービス・テーブルは、サーバーごとに提供されるサービス名が記述されたファイルであり、Tmax環境ファイルに登録されたサーバーとサービスを参照して作成されます。サーバー・プログラムを作成するとき、サービス・テーブルと一緒にコンパイルしないと、サービス要求がサーバーに渡されてもサーバー内でサービス・ルーチンの場所が把握できません。

サービス・テーブルはバイナリ環境設定ファイルを参照し、gstコマンドを使って作成されます。詳細については『Tmax リファレンスガイド』の「2.3. gst」を参照してください。

```
$ gst [-f バイナリTmax環境ファイル名 | -v server_name | -h | -n node_name | -V]
```

項目	説明
[-f バイナリTmax環境ファイル名]	参照するバイナリTmax環境ファイル名を指定します。 <b>cfl</b> コマンドの結果であり、 <b>tmboot</b> と <b>tmdown</b> によっても参照されるファイルです。  パスと一緒に指定することができます。省略する場合、デフォルト値として <b>\$TMAXDIR/config</b> に指定されたディレクトリー配下の <b>config</b> ディレクトリーで <b>tmconfig</b> を参照します
[-v server_name]	サーバー名に該当するサービス・テーブルを設定します
[-h]	ヘルプを照会します
[-n node_name]	マルチノード環境で他のノードのサーバー・プロセスのサービス・テーブルを現行ノードの <b>\$TMAXDIR/svct</b> に作成します
[-V]	実行ファイルのバージョンを確認できます

## 3.13. サービス・タイムアウトの監視

Tmaxシステムでは、タイムアウトが経過したサービスが実際にタイムアウトされていない場合、周期的に監視してタイムアウトを強制実行する機能を提供します。

### 3.13.1. tmapm

Tmaxシステムでサービス・タイムアウトはシグアラームを使用して動作します。tmapmは、サービスのシグアラームを再定義したり、シグアラームが使用できないサービスにサービス・タイムアウトを提供するサーバーです。

tmapmはUSCサーバーであり、環境設定ファイルのSERVERセクションに以下のように設定します。

#### ● 使用方法

```
*SERVER
tmapm      CLOPT = [ -i sec ]
              [-r sec ]
```



```
[ -c command ]  
[ -l [ 0 | 1 | 2 ] ]
```

項目	説明
[ -i <i>sec</i> ]	サービス・タイムアウトをチェックする周期(秒)です。周期が短いほど、システムの負荷が増加します。小数点も設定できます
[ -r <i>sec</i> ]	サービス・タイム以降コマンドを実行するまでの制限時間(秒)です。整数型で設定します
[ -c <i>command</i> ]	サービスが実行時間(SCVTIME + コマンド実行制限時間)までRUNNING状態である場合に実行するコマンドです。シェル・コマンドやスクリプトを使用できます
[ -l [ 0   1   2 ] ]	ユーザー・ログのレベルです。デフォルト値は0であり、数値が高いほど、より詳細な情報を出力します

[ -c ] オプションで設定したコマンドを実行する場合、以下のようにパラメータを渡します。

User Command	PID	SVRNAME	SVCNAME[Elapse Time]
kill.sh	9659	svr2	TOUPPER[5]

- 例

以下は、tmapmの設定例です。

```
*SERVER  
tmapm    SVGNAME = svg,  
          SVRTYPE = UCS,  
          CLOPT = "-o ulog -- -i 5 -r 1 -c kill.sh -l 1"
```



## 第4章 起動と終了

本章では多様な環境におけるTmaxシステムの起動と終了方法について説明します。

### 4.1. Tmaxの起動

Tmaxを実行するための環境が用意されたら、管理者はTmaxシステムを起動(boot)させることができます。Tmaxシステムが起動するとシステム・ソフトウェアの特性上、致命的な原因(ハードウェア障害、運用体制上のエラーなど外部要因)でなければ異常終了しません。ただし、管理の面では環境ファイルの再作成、外部要素によるシステム・ダウンなどの理由でシステムを再起動させる必要がある場合もあります。

Tmaxシステムを起動するには、基本環境設定が正しく行われている必要があります。システムを起動する前に以下の事項を確認してください。

- バイナリーTmax環境ファイルが存在するかどうか
- 環境ファイルのNODEセクションのTMAXDIR項目に設定されたディレクトリーにTmax実行プログラム(TMM、CLL、CLH、TMSなど)が存在するかどうか
- 環境ファイルのNODEセクションのAPPDIR項目に設定されたディレクトリーにSERVERセクションに登録されたサーバー・プログラムが存在するかどうか
- 中央管理を行うためのracdが各ノードで動作しているかどうか

一般的にTmaxシステムは複数ノードで構成され、中央管理を必要とします。そのためにTmaxシステムではracd(remote access control daemon)を使用します。各ノードに予めracdをインストールすることで、1つのノードでTmaxシステム全体を管理することができます。つまり、環境ファイルのコンパイル、Tmaxシステムの起動および終了、動的環境ファイルの変更などの動作を1回のコマンドで可能にします。

環境変数にTMAX\_RAC\_PORTが定義されていないとracdは起動しません。中央管理に使用されたracdのポート番号を定義し、「racd -k」を使用してracdを起動するとTmax環境ファイルでの環境変数を参照せずにシステムを起動できます。

#### 4.1.1. racd

マルチノードで分散された環境で各ノードを中央管理するためのコマンドです。racdコマンドは、複数ノードを1つのドメインでTmaxシステムを構築した場合、1つのノードでTmaxシステムを集中管理するために、それぞれのノードであらかじめ起動されるデーモン・プロセスです。Tmaxシステムを管理するノードでは、racdコマンドを実行しなくても構いません。

racdコマンドは、ドメイン内の1ノードでtmadminコマンドを使用して全体ノードを管理したり、またはcflコマンドで環境ファイルをドメイン内のすべてのノードに同一内容で適用可能にします。

IPバージョンがIPv6またはInfiniBandのSDP(Socket Direct Protocol)である場合、環境ファイルに下記を指定する必要があります。

```
SYSTEM_PROTOCOL="IPV6"
SYSTEM_PROTOCOL="SDP"
```

racdを-kオプションと一緒に使用する場合は、環境変数に下記を指定する必要があります。

```
TMAX_RAC_IPV6="IPV6"
TMAX_RAC_IPV6="SDP"
```

以下は、racdコマンドの使用方法についての説明です。

#### ● 使用方法

```
$ racd [-d] [-f バイナリTmax環境ファイル名] [-h] [-k]
        [-i filename] [-l Label] [-P umask] [-V]
```

項目	説明
[-d]	デバッグ・モードでracdコマンドを実行する場合に設定します
[-f バイナリTmax環境ファイル名]	参照するバイナリTmax環境ファイル名を設定します。cflコマンドの結果で、tmbootコマンドとtmdownコマンドでも参照されるファイルです。パスと一緒に指定でき、省略した場合、デフォルトとしてTMAXDIRと指定されたディレクトリ下位のconfigディレクトリでtmconfigを参照します
[-h]	コマンドのヘルプ・オプションです
[-k]	バイナリTmax環境ファイルの参照可否を設定します。  オプションを指定した場合、バイナリTmax環境ファイルを参照しません。一般的に、racdコマンドはこのオプションを使用して実行します(passive listen mode)。  IPバージョンがIPv6またはInfiniBandのSDP(Socket Direct Protocol)である場合、このオプションを使用すると、環境ファイルを参照しないため、環境変数に「TMAX_RAC_IPV6=IPV6」または「TMAX_RAC_IPV6=SDP」を指定する必要があります
[-i filename]	1つの物理的なマシンで複数の論理的ノードを定義する場合に使用します。  論理ノードのNODETYPEがSHM_RACDの場合、各論理ノードあたり1つずつのracdを実行する必要があります。RACPORTはすべて異なる必要があります。racdコマンドを実行前に、環境変数のTMAX_RAC_PORT変数を設定しなければなりません。しかし、論理ノードの数が多い場合はすべて変えることができないため、TMAXHOME、TMAXDIR、TMAX_RAC_PORTが定義されたファイル名を指定できます。詳細内容は例を参照してください

項目	説明
[ -l <i>Label</i> ]	Labelは、ファイル内に登録された環境情報の記述子です。2個以上のシステム情報を1つのファイルに登録する場合、それぞれのシステムを区別できる値です
[ -P <i>umask</i> ]	racdコマンドを使用して起動するプロセスの場合、umaskをユーザーが設定して、必要な権限のファイルを実行できるようにします
[ -V ]	実行ファイルのバージョンを確認できます

- 適用環境

Tmaxがインストールされている運用システム環境で使用できます。

- 例

– 以下は、racdコマンドに[-P]オプションを指定し、umaskを設定する例です。

<tmax.racd>

```
[tmaxs1]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/tmax32
TMAX_RAC_PORT = 3333
[NODE1]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/proj1
TMAX_RAC_PORT = 4335
[NODE2]
TMAXHOME = /user2/starbj81/tmax32
TMAXDIR = /user2/starbj81/proj2
TMAX_RAC_PORT = 4337
```

## 1. 環境ファイルを作成します。

```
*DOMAIN
tmax1      SHMKEY = @SHMEMKY@, MINCLH = 1, MAXCLH = 3,
           TPORTNO = @TPORTNO@, BLOCKTIME = 30, RACPORT = 3255

*NODE
@HOSTNAME@ TMAXDIR = "@TMAXDIR@",
           APPDIR = "@TMAXDIR@/appbin",
           PATHDIR = "@TMAXDIR@/path",

@RMTNAME@  TMAXDIR = "@RMTDIR@",
           APPDIR = "@RMTDIR@/appbin",
           PATHDIR = "@RMTDIR@/path",

*SVRGROUP
```

```
svg1      NODENAME = "@HOSTNAME@", COUSIN = "svg2"
svg2      NODENAME = "@RMTNAME@"

*SERVER
svr2      SVGNAME = svg1, CLOPT = "-o $(SVR).out -e $(SVR).err"

*SERVICE
TOUPPER   SVRNAME = svr2
```

2. リモートノードのracdを起動します。

```
$ export TMAX_RAC_PORT = 3255
$ racd -k -P 055
```

3. HOSTノードで全体Tmaxを起動します。(tmboot)

4. RMTノードのULOGDIRにsvr2.outのファイル権限を確認します。

– 以下は、NODE1のRACDを実行させる例です。

```
$ racd -k -i tmax.racd -l NODE1
```

– 以下は、環境ファイルを参照せずに、他のコマンド(tmboot)で使用した情報のみを利用する例です。

```
$ racd -k
```

---

## 参考

1. tmboot、tmdownコマンドについての詳細は、「[4.1.2. tmboot](#)」、「[4.2.1. tmdown](#)」を参照してください。
  2. 論理ノード設定についての詳細は、「[3.2 NODEセクション](#)」のHOSTNAME項目の説明を参照してください。
- 

## 4.1.2. tmboot

Tmaxシステムの全体または一部分を実行させるコマンドです。Tmax環境ファイルをベースにシステムを実行します。オプションなしで実行した場合、あるいは[-f]オプションのみを使用した場合、すべてのTmax管理プロセスとTmax環境ファイルのSERVERセクションに登録されているすべてのサーバー・プロセスを実行させます。NODEセクションに登録されているすべてのノードでTmax管理プロセスのTMM、CLL、CLHプロセスが順番に実行されます。

SVRGROUPセクションにOPENINFO項目が登録されているサーバー・グループが存在する場合、各サーバー・モジュール別に**TMSNANE**と**MINTMS**項目を参照し、TMSプロセスが実行されます。Tmax管理プロセスは、ノード別に定義されたTMAXDIRディレクトリー下位のbinディレクトリーで実行されます。

Tmax管理プロセスの生成後、SERVERセクションのすべての応用サーバー・プロセスが実行されます。応用サーバー・プロセスは、SERVERセクションに登録されている順番に実行されます。tmbootコマンドは実行されたサーバー・プロセスの初期化を実行後に、次のサーバー・プロセスを実行させます。プロセスの初期化は、tpsvrinit()を使用して進めます。すべてのサーバー・プロセスが初期化を終えるまでは、次のプロセスが実行されません。tmbootコマンドは、各応用サーバー・プロセスをそれらのMIN項目に定義された個数の分だけ実行させます。MIN項目が定義されていない場合、デフォルト値は1個です。

tmbootコマンドは、SERVERセクションのサーバーに対して、**CLOPT**、**MIN**、**MAX**項目の値を使用します。サーバー・プロセスの実行時、tmbootコマンドによって使用されるサーバーのbootパラメータです。サーバーの残りの項目は、サーバーの実行後にシステムによって実行されるruntimeパラメータです。設定情報は、ソース設定ファイルのSERVERセクションを参照してください。

すべての応用サーバー・プロセスは、動作するノードに定義されているAPPDIRディレクトリーで実行されます。

#### ● 使用方法

```
$tmboot [-A] [-b] [-c] [-f バイナリTmax環境ファイル名]
        [-g servergroup_name [-a]] [-h] [-i] [-V]
        [-n node_name] [ -o clopt_string ] [-q RQ svg_name]
        [-s server_name [-k count] [-a]]
        [-S server_name [-a]] [-t TMS_name[-k all]]
        [-B TRB_node] [-T] [-w] [-d boot_time] [-D]
        [-e clh | cas | tlm ] [-k] [-R]
```

項目	説明
[-A]	Tmax環境ファイルの <b>SERVER</b> セクションに定義されたすべての応用サーバー・プロセスを実行します
[-b]	バックアップで指定されたサーバー・プロセスを任意で起動させる際に使用します
[-c]	CLHプロセスをもう1つ実行させるオプションです。  現在動作中のCLHプロセス数がTmax環境ファイルに定義された <b>MAXCLH</b> 値を超えない範囲内でのみ使用可能です
[-f Tmax ]	参照するバイナリTmax環境ファイル(ソース設定ファイルをcflでコンパイルした結果)をパスと一緒に指定する必要があります。  [-f]オプションが省略された場合、デフォルトとしてTMAXDIRディレクトリー下位のconfigディレクトリーで <b>tmconfig</b> ファイルを参照します

項目	説明
[ -g <i>servergroup_name</i> [-a] ]	<p>指定されたサーバー・グループに存在するサーバー・プロセスを実行するオプションです。</p> <p>使用されるサーバー・グループ名は、Tmax環境ファイル内の<b>SVRGROUP</b>セクションに登録します。[-a]オプションと一緒に使用する場合、サーバーの起動をTMBOOTプロセスではなく、TMMプロセスに委任します</p>
[ -h ]	コマンドのヘルプ・オプションです
[ -i ]	特定のサーバーがmaxまで達しても、起動を停止せずに起動されていない残りのサーバーを起動する場合に使用します
[ -V ]	実行ファイルのバージョンを確認できます
[ -n <i>node_name</i> ]	<p>指定されたノードに存在するサーバー・プロセスを実行させるオプションです。ノード名はTmax環境ファイル内の<b>NODE</b>セクションに事前に登録されている必要があります</p>
[ -o <i>clopt_string</i> ]	CLOPT stringを追加します
[ -q <i>RQ svg_name</i> ]	RQSを始動させます
[ -s <i>server_name</i> [-k <i>count</i> ] [-a] ]	<p>指定されたサーバー・プロセスのみを実行させます。</p> <p>使用されるサーバー・プロセス名は、Tmax環境ファイル内の<b>SVRGROUP</b>セクションに事前に定義されている必要があります。</p> <p>[-k]オプションと一緒に使用してサーバー・プロセス数を指定できます。サーバー・プロセス数は、現在実行されている数を含み、SERVERセクションのMAX項目に定義された数を超えてはいけません。</p> <p>[-k]オプションを省略した場合、該当サーバー・プロセスは1つのみ実行されます。[-a]オプションと一緒に使用する場合、サーバーの起動をTMBOOTプロセスではなく、TMMプロセスに委任します</p>
[ -S <i>server_name</i> ]	<p>指定されたサーバー・プロセスをMINの数だけ実行します。</p> <p>[-a]オプションと一緒に使用する場合、サーバーの起動をTMBOOTプロセスではなく、TMMプロセスに委任します</p>
[ -t <i>TMS_name</i> [-k <i>all</i> ] ]	<p>指定されたTMSプロセスをもう1つ実行します。</p> <p>Tmax環境ファイルに定義された<b>MAXTMS</b>値を超えない場合にのみ可能です。</p> <p>[-k <i>all</i>]は、トランザクション・リカバリー機能を使用する場合に必要なオプションです。グループ全体を終了後に起動する場合、リカバリーが実行されるため、リカバリーはTMSグループ単位で可能です。特定名をもつTMS全体を起動/終了する場合、このオプションを使用できます</p>



項目	説明
[ -B <i>TRB node</i> ]	TRBノードを起動させます
[ -T ]	Tmaxシステムのプロセス(TMM、CLL、CLH、TMS)だけを実行させます
[ -w ]	<p>オプションがない場合は、登録されているサーバー・プロセスを同時に起動させます。この場合、OSによっては同時にリソースを作成できず、サーバー・プロセスが正常に起動しない場合が発生します。この問題を解決するために、プロセスを1つずつ起動させ、正常に起動させるオプションです。</p> <ul style="list-style-type: none"> <li>– サーバー・プロセスがTMMに接続時のLOCK使用条件(LOCK   NOLOCK)</li> <li>– サーバー・プロセスが起動時のWAIT条件(NO-WAIT   FINITE-WAIT) <ul style="list-style-type: none"> <li>• 「-d -1000000 (1sec)」と同一効果を持ちます</li> <li>• [-d]オプションがない場合にのみ意味があります</li> <li>• [-d]オプションが使用された場合、[-w]オプションは無視されます</li> </ul> </li> </ul>
[ -d <i>boot_time</i> ]	<p>一度に多くのサーバー・プロセスを起動させた場合、CLHでの登録要求が集中します。その際に生じる問題を解決するために、サーバー・プロセスの起動にかかる総時間を指定して、登録間隔を調節できます</p> <p>(デフォルト値: LOCK、NO-WAIT、単位: usec)</p> <ul style="list-style-type: none"> <li>– サーバー・プロセスがTMMに接続時のLOCK使用条件(LOCK   NOLOCK)</li> <li>– サーバー・プロセス起動時のWAIT条件(NO-WAIT   FINITE-WAIT) <ul style="list-style-type: none"> <li>• -d val &lt; 0 : LOCK,  VAL  FINITE-WAIT *</li> <li>• -d val = 0 : NO-LOCK, NO-WAIT</li> <li>• -d val &gt; 0 : NO-LOCK,  VAL  FINITE-WAIT</li> <li>• 基本的に[-d]オプションのvalが0でない場合、絶対値( VAL )を使用し、単位はusecです</li> </ul> </li> <li>– FINITE-WAITの場合、 VAL 値は各プロセスごとに最大のWAIT時間です (全体プロセスの総WAIT時間ではない)</li> <li>– サーバー・プロセスがシグナルを与えた場合、WAITは解除されます。つまり、 VAL 時間になっていなくても、シグナルを受けると次のプロセスの起動を行います。VALが負数の場合はLOCKを使用します。このオプションが使用された場合、[-w]オプションは無視されます</li> </ul>

項目	説明
[ -D ]	[ -d ]オプションとほぼ似ていますが、finite-waitのときにシグナルが来ても VAL までは必ず待機します
[ -e clh   cas   tlm ]	<p>Tmaxエンジン・プロセスのうち、CLH、CAS、TLMを起動するためのオプションです。tmbootの実行時に問題が発生した場合や、killによってエンジン・プロセスが異常終了した場合に使用します。</p> <p>tmdownコマンドでは、このオプションを提供していません</p> <ul style="list-style-type: none"> <li>– clh : CLHを起動します</li> <li>– cas : CASを起動します</li> <li>– tlm: TLMを起動します</li> </ul>
[ -R ]	リモート・シェル(rshまたはremsh)を使用して、リモート・ノードのTmaxシステムを起動させる場合、[-R]オプションを使用します

## ● 適用環境

Tmaxシステムがインストールされている運用システム環境で使用できます。

## ● 例

- 以下は、TMAXDIRディレクトリー下位のconfigディレクトリーにあるtmconfigファイルを参照し、Tmaxプロセスと応用サーバー・プロセスをすべて実行する例です。

```
$ tmboot
```

- 以下は、tms\_nameで生成されたすべてのTMSが起動する例です。

```
$ tmboot -t tms_name -k all
```

- 以下は、サーバー・グループの設定オプションを使用した例です。互いに異なるサーバー・グループでtms\_nameを同一に使用する場合、以下のように、[-g]オプションで該当TMSが属しているサーバー・グループ名を指定します。そうでない場合、該当TMS名をもつ最初のサーバー・グループのTMSが起動します。

```
$ tmboot -t tms_name -k all -g svgname
```

- 以下は、tmconfig環境ファイルを参照し、SERVERセクションに定義されたすべての応用サーバー・プロセスを実行する例です。

```
$ tmboot -A
```

- 以下は、/user1/tmax/conディレクトリーのexconfig環境ファイルを参照し、NODEセクションに登録されているcosmoノードに存在する応用サーバー・プロセスを実行する例です。

```
$ tmboot -n cosmo -f /user1/tmax/con/exconfig
```

- 以下は、tmconfig2環境ファイルを参照し、svr1プロセスを5つ実行する例です。

```
$ tmboot -s svr1 -k 5 -f tmconfig2
```

---

## 参考

cfl、tmboot、tmdownコマンドについての詳細は、それぞれ[???](#)、[「4.1.2. tmboot」](#)、[「4.2.1. tmdown」](#)を参照してください。

---

## tmbootコマンドを実行時のtmconfigパス参照

tmbootコマンドでTmaxを起動する場合、バイナリ環境ファイル\${TMAXDIR}/config/tmconfigを\${TMAXDIR}/path/tmconfigにコピー後、\${TMAXDIR}/pathディレクトリーにある環境ファイルを使用します。

しかし、すでにTmaxエンジンが起動している状況でtmboot -Sまたは-sを使用して特定サーバーのみを起動しようとした場合にも\${TMAXDIR}/config/tmconfigを参照した場合、以下のような環境で問題になることがあります。

- 環境ファイル

<既存の運用環境ファイル>

```
*SVRGROUP
svg1      NODENAME = "tmaxh4"
*SERVER
svr1      SVGNAME = svg1
svr2      SVGNAME = svg1
*SERVICE
TOUPPER1  SVRNAME = svr1
TOUPPER2  SVRNAME = svr2
```

<運用中に変更された環境ファイル>

```
*SVRGROUP
svg1      NODENAME = "tmaxh4"
*SERVER
svr1      SVGNAME = svg1
svr3      SVGNAME = svg1
svr2      SVGNAME = svg1
*SERVICE
TOUPPER1  SVRNAME = svr1
```

```
TOUPPER3    SVRNAME = svr3
TOUPPER2    SVRNAME = svr2
```

以下のように、運用中に変更された環境ファイルをCFLで再コンパイルします。

```
$ cfl -i node1.m
```

新規追加されたサーバーを起動します。

```
$ tmboot -S svr3
```

運用中の環境で環境ファイルを変更後、tmboot-Sを実行しようとした場合、以下のようなエラーが発生することがあります。

```
(E) BOOT3007 maxsvr (1) is over for svr(svr3:svr2): nodeno = 0, svri = 5, cur =
1,
ksvr = 1 [BOOT0015]
```

運用環境でCFLを実行した場合、\${TMAXDIR}/config/tmconfigは変更された内容が適用されます。しかし、実際の共有メモリには変更前の\${TMAXDIR}/path/tmconfigと同一構成されているため、tmboot-Sで新規追加したサーバー・プロセスが起動時、実際にはすでに実行中のサーバー・プロセスを追加で起動します。Tmaxエンジンが起動している状況でのCFLは許容されませんが、このミスによるエラーが発生した場合、該当エラーはデバッグが難しいです。

したがって、Tmaxエンジンの動作中に、tmboot[-S]、[-s]、[-g]、[-q]、[-t]、[-A]などのオプションで各サーバーを起動する場合に参照するtmconfigのパスは、\${TMAXDIR}/config/tmconfigではなく、\${TMAXDIR}/path/tmconfigです。ただし、エンジンを起動する場合は\${TMAXDIR}/path/tmconfigを参照します。

```
$ cfl -i new_config.m -o tmchg
$ tmaxadmin : cfgadd -I tmchg
$ tmboot -S new_svr -f tmchg
```

---

#### 注

tmbootコマンドを実行時、[-f]オプションを使用して特定バイナリ環境ファイルを指定した場合、\${TMAXDIR}/config/tmconfigを参照してサーバーを起動します。動的にサーバーを追加した場合は、必ず[-f]オプションで特定環境ファイルを指定し、\${TMAXDIR}/configの変更されたバイナリ環境ファイルを参照するようにします。

---

## 4.2. Tmaxの終了

Tmaxシステムの終了はバイナリー形式のTmax環境ファイルに基づいて実行されます。システムで使用していた共有メモリを削除し、起動されたTmaxプロセス(TMM、TMS、CLL、CLH、RQS)とアプリケーション・サーバー・プロセスを終了します。

## 4.2.1. tmdown

Tmaxシステム全体、あるいは一部分を終了させるコマンドです。tmdownコマンドはTmax環境ファイルに基づいてTmaxシステムを終了させるため、基本的に[-f]オプションを使用し、参照するバイナリTmax環境ファイル(ソース設定ファイル)をcflコマンドでコンパイルした結果をパスと一緒に指定する必要があります。[-f]オプションが省力された場合、デフォルト値としてTMAXDIRディレクトリー下位のconfigディレクトリーにあるtmconfigファイルを参照します。

[-f]以外のオプションが使用されていない場合、tmdownコマンドはTmaxのすべての管理プロセスとTmax環境ファイルのSERVERセクションに登録されているすべてのサーバー・プロセスを終了させます。そして、Tmaxシステムと関連したIPCリソースを削除します。終了の順序は、以下のとおりです。

1. SERVERセクションに登録されている応用サーバー・プロセスが終了します。
2. サーバー・グループ別にTMSプロセスが動作中の場合は、TMSプロセスが終了します。
3. Tmaxシステム管理プロセスが終了します。プロセスの終了は、**CLH**、**CLL**、**TMM**の順に行われます。これが一般的なシステム管理プロセスの終了順序ですが、CLHのMIN値が1でない場合は、CLLがCLHより先に終了することもあります。

バックアップで起動されたサーバーに動的に登録されているサービスがある場合に、このサービスがすべてダウンして、動的に登録されているサービスが共有メモリーから消える場合を仮定します。この場合、障害復旧後(バックアップ・サーバーがすべて終了し、正常ノードのサーバーが再起動している状態)、バックアップ・ノードに接続したクライアントに対してnamingサービスを提供できなくなります。したがって、バックアップ・サーバーの動的サービスは、該当サーバーがすべて終了しても、共有メモリーから削除されないように処理します。

### ● 使用方法

```
$tmdown [-A] [-f バイナリTmax環境ファイル名] [-g servergroup_name] [-h] [-V][-i]
        [-n node_name] [-p server_num] [-q RQ svg_name]
        [-s server_name [-k count]] [-S server_name] [-t TMS_name[-k all]]
        [-w wait_time] [-B Nodename][-R] [-y]
```

項目	説明
[-A]	すべての応用サーバー・プロセスを終了させます
[-f バイナリTmax環境ファイル名]	システム終了時に参照するバイナリTmax環境ファイルの名前を指定する項目です。ファイル名を指定していない場合、デフォルトでTMAXDIRディレクトリー下位のconfigディレクトリーにあるtmconfigファイルを参照します
[-g servergroup_name]	指定されたサーバー・グループに属するサーバー・プロセスを終了します
[-h]	コマンドのヘルプ・オプションです
[-V]	実行ファイルのバージョンを確認できます

項目	説明
[ -i ]	tmddownコマンドを即時実行します。基本的にtmddownコマンドは該当アプリケーションをすべて終了して実行されますが、[-i](immediately)オプションによる終了は、現在実行中のアプリケーションをすべて中断するため、慎重に使用する必要があります
[ -n node_name ]	指定されたノードに存在するすべてのサーバー・プロセスを終了させます。ノード名は、Tmax環境ファイルの <b>NODE</b> セクションに登録されている必要があります
[ -p server_num ]	指定されたサーバー・プロセスを終了させます。[-s]オプションによる終了とは異なり、tmadminで「st-p」コマンドで確認できるプロセス番号(spr_no)を使用して特定プロセスを終了させます
[ -q RQ svg_name ]	RQSを終了させます
[ -s server_name [-k count] ]	指定されたサーバー・プロセス1つのみを終了させます。使用されるサーバー・プロセス名は、Tmax環境ファイル内のSERVERセクションにあらかじめ登録されている必要があります
[ -S server_name ]	指定されたサーバー・プロセスをすべて終了させます。1つ以上のプロセスが指定されている場合、該当するすべてのプロセスを終了させます
[ -t TMS_name [-k all] ]	指定されたTMSプロセスを1つのみ終了させます。  [-k all]は、トランザクション・リカバリー機能を使用する場合に必要なオプションです。グループ全体を終了後に起動する場合、リカバリーが実行されるため、リカバリーはTMSグループ単位で可能です。特定名をもつTMS全体を起動/終了する場合、このオプションを使用できます
[ -w wait_time ]	wait_timeに指定された時間が過ぎた場合、tmddownコマンドを実行します
[ -B Nodename ]	TRBノードを終了させます
[ -R ]	Rolling Down時に使用します
[ -y ]	tmddown時に、終了の可否(y   n)を聞かずに即終了させます

#### ● 例

- 以下は、TMAXDIRディレクトリー下位のconfigディレクトリーにあるtmconfigファイルを参照し、全体Tmaxシステムを終了する例です。つまり、Tmax管理プロセスと応用サーバー・プロセスをすべて終了します。

```
$tmddown
```

- 以下は、名前がtms\_nameのすべてのTMSが終了する例です。

```
$tmddown -t tms_name -k all
```

- 以下は、サーバー・グループの設定オプションを使用した例です。互いに異なるサーバー・グループで tms\_name を同一に使用する場合、上記のように、[-g] オプションで該当 TMS が属しているサーバー・グループ名を指定します。

そうでない場合、該当 TMS 名をもつ最初のサーバー・グループの TMS が終了します。

```
$tmdown -t tms_name -k all -g svgname
```

- 以下は、tmconfig2 環境ファイルを参照し、全体 Tmax システムを終了する例です。

```
$tmdown -f tmconfig2
```

- 以下は、tmconfig 環境ファイルを参照し、svr1 という応用サーバー・プロセスをすべて終了する例です。

```
$tmdown -S svr1
```

- 以下は、tmconfig 環境ファイルを参照し、svr1 という応用サーバー・プロセスをすべて強制終了する例です。svr1 のうち、特定サーバー・プロセスのサービスが終了していない場合、[-i] オプションを使用して強制終了します。

```
$tmdown -S svr1 -i
```

- 以下は、tmconfig 環境ファイルを参照し、<spr\_no> のサーバー・プロセスのみ強制終了する例です。該当サーバーのサーバー・プロセスが複数の場合、特定サーバーが looping の場合、該当プロセスのみ強制終了します。

```
$tmdown -k <spr_no> -i
```

- 以下は、/user1/tmax/con ディレクトリーの exconfig 環境ファイルを参照し、NODE セクションに登録されている cosmo ノードに存在する応用サーバー・プロセスを終了する例です。

```
$tmdown -n cosmo -f /user1/tmax/con/exconfig
```

- 以下は、tmconfig2 環境ファイルを参照し、動作中の svr1 プロセス1つのみを終了する例です。

```
$tmdown -s svr1 -f tmconfig2
```

## Rolling Down機能

既存バージョンでは、クライアントの要求を処理していた Tmax システムが異常終了した場合、現在処理中の要求に対してのみ応答を処理して渡した後、キューに蓄積している要求に対して TPECLOSE エラーを渡していました。しかし Tmax v5.0 からは、Tmax エンジンダウンさせる前に、要求されたすべてのクライアントに対して正常に応答する Rolling Down 機能を提供します。

- 使用方法

```
$ tmdown -R -n node_name
```

項目	説明
-n node_name	終了されるノード名を指定します

- 適用環境

Tmaxシステムがインストールされている運用システム環境で使用できます。

- 例

以下は、NODEAとNODEBがマルチノード(またはマルチドメイン)で構成されており、総100個のクライアントがNODEAに接続されていると仮定した場合の処理過程を説明する例です。

– NODEAのTmaxシステムを終了させる場合

```
$tmdown -R -n NODEA
```

1. NODEAのCLLは、クライアントからのListenポートを遮断します。
2. NODEAで既存処理されていた要求に対する処理の完了後、クライアントに処理結果を渡します。
3. キューに蓄積している要求に対しては、TMAX\_BACKUP\_ADDRと設定されたNODEBに要求を送ります。
4. NODEAのTmaxシステムが終了します。
5. NODEBでは、NODEAから受信した要求を処理後、初めに該当リクエストを要求したクライアントに処理結果を直接渡します。
6. NODEAに接続されているすべてのクライアントは、正常に応答を受信します。

– NODEBのTmaxシステムを終了させる場合

```
$tmdown -R -n NODEB
```

1. 100個のクライアント要求を、NODEAのCLHが約50:50でNODEAとNODEBに分配されます。
2. tmdown -R -n NODEBで、NODEBのTmaxシステムを終了させます。
3. NODEBのCLLは、クライアントからListenポートを遮断します。
4. NODEBで既存処理されていた要求に対しては、処理を完了します。



5. クライアントはNODEAに接続されている状況のため、該当処理結果をNODEAのCLHに渡し、NODEAのCLHはクライアントに処理結果を渡します。
6. NODEBのキューに蓄積している要求に対しては、TMAX\_BACKUP\_ADDRと設定されたNODEAに要求を送ります。
7. NODEBのTmaxシステムが終了します。
8. NODEAでは、NODEBから受信した要求を処理後、初めに該当リクエストを要求したクライアントに処理結果を渡します。
9. NODEAに接続されているすべてのクライアントは、正常に応答を受信します。100個のクライアントがすべて正常に応答を受信する必要があります。

---

#### 注

NODEAの要求をNODEBが処理するためには、NODEAに接続したクライアントのTMAX\_BACKUP\_ADDR、TMAX\_BACKUP\_PORTがNODEBに設定されている必要があります。そうでない場合、NODEAのTmaxシステムが終了した瞬間、未処理のクライアント要求に対してTPESYSTEMエラーを渡します。

---



# 第5章 管理ツール

本章では、Tmaxの効率的な管理のために提供される管理ツールについて説明します。

## 5.1. 概要

Tmaxシステムの動作中には、現在の環境設定の情報を確認あるいは動的に変更したり、サーバー・プログラムにサービスを追加したりするなどのシステム管理が必要です。また、提供しているサービスのうち特定のサービスに対する現在の処理状態、たとえば、サービス処理件数、平均処理時間、要求待機件数、サービス待機時間などの情報を確認する必要があります。これらの情報に基づいて起動中のサーバー・プロセスの数を増やすか、起動を終了させるなどの措置も必要になります。

Tmaxシステムは**tmadmin**を提供し、コマンド・インタープリター(command interpreter)形態のコマンドを使って動的なシステム管理ができます。

## 5.2. tmadmin

tmadminプログラムはUNIX環境のシェル(shell)に似たコマンド・インタープリターです。通常はプロンプト状態で待機し、コマンドが入力されるとそのコマンドを解釈して実行します。複数ノードを1つのドメインで使用する場合、tmadminで全体に対して中央管理を行うことができ、各ノード別にローカルで管理できます。

### • 使用方法

```
$ tmadmin [-l] [-s|m] [-h] [-f [Config File ]] [-n [Node Name]]
          [-v] [-V] [-p] [-t]
```

項目	説明
[-l]	racdによって複数ノードで構成されたシステムが中央で集中管理される場合、ローカルノードのみを管理するために使用するオプションです。各ノードはこのオプションで自分のシステムのみを管理できます
[-s]	読み取り専用モードでそれぞれのtmadminツールを10個まで使用できるオプションです。このオプションを使用すると環境を動的に変更できません(デフォルト値)
[-m]	動的に環境を変更できるマスター・モードです。マスター・モードは1人のユーザーのみ使用することを推奨します。2人以上のユーザーが環境を変更できる場合、深刻なシステム問題が生じる可能性があります。環境ファイルは変更せずに決まったマニュアルに従います
[-h]	コマンド・ヘルプを表示します

項目	説明
[ -f [ <i>Config File</i> ] ]	指定されたバイナリー・ファイルを管理します。環境ファイルをデフォルト値の tmconfig 以外の名前でバイナリー・ファイルを作成した場合、tmadmin を起動する際にバイナリー・ファイル名を指定します
[ -n [ <i>Node Name</i> ] ]	特定ノードをモニタリングできます。マルチノード環境で tmadmin を起動させる場合、このオプションを設定すると指定されたノード情報だけを確認するのでより便利にシステムを管理できます
[ -v ]	Tmax のバージョンを確認します。このオプションを設定すると Tmax の起動と関係なく、どの位置でもバージョンを確認できます
[ -V ]	実行ファイルのバージョンを確認します
[ -p ]	st -p、st -s、si、ci、cfg コマンドを実行したとき、結果を画面単位で出力するオプションです (more 機能)
[ -t ]	コマンド実行の開始および終了時間を出力します。詳しくは以下で説明します

[ -t ] オプションを指定して tmadmin を実行すると、以下のような形式で時間を出力します。

```
[TIME][タイプ] : hh:MM:ss:millisec
```

タイプ	説明
START	コンソールでコマンドを実行する際、開始時間を出力します
END	コンソールでコマンド実行が終了したら、終了時間を出力します (リモートノードのコマンドまで実行した後の時間です)
R_END	コンソールでコマンドを実行する際、リモートノードの実行が終了したら終了した時間 (tmadmin を実行したローカルの時間であり、リモートノードの時間ではない) を出力します
RP_START	tmadmin の repeat コマンドを使用して実行する場合、1 件のコマンドを開始する前の時間を出力します
RP_END	tmadmin の repeat コマンドを使用して実行する場合、1 件のコマンドが終了したら、終了した時間を出力します (リモートノードのコマンドまで実行した後の時間です)

## コマンドの実行

tmadmin ではコマンドにより Tmax システムを起動または終了したり、現在動作中のシステムの環境設定内容を照会及び修正したりできます。また、各サーバー・プロセスの状態とサービスの処理状態などを確認できます。

tmadmin プログラムは以下のように実行します。

```
$tmadmin
```

以下のようなメッセージと共にプロンプト(prompt)が表示されます。これはtmadminプログラムが実行状態であることを意味します。

```
--- Welcome to Tmax Admin (Type "quit" to leave) ---  
$$1 tmax1 (tmadm):
```

tmadminを終了するには、quit または qコマンドを使用します。

```
$$1 tmax1 (tmadm):quit
```

以下は、tmadminの実行後に使用できるコマンドの一覧です。

- 環境情報の照会

コマンド	説明
<a href="#">config(cfg)</a>	環境設定の内容を照会します
<a href="#">configopt(cfgopt)</a>	TMMOPTなどの環境設定に定義されているオプションのうち、動的変更が可能な設定の現在の設定値を照会します
<a href="#">history(hist)</a>	以前保存されたコマンドを照会します
<a href="#">tmaxinfo(ti)</a>	Tmaxのシステム情報を確認します

- 動作状態情報の照会

コマンド	説明
<a href="#">stat(st)</a>	プロセスおよびサービス状態を統計します
<a href="#">gwinfo</a>	GATEWAYセクションに定義したゲートウェイのチャンネル状態を確認します
<a href="#">txgwinfo(txgwi)</a>	Tmaxゲートウェイの情報を確認します
<a href="#">nontxgwinfo</a>	Tmax ノン・トランザクション・ゲートウェイの情報を確認します
<a href="#">jgwinfo</a>	JEUSゲートウェイの情報を確認します
<a href="#">ajgwinfo</a>	JEUS Asyncゲートウェイの情報を確認します
<a href="#">wsgwinfo</a>	Webサービス・ゲートウェイの情報を確認します
<a href="#">smtrc</a>	GIDを利用して当該サービスの実行状態を照会します
<a href="#">clhsinfo</a>	マルチノード環境でCLH間の接続状態情報を確認します
<a href="#">tmmsinfo</a>	マルチノード環境でTMM間の接続状態情報を確認します
<a href="#">repeat(r)</a>	コマンドを繰り返します
<a href="#">clientinfo(ci)</a>	接続中のクライアントを確認します
<a href="#">svrinfo(si)</a>	サーバー情報を確認します
<a href="#">txquery(txq)</a>	トランザクション処理情報を確認します

コマンド	説明
<a href="#">rqstat(rqs)</a>	RQ状態を確認し、ディスクキューにたまっているサービスを処理します

- 運用管理

コマンド	説明
<a href="#">suspend(sp)</a>	動作中のサーバー・プロセスを中止します
<a href="#">resume(rs)</a>	中止されたサーバー・プロセスを再開します
<a href="#">advertise/unadvertise</a>	特定のサービス名をアドバイズまたはアンアドバイズします
<a href="#">restat</a>	特定サーバー・プロセスまたは全サーバー・プロセスの統計情報をリセットします
<a href="#">rebootsvr(rbs)</a>	サーバー・プログラムを取替えます
<a href="#">cfgadd(ca)</a>	動的にサービスを追加します
<a href="#">set</a>	現在設定された環境設定値を動的に変更します
<a href="#">setopt</a>	TMMOPTなどの環境設定に定義されているオプションのうち、動的変更が可能な設定の現在の設定値を照会します
<a href="#">qpurge(qp)</a>	キューにたまっている業務を削除します
<a href="#">discon(ds)</a>	接続中のクライアントを強制に解除します
<a href="#">logstart / logend</a>	ロギングを開始・終了します
<a href="#">chtrc</a>	トレースを管理します
<a href="#">chlog</a>	TMM、CLH、特定サーバーのログレベルをランタイム中に動的に変更します
<a href="#">chlog2</a>	ログ・レベルをランタイム中に動的に変更します
<a href="#">txcommit / txrollback</a>	トランザクション処理中に障害が発生した場合、コミットまたはロールバックを再発行して当該トランザクションを終了します
<a href="#">wsgwreload</a>	Webサービス・ゲートウェイのサービス情報、設定変更を適用します
<a href="#">restart</a>	サーバー・プロセスを再起動します
<a href="#">notify_reconnect_clh</a>	特定のサーバー・プロセスに特定のCLHIに接続するよう命令します
<a href="#">admnoti</a>	TCS、UCS、RDPサーバーにイベントを送信します

- その他

コマンド	説明
<a href="#">!</a>	直前のコマンドを繰り返します
<a href="#">quit(q)</a>	tmadminを終了します
<a href="#">help(h)</a>	使用できるオプション・リストを照会します
<a href="#">nodeset(ns)</a>	マルチノード環境で特定ノードの情報のみを取得するときに使用します

コマンド	説明
nodeunset(nus)	マルチノード環境で特定ノードの情報を取得するための設定を解除します
tmd	仮想のクライアント・エミュレーターです。サービス・プログラムの有効性をチェックする際、クライアント・プログラムを別途作成することなく、このユーティリティを使用します

## 5.3. 環境情報の照会

### 5.3.1. tmaxinfo(ti)

現在接続されているTmaxシステムの環境情報を示します。

- 使用方法

```
$$1 tmax1 (tmadm): ti
```

- 例

以下のとおり、システム・バージョン(version)や最大ユーザー数(maxuser) などを確認できます。

```

$$1 tmax8 (tmadm): ti

Tmax System Info: REAL version 6.0:

    maxuser = UNLIMITED,
    Supported maximum user per node = 16076,
    Supported maximum user per handler = 16077,
    domaincount = 1,
    nodecount = 1,
    svgrpcount = 1,
    svrcount = 1, svccount = 1
    rout_groupcount = 0, rout_elemcount = 0
    cousin_groupcount = 0, cousin_elemcount = 0
    backup_groupcount = 0, backup_elemcount = 0

Tmax All Node Info: nodecount = 1:
-----
  no   name      portno  racport  shmkey  shmsize  minclh  maxclh
-----
    0   tmax8      7789    3378     87789   755872      1      3
-----

```

## 5.3.2. history

シェルと同様、コマンドを記憶することで便利に使用できます。

- 使用方法

```
$ $1 tmax1 (tmadm): history
```

- 例

以下はコマンドの使用例です。

```
$ $15 tmax1 (tmadm): history
5 : si
4 : txq
3 : ci
2 : st -p
1 : st -s
0 : ci
```

感嘆符(!)を使用すると直前のコマンドを繰り返すことができ、historyで出力された番号を使用してコマンドを再実行することもできます。

```
$ $1 tmax1 (tmadm): !5
si
-----
clh      svri      status      count      qcount      qpcount      emcount
-----
0         0        RDY         2          0           0           0
0         1        RDY         0          0           0           0
0         2        RDY         0          0           0           0
0         3        RDY        45          0           0           0
```

## 5.3.3. config(cfg)

現在動作中のシステムの環境情報を表示します。環境ファイルで定義されたドメイン、ノード、サーバー・グループ、サーバー、サービス別に、デフォルト値を含んだすべての環境情報を確認できます。出力項目については「[第3章 環境ファイルの設定](#)」を参照してください。

- 使用方法

```
$ $1 tmax1 (tmadm): config (cfg) [-d] [-n] [-g[server_group_name]]
                                [-v[service_name]] [-s [service_name] [-x]]
                                [-w [gateway_name]] [-r] [-b] [-f] [-pr]
```



項目	説明
[ -d ]	ドメインの環境情報を照会します。  環境ファイルのDOMAINセクションに関連するすべての項目内容とドメインに存在する情報を確認できます
[ -n ]	全体ノードの環境情報を照会します。  環境ファイルのNODEセクションに関連するすべての項目内容とシステムで内部的に定義された情報を確認できます
[ -g [server_group_name] ]	サーバー・グループの環境情報を照会します。  サーバー・グループ名を指定すると、そのサーバー・グループの情報のみを出力し、サーバー・グループ名を指定しないと、すべてのサーバー・グループの情報を出力します。  サーバー・グループ別に環境ファイルのSVRGROUPセクションに関連するすべての項目を確認できます
[ -v [service_name] ]	サーバーの環境情報を照会します。  サーバー名を指定すると、そのサーバーの情報のみを出力し、サーバー名を指定しないと、すべてのサーバーの情報を出力します。  サーバー別に環境ファイルのSERVERセクションに関連するすべての項目内容と内部的に定義されたサーバー番号(svr_no)、サーバーに関連するデータ依存型のルーティング情報(ddri)を確認できます
[ -s [service_name] [-x] ]	サービスの環境情報を照会します。  サービス名を指定すると、そのサービスの情報のみを出力し、サービス名を指定しないと、すべてのサービスの情報を出力します。  サービス別に環境ファイルのSERVICEセクションに関連するすべての項目内容を確認できます。  [-x]オプションを使用する場合、txtime情報を追加で出力します
[ -w [gateway_name] ]	ゲートウェイの環境情報を照会します。  ゲートウェイ名を指定すると、そのゲートウェイの情報のみを出力し、ゲートウェイ名を指定しないと、すべてのゲートウェイの情報を出力します
[ -r ]	ルーティング環境情報を照会します
[ -b ]	バックアップ環境情報を照会します
[ -f ]	TopEndの関数情報を照会します
[ -pr ]	TopEndのproduct情報を照会します

- 例

- ドメインの環境情報(-d)

以下は、ドメインの環境情報を確認する例です。

```
$$1 tmax8 (tmadm): cfg -d
  domain_name = tmax1,
    shmkey = 79190,
    minclh = 1,
    maxclh = 3,
    maxuser = UNLIMITED,
    portno(portno) = 7789,
    racport = 3333,
    cmtret = YES,
    blocktime(bt) = 30.000,
    txttime(tt) = 0.000,
    nliveinq(ni) = 30,
    security = NONE,
    cpc = 1,
    maxfunc = 0,
    clichkint(clichkint) = 0,
    idletime(idletime) = 0,
    node_count = 1,
    svg_count = 6,
    svr_count = 14,
    cousin_count = 0,
    cousin_gcount = 0,
    backup_count = 0,
    backup_gcount = 0,
    rout_count = 0,
    rout_gcount = 0,
    maxsacall = 8,
    maxcacall = 16,
    nclhchctime(nclhchctime) = -1,
    txpendingtime(txpendingtime) = 0,
    maxconv_node = 16,
    maxconv_server = 8,
    maxnode = 32,
    maxsvg = 32,
    maxspr = 64,
    maxsvr = 64,
    maxsvc = 512,
    maxcpc = 150,
    maxtms = 32,
    maxrout = 16,
    maxroutsvg = 32,
    maxrq = 2,
```

```

maxgw = 2,
maxcousin = 16,
maxcousinsvg = 32,
maxbackup = 16,
maxbackupsvg = 32,
maxtotalsvg = 64,
maxprod = 0,
tipsvc = ,
crypt = NO,
maxthread = 128,
tmmloglvl = DETAIL,
clhloglvl = DETAIL,
tlmloglvl = DETAIL,
casloglvl = DETAIL,
rsloglvl = DETAIL,
sqlloglvl = DETAIL,
tgloglvl = DETAIL,
tmsloglvl = DETAIL,
hmsloglvl = DETAIL,
rqloglvl = DETAIL,
gwloglvl = DETAIL,
c1lloglvl = DETAIL,
loglvl = DETAIL,
cllblock = NO,
tdl = NO,
maxconvn = 16,
maxconvs = 8,
domainid = 0,
fdlversion = 1,
tgshmsize = 0
tgmax = -1
tgmax_child = -1
tgmax_watcher = -1
tgheartbeat = 0
tgtimeout = 0
tg_mcast_ip = 224.0.0.100,
tg_portno = 9999,
tghistorycount = 50
tgstandbycount = 50
tgmaxbuffersize = 65000
tgdownwaittime = 30

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
nodecount	ドメインに存在するノード数です

項目	説明
svgcount	サーバー・グループ数です
cousin_gcount	COUSIN項目が定義されたグループ数で、負荷調整のための複製サーバー・グループ数です
cousin_count	すべての複製サーバー・グループの数です
rout_count	ROUTINGセクションに定義されたルーティング個体の数です
relem_count	各ルーティング個体の範囲項目に定義された各範囲別ルーティングの単位数です

#### – ノード全体の環境情報(-n)

以下は、ノード全体の環境情報を確認する例です。

```

$$1 tmax8 (tmadm): cfg -n
node_name = tmax8, hostname = tmax8, node_no = 0
load = 0,
shmkey = 79190,
minclh = 1,
maxclh = 3,
maxuser = UNLIMITED,
Supported maximum user per node = 15931,
Supported maximum user per handler = 15932,
clhgttimeout(cgt) = 0,
portno(portno) = 7789,
racport = 3333,
tmaxhome = /data/tmaxha/tmax/,
tmaxdir = /data/tmaxha/tmax/,
appdir = /data/tmaxha/tmax/appbin/,
pathdir = /data/tmaxha/tmax/path/,
tlogdir = /data/tmaxha/tmax/log/tlog/,
slogdir = /data/tmaxha/tmax/log/slog/,
ulogdir = /data/tmaxha/tmax/log/ulog/,
envfile = ,
svgcount = 6,
svrcount = 14,
svccount = 24,
curclh = 1,
sprcount = 14,
tmscount = 6,
cpccount = 10,
cmprsize(cmprsize) = -1,
ipcperm = 1c0,
clichkint(clichkint) = 0,
idletime(idletime) = 0,
tmmpopt = ,

```

```

clhopt = ,
tlmopt = ,
realsvr = ,
rscpc = 4,
maxsvg = 32,
maxsprs = 64,
maxsvr = 64,
maxtmss = 32,
maxcpc = 150,
maxgwsvr = 2,
maxrgsvr = 2,
maxgwcpc = 8,
restart(rs) = YES,
maxrstart(mr) = -1,
gperiod(gp) = 86400,
autobackup(ab) = YES,
extport = 0,
maxthread = 128,
cllblock(cb) = NO,
cllconnlb(cllconnlb) = RR,
cllbindip = YES,
tdl = NO,
sqkey = 78550,
sqsize = 8388608,
sqmax = 1024,
sqkeymax = 64,
sqtimeout = 30,
smsupport = NO,
msgsizewarn(msw) = 1073741824,
msgsizemax(msx) = 1073741824,
tgshmkey = -1,
tgid = -1,
tgmcast_ttl = 1,
tgmcast_if = ,
crypt_algorithm = ""
svclog_format = ""
casthread = -1
cas = ""
security_lib = ""
crypt_lib = ""

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
node_no	システムで内部的に定義されたノード番号です
svgcount	ノード内に存在するサーバー・グループ数です

項目	説明
svrcount	サーバー数です
svccount	サービス数です
curclh	ノードで動作中のCLHプロセス数です
maxsprs	動作可能な最大のサーバー・プロセス数です
maxtms	最大のTMSプロセス数です
autobackup	<p>backupに宣言されているsvgのサーバーを自動で起動するかどうかを設定します。(Y  N)</p> <p>主に、tmdown -iでmainノードをダウンさせる場合、自動でバックアップ・サーバーが起動されないようにするためにNに変更し、tmdownさせた後再びYに変更する方法で運用します。</p> <p>tmaxadminを実行するノードまたは接続されたノードがsetコマンドで設定しようとするノードでない場合は、以下のようにエラーが出力され、変更事項が反映されません。</p> <pre>no such name(nodename) for the section defiend in config.</pre> <p>他のノードの設定を変更するには、対象ノードにracdが実行されている必要があります</p>

#### – サーバー・グループの環境情報(-g)

以下は、サーバー・グループの環境情報を確認する例です。

```

$$2 tmax8 (tmadm): cfg -g svg2
    svg_name = svg2, svg_no = 3
        xaoption = ,
        openinfo =
Oracle_XA+Acc=P/scott/tiger+SesTm=60+DbgFl=7+LogDir=/data/tmaxha/tmax/log/xalog,

        closeinfo = ,
        appdir = ,
        ulogdir = ,
        svgtype = TMAX,
        envfile = ,
        tmsname = tms_ora,
        mintms = 2,
        maxtms = 3,
        curtms = 2,
        tmstype = STD,
        tmsthreads = 0,
        tmsopt = -o svg2 -e svg2,

```

```

tmsrecovery = YES,
tmsloglvl = DEBUG4,
tmsrange = DOMAIN,
tmsxatime(tmsxatime) = 0,
load(ld) = -9,
tms starti = 0,
tms endi = 2,
restart(rs) = YES,
maxrstart(mr) = -1,
gperiod(gp) = 86400,
autobackup = YES,
dummy = NO,
dbname = ORACLE
rmid = 0
svclog_format = ""

```

#### – サーバーの環境情報(-v)

以下は、サーバーの環境情報を確認する例です。

```

$$4 tmax8 (tmadm): cfg -v svr2
svr_name = svr2, svr_no = 5
    svgno = 2,
    cursvr = 1,
    clopt = ,
    seq = -1,
    minsvr = 1,
    maxsvr(max) = 1,
    ulogdir = ,
    maxqcount(mq) = -1,
    asqcount(aq) = -1,
    conv = NO,
    ddri = DDR_NO_ROUT,
    lifespan(ls) = -1,
    restart(rs) = YES,
    maxrstart(mr) = 5,
    gperiod(gp) = 86400,
    svrtype = TMAX_STD,
    schedule(schedule) = FA,
    cpc = 1,
    dummy = NO,
    aus = NO,
    mac(mac) = NO,
    roc(roc) = NO,
    multiclh = YES,
    ctx_ereply = NO,
    svrqtimeout(sqt) = -1,

```

```
inbound_cpc = 1,
svclog_format = ""
```

## – サービスの環境情報(-s)

以下は、サービスの環境情報を確認する例です。

```
$$5 tmax8 (tmadm): cfg -s
```

svc_name svgno	funcname	prio(pr)	autotran	svctime(st)	routno	svrname
_hms01 4	TOLOWER	50	NO	0.000	-1	_hms01
2	TOUPPER	50	NO	0.000	-1	svr2
2	HMS	50	NO	0.000	-1	svr_hms
2	FDLTOLOWER	50	NO	0.000	-1	svr3
2	FDLTOUPPER	50	NO	0.000	-1	svr3
gw1 0	gw1	50	NO	0.000	-1	gw1
gw2 1	SDLTOLOWER	50	NO	0.000	-1	svr1
2	SDLTOUPPER	50	NO	0.000	-1	svr1
2	_rq1	50	NO	0.000	-1	_rqsvg
5	LOGIN	50	NO	0.000	-1	svr_ucs
2	TPDEQ	50	NO	0.000	-1	svr_rq
2	TPENQ	50	NO	0.000	-1	svr_rq
2	FDLDEL	50	NO	0.000	-1	fdltest
3	SDLDEL	50	YES	0.000	-1	sdltest
3	FDLSEL	50	NO	0.000	-1	fdltest
3	FDLINS	50	NO	0.000	-1	fdltest



TOUPPER_CONV	50	NO	0.000	-1	svr_conv
2					
SDLSEL	50	YES	0.000	-1	sdltest
3					
SDLINS	50	YES	0.000	-1	sdltest
3					
FDLUPT	50	NO	0.000	-1	fdltest
3					
SDLUPT	50	YES	0.000	-1	sdltest
3					
GET_SQ	50	NO	0.000	-1	svr_sq
2					

#### – ゲートウェイの環境情報(-w)

以下は、ゲートウェイの環境情報を確認する例です。

```

$$7 tmax8 (tmadm): cfg -w gw1
  gw_name = gw1, node_no = 0, gw_no = 0
  gw_type = TMAXNONTX,
  portno = 4021,
  rgwaddr = 192.168.1.86,
  rgwportno = 4021,
  backup_rgwaddr = ,
  backup_rgwportno = -1,
  backup_rgwaddr2 = ,
  backup_rgwportno2 = 0,
  backup_rgwaddr3 = ,
  backup_rgwportno3 = 0,
  cpc = 1,
  timeout(timeout) = 11000,
  direction(direction) = BIDIR,
  maxinrgw = 32,
  clopt = -i,
  ptimeout(ptimeout) = -1,
  ptimeint(ptimeint) = -1
  gwchkint(gwchkint) = -1
  gwconnect_timeout = -1
  nliveinq(nliveinq) = -1

```

### 5.3.4. configopt(cfgopt)

TMMOPTなどの環境設定に定義されているオプションのうち、動的変更が可能なオプションの現在の設定値を照会します

- 使用方法

- `$$1 tmax1 (tmadm): configopt (cfgopt) [-tmm]`

項目	説明
<code>[-tmm]</code>	TMMOPT環境設定に定義されているオプションのうち、動的変更が可能なオプションの現在の設定値を確認します

- 例

#### – TMMOPTの環境情報(-d)

以下は、TMMOPT環境設定の環境情報を確認する例です。

```

$$1 tmax1 (tmadm): cfgopt -tmm
-----
-tmm configurable value
-----

accept retry count(-A) = 100
max forked threshold(-F) = 765
booting timeout(-t) = 10
-----

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
accept retry count(-A)	サーバー・プロセスの接続要求に対してACCEPT処理する際、一度に処理される回数です。TMMOPTの[-A]オプションを参照してください
max forked threshold(-F)	TMMでサーバー・プロセスの追加起動やプロセスの再起動などにより、新しいプロセスを作成する際、同時に処理できる最大数です。  TMMOPTの[-F]オプションを参照してください
booting timeout(-t)	ASQCOUNTで起動されたサーバー・プロセスの接続タイムアウト時間です。  TMMOPTの[-t]オプションを参照してください

## 5.4. 動作状態情報

### 5.4.1. stat(st)

実際のシステム動作状態を表示します。動作中のサーバー・プロセスとサービスに関する情報を確認できます。

サーバー・プロセスの現在状態、処理中のサービス名、処理したサービス数、サービス状態、サービスキューに存在するサービス要求数など、動的情報を確認できます。statコマンドの省略形は**st**です。

## ● 使用方法

```

$ $1 tmax1 (tmadm): stat (st) [-p [server_process_name] [-b]] [-s [service_name]]

                                [-t [TMS_name]] [-v [server_process_name] [-pod]
                                [-b]] [-o [ソート条件]] [-n [出力メッセージライン]]
                                [-x] [-X] [-q [destination_name [-c]] ] [-d]
                                [-j] [-tg]

```

項目	説明
[-p [server_process_name]] ]	<p>サーバー・プロセスの情報を照会します。</p> <p>Tmax環境ファイルに登録されたサーバーはMIN項目とMAX項目を使用して複数動作でき、それぞれの動作状況を確認できます。サーバー名を指定すると、当該サーバー名のサーバー・プロセスの情報を出力し、サーバー名を指定しないと、動作中のすべてのサーバー・プロセスの情報を出力します。</p> <p>「st -p」コマンドの最後のラインに総処理件数、全体の平均処理時間、現在実行中のサービス総数の統計情報を出力し、システム処理状況を一覧できます</p>
[-s [service_name]]	<p>サービスの情報を照会します。</p> <p>サービス名を指定すると、当該サービスの情報のみ出力し、サービス名を指定しないと、提供されるすべてのサービスの情報を出力します</p>
[-t [TMS_name]]	<p>システムで提供するTMS関連の動的情報を表示します。</p> <ul style="list-style-type: none"> <li>– TMS名を指定すると、当該TMSの情報のみ出力します</li> <li>– TMS名を指定しないと、提供されるすべてのTMSの情報を出力します</li> </ul>
[-v [server_process_name]] [-pod]]	<p>サーバー・プロセスの情報を照会します。(siと同じ)</p> <p>[-pod]オプションを使用すると、PODサーバーの状態部分に常に「(POD)」が出力されます。</p> <p>このオプションがない場合のPODサーバーの状態は、起動時にはRDY、未起動時にはNRDYに表示されます。PODサーバーはNRDY状態であってもサービスの要求時には自動的に起動され処理されるため、起動状態は特に意味はありませんが、モニタリング時に問題のあるサーバーとして認識されることがあります。したがって、このオプションを使用して不要な判断ミスを減らすことができます</p>

項目	説明
[ -o [ソート条件] ]	<p>[-o]オプションは特定条件を基準に照会結果をソートして出力します。</p> <p>Tmaxシステムで提供するサーバー・プロセスやサービスの動的情報を確認したい場合、特定の条件を基にソートして必要な情報を表示できます。必ず[-p]オプション、または[-s]オプションと一緒に使用する必要があります。</p> <p>ソートは降順、昇順で設定できます</p> <ul style="list-style-type: none"> <li>- [ -o ca ] : 降順</li> <li>- [ -o ca- ] : 昇順</li> </ul>
[ -n [出力メッセージライン] ]	<p>ソートの後出力されるメッセージ・ラインを指定します。</p> <p>ソートの後出力されるメッセージ・ライン(サーバー・プロセスまたはサービス数)を指定するオプションで、必ず[-o]オプションと一緒に使用します</p>
[ -x ]	<p>「st-p」または「st-s」コマンドを使用してサーバー・プロセスやサービスの状態を照会し、次の項目を追加照会します。必ず[-s]オプション、または[-p]オプションと一緒に使用します。詳細については、<a href="#">詳細情報(-x)</a>を参照します</p> <ul style="list-style-type: none"> <li>- OSで提供するプロセスID(PID)、Fail count、Error count</li> <li>- 最小および最大サービス実行時間(min_time、max_time)</li> <li>- TMS(-t)情報を照会する際のxid、xa_status</li> </ul>
[ -X ]	<ul style="list-style-type: none"> <li>- [-s]オプションと一緒に使用する場合</li> </ul> <p>COUSINサービスについて、状態1(状態2)の形式でローカル状態を別々に照会できます(例: RDY(NRDY))</p> <p>(状態2)が実際のローカル状態であり、[-X]オプションを使用しない場合COUSIN環境でローカル状態が常にRDYで照会される問題点を解決できます。必ず[-s]オプションと一緒に使用します</p> <ul style="list-style-type: none"> <li>- [-p]オプションと一緒に使用する場合</li> </ul> <p>サーバー・プロセスのSuspend(一時停止)状態を照会します。Suspend状態にある場合は以下のように表示されます(例: RDY(BLK))</p> <p>サーバー・プロセスの一部だけを一時停止することができ、部分的に一時停止となる場合には、stat -vで照会すればサーバー状態はRDYで表示され、stat -p -Xで照会すれば一時停止になっているプロセスは実際の状態の後に「(BLK:Pp)」が一緒に表示されます</p>

項目	説明
	<p>(BLK:Pp)のPp状態は、suspendコマンドを実行時の-n/-Nオプションによって異なります</p> <ul style="list-style-type: none"> <li>• Pp <pre>sp -n P -N P</pre> </li> <li>• Tp <pre>sp -n T -N P</pre> </li> <li>• Tt <pre>sp -n T -N T</pre> </li> <li>• Pt <pre>sp -n P -N T</pre> </li> </ul>
[ -q <i>destination_name</i> ]	<p>HMSデスティネーションの情報を表示します。</p> <p>環境設定ファイルで設定したデスティネーションの一覧と各デスティネーションで処理中のメッセージとクライアントの情報を確認できます</p>
[ -d ]	<p>サーバー・プロセスを照会(st -p)する際、サーバー・プロセスがRUN状態の場合に経過時間が表示されます。それ以外の状態は「-」に表示されます。必ずst -pと一緒に使用する必要があります</p> <p>(例: st -p -d , st -p -x -d)</p>
[ -b ]	<p>サーバーを照会(st -v)したり、サーバー・プロセスを照会(st -p)するときに、サーバーの最終起動時間を一緒に表示します</p>
[ -j ]	<p>st -p -x、st -p -dの使用時に複数行で表示される項目を1行で表示されるようにします。</p> <p>st -pオプションと一緒に使用されます</p> <pre>st -p -j st -p -x -j st -p -d -j st -p -d -x -j</pre>
[ -tg ]	<p>TmaxGridの情報を照会します</p>

- 例

- サーバー・プロセスの情報 (-p)

以下は、サーバー・プロセスの情報を確認する例です。

```
– $$24 tmax8 (tmadm): st -p
```

CLH 0:

svr_name	svgname	spr_no	status	count	avg	svc
gw1	gw1	32	RDY	0	0.000	-1
gw2	gw2	33	RDY	0	0.000	-1
_hms01	hms01	34	RDY	2	0.000	-1
_hms01	hms01	105	RDY	0	0.000	-1
_hms01	hms01	106	RDY	0	0.000	-1
_hms01	hms01	107	RDY	0	0.000	-1
_rqsvg	rqsvg	35	RDY	0	0.000	-1
_rqsvg	rqsvg	109	RDY	0	0.000	-1
_rqsvg	rqsvg	110	RDY	0	0.000	-1
_rqsvg	rqsvg	111	RDY	0	0.000	-1
svr1	svg1	36	RDY	0	0.000	-1
svr2	svg1	37	RDY	0	0.000	-1
svr3	svg1	38	RDY	0	0.000	-1
svr_ucs	svg1	39	RDY	0	0.000	-1
svr_conv	svg1	40	RDY	0	0.000	-1
svr_rq	svg1	41	RDY	0	0.000	-1

svr_sq	svg1	42	RDY	0	0.000	-1
svr_hms	svg1	43	RDY	0	0.000	-1
fdltest	svg2	44	RDY	0	0.000	-1
sdltest	svg2	45	RDY	0	0.000	-1
-----						
TOTAL COUNT = 2						
TOTAL AVG = 0.000						
TOTAL RUNNING COUNT = 0						

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
svr_name	サーバー・プロセス名です
svgname	サーバー・プロセスが属しているサーバー・グループ名です
spr_no	各サーバー別プロセスのIDです
status	<p>サーバー・プロセスの現在状態です。</p> <p>以下は、システムの動作状態情報です</p> <ul style="list-style-type: none"> <li>– RDY : Ready状態です</li> <li>– NRDY : Non-Ready状態です。これは、tpstartして、クライアントがソケットで接続が確立された状態で、サーバーからデータが転送されない場合に発生することがあります。たとえば、ネットワーク状態がよくない場合に発生できます</li> <li>– RUN : 実行中の状態です</li> <li>– Unregistered : 未登録の状態で、クライアントへのtpendの直後にしばらく表示された後、状態が変更されます</li> <li>– BLK : 当該サービスの全プロセスがsuspendされます</li> <li>– PBLK : 当該サービスの一部プロセスがsuspendされます</li> <li>– UNADV : 当該サービスが属している全サーバー・プロセスがunadvertiseされた状態です</li> <li>– PUNADV : 当該サービスが属している一部サーバー・プロセスがunadvertiseされた状態です</li> </ul>
count	処理したサービス数です

項目	説明
avg	プロセスの平均待機時間です
svc	処理中のサービス名です。現在処理中のサービスがない場合、-1に表示されます

stコマンドはワイルドカードを使用できます。「st -p s\*」を入力すると、sで始まるsvr\_nameの情報のみ照会されます。

```

$$25 tmax8 (tmadm): st -p s*

CLH 0:
-----
svr_name      svgname      spr_no      status      count      avg      svc
-----
svr1          svg1             36          RDY          0          0.000     -1
svr2          svg1             37          RDY          0          0.000     -1
svr3          svg1             38          RDY          0          0.000     -1
svr_ucs       svg1             39          RDY          0          0.000     -1
svr_conv      svg1             40          RDY          0          0.000     -1
svr_rq        svg1             41          RDY          0          0.000     -1
svr_sq        svg1             42          RDY          0          0.000     -1
svr_hms       svg1             43          RDY          0          0.000     -1
sdltest       svg2             45          RDY          0          0.000     -1
-----

TOTAL COUNT = 0
TOTAL RUNNING COUNT = 0

```

#### – サービス情報(-s)

以下は、サービスの情報を確認する例です。



```
$$13 tmax8 (tmadm): st -s
```

```
CLH 0:
```

svc_name	svr_name	count	q_cnt	aq_cnt
q_avg	avg	status		
_hms01	_hms01	2	0	1
0.000	0.000	RDY		
TOLOWER	svr2	0	0	0
0.000	0.000	RDY		
TOUPPER	svr2	0	0	0
0.000	0.000	RDY		
HMS	svr_hms	0	0	0
0.000	0.000	RDY		
FDLTOLOWER	svr3	0	0	0
0.000	0.000	RDY		
FDLTOUPPER	svr3	0	0	0
0.000	0.000	RDY		
gw1	gw1	0	0	0
0.000	0.000	RDY		
gw2	gw2	0	0	0
0.000	0.000	RDY		
SDLTOLOWER	svr1	0	0	0
0.000	0.000	RDY		
SDLTOUPPER	svr1	0	0	0
0.000	0.000	RDY		
_rq1	_rqsvg	0	0	0
0.000	0.000	RDY		
LOGIN	svr_ucs	0	0	0
0.000	0.000	RDY		
TPDEQ	svr_rq	0	0	0
0.000	0.000	RDY		
TPENQ	svr_rq	0	0	0
0.000	0.000	RDY		
FDLDEL	fdltest	0	0	0
0.000	0.000	RDY		
SDLDEL	sdltest	0	0	0
0.000	0.000	RDY		
FDLSEL	fdltest	0	0	0
0.000	0.000	RDY		
FDLINS	fdltest	0	0	0
0.000	0.000	RDY		

TOUPPER_CONV	svr_conv	0	0	0
0.000 0.000 RDY				
SDLSEL	sdltest	0	0	0
0.000 0.000 RDY				
SDLINS	sdltest	0	0	0
0.000 0.000 RDY				
FDLUPT	fdltest	0	0	0
0.000 0.000 RDY				
SDLUPT	sdltest	0	0	0
0.000 0.000 RDY				
GET_SQ	svr_sq	0	0	0
0.000 0.000 RDY				

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
svc_name	サービス名です
svr_name	サービスが属しているサーバー名です
count	処理したサービス数です
avg	サービスの平均処理時間です
q_count	現在待機中のサービス要求数です
aq_count	一時でも待機されたことのあるサービス要求の総数です
q_avg	サービスの平均待機時間です
status	サービスの現在状態です。状態に対する詳細については、 <a href="#">「システム動作状態」</a> を参照してください

ワイルドカードを使用して「st -s \*W」を入力すると、W字で終わるサービスの情報のみを確認します。

```
$$27 tmax8 (tmadm): st -s *R
```

```
CLH 0:
```

svc_name	svr_name	count	q_cnt	aq_cnt
q_avg avg status				
TOLOWER	svr2	0	0	0
0.000 0.000 RDY				
TOUPPER	svr2	0	0	0
0.000 0.000 RDY				
FDLTOLOWER	svr3	0	0	0

0.000	0.000	RDY				
	FDLTOUPPER		svr3	0	0	0
0.000	0.000	RDY				
	SDLTOLOWER		svr1	0	0	0
0.000	0.000	RDY				
	SDLTOUPPER		svr1	0	0	0
0.000	0.000	RDY				

#### – TMS情報(-t)

以下は、TMSの情報を確認する例です。

```

$$29 tmax8 (tmadm): st -t

CLH 0:

-----

  tms_name      svgname      spr_no      status      count      avg      cqcount      aqcount
qavg

-----

  tms_ora      svg2          0          RDY          0          0.00          0          0
0.00
  tms_ora      svg2          1          RDY          0          0.00      ( 0 )      ( 0 )
(0.00)

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
tms_name	TMS名です
svgname	TMSが属しているサーバー・グループ名です
spr_no	TMSのプロセスIDです
status	TMSの現在状態であり、詳細は「 <a href="#">システム動作状態</a> 」を参照してください
count	処理したTMS件数です
avg	TMSの平均処理時間です
cqcount	現在待機中の要求数です
aqcount	少しでも待機した総サービス要求数です
qavg	サービスの平均待機時間です

TMSがマルチスレッドTMSの場合には、各スレッドの状態を照会できます。

```

$$1 tmax1 (tmadm): st -t -x
CLH 0:
-----
tms_name      svgname      spr_no(tid) status    count    avg    cqcount
              XID              xastate
-----
tms_ora_mt    svgora1          0      RUN      0    0.00    0
000:000:13505      commit
tms_ora_mt    svgora1          0( 1)  RDY      0%    0.00    ( 0)
000:000:00000      -
tms_ora_mt    svgora1          0( 2)  RDY      0%    0.00    ( 0)
000:000:00000      -
-----
tms_ora_mt    svgora1          1      RDY      0    0.00    ( 0)
000:000:00000      -
tms_ora_mt    svgora1          1( 1)  RDY      0%    0.00    ( 0)
000:000:00000      -
tms_ora_mt    svgora1          1( 2)  RDY      0%    0.00    ( 0)
000:000:00000      -
-----
tms_ora_mt    svgora2          10     RDY      0    0.00    0
000:000:00000      -
tms_ora_mt    svgora2          10( 1)  RDY      0%    0.00    ( 0)
000:000:00000      -
tms_ora_mt    svgora2          10( 2)  RDY      0%    0.00    ( 0)
000:000:00000      -
-----
tms_ora       svgora3          20     RDY      0    0.00    0
000:000:00000      -
-----
tms_ora       svgora3          21     RDY      0    0.00    ( 0)
000:000:00000      -
-----

```

#### – 詳細情報(-x)

以下は、「st -p」コマンドと一緒に使用した例です。

```

$$1 tmax8 (tmadm): st -p -x
CLH 0:
-----
svr_name      svgname      spr_no      status    count    avg    svc
              PID          RBS_TAG      fail_cnt  err_cnt  min_time  max_time
              utime      umin_time    umax_time  stime    smin_time  smax_time
-----
gw1           gw1           32          RDY      0        0.000    -1
10702         00000000     0           0        0        0.000    0.000

```

```

0.000      0.000      0.000      0.000      0.000      0.000
svr2      svg1      37      RDY      0      0.000      -1
10700      00000000      0      0      0.000      0.000
0.000      0.000      0.000      0.000      0.000      0.000
-----
TOTAL COUNT = 0
TOTAL SVCFAIL COUNT = 0
TOTAL ERROR COUNT = 0
TOTAL RUNNING COUNT = 0

```

以下は、「st -s」コマンドと一緒に使用した例です。

```

$$2 tmax8 (tmadm): st -s -x

CLH 0:

-----

svc_name      svr_name      count      q_cnt      aq_cnt      q_avg
  avg  status
                                fail_cnt  err_cnt
mintime  maxtime
                                utime umintime umaxtime      stime
smintime smaxtime

-----

TOLOWER      svr2      0      0      0      0.000
0.000      RDY
                                0      0
0.000      0.000
                                0.000      0.000      0.000      0.000
0.000      0.000
TOUPPER      svr2      0      0      0      0.000
0.000      RDY
                                0      0
0.000      0.000
                                0.000      0.000      0.000      0.000
0.000      0.000

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
PID	サーバ・プロセスのPIDです
fail_cnt	サーバー・プロセス・サービスのFAIL COUNTです
err_cnt	サーバー・プロセス・サービスのERROR COUNTです
mintime	サービスを処理した最小時間です

項目	説明
maxtime	サービスを処理した最大時間です
utime	サーバー・プロセスで毎回の処理時に使われたユーザー・タイムです
umin_time	ユーザー・タイムの最小使用時間です
umax_time	ユーザー・タイムの最大使用時間です
stime	サーバー・プロセスで毎回のサービス処理時に使われたシステム・タイムです
smin_time	システム・タイムの最小使用時間です
smax_time	システム・タイムの最大使用時間です

#### – 照会結果のソート(-o)

以下、st -p コマンドに、[-o] オプションを使用した例です。

```

$$1 tmax1 (tmadm): st -p -o ca
CLH 0 :
- - - - -
svr_name   svpname       spr_no     status     count     avg       svc
- - - - -
svr2       svg1         37        RDY        10        0.000     -1
svr1       svg1         36        RDY         6        0.000     -1
svr3       svg1         38        RDY         2        0.000     -1
TOTAL COUNT = 18
TOTAL AVG = 0.000
TOTAL RUNNING COUNT = 0

```

以下は、st -s コマンドに[-o] オプションを使用した例です。

```

$$1 tmax1 (tmadm): st -s -o ca
CLH 0:
- - - - -
svc_name   svr_name   count   avg   cq_count   aq_count   q_avg   status
- - - - -
TOUPPER    svr2        10     0.000   0         0         0.000   RDY
SDLTOUPPER svr1         5     0.000   0         0         0.000   RDY
FDLTOUPPER svr3         2     0.000   0         0         0.000   RDY
SDLTOLOWER svr1         1     0.000   0         0         0.000   RDY
FDLTOLOWER svr3         0     0.000   0         0         0.000   RDY
TOLOWER    svr2         0     0.000   0         0         0.000   RDY

```

- 降順でソートする条件

条件	説明
ca	処理件数(count)

条件	説明
aa	平均処理時間(avg)
cq	現在のキューイング数(cq_count)
aq	平均のキューイング数(aq_count)
qa	平均のキューイング時間(q_avg)
ce	各サーバーの処理件数を基準にしてサーバー別にソート
ae	各サーバーの平均処理時間を基準にしてサーバー別にソート

- 昇順でソートする条件

条件	説明
ca-	処理件数(count)
aa-	平均処理時間(avg)
cq-	現在のキューイング数(cq_count)
aq-	平均のキューイング数(aq_count)
qa-	平均のキューイング時間(q_avg)
ce-	各サーバーの処理件数を基準にしてサーバー別にソート
ae-	各サーバーの平均処理時間を基準にしてサーバー別にソート

## – 出力メッセージのライン数の指定(-n)

以下は、出力メッセージのライン数を2に設定した例です。

```

$$1 tmax1 (tmadm): st -p -o ca -n 2
CLH 0 :
- - - - -
svr_name  svaname      spr_no   status    count    avg      svc
- - - - -
svr2      svg1         37      RDY       10      0.000    -1
svr1      svg1         36      RDY       6       0.000    -1
TOTAL COUNT = 18
TOTAL AVG = 0.000
TOTAL RUNNING COUNT = 0

```

以下は、出力メッセージのライン数を3に設定した例です。

```

$$1 tmax1 (tmadm): st -s -o ca -n 3
CLH 0:
- - - - -
svc_name svr_name count  avg   cq_count  aq_count q_avg  status
- - - - -

```

TOUPPER	svr2	10	0.000	0	0	0.000	RDY
SDLTOUPPER	svr1	5	0.000	0	0	0.000	RDY
FDLTOUPPER	svr3	2	0.000	0	0	0.000	RDY

#### – HMSデスティネーション情報(-q)

以下は、HMSデスティネーション情報を確認する例題です。

```

$$1 tmax1 (tmadm): st -q
-----
G  dest      cqcount  type    apqcnt   acqcnt   f_dscrd  t_dscrd  cons_cnt  prod_cnt
-----
-  queue01    58    QUEUE   169      111      0         0        30
5
-  topic01    32    TOPIC   42       283      0         0        28
3

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
G	デスティネーションがGLOBALに設定された場合は「O」が表示され、GLOBALでない場合はハイフン(-)が表示されます
dest	デスティネーションの名前です。環境設定ファイルのHMSセクションに設定された名前が表示されます
cqcount	デスティネーションでまだ処理されていないメッセージ数です
type	デスティネーションのタイプを示します。QUEUEとTOPICで表示されます
apqcnt	現在まで送信されて溜まっているメッセージの総数です
acqcnt	コンシューマーによって処理されたメッセージの総数です。トピック・タイプは、1つのメッセージに対してすべてのコンシューマーが受信しないと処理が完了しないため、apqcntよりacqcntが大きくなることがあります
f_dscrd	受信したメッセージに対してfailed処理し、受信に失敗したメッセージ数です
t_dscrd	メッセージの送信にTTLが設定されている場合、有効時間が過ぎて失敗したメッセージ数です
cons_cnt	デスティネーションに接続したコンシューマーの数です
prod_cnt	デスティネーションに接続したプロデューサーの数です

以下は、特定のデスティネーションに対するクライアント詳細情報を照会する例です。

```

$$1 tmax1 (tmadm): st -q queue01 -c
-----
      sesi      clid      cname      clitype  qcnt   cnt   f_dscrd  t_dscrd      listener
-----

```



0	0x39d	prodasync	ARCV	49	0	0	0	ASYNCSVC
0x1	0	prod41	SND	0	32	0	0	
0x1	0	prod42	SND	0	11	0	0	
0x1	0	cons51	RCV	3	0	0	0	
0x1	0	cons52	RCV	9	0	0	0	
<pre> \$\$\$2 tmax1 (tmadm): st -q topic01 -c </pre>								
sesi	clid	cname	clitype	qcnt	cnt	f_dscrd	t_dscrd	listener
0x1	0	prod11	PUB	0	2	0	0	
0x1	0	prod12	PUB	0	1	0	0	
0x1	0	cons11	SUB	30	0	0	0	
0x2	0x2	prod21	PUB	0	16	0	0	
0x2	0x2	prod22	PUB	0	8	0	0	
0x2	0x2	cons21	SUB	23	0	0	0	
0x3	0	prod31	PUB	0	3	0	0	
0x4	0x4	cons41	ADSUB	32	0	0	0	ASYNCSVC2

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
sesi	クライアントを作成したセッションの番号です
clid	クライアントIDです
cname	クライアント名です
clitype	デスティネーションに接続されたクライアントの種類です。 – SND : キュータイプの送信者(hms_create_sender())で生成)

項目	説明
	<ul style="list-style-type: none"> <li>– PUB : トピックタイプの発行者(hms_create_publisher())で生成)</li> <li>– RCV : キュータイプの受信者(hms_create_receiver())で生成)</li> <li>– SUB : トピックタイプのサブスクライバー(hms_create_subscriber())で生成)</li> <li>– DSUB : トピックタイプの永続サブスクライバー(hms_create_durable_subscriber())で生成)</li> <li>– ARCV : Asyncセッションで生成されたキュータイプの受信者</li> <li>– ASUB : Asyncセッションで生成されたトピックタイプのサブスクライバー</li> <li>– ADSUB : Asyncセッションで生成されたトピックタイプの永続サブスクライバー</li> </ul>
qcnt	各サブスクライバーが実際に持っているメッセージ数です
cnt	各クライアントが送信または受信したメッセージ数です
f_dscrd	受信後メッセージをfailedに処理し、失敗したメッセージ数です.
t_dscrd	送信時に設定したTTL(有効時間)が過ぎて失敗処理されたメッセージ数です
listener	ASYNcセッションで作成したコンシューマーの場合、メッセージを受信するサービス名です

以下は、TmaxGridの詳細情報を照会する例です。

```

$$1 tmax1 (tmadm): st -tg
TG Info:
  Preferences:
    shmkey           : 38852 (0x97c4)
    shmsize          : 40960 Kbyte
    max data         : 10000
    max watcher      : 10
    id               : 1
    portno           : 9999 (UDP)
    heartbeat        : 1000 ms
    timeout           : 5000 ms
    historycount      : 100
    standbycount     : 10
    gqmaxbuffersize  : 1000 byte
    gqdownwaittime   : 30
  Status:
    status           : INIT
    suspend status    : NOT SUSPENDED
    created node count : 0
    deleted node count : 0
    set data count    : 0

```

```

get data count      : 0
current node count  : 0
temporary node count : 0
watcher count       : 0
current data count   : 0
total data size      : 0 byte
current shm size     : 4460544 byte (10.64%)

```

## 5.4.2. gwinfo

tmadminでリモート・ゲートウェイとの接続状態を確認するコマンドです。また、現在接続されているノード(メインノードまたはバックアップノード)の詳しい情報を確認することができます。

全ゲートウェイの接続情報を確認することができます。(ただし、Webサービス・ゲートウェイはサポートしていません)

- 使用方法

- `$$1 tmax1 (tmadm): gwinfo [-w gw_name] [-t gw_type]`

項目	説明
<code>[-w gw_name]</code>	特定のゲートウェイ情報のみを照会します。gw_nameはGATEWAYセクションに設定した名前です
<code>[-t gw_type]</code>	特定のゲートウェイ種類のみを照会します。  GATEWAYセクションに定義するTYPE(TMAX、TMAXNONTX、JEUS、JEUS_ASYNC、TUXEDO、TUXEDO_ASYNC、XAGW)で入力します(例: TMAX)

- 例

以下は、gwinfoコマンドの使用例です。接続されているノード(メインノードまたはバックアップノード)のタイプに応じて照会結果は異なります。

– 以下は、メインノードに接続されている場合の例です。

```

10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): gwinfo
-----
gw_no   channel   type      foreign_address      status

```

0	OUTCH	PRIM	tmaxc1(192.168.1.12:9400)	RDY
1	OUTCH	PRIM	tmaxc1(192.168.1.12:9400)	RDY
4	INCH	-	tmaxc1(192.168.1.12:1434)	RDY
5	INCH	-	tmaxc1(192.168.1.12:1436)	RDY
6	INCH	-	tmaxh2(192.168.1.48:58094)	RDY
7	INCH	-	tmaxh2(192.168.1.48:58093)	RDY

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
gw_no	ゲートウェイ番号です
channel	チャンネルのタイプ(INBOUNDあるいはOUTBOUND)を示します
type	現在接続されているノードのタイプ(メインノードまたはバックアップノード)を示します
foreign_address	接続されているリモート・ゲートウェイのIPアドレスです
status	webt、jtmax、webtasyncとの接続状態です。状態についての詳細説明は、 <a href="#">「システム動作状態」</a> を参照してください

– 以下は、バックアップノードに接続されている場合の例です。

```

$$1 tmax1 (tmadm): gwinfo

```

gw_no	channel	type	foreign_address	status
0	OUTCH	BACK	tmaxh2(192.168.1.48:9200)	RDY
1	OUTCH	BACK	tmaxh2(192.168.1.48:9200)	RDY
6	INCH	-	tmaxh2(192.168.1.48:58091)	RDY
7	INCH	-	tmaxh2(192.168.1.48:58092)	RDY

### 5.4.3. txgwinfo / nontxgwinfo

tmadminでリモート・ゲートウェイとの接続状態を確認できるコマンドです。また、現在接続されているノード(メインノードまたはバックアップノードなど)の詳しい情報を確認することができます。txgwinfo(txgwi)はTmaxゲートウェイの情報を確認し、nontxgwinfo(ntxgwi)はTmax非トランザクション(TMAXNONTX)ゲートウェイの情報を確認します。

- 使用方法
- – txgwinfo

```

$$1 tmax1 (tmadm): txgwinfo

```

– nontxgwinfo

```
$$1 tmax1 (tmadm): nontxgwinfo
```

- 例

以下は、コマンドの使用例です。

– txgwinfo

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): txgwinfo

-----
gw_no   channel   type    foreign_address      status
-----
0       OUTCH      PRIM    tmaxc1(192.168.1.12:9400)  RDY
1       OUTCH      PRIM    tmaxc1(192.168.1.12:9400)  RDY
4       INCH       -       tmaxc1(192.168.1.12:1434)  RDY
5       INCH       -       tmaxc1(192.168.1.12:1436)  RDY
6       INCH       -       tmaxh2(192.168.1.48:58094)  RDY
7       INCH       -       tmaxh2(192.168.1.48:58093)  RDY
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
gw_no	ゲートウェイ番号です
channel	チャンネルのタイプ(INBOUNDあるいはOUTBOUND)を示します
type	現在接続されているノードのタイプ(メインノードまたはバックアップノード)を示します
foreign_address	接続されているリモート・ゲートウェイのIPアドレスです
status	webt、jimax、webtasyncとの接続状態です。状態についての詳細説明は、「 <a href="#">システム動作状態</a> 」を参照してください

– nontxgwinfo

以下は、バックアップノードに接続されている場合の例です。

```
$$1 tmax1 (tmadm): nontxgwinfo

-----
gw_no   channel   type    foreign_address      status
-----
0       OUTCH      BACK    tmaxh2(192.168.1.48:9200)  RDY
1       OUTCH      BACK    tmaxh2(192.168.1.48:9200)  RDY
```

6	INCH	-	tmaxh2(192.168.1.48:58091)	RDY
7	INCH	-	tmaxh2(192.168.1.48:58092)	RDY

## 5.4.4. jgwinfo / ajgwinfo

tmadminでJTmaxまたはWebtAsyncとの接続状態を確認するコマンドです。また、現在接続されているノード(メインノードまたはバックアップノードなど)の詳しい情報を確認することができます。jgwinfoは、Java(JEUS)ゲートウェイの情報を確認し、ajgwinfoはAsync Java(ASYNC\_JEUS)ゲートウェイの情報を確認します。

### ● 使用方法

#### – jgwinfo

```
$$1 tmax1 (tmadm): jgwinfo
```

#### – ajgwinfo

```
$$1 tmax1 (tmadm): ajgwinfo
```

### ● 例

#### – jgwinfo

以下は、コマンドの使用例です。

```
10/user2/starbj81>tmadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxs1 (tmadm): jgwinfo
-----
gw_no name                channel type  foreign_address          status
-----
0 gw1                    OUTCH  PRIM   unknown(192.168.35.47:6555)  RDY
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
gw_no	ゲートウェイ番号です
name	ゲートウェイ名です
channel	チャンネルのタイプ(INBOUNDあるいはOUTBOUND)を示します
type	現在接続されているノードのタイプ(メインノードまたはバックアップノード)を示します
foreign_address	接続されているリモート・ゲートウェイのIPアドレスです

項目	説明
status	webt、jtxmax、webtasyncとの接続状態です。状態についての詳細説明は、「システム動作状態」を参照してください

#### – ajgwinfo

以下は、バックアップ・ノードに接続されている場合の例です。

```
10/user2/starbj81>tmaxadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmax1 (tmadm): ajgwinfo

-----
gw_no name          channel type  foreign_address          status
-----
1 gw2              OUTCH  BACK   unknown(192.168.35.47:6555)  RDY
```

## 5.4.5. wsgwinfo

tmaxadminでWebサービス・ゲートウェイの状態を確認するコマンドです。

#### ● 使用方法

```
$$1 tmax1 (tmadm): wsgwinfo [-i svrid[ svrid]]
```

項目	説明
[-i svrid [ svrid]]	確認するsvridリストを入力します

#### ● 例

以下は、コマンドの使用例です。

```
10/user2/starbj81>tmaxadmin
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmax1 (tmadm): wsgwinfo

-----
svrid    load_time          current_client    max_attach_time
-----
2        Wed Jan 13 12:43:39 2010      0                0.0sec
-----
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
svrid	ゲートウェイのsvr indexです
load_time	Webサービス・ゲートウェイが起動された時間です
current_client	現在接続中のクライアント数です
max_attach_time	1つのサービスを処理するまでにかかった最長時間です

## 5.4.6. smtrc

環境ファイルのNODEセクションのSMSUPPORTが「Y」に設定されている場合、「st -p -x」を実行すると実行中のサービスに対応するGIDを出力します。

- 使用方法

```
$$1 tmax1 (tmadm): smtrc [-a] GID0 GID1
```

項目	説明
[-a]	既存の情報のほか、サーバー・プロセス・インデックス(spri)、ユーザーCPU使用時間、システムCPU使用時間、リターン情報などをすべて表示します
GID0	SysMasterのGID上位4バイトを16進法(Hexa decimal)で表記します
GID1	SysMasterのGID下位4バイトを16進法(Hexa decimal)で表記します

- 例

以下は、「st ?p ?x」を実行した後、サービスに対応するGIDを出力するためのsmtrcを実行する例です。

```
$$1 tmax1 (tmadm): st ?p -x
```

```
CLH 0:
```

```
-----
svr_name      svgname      spr_no  status   count    avg      svc
              PID        fail_cnt err_cnt  min_time max_time
              SysMaster_GID
-----
  evtsvr      svg1         36     RDY      0        0.000   -1
              17980         0        0        0.000   0.000
              00000000-00000000-00000000
  svr1        svg1         37     RUN      0        0.000   SDLTOUPPER
              17981         0        0        0.000   0.000
              00000000-00000101-00000000
  svr2        svg1         38     RUN      0        0.000   SDLTOUPPER2
              17982         0        0        0.000   0.000
              00000000-00080101-00000000
  svr3        svg1         39     RUN      0        0.000   SDLTOUPPER3
```



```

17983      0      0      0.000      0.000
00000000-00100101-00000000
svr_sys    svgl      40      RDY      3      0.000      -1
17984      0      0      0.000      0.000
00000000-00000000-00000000

-----

TOTAL COUNT = 3
TOTAL SVCFAIL COUNT = 0
TOTAL ERROR COUNT = 0
TOTAL AVG = 0.000
TOTAL RUNNING COUNT = 3

$$$1 tmaxl (tmadm): smtrc 0 0101
CLH 0:

-----

sysmaster_global_id      status      svc_name
-----
00000000:00000101:00000000      SVC_RUNNING      SDLTOUPPER2

45670701 tmaxil (tmadm): smtrc -a 0 0101
CLH 0:

-----

sysmaster_global_id      status      svc_name
ctime      svctime      spri      ucpu      scpu
-----
00000000:00000101:00000000      SVC_RUNNING      SDLTOUPPER2
14:05:32:159      0.000      38      0.000      0.000
00000000:00000101:00010000      SVC_RUNNING      SMTRACE
14:05:32:159      0.000      40      0.000      0.000
00000000:00000101:00010000      SVC_DONE      SMTRACE
14:05:32:159      0.000      40      0.000      0.000

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
sysmaster_global_id	SysMasterのgidです
status	サービス状態です
svc_name	サービス名です
ctime	ログを生成した時間です
svctime	サービスの実行時間です
spri	サービスを実行したサーバー・プロセス・インデックスです
ucpu	ユーザーCPU時間
scpu	システムCPU時間

## 5.4.7. clhsinfo

clhsinfoはマルチノード環境でそれぞれのCLH間の接続状態の情報を確認します。マルチノード環境では互いに異なるノード間の接続が必要ですが、ファイアーウォールや装備の老朽化によってネットワークが不安定なときはノード間の接続が切断されて正常なサービスが実行されない場合があります。そのため、このような問題を確認できるコマンド(clhsinfo、tmmsinfo)が追加されました。

- 使用方法

```
$$1 tmax1 (tmadm): clhsinfo
```

- 例

以下は、「minclh = 1」、「maxclh = 2」に設定した場合、clhsinfoを使用した例です。

```
tmaxh4@starbj81:/EMC01/starbj81/tmax/config>tmadmin
TMADMIN for rnode (tmaxh2): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh4 (tmadm): clhsinfo

CLH 0:
-----
  nodename      clhno      cpc      status
-----
  tmaxh2         0         2      RDY
  tmaxh2         1         0      NRDY
-----

CLH 1 is not available
Msg from rnode(tmaxh2):
CLH 0:
-----
  nodename      clhno      cpc      status
-----
  tmaxh4         0         2      RDY
  tmaxh4         1         0      NRDY
-----

CLH 1 is not available
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
nodename	ノード名です
clhno	CLH番号です

項目	説明
cpc	CPC番号です
status	<p>ノードの状態です。</p> <p>以下は、ノードの状態値の説明です。</p> <ul style="list-style-type: none"> <li>– NRDY(NOT_READY) : 対象のCLHと接続されていないか、接続が切断された状態です</li> <li>– RDY(READY) : 正常に接続されている状態です</li> <li>– REG(REGISTERED) : ノード間の接続を進行中の状態です</li> <li>– UNR(UNREGISTERED) : ノードがtmdown要求で終了された状態です</li> <li>– NSTRT(NCLHSTARTED) : ノード間接続中の状態で、対象ノードのCLHが開始された状態です</li> <li>– CTNG(TRYINGTOCONNECT) : ノード間TCP/IPソケットのみ接続され、対象ノードに接続メッセージを送っていない状態です</li> <li>– NPING(ALIVECHECK) : ノードが接続されている状態で、アライブ・チェックを実行している状態です</li> <li>– NPING2(ALIVECHECK2) : ノードからアライブ・チェックについての応答が送られていない状態です</li> </ul>

## 5.4.8. tmmsinfo

tmmsinfoコマンドは、マルチノード環境でTMM間の接続状態の情報を確認できます。

### ● 使用方法

```
$$1 tmax1 (tmadm): tmmsinfo
```

### ● 例

以下は、コマンドの使用例です。

```

tmaxh4@starbj81:/EMC01/starbj81/tmax/config>tmdadmin
TMADMIN for rnode (tmaxh2): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh4 (tmadm): tmmsinfo

-----
no      nodename      livetime      status
-----

```

```

1    tmaxh2          13:53:53    RDY
Msg from rnode(tmaxh2):
-----
no    nodename      livetime    status
-----
0    tmaxh4          13:53:08    RDY

```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
no	tmm番号です
nodename	ノード名です
livetime	最終のチャンネル使用時間です
status	ノードの状態です。ノード状態の詳細については、「 <a href="#">ノード動作状態</a> 」を参照してください

- 

## 5.4.9. repeat(r)

repeatは状態情報の繰り返し表示機能であり、以下の方式でも使用できます。反復的なコマンドの実行は状態情報をモニタリングするだけでなく、業務実行のデバッグにも大いに役に立ちます。

### • 使用方法

```
$ $1 tmax1 (tmadm): r -k n -i ms 反復的なコマンドの実行
```

項目	説明
-k n	n回繰り返し実行します
-i ms	msミリ秒間隔で繰り返し実行します
反復実行コマンド	繰り返す対象コマンドを指定します

### • 例

以下は、5秒の間隔で「st -s」を30回実行することを繰り返す例です。

```
$ $1 tmax1 (tmadm): r -k 30 -i 5000 st -s
```

以下は、5秒の間隔で「st -p」を30秒間実行することを繰り返す例です。

```
$ $1 tmax1 (tmadm): r -k 30 -i 5000 st -p
```

## 5.4.10. clientinfo

現在接続されているクライアントの環境情報を照会します。

- 使用方法

```
$$1 tmax1 (tmadm): ci [-s]
```

項目	説明
[-s]	接続されているクライアントの総数を把握できます

- 例

以下は、コマンドの使用例です。

```
$$1 tmax1 (tmadm): ci
CLH 0:
-----
cli_id    status    count    lastin_time    ipaddr    username
-----
0         RDY        0        20             61.77.153.1
1         RDY        2        10             61.77.153.1    tmax
2         RDY        4        123            61.77.153.1
-----
Total Connected Clients = 1
-----
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
cli_id	クライアントIDです
status	クライアントの現在状態です。  以下は、status項目で確認できる状態値の説明です <ul style="list-style-type: none"><li>– RDY : クライアントが正常にCLHに接続されている状態です</li><li>– NRDY : CLHと接続を確立しているクライアントのうちTCP/IPソケットのみ接続され、正常に接続メッセージ(TPSTARTメッセージ)を送っていない状態です</li><li>– QED : クライアントのサービス要求がサーバー・キューに溜まっている状態です</li><li>– CTING : クライアントがCLHに接続中の状態です</li><li>– UNR : クライアントがtpend()のプロセス中か、NCLHがダウン時の状態です</li><li>– RUN : クライアントが要求したサービスが実行中の状態です</li></ul>

項目	説明
count	サービス処理件数です
lastin_time	クライアントがサービスの呼び出し後、待機する時間です
ipaddr	クライアントの接続IPアドレスです
username	TPSTART_T構造体のusernameフィールドに定義したユーザー名です

異常に接続されたクライアントがCLHIに無限に接続されている場合、他のクライアントが接続できない現象が発生することがあります。CLHでソケット接続以後60秒以内にTPSTART接続メッセージが受信されないクライアントに対しては自動的に接続を終了します。クライアント接続を終了するとき、以下のようなエラーが発生します。

```
(I) CLH0209 internal error : disconnect client because client didn't send tpstart
msg for 60 sec.(192.168.1.43) [CLH0058]
```

## 5.4.11. svrinfo(si)

現在動作中の各サーバーの情報を表示します。

- 使用方法

```
$$1 tmax1 (tmadm): si
```

- 例

以下は、コマンドの使用例です。

```
$$5 tmaxs1 (tmadm): si
- - - - -
clh   svrname   (svri)   status   count   qcount   qpcount   emcount
- - - - -
0     scoresd1  ( 0)     NRDY     0       0       0       0
0     bank2    ( 1)     RDY      0       0       0       0
0     svr1     ( 2)     RDY      0       0       0       0
0     svr2     ( 2)     RDY      0       0       0       0
0     svr3     ( 2)     NRDY     0       0       0       0
0     api      ( 2)     RDY      0       0       0       0
0     syncrtn  ( 2)     RDY      0       0       0       0
0     ucs      ( 2)     NRDY     0       0       0       0
0     ucssvr   ( 2)     RDY      0       0       0       0
0     broad    ( 2)     NRDY     0       0       0       0
0     selins   ( 2)     NRDY     0       0       0       0
```

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
clh	CLH番号です
svrname	サーバー名です
(svri)	サーバーIDです
status	サーバーの現在状態です。詳細は、「システム動作状態」を参照してください
count	処理件数です
qcount	サーバーのキューイング数です
qpcount	キューから削除された数です
emcount	最大のキューイング数(maxqcount)を超過して返された数です
boottime	サーバーの最終起動時間です。-bオプションを使用した場合に出力されます

## 5.4.12. txquery(txq)

現在処理中のトランザクション情報を表示します。

- 使用方法

```

$$1 tmax1 (tmdm): txquery(txq) [-x] [-g svgrname] [-w [-r] [-G gwname]]
                        [<upper-global-xid><lower-global-xid>]

```

項目	説明
[-x]	トランザクション・ブランチ別の詳細情報を照会します
[-g svgrname]	svgrnameのグループに対するトランザクション・ブランチを照会します
[-w]	ドメイン・サブトランザクション・ツリーを照会します
[-r]	リモート・ドメインのXIDで検索します
[-G gwname]	gwnameのドメイン・ゲートウェイを照会します
upper-global-xid	検査するxidのgtidの前の4バイトです
lower-global-xid	検査するxidのgtidの後ろの4バイトです

- 例

- トランザクションの照会

以下は、トランザクションを照会する例です。

```

-- $$14 tmax1 (tmdm): txq
-- CLH0:
--

```

cli_id	txstime	txqcount	txrcount	XID	txstate	xastate
c0023	8:33:34	1	0	000:000:00022	TXBEGIN	TX_OK
c0024	18:33:34	1	0	000:000:00023	TXBEGIN	TX_OK
c0025	18:33:34	1	0	000:000:00024	TXBEGIN	TX_OK
c0026	18:33:34	1	0	000:000:00025	TXBEGIN	TX_OK
c0027	18:33:34	1	0	000:000:00026	TXBEGIN	TX_OK
c0028	18:33:34	1	0	000:000:00027	TXBEGIN	TX_OK
c0029	18:33:34	1	0	000:000:00028	TXBEGIN	TX_OK
c0030	18:33:34	1	0	000:000:00029	TXBEGIN	TX_OK
c0031	18:33:34	1	0	000:000:00030	TXBEGIN	TX_OK

以下は、コマンドの実行時に出力される項目の説明です。

項目	説明
cli_id	現在接続されて処理中のクライアント識別番号です
txstime	トランザクション処理の開始時間です
txqcount	現在トランザクション・キューに蓄積されているトランザクション・キューイング数です
txrcount	トランザクションの処理数です
XID	トランザクションの識別番号です
txstate	トランザクションの処理状態です(TXBEGIN、TXCOMMIT、TXROLLBACK)
xastate	XA処理状態です

## – トランザクション情報の照会(-x)

以下は、トランザクション情報を照会する例です。

```

$$6 tmaxh4 (tmadm): txquery -x
CLH 0:
-----
cli_id  clid   txstime txqcount txrcount   XID                txstate
xastate bqualno  nodename      svgname    txbstate    xabstate
-----
  c1526 0x000005f6 15:13:35  4        0    34800000:0008c087 TXBEGIN
TX_OK   (B)00000000 tmaxh4    svg12301X  BEGIN      XA_OK
        (B)00000001 tmaxh4    gw2301X   BEGIN      XA_OK
        (B)00000002 tmaxh4    gw2302X   BEGIN      XA_OK
        (B)00000003 tmaxh4    svg12302X BEGIN      XA_OK
  c1527 0x000005f7 15:13:35  4        0    34800000:0008c089 TXBEGIN
TX_OK   (B)00000000 tmaxh4    svg12301X  BEGIN      XA_OK
        (B)00000001 tmaxh4    gw2301X   BEGIN      XA_OK
        (B)00000002 tmaxh4    gw2302X   BEGIN      XA_OK
        (B)00000003 tmaxh4    svg12302X BEGIN      XA_OK

```



– グループのトランザクション照会(-g)

以下は、グループのトランザクションを照会する例です。

```

$$7 tmaxh4 (tmadm): txq -g svg12301X

CLH 1:
-----
cli_id      clid      txstime txqcount  txrcount      XID      txstate
xastate bqualno      nodename      svgname      txbstate      xabstate
-----
c1595 0x0000463b 15:14:10 1          0  34800001:0008e953 TXBEGIN
TX_OK  (B)00000000      tmaxh4      svg12301X      BEGIN          XA_OK
c1596 0x0000463c 15:14:10 1          0  34800001:0008e956 TXBEGIN
TX_OK  (B)00000000      tmaxh4      svg12301X      BEGIN          XA_OK
c1597 0x0000463d 15:14:10 1          0  34800001:0008e957 TXBEGIN
TX_OK  (B)00000000      tmaxh4      svg12301X      BEGIN          XA_OK

```

– トランザクション・ツリーの照会(-w)

以下は、トランザクション・ツリーを照会する例です。

```

$$9 tmaxh4 (tmadm): txquery -w

CLH 0:
-----
gw_no txstime txqcount txrcount      XID      RXID      txstate xastate
-----
g0004 17:00:00 2          0  34800000:0008c38e 39800000:0003bcf7 PHASE2 TX_OK
g0004 17:00:00 2          0  34800000:0008c38f 39800000:0003bcf8 PHASE1 TX_OK
g0004 17:00:00 2          0  34800000:0008c391 39800000:0003bcf9 PHASE2 TX_OK
g0004 17:00:00 2          0  34800000:0008c39a 39800000:0003bd00 PHASE1 TX_OK
g0004 17:00:00 2          0  34800000:0008c3a1 39800000:0003bd08 PHASE1 TX_OK
g0004 17:00:00 1          0  34800000:0008c3a3 39800000:0003bd09 sTXBEGIN TX_OK
g0005 17:00:00 2          0  34800000:0008c38b 3e800100:000177f6 PHASE2 TX_OK
g0005 17:00:00 2          0  34800000:0008c38d 3e800101:00014ff7 PENDING TX_OK
g0005 17:00:00 2          0  34800000:0008c390 3e800101:00014ff8 PENDING TX_OK
g0005 17:00:00 2          0  34800000:0008c392 3e800000:0006e050 PHASE1 TX_OK
g0005 17:00:00 2          0  34800000:0008c396 3e800000:0006e052 PHASE1 TX_OK
g0005 17:00:00 2          0  34800000:0008c397 3e800100:000177f8 PHASE1 TX_OK
g0005 17:00:00 2          0  34800000:0008c398 3e800000:0006e055 PHASE1 TX_OK
g0005 17:00:00 2          0  34800000:0008c399 3e800101:00014ff9 sTXBEGIN TX_OK
g0005 17:00:00 2          0  34800000:0008c39c 3e800100:000177f9 PENDING TX_OK
g0005 17:00:00 2          0  34800000:0008c39d 3e800000:0006e05a sTXBEGIN TX_OK
g0005 17:00:00 2          0  34800000:0008c39e 3e800101:00014ffa PHASE1 TX_OK
g0005 17:00:00 1          0  34800000:0008c39f 3e800000:0006e05b sTXBEGIN TX_OK
g0005 17:00:00 1          0  34800000:0008c3a2 3e800100:000177fa sTXBEGIN TX_OK
g0004 17:00:00 2          0  34800000:0008eb51 39800001:0003e50d PHASE2 TX_OK
g0004 17:00:00 2          0  34800000:0008eb53 39800001:0003e510 PHASE2 TX_OK
g0004 17:00:00 2          0  34800000:0008eb58 39800001:0003e516 PHASE1 TX_OK

```

```

g0004 17:00:00      2      0 34800000:0008eb5b 39800001:0003e519 sTXBEGIN TX_OK
g0005 17:00:00      2      0 34800000:0008eb4d 3e800001:0007084f PHASE2 TX_OK
g0005 17:00:00      2      0 34800000:0008eb50 3e800001:00070852 PHASE2 TX_OK
g0005 17:00:00      2      0 34800000:0008eb52 3e800001:00070854 PHASE2 TX_OK
g0005 17:00:00      1      0 34800000:0008eb56 3e800001:00070858 sTXBEGIN TX_OK
g0005 17:00:00      1      0 34800000:0008eb5a 3e800101:00014ffb sTXBEGIN TX_OK
g0005 17:00:00      1      0 34800000:0008eb5c 3e800001:0007085b sTXBEGIN TX_OK

```

以下は、[-w]、[-r]オプションを使用する例です。

```

### txquery -w -r ###
$$$11 tmaxh4 (tmadm): txquery -w -r

CLH 1:
-----
  gw_no txstime txqcount txrcount      XID      RXID      txstate xastate
-----
  g0004 17:00:00      2      0 34800001:0008ed0a 39800001:0003e6bc PENDING TX_OK
  g0004 17:00:00      2      0 34800001:0008ed0b 39800001:0003e6bd PENDING TX_OK
  g0004 17:00:00      2      0 34800001:0008ed0c 39800001:0003e6be PHASE1 TX_OK
  g0004 17:00:00      2      0 34800001:0008ed0f 39800001:0003e6c1 PHASE1 TX_OK
  g0005 17:00:00      2      0 34800001:0008ed00 3e800001:000709aa PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008ed03 3e800101:00015046 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008ed04 3e800100:00017846 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008ed05 3e800001:000709b1 PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008ed06 3e800100:00017847 PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008ed07 3e800001:000709b2 sTXBEGIN TX_OK
  g0005 17:00:00      2      0 34800001:0008ed08 3e800100:00017848 PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008ed09 3e800100:00017849 PHASE2 TX_OK
  g0004 17:00:00      2      0 34800001:0008c53f 39800000:0003bea9 PHASE2 TX_OK
  g0004 17:00:00      2      0 34800001:0008c540 39800000:0003beab PHASE2 TX_OK
  g0004 17:00:00      2      0 34800001:0008c545 39800000:0003beaf PHASE1 TX_OK
  g0004 17:00:00      2      0 34800001:0008c546 39800000:0003beb0 PHASE1 TX_OK
  g0004 17:00:00      2      0 34800001:0008c548 39800000:0003beb2 sTXBEGIN TX_OK
  g0005 17:00:00      2      0 34800001:0008c51c 3e800000:0006e192 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008c536 3e800000:0006e1a5 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008c537 3e800100:00017845 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008c539 3e800000:0006e1a7 PHASE2 TX_OK
  g0005 17:00:00      2      0 34800001:0008c53e 3e800000:0006e1ad PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008c541 3e800101:00015047 PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008c542 3e800000:0006e1af PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008c543 3e800101:00015048 PENDING TX_OK
  g0005 17:00:00      2      0 34800001:0008c544 3e800101:00015049 PHASE1 TX_OK

```

## 5.4.13. rqstat(rqs)

現在使用できるRQの状態を示したり、ディスク・キューに蓄積されているサービスを処理したりできます。

- 使用方法

```
$$1 tmax1 (tmadm): rqs -l [-s rqname] [-f rqname] [-c rqname] [-a]
```

項目	説明
[-l]	使用可能なRQを示します
[-s rqname]	RQの状態を表示します
[-f rqname]	RQに蓄積されているサービスを処理します
[-c rqname]	停滞しているRQを削除します
[-a]	すべてのRQの状態をリスト形式で出力します

## 参考

[-f]オプションと[-c]オプションを使用してRQを処理するには、tmadmin -mオプションを使ってマスター・モードで動作するようにします。

### ● 例

#### – RQの状態照会(-a)

各レコードは以下の項目を表現します。

```
$$1 tmaxi9 (tmadm): rqs -a

          rq1 RDY  0 0 0 0 0 5 5
total count : 1
```

出力されるデータは、以下の形式で表示されます。

```
rq名 状態 request reply fail requestenqueued requestdequeued replyenqueued
replydequeued
```

最後のレコードはtotal countを表示します。

#### – RQの照会(-l)

以下は、使用できるRQを照会する例です。

```
$$1 tmax1 (tmadm): rqs -l
list of available RQs:
rq1 rq2
```

#### – RQ状態の照会(-s)

以下は、RQの状態を照会する例です。

```
$$1 tmax1 (tmadm): rqs -s rq1
- - - - -
```

```

Number of queue entries
- - - - -
request                      0
reply                        0
fail                         0
- - - - -
Accumulated queue activities
- - - - -
request enqueued             0
request dequeued             0
reply enqueued               10
reply dequeued               0
- - - - -

```

#### – RQサービス処理(-f)

以下は、RQに蓄積されているサービス进行处理する例です。

```

$$1 tmax1 (tmadm): rqs -f rq2
RQ(rq2) flushed

```

#### – RQの削除(-c)

以下は、停滞しているRQを削除する例です。

```

$$1 tmax1 (tmadm): rqs -c rq2
RQ(rq2) statics cleared

```

## 5.5. 運用管理

### 5.5.1. suspend(sp)

アプリケーション・サーバー・プログラムのエラーなどで業務処理が不可能になった場合、解決策として動作しているサーバー・プロセスを停止させる際に使用する機能です。コマンドが実行されると、停止されたサーバー・プロセスが現在処理中のサービスを正常に完了した後、それ以上の動作を中止し、キューにたまっているサービスは待機状態になります。この際、継続的に要求されるサービスはすべてキューにたまります。suspendコマンドの省略形は**sp**です。

#### ● 使用方法

```

$$1 tmax1 (tmadm): Usage: suspend {-s svc_name [-p pid | -i spri] [-a]
                                [-n T|P] [-N T|P] | -v svr_name [-k num] [-a]
                                [-q] [-n T|P] [-N T|P]}

```

項目	説明
-s <i>svc_name</i>	サービス名です
[ -p <i>pid</i> ]	サービスをサーバー・プロセスごとに一時停止(suspend)するオプションで、[-s]オプションと一緒に使われます
[ -i <i>spri</i> ]	サービスをspriごとに一時停止するオプションであり、[-s]オプションと一緒に使われます
[ -n <i>T/P</i> ]	正常終了後に再起動された際にstateを決定します – T : RDYで起動されます – P : 異常終了直前の状態で起動されます
[ -N <i>T/P</i> ]	異常終了後に再起動された際にstateを決定します – T : RDYで起動されます – P : 異常終了直前の状態で起動されます
-v <i>svr_name</i>	サーバー・プロセスのスケジューリングを防ぐオプションです。要求を受けたサービスはキューに待機し、再会(resume)されるまで待機するか、MAXQCOUNT、CLHQTIMEOUTの設定によってキューから除去することができます
[ -k <i>num</i> ]	対象サーバーのうち一部のプロセスを一時停止するオプションです。  [-v]オプションと一緒に使われます。オプションで指定した数の分だけ一時停止させます。すでに一部サーバー・プロセスが一時停止されている場合にはオプションの設定値の分だけ追加で一時停止されます。オプションの設定値は当該サーバーのMAXを超えることができません。  サーバー・プロセスごとに一時停止状態を確認するには、[stat -p -X]コマンドを実行します。  <pre>suspend(sp) -v svr_name [-k 5]</pre> [-k]オプションを使用してサーバー・プロセスがMAX個すべてが一時停止されるとサーバーの状態はBLKに変わります。[-k]オプション無しにsp -v svr_nameを実行したのと同じです
[ -q ]	asqcountによってサーバーを追加で起動します。[-v]オプションと一緒に使われます
[ -a ]	一時停止の要求時に対象のサーバーがサービスを実行中であれば、サービスが終了されるまで待機することになります。このオプションを使用すれば対象サーバーが実行中でも待機せずに即刻一時停止状態に変更します。[-v]オプションと一緒に使われます

## 5.5.2. resume(rs)

動作が停止されたサーバー・プロセスの活動を再開させます。活動が再開されたサーバー・プロセスはキューに待機していたサービスの処理を始め、要求されるサービスに対して処理可能な状態になります。resume コマンドの省略形はrsです。

- 使用方法

```
$$1 tmax1 (tmadm): resume {-s svc_name [-p pid | -i spri] |  
                           -v svr_name [-k count ] }
```

項目	説明
-s <i>svc_name</i>	サービスのスケジューリングを再開(resume)するオプションです
[-p <i>pid</i> ]	<p>サービスをサーバー・プロセスごとに再開します。サービスをサーバー・プロセスごとに一時停止／再開しようとするとき、一時停止(suspend)を実行すると、関連サービスが実行中の場合は実行が完了してから一時停止されるため、時間が遅延されることがあります。この場合、tmadminを強制に終了させると当該サービスが再開されます。</p> <p>一時停止(suspend)以後の当該サービスへの要求はキューにたまります。[-s]オプションと一緒に使用されます</p> <pre>resume(rs) -s svc_name [-p pid]</pre>
[-i <i>spri</i> ]	<p>サービスをspriごとに再開します。サービスをspriごとに一時停止／再開しようとするとき、一時停止を実行すると、関連サービスが実行中の場合は実行が完了してから一時停止されるため、時間が遅延されることがあります。この場合、tmadminを強制終了させると、当該サービスが再開されます。</p> <p>一時停止以後の当該サービスへの要求はキューにたまります。[-s]オプションと一緒に使用されます</p> <pre>resume(rs) -s svc_name [-i spri]</pre>
-v <i>svr_name</i>	サーバー・プロセスのスケジューリングを再開するオプションです
[-k <i>count</i> ]	<p>対象サーバーのうち一部プロセスを再開するオプションです。</p> <p>[-v]オプションと一緒に使われます。オプションで指定した数の分だけ一時停止されたサーバー・プロセスを再開させます。すでにすべてのサーバー・プロセスが一時停止状態でなければ、コマンドは取り消されます。オプションの設定値は当該サーバーのMAXを超えることができません。</p> <p>サーバー・プロセスごとに一時停止状態を確認するには、stat -p -Xコマンドを実行します。</p> <pre>resume(rs) -v svr_name [-k 5]</pre>

項目	説明
	[-k]オプションを使用してサーバー・プロセスが1つでも再開されると、サーバーの状態はRDYに変わります。もし起動しているプロセスがなければNRDY状態になります

以下は、resumeに関する説明です。

- 1ノードのCOUSINグループに属したサーバーに対するsuspend/resume

1ノードに2つ以上のサーバー・グループがCOUSINIに設定されている場合、COUSINサーバー・グループに属するすべてのサーバーに対して一時停止／再開が適用されます。

#### 注

ASQCOUNTはSVRGROUPセクションで設定します。サーバー・プロセスをBLKに設定するには[-v]オプションを使用する必要があります。

## 5.5.3. advertise/unadvertise

tmadminのadvertise/unadvertiseコマンドを使って特定のサービス名を登録または登録解除できます。

特定サービスを登録(advertise)する場合、CLHが管理するサーバー別サービス名称テーブルに当該サービスの名前を登録することができ、登録解除(unadvertise)する場合は、サーバー別サービス名称テーブルから削除されます。

登録(advertise)はサーバー・プロセスのサーバーが提供する新しいサービスを登録し、登録解除(unadvertise)は当該サーバーが提供するサービスを登録解除します。登録解除されたサービスを呼び出す場合、TPENOENTエラーを受信します。特定サービスが1つのサーバー・プロセスで登録解除されたとしても、他のサーバー・プロセスで提供されていれば、依然としてサービスルーチンを実行できるようになります。

- 使用方法

- 登録(advertise)

```
$$1 tmax1 (tmadm): advertise {-s svc_name [-p pid]}
```

項目	説明
-s <i>svc_name</i>	サービス名を登録します
[-p <i>pid</i> ]	サーバー・プロセスのPIDを登録します

- 登録解除(unadvertise)

```
$$1 tmax1 (tmadm): unadvertise {-s svc_name [-p pid]}
```

項目	説明
-s svc_name	サービス名を登録解除します
[-p pid]	サーバー・プロセスのPIDを登録解除します

- 例

- 特定サービスの解除(unadvertise -s)

以下は、状態情報を確認したサービスのうち名前がTOUPPERで、サーバー・プロセスIDが15287のサービスをunadvertiseにより登録解除する例題です。サービスを登録/解除(advertise / unadvertise)する前にstコマンドを使ってサービスの状態情報を確認します。

```
– $$1 tmax1 (tmadm): st -p -x
```

```
CLH 0:
```

```
-----
svr_name      svpname      spr_no      status      count      avg      svc
              PID          fail_cnt    err_cnt     min_time   max_time
              utime      umin_time   umax_time   stime      smin_time  smax_time
-----
svr2          svg1          36          RDY          0          0.000    -1
              15285          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
svr2          svg1          37          RDY          0          0.000    -1
              15286          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
svr2          svg1          38          RDY          0          0.000    -1
              15287          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
-----
```

```
TOTAL COUNT = 0
TOTAL SVCFAIL COUNT = 0
TOTAL ERROR COUNT = 0
TOTAL RUNNING COUNT = 0
```

```
$$1 tmax1 (tmadm): unadvertise -s TOUPPER -p 15287
```

```
TOUPPER is unadvertise
```

```
$$8 tmaxh4 (tmadm): st -s
```

```
CLH 0:
```

```
-----
svc_name      svr_name      count      cq_cnt      aq_cnt      q_avg      avg      status
-----
TOUPPER      svr2          0          0           0           0.000      0.000    PUNADV
-----
```

- 特定サービスの登録 (advertise -s)



以下は登録解除したサービスをadvertiseにより登録する例題です。サービスを登録/解除(advertise / unadvertise)する前にstコマンドを使ってサービスの状態情報を確認します。

```
$$1 tmax1 (tmadm): st -p -x

CLH 0:
-----
svr_name      svgrname      spr_no      status      count      avg      svc
              PID      fail_cnt    err_cnt    min_time   max_time
              utime   umin_time   umax_time   stime      smin_time   smax_time
-----
svr2          svgr1          36          RDY          0          0.000     -1
              15285          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
svr2          svgr1          37          RDY          0          0.000     -1
              15286          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
svr2          svgr1          38          RDY          0          0.000     -1
              15287          0           0           0.000      0.000
              0.000      0.000      0.000      0.000      0.000      0.000
-----

TOTAL COUNT = 0
TOTAL SVCFAIL COUNT = 0
TOTAL ERROR COUNT = 0
TOTAL RUNNING COUNT = 0

$$1 tmax1 (tmadm): advertise -s TOUPPER -p 15287
TOUPPER is advertise

$$11 tmaxh4 (tmadm): st -s

CLH 0:
-----
svc_name      svr_name      count      cq_cnt      aq_cnt      q_avg      avg      status
-----
TOUPPER       svr2          0          0           0           0.000      0.000     RDY
```

5.5.4. restat

特定のサーバー・プロセスまたはすべてのサーバー・プロセスの統計情報を初期化するコマンドです。マスター・モードで使用する時のみ実行できます。

● 使用方法

```
$$1 tmax1 (tmadm): restat [ -v [server_process_name] ] [ -a ]
```

項目	説明
[ -v [server_process_name] ]	指定されたサーバー・プロセスの統計情報を初期化します

項目	説明
[-a]	すべてのサーバー・プロセスの統計情報を初期化します

### 5.5.5. rebootsvr(rbs)

rbs(Reboot Server Process)は、現在使用中のサーバー・プロセスを新しいプロセスに変更します。

TMAX\_BKAPPDIRを「.profile」の環境変数に指定し、新しいプログラムの実行ファイルとして指定します。たとえば、\$TMAXDIR/bk\_appbinに移動させ、以下のコマンドを実行します。

- 使用方法

```
$ $1 tmax1 (tmadm): rbs new_file old_file
```

項目	説明
new_file	新しいファイル名を指定します
old_file	使用中のファイル名を指定します

### rbsの実行手順

以下は、rebootsvr(rbs)の実行手順です。

1. .profileに以下のように環境変数を追加します。

```
export TMAX_BKAPPDIR=/data2/starbj81/tmax64/bk_appbin
```

2. 環境変数に設定した位置と同じ位置にbk\_appbinディレクトリーを作成します。

```
$mkdir bk_appbin
```

3. appbinディレクトリーに存在する実行ファイルをbk\_appbinディレクトリーに移動させます。

```
$cp appbin/svr2 bk_appbin/
```

4. tmadmin -mを実行した後、rbsを実行します。

- サーバーがMIN=1 && MAX=1であるか起動されていないと、サーバーの切り替え中に一時停止または再開が行われます。

以下は、現在MIN=1、MAX=1のsvr2サーバー・プロセスが起動される例です。

```
$ $1 tmax1 (tmadm): rbs svr2 svr2
>> suspend ok
>> down ok
```

```
>> cp ok
>> boot ok
>> resume ok
>> reboot svr /data2/starbj81/tmax64/appbin/svr2 finished
```

- それ以外の場合、サーバー・プロセスを順次終了および起動しながら切り替えを行います。

以下は、現在MIN=3のsvr2サーバー・プロセスが起動される例です。

```
$$1 tmax1 (tmadm): rbs svr2 svr2
TMBOOT for node(tmaxh2) is starting:
    TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
    Fri Dec 19 11:21:06 2003
TMBOOT for node(tmaxh2) is starting:
    TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
    Fri Dec 19 11:21:06 2003
TMBOOT for node(tmaxh2) is starting:
    TMBOOT: SVR(svrname: svr2, execname: _rbs00_svr2) is starting:
    Fri Dec 19 11:21:06 2003

>> 3 servers booted using tmp execfile _rbs01_svr2
>> reboot svr /data2/starbj81/tmax64/appbin/svr2 finished
```

5. psを使用してサーバー・プロセスが正常に起動されたことを確認します。

- 現在3つのsvr2サーバー・プロセスが起動された場合

```
$ps -ef | grep svr2
starbj81 21710      1  0 11:46:32 pts/ts      0:00 _rbs00_svr2 -b -21709
               -S svr2 -s svr2 -d -1 -v 21689
starbj81 21713      1  0 11:46:32 pts/ts      0:00 _rbs00_svr2 -b -21712
               -S svr2 -s svr2 -d -1 -v 21689
starbj81 21716      1  0 11:46:32 pts/ts      0:00 _rbs00_svr2 -b -21715
               -S svr2 -s svr2 -d -1 -v 21689
```

- 現在1つのsvr2サーバー・プロセスが起動された場合

```
$ps -ef | grep svr2
starbj81 21607      1  0 11:43:46 pts/ts      0:00 svr2 -s svr2 -g 2
```

当該名のサーバー・プロセスが2つ以上であれば、当該プロセスをすべて終了させた後、新規プロセスを起動させて処理すると業務の空白状態が発生してしまいます。したがって、こうした場合には、サーバー・プロセスを1つずつ終了させながら起動します。たとえば、svr1サーバーのMIN=2の状況でrbsを実行させた場合、以下の順でrbsが実行されます。

1. svr1 1を終了します。
2. bk\_appbinディレクトリーに一時ファイル(\_rbs00\_svr1)をコピーします。
3. \_rbs00\_svr1が起動します。
4. svr1 2を終了します。
5. bk\_appbinディレクトリーに一時ファイル(\_rbs00\_svr1)をコピーします。
6. \_rbs00\_svr1が起動します。

## 同時性を保証するサーバーの切り替え

同時性の保証は、互いに異なるバージョンのプロセスが同時にサービスを実行しないように制限することを意味します。rbsを実行するとき、同時性の保証および、非変更サービスに対する無停止サーバー・プロセスの切り替えができます。

### ● 使用方法

```
$$1 tmax1 (tmaxadm): rbs [-S | -s svc_name,...] newfile svr_name [svg_name]
```

項目	説明
[-S]	当該サーバーの全サービスを一時停止(suspend)します
[-s svc_name,...]	当該サーバーの1つのサービスを一時停止します
newfile	新しいファイル名を指定します
svr_name	一時停止の対象となるサーバー名です
[svg_name]	一時停止の対象となるサーバーが属しているサーバー・グループです

### ● 例

#### ー サーバーの特定サービスの変更(-s)

以下は、サーバーの特定サービスのみ変更する場合です。svr2サーバーのTOUPPERサービスのみ一時的に停止され、他のサービスは継続してサービスを実行できます。

```
$$1 tmax1 (tmaxadm): rbs -s TOUPPER svr2_new svr2
```

以下は、サーバーの特定サービス2つを変更する場合です。svr2サーバーのTOUPPERおよびTOWERサービスのみ一時的に停止され、他のサービスは継続してサービスを実行できます。svc\_nameを複数指定する場合、コンマ(,)で区切り、空白が含まれてはなりません。

```
$$1 tmax1 (tmaxadm): rbs -s TOUPPER,TOWER svr2_new svr2
```

#### – サーバーの全サービスの変更(-S)

以下は、サーバーの全サービスを変更する場合です。svr2サーバーの全サービスが一時的に停止されることがあります。

```
$$1 tmax1 (tmadm): rbs -S svr2_new svr2
```

以下は、特定のサーバー・グループに属しているサーバーの全サービスを変更する場合です。svg1に属しているsvr2サーバーの全サービスが一時的に停止されることがあります。

```
$$1 tmax1 (tmadm): rbs -S svr2_new svr2 svg1
```

---

#### 注

1. rbsにオプション(-sまたは-S)を指定しない場合、同時性が保証されません。
  2. 1ノードにCOUSINグループとして設定された環境では、rbsでsvgname別のrbsが制限されます。
- 

## マルチノードの同時性を保証するサーバーの切り替え

マルチノードの同時性の保証は、マルチノードで互いに異なるバージョンのプロセスが同時にサービスを実行しないように制限することを意味します。負荷分散が設定されたマルチノードでrbsを実行するとき、同時性の保証および、非変更サービスに対する無停止サーバー・プロセスの切り替えができます。

#### ● 使用方法

```
$$1 tmax1 (tmadm): mrbs [-S | -s svc_name,...] newfile svr_name
```

項目	説明
[ -S ]	サーバーの全サービスを一時停止します
[ -s svc_name ]	サーバーの1つのサービスを一時停止します
newfile	新しいファイル名を指定します
svr_name	一時停止の対象となるサーバー名です

#### ● 例

#### – サーバーの特定サービスの変更(-s)

以下は、当該サーバーの特定サービスのみ変更する場合です。svr2サーバーのTOUPPERサービスのみ一時的に停止され、他のサービスは継続してサービスを実行できます。

```
$$1 tmax1 (tmadm): mrbs -s TOUPPER svr2_new svr2
```

以下は、サーバーの特定サービス2つを変更する場合です。svr2サーバーのTOUPPERおよびTOLOWERサービスのみ一時的に停止され、他のサービスは継続してサービスを実行できます。svc\_nameを複数指定する場合、カンマ(,)で区切り、空白が含まれてはなりません。

```
$$1 tmax1 (tmadm): mrbs -s TOUPPER,TOLOWER svr2_new svr2
```

#### – サーバーの全サービスの変更(-S)

以下は、サーバーの全サービスを変更する場合です。svr2サーバーの全サービスが一時的に停止されることがあります。

```
$$1 tmax1 (tmadm): mrbs -S svr2_new svr2
```

## 1つのノードのCOUSINグループに属するサーバーのrbs

1つのノードに2つ以上のサーバー・グループがCOUSINで設定された場合、COUSINサーバー・グループに属するすべてのサーバーをに対してrbs(サーバーの切り替え)ができます。

### 5.5.6. cfgadd(ca)

cfgadd(ca)は、Tmaxの運用中に特定のサーバー・プログラムにサービスを追加するとき、該当するサーバーのみを停止させ、動的に追加することができます。また、サービス以外にも、運用中にサーバー、サーバー・グループ、ノードを動的に追加することができます。Tmax v5.0バージョンからは、cflの[-a]オプションを使用しないと、作成したバイナリ環境ファイルを動的に追加することができないため、[-a]オプションを使って環境ファイルをコンパイルする必要があります。

cfgadd(ca)を使用してサービスを動的に追加するときの制約事項は、[5.5.6節「cfgadd\(ca\)を使用してサービスを動的に追加するときの制限事項」](#)を参照してください。

#### ● 使用方法

```
$$1 tmax1 (tmadm) : cfgadd (ca) - i cfgfile
```

項目	説明
- i <i>cfgfile</i>	追加するサービスの環境ファイル名を指定します

## サービスの動的追加

以下は、サービスの動的追加に該当するcfgadd(ca)の実行手順です。Tmax v6.0から、追加する環境ファイルを作成せず既存の環境ファイルを修正する方式に変更されました。

1. 環境ファイルを修正します。

```
$vi modify.m
```

- 修正した環境ファイルを-aでコンパイルし、-oオプションを使って別の名前のバイナリ環境ファイルを作成します。

```
$cfl -a modify.m -o tmchg
```

- gstでサービス・テーブルを作成します。

```
$gst -f tmchg
```

- 「tmadmin -m」を実行し、cfgadd(ca)を使ってサービスを追加します。

```
$cfgadd -i tmchg
```

- 該当するサーバー・プログラムをコンパイルします。

```
$compile c svr1
```

- サーバー・プロセスを起動します。

```
$tmboot -S svr1
```

## サーバーの動的追加

以下は、サーバーの動的追加に該当するcfgadd(ca)の実行手順です。

- 新規サーバーを追加して環境ファイルを修正(追加)します。

<modify.m>

```
*SERVER
既存の項目
.....
hello          SVGNAME = svg1
*SERVICE
既存の項目
.....
HELLO          SVRNAME = hello
```

- 環境ファイルをコンパイルします。modify.mは、既存の環境ファイルで追加された項目を入力して修正したファイルです。

```
cfl -a modify.m -o tmchg
```

### 3. サービス・テーブルを作成します。

```
gst -f tmchg
```

### 4. cfgadd -iで環境ファイルを追加します。

```
tmaxil@dhjang ./config > tmdadmin -m
$$3 tmaxil (tmadm): si
-----
clh  svrname (svri) status  count  qcount  qpcount  emcount
-----
0    tmaxgw  ( 0)   RDY    0       0       0       0
0    toupper (18)   RDY    0       0       0       0

$$4 tmaxil (tmadm): cfgadd -i tmchg
config is successfully added
$$5 tmaxil (tmadm): si
-----
clh  svrname (svri) status  count  qcount  qpcount  emcount
-----
0    tmaxgw  ( 0)   RDY    0       0       0       0
0    toupper (18)   RDY    0       0       0       0
0    hello   (19)  NRDY    0       0       0       0
```

helloサーバーはまだNRDY状態です。

### 5. tmboot -S hello -f tmchgを入力します。

```
tmaxil@dhjang ./config > tmboot -S hello -f tmchg
TMBOOT for node(tmaxil) is starting:
Welcome to Tmax demo system: it will expire 2002/8/31
Today: 2002/8/19
      TMBOOT: SVR(hello) is starting: Mon Aug 19 15:37:44 2002
```

### 6. tmdadminでhelloサーバーがRDY状態か確認します。

```
$$6 tmaxil (tmadm): si
-----
clh  svrname (svri) status  count  qcount  qpcount  emcount
-----
0    tmaxgw  ( 0)   RDY    0       0       0       0
0    toupper (18)   RDY    0       0       0       0
0    hello   (19)   RDY    0       0       0       0
```



## サーバー・グループの動的追加

サーバー・グループを動的に追加する方法は、新しい追加環境ファイルだけ少し異なり、その他は同じです。新しく登録されるサーバー・グループ、サーバー、およびサービスを追加環境ファイルで作成します。その他は、サーバーの動的追加方法と同じです。

<tmchg.m>

```
*SVRGROUP
svg2          NODENAME = "aix51"

*SERVER
svr3          SVGNAME = svg2, MIN = 1, MAX = 10

*SERVICE
FDLToupper   SVRNAME = svr3
FDLTolower   SVRNAME = svr3
```

## ノードの動的追加

cfgaddを利用してノードを動的に追加できます。COUSINサーバー・グループが属するノードも動的に追加できます。

以下は、ノードの動的追加に該当するcfgadd(ca)の実行手順です。

1. 環境ファイルを修正します。

1. <tmconfig\_add.m>

```
*DOMAIN
tmax1          SHMKEY = @SHMEMKY@, MINCLH = 1, MAXCLH = 3,
               TPORTNO = @TPORTNO@, BLOCKTIME = 300, MAXCPC = 100,
               RACPORT = @TRACPORT@

*NODE
@HOSTNAME@     TMAXDIR = "@TMAXDIR@",
               APPDIR  = "@TMAXDIR@/appbin",
@RMTNAME@      TMAXDIR = "@RMTDIR@",
               APPDIR  = "@RMTDIR@/appbin",

*SVRGROUP
#svg1          NODENAME = "@HOSTNAME@", COUSIN = "svg2", LOAD = 2 <- 既存の内容
svg1          NODENAME = "@HOSTNAME@", COUSIN = "svg2,svg3", LOAD = 2 # <- 修正
した内容
svg2          NODENAME = "@RMTNAME@", LOAD = 1
```

```

*SERVER
svr2          SVGNAME = svg1

*SERVICE
TOUPPER       SVRNAME = svr2
TOLOWER       SVRNAME = svr2

#追加した内容

*NODE
@RMTNAME2@    TMAXDIR = "@RMTDIR2@",
              APPDIR  = "@RMTDIR2%/appbin",

*SVRGROUP
svg3          NODENAME = "@RMTNAME2%",LOAD = 1

```

2. 追加された新しいノードにracdを起動します。

```
Node3>$ racd -k
```

3. 環境ファイルをコンパイルします。新しいノードが追加された環境ファイルのtmconfig\_add.mをコンパイルします。コンパイルするとき、[-o]オプションを使って別の名前のバイナリ環境ファイルを作成します。

```

node1>$cfl -a tmconfig_add.m -o tmchg
CFL is done successfully for node(node1)

CFL: rcfl start for rnode (node2)
CFL is done successfully for node(node2)

CFL: rcfl start for rnode (node3)
CFL is done successfully for node(node3)

```

4. サーバーをコンパイルします。新規追加するノードのサーバーをコンパイルします。

```

node3>$ gst -f tmchg
node3>$ compile c svr2

```

5. 動的ノードを追加します。tmadminのcfgaddコマンドを使って動的にノードを追加します。それぞれのノードでtmadmin -lで当該コマンドを実行する必要があることに注意します。

以下は、node1、node2が存在する状況でnode3を追加する場合の例です。

```

# node1
$ node1>tmadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$2 node1 (tmadm): cfgadd -i tmchg

```

```

(I) TMM0211 General Infomation : CFGADD started [TMM0902]
(I) TMM0211 General Infomation : CFGADD completed [TMM0907]
config is successfully added

# node2
$ node2>tmadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$2 node2 (tmadm): cfgadd -i tmchg
(I) TMM0211 General Infomation : CFGADD started [TMM0902]
(I) TMM0211 General Infomation : CFGADD completed [TMM0907]
config is successfully added

```

## 6. 新規追加されたノード(node3)を起動します。

```

Node3>tmboot -n tmaxh4 -f tmchg
TMBOOT for node(tmaxh4) is starting:
Welcome to Tmax demo system: it will expire 2008/11/23
Today: 2008/9/24
    TMBOOT: TMM is starting: Wed Sep 24 11:11:59 2008
    TMBOOT: CLL is starting: Wed Sep 24 11:11:59 2008
(I) TMM0211 General Infomation : node register (nodeno = 0(0)) success [TMM0404]
(I) TMM0211 General Infomation : node register (nodeno = 1(1)) success [TMM0404]

    TMBOOT: CLH is starting: Wed Sep 24 11:11:59 2008
(I) CLH9991 Current Tmax Configuration: Number of client handler(MINCLH) = 1
    Supported maximum user per node = 7966
    Supported maximum user per handler = 7966 [CLH0125]
    TMBOOT: TLM(tlm) is starting: Wed Sep 24 11:11:59 2008
    TMBOOT: SVR(svr2) is starting: Wed Sep 24 11:11:59 2008

```

## 7. 新規追加されたノードを確認します。ノードが正常に追加されたことを確認します。

```

node1>tmadmin
TMADMIN for rnode (node2): starting to connect to RAC
TMADMIN for rnode (node3): starting to connect to RAC
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 node1 (tmadm): ti

Tmax System Info: DEMO version 4.0 SP #3 Fix #8:

    expiration date = 2008/11/22
    maxuser = UNLIMITED,
    domaincount = 1,
    nodecount = 3,
    svgrpcount = 3,

```

```

svrcount = 9, svccount = 6
rout_groupcount = 0, rout_elemcount = 0
cousin_groupcount = 1, cousin_elemcount = 3
backup_groupcount = 0, backup_elemcount = 0

Tmax All Node Info: nodecount = 3:
-----
no      name      portno  racport  shmkey  shmsize  minclh  maxclh
-----
0      node1      8350    3155     88350   225760    1       3
1      node2      8350    3155     88350   225760    1       3
2      node3      8350    3155     88350   225760    1       3

$$2  (tmadm): st -s

CLH 0:
-----
svc_name  svr_name  count  cq_cnt  aq_cnt  q_avg  avg  status
-----
TOLOWER   svr2       0       0       0     0.000  0.000  RDY
TOUPPER   svr2       0       0       0     0.000  0.000  RDY

Msg from rnode(node2):

CLH 0:
-----
svc_name  svr_name  count  cq_cnt  aq_cnt  q_avg  avg  status
-----
TOLOWER   svr2       0       0       0     0.000  0.000  RDY
TOUPPER   svr2       0       0       0     0.000  0.000  RDY

Msg from rnode(node3):

CLH 0:
-----
svc_name  svr_name  count  cq_cnt  aq_cnt  q_avg  avg  status
-----
TOLOWER   svr2       0       0       0     0.000  0.000  RDY
TOUPPER   svr2       0       0       0     0.000  0.000  RDY

```

## cfgadd(ca)を使用してサービスを動的に追加するときの制限事項

cfgadd(ca)を使用してサービス、サーバー、サーバー・グループ、ノードなどを動的に追加する際、以下のよう  
な制限事項があります。

- 各項目に対して追加のみ可能であり削除はできません。削除するときは、既存の環境ファイルから項目を削除してcflを実行した後、Tmaxを再起動します。
- 既存の環境設定ファイルに存在する項目の名前が、新しく追加される環境設定ファイルに重複して存在してはなりません。
- 新規追加された環境設定ファイルの内容を既存の環境設定ファイルに適用する場合、各セクションの最後に追加する必要があります。追加項目を既存の環境設定の中間に挿入したり、順序を変更したりすると、運用環境の情報が異常変更される可能性があるため注意が必要です。
- MAXNODE、MAXSVG、MAXSVR、MAXSPR、MAXSVC、MAXTMS、MAXCPCなどにだけ項目を追加することができます。また、これらの項目の値を変更してはなりません。
- サーバー・グループを追加する際、XAサーバー・グループの場合はTMSを指定し、該当するプログラムが作成されたことを確認します。
- RQは動的に追加できません。
- COUSINとBACKUPが存在するサーバー・グループは追加および削除できません。
- COUSINとBACKUPが存在するゲートウェイは追加および削除できません。
- COUSINとBACKUPが存在するサーバー・グループに動的に追加されたサービスは削除できません。
- サービス、サーバー・サーバー・グループ、ノードなどを追加する際、cflを使用して動的に追加するためのバイナリ環境ファイルを作成します。

cflの[-a]オプションを使用せずに作成されたバイナリ環境ファイルを動的に追加すると、以下のようなエラーメッセージが出力されます。

```
(E) ADM2048 Engine type mismatch (0): 'a' option must be used [ADM0417]
```

## 5.5.7. set

現在設定されている環境ファイルの設定値を動的に変更できるコマンドです。

変更可能な項目については、tmaxでcfgコマンドにより確認できます。各項目のうち括弧の中に省略形が表示された項目が動的に変更できる項目です。

- 使用方法

```
$$$1 tmax1(tmadm): set [-d [domain]| -g [server_group]| -v [service]|  
-s [server]] 項目 値
```

項目	説明
[ -d [domain] ]	DOMAINセクションを変更する場合に設定します。特定のドメイン情報を変更する場合、ドメイン名を設定します
[ -g [server_group] ]	GATEWAYセクションを変更する場合に設定します。特定のサーバー・グループ情報を変更する場合、サーバー・グループ名を設定します
[ -v [service] ]	SERVICEセクションを変更する場合に設定します。特定のサービス情報を変更する場合、サービス名を設定します
[ -s [server] ]	SERVERセクションを変更する場合に設定します。特定のサーバー情報を変更する場合、サーバー名を設定します
項目	変更する項目を設定します
値	変更する値を設定します

- 例

以下は、コマンドの使用例です。

```

$$1 tmax1 (tmadm): set -d res1 bt 100
$$1 tmax1 (tmadm): set -g svg1 ld 5
$$1 tmax1 (tmadm): set -v kfd11 mq 1000
$$1 tmax1 (tmadm): set -s SYNC pr 100
$$1 tmax1 (tmadm): set -n tmaxh4 cb n
$$1 tmax1 (tmadm): set -s svc_name pr 99
$$1 tmax1 (tmadm): set -v svr2 maxsvr 10 (max(svr2)の場合、既存の値より小さい値だけ変更可能)
$$1 tmax1 (tmadm): set -d dom1 portno 9999
$$1 tmax1 (tmadm): set -g svg2 maxrstart 10
$$1 tmax1 (tmadm): set -v svr140_A restart N
$$1 tmax1 (tmadm): set -v gw1 schedule FA
$$1 tmax1 (tmadm): set -d dom nclhchctime 10

```

## 5.5.8. setopt

TMMOPTなどの環境設定に定義されているオプションのうち、動的変更が可能な環境設定値を動的に変更できるコマンドです。変更可能な項目は、tmadminでcfgoptコマンドを実行して確認できます。cfgoptで出力される項目が動的に変更可能な項目です。

- 使用方法

```

$$1 tmax1 (tmadm): setopt [-tmm][-gw] 項目 値

```

項目	説明
[-tmm]	TMMOPT環境設定に定義されているオプションの値を変更する場合に設定します
[-gw]	GATEWAYセクションのCLOPT環境設定に定義されているオプションの値を変更する場合に設定します
項目	変更する項目を設定します。cfgoptで出力される項目のオプションを設定します(例: -F)
値	変更する値を設定します

- 例

以下は、コマンドの使用例です。

```

$ $1 tmax1 (tmadm): setopt -tmm -F 30
new value (30) is set for section = -tmm, name = N/A, fld = -F
$ $1 tmax1 (tmadm): setopt -tmm -t 15
new value (15) is set for section = -tmm, name = N/A, fld = -t

```

## 5.5.9. qpurge(qp)

業務トラフィックの殺到でたくさんの業務が溜まり、トランザクションを正常に処理できない場合、現在キューに蓄積されているサービス要求を削除する機能です。1日に数十万件の業務を処理する銀行や官公庁で有用な機能です。

削除された業務はクライアントの再要求により再処理できます。Tmaxではサーバー・プロセス別にキューを管理し、管理者は特定のサーバー別にキューを削除できるため、他業務の効果的な処理にも役に立ちます。

Queue Purgeコマンドの省略形は**qp**です。削除されたサービスはクライアントにTPEQPURGE(tperrno = 27)を返すので、クライアントはそれに合わせて適切に対処できます。qpの内容と削除されたクライアントIDはslogと一緒に記録されます。

- 使用方法

```

$ $1 tmax1 (tmadm): qpurge {-v svr_name [ -k number ] | -s svc_name}

```

項目	説明
-v svr_name	キューに蓄積されている特定サーバーの要求を削除します
[-k number]	サーバーのキューに蓄積されている要求のうち、一部を消去(Purge)する機能です。必ず[-v]オプションと一緒に使用します。  number値が0以上の数のNであれば、最初に入ったN個のサービス要求のみ残し、以降の要求を消去します

項目	説明
	<ul style="list-style-type: none"> <li>- 1: 現在、キューで待機しているサービス要求のうち、MAXQCOUNTの数の分だけ残し、以降の要求は消去します</li> <li>- 0: 現在、キューで待機しているすべてのサービス要求を消去します</li> </ul>
-s <i>svc_name</i>	キューに蓄積されている特定サービスの要求を削除します

## 5.5.10. **discon(ds)**

現在接続されているものの、何の作業も行っていないクライアントの接続を強制に解除します。クライアント情報を取得できる**ci**コマンドで確認後、使用します。disconコマンドの省略形は**ds**であり、以下のようなオプションが提供されます。

- 使用方法

```
$ $1 tmax1 (tmadm): ds [-h clhno] [-f] {-a | -n | -i idle_time | -c clid | -A}
```

項目	説明
[ -h clhno ]	CLHIに接続しているクライアントの接続を解除します
[ -f ]	クライアントとの接続を即刻解除するオプションです
{ -a }	CLHIに接続しているすべてのクライアントの接続を解除します。  [ -h ]オプションが適用されていない場合、デフォルトで0番CLHIに接続しているクライアントの接続が解除されます
{ -n }	正確な情報を持っていないクライアントとの接続を解除します。  「tpalloc」で割り当てられたバッファは使用しません
{ -i idle time }	指定時間(秒単位)を超えたセッションのクライアントとの接続を解除します
{ -c clid }	クライアントに付与したID番号で接続を解除します。ID番号は必ず設定します
{ -A }	すべてのCLHIに接続しているクライアントとの接続を解除します

## 5.5.11. **logstart/logend**

管理者はtmadminを使用して様々な情報をリアルタイムで照会できます。また与えられた状況に合わせて即刻的且つ効果的な措置を取れます。tmadminは統計情報の分析のために管理者データをログで残せます。ロギング・ファイルは現在のディレクトリーに作成されます。ログデータにより、業務トラフィックの殺到時間、不要なサーバー・プロセス、キューイング状態などの情報を確認して全般的なシステム分析が可能です。

logstartコマンドを使ってログ処理を開始し、logendコマンドで終了します。



- 使用方法

- ログ処理の開始(logstart)

```
$$1 tmax1 (tmadm): logstart filename
```

項目	説明
filename	ログファイル名を設定します

- ログ処理の終了(logend)

```
$$1 tmax1 (tmadm): logend
```

## 5.5.12. chtrc

TMAX\_TRACEは、Tmaxアプリケーションの実行に対するランタイム・トレースを行う機能(Runtime tracing facility)です。この機能は、環境変数のTMAX\_TRACEを設定して使用でき、システムの運用中に動的に設定を変更する場合にはtmadminの**chtrc**コマンドを使用できます。

- 使用方法

```
chtrc [-g svgname | -v svrname | -g svgname -v svrname |  
      -i spri | -g svgname -i spri | -e gqs] -s spec
```

項目	説明
[ -g svgname ]	サーバー・グループのスペックを変更する場合に設定します
[ -v server ]	サーバーのスペックを変更する場合に設定します
[ -g svgname -v server ]	サーバー・グループの当該サーバー・スペックを変更する場合に設定します
[ -i spri ]	サーバー・プロセスのスペックを変更する場合に設定します
[ -g svgname -i spri ]	サーバー・グループの当該サーバー・プロセスのスペックを変更する場合に設定します
[ -e gqs ]	gqsのスペックを変更する場合に設定します
-s newspec	変更するスペックを指定します

- 使用例

- サーバーのスペック変更(-s)

以下は、単一ノードで特定のノードに属するすべてのサーバーのスペックを変更する例です。

```
$$1 tmaxs1 (tmadm): chtrc -s newspec
```

以下は、マルチノード環境で特定のノードにのみ新しいトレース・スペックを適用する例です。

```
$$1 tmaxh3 (tmadm): nodeset $HOSTNAME
node is set to $HOSTNAME
$$2 tmaxh3 (tmadm): chtrc -s newspec
```

#### – サーバー・グループのスペック変更(-g)

以下は、特定のサーバー・グループに属するすべてのサーバーのスペックを変更する例です。

```
$$1 tmaxs1 (tmadm): chtrc -g svgname -s newspec
```

#### – サーバーのスペック変更(-v)

以下は、特定のサーバー・グループに属するサーバーのスペックを変更する例です。

```
$$1 tmaxs1 (tmadm): chtrc -g svgname -s newspec
```

以下は、特定のサーバー・グループのサーバーのスペックを変更する例です。

```
$$3 tmaxs1 (tmadm): chtrc -g svgname -v server -s newspec
```

#### – サーバー・プロセスのスペック変更(-i)

以下は、特定のサーバー・プロセスのスペックを変更する例です。

```
$$4 tmaxs1 (tmadm): chtrc -g svgname -i sprno -s newspec
```

## 5.5.13. chlog

特定のエラー状況に迅速に対応するために、tmadminのchlogコマンドを使ってランタイム中にログレベルを動的に変更することができます。ログレベルの変更機能を使用してログを確認するには、必ず当該モジュールをデバッグ・モードで使用する必要があります。tmadminのcfgコマンドを使って、ログレベルが実際に変更されたか確認することができます。

### ● 使用方法

```
$$1 tmaxs1 (tmadm) : chlog [-t | -c | -v [server_name] |
                        -g [server_group_name]| -m] -l [loglvl]
```

項目	説明
[-t]	TMMのログレベルを設定します
[-c]	CLHのログレベルを設定します
[-v [server_name] ]	指定したサーバーのログレベルを設定します

項目	説明
[ -g [server_group_name] ]	指定したサーバー・グループのログレベルを設定します
[ -m ]	TMSのログレベルを設定します
-l [loglvl]	以下のいずれかを選択して実際のログレベルを設定します。  – COMPACT, BASIC, DETAIL, DEBUG1, DEBUG2, DEBUG3, DEBUG4  右側に行くほどより詳しいログを確認できます

- 例

- TMMの動的ログレベルの変更(-t)

以下は、TMMのログレベルを変更する例です。

```

tmaxh2:/data1/starbj81/tmax/config> tmaxadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog -t -l DEBUG4
log level is updated
$3 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
...
    tmmloglvl = DEBUG4,
...

```

- CLHの動的ログレベルの変更(-c)

以下は、CLHの動的ログレベルを変更する例です。

```

tmaxh2:/data1/starbj81/tmax/config> tmaxadmin -m -l
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog -c -l COMPACT
log level is updated
$$2 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
    tmmloglvl = DEBUG4,
    clhloglvl = COMPACT,
...

```

- TMの動的ログレベルの変更(-m, -g)

以下は、TMSの動的ログレベルを変更する例です。

```

$$3 tmaxh4 (tmadm): chlog -m -g svg32306X -l debug3
log level is updated
$$4 tmaxh4 (tmadm): cfg -g
    svg_name = svg32306X, svg_no = d
    tmsloglvl = DEBUG3,
...

```

#### – 特定のサーバー・グループの動的ログレベルの変更(-g)

以下は、特定のサーバー・グループの動的ログレベルを変更する例です。

```

$$4 tmaxh2 (tmadm): chlog -g svg3 -l DETAIL
log level is updated
$$5 tmaxh2 (tmadm): cfg -g
    svg_name = svg3, svg_no = 10002
    loglvl = DETAIL
...

```

## 5.5.14. chlog2

特定のエラー状況により迅速に対応するために、tmadminのchlog2コマンドを使ってランタイム中にログレベルを動的に変更することができます。ランタイム中にログレベルを変更できるtmadminのchlogコマンドに欠落している構成要素のTLM、CAS、CLL、RS、SQ、HMS、RQS、GATEWAYに対してもランタイム中にログレベルを変更、設定できます(既存のchlogの機能をすべて含みます)。tmadminのcfgコマンドを使って、ログレベルが実際に変更されたか確認することができます。

### ● 使用方法

```

$$1 tmaxs1 (tmadm) : chlog2 -t [type_name] [-n name] -l [loglvl]

```

項目	説明
-t [type_name]	ログレベルを変更するモジュールを設定します – TMM : TMM – CLH : CLH – TLM : TLM – CAS : CAS – RS : Real Server – SQ : Session Queue Server – TG : Tmax Grid Server

項目	説明
	<ul style="list-style-type: none"> <li>- TMS : TMS</li> <li>- HMS : HMS</li> <li>- RQ : Reliable Queue Server</li> <li>- GW : GATEWAY</li> <li>- SVG : Server Server Group</li> <li>- SVR : Server</li> <li>- CLL : CLL</li> </ul>
-n [ <i>name</i> ]	<p>TMS、HMS、RQは設定したサーバー・グループ名、GWはゲートウェイ名、SVGはサーバー・グループ名、SVRはサーバー名です。その他は、名前が不要です</p> <ul style="list-style-type: none"> <li>- TMM : TMM</li> <li>- CLH : CLH</li> <li>- TLM : TLM</li> <li>- CAS : CAS</li> <li>- RS : Real Server</li> <li>- SQ : Session Queue Server</li> <li>- TG : Tmax Grid Server</li> <li>- TMS : TMS</li> <li>- HMS : HMS</li> <li>- RQ : Reliable Queue Server</li> <li>- GW : GATEWAY</li> <li>- SVG : Server Server Group</li> <li>- SVR : Server</li> <li>- CLL : CLL</li> </ul>
-l [ <i>loglvl</i> ]	<p>以下のいずれかを選択して実際のログレベルを設定します。</p> <ul style="list-style-type: none"> <li>- COMPACT, BASIC, DETAIL, DEBUG1, DEBUG2, DEBUG3, DEBUG4</li> </ul> <p>右側に行くほどより詳しいログを確認できます</p>

- 例

- TMMの動的ログレベルの変更

以下は、TMMのログレベルを変更する例です。

```
tmaxh2:/data1/starbj81/tmax/config> tmaxadmin -l -m
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog2 -t TMM -l DEBUG4
log level is updated
$3 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
...
    tmmloglvl = DEBUG4,
...
```

- CLHの動的ログレベルの変更

以下は、CLHの動的ログレベルを変更する例です。

```
tmaxh2:/data1/starbj81/tmax/config> tmaxadmin -m -l
--- Welcome to Tmax Admin (Type "quit" to leave) ---

$$1 tmaxh2 (tmadm): chlog2 -t CLH -l COMPACT
log level is updated
$$2 tmaxh2 (tmadm): cfg -n
    node_name = tmaxh2, hostname = tmaxh2, node_no = 1
    tmmloglvl = DEBUG4,
    clhloglvl = COMPACT,
...
```

## 5.5.15. txcommit/txrollback

トランザクション処理中、TMS障害あるいはネットワーク障害により、一定時間の間トランザクションが処理されない場合、管理者がコミットまたはロールバックを再発行して当該トランザクションを終了する機能です。txcommitはPHASEの第2段階でDecisionがコミットの場合にのみ使用でき、txrollbackはPHASEの第1段階またはPHASEでDecisionがロールバックの場合にのみ使用できます。

- 使用方法

```
$$1 tmax1 (tmadm): txrollback(txr) | txcommit(txc) [-w] [-y]
                  <upper-global-xid><lower-global-xid>
```

項目	説明
[ -w ]	ドメイン・サブトランザクションの場合に使用します
[ -y ]	txcommit/txrollbackの確認手続きを行いません
<i>upper-global-xid</i>	処理するxidのgtidの前の4バイトです
<i>lower-global-xid</i>	処理するxidのgtidの後ろの4バイトです

#### 注

txcommit/txrollbackはtxqueryで照会されたトランザクションの再処理ツールであり、再発行の後処理結果を再びtxqueryで確認する必要があります。

## 5.5.16. wsgwreload

Tmaxの運用中にサービス情報の設定やWebサービス・ゲートウェイの設定を変更する状況が発生することがあります。このような場合、Tmaxシステムはtmadminのwsgwreloadコマンドを実行して適用します。コマンドが実行されると、Webサービス・ゲートウェイは新規要求されるサービスには応答せず、すでに処理中のサービスのみ処理した後、設定を適用してサービスの処理を始めます。

#### ● 使用方法

```
$$1 tmax1 (tmadm): wsgwreload [-i svrid[svrid]]
```

項目	説明
[ -i svrid [ svrid ] ]	適用したsvridリストを入力します

## 5.5.17. restart

restartコマンドは、現在起動しているサーバー・プロセスを停止してから、MINの数の分だけ再起動します。

#### ● 使用方法

```
$$1 tmax1 (tmadm): restart [-v[server_name]] [-g[server_group_name]]
```

項目	説明
[ -v[server_name] ]	適用したサーバーを再起動します
[ -g[server_group_name] ]	適用したサーバー・グループのサーバーを再起動します

#### ● 例

– 以下は使用例です。

```
– $$2 tmaxcl (tmadm): restart -v svr01021
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01021 finished

$$2 tmaxcl (tmadm): restart -g svgl
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01021 finished
>> suspend ok
>> down ok
>> boot ok
>> resume ok
>> reboot svr svr01022 finished
```

## 5.5.18. notify\_reconnect\_clh(nrc)

Tmaxを運用中に特定のサーバー・プロセスが特定のCLHに接続を確立できない状況が発生することがあります。そのようなとき自動で接続を確立できない場合に、手動で特定のサーバー・プロセスに特定のCLH接続を確立できるようにするコマンドです。

- 使用方法

```
$$1 tmaxl (tmadm): notify_reconnect_clh clh index spr index
```

項目	説明
<i>clh index clh index</i>	接続を希望するclh番号を入力します
<i>spr index spr index</i>	接続を希望するspr番号を入力します。  tmadminでst -pに設定する場合に出力するspriです



## 5.5.19. admnoti(an)

TCS、UCS、RDPサーバーにイベントを送信する機能です。tmadminで指定したサーバーにイベントを要求すると、当該サーバーはユーザーが指定したコールバック関数を呼び出してイベントを処理します。コールバック関数が登録されていない場合、イベントは無視されます。コマンドはtmadmin -mオプションでのみ動作できます。

TCS、UCS、RDPサーバーではtpregancb apiを利用してイベント処理コードを作成できます。

- 使用方法

```
$$1 tmax1 (tmadm): admnoti [-a | -g svgrname | -p spri | -P pid] [notification message]
```

項目	説明
-a -a	すべてのサーバー・プロセスにイベントを送信します
-g svgrname	svgrnameに属するすべてのサーバー・プロセスにイベントを送信します
-v svrname	svrnameに該当するすべてのサーバー・プロセスにイベントを送信します
-p spri	当該spriのサーバー・プロセスにイベントを送信します
-Ppid	当該pidのサーバー・プロセスにイベントを送信します
notification message	コールバック関数に渡すメッセージを入力します



# 第6章 IPv6の設定

本章では、TmaxでサポートしているIPv6の概念と対応範囲について説明します。

## 6.1. 概要

IPv6の基本概念と特性、IPv6への移行技術であるデュアル・スタック(Dual Stack)について説明します。

### 6.1.1. 基本概念

IPv6(Internet Protocol Version 6)は、インターネット・プロトコル・スタックのネットワーク層のプロトコルとして制定された次世代インターネット・プロトコル(IP)です。従来のインターネットを構築していたIPv4プロトコルの限界により、インターネットの持続的な発展が困難と判断され、その代案としてIETFで制定されました。

IPv4ではアドレスを32ビットで表現するため、4,294,967,296のアドレスを表現することができます。しかし、インターネットに接続されたコンピューターが幾何級数的に増加し、IPv4アドレスの枯渇が加速化されています。このような問題やその他の様々な問題を解消するためにIPv6が提案されました。

IPv6は128ビットで構成されており、 $2^{128}$ の $3.4 \times 10^{38}$ のアドレスを使用することができるため、無限大のアドレスを利用することが可能です。地表面の10m<sup>2</sup>あたり1つずつのIPv6/48ネットワークを提供できるほどの数です。

IPv4との大きな違いはアドレスの表現方法です。IPv6アドレスは、16ビットの4桁をコロン(:)で区切り16進数で表記します。

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7334
```

### 6.1.2. 移行技術

IPv4からIPv6に完全に移行するまでは、IPv4とIPv6を同時に使用することになります。基本プラットフォーム(OS、ルーター、スイッチなど)の業者は以下のような技術を提供してIPv4とIPv6を同時にサポートしています。

- デュアル・スタック(Dual Stack)

IPv4/IPv6デュアル・スタックは、IPv6ノードがIPv4専用のノードと互換性を維持するための最も簡単な方法です。IPv6/IPv4デュアル・スタック・ノードは、IPv4パケットとIPv6パケットのいずれからの送受信も可能なため、IPv4パケットを使用してIPv4ノードと直接互換されます。また、IPv6パケットを使用してIPv6ノードと直接互換されます。

- トンネリング(Tunneling)

トンネリングは、IPv6/IPv4ホストとルーターでIPv6データ・グラムをIPv4パケットにカプセル化してIPv4ルーティング・トポロジー領域を介して送信する方法です。トンネリングは既存のIPv4ルーティング・インフラを活用してIPv6トラフィックを送信する方法を提供します。

## 6.2. TmaxのIPv6対応

Tmaxでは設定によりIPv6を使用することができます。IPv6を設定しない場合は、基本的にIPv4で動作します。

IPv6の設定は必要に応じて環境変数、Tmax環境設定ファイルに適用する必要があります。クライアントでは環境設定ファイルを使用できないため、環境変数に設定します。

### 6.2.1. 対応機能

Tmaxは、クライアント、マルチノード、マルチドメイン、TCPゲートウェイなどがすべてソケットで実装されており、それぞれの構成要素がIPv4とIPv6のどちらを使用するかを決定できるように対応しています。

ソケットの概念に応じて以下のように構成できます。これにより、各モジュールのIPv6対応の設定を行います。

- 接続受信用の構成

以下は、接続受信用の構成におけるTmaxの構成要素と設定についての説明です。

構成要素	説明
cil	クライアントの接続を受信します – 環境設定ファイル：DOMAINセクション、NODEセクション – 環境変数：CLIENT_IPV6
tmm, clh	マルチノード環境でノード間の接続を受信します。 – 環境設定ファイル：DOMAINセクション、NODEセクション – 環境変数：SYSTEM_IPV6  外部サーバーの接続を受信します – 環境設定ファイル：DOMAINセクション、NODEセクション – 環境変数：EXTSVR_IPV6

構成要素	説明
racd	マルチノード環境でユーティリティーの接続を受信します – 環境設定ファイル：DOMAINセクション、NODEセクション – 環境変数：TMAX_RAC_IPV6、SYSTEM_IPV6
webagent(twagent)	WebAdminの接続を受信します – 環境変数：TMAX_WEBADM_IPV6
gateway([non]tx, [async] tuxedo, [async] java, wsgw, xagw)	接続を受信します – 環境設定ファイル：GATEWAYセクション – 環境変数：LOCAL_IPV6
hms	cluster hmsの接続を受信します – 環境設定ファイル：DOMAINセクション、NODEセクション – 環境変数：SYSTEM_IPV6
tcp g/w	接続を受信します – 環境設定ファイル：SERVERセクションのCLOPT項目、別の設定ファイル(『Tmax ゲートウェイガイド(TCPIP)』を参照) – 環境変数：-X SERVER_IPV6
tmsnmpd	SNNP要求の接続を受信します – 実行時の引数：upd6:portまたはtcp6:port
Javaの構成要素	jimax、webtasyncの接続を受信します。Javaの構成要素は、Javaの実行時に環境変数で設定します(WASで動作する場合は、WASの起動時に環境変数に適用します) – java.net.preferIPv4Stack – java.net.preferIPv6Addresses

● 接続要求用の構成

以下は、接続要求用の構成におけるTmaxの構成要素と設定についての説明です。

構成要素	説明
c client	Tmaxに接続を要求します

構成要素	説明
	<ul style="list-style-type: none"> <li>環境変数 : TMAX_HOST_IPV6, TMAX_BACKUP_IPV6</li> </ul>
tmm	<p>マルチノード環境で他のノード(tmm)に接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : DOMAINセクション、NODEセクション</li> <li>環境変数 : SYSTEM_IPV6</li> </ul>
clh	<p>マルチノード環境で他のノード(clh)に接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : DOMAINセクション、NODEセクション</li> <li>環境変数 : SYSTEM_IPV6</li> </ul>
tdl	<p>TDLユーティリティの実行時、マルチノード環境で他のノードに接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : DOMAINセクション、NODEセクション</li> <li>環境変数 : SYSTEM_IPV6</li> </ul>
hms	<p>cluster hms環境で他のノードのhmsに接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : DOMAINセクション、NODEセクション</li> <li>環境変数 : SYSTEM_IPV6</li> </ul>
xa client	<p>XAゲートウェイに接続します</p> <ul style="list-style-type: none"> <li>関数引数 : tmax_xa_open</li> <li>環境変数 : ipv6</li> </ul>
tcp g/w	<p>外部に接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : SERVERセクションのCLOPT項目、別の設定ファイル(『Tmax ゲートウェイガイド(TCPIP)』を参照)</li> <li>環境変数 : -X CLIENT_ IPV6</li> </ul>
gateway([non]tx, [async] tuxedo, [async] java)	<p>他のゲートウェイに接続します</p> <ul style="list-style-type: none"> <li>環境設定ファイル : GATEWAYセクション</li> <li>環境変数 : RGW_IPV6、RGW_B1_IPV6、RGW_B2_IPV6、RGW_B3_IPV6</li> </ul>
gateway(wsgw)	<p>Webサービス・サーバーに接続します</p> <ul style="list-style-type: none"> <li>別の環境設定ファイル(xml)</li> <li>URLを使用してIPv96を区別</li> </ul>

構成要素	説明
Javaの構成要素	<p>jtmax、webtasync、twadminの接続を要求します。Javaの実行時に環境変数で設定します(WASで動作する場合は、WASの起動時に環境変数に適用します)</p> <ul style="list-style-type: none"> <li>– java.net.preferIPv4Stack</li> <li>– java.net.preferIPv6Addresses</li> </ul>

受信時にIPv6をサポートするように設定すると、Tmaxは該当するソケットをIPv6用に作成します。このとき、Tmaxがインストールされているプラットフォームがデュアル・スタックのような移行技術を提供していれば、TmaxはIPv4とIPv6の接続要求を両方とも受信することができます。

ただし、プラットフォームがIPv6とIPv4の両方をサポートしても、TmaxでIPv6を設定していなければ、TmaxはIPv4の接続のみ受信することができ、IPv6要求は受信できません。

## 6.2.2. 付加機能

上記で説明した接続および受信のほか、ACL、tmadminによるci情報の照会など、情報入力・照会機能もサポートします。

以下は、IPv6対応の付加機能です。

- ACL

クライアント接続IPの制限

- 別の設定ファイルでIPv6アドレスに対応

- [2011::xxx]の形式で設定

- tmadmin

コマンド	説明
ci	接続したクライアントのIPを出力します
txgwi, ntxgwi, jgwi, ajgwi	接続したゲートウェイのIPを出力します
sqi	セッションQのクライアントIPを出力します

- クライアントおよびサーバーAPI

API	説明
tpgetsockname	サーバーとクライアントでTmaxシステム内部的に使用されるソケット・アドレスを取得する関数です。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.1.72. tpgetsockname」を参照してください
tpgetpeername	サーバーとクライアントで接続されたノードのソケット・アドレスを取得する関数であり、Tmaxシステムへの接続が確立した後、ノードのソケット・アドレスを返します。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.1.68. tpgetpeername」を参照してください
tpgetpeer_ipaddr	サーバーで接続されたノードのソケット・アドレスを取得する関数です。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.2.35. tpgetpeer_ipaddr」を参照してください
tpgetcliaddr_ipv6	Tmaxシステムに接続されたクライアントのうち、clidに該当するクライアントのIPとポート番号を取得する関数です。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.1.64. tpgetcliaddr_ipv6」「3.1.64. tpgetcliaddr_ipv6」を参照してください
tpbroadcast	Tmaxシステムに登録されたクライアントに要求していないメッセージを送信する関数です。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.1.46. tpbroadcast」を参照してください
tmadmin	Tmaxシステム管理ツールのtmadminにより照会できる統計情報を出力する関数です。詳細内容は、『 <i>Tmax リファレンスガイド</i> 』の「3.2.4. tmadmin」を参照してください



# 付録 A. ゲートウェイのCLOPTセクション

本章では、Tmaxドメインゲートウェイ、Javaゲートウェイ、TuxedoゲートウェイのCLOPT項目の設定オプションについて説明します。

## A.1. Tmax

以下は、Tmaxドメイン・ゲートウェイのCLOPT項目のオプションに関する説明です。

### A.1.1. Tmaxトランザクション・ドメイン・ゲートウェイ

GWTYPEが「TMAX」であるゲートウェイのCLOPT項目のオプションについて説明します。

オプション	説明
[ -r ]	<a href="#">「3.2.6. GATEWAYセクション」</a> のCLOPTの[-r]オプションの説明を参照してください
[ -h ]	<a href="#">「3.2.6. GATEWAYセクション」</a> のCLOPTの[-h]オプションの説明を参照してください
[ -i ]	オプションを設定しない場合、リモート・ゲートウェイと接続されていない状態では、要求がある場合にのみリモート・ゲートウェイへの接続を試みます。  オプションを設定した場合は、設定したNLIVEINQの周期で接続を試みます。この接続がない状態での要求に対しては、クライアントにTPNOREADYと応答します
[ -R DECISION ]	リモート・ゲートウェイのバージョンがTmax 3.xの場合に設定します。  リモート・ゲートウェイで始まったトランザクションに対してローカルでペンディング・トランザクションが発生した場合、そのトランザクションについてゲートウェイでロールバックするかコミットするかを決定する必要があります  – RBK : ロールバックします – COM : コミットします – IGN : 何にも行いません
[ -c TIME ]	ゲートウェイとCLH接続が終了された場合、TIME周期でCLHIに接続を試みます

オプション	説明
[-l "ip-addr,ip-addr2..."]	L4はアライブ・チェックのため、ゲートウェイに定期的にpingを投げ、接続を強制解除します。この場合、接続が切断される度に、ゲートウェイがログを残すことになるため、不要な情報が過剰に蓄積される問題が生じます。したがって、このオプションの設定して、pingによる接続を区別して不要なログを残さないようにします
[-m]	SERVERセクションのMAC = Y起動と同様、tmmが終了するとtmgw(nt)も終了します
[-k]	<a href="#">「3.2.6. GATEWAYセクション」</a> のCLOPTの[-k]オプションの説明を参照してください

## A.1.2. Tmax非トランザクション・ドメイン・ゲートウェイ

GWTYPEが「TMAXNONTX」であるゲートウェイのCLOPT項目のオプションについて説明します。

オプション	説明
[-h]	<a href="#">「3.2.6. GATEWAYセクション」</a> のCLOPTの[-h]オプションの説明を参照してください
[-i]	オプションを設定した場合、設定したNLIVEINQの周期で接続を試みます。接続がない状態での要求に対しては、クライアントにTPNOREADYと応答します。  オプションを設定していない場合、リモート・ゲートウェイと接続されていない状態では、要求がある場合にのみリモート・ゲートウェイへの接続を試みます
[-c TIME]	ゲートウェイとCLH接続が終了している場合、TIME周期でCLHに接続を試みます
[-n]	Tmaxのゲートウェイ間のチャンネル数は2つに固定されています。これにより、GW1を介して先に送信したメッセージ1をチャンネル1に、その次のメッセージ2をチャンネル2に送信した場合、受信するGW2側でメッセージ2が先に受信され処理される場合があります。  これを防ぐため、環境設定ファイルでGATEWAYセクションのCLOPTの[-n]オプションを追加し、ゲートウェイ間のチャンネル数を1つに設定することができます。[-n]オプションは引数を受け取りません
[-l "ip-addr,ip-addr2..."]	Tmaxトランザクション・ドメイン・ゲートウェイの[-l]オプションの説明を参照してください
[-m]	SERVERセクションのMAC = Y機能と同様、tmmが終了するとtmgw(nt)も終了します
[-k]	<a href="#">「3.2.6. GATEWAYセクション」</a> のCLOPTの[-k]オプションの説明を参照してください

## A.2. Java

以下は、JavaゲートウェイのCLOPT項目オプションについての説明です。

### A.2.1. JEUSゲートウェイ

GWTYPEが「JEUS」であるゲートウェイのCLOPT項目のオプションについて説明します。

オプション	説明
[ -D DEBUG_LEVEL ]	デバッグ・レベルを指定します。  – 1：要求および応答関連のログを出力します  – 2：XA関連ログを出力します  – 4：メッセージ・ダンプ関連ログを出力します  パイプ( )演算を行い、出力するログを決めることができます
[ -e LOGFILE_PATH ]	標準エラーを保存するログファイルのパスを設定します
[ -o LOGFILE_PATH ]	標準出力を保存するログファイルのパスを設定します
[ -r ]	WebT 3.14以降バージョンでは常に設定します
[ -h VERSION ]	– WebT 3.xバージョンとWebT 5.xバージョンでヘッダー設定をデフォルト値にする場合は1に設定します  – WebT 5.xバージョンでヘッダー設定をextendedV4にする場合は4に設定します
[ -t ]	複数のドメインから1つのJTmaxに接続する場合、必ず設定します。  複数のドメインが1つのJTmaxに接続する環境でこのオプションを設定しないと、復旧作業が異常動作します

### A.2.2. JEUS Asyncゲートウェイ

GWTYPEが「JEUS\_ASYNC」であるゲートウェイのCLOPT項目のオプションについて説明します。

オプション	説明
[ -D DEBUG_LEVEL ]	デバッグ・レベルを指定します。  – 1：要求および応答関連ログを出力します  – 2：XA関連ログを出力します

オプション	説明
	<ul style="list-style-type: none"> <li>– 4: メッセージ・ダンプ関連ログを出力します</li> <li>– 7: 1、2、3で出力されるすべてのメッセージ・ログを出力します</li> </ul> <p>パイプ( )演算を行い、出力するログを決めることができます</p>
[ -e LOGFILE_PATH ]	標準エラーを保存するログファイルのパスを設定します
[ -o LOGFILE_PATH ]	標準出力を保存するログファイルのパスを設定します
[ -r ]	常に設定します
[ -h 4 ]	常に4に設定します
[ -A TIME ]	外部と接続されたコネクションでアライブ・チェックを行う周期です。設定時間が過ぎても応答がなければ、接続を終了します
[ -a FILE_PATH ]	<p>RGWADDR、RGWPORTNOと設定したファイルの「IP:PORT」リストで接続を確立します。1つのトランザクションに対しては同じチャンネルにメッセージを送信します。</p> <p>デフォルトはラウンドロビンであり、各チャンネルあたり1回ずつメッセージを送信します。ファイルは1行あたり1つの「ip:port」形式で入力します</p>
[ -H ]	このオプションを設定すると、ログの出力時にアライブ・チェック・メッセージは出力しません
[ -t ]	<p>複数のドメインから1つのJTmaxに接続する場合、必ず設定します。</p> <p>複数のドメインが1つのJTmaxに接続する環境でこのオプションを設定しないと、復旧作業が異常動作します</p>
[ -m MAX_COUNT ]	<p>ゲートウェイからJTmaxに要求できる制限件数を設定します</p> <p>(デフォルト値: 500)</p>

## A.3. Tuxedo

以下は、TuxedoゲートウェイのCLOPT項目のオプションに関する説明です。

### A.3.1. Tuxedoゲートウェイ

GWTYPEが「TUXEDO」であるゲートウェイのCLOPT項目のオプションについて説明します。

オプション	説明
[ -a LOCAL_DOMAIN_NAME ]	<p>Tuxedoのドメイン・ゲートウェイと接続するために使用するドメイン名を設定します。</p> <p>この項目が設定されていない場合、以下のエラー・メッセージを出力します</p> <pre>3005 gateway name (-a domname) not defined</pre>
[ -e LOGFILE_PATH ]	標準エラーを保存するログファイルのパスを設定します
[ -o LOGFILE_PATH ]	標準出力を保存するログファイルのパスを設定します
[ -u UID ]	ACLを使用する場合、Tmaxクライアントが開始していないコールが要求されることがあるため、UIDを指定します
[ -F ]	Tuxedoが送信するメッセージ・タイプがFML16の場合に設定します。設定しない場合、FML32で処理します
[ -v ]	Tuxedoが送信するメッセージ・タイプがVIEW16の場合に設定します。設定しない場合、VIEW32で処理します
[ -r REMOTE_DOMAIN_NAME ]	<p>TuxedoからTmaxに接続を試みるとき、認証作業を行います。REMOTE_DOMAIN_NAMEでローカル・ドメイン名が設定されたTuxedoドメイン・ゲートウェイの接続のみ許可するよう検査を行います。</p> <p>認証に失敗したら、以下のエラー・メッセージを出力します。</p> <pre>0046 incorrect local name(REMOTE_DOMAIN_NAME), remote domain name(相手側のローカル・ドメイン名)</pre>
[ -h ]	<p>ローカル・ゲートウェイを複数設定する場合、各ゲートウェイがそれぞれリモートに接続しようとするため、TmaxのTuxgwでは1つの接続が確立したら、2番目の接続に対しては以下のエラー・メッセージを出力します。</p> <pre>(E) GATEWAY3010 connection error from remote gateway [TUXGW0002]</pre> <p>このようなメッセージが不要な場合、同オプションを適用すると、ログを出力しません</p>
[ -D ]	<p>Tuxedoゲートウェイを介して送受信されるメッセージのうち、正常に処理されないメッセージを以下の形式の情報とメッセージのバイナリ・データで出力します</p> <pre>&lt;時間&gt;:discarded [tmax   tuxedo] message(size:&lt;size&gt;)</pre>
[ -c time(sec) ]	<p>time周期で接続されているTUXEDOにtpcallを実行します。</p> <p>-Cオプションのサービス名を呼び出します。-Cオプションが設定されていない場合、dus\$%@tjqの名前でサービスを呼び出します。time*2まで応答が返されない場合は、接続が終了したと判断し、接続を終了します</p>

オプション	説明
[ -C サービス名 ]	-cオプションによって接続されているTUXEDOにtime周期でtpcallを実行時に使用するサービス名を入力します
[ -i ]	<p>TuxedoゲートウェイがCOUSINIに設定されている場合、チャンネル障害が発生しても正常にスケジューリングできるようにする機能です。</p> <p>既存のチャンネルIRTと異なる点は、Tuxedoゲートウェイはトランザクションでまとめられているため、最初の数個のコールに対しては失敗の応答を返すことがあります。その後は、正常なところにのみスケジューリングします</p>

### A.3.2. Tuxedo Asyncゲートウェイ

GWTYPEが「TUXEDO\_ASYNC」であるゲートウェイのCLOPT項目のオプションは、前述した「Tuxedoゲートウェイ」の説明を参照してください。

# 付録 B. 使用時の参考事項

本章では、Tmaxの使用時に参考となる内容について説明します。

## B.1. 複数のCLHの使用

以下は、複数のCLHの使用時におけるスケジューリングの特性について説明します。

### B.1.1. ASQCOUNT

ASQCOUNTはエンジン全体ではなく、1つのCLHに付与される値です。したがって、CLHごとにキューの状態を確認し、ASQCOUNTを適用して不足するサーバーを起動します。

たとえば、ASQCOUNT=4に設定し、1つのノードに現在待機中の要求が5つあるとした場合、現行キューのサイズが5ではなく、CLHごとにsvr待機キューを持っているため、1番CLHに2つ、2番CLHに3つになれる状態ではASQCOUNTに達していないため、サーバーを起動しません。

### B.1.2. 同時スケジューリング

複数の多数のCLHでは、すべてのCLHで同時に同じsprにスケジューリングする場合があります。これを避けるために、他のCLHの状態もチェックしてスケジューリングを行います。要求が急増したり、同時に要求がある場合には発生することがあります。このような場合、後で要求されたsprのソケットに待機し、tadminのキューでは表示されず、各CLHのst -pにRUN状態で表示される必要があります。このような状態にある要求はCLHのキューにあるのと同様に判断し、svrに実際のサービスを実行する前にqtimeoutをチェックして、時間がすでに過ぎた場合はTPEQPURGEと応答を返します。

---

#### 参考

無条件に実行させるには、SERVERセクションのCLOPT項目に[-B]オプションを追加します。

---

## B.2. ドメインゲートウェイCOUSINの設定方法

本節では、ドメイン・ゲートウェイ間のルーティングが必要な場合、ドメイン・ゲートウェイの可用性のための構成方法について説明します。

## B.2.1. SVRGROUPセクション

```
*SVRGROUP
ServerGroup Name  [COUSIN = cousin-svg-name,cousin-gw-name,]
                  [LOAD = numeric]
```

- COUSIN = literal
  - 範囲: 7999字以内
  - サーバー・グループとゲートウェイが共有すべきプロセスがあったり、サーバー・グループとゲートウェイ間ルーティングが必要な場合、サーバー・グループとゲートウェイの名前を指定します。Tmax v4.0 SP1バージョンからゲートウェイ名を指定することができます。
- LOAD = numeric
  - [「3.2.3. SVRGROUPセクション」](#)のLOAD項目を参照してください。

## B.2.2. GATEWAYセクション

```
*GATEWAY
Gateway Name      [LOAD = numeric]
```

- LOAD = numeric
  - [「3.2.3. SVRGROUPセクション」](#)のLOAD項目を参照してください。

## 例

以下のように設定する場合、svg1(tmaxh4)、svg2(tmaxh2)、gw1(tmaxh4)、gw2(tmaxh2)が1つのCOUSINにまとめられます。LOADの値がすべて1なのでクライアントでTOUPPERサービスを呼び出す場合、それぞれsvg1、svg2、gw1、gw2が1:1:1:1でサービスを処理します。

<domain1.m>

```
*DOMAIN
tmax1      SHMKEY = 78350, MINCLH = 2, MAXCLH = 3,
           TPORTNO = 8350, BLOCKTIME = 10000,
           MAXCPC = 100, MAXGW = 10, RACPORT = 3355

*NODE
tmaxh4     TMAXDIR = "/data1/tmaxqam/tmax",
```



```

tmaxh2      TMAXDIR = "/data1/tmaxqam/tmax",

*SVRGROUP
svg1        NODENAME = tmaxh4,
            COUSIN = "svg2, gw1, gw2", LOAD = 1
svg2        NODENAME = tmaxh2, LOAD = 1

*SERVER
svr2        SVGNAME = svg1

*SERVICE
TOUPPER      SVRNAME = svr2

*GATEWAY
#Gateway for domain 2
gw1         GWTYPE = TMAXNONTX,
            PORTNO = 7510,
            RGWADDR = "192.168.1.43",
            RGWPORTNO = 8510,
            NODENAME = tmaxh4,
            CPC = 3, LOAD = 1

#Gateway for domain 3
gw2         GWTYPE = TMAXNONTX,
            PORTNO = 7520,
            RGWADDR = "192.168.1.48",
            RGWPORTNO = 8520,
            NODENAME = tmaxh2,
            CPC = 3, LOAD = 1

```

<domain2.m>

```

*DOMAIN
tmax1       SHMKEY = 78500, MINCLH=2, MAXCLH=3,
            TPORTNO=8590, BLOCKTIME=10000

*NODE
tmaxh4      TMAXDIR = "/EMC01/starbj81/tmax",

*SVRGROUP
svg1        NODENAME = "tmaxh4"

*SERVER
svr2        SVGNAME = svg1

*SERVICE

```

```

TOUPPER      SVRNAME = svr2

*GATEWAY
gw1          GWTYPE = TMAXNONTX,
             PORTNO = 8510,
             RGWADDR="192.168.1.43",
             RGWPORTNO = 7510,
             NODENAME = tmaxh4,
             CPC = 3

```

<domain3.m>

```

*DOMAIN
tmax1        SHMKEY = 78500, MINCLH=2, MAXCLH=3,
             TPORTNO=8590

*NODE
tmaxh2        MAXDIR = "/data1/starbj81/tmax",

*SVRGROUP
svg2          ODENAME = "tmaxh2"

*SERVER
svr2          SVGNAME = svg2

*SERVICE
TOUPPER      SVRNAME = svr2

*GATEWAY
gw2          GWTYPE = TMAXNONTX,
             PORTNO = 8520,
             RGWADDR="192.168.1.48",
             RGWPORTNO = 7520,
             NODENAME = tmaxh2,
             CPC = 3

```

## 参考事項

Tmax v5.0 SP1バージョンまでは、ゲートウェイが属するノードは必ず当該ノードの一般サーバー・グループ (通常、ダミー・サーバー) も登録され、COUSINに一緒にまとめられている必要があります。上記の例 <domain1.m>におけるgw1はtmaxh4ノードに属しており、gw2はtmaxh2ノードに属しています。この場合、tmaxh4とtmaxh2の一般的なサーバー・グループも必ずCOUSINサーバー・グループ内に含まれている必要があります。

一方、Tmax v5.0 SP2/Tmax v6.0バージョンからは、上記のような不便さを改善し、ゲートウェイが属するノードはまとめず、ゲートウェイだけをまとめます。

- Tmax v5.0 SP2以前バージョンの環境設定

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2"
svg2      NODENAME = tmaxh2

*GATEWAY
gw1       NODENAME = tmaxh4
gw2       NODENAME = tmaxh2
```

- Tmax v5.0 SP2/Tmax v6.0以降バージョンの環境設定

Tmax v5.0 SP2/Tmax v6.0バージョンでは、Tmax v5.0 SP2以前バージョンの設定およびゲートウェイのみCOUSINIに設定して使用することができます。

Tmax v5.0 SP2以前バージョンの設定

```
*SVRGROUP
svg1      NODENAME = tmaxh4,
          COUSIN = "svg2, gw1, gw2"
svg2      NODENAME = tmaxh2

*GATEWAY
gw1       NODENAME = tmaxh4
gw2       NODENAME = tmaxh2
```

Tmax v5.0 SP2以降バージョンの設定

```
*GATEWAY
gw1       NODENAME = tmaxh4
          COUSIN = "gw2"
gw2       NODENAME = tmaxh2
```

---

## 参考

Tmax v5.0 SP1バージョンまでは、GATEWAYセクションにCOUSIN項目を直接設定することができず、一般的なサーバー・グループにだけCOUSIN項目の設定が可能でした。

一方、Tmax v5.0 SP2/Tmax v6.0バージョンからは、GATEWAYセクションでCOUSIN設定ができるように機能が追加されました。

---

## B.3. ドメイン・ゲートウェイの知能型ルーティング

マルチノード環境での負荷分散の際、ローカルノードのサーバーが終了された場合には、CLHでこれを検知し、リモート・ノードのサーバーに再スケジュールします。そのため、エラーの発生なくクライアントの要求をすべて処理することができます。リモートノードの特定のサーバーが非アクティブになった場合、ローカルノードのCLHはリモートノードでのサーバー状態が把握できず、リモートノードに要求を送るため、クライアントにはTPENOREADYエラーが返されることとなります。

このような問題を解決するため、知能型ルーティング機能(IRT)が導入されました。この機能は、静的負荷分散に対してリモートサーバーの状態を管理し、処理できるサーバーとして再スケジュール(IRT - Intelligent Routing)されます。したがって、リモートノードのプロセスが異常終了され、サービスが処理できなくなると、処理できるサーバーに再スケジュールされます。

---

### 参考

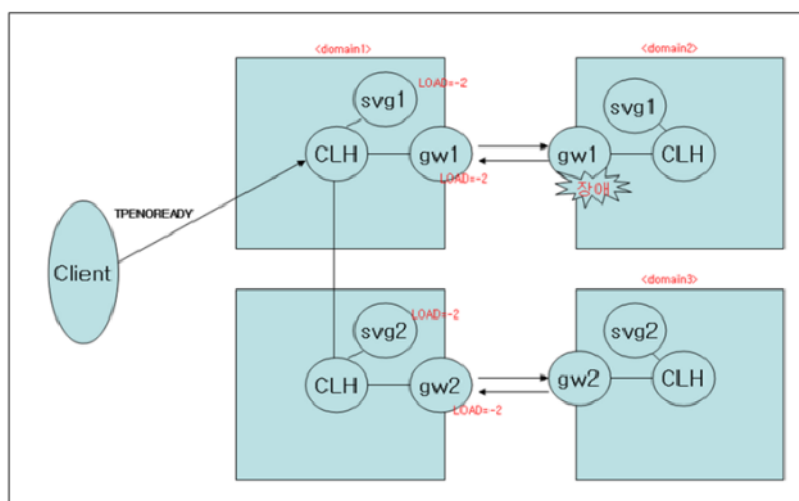
再スケジュールされるサーバー・グループの数は10に制限され、TPENOTREADYエラーに対してのみ再スケジュールされます。

---

### B.3.1. 従来のドメイン・ゲートウェイのルーティング

従来のドメイン・ゲートウェイは、リモートノードとリモートドメインの状態がチェックできなかったため、リモートノードの特定のサーバー(ゲートウェイを含む)に障害が発生してもそれを検知できず、そのままスケジューリングされ、クライアントにTPENOREADYエラーを返していました。

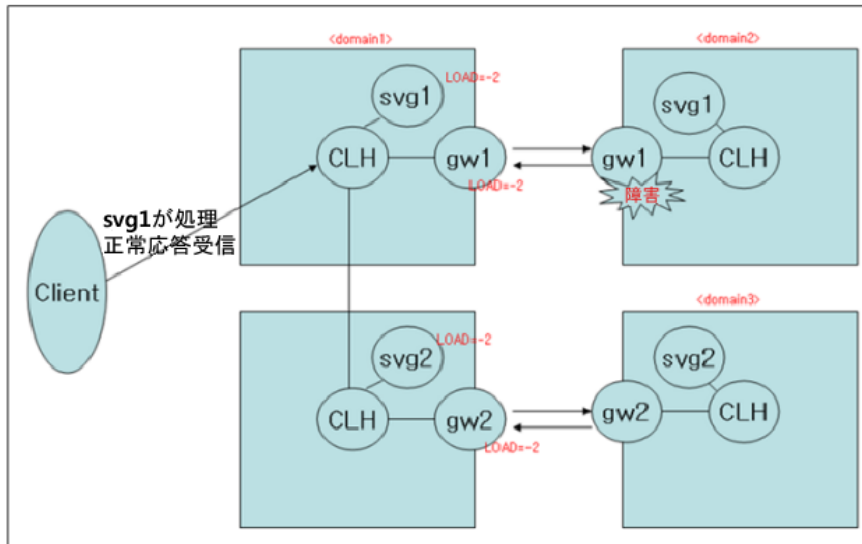
[図 B.1] 従来のドメイン・ゲートウェイのルーティング動作



### B.3.2. 現在のドメイン・ゲートウェイのルーティング

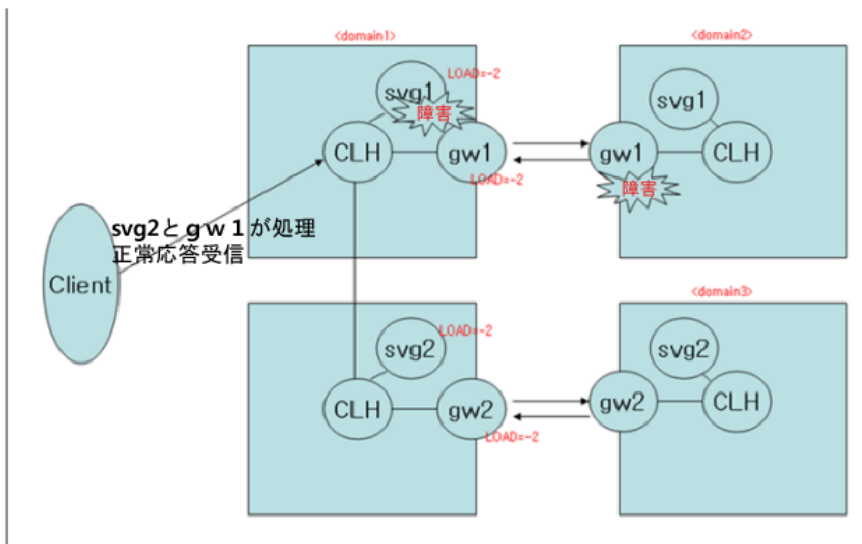
リモートノードとドメインの状態をチェックし、リモート・ゲートウェイに障害が発生した場合は、残りの他のサーバー・グループが処理を行います。つまり、svg1が代わりに処理します。

[図 B.2] 現在のドメイン・ゲートウェイのルーティング動作(1)



svg1がdummyサーバー・グループの場合、あるいはゲートウェイ間のネットワーク障害が発生した場合、リモートノードのsvg2とgw2がそれぞれ1:1で処理を行います。

[図 B.3] 現在のドメイン・ゲートウェイのルーティング動作(2)



### B.3.3. 知能型ルーティング機能に対応するゲートウェイ

Tmax v5.0 SP1バージョンまでは、非トランザクション・ゲートウェイに限ってのみ知能型ルーティング機能 (IRT)をサポートし、トランザクション・ゲートウェイには対応していませんでした。一方、Tmax v5.0 SP2/Tmax v6.0バージョンからは、TMAXNONTX、TMAX、JEUS、JEUS\_ASYNC、TUXEDO、TUXEDO\_ASYNC ゲートウェイで知能型ルーティング機能(IRT)をサポートします。

## B.4. 各バージョンにおけるFDの計算方法

TmaxでのFD(最大ユーザー)の計算方法はバージョンに応じて異なります。各バージョンでFDの値を計算する式は以下のとおりです。

#### ● Tmax v5.0 ~ Tmax v5.0 SP1 Fix#1 b.3.5

```
maximum user = max_fd
               - maxspr
               - maxcpc
               - maxtms
               - maxclh
               - Tmax内のFD(21)
               - [(nodecount(実際にアクティブ化したノード数) - 1(自身のノード)) * maxclh
                  * domain.cpc]
```

#### ● Tmax v5.0 SP1 Fix#1 b.3.7 ~ Tmax v6.0

```
maximum user = max_fd
               - maxspr
               - maxcpc
               - maxtms
               - maxclh
               - Tmax内のFD(21)
               - [マルチノードのためのCLH間の接続FD(maxnode(default:32) * maxclh *
                  domain.cpc * 2)]
```

「**max\_fd**」値は、Tmax内で定義したmax\_fd値と使用システムで定義された値のうち、小さい値が適用されます。「**domain.cpc**」は、tmadminのcfg -dで確認できるcpc値です。各項目の値は、tmadminのcfgコマンドを使って確認することができます。

# 索引

## シンボル

2PC, 153

## A

admnoti, 293

advertise, 267

    advertise -s, 268

ajgwinf, 250

## B

BLK, 235

## C

ca, 274

CAS, 3

caによるサーバーの動的追加, 275

caによるサーバー・グループの動的追加, 277

caによるサービスの動的追加, 274

caによるノードの動的追加, 277

cfg, 220

    cfg -d, 222

    cfg -g, 226

    cfg -n, 224

    cfg -s, 228

    cfg -v, 227

    cfg -w, 229

cfgopt, 229

    cfgopt -tmm, 230

chlog, 286

    chlog -c, 287

    chlog -g, 288

    chlog -m -g, 287

    chlog -t, 287

chlog2, 288

    chlog2, 290

chtrc, 285

    chtrc -g, 286

    chtrc -i, 286

    chtrc -s, 285

    chtrc -v, 286

CLH, 2

clhsinfo, 254

clientinfo, 257

clientinfo status

    CTING, 257

    NRDY, 257

    QED, 257

    RDY, 257

    RUN, 257

    UNR, 257

clitype

    ADSUB, 246

    ARCV, 246

    ASUB, 246

    DSUB, 246

    PUB, 246

    RCV, 246

    SND, 245

    SUB, 246

CLL, 2

## D

DDR, 157, 161

DLM, 157

domain, 4

DOMAINセクション, 173

ds, 284

## F

FDLFILE, 12, 15

FDLVERSION, 17

## G

GW, 3

gwinf, 247

## H

history, 220

HMS, 173  
HMSセクション, 177

## I

IPv4, 295  
IPv6, 295  
IPv6の移行技術  
    デュアル・スタック, 295  
    トンネリング, 296

## J

jgwinf, 250

## L

logend, 284  
logstart, 284

## M

mrbs, 273  
    mrbs -s, 273  
    mrbs -S, 274

## N

node status  
    CTNG, 255  
    NPING, 255  
    NPING2, 255  
    NRDY, 255  
    NSTRT, 255  
    RDY, 255  
    REG, 255  
    UNR, 255  
NODEセクション, 173  
nontxgwinf, 248  
notify\_reconnect\_clh, 292  
NRDY, 235

## P

PB\_TPCALL\_RETRY, 17  
PB\_TPCALL\_RETRY\_TIMEOUT, 17  
PB\_TUXCOMPAT, 17

PBLK, 235  
PUNADV, 235

## Q

qp, 283

## R

r, 256  
racd, 4, 199  
rbs, 270  
    rbs -s, 272  
    rbs -S, 273  
rbsの実行手順, 270  
RDY, 235  
restart, 291  
restat, 269  
Rolling Down [tmdown -R], 211  
RQS, 3  
rqs, 262  
    rqs -c, 264  
    rqs -f, 264  
    rqs -l, 263  
    rqs -s, 263  
RQセクション  
    BOOT, 170  
    BUFFERING, 171  
    FILEPATH, 172  
    FSYNC, 171  
    MAX\_MEMQCOUNT, 172  
    QSIZE, 171  
    RQ Name, 170  
    SVGNAME, 170  
rs, 266  
RUN, 235

## S

SDLFILE, 12, 15  
set, 281  
setopt, 282  
si, 258  
SLM, 157  
slog, 9



smtrc, 252

sp, 264

st, 230

st -n, 243

st -o, 242

st -p, 234

st -q, 244

st -s, 236

st -t, 239

st -x, 240

status

BLK, 235

NRDY, 235

PBLK, 235

PUNADV, 235

RDY, 235

RUN, 235

UNADV, 235

Unregistered, 235

SVRGROUPセクション

CPC, 168

NODENAME, 168

SVGTYPE, 169

SVRGROUP Name, 168

TMSNAME, 169

SVRGROUPセッション, 174

## T

ti, 219

tlog, 9

tadmin, 4

tmapm, 196

TMAX\_ACTIVATE\_AUTO\_TPSTART, 16

TMAX\_APPLY\_IPCPERM, 14

TMAX\_BACKUP\_ADDR, 12

TMAX\_BACKUP\_IPV6, 13, 16

TMAX\_BACKUP\_PORT, 12

TMAX\_BKAPPDIR, 11

TMAX\_CONNECT\_TIMEOUT, 15

TMAX\_CRYPT\_ALGORITHM, 15

TMAX\_DEBUG, 12, 16

TMAX\_ERR\_MSG, 12

TMAX\_HOST\_ADDR, 12, 15

TMAX\_HOST\_IPV6, 13, 16

TMAX\_HOST\_PORT, 12, 15

TMAX\_IPV6\_LINK\_IF, 13, 16

TMAX\_LOGLVL, 14, 17

TMAX\_PATHDIR, 12

TMAX\_RAC\_IPV6, 13

TMAX\_RAC\_PORT, 12

TMAX\_SEMANTICS, 12, 15

TMAX\_STRING\_NULL, 13

TMAX\_TENC\_CIPHER, 14

TMAX\_TENC\_KEYFILE, 14

TMAX\_TRACE, 12, 15

TMAX\_WEBADM\_IPV6, 13

TMAXDIR, 11

TMAXHOME, 12

tmboot, 4, 202

tmbootコマンドを実行時のtmconfigパス参照, 207

tmdown, 4, 209

TMM, 2

tmmsinfo, 255

TMS, 3

TPモニター, 151

txcommit, 290

txgwinf, 248

txq, 259

txq -g, 261

txq -w, 261

txq -w -r, 262

txq -x, 260

txrollback, 290

## U

ULOG, 15

ulog, 9

UNADV, 235

unadvertise, 267

unadvertise -s, 268

Unregistered, 235

## W

wsgwinf, 251

wsgwreload, 291

## か

環境設定

RQセクション, 170

RQトランザクション, 169

SVRGROUPセクション, 168

クライアントの動作状態, 257

## さ

静的管理, 6

サーバーの環境変数, 11

サービス・テーブル, 196

## た

知能型ルーティング機能(IRT), 312

同時性を保証するサーバーの切り替え, 272

動的管理, 6

ドメイン, 23

## な

ノードの動作状態, 255

## は

分散トランザクション処理, 151

## ま

マルチノードの同時性を保証するサーバーの切り替え,  
273

メッセージング・システム(Messaging System), 172