

OpenTP1 Version 7
分散トランザクション処理機能

TP1/Server Base Enterprise Option プログラム作成の手引

手引・文法書

3000-3-F52-80

前書き

■ 対象製品

・適用 OS : Red Hat Enterprise Linux Server 6 (64-bit x86_64), Red Hat Enterprise Linux Server 7 (64-bit x86_64), Red Hat Enterprise Linux Server 8 (64-bit x86_64)

P-8264-2A21 uCosminexus TP1/Server Base Enterprise Option(64) 07-93

・適用 OS : Red Hat Enterprise Linux Server 9 (64-bit x86_64)

P-8864-2A21 uCosminexus TP1/Server Base Enterprise Option(64) 08-01

これらのプログラムプロダクトのほかにもこのマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

記載の会社名、製品名などは、それぞれの会社の商標もしくは登録商標です。

■ 発行

2025 年 4 月 3000-3-F52-80

■ 著作権

All Rights Reserved. Copyright (C) 2008, 2025, Hitachi, Ltd.

変更内容

変更内容 (3000-3-F52-80) uCosminexus TP1/Server Base Enterprise Option(64) 08-01, uCosminexus TP1/Server Base Enterprise Option(64) 07-93

追加・変更内容	変更箇所
対象製品を更新した。 • P-1M64-2A21, P-1M64-1A11, P-9S64-2A11, P-9W64-2A21, および P-9W64-9211 を削除した。 • Red Hat Enterprise Linux Server 8 (64-bit x86_64)をサポートした (P-8264-2A21)。 • Red Hat Enterprise Linux Server 9 (64-bit x86_64)をサポートした (P-8864-2A21 の追加)。	—
【TP1/FSP 限定】 ee_rpc_poll_any_replies 関数の timeout の設定に関する説明を追加した。	4 章 ee_rpc_poll_any_replies
【TP1/FSP 限定】 CBLEERPC('POLLANYR')関数のデータ名 F の設定に関する説明を追加した。	5 章 CBLEERPC('POLLANYR')

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルは、次に示すプログラムプロダクトのアプリケーションプログラムの作成方法、API 関数の文法について説明したものです。

- uCosminexus TP1/Server Base Enterprise Option(64)

以降、このマニュアルでは、「uCosminexus TP1/Server Base Enterprise Option」および「uCosminexus TP1/Server Base Enterprise Option(64)」を「TP1/EE」と表記します。また、「uCosminexus TP1/EE/Extended Transaction Controller」を「XTC」と表記します。

■ 対象読者

TP1/EE で使うアプリケーションプログラムを作成するプログラマの方々を対象としています。

また、次に示す知識があることを前提としています。

- オペレーティングシステム（Linux など）とオンラインシステムの基礎的な知識
- 使用するマシンの操作に関する知識
- アプリケーションプログラムのコーディングに使う高級言語（C 言語（ANSI C, C++）、COBOL 言語）の文法の知識
- TP1/EE とあわせて使用する TP1/Server Base の知識

なお、このマニュアルの記述は、マニュアル「TP1/Server Base Enterprise Option 使用の手引」の知識があることを前提としていますので、あらかじめお読みいただくことをお勧めします。

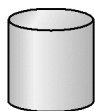
■ 図中で使用する記号

このマニュアルの図中で使用する記号を、次のように定義します。

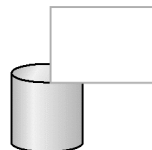
●ワークステーション、
●入出力の動作
端末



●ファイル



●ファイルの内容



●プログラム



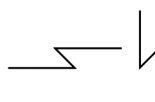
●障害



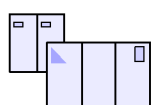
●キュー



●通信回線



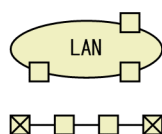
●メインフレーム



●分散機



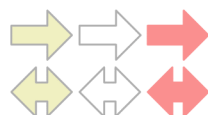
●ネットワーク
(LAN)



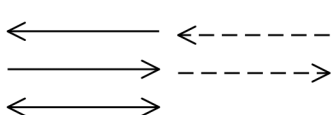
●ネットワーク
(WAN)



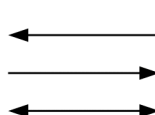
●データの流れ



●制御の流れ



●その他の流れ



■ JIS コード配列のキーボードと ASCII コード配列のキーボードとの違いについて

JIS コード配列と ASCII コード配列では、次に示すコードで入力文字の違いがあります。このマニュアルの文字入力例（コーディング例）の表記は、JIS コード配列（日本語のキーボード）に従った文字に統一しています。

コード	JIS コード配列	ASCII コード配列
(5c) ₁₆	'¥' (円記号)	'\' ' (バックスラッシュ)
(7e) ₁₆	'ー' (オーバーライン)	'`' (チルダ)

■ 文法の記号

このマニュアルで使用する各種記号を説明します。特に断りがない場合は、C 言語、COBOL 言語の説明で共通に使用する記号です。なお、C 言語と COBOL 言語で共通の内容を説明するときは、C 言語の用語に統一して説明します。C 言語の用語に対応する COBOL 言語の用語については、マニュアル「OpenTP1 プログラム作成リファレンス COBOL 言語編」でご確認ください。

(1)文法記述記号

指定する値の説明で使用する記号の一覧を示します。

文法記述記号	意味
[]	この記号で囲まれている項目は省略してもよいことを示します。この記号は、C 言語の説明だけに使用しています。 (例) {EENOFLLAGS EERPC_NOREPLY} [EERPC_TPNOTRAN] EERPC_TPNOTRAN は省略できることを示します。
{ }	この記号で囲まれている複数の項目のうちから一つを選択することを示します。項目の区切りは で示します。この記号は、C 言語の説明だけに使用しています。 (例) {EENOFLLAGS EERPC_NOREPLY} これは EENOFLLAGS と EERPC_NOREPLY の二つの項目のうち、どちらかを指定することを示します。
【 】	第 1 章、第 2 章、および第 3 章では、関数を最初に説明する個所で、C 言語の関数名に対応する COBOL 言語の関数名をこの記号で囲んで表記しています。それ以降は、C 言語の関数名に統一して説明します。
【XTC 限定】	関連製品である XTC を使用する場合に限定して関係する記述であることを示します。XTC を使用しない場合は記述を無視してください。
【TP1/FSP 限定】	関連製品である TP1/FSP を使用する場合に限定して関係する記述であることを示します。TP1/FSP を使用しない場合は記述を無視してください。
△	半角スペースを示します。この記号は、COBOL 言語の説明だけに使用しています。

(2) 引数として指定する記号の説明

関数の引数として指定する記号を示します。

記号	意味
	一つの引数に複数の項目を指定するときに、項目同士の区切りを示します。この記号を項目との間に入れて指定してください。この記号は、C 言語の説明だけに使用しています。 (例) EENOFLLAGS EERPC_TPNOTRAN

(3) 構文要素記号

説明で使用する構文要素の一覧を示します。すべて半角文字で指定します。

構文要素記号	意味
英字	アルファベット (A~Z, a~z) と _ (アンダスコア)
英数字	英字と数字 (0~9)
数字	0~9
パス名	英数字, /, および . (ピリオド) (ただし、パス名は使用する OS に依存)

(4)設定する値の条件

設定する値の条件を示します。

設定する値	指定できる値の条件
サービスグループ名	31 バイトのアスキー文字列で設定します。ただし NULL 文字、空白、ピリオド、および@（アットマーク）は除きます。 C 言語をご使用の場合、引数に設定するときは NULL 文字で終了させてください。この NULL 文字は文字列の長さには数えません。 COBOL 言語をご使用の場合、データ領域に設定するときは空白で終了させてください。この空白は文字列の長さには数えません。
サービス名	31 バイトのアスキー文字列で設定します。ただし NULL 文字、空白は除きます。 C 言語をご使用の場合、引数に設定するときは NULL 文字で終了させてください。この NULL 文字は文字列の長さには数えません。 COBOL 言語をご使用の場合、データ領域に設定するときは空白で終了させてください。この空白は文字列の長さには数えません。
物理ファイル名	スペシャルファイル名に 14 バイト以内の名称を続けたパス名で設定します。このパス名は 63 文字以内で設定してください。
論理ファイル名	1～8 バイト以内の、先頭が英字の英数字の名称で設定します。

■ 謝辞

COBOL 言語仕様は、CODASYL (the Conference on Data Systems Languages：データシステムズ言語協議会) によって、開発された。原開発者に対し謝意を表すとともに、CODASYL の要求に従って以下の謝辞を掲げる。なお、この文章は、COBOL の原仕様書「CODASYL COBOL JOURNAL OF DEVELOPMENT 1984」の謝辞の一部を再掲するものである。

いかなる組織であっても、COBOL の原仕様書とその仕様の全体又は一部分を複製すること、マニュアルその他の資料のための土台として原仕様書のアイデアを利用することは自由である。ただし、その場合には、その刊行物のまえがきの一部として、次の謝辞を掲載しなければならない。書評などに短い文章を引用するときは、"COBOL"という名称を示せば謝辞全体を掲載する必要はない。

COBOL は産業界の言語であり、特定の団体や組織の所有物ではない。

CODASYL COBOL 委員会又は仕様変更の提案者は、このプログラミングシステムと言語の正確さや機能について、いかなる保証も与えない。さらに、それに関連する責任も負わない。

次に示す著作権表示付資料の著作者及び著作権者

FLOW-MATIC (Sperry Rand Corporation の商標), Programming for the Univac

(R) I and II, Data Automation Systems, Sperry Rand Corporation 著作権表示

1958 年, 1959 年;

IBM Commercial Translator Form No.F 28-8013, IBM 著作権表示 1959 年;

FACT, DSI 27A5260-2760, Minneapolis-Honeywell, 著作権表示 1960 年

は、これら全体又は一部分を COBOL の原仕様書中に利用することを許可した。この許可は、COBOL 原仕様書をプログラミングマニュアルや類似の刊行物に複製したり、利用したりする場合にまで拡張される。

■ 注意事項

製品形名 P-8864-2A21 ではデータベースとして Oracle を使用できません。データベースは HiRDB だけ使用できます。

目次

前書き	2
変更内容	3
はじめに	4

1	TP1/EE のアプリケーションプログラム	17
1.1	ユーザアプリケーションプログラムと業務形態の関係	18
1.1.1	クライアント／サーバ形態のユーザアプリケーションプログラム	18
1.1.2	ユーザアプリケーションプログラムのトランザクション処理	20
1.2	ユーザアプリケーションプログラムの種類	21
1.2.1	サーバ UAP (SPP) としての TP1/EE	21
1.3	SPP (TP1/EE) の作成の概要	24
1.4	アプリケーションプログラミングインタフェースの機能	25
2	TP1/EE の基本機能	26
2.1	リモートプロシジャコール	27
2.1.1	リモートプロシジャコールの実現方法	27
2.1.2	リモートプロシジャコールでのデータの受け渡し	28
2.1.3	リモートプロシジャコールの形態	29
2.1.4	同期応答型 RPC	30
2.1.5	非同期応答型 RPC	33
2.1.6	非応答型 RPC	36
2.1.7	通信先を指定したリモートプロシジャコール	42
2.1.8	リモートプロシジャコールでメッセージを送受信できない場合	47
2.1.9	サービスのネスト	49
2.1.10	サービス要求の応答待ち時間の参照と更新	49
2.1.11	トランザクショナル RPC	50
2.1.12	トランザクションの処理から非トランザクショナル RPC の発行	51
2.1.13	別トランザクションによる RPC 同期応答	52
2.1.14	TP1/Server Base と RPC 通信をする場合の設定	54
2.2	トランザクション制御	55
2.2.1	クライアント／サーバ形態の通信のトランザクション	55
2.2.2	リモートプロシジャコールの形態と同期点の関係	58
2.2.3	2 相コミット	62
2.2.4	トランザクションの最適化	65
2.3	タイマトランザクションの登録	72

2.3.1	即時登録方式	72
2.3.2	トランザクション同期登録方式	73
2.3.3	時刻または経過時間の指定方法	73
2.3.4	タイマトランザクションの取り消し	74
2.3.5	システム終了時のタイマトランザクションの扱い	75
2.3.6	時刻または経過時間の指定と TP1/EE サービス定義との関係	75
2.4	DB キューによるメッセージ制御	79
2.4.1	DB キューを使用した AP 間通信	79
2.4.2	DB キューサービスのトランザクションのコミット決着	82
2.4.3	DB キューを使用したシステム間通信	82
2.4.4	DB キュー読み出しサーバ変更機能	87
2.4.5	DB キュー書き込み抑止機能	87
2.4.6	DB キューの通番の管理	88
2.4.7	DB キュー自動読み出し停止機能	88
2.4.8	DB キューメッセージスキップ機能	88
2.4.9	DB キュースケジュール状態一括変更機能	89
2.5	オンラインバッチ機能	90
2.5.1	オンラインバッチ機能の概要	90
2.5.2	オンラインバッチ機能の処理形態	91
2.5.3	オンラインバッチ機能で実行できる処理	93
2.6	運用コマンド実行機能	95
2.6.1	運用コマンド実行機能の環境設定	99
2.6.2	運用コマンド実行機能の処理	99
2.6.3	運用コマンド実行機能の時間監視	100
2.6.4	運用コマンド実行サーバ	101
2.6.5	運用コマンド実行機能使用時の定義	102
2.7	エラートランザクション	107
2.7.1	不正サービス名称検出通知 (ERRTRN1)	107
2.7.2	サービス閉塞通知 (ERRTRN2)	108
2.7.3	UAP 異常終了通知 (ERRTRN3)	109
2.7.4	タイマトランザクション起動破棄通知 (ERRTRN4)	110
2.7.5	コミット決着失敗通知 (ERRTRNR)	111
2.7.6	送信障害通知 (ERRTRNS)	112
2.7.7	エラートランザクションの登録に失敗した場合の後処理	113
2.8	リモート API 機能	114
2.8.1	常設コネクションの確立	116
2.8.2	API の代理実行	117
2.8.3	常設コネクションの解放	119
2.8.4	メッセージ受信処理を実行できない場合	121

2.9	スレッド制御	122
2.9.1	保護区／非保護区の制御	122
2.10	資源の排他制御	124
2.10.1	排他の対象となる資源	124
2.10.2	排他の種類	124
2.10.3	排他待ち限界経過時間の指定	125
2.10.4	排他の解除方法	125
2.10.5	ロックマイグレーション	125
2.10.6	排他のテスト	127
3	SPP (TP1/EE) の作成の詳細	128
3.1	SPP (TP1/EE) の作成	129
3.1.1	サービス関数のコーディング	130
3.1.2	UAP 共用ライブラリの作成	133
3.1.3	UAP 実行形式ファイルの作成	133
3.1.4	SPP (TP1/EE) の環境設定	134
3.1.5	SPP (TP1/EE) の実行	135
3.2	TP1/EE および XTC のアプリケーションプログラムと API	136
3.2.1	TP1/EE および XTC の API	136
3.2.2	トランザクションで利用できる API 関数	139
4	TP1/EE および XTC のライブラリ関数の文法 (C 言語編)	145
	関数の説明形式	146
	サービス関数の作成 (SPP)	148
	トランザクションインタフェース情報	157
	システム運用の管理 (ee_adm_～)	178
	ee_adm_call_command	179
	DB キュー制御 (ee_dbq_～)	183
	ee_dbq_acsctl	185
	ee_dbq_clear	189
	ee_dbq_conctl	192
	ee_dbq_inqsrv	195
	ee_dbq_inquire	198
	ee_dbq_inquire2	202
	ee_dbq_msgget	208
	ee_dbq_msgput	212
	ee_dbq_msgrefer	215
	ee_dbq_msgrefer2	218
	ee_dbq_obscancel	221
	ee_dbq_obschdata	223
	ee_dbq_obsend	225
	ee_dbq_obsinquire	227

ee_dbq_obsrbskip 230
ee_dbq_obsrestart 232
ee_dbq_obsstop 234
ee_dbq_obsstart 236
ee_dbq_obsstop 239
ee_dbq_rbskip 241
ee_dbq_skip 243
ee_dbq_srvctl 246
資源の排他制御 (ee_lck_～) 248
ee_lck_get 249
ee_lck_release_all 251
ee_lck_release_byname 252
メッセージログの出力 (ee_log～) 254
ee_logprint 255
高速メッセージ送信 (ee_mch_～) 259
ee_mch_cmtsend 【XTC 限定】 260
ee_mch_cmtsend_get 【XTC 限定】 264
ee_mch_cmtsend_sync 【XTC 限定】 268
ee_mch_otqbctl 【XTC 限定】 271
ee_mch_otqinquire 【XTC 限定】 273
ee_mch_otqskip 【XTC 限定】 276
メモリ管理 (ee_mem_～) 278
ee_mem_getwkseg 279
ee_mem_mdpstart 【XTC 限定】 281
ee_mem_rlswkseg 283
リモートプロシジャコール (ee_rpc_～) 284
ee_rpc_call 285
ee_rpc_call_to 293
ee_rpc_cmtsend 301
ee_rpc_cmtsend_to 305
ee_rpc_discard_further_replies 308
ee_rpc_discard_specific_reply 310
ee_rpc_get_error_descriptor 312
ee_rpc_get_watch_time 313
ee_rpc_poll_any_replies 314
ee_rpc_reply_send 319
ee_rpc_reply_suspend 322
ee_rpc_set_watch_time 324
滞留メッセージ受信 (ee_scd_～) 325
ee_scd_clear_rollback_cnt 【XTC 限定】 326
ee_scd_msg_receive 【XTC 限定】 328
ee_scd_msg_rollback 【XTC 限定】 331
ee_scd_msg_skip 【XTC 限定】 333
スレッド制御 (ee_thd_～) 335
ee_thd_abdctl 336

ee_thd_gettime 338
 ee_thd_protend 344
 ee_thd_protstart 345
 ee_thd_thdcntget 346
 ee_thd_uifaget 348
 タイマトランザクション (ee_tim_～) 350
 ee_tim_execap 351
 ee_tim_execap_cancel 356
 ee_tim_execap_cancel_all 358
 トラブルシュート (ee_trb_～) 360
 ee_trb_uinfo_tm 361
 ee_trb_utrace_put 363
 トランザクション制御 (ee_trn_～) 365
 ee_trn_chained_commit 366
 ee_trn_chained_rollback 369
 ee_trn_getinfo 372
 ee_trn_rm_close_mark 375
 ee_trn_rollback_mark 376
 ee_trn_set_exp_time 378

5 TP1/EE および XTC のライブラリ関数の文法 (COBOL 言語編) 380

COBOL-UAP 作成用プログラムの説明形式 381
 サービスプログラムの作成 (SPP) 383
 トランザクションインタフェース情報 393
 システム運用の管理 (CBLEEADM) 414
 CBLEEADM('COMMAND ') 415
 DB キュー制御 (CBLEEDBQ) 419
 CBLEEDBQ('ACSCCTL ') 421
 CBLEEDBQ('CLEAR ') 425
 CBLEEDBQ('CONCTL ') 429
 CBLEEDBQ('INQSRV ') 432
 CBLEEDBQ('INQUIRE ') 435
 CBLEEDBQ('INQUIRE2') 439
 CBLEEDBQ('MSGGET ') 445
 CBLEEDBQ('MSGPUT ') 449
 CBLEEDBQ('MSGREFE2') 452
 CBLEEDBQ('MSGREFER') 455
 CBLEEDBQ('OBSCAN ') 458
 CBLEEDBQ('OBSCDT ') 460
 CBLEEDBQ('OBSEND ') 463
 CBLEEDBQ('OBSINQ ') 465
 CBLEEDBQ('OBSRBSP ') 469
 CBLEEDBQ('OBSRST ') 471
 CBLEEDBQ('OBSSKIP ') 474
 CBLEEDBQ('OBSSTART') 477

CBLEEDBQ('OBSSTOP ')	480
CBLEEDBQ('RBSKIP ')	482
CBLEEDBQ('SKIP ')	484
CBLEEDBQ('SRVCTL ')	487
資源の排他制御 (CBLEELCK)	489
CBLEELCK('GET ')	490
CBLEELCK('RELALL ')	493
CBLEELCK('RELNAME ')	495
メッセージログの出力 (CBLEELOG)	497
CBLEELOG('PRINT ')	498
高速メッセージ送信 (CBLEEMCH)	503
CBLEEMCH('CMTSND ') 【XTC 限定】	504
CBLEEMCH('CMTSNDGT') 【XTC 限定】	508
CBLEEMCH('CMTSNDSY') 【XTC 限定】	512
CBLEEMCH('OTQBKCTL') 【XTC 限定】	516
CBLEEMCH('OTQINQIR') 【XTC 限定】	519
CBLEEMCH('OTQSKIP ') 【XTC 限定】	522
メモリ管理 (CBLEEMEM)	525
CBLEEMEM('GETWK ')	526
CBLEEMEM('MDPSTART') 【XTC 限定】	528
CBLEEMEM('RLSWK ')	530
リモートプロシジャコール (CBLEERPC)	532
CBLEERPC('CALL ')	533
CBLEERPC('CALLTO ')	541
CBLEERPC('CMTSND ')	549
CBLEERPC('CMTSNDTO')	553
CBLEERPC('DISCARDF')	557
CBLEERPC('DISCARDS')	559
CBLEERPC('GETERDES')	561
CBLEERPC('GETWATCH')	563
CBLEERPC('POLLANYR')	565
CBLEERPC('R-SEND ')	570
CBLEERPC('R-SPEND ')	573
CBLEERPC('SETWATCH')	576
滞留メッセージ受信 (CBLEESCD)	578
CBLEESCD('CLRRBCNT') 【XTC 限定】	579
CBLEESCD('MSGRECV ') 【XTC 限定】	581
CBLEESCD('MSGRLBK ') 【XTC 限定】	584
CBLEESCD('MSGSKIP ') 【XTC 限定】	586
スレッド制御 (CBLEETHD)	588
CBLEETHD('ABDCTL ')	589
CBLEETHD('CNTGET ')	591
CBLEETHD('GETTIME ')	593
CBLEETHD('PRTEND ')	599
CBLEETHD('PRTSTART')	601

CBLEETHD('UIFAGET ')	603
タイマトランザクション (CBLEETIM)	605
CBLEETIM('CANCEL ')	606
CBLEETIM('CANCELAL')	609
CBLEETIM('EXECAP ')	611
トラブルシュート (CBLEETRB)	616
CBLEETRB('UINFOTM ')	617
CBLEETRB('UTPUT ')	619
トランザクション制御 (CBLEETRN)	621
CBLEETRN('C-COMMIT')	622
CBLEETRN('C-ROLL ')	625
CBLEETRN('EXPTIME ')	629
CBLEETRN('GETINFO ')	631
CBLEETRN('RMCLSMK ')	635
CBLEETRN('ROLLMARK')	637

6 ユーザOWNコーディング (UOC) 639

6.1	UOC インタフェース	640
6.2	UOC で使用できる関数	641
6.3	オンライン開始 UOC インタフェース	642
6.3.1	形式	642
6.3.2	説明	642
6.3.3	パラメタの内容	642
6.3.4	XTC が値を設定する項目	643
6.3.5	リターン値	643

7 コーディング例 644

7.1	C 言語のコーディング例	645
7.2	COBOL 言語のコーディング例	652

付録 676

付録 A	トランザクション種別	677
付録 B	各バージョンの変更内容	679
付録 B.1	変更内容 (3000-3-F52-70)	679
付録 B.2	変更内容 (3000-3-F52-60)	679
付録 B.3	変更内容 (3000-3-F52-50)	680
付録 B.4	変更内容 (3000-3-F52-40)	680
付録 B.5	変更内容 (3000-3-F52-30)	681
付録 B.6	変更内容 (3000-3-F52-20)	682
付録 B.7	変更内容 (3000-3-F52-10)	687
付録 B.8	変更内容 (3000-3-F52)	688
付録 B.9	変更内容 (3000-3-983-40)	690

付録 B.10	変更内容 (3000-3-983-30)	691
付録 C	このマニュアルの参考情報	692
付録 C.1	関連マニュアル	692
付録 C.2	このマニュアルでの表記	694
付録 C.3	英略語	696
付録 C.4	KB (キロバイト) などの単位表記について	698

索引 699

1

TP1/EE のアプリケーションプログラム

この章では、TP1/EE のアプリケーションプログラムの概要について説明します。

1.1 ユーザアプリケーションプログラムと業務形態の関係

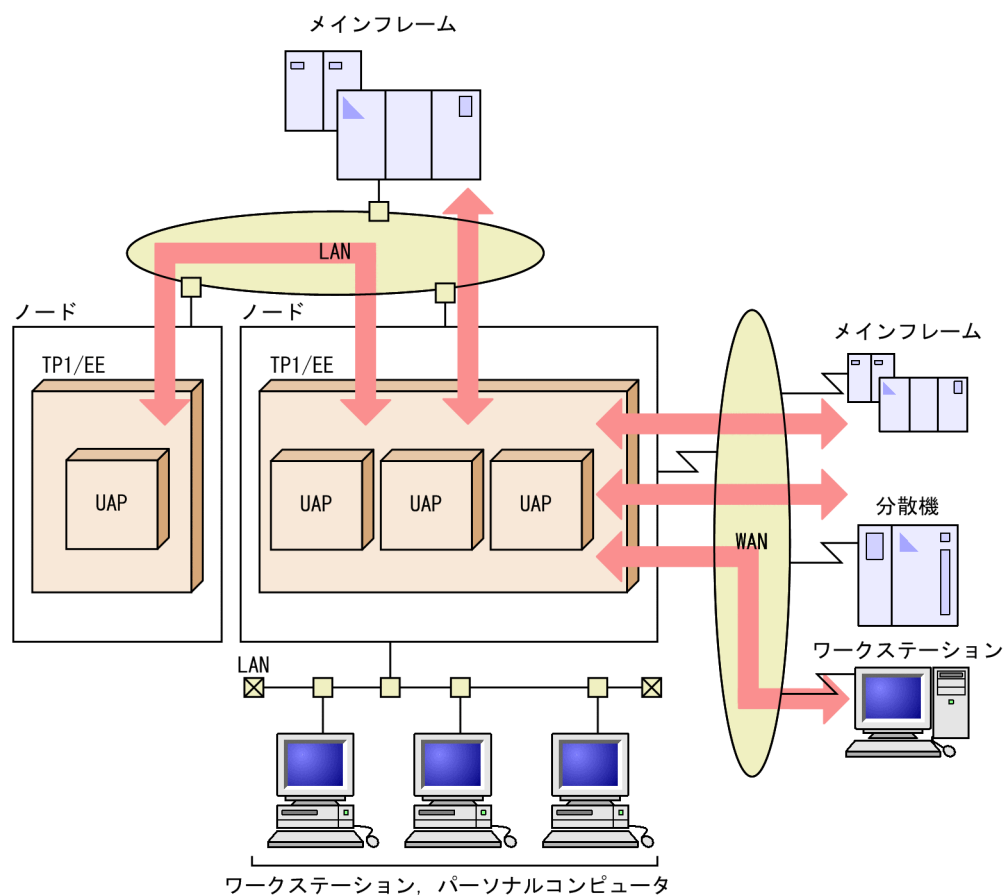
TP1/EE のユーザアプリケーションプログラム (UAP) は、ネットワーク (LAN, または WAN) でつながれているメインフレーム, ワークステーション (WS), パーソナルコンピュータ (PC), および分散機と通信する, オンライントランザクション処理を実現するために作成します。

TP1/EE の UAP でできる通信形態は, 次のとおりです。

- ・ クライアント／サーバ形態の UAP

TP1/EE と UAP のネットワーク内の位置を次の図に示します。

図 1-1 TP1/EE と UAP のネットワーク内の位置



1.1.1 クライアント／サーバ形態のユーザアプリケーションプログラム

クライアント／サーバ形態の UAP では, ほかの UAP の処理を呼び出して業務処理を実行できます。呼び出して利用できるプログラムの単位をサービス, サービスを提供するプロセスをサーバといいます。UAP のサーバをユーザサーバといいます。

クライアント／サーバ形態の通信では、サービスを要求する UAP（クライアント UAP）とサービスを提供する UAP（サーバ UAP）で一つの業務になります。サーバ UAP のサービスは、複数のクライアント UAP で共用できます。

TP1/EE は、TP1/Server Base のサーバ UAP として動作します。

サービスを要求するときは、リモートプロシジャコール（RPC）を使います。RPC では、サーバ UAP に論理的に付けた名称（サービス名）でサービスを要求できます。

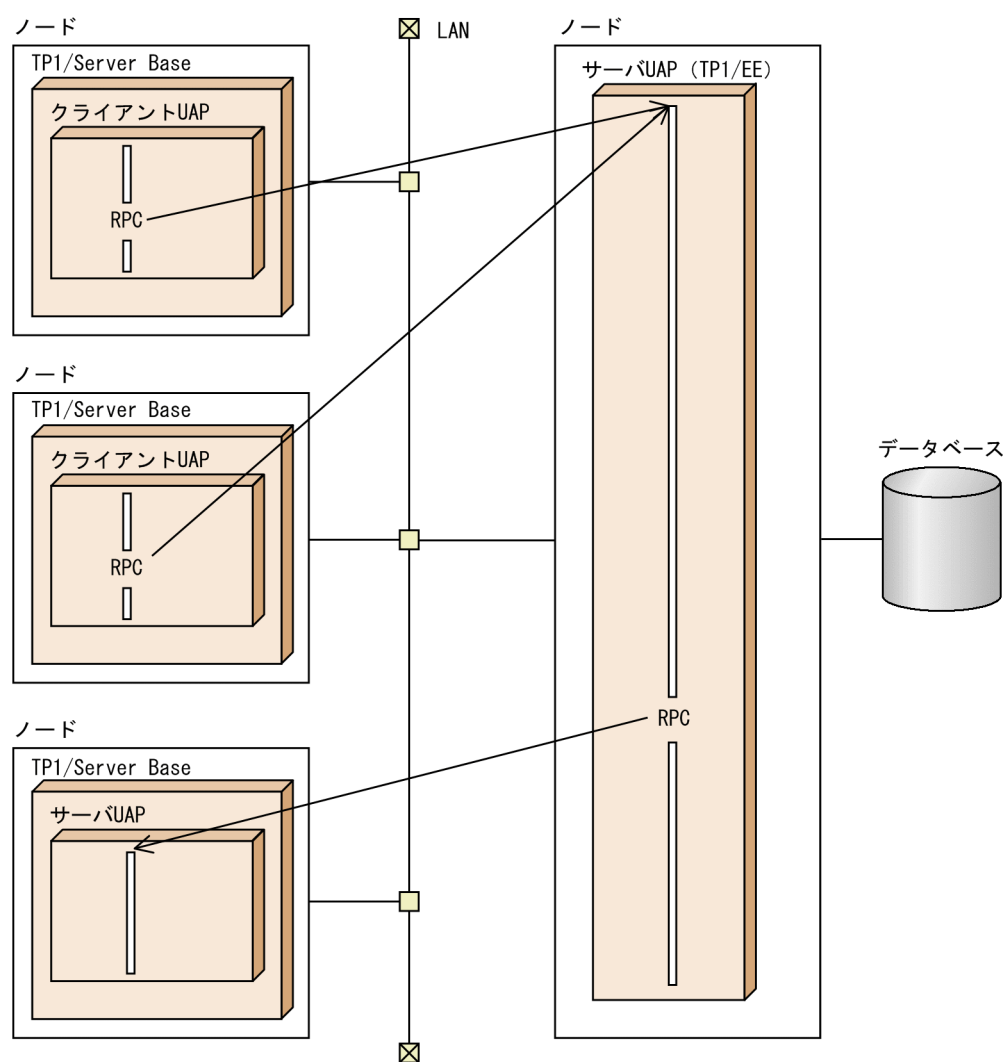
TP1/EE を使用する場合、クライアント／サーバ形態の通信プロトコルに TCP/IP を使えます。UAP でノード間の通信プロトコルを意識する必要はありません。

ノードについて：

このマニュアルで意味する「ノード」とは、ネットワークにつながれた、TP1/EE が稼働する一つの計算機（マシン）のことです。ただし、マルチ TP1/EE の場合は、複数の TP1/EE から構成される一つのノードになります。

クライアント／サーバ形態の UAP の概要を次の図に示します。

図 1-2 クライアント／サーバ形態の UAP の概要



1.1.2 ユーザアプリケーションプログラムのトランザクション処理

UAP の処理は、業務処理ごとの単位に区切って、それぞれの処理の結果を有効にするか無効にするかを明確に分ける必要があります。処理が有効であるか無効であるか、どちらかに必ず決定させる単位をトランザクションといいます。

TP1/EE では、RPC を使ったクライアント／サーバ形態の UAP の処理をトランザクションとして処理できます。

このように TP1/EE では、従来のデータコミュニケーションでのトランザクション処理の信頼性を、クライアント／サーバ形態の UAP で実現できます。

トランザクションの業務処理ごとの区切りを**同期点**といいます。トランザクションの処理が同期点に達した時点で、トランザクションの処理が正常に終了したか（有効）異常が起こったか（無効）を決定します。処理が正常に終了したとする同期点取得を**コミット**といいます。同期点まで正常に終了できなかったトランザクションの処理は、TP1/EE でそれまでの処理を取り消して、その処理がなかったように回復します。このような同期点処理を**ロールバック**（部分回復）といいます。

1.2 ユーザアプリケーションプログラムの種類

TP1/EE を適用するクライアント／サーバ形態のシステムで使用する UAP について説明します。

- サービス提供プログラム (SPP Service Providing Program)

SPP は、サーバ UAP (サービスを提供する側の UAP) およびクライアント UAP (サービスを要求する側の UAP) として動作します。クライアント UAP からの要求に対してサービスを提供したり、サーバ UAP に対してサービスを要求したりします。SPP は、TP1/Server Base が前提となります。TP1/EE は、TP1/Server Base の SPP として動作します。

- サービス利用プログラム (SUP Service Using Program)

SUP は、クライアント UAP (サービスを要求する側の UAP) として動作する、クライアント専用の UAP です。サーバ UAP に対してサービスを要求します。SUP は、TP1/Server Base が前提となります。

SUP の詳細については、マニュアル「OpenTP1 プログラム作成の手引」を参照してください。

- クライアントユーザプログラム (CUP Client User Program)

CUP は、クライアント UAP (サービスを要求する側の UAP) として動作する、クライアント専用の UAP です。WS または PC から、TP1/Client のライブラリ関数を使って、サーバ UAP に対してサービスを要求します。CUP は、OpenTP1 クライアント機能 (TP1/Client/W, TP1/Client/P, または TP1/Client/J) が前提となります。

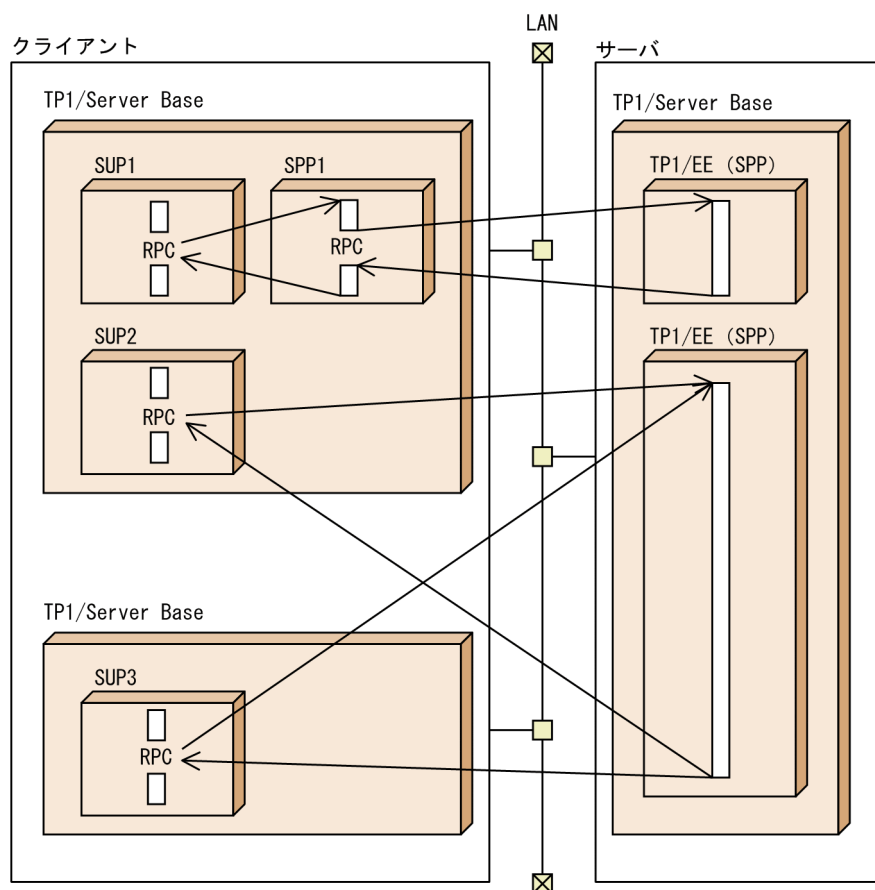
CUP の詳細については、マニュアル「OpenTP1 クライアント使用の手引 TP1/Client/W, TP1/Client/P 編」またはマニュアル「OpenTP1 クライアント使用の手引 TP1/Client/J 編」を参照してください。

1.2.1 サーバ UAP (SPP) としての TP1/EE

TP1/EE は、サーバ UAP (SPP) として使用します。TP1/EE は、UAP 共用ライブラリと実行形式ファイルを組み合わせて SPP になります。

クライアント／サーバ形態の通信で使用する UAP の概要を次の図に示します。

図 1-3 クライアント／サーバ形態の通信で使用する UAP の役割と位置の概要



クライアント UAP は関数を呼び出して、SPP のサービスを要求します。SPP がどのノードにあるかは、クライアント UAP で意識する必要はありません。

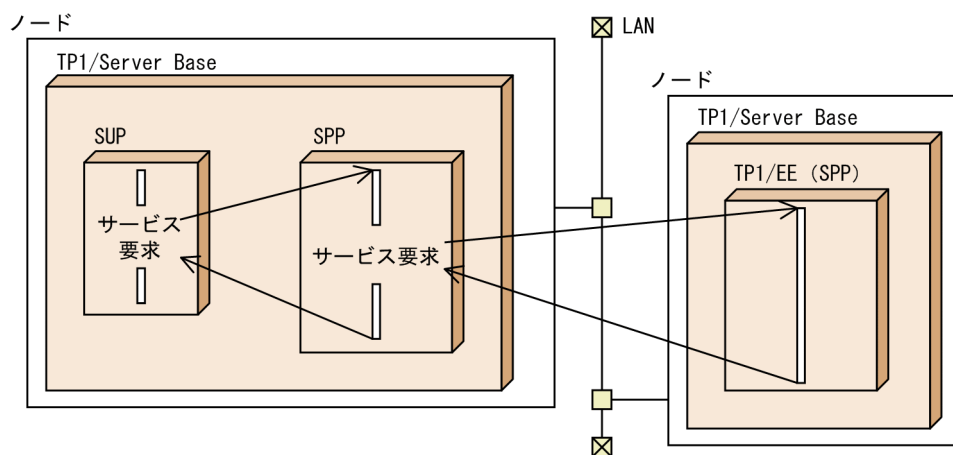
SPP は、サービスを要求されてから業務を開始します。サービスを要求されない間は、要求されるのを待っている状態となります。

SPP では、SPP のノードにあるユーザファイルへアクセスして、サーバの業務をします。TP1/EE 専用のファイルヘライブラリ関数でアクセスしたり、DBMS へ SQL 文でアクセスしたりできます。

SPP からさらに別の SPP へサービスを要求して、業務処理をネストさせることもできます。

SPP の概要を次の図に示します。

図 1-4 SPP の概要



1.3 SPP (TP1/EE) の作成の概要

SPP (TP1/EE) を作成する手順を、次に説明します。

1. SPP のサービス関数のコーディング

詳細については、「[3.1.1 サービス関数のコーディング](#)」を参照してください。

2. UAP 共有ライブラリの作成

詳細については、「[3.1.2 UAP 共有ライブラリの作成](#)」を参照してください。

3. UAP 実行形式ファイルの作成

詳細については、「[3.1.3 UAP 実行形式ファイルの作成](#)」を参照してください。

4. SPP (TP1/EE) の環境設定

詳細については、「[3.1.4 SPP \(TP1/EE\) の環境設定](#)」を参照してください。

5. SPP (TP1/EE) の実行

詳細については、「[3.1.5 SPP \(TP1/EE\) の実行](#)」を参照してください。

1.4 アプリケーションプログラミングインタフェースの機能

TP1/EE のライブラリ関数を使ってできる機能を、次に示します。

- リモートプロシジャコール
UAP 間で、関数呼び出しと同様の方法で通信できます。
- トランザクション制御
UAP の処理をトランザクションとして制御できます。
- タイマトランザクション
指定時間経過後にトランザクションを起動できます。
- DB キューによるメッセージ制御
DB キューを使用して、メッセージを管理できます。
- システム運用の管理
UAP からコマンドを実行できます。
- メッセージログの出力
UAP からユーザ任意の情報を、メッセージログとして出力できます。
- スレッド制御
スレッドの生成と終了、プロセスダウンの局所化、および TP1/EE のハングアップ監視ができます。
- 高速メッセージ送信
クライアントとサーバの間で UDP プロトコルで高速通信できます。
- 滞留メッセージ受信
滞留メッセージを UAP 起動中に受信できます。
- メモリ管理
ワークセグメントの割り当て、および返還ができます。
- トラブルシュート
ユーザ固有のトレースデータを取得できます。

各機能を実現する関数の一覧については、「[3.2 TP1/EE および XTC のアプリケーションプログラムと API](#)」を参照してください。

関数の詳細な説明については、「[4. TP1/EE および XTC のライブラリ関数の文法 \(C 言語編\)](#)」、または「[5. TP1/EE および XTC のライブラリ関数の文法 \(COBOL 言語編\)](#)」を参照してください。

2

TP1/EE の基本機能

この章では、TP1/EE の各種機能のうち、主な実現方法が API である機能について説明します。

主な実現方法がオペランドやコマンドである各種機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.1 リモートプロシジャコール

TP1/EE は、ほかの UAP と RPC を使用して通信できます。RPC とは、クライアント UAP からサーバ UAP へサービスを要求し、サービスを要求されたサーバ UAP は要求元のクライアント UAP へ処理結果を返す通信です。TP1/EE の UAP では、サービスを提供する UAP がネットワークのどのノードにあるかを意識しなくても、UAP のサービスを関数呼び出しのように要求できます。サービスを要求されたサーバ UAP は、クライアント UAP とは別のスレッドで実行されます。

TP1/EE では OpenTP1 独自のインタフェースを使用して、RPC を実現します。通信プロトコルには TCP/IP を使用します。

2.1.1 リモートプロシジャコールの実現方法

TP1/EE を使用するシステムの場合、サーバ UAP は SPP です。クライアント UAP は、SUP, SPP, および CUP です。TP1/EE は SPP として動作します。

遠隔サービスを要求するサービス関数を次に示します。

クライアント UAP が TP1/EE の場合

- ee_rpc_call 関数 【CBLEERPC('CALL ')】
- ee_rpc_call_to 関数 【CBLEERPC('CALLTO ')】

クライアント UAP が TP1/Server Base の SUP, SPP の場合、または TP1/Client の UAP (CUP) の場合

- dc_rpc_call 関数 【CBLDCRPC('CALL ')】
- dc_rpc_call_to 関数

なお、dc_rpc_call_to 関数に対応する COBOL-UAP 作成用プログラムはありません。

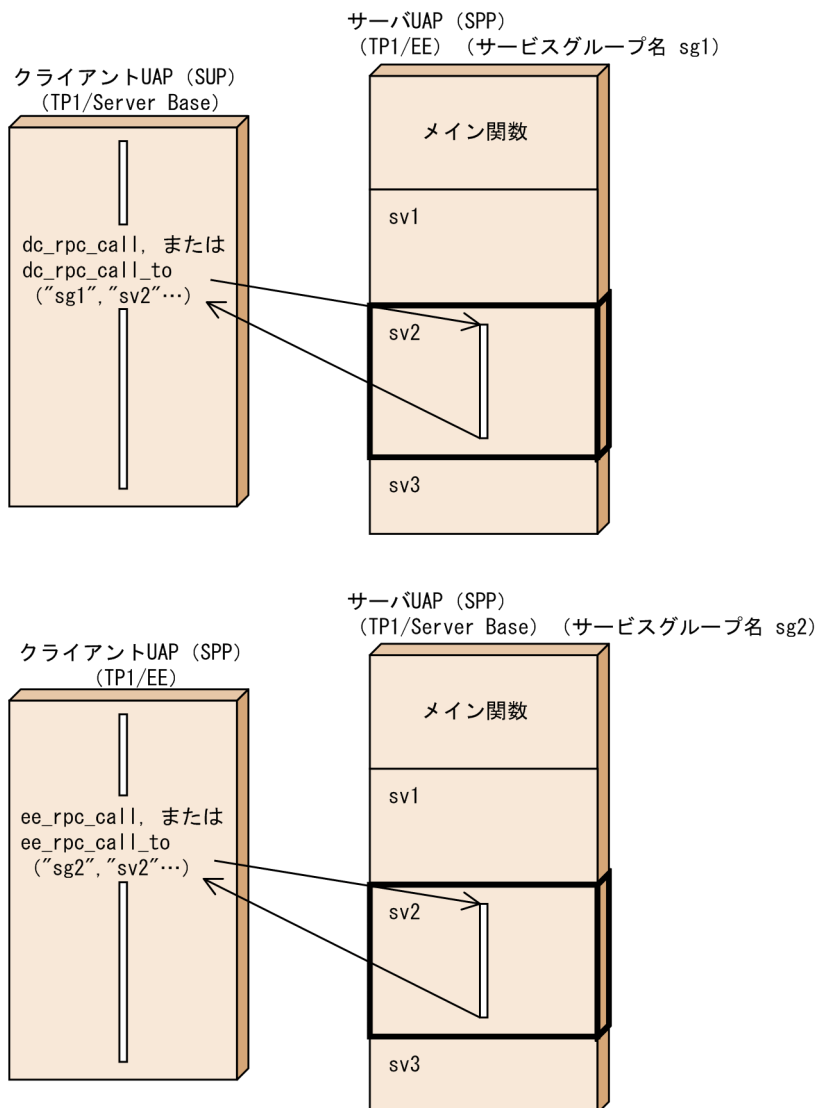
クライアント UAP から、サービスを要求する関数を使ってサーバ UAP (SPP) のサービスを要求できます。クライアント UAP からサービスを要求するときは、サーバ UAP のサービスグループ名とサービス名を引数に設定した dc_rpc_call 関数、dc_rpc_call_to 関数、ee_rpc_call 関数、または ee_rpc_call_to 関数を呼び出します。

dc_rpc_call 関数、または ee_rpc_call 関数では、要求されるサービスが、クライアント UAP と別のノードにあっても同じノードにあってもかまいません。要求されるサービスがどのノードにあるかは、TP1/Server Base のネームサービスで管理しているので、UAP で意識する必要はありません。これに対し、dc_rpc_call_to 関数、または ee_rpc_call_to 関数を使えば、特定のサービス要求先にサービスを要求できます。dc_rpc_call_to 関数、または ee_rpc_call_to 関数では、サービスグループ名とサービス名を引数に設定するのに加え、ホスト名またはノード識別子を指定して、特定のサービス要求先にサービスを要求します。

クライアント UAP が、開始していない TP1/EE（サーバ UAP）にサービスを要求すると、dc_rpc_call (dc_rpc_call_to) 関数、ee_rpc_call (ee_rpc_call_to) 関数はエラーリターンします。

RPC を使った通信のクライアント／サーバの関係を次の図に示します。

図 2-1 RPC を使った通信のクライアント／サーバの関係



2.1.2 リモートプロシジャコールでのデータの受け渡し

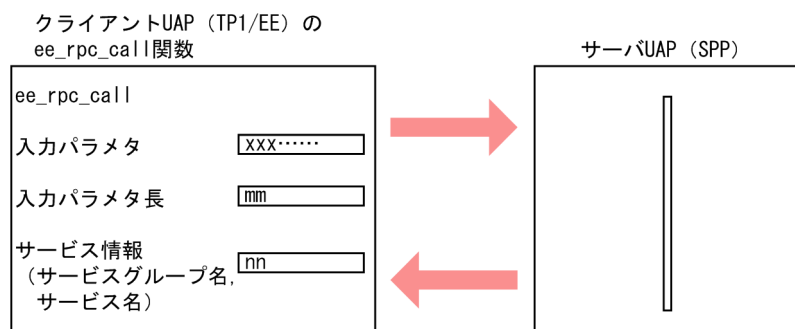
TP1/EE がクライアント UAP (SPP) の場合のデータの受け渡しについて説明します。

(1) ee_rpc_call 関数または ee_rpc_call_to 関数の引数の設定

クライアント UAP (TP1/EE) からサーバ UAP (SPP) にサービスを要求するときには、ee_rpc_call 関数または ee_rpc_call_to 関数の引数に入力パラメタ、入力パラメタ長などを設定します。

ee_rpc_call 関数の場合を例に RPC のデータの受け渡しを次の図に示します。

図 2-2 RPC のデータの受け渡し



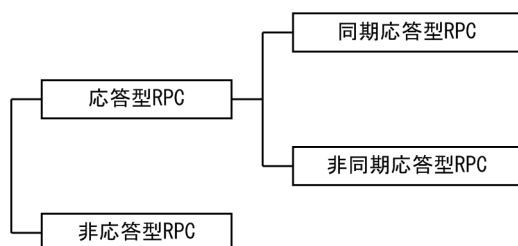
(2) フラグメントメッセージの送信

メッセージのサイズが送信バッファサイズ（メモリ関連定義の `send_message_buf_size` オペランドで指定）を超えた場合、クライアント UAP（TP1/EE）はメッセージを複数のフラグメントに分割して送信します。

2.1.3 リモートプロシジャコールの形態

RPC の形態を次の図に示します。

図 2-3 RPC の形態



RPC の形態は、クライアント UAP の遠隔サービスを要求するサービス関数（`ee_rpc_call` 関数, `ee_rpc_call_to` 関数, `dc_rpc_call` 関数, または `dc_rpc_call_to` 関数）のフラグで指定します。

- 応答型 RPC

サーバ UAP の処理結果をクライアント UAP に返す RPC です。応答型 RPC には、遠隔サービスを要求するサービス関数を呼び出してから、サーバ UAP の処理結果が返ってくるのを待つ同期応答型 RPC と、処理結果を非同期に受信する非同期応答型 RPC があります。

- 非応答型 RPC

サーバ UAP の処理結果をクライアント UAP に返さない RPC です。遠隔サービスを要求するサービス関数の呼び出し後にリターンして、処理を続けます。クライアント UAP ではサーバ UAP の処理結果を受信できません。

トランザクション処理で RPC を使った場合の、同期点と RPC との関係については「[2.2.2 リモートプロシジャコールの形態と同期点の関係](#)」を参照してください。

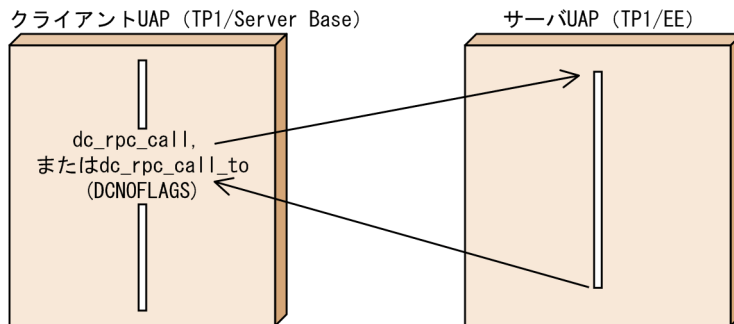
2.1.4 同期応答型 RPC

サーバ UAP の処理結果をクライアント UAP に返す RPC です。クライアント UAP では、サーバ UAP の処理結果を受信するまで処理を待ち合わせます。

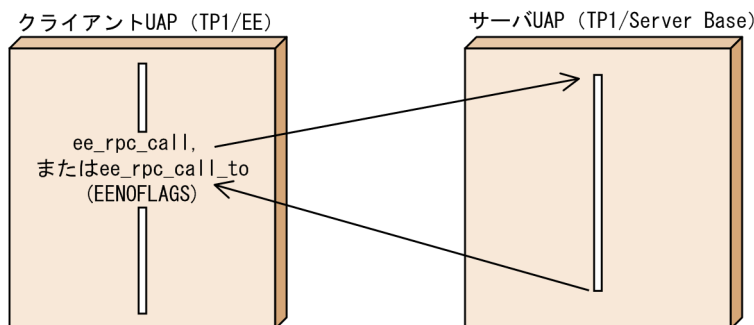
同期応答型 RPC を次の図に示します。

図 2-4 同期応答型 RPC

●TP1/EEがサーバの場合



●TP1/EEがクライアントの場合



同期応答型 RPC は時間監視をします。監視時間内に応答が返らない場合はエラーリターンします。

(1) TP1/EE がサーバ UAP の場合の同期応答型 RPC

サーバ UAP (TP1/EE) では、クライアント UAP (SUP, SPP) からの RPC メッセージを受信するための受信スレッドは、自ポート番号ごとに存在します。

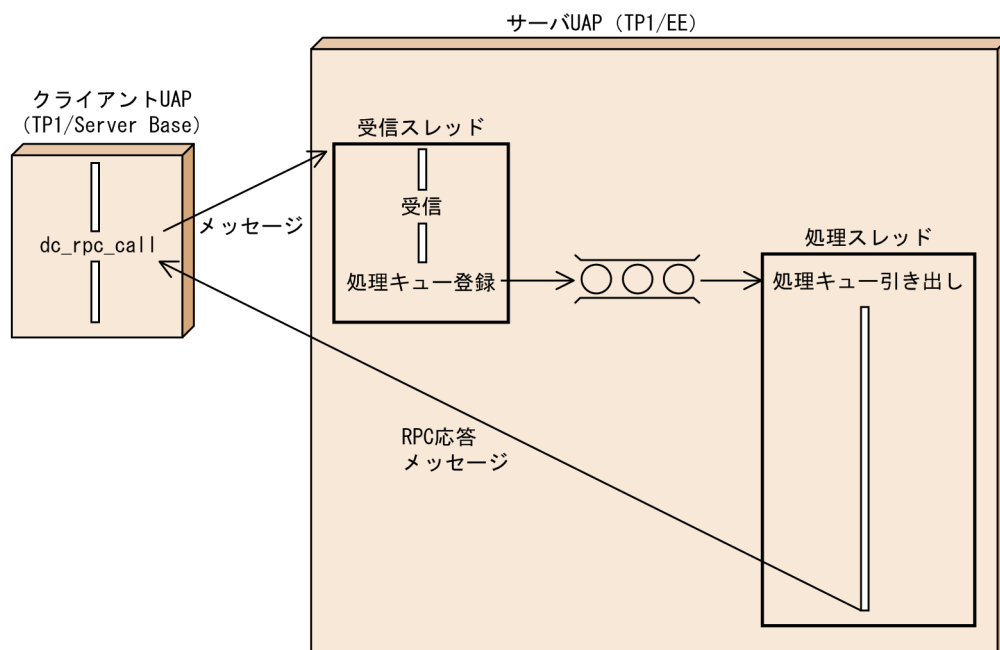
クライアント UAP からのメッセージの受信には、ソケット API (TCP/IP ソケットインタフェース) が使用されます。

サービスは、受信したメッセージ中のサービス名を基に、処理キューに登録されます。処理キューに登録されたサービスは、処理スレッドによって引き出され、ユーザデータをサーバ UAP に引き渡します。

サーバ UAP の処理終了後、ソケット API を使用して RPC 応答メッセージをクライアント UAP に送信します。

TP1/EE が同期応答型 RPC でメッセージを受信したときの処理の流れを次の図に示します。

図 2-5 同期応答型 RPC でメッセージを受信したときの処理の流れ



(a) ノード間負荷バランス機能を使用した場合の RPC メッセージ受信

ノード間負荷バランス機能とは、他ノードにある同じサービスグループ名のサーバ UAP 間で、サービス処理の負荷を分散させる機能です。ノード間負荷バランス機能を使用して RPC メッセージを受信した場合、自プロセスの負荷が高いとき、または、スケジューラダイレクト機能を使用した RPC メッセージのときに、他ノードにある同じサービスグループ名のサーバ UAP の中から負荷の低いプロセスを選択し、メッセージを転送します。他ノードにある同じサービスグループ名のサーバ UAP の中に、負荷の低いプロセスがない場合は、自プロセスで処理します。

ノード間負荷バランス機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

(b) 他サービスグループ名を指定した場合の RPC メッセージ受信

RPC メッセージの受信時、クライアント UAP で指定したサービスグループ名と TP1/EE のサービスグループ名が一致しない場合、ネーム情報から該当するサービスグループを選択し、メッセージを転送します。

(2) TP1/EE がクライアント UAP の場合の同期応答型 RPC

TP1/EE がクライアント UAP (SPP) の場合の同期応答型 RPC の送信形態は、クライアント UAP からの要求を受けて即時にメッセージ送信を行う即時送信 (TP1/Server Base の `dc_rpc_call` 関数または `dc_rpc_call_to` 関数と同じ形態) です。

クライアント UAP として使用する TP1/EE が要求できるサーバ UAP は、キュー受信型サーバと、ソケット受信型サーバのどちらでもかまいません。

ネームサービスを使用する場合は、RPC 関連定義の `rpc_destination_mode` オペランドでメッセージ送信先の取得方法を指定します。また、メッセージ送信先の決定時に、通信障害によって送信先のサーバ UAP を起動できなかった場合、または、送信先のサーバ UAP が未起動などの旨の応答メッセージが返ってきた場合、別のあて先を取得するためにリトライ処理をします。ただし、ネームサービスからネーム情報を取得する場合は、同じサービスグループの同じあて先に対してリトライ処理をする場合もあります。

ネームサービスを使用しない場合は、サービスグループ情報関連定義の `eesvgdef` 定義コマンドで指定したあて先にメッセージを送信します。送信時に障害が発生しても、リトライ処理はしません。

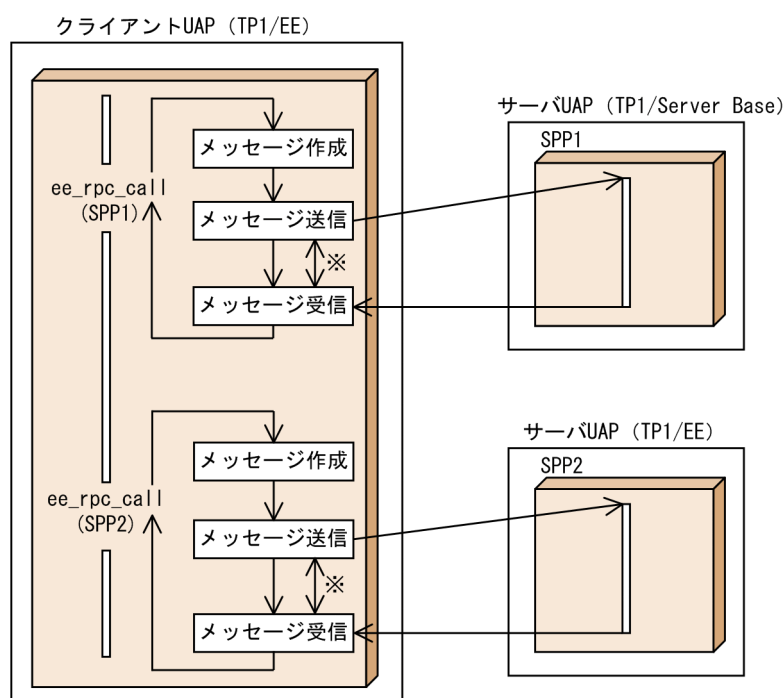
ネームサービスの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

(a) 即時送信

即時送信は、クライアント UAP からの要求で指定されたサーバ UAP に同期応答型 RPC でメッセージを送信します。クライアント UAP は、`ee_rpc_call` 関数でメッセージを作成し、サーバ UAP にメッセージを送信します。即時送信時に障害が発生した場合は、クライアント UAP（即時送信要求元）にエラーリターンします。

即時送信の流れを次の図に示します。

図 2-6 即時送信の流れ（同期応答型 RPC）



注※
RPC応答待ち時間監視

同期応答型 RPC の時間監視

`ee_rpc_call` 関数を呼び出してから、応答が返るまでの時間は、RPC 関連定義の `watch_time` オペランドに指定した監視時間で監視されています。

サーバ UAP の処理に時間が掛かって、指定した監視時間を過ぎた場合は、ee_rpc_call 関数はエラーリターンします。

サーバ UAP が異常終了すると、ee_rpc_call 関数はすぐにエラーリターンします。ただし、次に示す場合には、監視時間が過ぎてからエラーリターンする場合があります。

- サーバ UAP があるノードの TP1/EE または TP1/Server Base 全体が異常終了した場合
- サービス要求のデータがサーバ UAP に届く前に障害が発生した場合
- サーバ UAP の処理が終了してからクライアント UAP に結果が届く前に障害が発生した場合

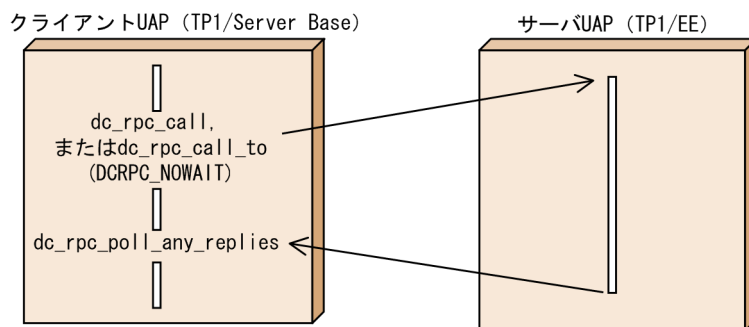
2.1.5 非同期応答型 RPC

サーバ UAP の処理結果をクライアント UAP に返す RPC です。クライアント UAP では、サーバ UAP の処理結果を待たないで処理を続行し、サーバ UAP の処理結果を非同期で受信します。

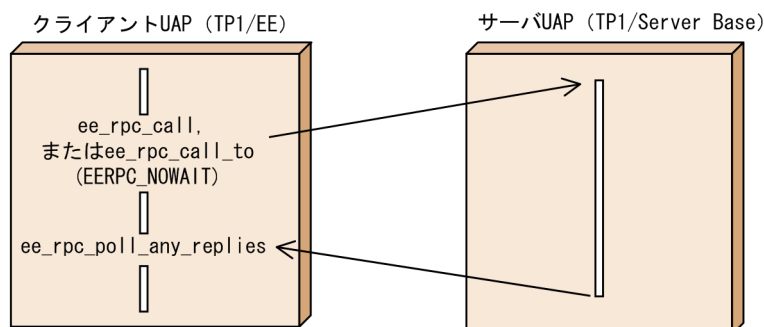
非同期応答型 RPC を次の図に示します。

図 2-7 非同期応答型 RPC

●TP1/EEがサーバの場合



●TP1/EEがクライアントの場合



非同期応答型 RPC は時間監視をします。監視時間内に応答が返らない場合はエラーリターンします。

(1) TP1/EE がサーバ UAP の場合の非同期応答型 RPC

サーバ UAP (TP1/EE) では、クライアント UAP (TP1/Server Base) からのメッセージを受信するための受信スレッドは、自ポート番号ごとに存在します。

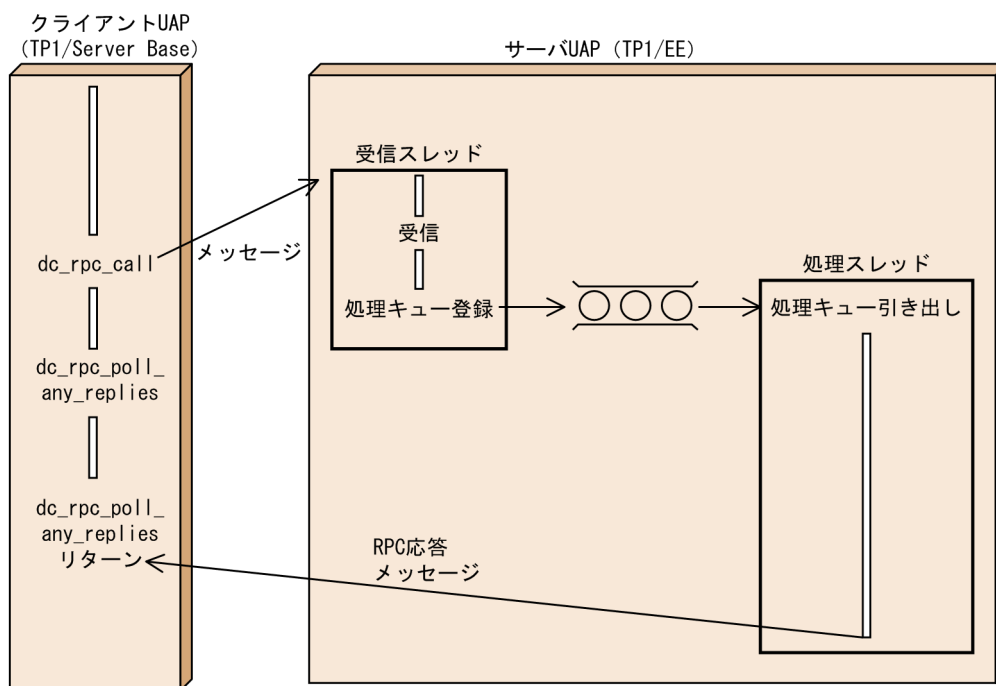
クライアント UAP からのメッセージの受信には、ソケット API (TCP/IP ソケットインタフェース) が使用されます。

サービスは、受信したメッセージ中のサービス名を基に処理キューに登録されます。処理キューに登録されたサービスは、処理スレッドによって引き出され、ユーザデータをサーバ UAP に引き渡します。

サーバ UAP の処理終了後、ソケット API を使用して RPC 応答メッセージをクライアントに送信します。

TP1/EE が非同期応答型 RPC でメッセージを受信したときの処理の流れを次の図に示します。

図 2-8 非同期応答型 RPC でメッセージを受信したときの処理の流れ



(a) ノード間負荷バランス機能を使用した場合の RPC メッセージ受信

ノード間負荷バランス機能とは、他ノードにある同じサービスグループ名のサーバ UAP 間で、サービス処理の負荷を分散させる機能です。ノード間負荷バランス機能を使用して RPC メッセージを受信した場合、自プロセスの負荷が高いとき、または、スケジューラダイレクト機能を使用した RPC メッセージのときに、他ノードにある同じサービスグループ名のサーバ UAP の中から負荷の低いプロセスを選択し、メッセージを転送します。他ノードにある同じサービスグループ名のサーバ UAP の中に、負荷の低いプロセスがない場合は、自プロセスで処理します。

ノード間負荷バランス機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

(b) 他サービスグループ名を指定した場合の RPC メッセージ受信

RPC メッセージの受信時、クライアント UAP で指定したサービスグループ名と TP1/EE のサービスグループ名が一致しない場合、ネーム情報から該当するサービスグループを選択し、メッセージを転送します。

(2) TP1/EE がクライアント UAP の場合の非同期応答型 RPC

TP1/EE がクライアント UAP (SPP) の場合の非同期応答型 RPC の送信形態は、クライアント UAP からの要求を受けて即時にメッセージ送信を行う即時送信 (TP1/Server Base の `dc_rpc_call` 関数または `dc_rpc_call_to` 関数と同じ形態) です。

クライアント UAP として使用する TP1/EE が要求できるサーバ UAP は、キュー受信型サーバと、ソケット受信型サーバのどちらでもかまいません。

ネームサービスを使用する場合は、RPC 関連定義の `rpc_destination_mode` オペランドでメッセージ送信先の取得方法を指定します。また、メッセージ送信先の決定時に、通信障害によって送信先のサーバ UAP を起動できなかった場合、または、送信先のサーバ UAP が未起動などの旨の応答メッセージが返ってきた場合、別のあて先を取得するためにリトライ処理をします。ただし、ネームサービスからネーム情報を取得する場合は、同じサービスグループの同じあて先に対してリトライ処理をする場合もあります。

ネームサービスを使用しない場合は、サービスグループ情報関連定義の `eesvgdef` 定義コマンドで指定したあて先にメッセージを送信します。送信時に障害が発生しても、リトライ処理はしません。

ネームサービスの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

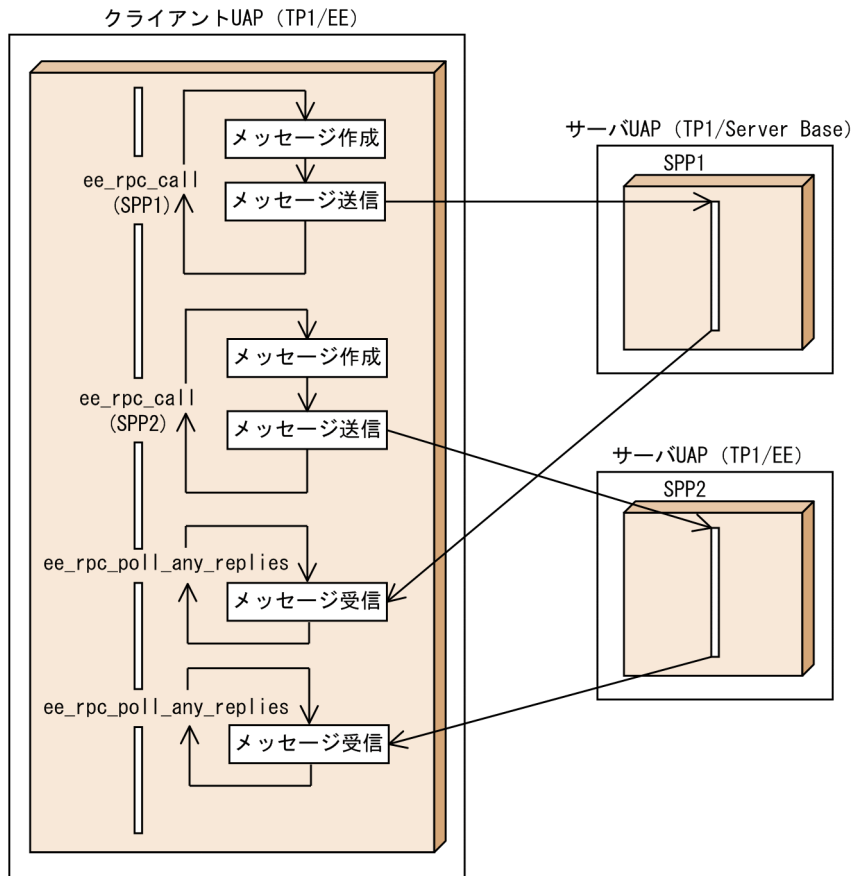
(a) 即時送信

即時送信は、クライアント UAP からの要求で指定されたサーバ UAP に非同期応答型 RPC でメッセージを送信します。クライアント UAP は、`ee_rpc_call` 関数でメッセージを作成し、サーバ UAP にメッセージを送信します。即時送信時に障害が発生した場合は、クライアント UAP (即時送信要求元) にエラーリターンします。

クライアント UAP が `ee_rpc_poll_any_replies` 関数を発行することによって、RPC 応答メッセージを受信します。

即時送信の流れを次の図に示します。

図 2-9 即時送信の流れ（非同期応答型 RPC）



非同期応答型 RPC の時間監視

非同期応答型 RPC の場合、`ee_rpc_poll_any_replies` 関数で時間監視します。RPC 関連定義の `watch_time` オペランドに指定した監視時間、または `ee_rpc_set_watch_time` 関数で指定した監視時間は無視されます。

(3) エラーが発生した非同期応答型 RPC 要求の記述子の取得

エラーが発生した非同期応答型 RPC 要求に対応する記述子は、`ee_rpc_get_error_descriptor` 関数で取得します。非同期応答の記述子を取得できるのは、サーバ UAP 側でエラーが発生した場合だけです。`ee_rpc_get_error_descriptor` 関数の呼び出し側でエラーが発生した場合は、非同期応答の記述子を取得できません。

2.1.6 非応答型 RPC

サーバ UAP の処理結果をクライアント UAP に返さない RPC です。クライアント UAP ではサーバ UAP の処理結果を受信できません。

非応答型 RPC は時間監視をしません。

非応答型 RPC の遠隔サービスを要求するサービス関数は、サービス要求後にリターンします。クライアント UAP は、サービス関数のリターン後に処理を続けます。サーバ UAP の処理結果は受信できません。

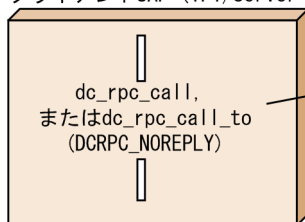
非応答型 RPC の遠隔サービスを要求するサービス関数は、サービス要求がサーバ UAP に受け付けられたことを確認しないでリターンします。そのため、通信障害などによってサービス要求が消失しても、クライアント UAP で認識することはできません。また、一つのクライアント UAP から同じサービスグループに複数の非応答型 RPC でサービスを要求した場合、サーバ UAP 側でこの要求順どおりにサービスを受け付けるとは限りません。

非応答型 RPC を次の図に示します。

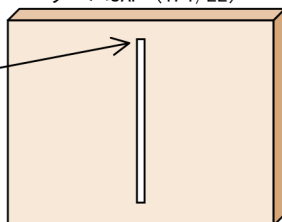
図 2-10 非応答型 RPC

●TP1/EEがサーバUAPの場合

クライアントUAP (TP1/Server Base)

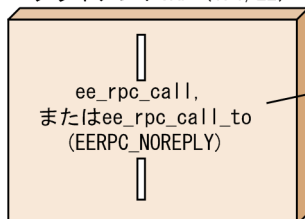


サーバUAP (TP1/EE)

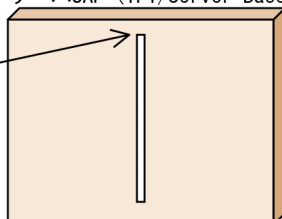


●TP1/EEがクライアントUAPの場合

クライアントUAP (TP1/EE)



サーバUAP (TP1/Server Base)



(1) TP1/EE がサーバ UAP の場合の非応答型 RPC

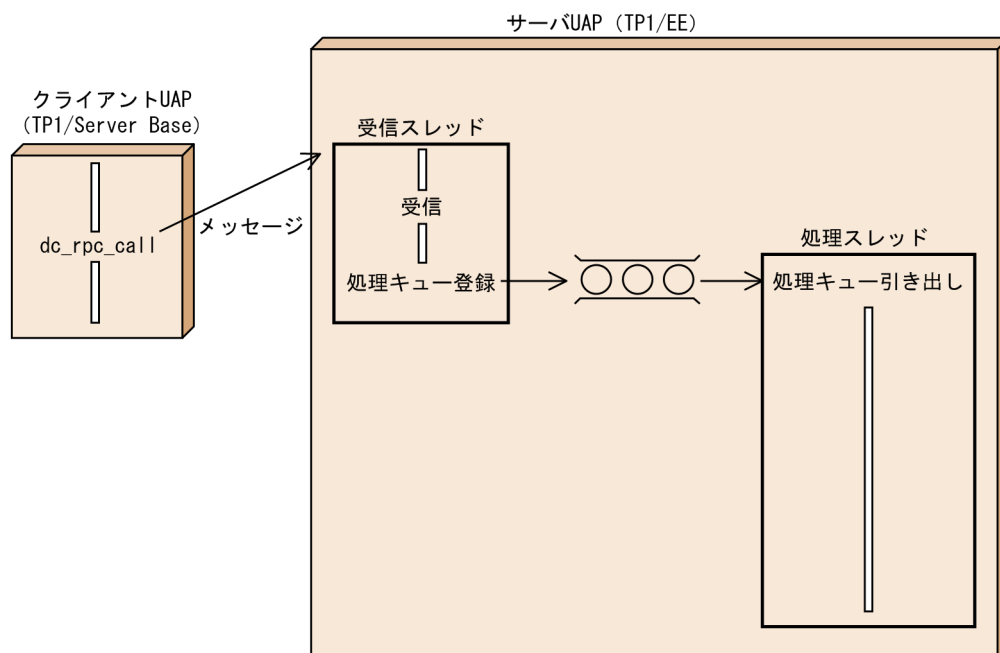
サーバ UAP (TP1/EE) では、クライアント UAP (SUP, SPP) からのメッセージを受信するための受信スレッドは、自ポート番号ごとに存在します。

クライアント UAP からの要求メッセージの受信には、ソケット API (TCP/IP ソケットインタフェース) が使用されます。

サービスは、受信したメッセージ中のサービス名を基に処理キューに登録されます。処理キューに登録されたサービスは、処理スレッドによって引き出され、ユーザデータをサーバ UAP に引き渡します。

TP1/EE が非応答型 RPC でメッセージを受信したときの処理の流れを次の図に示します。

図 2-11 非応答型 RPC でメッセージを受信したときの処理の流れ



(a) ノード間負荷バランス機能を使用した場合の RPC メッセージ受信

ノード間負荷バランス機能とは、他ノードにある同じサービスグループ名のサーバ UAP 間で、サービス処理の負荷を分散させる機能です。ノード間負荷バランス機能を使用して RPC メッセージを受信した場合、自プロセスの負荷が高いとき、または、スケジューラダイレクト機能を使用した RPC メッセージのときに、他ノードにある同じサービスグループ名のサーバ UAP の中から負荷の低いプロセスを選択し、メッセージを転送します。他ノードにある同じサービスグループ名のサーバ UAP の中に、負荷の低いプロセスがない場合は、自プロセスで処理します。

ノード間負荷バランス機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

(b) 他サービスグループ名を指定した場合の RPC メッセージ受信

RPC メッセージの受信時、クライアント UAP で指定したサービスグループ名と TP1/EE のサービスグループ名が一致しない場合、ネーム情報から該当するサービスグループを選択し、メッセージを転送します。

(2) TP1/EE がクライアント UAP の場合の非応答型 RPC

TP1/EE がクライアント UAP (SPP) の場合の非応答型 RPC の送信形態には、次の種類があります。

- 即時送信
クライアント UAP からの要求を受けて即時にメッセージ送信を行います。TP1/Server Base の `dc_rpc_call` 関数または `dc_rpc_call_to` 関数と同じ形態です。
- トランザクション同期型送信
クライアント UAP からの要求を受けても即時にメッセージ送信しないで、トランザクションの同期点で要求を有効とし、コミット時にメッセージ送信を行います。

トランザクション同期型送信で要求した RPC は、トランザクショナル RPC になりません。

トランザクション同期型送信は、さらに次の種別に分かれます。

- コミット送信
指定されたサーバ UAP に対して送信します。
- 自プロセス送信
クライアント UAP と同じノードのサーバ UAP に送信します。

各送信形態の概要を次の表に示します。

表 2-1 各送信形態の概要

送信形態	送信種別	送信先	サービス関数
即時送信	即時送信	他ノードのサーバ UAP	ee_rpc_call ee_rpc_call_to
トランザクション同期型送信※	コミット送信	他ノードのサーバ UAP	ee_rpc_cmtsend 【CBLEERPC('CMTSND ')】 ee_rpc_cmtsend_to 【CBLEERPC('CMTSNDTO')】
	自プロセス送信	自ノードのサーバ UAP	ee_rpc_cmtsend

注※

同期点処理がコミットの場合だけ送信します。ロールバックの場合は送信しません。

非応答型 RPC の場合、応答メッセージがないため、コネクションの障害などが発生し、メッセージが消失しても検知できません。ユーザによる送達確認が必要になります。

クライアント UAP として使用する TP1/EE が要求できるサーバ UAP は、キュー受信型サーバと、ソケット受信型サーバのどちらでもかまいません。

ネームサービスを使用する場合は、RPC 関連定義の rpc_destination_mode オペランドでメッセージ送信先の取得方法を指定します。また、メッセージ送信先の決定時に、通信障害によって送信先のサーバ UAP を起動できなかった場合、または、送信先のサーバ UAP が未起動などの旨の応答メッセージが返ってきた場合、別のあて先を取得するためにリトライ処理をします。ただし、ネームサービスからネーム情報を取得する場合は、同じサービスグループの同じあて先に対してリトライ処理をする場合もあります。

ネームサービスを使用しない場合は、サービスグループ情報関連定義の eesvgdef 定義コマンドで指定したあて先にメッセージを送信します。送信時に障害が発生しても、リトライ処理はしません。

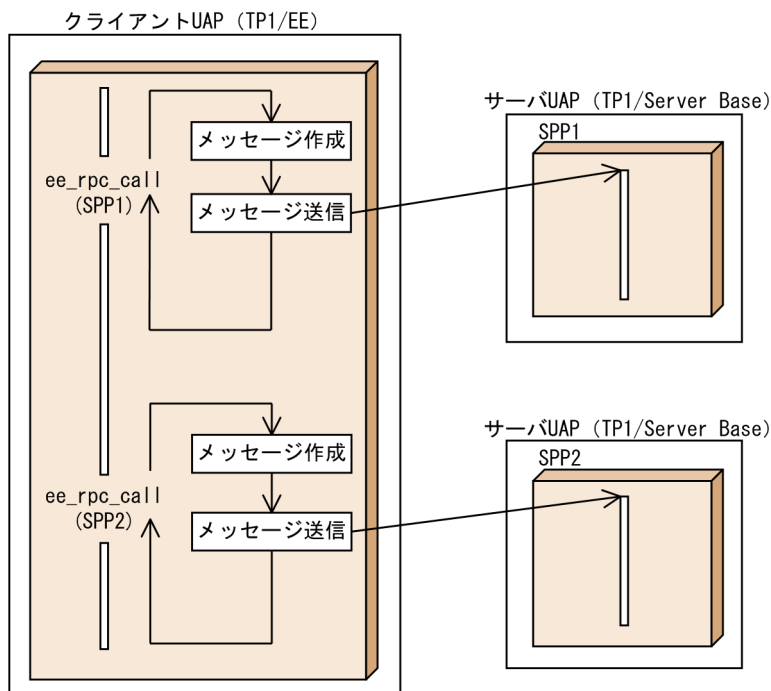
ネームサービスの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

(a) 即時送信

即時送信は、クライアント UAP からの要求で指定されたサーバ UAP に非応答型 RPC でメッセージを送信します。クライアント UAP は、`ee_rpc_call` 関数でメッセージを作成し、サーバ UAP にメッセージを送信します。サーバ UAP から応答は受け取りません。即時送信時に障害が発生した場合は、クライアント UAP（即時送信要求元）にエラーリターンします。

即時送信の流れを、次の図に示します。

図 2-12 即時送信の流れ（非応答型 RPC）



(b) トランザクション同期型送信

トランザクション同期型送信は、トランザクションのコミット時に、サーバ UAP へのサービス要求を有効にします。ロールバック時（ロールバックリターン含む）は、サービス要求は無効になります。

トランザクション同期型送信は、非グローバルトランザクション、非応答型 RPC の場合だけ有効です。

トランザクション同期型送信は、サービス要求が有効になるタイミングがコミット時ということ以外は、即時送信と変わりません。

■ コミット送信

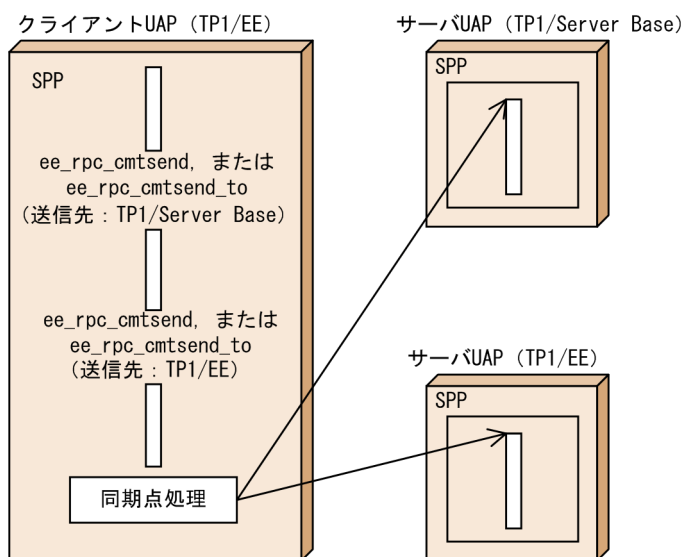
`ee_rpc_cmtsend` 関数または `ee_rpc_cmtsend_to` 関数でサーバ UAP を指定してサービス要求を行います。`ee_rpc_cmtsend` (`ee_rpc_cmtsend_to`) 関数の flags には `EENOF` を指定します。

`ee_rpc_cmtsend` (`ee_rpc_cmtsend_to`) 関数の呼び出し時は、実際の送信処理はしないで、サービス要求の受け付けだけをして、メッセージ作成後にリターンします。メッセージはメモリ（バッファ）上に退避します。退避していたメッセージは同期点処理（コミット）時に送信されます。同期点処理がロールバックの場合は、メッセージは送信されないで破棄されます。

トランザクション開始から同期点取得までで、複数回 ee_rpc_cmtsend (ee_rpc_cmtsend_to) 関数を呼び出して複数のメッセージを作成できます。メッセージごとに異なるサーバ UAP を指定できます。

コミット送信の流れを次の図に示します。

図 2-13 コミット送信



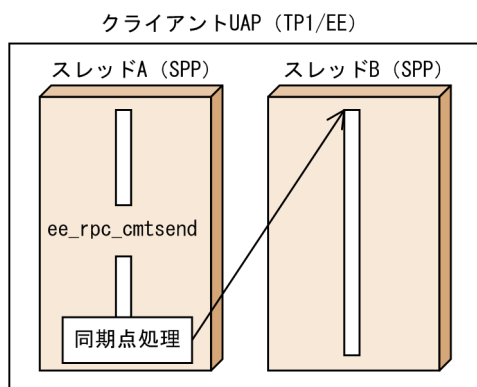
■ 自プロセス送信

クライアント UAP (TP1/EE) と同じノードのサーバ UAP に対してサービス要求をする場合、ee_rpc_cmtsend 関数に要求先サーバ UAP を指定して呼び出します。自プロセス送信は、コミット送信と同様に、トランザクションの同期点処理 (コミット) 時にサービス要求を有効にし、メッセージを送信します。このメッセージの送信は、クライアント UAP とは別スレッドで実行されます。

ネームサービスの使用時に自プロセス送信をする場合は、ee_rpc_cmtsend 関数の flags に EERPC_CMTSEND_MYPROC を指定してください。指定しないと、他プロセスにメッセージが送信されることがあります。

自プロセス送信の流れを次の図に示します。

図 2-14 自プロセス送信



2.1.7 通信先を指定したリモートプロシジャコール

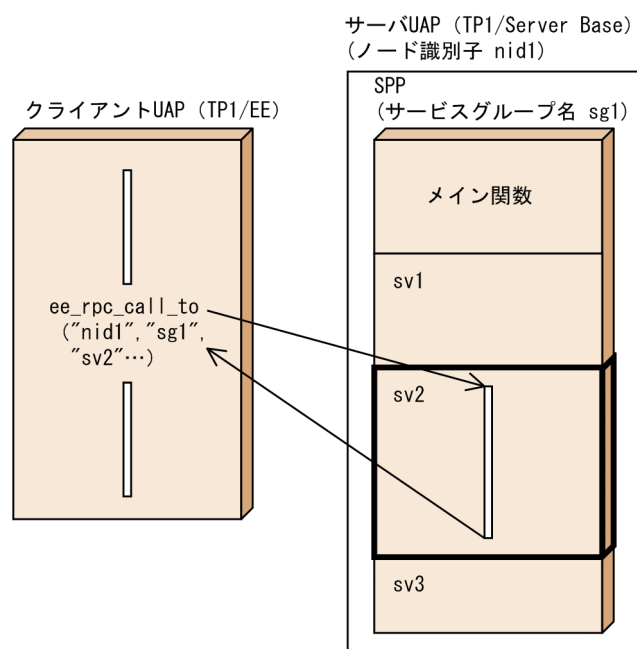
ee_rpc_call 関数または ee_rpc_cmtsend 関数を使ってサービスを要求する場合、要求するサービスがどこにあるかは、OpenTP1 のネームサービスで管理しているので、クライアント UAP で意識する必要はありません。しかし、サービス要求先ノードはネームサービスまたは定義によって管理するため、サービス要求先となるノードが複数存在した場合、サービス要求先となるノードを限定できません。

これに対し、ee_rpc_call_to 関数または ee_rpc_cmtsend_to 関数を使えば、特定のサービス要求先にサービスを要求できます。

サービス要求先を特定するには、ee_rpc_call_to 関数または ee_rpc_cmtsend_to 関数の引数に、ノード識別子、ホスト名、ホスト名とポート番号のどれかを指定します。

ノード識別子を指定して、サービス要求先を特定した ee_rpc_call_to 関数を使った通信の例を次の図に示します。

図 2-15 ee_rpc_call_to 関数を使った通信の例



(1) サービス要求先の特定方法

サービス要求先を特定するには、ee_rpc_call_to 関数または ee_rpc_cmtsend_to 関数の引数に次のどれかを指定します。

(a) ホスト名

/etc/hosts ファイル、または DNS など IP アドレスとマッピングできるホスト名を指定して、サービス要求先ノードを特定します。指定したホストにサービスグループまたはサービスが存在しない場合、RPC 要求はエラーとなります。

ネームサーバを指定した RPC の場合

- ポート番号には、サービス要求元 (ee_rpc_call_to 関数を呼び出した側) の name_port オペランドに指定した値が使用されます。サービス要求先の TP1/Server Base のシステム共通定義の name_port オペランドに指定した値と、サービス要求元の name_port オペランドに指定した値が同じであることが前提です。
- サービス要求先のホストは指定したネームサーバのホストだけであり、対象のネームサーバが管理するほかのホストは、サービス要求先になりません。
- RPC 関連定義の rpc_destination_mode オペランドの指定値に関係なく、ネーム検索によって送信先を決定します。
- ノード識別子を指定した場合、ノード識別子の指定は無効とします。

スケジュールサービスのポート (scd_port), または TP1/EE システムの受信用ポートを指定した RPC の場合

- ポート番号には、サービス要求元 (ee_rpc_call_to 関数を呼び出した側) のサービスグループ情報関連定義の mysvgdef 定義コマンドで指定した TP1/EE システムの受信用ポート番号が使用されます。サービス要求先のポート番号と、サービス要求元のサービスグループ情報関連定義の mysvgdef 定義コマンドで指定した TP1/EE システムの受信用ポート番号が同じであることが前提です。
- キュー受信型の RPC で送信します。ソケット受信型の TP1/Server Base のユーザサーバに対してこの機能による RPC 要求はできません。指定したホストに対する RPC 要求がエラーとなった場合、RPC 関連定義の rpc_retry オペランドの指定値を無視し、RPC 要求のリトライは行いません。
- ノード識別子を指定した場合、エラーリターンします。

(b) ノード識別子

TP1/Server Base のシステム共通定義の node_id オペランドに指定されているノード識別子を指定して、サービス要求先ノードを特定します。

指定したノード識別子に対応するサービス要求先の OpenTP1 ノードのホスト名がグローバルドメイン※内にあることが前提です。

該当の RPC 要求では、RPC 関連定義の rpc_destination_mode オペランドの指定値に関係なく、ネーム検索によって送信先を決定します。サービスグループ情報関連定義の eesvgdef 定義コマンドに指定した送信先は対象外とします。また、ホスト名とポート番号を指定した場合、ノード識別子の指定は無効とします。

サービス要求先が TP1/EE の場合も、ノード識別子にはサービス要求先の TP1/Server Base のノード識別子を指定します。

注※

ここでのグローバルドメインとは、次のノード名の集合を指します。

システム共通定義の name_domain_file_use オペランドに N を指定している場合

システム共通定義の all_node オペランド、all_node_ex オペランドで指定したノード名の集合です。

システム共通定義の name_domain_file_use オペランドに Y を指定している場合

ドメイン定義ファイルに指定したノード名の集合です。なお、ドメイン定義ファイルの格納場所は次のとおりです。

- all_node のドメイン定義ファイル
\$DCCONFPATH/dcnamnd ディレクトリ下
- all_node_ex のドメイン定義ファイル
\$DCCONFPATH/dcnamndex ディレクトリ下

(c) ホスト名とポート番号

/etc/hosts ファイル、または DNS など IP アドレスとマッピングできるホスト名、およびポート番号を指定することでサービス要求先ノードを特定します。

ネームサーバを指定した RPC の場合

- ポート番号には、サービス要求先の TP1/Server Base のシステム共通定義の name_port オペランドに指定したネームサービスのポート番号を指定します。
- サービス要求先のホストは指定したネームサーバのホストだけであり、対象のネームサーバが管理するほかのホストは、サービス要求先になりません。
- RPC 関連定義の rpc_destination_mode オペランドの指定値に関係なく、ネーム検索によって送信先を決定します。
- ノード識別子を指定した場合、ノード識別子の指定は無効とします。

スケジュールサービスのポート (scd_port)、または TP1/EE システムの受信用ポートを指定した RPC の場合

- ポート番号には、サービス要求先が TP1/Server Base の場合は、サービス要求先の OpenTP1 システムのスケジュールサービス定義の scd_port オペランドで指定したポート番号を指定します。サービス要求先が TP1/EE の場合は、サービス要求先のサービスグループ情報関連定義の mysvgdef 定義コマンドで指定した TP1/EE システムの受信用ポート番号を指定します。
- キュー受信型の RPC で送信します。ソケット受信型の TP1/Server Base のユーザサーバに対してこの機能による RPC 要求はできません。指定したホストに対する RPC 要求がエラーとなった場合、RPC 関連定義の rpc_retry オペランドの指定値を無視し、RPC 要求のリトライは行いません。
- ノード識別子を指定した場合、エラーリターンします。

(2) 送信先の決定とリトライ処理

ネームサーバ、またはノード識別子を指定した RPC では、ネームサービスから送信先を取得して送信します。送信失敗時はリトライ処理を行います。ただし、送信先はノード識別子の一致するノードまたはネームサーバのノードであり、1 ノードに限定されます。このため、リトライ中にネーム情報に変化がない場合は、同一のノードに対して再送します。

スケジュールサービスのポート (scd_port), または TP1/EE システムの受信用ポートを指定した RPC では、関数の引数に指定された送信先に対して送信します。送信失敗時はリトライ処理を行いません。

(3) 関数の指定値の関係

ee_rpc_call_to 関数または ee_rpc_cmtsend_to 関数の指定値の関係を次の表に示します。

表 2-2 ee_rpc_call_to 関数または ee_rpc_cmtsend_to 関数の指定値の関係

項番	関数の指定値				指定の推奨	処理内容
	ホスト名	ノード識別子	ポート番号	サービス要求先※		
1	—	—	—	EENOFLLAGS	△	ee_rpc_call 関数または ee_rpc_cmtsend 関数と同じです。
2	—	—	—	EERPC_NAMPORT	×	関数がエラーリターンします。
3	—	—	—	EERPC_SCDPORT	×	
4	—	—	○	EENOFLLAGS	×	
5	—	—	○	EERPC_NAMPORT	×	
6	—	—	○	EERPC_SCDPORT	×	
7	○	—	—	EENOFLLAGS	◎	ネームサーバを指定した RPC になります。 サービス要求先のポート番号には、サービス要求元のシステム共通定義の name_port オペランドの指定値を仮定します。
8	○	—	—	EERPC_NAMPORT	×	関数がエラーリターンします。
9	○	—	—	EERPC_SCDPORT	◎	スケジュールサービスのポート、または TP1/EE システムの受信用ポートを指定した RPC になります。 サービス要求先のポート番号には、サービス要求元の自サービスグループ情報関連定義の mysvgdef 定義コマンドで指定した受信ポート番号を仮定します。
10	○	—	○	EENOFLLAGS	×	関数がエラーリターンします。
11	○	—	○	EERPC_NAMPORT	◎	ネームサーバを指定した RPC になります。
12	○	—	○	EERPC_SCDPORT	◎	スケジュールサービスのポート、または TP1/EE システムの受信用ポートを指定した RPC になります。
13	—	○	—	EENOFLLAGS	◎	ノード識別子を指定した RPC になります。

項番	関数の指定値				指定の推奨	処理内容
	ホスト名	ノード識別子	ポート番号	サービス要求先※		
14	—	○	—	EERPC_NAMPORT	×	関数がエラーリターンします。
15	—	○	—	EERPC_SCDPORT	×	
16	—	○	○	EENOFLLAGS	×	
17	—	○	○	EERPC_NAMPORT	×	関数がエラーリターンします。 ポート番号の指定は無視します（項番 14 と同じ）。
18	—	○	○	EERPC_SCDPORT	×	関数がエラーリターンします。
19	○	○	—	EENOFLLAGS	△	ネームサーバを指定した RPC になります。 ノード識別子の指定は無視します（項番 7 と同じ）。
20	○	○	—	EERPC_NAMPORT	×	関数がエラーリターンします。
21	○	○	—	EERPC_SCDPORT	△	スケジュールサービスのポート，または TP1/EE システムの受信用ポートを指定した RPC になります。 ノード識別子の指定は無視します（項番 9 と同じ）。
22	○	○	○	EENOFLLAGS	×	関数がエラーリターンします。
23	○	○	○	EERPC_NAMPORT	△	ネームサーバを指定した RPC になります。 ノード識別子の指定は無視します（項番 11 と同じ）。
24	○	○	○	EERPC_SCDPORT	△	スケジュールサービスのポート，または TP1/EE システムの受信用ポートを指定した RPC になります。 ノード識別子の指定は無視します（項番 12 と同じ）。

(凡例)

○：ノード識別子，ホスト名，またはポート番号値を指定します。

—：ノード識別子，ホスト名，またはポート番号値を指定しません。

◎：指定を推奨します。

△：指定できますが，一部の指定値を無視した動作，または通信先を指定しない動作となるため，指定を推奨しません。

×：指定できません。

注※

サービス要求先の指定値「EERPC_NAMPORT」，「EERPC_SCDPORT」，および「EENOFLLAGS」は，C 言語の値です。COBOL 言語では，「VALUE 'N」，「VALUE 'S」，および「VALUE 'D」を指定します。詳細については，各関数の説明を参照してください。

2.1.8 リモートプロシジャコールでメッセージを送受信できない場合

リモートプロシジャコールでメッセージを送受信できない場合について、状況別に説明します。

(1) TP1/EE がサーバ UAP の場合で、メッセージを受信できないとき

次の場合はメッセージ受信処理を実行できません。

- サービス名が TP1/EE の処理キューに定義されていない場合（同期応答型 RPC，非同期応答型 RPC の場合）

メッセージ受信時，クライアント UAP で指定したサービス名がサーバ UAP（TP1/EE）の処理キューに定義されていない場合，処理キュー登録要求はエラーリターンします。このとき，クライアント UAP に RPC エラー応答，およびサーバ UAP にエラー通知を行うため，RPC エラー応答トランザクション，およびエラートランザクション（ERRTRN1）を起動します。ユーザサービス関連定義の `errtrn1` オペランドの指定がない場合は，エラーメッセージを出力し，メッセージを破棄します。

RPC エラー応答トランザクションは，クライアント UAP に対して RPC エラー応答メッセージを送信します。

- サービス名が TP1/EE の処理キューに定義されていない場合（非応答型 RPC の場合）

メッセージ受信時，クライアント UAP で指定したサービス名がサーバ UAP（TP1/EE）の処理キューに定義されていない場合，処理キュー登録要求はエラーリターンします。このとき，サーバ UAP にエラー通知を行うため，エラートランザクション処理キューへ登録要求をして，エラートランザクション（ERRTRN1）を起動します。ユーザサービス関連定義の `errtrn1` オペランドの指定がない場合は，エラーメッセージを出力し，メッセージを破棄します。

- クライアント UAP で指定したサービスグループのネーム情報を取得できない場合（同期応答型 RPC，非同期応答型 RPC の場合）

メッセージ受信時，クライアント UAP で指定したサービスグループのネーム情報を取得できない場合は，エラーリターンします。このとき，クライアント UAP に RPC エラー応答，およびユーザにエラー通知を行うため，RPC エラー応答トランザクションを起動します。RPC エラー応答トランザクションは，クライアント UAP に対して RPC エラー応答メッセージを送信します。

- クライアント UAP で指定したサービスグループのネーム情報を取得できない場合（非応答型 RPC の場合）

メッセージ受信時，クライアント UAP で指定したサービスグループのネーム情報を取得できない場合は，エラーメッセージを出力し，メッセージを破棄します。

- 制御メッセージを受信した場合

クライアント UAP からソケットの一時終了やソケット終了を要求するメッセージを受信した場合，エラーメッセージを出力し，該当するコネクションを解放します。

- 不正メッセージを受信した場合

クライアント UAP から受信したメッセージのサイズが不正であったり，RPC ヘッダに未サポート情報が設定されていたりした場合は，次の処理をします。

1. エラーメッセージログを出力する。

2. サイズ不正など、後続処理に影響がある障害が発生している場合は、該当コネクションを解放する。

3. RPC エラー応答が可能であれば、RPC エラー応答メッセージを送信する。

3.の処理は、非応答型 RPC の場合は該当しません。

- リソース不足を検知した場合

クライアント UAP からメッセージ受信時、受信バッファ不足や処理キュー不足などのリソース不足によってメッセージ受信処理を実行できない場合は、次の処理をします。

1. エラーメッセージを出力する。

2. 該当コネクションを解放する。

3. RPC エラー応答が可能であれば、RPC エラー応答メッセージを送信する。

3.の処理は、非応答型 RPC の場合は該当しません。

- 業務処理中にスレッドダウンした場合（同期応答型 RPC、非同期応答型 RPC の場合）

業務処理中にスレッドダウンした場合、次の処理をします。

1. モニタスレッドが回復トランザクションおよびエラートランザクション（ERRTRN3）を登録し、処理スレッドを再起動する。

2. 再起動された処理スレッドは、エラートランザクション（ERRTRN3）を起動する。

RPC 応答メッセージは回復トランザクションを引き出した回復スレッドが送信します。

ユーザサービス関連定義の errtrn3 オペランドの指定がない場合は、エラーメッセージを出力し、メッセージを破棄します。

(2) TP1/EE がクライアント UAP の場合で、メッセージを送信できないとき (非応答型 RPC の場合)

送信エラーになる場合を次に示します。

- コミット送信エラーになる場合

コミット送信でメッセージ送信処理中に障害が発生し、送信エラーになった場合は、プロセス関連定義の rpc_cmtsend_retry オペランドの指定に従い、エラートランザクション（ERRTRNS）起動、またはリトライ送信を行います。リトライ送信は、別スレッド（送信専用スレッド）で送信要求を行うため、送信順序は保証しません。また、リトライ送信がエラーになった場合は無条件でエラートランザクション（ERRTRNS）を起動します。ユーザサービス関連定義の errtrns オペランドの指定がない場合は、エラーメッセージを出力し、メッセージを破棄します。

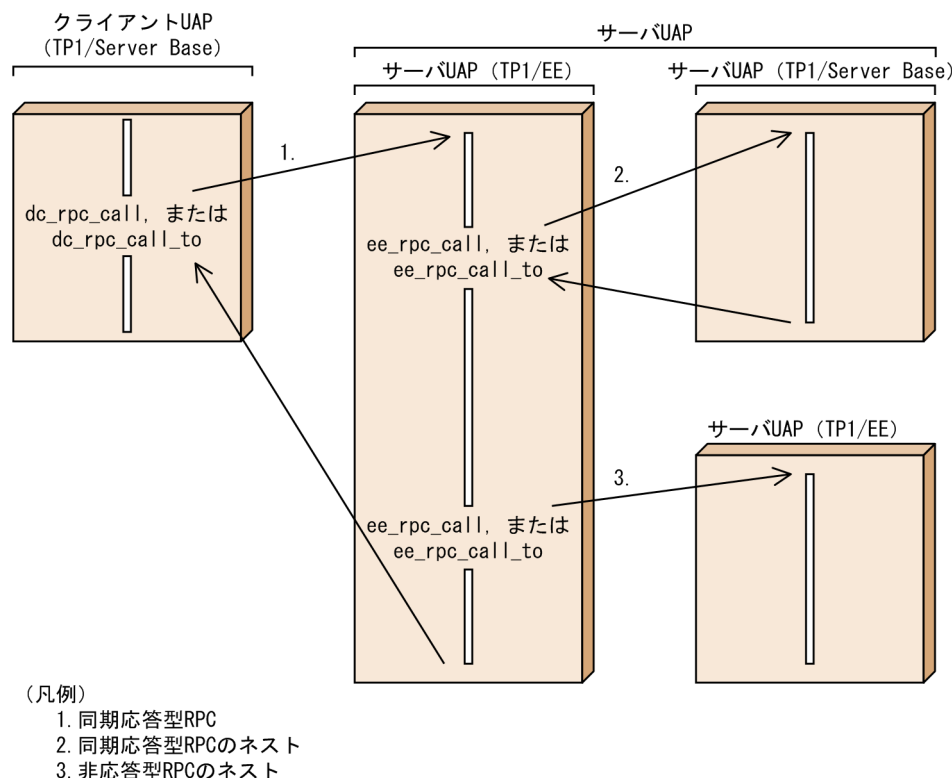
- サーバ UAP の正常リターン後に TP1/EE がロールバックした場合

ee_rpc_cmtsend 関数が正常に終了したあと、クライアント UAP（TP1/EE）がコミットするまでに、障害が発生し、ロールバックした場合、エラー要因に応じてエラートランザクション（ERRTRN3、ERRTRNR）を起動します。メッセージは破棄され、送信できません。

2.1.9 サービスのネスト

クライアントUAPにサービス要求されたサーバUAPから、さらにサーバUAPを呼び出すことができます。このようなサーバUAPのネストによって、サービス処理を分散化・階層化できます。RPCのネスト例を次の図に示します。

図 2-16 RPC のネスト例



2.1.10 サービス要求の応答待ち時間の参照と更新

クライアントUAP (TP1/EE) の処理の中で、サービス要求の応答待ち時間を一時的に変更できます。現在の応答待ち時間の設定を参照するときは `ee_rpc_get_watch_time` 関数【`CBLEERPC('GETWATCH')`】を、変更するときは `ee_rpc_set_watch_time` 関数【`CBLEERPC('SETWATCH')`】を使います。`ee_rpc_set_watch_time` 関数で変更した値は、そのサービス関数の終了時まで有効です。

`ee_rpc_get_watch_time` 関数は、`ee_rpc_set_watch_time` 関数で変更したサービス応答待ち時間をリターンします。応答待ち時間を変更していない場合は、RPC 関連定義の応答待ち時間 (`watch_time` オペランド) に指定した値がリターンされます。`ee_rpc_get_watch_time` 関数で得られる値は、`ee_rpc_call` 関数の応答待ち時間として有効です。

サービス要求の応答待ち時間を、`ee_rpc_set_watch_time` 関数を呼び出す前の値に戻すときは、事前に呼び出している `ee_rpc_get_watch_time` 関数で返された元の値を、`ee_rpc_set_watch_time` 関数で再度指定してください。

ee_rpc_set_watch_time 関数は、関数を呼び出したクライアント UAP のサービス要求に影響するだけで、RPC 関連定義の watch_time オペランドに指定した値は変更しません。ee_rpc_set_watch_time 関数で指定する値は、同じサービス内であとから呼び出す ee_rpc_call 関数にだけ影響します。

2.1.11 トランザクショナル RPC

次に示すどちらかの場合、RPC はトランザクショナル RPC となります。

- TP1/EE がクライアント UAP で、ee_rpc_call 関数の flags オプションに EERPC_TPNOTRAN を指定しない場合、かつ、TP1/Server Base がトランザクション属性のサーバ UAP の場合
- TP1/Server Base がトランザクション属性のクライアント UAP で、dc_rpc_call 関数の flags オプションに DCRPC_TPNOTRAN を指定しない場合

トランザクショナル RPC では、クライアント UAP とサーバ UAP が同一トランザクション（グローバルトランザクション）として動作します。

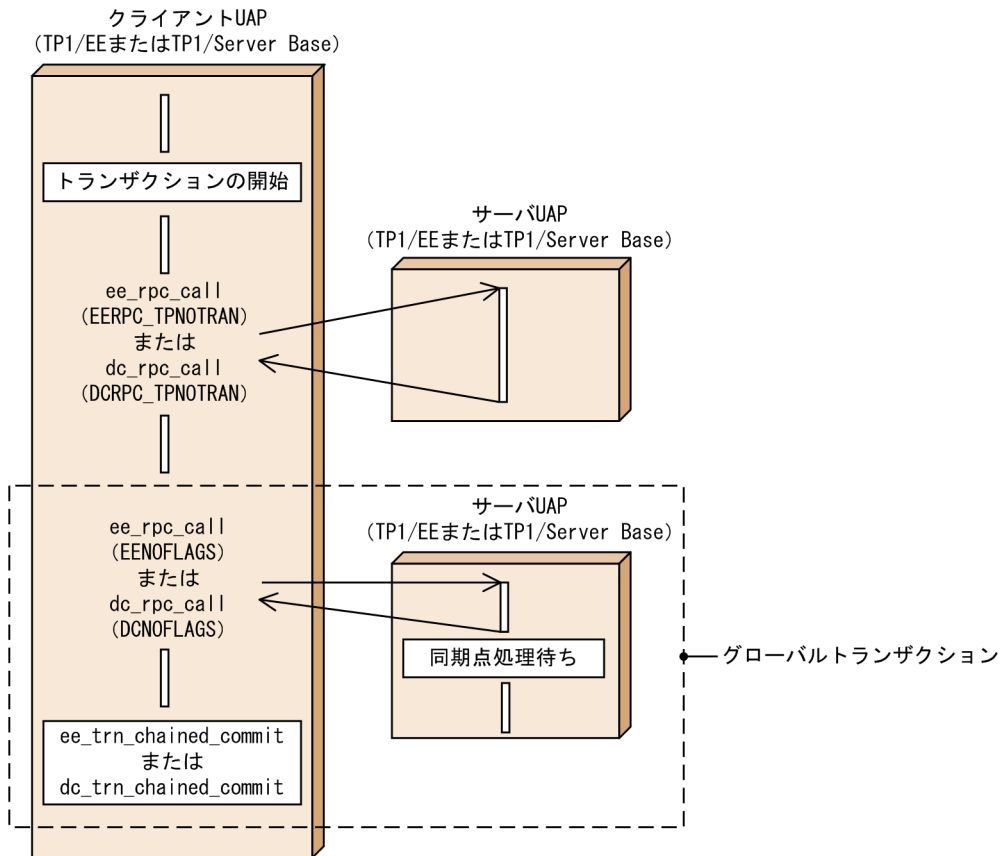
次に示すどちらかの場合、クライアント UAP とサーバ UAP が別トランザクションとして動作します。

- TP1/EE がクライアント UAP で、ee_rpc_call 関数の flags オプションに EERPC_TPNOTRAN を指定した場合
- TP1/Server Base がクライアント UAP で、dc_rpc_call 関数の flags オプションに DCRPC_TPNOTRAN を指定した場合

トランザクショナル RPC では、クライアント UAP のトランザクションの決着によって、サーバ UAP のトランザクションが決着します。

トランザクショナル RPC および非トランザクショナル RPC の流れを次の図に示します。

図 2-17 トランザクショナル RPC および非トランザクショナル RPC の流れ



注意事項

- トランザクション同期型送信で要求した RPC は、トランザクショナル RPC にはなりません。
- 同期点処理が終了するまで処理スレッドを占有するため、グローバルトランザクション内で時間の掛かる処理を行うと、処理スレッドの使用効率が低下します。
- 同期点処理が終了するまで対象サービスが処理中になるため、サービスの同時処理限界数の指定値によっては、グローバルトランザクション内から同一サービスに対してサービス要求をすると、サービス要求または同期点処理待ちのどちらかがタイムアウトします。
- オンラインテストを起動した状態の TP1/Server Base から TP1/EE に対してトランザクショナル RPC を実行することはできません。

2.1.12 トランザクションの処理から非トランザクショナル RPC の発行

サーバ UAP (TP1/EE) は、トランザクションとしても、グローバルトランザクションとしても動作できます。TP1/Server Base のクライアント UAP からサービス要求をするときは、次の関数の引数にトランザクションでない RPC (非トランザクショナル RPC) を指定する必要があります。

- `dc_rpc_call` 関数

- dc_rpc_call_to 関数
- ee_rpc_call 関数
- ee_rpc_call_to 関数

2.1.13 別トランザクションによる RPC 同期応答

別トランザクションによる RPC 同期応答は、応答型 RPC を受信したルートトランザクションブランチだけで使用できます。

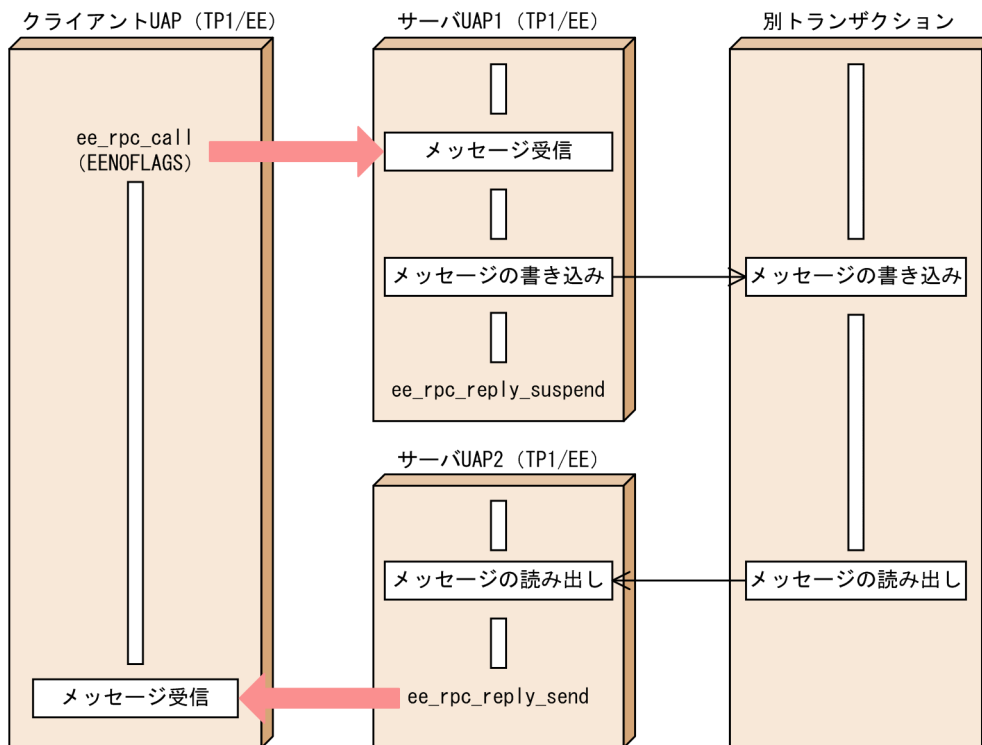
通常の応答型 RPC では、サービストランザクション起動時のサービス関数の引数として指定した応答データ格納領域 (out) に、サーバ UAP が応答データを格納し、サービス関数終了時に RPC 応答メッセージとしてクライアント UAP に送信します。

サーバ UAP で応答型 RPC を受信したトランザクション (サービス) は、ee_rpc_reply_suspend 関数【CBLEERPC('R-SPEND ')】を呼び出すと、サービス終了に伴う RPC 応答メッセージをクライアント UAP に送信しません。別の任意のトランザクション (サービス) から ee_rpc_reply_send 関数を呼び出して、クライアント UAP に RPC 応答メッセージを送信します。

ee_rpc_reply_suspend 関数を呼び出したあと、RPC 関連定義の rpc_reply_suspend_time オペランド、または ee_rpc_reply_suspend 関数の引数で指定した時間以内に ee_rpc_reply_send 関数【CBLEERPC('R-SEND ')】を呼び出さない場合、RPC エラー応答メッセージが自動的に送信されます。以降に ee_rpc_reply_send 関数を呼び出しても、ee_rpc_reply_send 関数はエラーになります。

別トランザクションによる RPC 同期応答処理の流れを次の図に示します。

図 2-18 別トランザクションによる RPC 同期応答処理の流れ



注意事項

- `ee_rpc_reply_suspend` 関数を呼び出している状態でスレッドダウンした場合、スレッドの回復処理時に RPC 応答メッセージは送信されません。そのため、ユーザの回復処理で RPC 応答メッセージを送信する必要があります。
- 応答型 RPC 以外では、`ee_rpc_reply_suspend` 関数を呼び出せません。応答型 RPC 以外で呼び出した場合、`ee_rpc_reply_suspend` 関数はエラーリターンします。
- RPC 応答メッセージ抑止限界時間の監視は 5 秒間隔で行われます。このため、RPC エラー応答メッセージの送信は、RPC 関連定義の `rpc_reply_suspend_time` オペランドの指定値より最大で 5 秒の遅延が発生する場合があります。

TP1/EE 再開始時の RPC 応答メッセージ抑止状態引き継ぎ

TP1/EE の再開始時に RPC 応答メッセージ抑止状態を引き継ぐかどうかは、RPC 関連定義の `rpc_reply_suspend_recover` オペランドで指定します。

`rpc_reply_suspend_recover` オペランドに Y を指定した場合、前回の RPC 応答メッセージ抑止状態を TP1/EE の再開始時に引き継ぎます。

`rpc_reply_suspend_recover` オペランドに N を指定した場合、前回の RPC 応答メッセージ抑止状態を TP1/EE の再開始時に引き継ぎません。

また、`rpc_reply_suspend_recover` オペランドに Y を指定している場合、`rpc_reply_suspend_autosend` オペランドに Y を指定したときは、TP1/EE の再開始時に RPC 応答メッセージ抑止状態を自動的に解除し、RPC エラー応答メッセージを送信します。

RPC 応答メッセージ抑止中の TP1/EE の終了

RPC 応答メッセージ抑止中に TP1/EE を正常終了させた場合、RPC 応答メッセージ抑止状態を自動的に解除し、RPC エラー応答メッセージを送信します。

2.1.14 TP1/Server Base と RPC 通信をする場合の設定

TP1/EE と TP1/Server Base との RPC 通信で設定する必要があるシステム定義または関数の概要について説明します。詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」の、TP1/Server Base と RPC 通信をする場合の設定に関する記述を参照してください。

なお、TP1/EE と TP1/Server Base との間でトランザクション連携をする場合は、RPC 関連定義の name_use オペランドに Y を指定してください。

(1) TP1/EE がサーバ UAP で、TP1/Server Base がクライアント UAP の場合

TP1/EE がサーバ UAP で、TP1/Server Base がクライアント UAP の場合、TP1/Server Base 側で次に示す定義および関数を設定する必要があります。

- ユーザサービス定義※
- ユーザサービスネットワーク定義※
- dc_rpc_call 関数
- dc_rpc_call_to 関数
- DCRPC_DIRECT_SCHEDULE 関数

注※

dc_rpc_call 関数を使用する場合に指定が必要です。

(2) TP1/Server Base がサーバ UAP で、TP1/EE がクライアント UAP の場合

TP1/Server Base がサーバ UAP で、TP1/EE がクライアント UAP の場合、TP1/EE 側で次に示す関数を設定する必要があります。

- ee_rpc_call 関数
- ee_rpc_call_to 関数

2.2 トランザクション制御

TP1/EE では、クライアント／サーバ形態の通信のトランザクション制御ができます。これによって、複数のプロセスにわたる UAP の処理を一つのトランザクションとして処理できます。TP1/EE は、XA インタフェースの関数を使用してトランザクション制御をします。

トランザクションマネージャが管理、実行するトランザクション種別については、「[付録 A トランザクション種別](#)」を参照してください。

2.2.1 クライアント／サーバ形態の通信のトランザクション

TP1/EE では、RPC を使用して複数の UAP プロセスにわたっている処理を、1 件のトランザクション処理にできます。このようなトランザクションをグローバルトランザクションといいます。このグローバルトランザクションによって、クライアント／サーバ形態のトランザクションができるようになります。

クライアント／サーバ形態の通信でトランザクションを制御するときは、トランザクションの開始および終了と同期点の取得を UAP で明示します。

(1) トランザクションの開始と終了

(a) トランザクションの開始

処理要求が発生した場合に、TP1/EE が、ルートトランザクション、またはトランザクションブランチとしてトランザクションを開始します。

(b) トランザクションの回復

トランザクション処理中に障害が発生した場合、ステータスファイル情報、またはテーブル状態を基にトランザクションを回復します。回復処理の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」のプロセスの回復、およびスレッドの回復の説明を参照してください。

(c) トランザクションの縮退

トランザクション開始時に、リソースマネージャに障害が発生した場合は、障害が発生したリソースマネージャにアクセスできないトランザクションとして縮退して開始します。縮退情報は、トランザクションインタフェースおよび ee_trn_getinfo 関数【CBLEETRN('GETINFO ')】で取得できます。

トランザクション処理中※に、リソースマネージャに障害が発生した場合も、障害が発生したリソースマネージャにアクセスできないトランザクションとして縮退して処理を続行します。縮退情報は、ee_trn_getinfo 関数で取得できます。

注※

トランザクションの処理には、再起動トランザクションの起動処理（ee_trn_chained_commit 関数【CBLEETRN('C-COMMIT')】または EETRN_KEEP を指定した ee_trn_chained_rollback 関数【CBLEETRN('C-ROLL ')】の呼び出し）も含まれます。

(d) トランザクションの終了

グローバルトランザクションは、コミットまたはロールバックのどちらかに決着して終了します。このとき、グローバルトランザクションとリンケージしたスレッドからの要求で実行したリソースマネージャの処理は、すべて実行される（コミット）か、またはすべて実行されない（ロールバック）かのどちらかです。

グローバルトランザクションの決着結果は、グローバルトランザクションのルートトランザクションブランチが完了する前に保持していた結果と一致します。グローバルトランザクションの決着結果と、グローバルトランザクションに属するトランザクションブランチのトランザクション決着結果との関係を次の表に示します。

表 2-3 トランザクション決着結果

グローバルトランザクションのトランザクション決着結果	各トランザクションブランチのトランザクション決着結果
COMMIT	グローバルトランザクションに属するすべてのトランザクションブランチがコミットできた状態。
ROLLBACK	グローバルトランザクションに属するトランザクションブランチの一つ以上がロールバックしたため、グローバルトランザクションがロールバックした状態。
HEURISTIC_HAZARD	障害のため、グローバルトランザクションに属するトランザクションブランチのうち、ヒューリスティックに完了したトランザクションブランチの決着結果が判明しない状態。
HEURISTIC_MIX	グローバルトランザクションに属するあるトランザクションブランチがコミットし、あるトランザクションブランチがロールバックした状態。ルートトランザクションブランチとトランザクションブランチの決着が矛盾している状態。

(2) 同期点の取得

グローバルトランザクションを構成するトランザクションブランチのすべてを、同期を取って同じ決着結果（コミットまたはロールバック）で終了させることを同期点の取得といいます。

(a) 同期点を取得する関数

同期点を取得するときは、次に示す関数を呼び出します。

- ee_trn_chained_commit 関数
- ee_trn_chained_rollback 関数

トランザクションとして実行中の UAP から要求されたサービスは、要求された時点ですでにトランザクションとして実行しています。

コミットの関数の呼び出し

トランザクションを UAP の判断でコミットとしたいときは、UAP からコミットを要求できます。

ロールバックの関数の呼び出し

トランザクションを UAP の判断でロールバックとしたいときは、UAP からロールバックを要求できます。関数を呼び出すときの指定で、該当するスレッドを強制終了させることもできます。

同期点を取得する処理でエラーが起こった場合

同期点を取得する関数の要求時、トランザクションの同期点を取得する処理でエラーが起こった場合は、エラーリターンします。

UAP が TP1/EE に制御を戻したあと、トランザクションの同期点を取得する処理でエラーが起こった場合は、エラートランザクション (ERRTRNR) を起動します。

同期点を取得する関数を呼び出さなかった場合

同期点を取得する関数を呼び出す前に UAP が TP1/EE に制御を戻した場合は、UAP の同期点の決着結果はコミットになります。同期点を取得する関数を呼び出す前に UAP が異常終了した場合は、UAP の同期点の決着結果はロールバックになります。

(b) リソースマネージャとのコネクションの解放

トランザクションの同期点処理が完了※するときに、スレッドにリンクされているすべてのリソースマネージャに対してコネクションの解放要求 (xa_close 関数) を指示できます。スレッドにリンクされているすべてのリソースマネージャに対してコネクションの解放を指示するには、ee_trn_rm_close_mark 関数【CBLEETRN('RMCLSMK ')】を呼び出します。同期点処理の完了時にコネクション確立済みのリソースマネージャが存在しない場合は、ee_trn_rm_close_mark 関数を呼び出してもコネクションの解放要求 (xa_close 関数) を指示しません。

注※

同期点処理が完了する時点は、次に示す関数または処理が終了した時点を示します。

- ee_trn_chained_commit 関数
- ee_trn_chained_rollback 関数
- UAP リターン後の同期点処理

リソースマネージャとのコネクションの再確立 (xa_open 関数) は、ee_trn_rm_close_mark 関数を呼び出したスレッドでの次のトランザクション開始時に TP1/EE が行います。

トランザクション関連定義の trnstring 定義コマンドで複数の Oracle に接続するように定義した場合で、SQL の実行時にエラーが発生したときは、ee_trn_rm_close_mark 関数を呼び出してリソースマネージャとのコネクションを解放してください。これは、SQL 実行時のエラーが発生したあとやトランザクションをロールバックしたあとに、Oracle の暗黙カーソルをクローズさせるためです。Oracle の暗黙カーソルがクローズされない場合に、Oracle に対する SQL の実行でエラーが発生したあとで別の Oracle に対して

SQL を実行すると、SQL エラーが発生します。コネクションの解放を指示して Oracle の暗黙カーソルをクローズすることで、以降のトランザクションでの SQL エラーを回避できます。

注意事項

トランザクションの同期点を取得する処理でエラーが起こった場合、コミット時に実行する次の関数の処理は無効となります。

- ee_rpc_cmtsend 関数 【CBLEERPC('CMTSND ')】
- ee_rpc_cmtsend_to 関数 【CBLEERPC('CMTSNDTO')】
- ee_tim_execap 関数 【CBLEETIM('EXECAP ')】 (acttype に EETIM_COMT 指定)
- ee_mch_cmtsend 関数 【CBLEEMCH('CMTSND ')】 (XTC を使用する場合)
- ee_mch_cmtsend_sync 関数 【CBLEEMCH('CMTSND SY')】 (XTC を使用する場合)
- ee_mcp_send 関数 【CBLEEMCP('SEND ')】 (MCP を使用する場合)

2.2.2 リモートプロシジャコールの形態と同期点の関係

トランザクションとして稼働しているクライアント UAP からトランザクショナル RPC で呼ばれたサーバ UAP はトランザクションとして稼働します。各トランザクションブランチは、一つのグローバルトランザクションとして同期を取れます。したがって、各サーバ UAP のプロセスは、処理終了後、ee_rpc_call 関数を呼び出した UAP にリターンしても、ルートトランザクションブランチに戻って同期点処理が完了するまでは、次のサービス要求を受け付けられません。また、サーバ UAP で確保している資源も解放されません。これは非同期応答型 RPC、非応答型 RPC の場合でも同様です。

このように、UAP の処理では RPC のトランザクション制御で、複数の UAP で同期が取れます。

RPC のトランザクション制御は、TP1/EE 同士、または、TP1/EE と TP1/Server Base との間で実行できます。詳細については、「[2.1.11 トランザクショナル RPC](#)」またはマニュアル「TP1/Server Base Enterprise Option 使用の手引」の TP1/Server Base とのトランザクション連携に関する説明を参照してください。

クライアント UAP の同期点処理が完了する前に、サーバ UAP のプロセスでほかのサービス要求を処理できる場合があります。これをトランザクションの最適化といいます。トランザクションの最適化については、「[2.2.4 トランザクションの最適化](#)」を参照してください。

(1) リモートプロシジャコールと同期点の関係

同期応答型 RPC での RPC 応答メッセージの送信タイミングは、サーバ UAP がルートトランザクションブランチかどうかで異なります。

- サーバ UAP がルートトランザクションブランチの場合

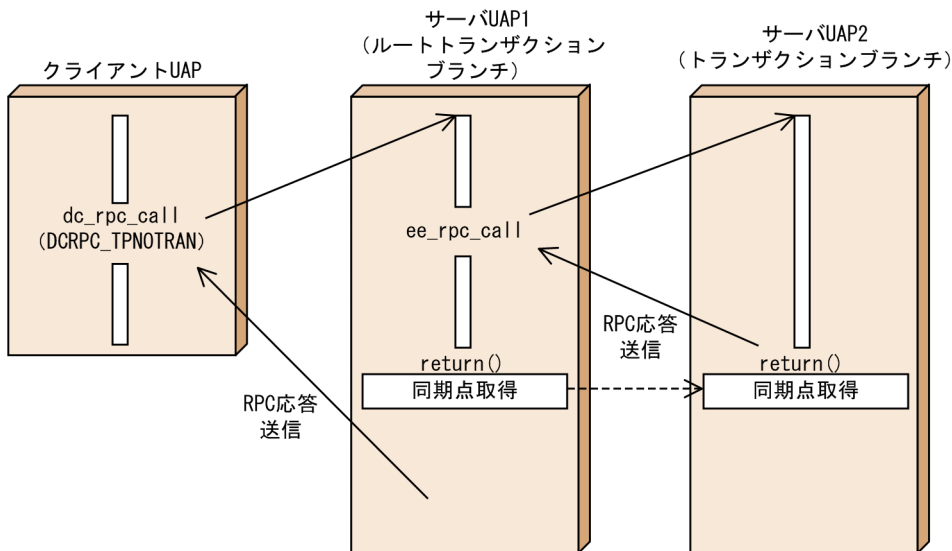
トランザクションの同期点処理が完了したあとに RPC 応答メッセージを送信します。

- サーバ UAP がルートトランザクションでない場合

UAP がリターンしたときに RPC 応答メッセージを送信します。

リモートプロシジャコールと同期点の関係を次の図に示します。

図 2-19 リモートプロシジャコールと同期点の関係



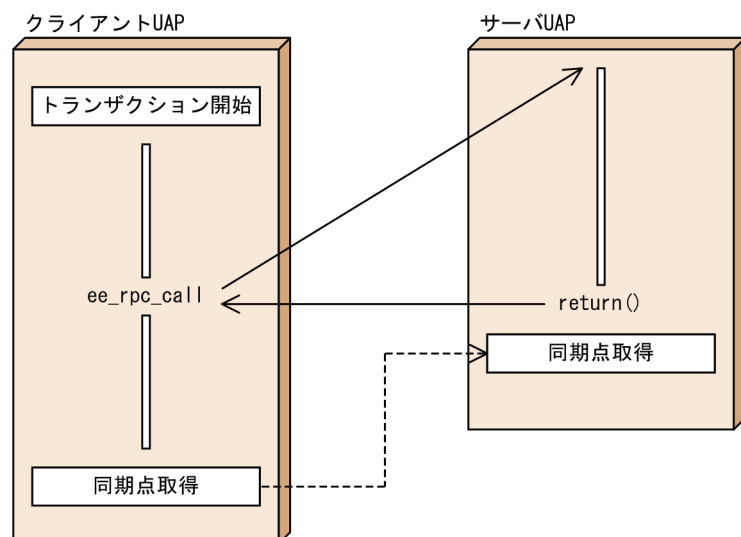
(2) 同期応答型 RPC と同期点の関係

同期応答型 RPC のトランザクション処理の場合は、ルートトランザクションブランチに処理結果が戻って、同期点取得処理を終えた時点でトランザクションの終了となります。

トランザクションを最適化する条件がそろっている場合、サーバ UAP のプロセスは処理が終了した時点で、次のサービス要求を受け付けられます。

同期応答型 RPC と同期点の関係を次の図に示します。

図 2-20 同期応答型 RPC と同期点の関係

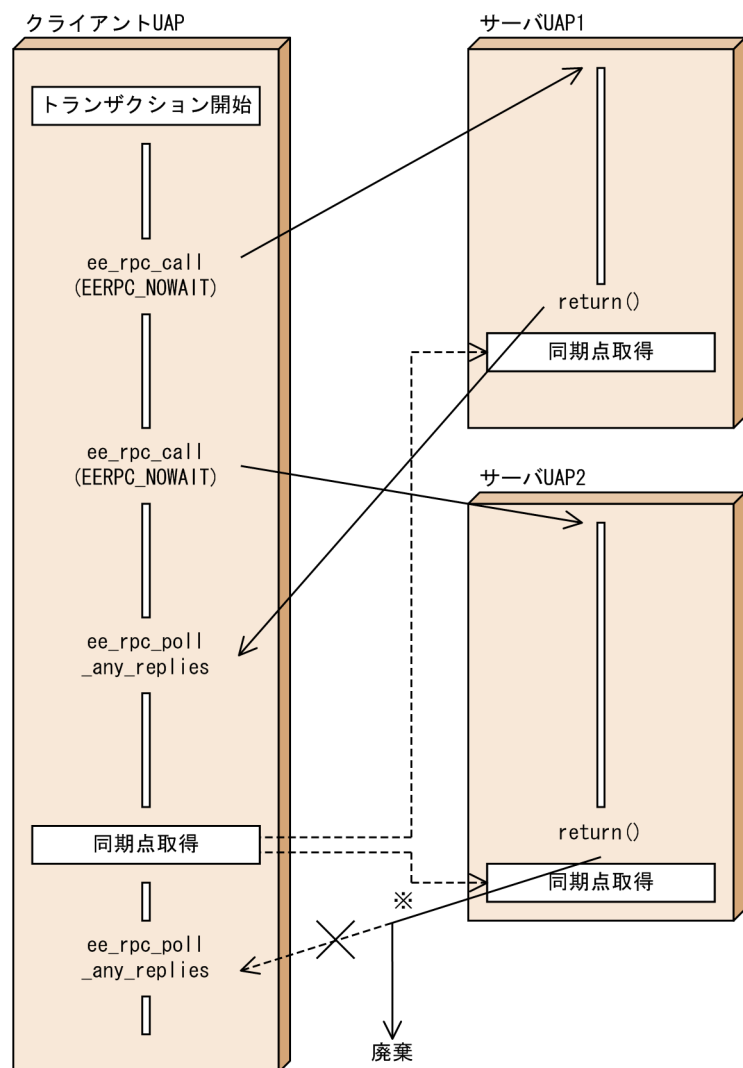


(3) 非同期応答型 RPC と同期点の関係

非同期応答型 RPC のトランザクション処理の場合は、クライアント UAP で同期点処理を終えた時点で RPC の処理を終了とします。同期点処理後にサーバ UAP から応答が返ってきても、ee_rpc_call 関数を呼び出した UAP では受信できません。

非同期応答型 RPC と同期点の関係を次の図に示します。

図 2-21 非同期応答型 RPC と同期点の関係



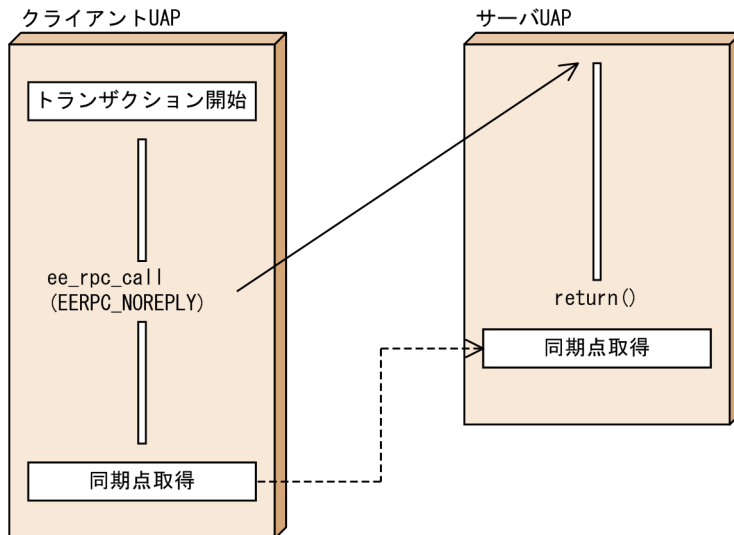
注※ クライアントUAPで同期点取得処理が完了したあとに `ee_rpc_poll _any_replies`関数を呼び出しても、応答は受信できません。非同期応答型RPCで要求したサービスの結果をすべて受信する場合は、`ee_rpc_call`関数を呼び出した数だけ、`ee_rpc_poll _any_replies`関数を、必ず同期点取得処理前に呼び出してください。また、応答の受信を拒否する場合は、`ee_rpc_discard_further_replies`関数を呼び出してください。

(4) 非応答型 RPC と同期点の関係

非応答型 RPC のトランザクション処理の場合は、クライアント UAP の同期点取得でサーバ UAP の処理終了を待って、そのあとで同期点取得処理をします。

非応答型 RPC と同期点の関係を次の図に示します。

図 2-22 非応答型 RPC と同期点の関係



(5) リモートプロシジャコールのエラーリターン値と同期点

ee_rpc_call 関数がエラーリターンしても，トランザクションの同期点がコミットとなる場合があります。

また，リターン値によっては，必ずトランザクションをロールバックさせなければならない場合があります。この場合は，ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数【CBLEETRN('ROLLMARK')】を呼び出してロールバックさせてください。

2.2.3 2相コミット

複数の資源を更新するとき，トランザクションは同期点で，2相コミットと呼ばれる方法で最終的に資源を更新するかどうか決定します。2相コミットとは，同期点処理を資源の更新準備処理（プリペア処理）と，資源の更新処理（コミット処理またはロールバック処理）の二段階に分ける方法です。同期点処理を二つの相に分けることで，複数の資源を矛盾なく更新できます。

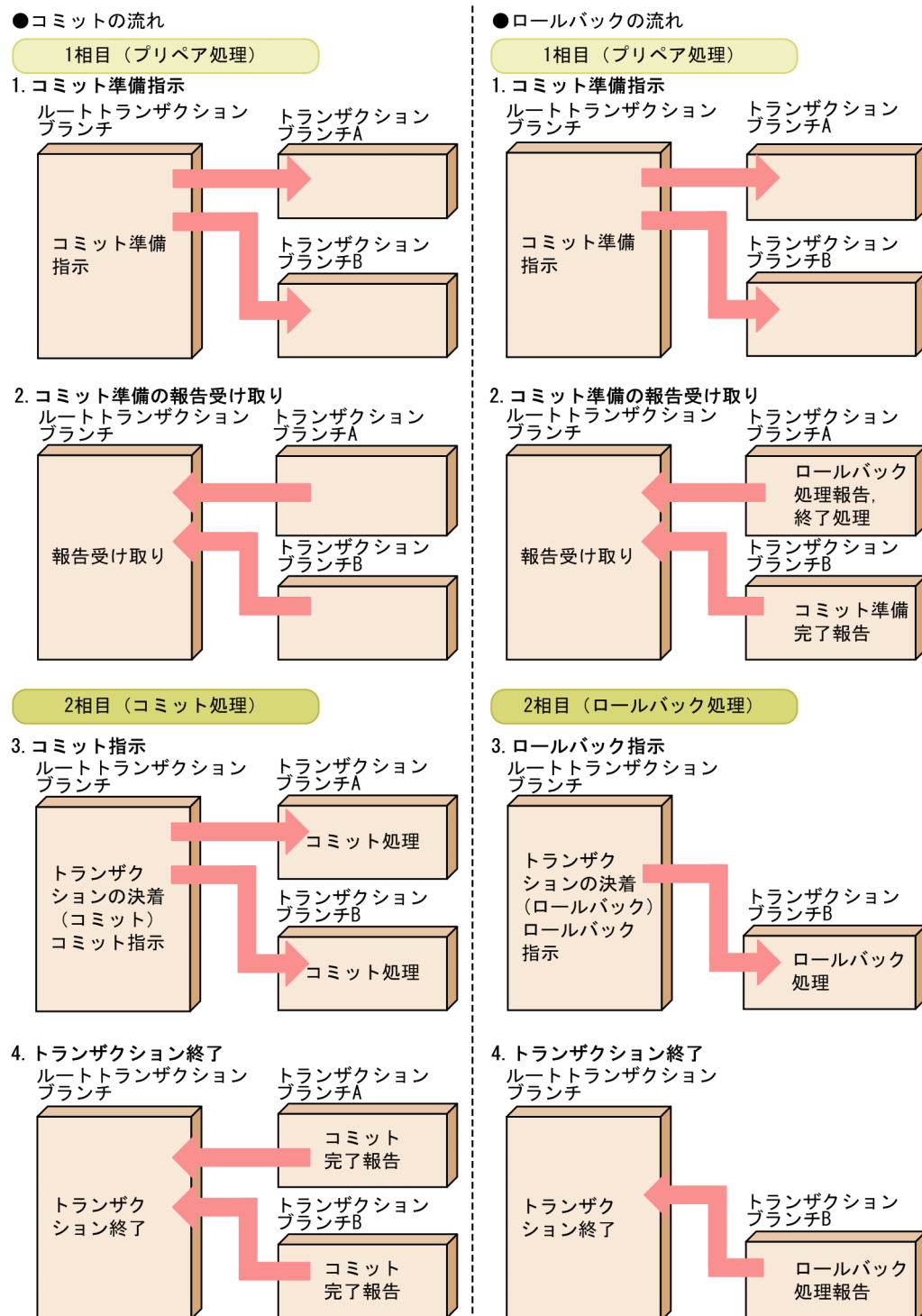
同期点の第1相で，ルートトランザクションブランチがグローバルトランザクション内のすべてのトランザクションブランチに，コミットの準備を要求します。トランザクションブランチは，コミットの準備が完了したか，ロールバックしたかを，ルートトランザクションブランチに報告します。

同期点の第2相で，グローバルトランザクションをロールバックするかコミットするか決定します。すべてのトランザクションブランチでコミットの準備が完了すると，ルートトランザクションブランチはグローバルトランザクション全体のコミットを決定します。トランザクションブランチのどれかがロールバックしたときには，ルートトランザクションブランチはグローバルトランザクション全体のロールバックを決定します。グローバルトランザクション内のトランザクションブランチはすべて，ルートトランザクションブランチでの決定に従います。

トランザクションの決着（コミット決定、またはロールバック決定）以降にシステムに障害が発生した場合に備えて、2相コミット処理の過程でトランザクションに関する履歴情報を、ステータスファイルに取得します。

2相コミットの流れを次の図に示します。

図 2-23 2相コミットの流れ



(1) ヒューリスティック決着

ルートトランザクションブランチがトランザクションブランチにコミットの準備を指示したあとで通信障害が発生して、ルートトランザクションブランチでの決着がトランザクションブランチに届かなくなるとを、ヒューリスティックハザードといいます。ヒューリスティックハザードが起こった場合には、コマンドを使って、トランザクションブランチごとにトランザクションを強制的に決着できます。ルートトランザクションブランチの同期点処理を待たないで、コマンドでトランザクションブランチのトランザクションを強制的に決着することを、ヒューリスティック決着といいます。

ヒューリスティック決着には、次に示す種類があります。

- ヒューリスティックコミット

ルートトランザクションの決着結果に関係なく、トランザクションブランチをコミットすることです。

- ヒューリスティックロールバック

ルートトランザクションの決着結果に関係なく、トランザクションブランチをロールバックすることです。

ヒューリスティックコミットにするときには `eetrncmt` コマンドを、ヒューリスティックロールバックにするときには `eetrnrbk` コマンドを使います。コミットまたはロールバックしたトランザクションブランチを強制終了する場合は、`eetrnfgt` コマンドを使います。

コミットかロールバックかは、ルートトランザクションブランチとトランザクションブランチとで一致させてください。一致させるためには、まずルートトランザクションブランチがコミットしたかロールバックしたかを `eetrnlstr` コマンドを使って調べます。`eetrnlstr` コマンドで表示した状態を参照した上で、`eetrncmt` コマンドまたは `eetrnrbk` コマンドを実行します。

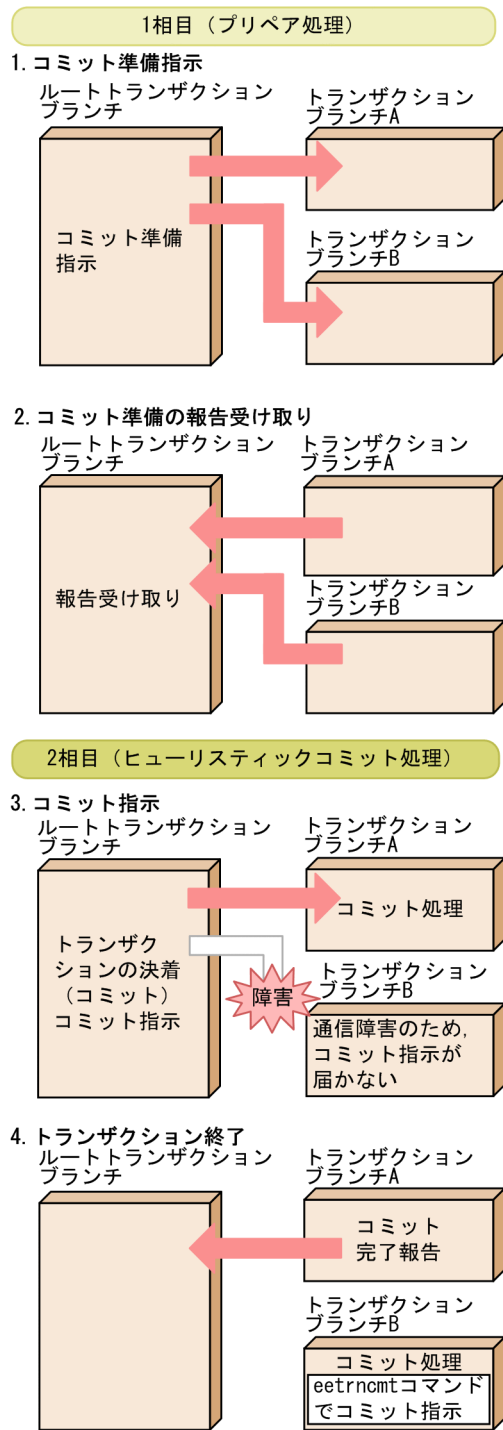
コマンドでトランザクションを決着させた場合、トランザクションの扱いについては十分考慮してください。ルートトランザクションブランチの決着とトランザクションブランチの決着とが矛盾（ヒューリスティックミックス）しないように十分注意してから `eetrncmt` コマンドまたは `eetrnrbk` コマンドを実行してください。

ヒューリスティック決着の操作方法については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」のトランザクションに関する運用の説明を参照してください。

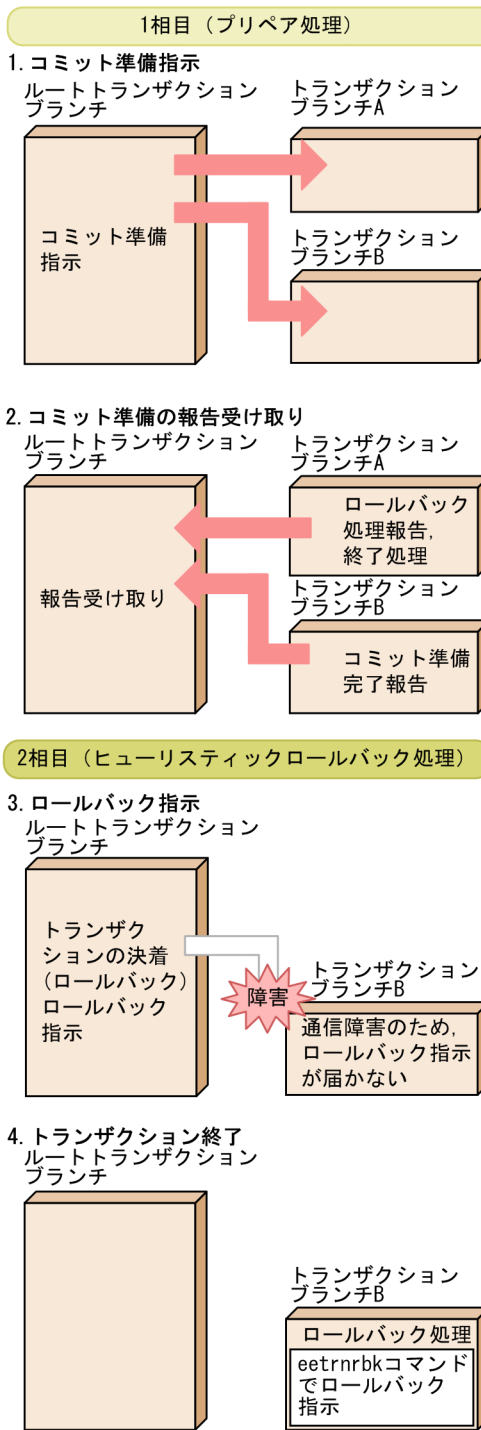
ヒューリスティック決着の流れを次の図に示します。

図 2-24 ヒューリスティック決着の流れ

●ヒューリスティックコミットの流れ



●ヒューリスティックロールバックの流れ



2.2.4 トランザクションの最適化

TP1/EE のトランザクションマネージャは、トランザクション処理の性能を上げるため、トランザクション関連定義の `trn_optimum_item` オペランドの指定に従って、次のような最適化の処理をしています。

- 1 相最適化

- リードオンリー最適化

それぞれの最適化には、最適化できる条件があります。条件を満たした UAP を作成すると、トランザクション処理の性能を上げることができます。

1 相最適化は、リードオンリー最適化より優先度が高い処理です。

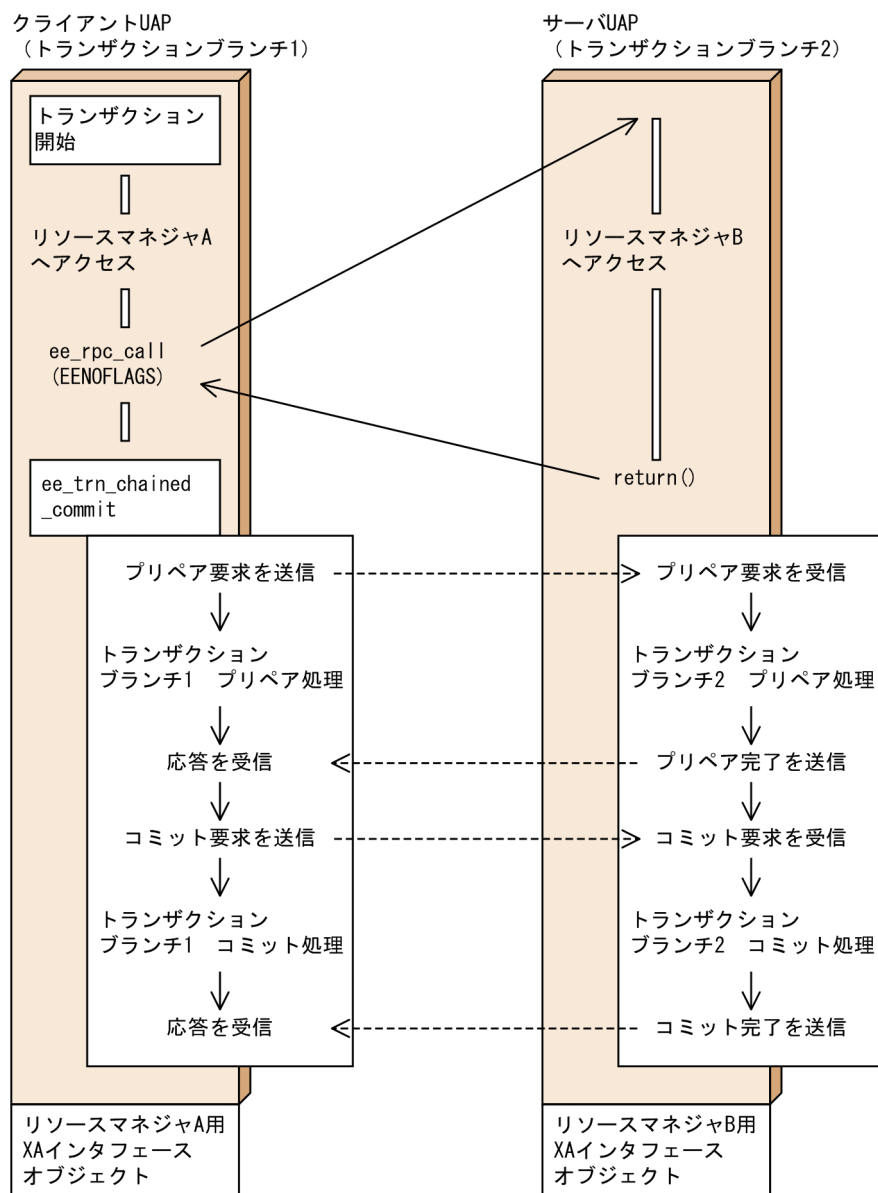
トランザクションの最適化は、クライアント側のトランザクションブランチとサーバ側のトランザクションブランチとの間で、同期点処理の効率を上げることを目的としています。一つのグローバルトランザクション内で複数の最適化を混在できます。

(1) 通常のトランザクション処理 (2 相コミット)

TP1/EE では、X/Open の XA インタフェースによってトランザクションを制御しています。XA インタフェースでは、**プリペア処理**と**コミット処理**に分けて、トランザクションの同期点を取得しています。したがって、クライアント UAP とサーバ UAP の間では、同期点処理要求の送信と確認の応答で、合計 **4 回**通信することになります。2 相コミットの処理では、トランザクション処理のプロセスは同期点を取得するまで、次のサービス要求を受け取れません。

通常のトランザクション処理 (2 相コミット) の概要を、次の図に示します。

図 2-25 通常のトランザクション処理（2 相コミット）の概要



(2) 1 相最適化

1 相最適化の条件は、トランザクションブランチがサーバ側にあるか、クライアント側にあるかによって異なります。

注意事項

サーバ側のトランザクションブランチがあり、トランザクション処理でコミット時に実行される次の関数の呼び出しがあった場合は、1 相最適化をしません。

- ee_rpc_cmtsend 関数 【CBLEERPC('CMTSND ')】
- ee_rpc_cmtsend_to 関数 【CBLEERPC('CMTSNDTO')】

- ee_tim_execap 関数 【CBLEETIM('EXECAP ')】 (acttype に EETIM_COMT 指定)
- ee_mch_cmtsend 関数 【CBLEEMCH('CMTSND ')】
- ee_mch_cmtsend_sync 関数 【CBLEEMCH('CMTSND SY')】
- ee_mcp_send 関数 【CBLEEMCP('SEND ')】 (MCP を使用する場合)

次の場合には、トランザクション結果は不定となります。

- 1 相最適化の実行中に通信障害やリソースマネージャ障害が発生し、ヒューリスティックハザード扱いとなった場合。
- 1 相最適化の実行中にプロセスダウンが発生し、ロールバック扱いとなった場合。

サーバ側のトランザクションブランチがある場合

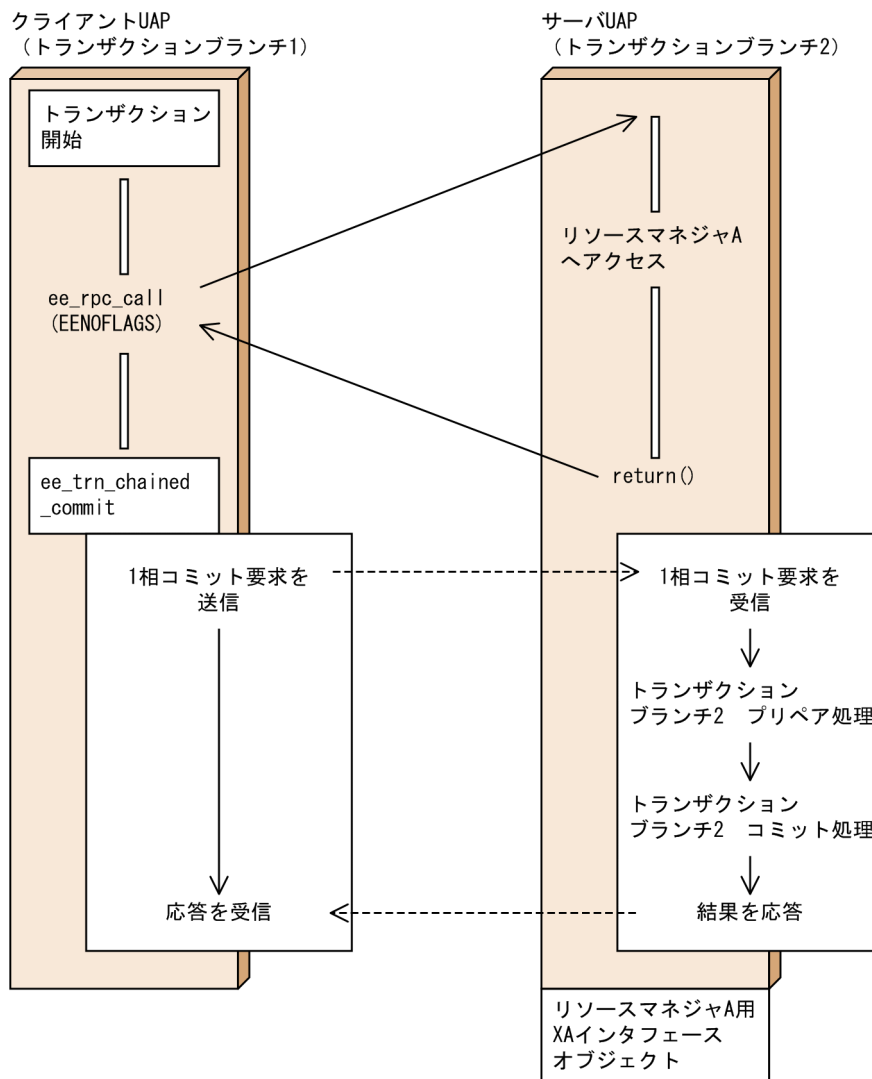
1 相最適化の条件を満たすと、クライアント側のトランザクションブランチがリソースマネージャへアクセスしないため、サーバ側のトランザクションブランチの同期点処理だけで済みます。このことで、2 回分のプロセス間通信が不要になって、トランザクション処理の性能が上がります。

次に示す条件をすべて満たした場合に、1 相最適化は実行されます。

- トランザクション関連定義の trn_optimum_item オペランドに onephase を指定している。
- クライアント側のトランザクションブランチにリンケージされているリソースマネージャは、動的登録をするリソースマネージャだけである。
- クライアント側のトランザクションブランチは、リソースマネージャへアクセスしていない。
- クライアント側のトランザクションブランチには、子トランザクションブランチが一つだけである。

サーバ側のトランザクションブランチがある場合の 1 相最適化の概要を次の図に示します。

図 2-26 サーバ側のトランザクションブランチがある場合の 1 相最適化の概要



クライアント側のトランザクションブランチだけの場合

1 相最適化の条件を満たすと、クライアント側の同期点処理だけで済みます。このことで、リソースマネージャへのアクセス 1 回分が不要になり、トランザクション処理の性能が上がります。

次に示す条件をすべて満たした場合に、1 相最適化は実行されます。

- クライアント側のトランザクションブランチにリンケージされているリソースマネージャは一つだけである。

(3) リードオンリー最適化

リードオンリー最適化は、トランザクション関連定義の `trn_optimum_item` オペランドに `readonly` を指定したときだけ実行されます。

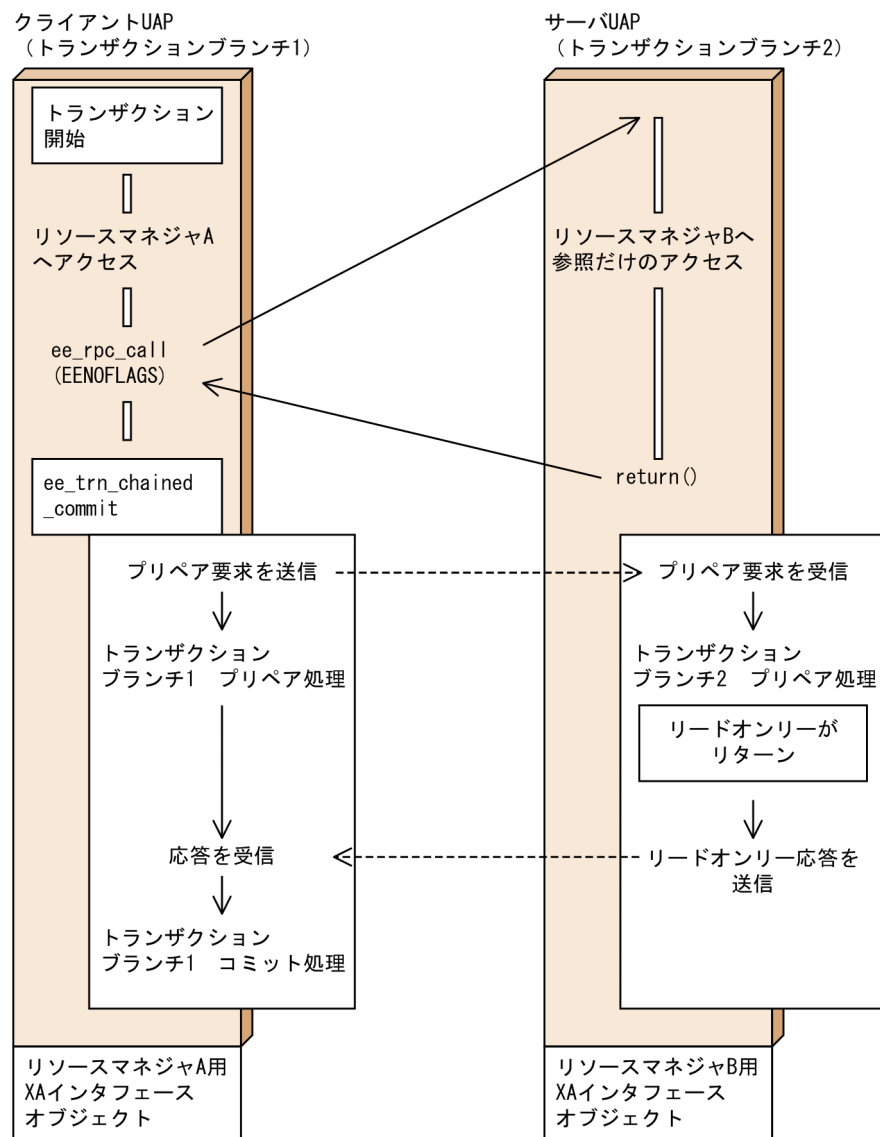
リードオンリー最適化の条件を満たすと、サーバ側のトランザクションブランチが更新処理をしていない場合に、同期点取得処理の 2 相目を実行しません。このことで、2 回分のプロセス間通信が不要になって、トランザクション処理の性能が上がります。

次に示す条件をすべて満たした場合に、リードオンリー最適化は実行されます。

- サーバ側のトランザクションブランチは、リソースの更新処理をしていない。

リードオンリー最適化の概要を次の図に示します。

図 2-27 リードオンリー最適化の概要



注意事項

トランザクション処理でコミット時に実行される次の関数の呼び出しがあった場合は、リードオンリー最適化をしません。

- ee_rpc_cmtsend 関数
- ee_rpc_cmtsend_to 関数
- ee_tim_execap 関数 (acttype に EETIM_COMT 指定)

- ee_mch_cmtsend 関数 (XTC を使用する場合)
- ee_mch_cmtsend_sync 関数 (XTC を使用する場合)
- ee_mcp_send 関数 (MCP を使用する場合)

2.3 タイマトランザクションの登録

タイマトランザクションは、一定時間経過後に実行する業務処理のトランザクションです。時刻または時間を指定して、サービストランザクションと同じサービスに登録すると、各サービスと連動して業務処理を実行できます。

タイマトランザクションを起動するときは、次に示す関数を呼び出します。

- ee_tim_execap 関数 【CBLEETIM('EXECAP ')】

ee_tim_execap 関数の service に起動させたいタイマトランザクションのサービス名を指定します。

タイマトランザクションの起動を取り消すときは、次に示す関数を呼び出します。

- ee_tim_execap_cancel 関数 【CBLEETIM('CANCEL ')】

タイマトランザクションの起動タイミングは、時刻または経過時間で指定します。タイマトランザクションの起動方式には、次の方式があります。

- 即時登録方式
- トランザクション同期登録方式

2.3.1 即時登録方式

即時登録方式でタイマトランザクションを起動するには、ee_tim_execap 関数の acttype に EETIM_INST を指定します。即時登録方式は、ee_tim_execap 関数の呼び出し時点から指定された時刻または時間に達したときに、タイマトランザクションを起動します。起動する時刻または時間の指定方法を次に示します。

- 時刻指定

タイマトランザクションを起動する時刻を指定します。指定時刻が現在の時刻以前の場合は、即時起動します。ee_tim_execap 関数の action に EETIM_TIME を、active に起動する時刻を指定します。

- 経過時間指定

ee_tim_execap 関数が呼び出されてから、タイマトランザクションを起動するまでの経過時間を指定します。ee_tim_execap 関数の action に EETIM_INTV を、active に経過時間を指定します。

ee_tim_execap 関数の action に EETIM_JUST を指定した場合、ee_tim_execap 関数が呼び出されたときに、タイマトランザクションを起動します。この場合、ee_tim_execap 関数の active に時刻または時間を指定しません。

即時登録方式で登録したタイマトランザクションは、トランザクションが異常終了またはロールバックに決着した場合でも有効です。

2.3.2 トランザクション同期登録方式

トランザクション同期登録方式でタイマトランザクションを起動するには、ee_tim_execap 関数の acttype に EETIM_COMT を指定します。トランザクション同期登録方式は、同期点を取得した時点（コミットした時点）から指定された時刻または時間に達したときに、タイマトランザクションを起動します。

指定した時刻または時間に達していても、同期点を取得していない場合、タイマトランザクションを起動しません。この場合、同期点の取得時にトランザクションを起動します。起動する時刻または起動するまでの経過時間の指定方法を次に示します。

- 時刻指定

タイマトランザクションを起動する時刻を指定します。指定時刻が現在の時刻以前の場合は、即時起動します。ee_tim_execap 関数の action に EETIM_TIME を、active に起動する時刻を指定します。

- 経過時間指定

同期点を取得してから、タイマトランザクションを起動するまでの経過時間を指定します。

ee_tim_execap 関数の action に EETIM_INTV を、active に経過時間を指定します。

ee_tim_execap 関数の action に EETIM_JUST を指定した場合、同期点を取得されたときに、タイマトランザクションを起動します。この場合、ee_tim_execap 関数の active に時刻または時間を指定しません。

トランザクション同期登録方式で登録したタイマトランザクションは、トランザクションが異常終了またはロールバックに決着した場合は無効になります。

トランザクション同期登録方式で登録したタイマトランザクションの取り消しは、トランザクションが異常終了またはロールバックに決着した場合は無効になります。このとき、起動する時刻または時間に達していたタイマトランザクションは、即時起動されます。

2.3.3 時刻または経過時間の指定方法

タイマトランザクションの起動タイミングは、時刻または経過時間で指定できます。タイマトランザクションの起動タイミングは、ee_tim_execap 関数の active に指定します。

(1) 時刻で指定する場合

起動タイミングを時刻で指定する場合は、時刻をバイナリ形式で指定します。指定できる範囲は 0～86399（0 時 0 分 0 秒～23 時 59 分 59 秒）で、単位は秒です。例えば、14 時 30 分 30 秒に起動させたい場合は、52230（ $14 \times 3600 + 30 \times 60 + 30$ ）と指定してください。なお、起動タイミングとして指定する時刻は、プロセス関連定義の execap_permission_time オペランドで指定した範囲の時刻にしてください。

(2) 経過時間で指定する場合

起動タイミングを経過時間で指定する場合は、経過時間をバイナリ形式で指定します。指定できる範囲は、0～3600000（0 ミリ秒～100 時間）で、単位は 100 ミリ秒です。例えば、1 と指定した場合は、100 ミリ秒と指定したことになります。

2.3.4 タイマトランザクションの取り消し

タイマトランザクションの起動を取り消すときは、`ee_tim_execap_cancel` 関数を呼び出します。タイマトランザクションは、個別に指定して起動を取り消せます。起動を取り消したいタイマトランザクションのリクエスト識別子（`ee_tim_execap` 関数の `reqid` の指定値）を、`ee_tim_execap_cancel` 関数の `reqid` に指定します。同じリクエスト識別子で、複数のタイマトランザクションの起動要求をしている場合は、そのタイマトランザクションの起動方式によって起動を取り消すタイマトランザクションが異なります。

即時登録方式で登録したタイマトランザクションの場合

`ee_tim_execap_cancel` 関数の `acttype` に `EETIM_INST` または `EETIM_CMTC` を指定すると、タイマトランザクションを起動する時刻または時間の指定が最も早いタイマトランザクションの起動を取り消します。

- `EETIM_INST` を指定したとき：関数の呼び出し時に、トランザクションの決着（コミット決定、またはロールバック決定）に関係なく、タイマトランザクションの起動を取り消します。
- `EETIM_CMTC` を指定したとき：トランザクションのコミット後に、タイマトランザクションの起動を取り消します。関数の呼び出し後に、対象のタイマトランザクションがコミット前に起動時刻に達しても、タイマトランザクションは起動しません。

すでに起動時刻に達しているタイマトランザクションの起動は取り消せません。

トランザクション同期登録方式で登録したタイマトランザクションの場合

`ee_tim_execap_cancel` 関数の `acttype` に `EETIM_INST`、`EETIM_COMT` または `EETIM_CMTC` を指定すると、`ee_tim_execap` 関数の呼び出しが最も早いタイマトランザクションの起動を取り消します。

- `EETIM_INST` を指定したとき：関数の呼び出し時に、コミット後のタイマトランザクションの起動を取り消します。
- `EETIM_COMT` を指定したとき：関数の呼び出し時に、コミット前のタイマトランザクションの起動を取り消します。
- `EETIM_CMTC` を指定したとき：トランザクションのコミット後に、コミット後のタイマトランザクションの起動を取り消します。関数の呼び出し後に、対象のタイマトランザクションがコミット前に起動時刻に達しても、タイマトランザクションは起動しません。

トランザクション同期登録方式で起動したタイマトランザクションの場合、`ee_tim_execap_cancel_all` 関数【`CBLEETIM('CANCELAL')`】を呼び出すと、コミット前のすべてのタイマトランザクションを取り消せます。

2.3.5 システム終了時のタイマトランザクションの扱い

タイマトランザクション登録中に、正常終了または計画停止 A でシステムを終了させた場合、タイマトランザクションの破棄を通知するエラートランザクション (ERRTRN4) が起動して、トランザクションの破棄を連絡します。終了モードが計画停止 B の場合は、エラートランザクションは起動しません。

プロセス終了処理中にタイマトランザクションの起動要求をした場合、エラーリターンします。

2.3.6 時刻または経過時間の指定と TP1/EE サービス定義との関係

タイマトランザクションの開始を時刻または経過時間で指定する場合の TP1/EE サービス定義との関係を説明します。

(1) タイマトランザクション起動時の誤差

時刻指定と経過時間指定のどちらの場合も、タイマトランザクションの起動時には、最大で次の合計値が誤差として発生するおそれがあります。

- ・ イベント監視インタバル間隔時間 (モニタスレッドの監視間隔時間)

プロセス関連定義の `clock_time_interval` オペランドで指定した間隔時間で、タイマトランザクションの起動時間を監視しています。このため、イベント監視を開始するモニタスレッドの起動タイミングによって、タイマトランザクションの起動タイミングに誤差 (遅れ) が発生するおそれがあります。

また、CPU 使用率の高騰によってモニタスレッドの監視間隔に間延びが生じるおそれがあります。

- ・ 処理キュー滞留時間 (処理キュー引き出し待ち時間)

タイマトランザクションが起動時間に達して、タイマトランザクション起動用の処理キューを登録するとき、処理スレッド空き待ちによる処理キュー滞留 (処理キューの引き出し待ち) のため、タイマトランザクションの起動タイミングに誤差 (遅れ) が発生するおそれがあります。

(2) 先行許容時間／後退許容時間と指定時刻の関係

時刻指定の場合、先行許容時間および後退許容時間の範囲内の時刻を指定します。先行許容時間および後退許容時間は、プロセス関連定義の `execap_permission_time` オペランドに指定します。

起動タイミングとして指定した時刻が、現在時刻以降の時刻の場合、先行許容時間の指定値に従って処理をします。起動タイミングとして指定した時刻が、先行許容時間内の時刻のときは、指定時刻にタイマトランザクションを起動します。起動タイミングとして指定した時刻が、先行許容時間外の時刻のときは、タイマトランザクションの起動要求はエラーリターンします。

起動タイミングとして指定した時刻が、現在時刻以前の時刻の場合、後退許容時間の値に従って処理をします。起動タイミングとして指定した時刻が、後退許容時間内の時刻のときは、タイマトランザクションを即時起動します。起動タイミングとして指定した時刻が、後退許容時間外の時刻のときは、タイマトランザクションの起動要求はエラーリターンします。

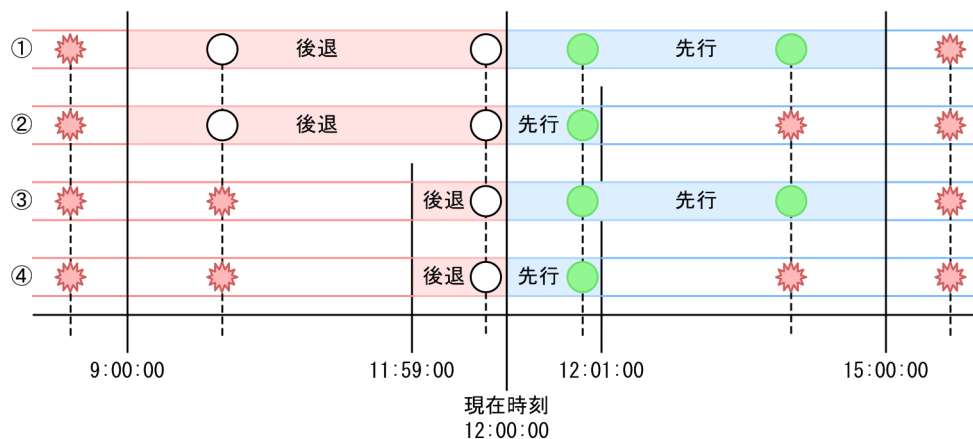
なお、トランザクション同期登録方式で登録要求した場合、ee_tim_execap 関数の呼び出し時に先行許容時間および後退許容時間をチェックします。

現在時刻が 12 時 00 分 00 秒の場合の、先行許容時間、後退許容時間と、起動タイミングとして指定した時刻との関係を次の表および図に示します。

表 2-4 先行許容時間／後退許容時間と起動タイミングの指定時刻との関係

項番	execap_permission_time オペランドの指定値 (単位：分)		起動タイミングの指定時刻	起動要求の処理
	先行許容時間	後退許容時間		
1	180	180	9:00:00～12:00:00 (32400～43200)	正常（即時起動）
2			12:00:01～15:00:00 (32401～54000)	正常（指定時刻で起動）
3			上記以外	エラーリターン
4	1	180	9:00:00～12:00:00 (32400～43200)	正常（即時起動）
5			12:00:01～12:01:00 (43201～43260)	正常（指定時刻で起動）
6			上記以外	エラーリターン
7	180	1	11:59:00～12:00:00 (43140～43200)	正常（即時起動）
8			12:00:01～15:00:00 (43201～54000)	正常（指定時刻で起動）
9			上記以外	エラーリターン
10	1	1	11:59:00～12:00:00 (43140～43200)	正常（即時起動）
11			12:00:01～12:01:00 (43201～43260)	正常（指定時刻で起動）
12			上記以外	エラーリターン

図 2-28 先行許容時間／後退許容時間と起動タイミングの指定時刻との関係



(凡例)

- ①： 先行許容時間：180，後退許容時間：180の場合
- ②： 先行許容時間：1，後退許容時間：180の場合
- ③： 先行許容時間：180，後退許容時間：1の場合
- ④： 先行許容時間：1，後退許容時間：1の場合

先行： 先行許容時間

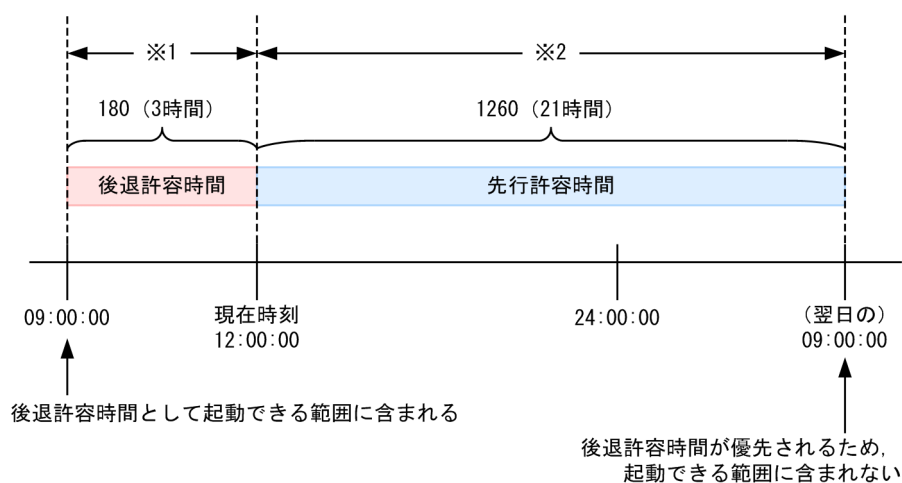
後退： 後退許容時間

○： 12:00:00に即時起動する起動要求

●： 指定時刻に起動する起動要求

★： エラーリターンする起動要求

先行許容時間と後退許容時間との合計が、1440（24 時間）になる場合、両方の許容時間の範囲内になる時刻が存在します。この場合、後退許容時間内の時刻と判断します。例えば、現在時刻が 12 時 00 分 00 秒の場合に、先行許容時間に 1260（21 時間）、後退許容時間に 180（3 時間）を指定すると、9 時 00 分 00 秒が、両方の許容時間の範囲内になります。この状態で、タイマトランザクションを起動する時刻に 32400（9 時 00 分 00 秒）を指定すると、当日の 9 時 00 分 00 秒、または翌日の 9 時 00 分 00 秒が起動時刻になる可能性があります。この場合、後退許容時間内の時刻と判断して、当日の 9 時 00 分 00 秒を起動時刻とします。当日の 9 時 00 分 00 秒は過ぎているので、タイマトランザクションを即時起動します。



注※1

タイマトランザクションを即時起動する範囲です。

注※2

指定した時刻にタイマトランザクションを起動する範囲です。

2.4 DB キューによるメッセージ制御

DB キュー機能を使用すると、データベースをキューとして、TP1/EE 内での AP 間通信や、TP1/EE と他システムとのシステム間通信で、非同期型のメッセージ送受信ができます。

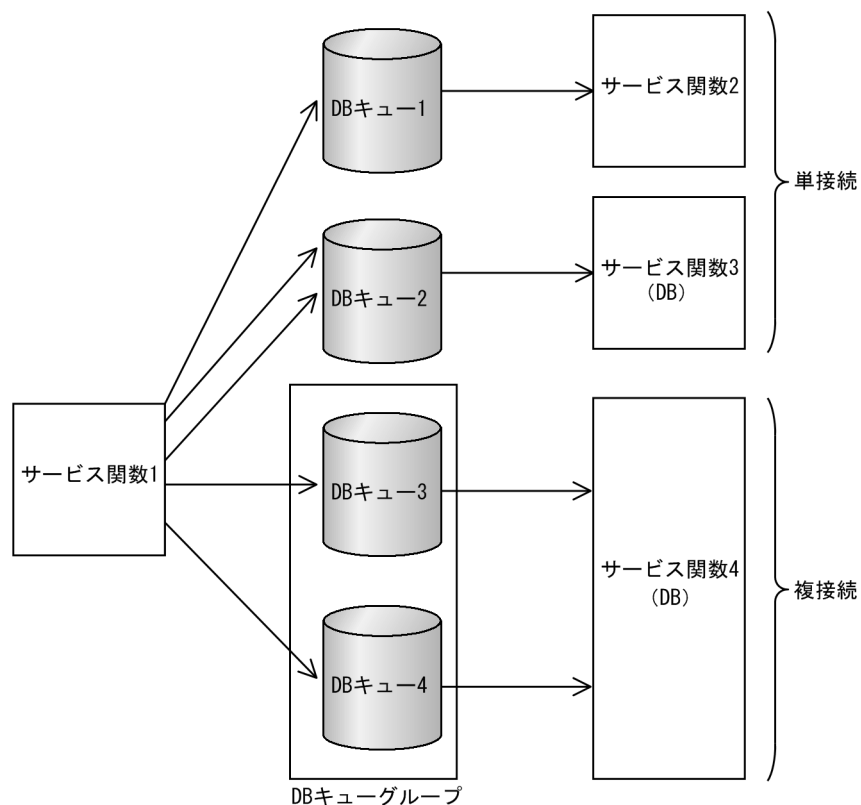
2.4.1 DB キューを使用した AP 間通信

DB キューとサービス関数の接続形態には、単接続と複接続とがあります。

DB キューにメッセージが書き込まれると、サービストラランザクションが起動されます。起動されたサービストラランザクションがコミットされると、要求先サービス関数が起動されます。

単接続と複接続の接続形態、およびサービス関数起動の概念図を次に示します。

図 2-29 単接続と複接続の接続形態、サービス関数起動の概念



(1) 単接続

単接続の接続形態とサービス関数起動について説明します。

(a) 接続形態

DB キューと要求先サービス関数とが 1 対 1 で対応する接続方法です。DB キューに書き込まれたメッセージに対応する要求先サービス関数を起動し、メッセージを引き渡します。

(b) サービス関数起動

図 2-29 のサービス関数 1 が起動したサービストランザクションから、DB キュー 1 に 1 メッセージ、DB キュー 2 に 2 メッセージが書き込まれた場合、サービス関数 2、およびサービス関数 3 が起動されます。コミット時にスケジュールされるトランザクションは、サービス関数 2、サービス関数 3 で 1 トランザクションずつです。DB キュー 2 に 2 メッセージが書き込まれても、サービス関数 3 で同時に 2 トランザクション起動されることはありません。

(2) 複接続

複接続の接続形態とサービス関数起動について説明します。

(a) 接続形態

DB キューと要求先サービス関数とが n 対 1 で対応する接続方法です。複数の DB キューに書き込まれたメッセージは DB キュー単位で読み出され、順にサービス関数に引き渡されます。

(b) サービス関数起動

図 2-29 のサービス関数 4 と接続されている DB キューグループに対して、1 トランザクションで複数の DB キューに書き込みがあった場合、メッセージ読み出しスケジュール方式に従い、順に読み出されてサービス関数 4 を起動します。

サービス関数の起動は DB キュー単位でシリアル化されるため、一つのサービス関数に対して最大で接続 DB キュー数分のトランザクションが同時に起動されます。

サービストランザクション起動の多重管理

サービス関数から DB キューにメッセージの書き込み要求をしたとき、一つのメッセージに対して一つのサービストランザクションが起動されます。そのため、複数の DB キューにメッセージの書き込み要求をすると、一時的に多数のサービストランザクションが起動されます。

DB キュー機能関連定義の dbqsrdef 定義コマンドで、サービス単位に同時に起動させるトランザクションの最大数を指定することで、起動するサービストランザクション数を制限できます。

メッセージ読み出しスケジュール方式

DB キューからメッセージを読み出すときには、次のスケジュール方式があります。

- 優先スケジュール方式

DB キュー機能関連定義の dbqsrdef 定義コマンドの -q オプションで最初に指定された DB キューから毎回読み出します。

DB キュー機能関連定義の dbqsrdef 定義コマンドの -q オプションで最初に指定された DB キューにメッセージがない場合、DB キュー機能関連定義の dbqsrdef 定義コマンドの -q オプションの 2 番目以降に指定された DB キューから、1 メッセージずつ読み出します。

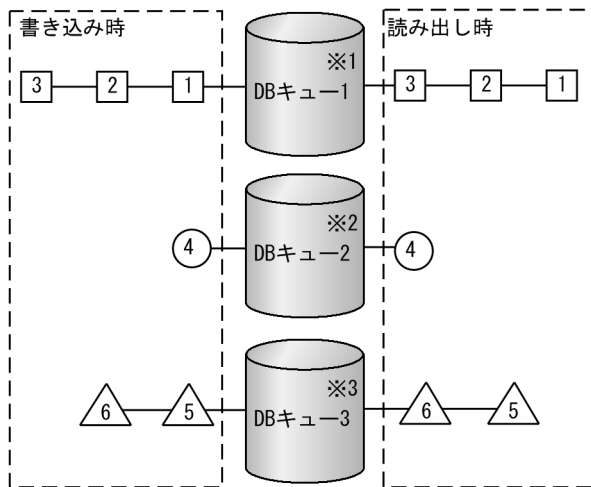
- 均等スケジュール方式

DB キュー機能関連定義の dbqsrdef 定義コマンドの -q オプションに指定した順番に、DB キューから、1 メッセージずつ読み出します。

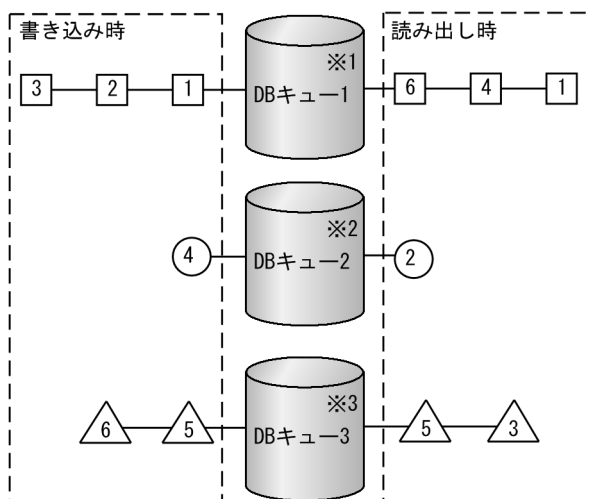
スケジュール方式は、DB キュー機能関連定義の dbqsrdef 定義コマンドの -c オプションで指定します。サービス単位に同時に起動できる最大トランザクション数を 1 としたときの優先スケジュール方式と均等スケジュール方式を、次の図に示します。

図 2-30 スケジュール方式

●優先スケジュール方式



●均等スケジュール方式



(凡例)

□ n ○ n △ n : メッセージ (nはDBキューに書き込まれた／DBキューから読み出された順番)

注※1

DBキュー関連定義のdbqsrdef定義コマンドの-qオプションで最初に指定されたDBキュー

注※2

DBキュー関連定義のdbqsrdef定義コマンドの-qオプションで2番目に指定されたDBキュー

注※3

DBキュー関連定義のdbqsrdef定義コマンドの-qオプションで3番目に指定されたDBキュー

2.4.2 DB キューサービスのトランザクションのコミット決着

DB キューサービスのトランザクションは、サービス関数の終了時（サービス関数からリターン時）にコミットします。

DB キュー以外の DB をサービス関数処理の途中でコミットする場合は、サービス関数から、`ee_trn_chained_commit` 関数を呼び出して DB キューをコミットします。

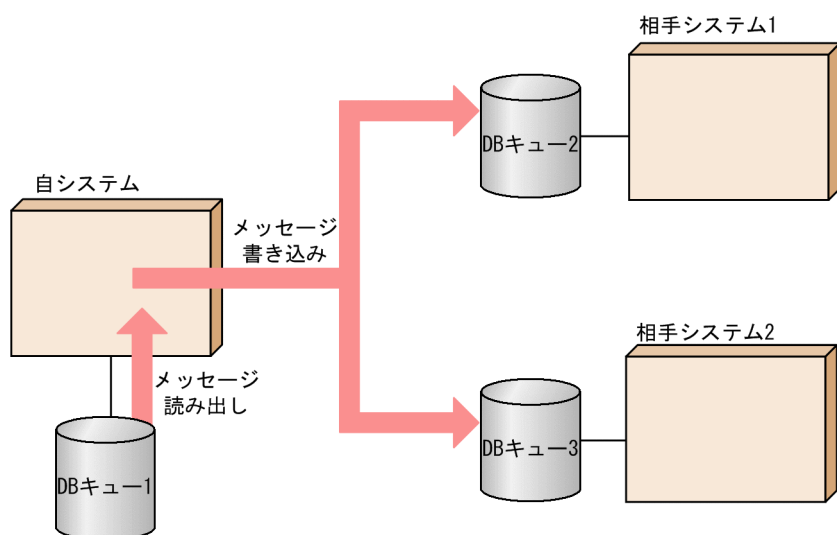
2.4.3 DB キューを使用したシステム間通信

DB キューを使用したシステム間通信の概要について説明します。

(1) DB キューを使用したシステム

DB キューは、DB キューからメッセージを読み出すシステムと関連づけられています。DB キューとシステムとの関連を次の図に示します。

図 2-31 DB キューとシステムとの関連



システムは、次の DB キューを使用してシステム間通信をします。

- 読み出しをする DB キュー（自システムと連携している DB キュー）
- 書き込みをする DB キュー（相手システムと連携している DB キュー）

この図の場合、自システムは、読み出しをする DB キューとして DB キュー 1 を使用します。書き込みをする DB キューとして DB キュー 2 および DB キュー 3 を使用します。

システムは、複数の DB キューと連携できます。複数の DB キューは、DB キューグループとしてまとめることができます。DB キューに関する定義は、DB キュー機能関連定義で指定します。

(2) メッセージの読み出しと書き込み

DB キューからメッセージを読み出す処理、および DB キューにメッセージを書き込む処理の流れを説明します。

(a) メッセージの読み出し

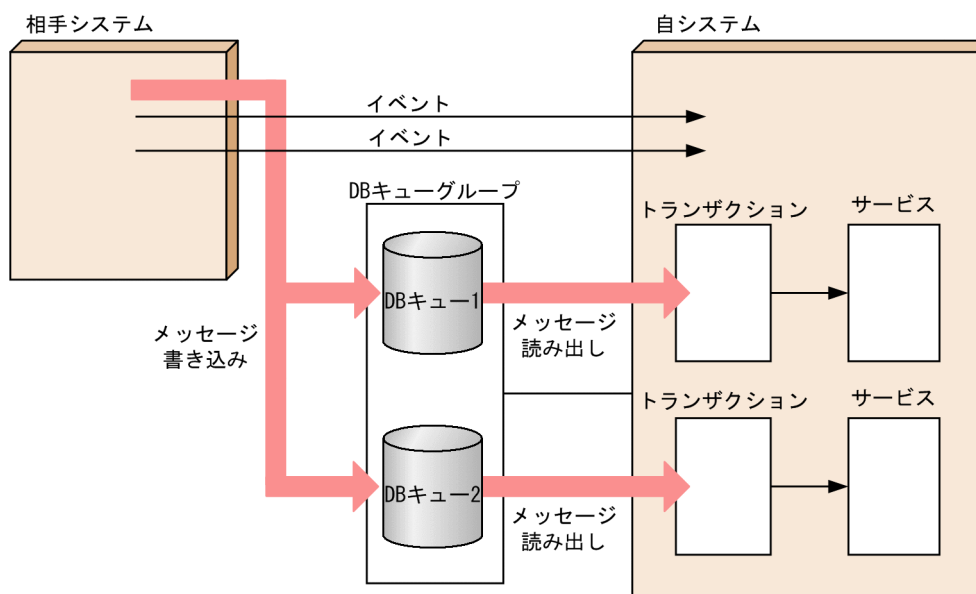
メッセージの読み出しは、AP 間通信をする場合とユーザキューアクセスをする場合で処理が異なります。

DB キュー機能を使用して AP 間通信をする場合

DB キューに書き込まれたメッセージは、イベント受信を契機にメッセージを読み出すトランザクションに読み出されます。メッセージを読み出すトランザクションは、DB キュー機能関連定義で指定されたサービスにメッセージの処理を要求します。メッセージを読み出すトランザクションは、メッセージの処理を実行するサービスごとに起動します。

メッセージの読み出し処理の流れについて、次の図に示します。

図 2-32 メッセージの読み出し処理の流れ



複数のシステムが DB キューにメッセージを書き込むため、自システムは複数のシステムからのイベントを受信しますが、イベント送信元（メッセージの書き込み元）を意識しないでメッセージの読み出しをします。

DB キュー機能を使用してユーザキューアクセスをする場合

メッセージ読み出しトランザクションで、メッセージの読み出し処理をする UAP を起動し、UAP 内で `ee_dbq_msgget` 関数【`CBLEEDBQ('MSGGET ')`】を呼び出して、DB キューに書き込まれたメッセージを読み込みます。

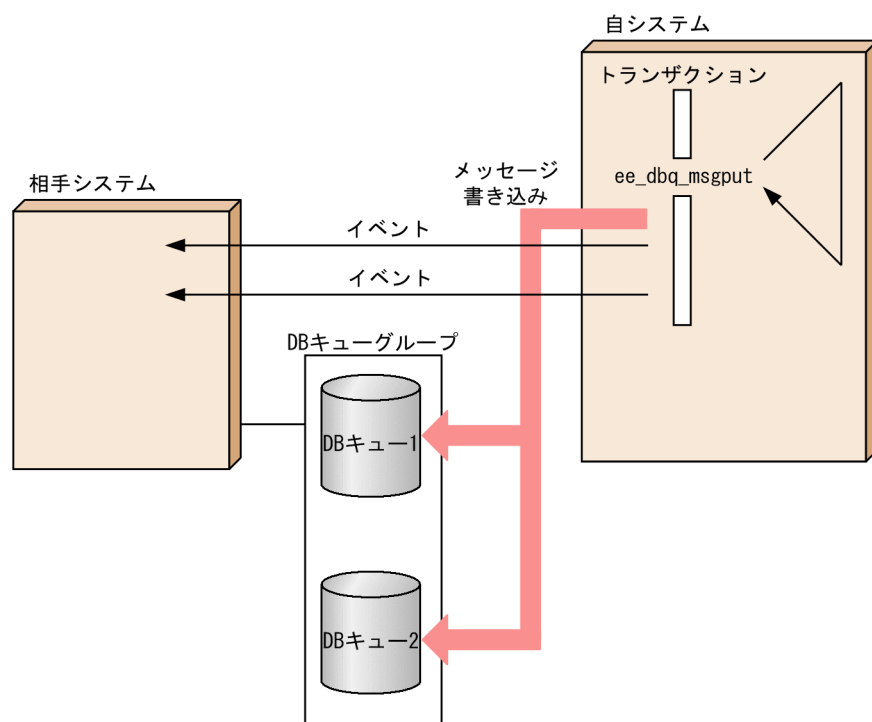
(b) メッセージの書き込み

メッセージを書き込むトランザクションで、`ee_dbq_msgput` 関数を呼び出します。トランザクションのコミット時に DB キューにメッセージを書き込みます。メッセージ書き込みが完了すると、メッセージを

書き込んだ DB キューと連携しているシステムにイベントを送信します。イベントは、DB キューごとに送信します。

メッセージの書き込み処理の流れについて、次の図に示します。

図 2-33 メッセージの書き込み処理の流れ



(3) TCP/IP を使用したシステム間通信

DB キューを使用したシステム間通信では、TCP/IP または RPC のどちらかの通信形態を、サービスグループ情報関連定義の `eesvgdef` 定義コマンド、`mysvgdef` 定義コマンド、または DB キュー機能関連定義の `dbqgrpdef` 定義コマンドの `-t` オプションで指定できます。ただし、TP1/EE 01-02 以降と通信する場合は、性能やリソースなどの点で、TCP/IP 通信を指定することをお勧めします。また、次に示す場合は、使用できる通信形態が決められています。

- TP1/EE 以外のシステム (J2EE システム) と通信する場合
TCP/IP 通信を指定する必要があります。
- TP1/EE 01-01 と通信する場合
RPC 通信を指定する必要があります。

なお、RPC を使用したシステム間通信をする場合は、ネームサービスを使用できません。

ここでは、TCP/IP を使用したシステム間通信について説明します。

(a) TCP/IP 通信でのイベントの送受信

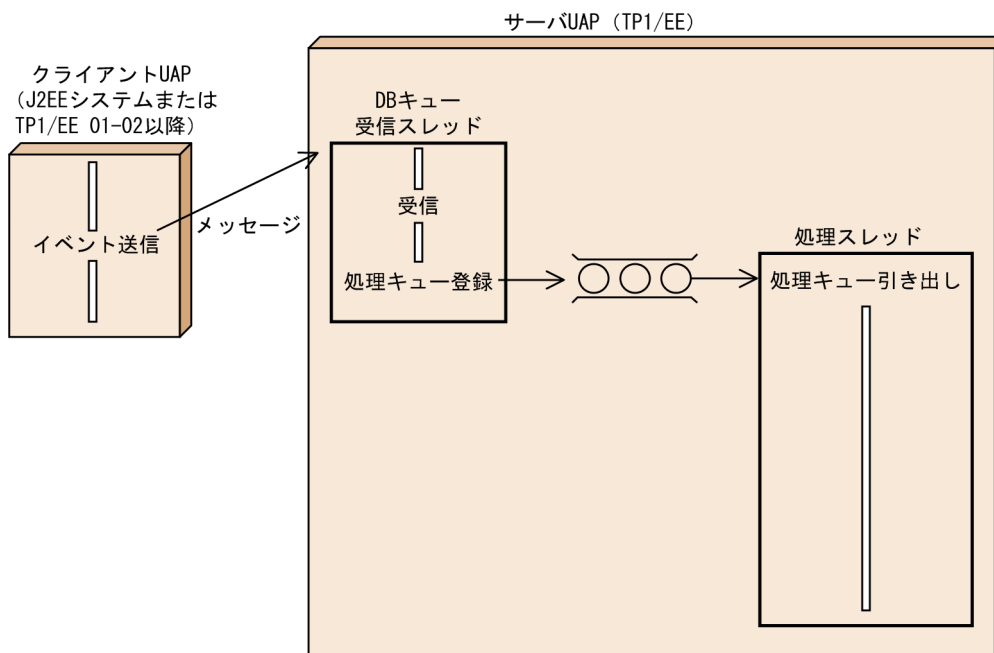
■ TP1/EE がサーバ UAP の場合の TCP/IP 通信

TP1/EE がサーバ UAP の場合の TCP/IP 通信では、クライアント UAP（J2EE システムまたは TP1/EE 01-02 以降）からイベントを受信するため、サービスグループ情報関連定義の mysvgdef 定義コマンドの -t オプションに DBQ を指定した、DB キュー受信スレッド情報を設定します。

受信したイベントは、処理キューに登録されます。処理キューに登録されたイベントは、処理スレッドによって引き出されます。

TCP/IP 通信でイベントを受信したときの処理の流れを次の図に示します。

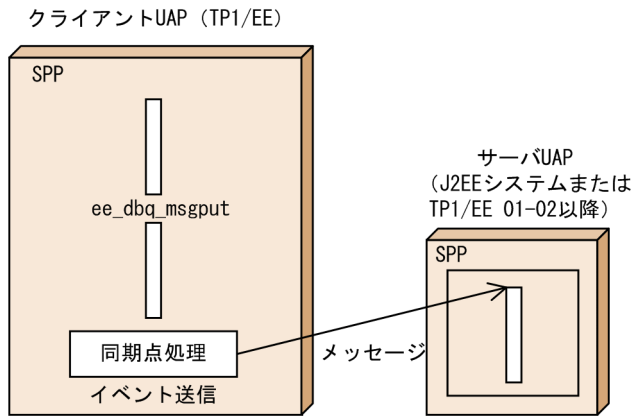
図 2-34 TCP/IP 通信でイベントを受信したときの処理の流れ



■ TP1/EE がクライアント UAP の場合の TCP/IP 通信

TP1/EE がクライアント UAP の場合の TCP/IP 通信で、サーバ UAP（J2EE システムまたは TP1/EE 01-02 以降）に対してイベントを送信したときの処理の流れを次の図に示します。

図 2-35 TCP/IP 通信でイベントを送信したときの処理の流れ



イベントを送信するには、TP1/EE サービス定義でイベントの送信先を指定しておくか、または DB キュー表でイベントの送信先を管理する必要があります。

• TP1/EE サービス定義でイベントの送信先を指定しておく方法

この方法には、サービスグループ情報関連定義または DB キュー機能関連定義を使用します。

サービスグループ情報関連定義を使用する場合

次に示す定義でイベントの送信先を指定しておくことで、指定したサービスグループ下のサーバ UAP に対してイベントを送信します。

- サービスグループ情報関連定義の `eesvgdef` 定義コマンドの `-t` オプションに DBQ を指定
- DB キュー機能関連定義の `dbqgrpdef` 定義コマンドの `-t` オプションを指定
- DB キュー機能関連定義の `dbqgrpdef` 定義コマンドの `-e` オプションにサービスグループ名を指定

DB キュー機能関連定義を使用する場合

DB キュー機能関連定義の `dbqprcdef` 定義コマンドの `-h` オプションにイベントの送信先を指定し、RPC 関連定義の `rpc_tcpsend_con_max_cnt` オペランド、`rpc_tcpsend_con_cnt` オペランド、および `rpc_tcpsend_proc_max_cnt` オペランドを指定することによって、TCP/IP 共有送信用コネクションを使用してイベントを送信できます。

TCP/IP 共有送信用コネクションの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」の TCP/IP 共有送信用コネクションに関する記述を参照してください。

• DB キュー表でイベントの送信先を管理する方法

DB キュー機能関連定義の `dbqprcdef` 定義コマンドの `-k` オプションに 1 を指定し、RPC 関連定義の `rpc_tcpsend_con_max_cnt` オペランド、`rpc_tcpsend_con_cnt` オペランド、および `rpc_tcpsend_proc_max_cnt` オペランドを指定することによって、TCP/IP 共有送信用コネクションを使用してイベントの送信先を管理できます。

TCP/IP 共有送信用コネクションを使用すると、イベントの送信先を動的に管理できるため、DB キューからメッセージを読み出すサーバを変更した場合、イベントの送信先も自動的に更新されます。

TCP/IP 共有送信用コネクションの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」の TCP/IP 共有送信用コネクションに関する記述を参照してください。

(b) TCP/IP 通信でイベントを送受信できない場合

TCP/IP 通信でイベントを送受信できない場合について、状況別に説明します。

■ TP1/EE がサーバ UAP の場合で、イベントを受信できないとき

• 不正なイベントを受信した場合

クライアント UAP から受信したイベントのサイズやヘッダ情報が不正である場合は、次の処理をします。

1. エラーメッセージログを出力する。
2. サイズ不正など、後続処理に影響がある障害が発生している場合は、該当コネクションを解放する。

• リソース不足を検知した場合

クライアント UAP からイベント受信時、受信バッファ不足によってイベント受信処理を実行できない場合は、リソース不足として次の処理をします。

1. エラーメッセージを出力する。
2. 該当コネクションを解放する。

■ TP1/EE がクライアント UAP の場合で、イベントを受信できないとき

• イベント送信エラーが発生した場合

イベント送信処理中に障害が発生し、送信エラーになった場合は、エラーメッセージを出力します。

2.4.4 DB キュー読み出しサーバ変更機能

DB キューからメッセージを読み出すサーバが障害でダウンした場合、DB キューに書き込まれるメッセージを読み出せなくなるため、DB キューが未読み出しメッセージで満杯になるおそれがあります。この現象は、DB キュー読み出しサーバを eedbqchgr コマンドで変更することによって回避できます。DB キュー読み出しサーバは、サーバがダウンした場合だけではなく、サーバが起動中の場合にも変更できます。

DB キュー読み出しサーバ変更機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.4.5 DB キュー書き込み抑止機能

DB キューからメッセージを読み出すサーバが終了した場合、ほかのサーバが DB キューに書き込むメッセージを読み出せなくなるため、DB キューが未読み出しメッセージで満杯になるおそれがあります。こ

の現象は、サーバ終了前に eedbqwtpr コマンドを実行して DB キューへのメッセージの書き込みを抑止することによって回避できます。

DB キュー書き込み抑止機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.4.6 DB キューの通番の管理

TP1/EE によって DB キューにメッセージが書き込まれるのは、サービストランザクション（メッセージ書き込み要求を行うトランザクション）の同期点処理のタイミングです。TP1/EE では、このときの書き込みの順番を通番管理しています。

サービストランザクション中で ee_dbq_msgput 関数が発行され、DB キューへのメッセージ書き込み要求が行われると、同期点処理でメッセージを書き込む順番にメッセージに通番が設定されます。この通番は、1 から順番に昇順で設定され、番号が抜け落ちることはありません。

通番が dbqdef 定義コマンドの -l オプションで指定した最大書き込みメッセージ数に達した場合の動作は、eedbqtblh コマンドで DB キューを作成するときの、-a オプションに指定する属性によって設定できます。

DB キューの通番の管理の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.4.7 DB キュー自動読み出し停止機能

DB キュー自動読み出し停止機能を使用すると、AP 間通信をする DB キューのスケジュール状態を読み出し停止にする条件を指定できます。これによって、TP1/EE の起動時に DB キューを読み出し停止にしたり、障害の発生時に自動的に DB キューを読み出し停止にしたりできます。

DB キュー自動読み出し停止機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.4.8 DB キューメッセージスキップ機能

DB キューから読み出したメッセージが不正なメッセージである場合、そのメッセージをスキップすることで業務を続行できます。また、メッセージ読み出しトランザクションで障害が発生した場合も、障害の発生時に読み出していたメッセージをスキップすることで、読み出しを停止することなく次のメッセージを読み出せます。

DB キューメッセージスキップ機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.4.9 DB キュースケジュール状態一括変更機能

自システムの TP1/EE が読み出すすべての DB キューのスケジュール状態を、1 回のコマンドの実行で読み出し停止 (eedbqstop コマンド) にしたり、再開 (eedbqrst コマンド) にしたりできます。

DB キュースケジュール状態一括変更機能の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.5 オンラインバッチ機能

TP1/EE では、UAP を利用して大量のデータのバッチ処理を実現できます。これを**オンラインバッチ機能**と呼びます。TP1/EE のオンラインバッチ機能を使用すると、大量のデータをロット単位で処理することによってデータベースの負荷を分散できたり、データ型とイベント型という 2 種類の処理形態によって多様な業務に対応できたりします。

オンラインバッチ機能を使用するには、プロセス関連定義の dbq_obs_use オペランドに Y を指定します。また、オンラインバッチ機能を使用するには、前提プログラムとして Justware for OLTP が必要です。Justware for OLTP の詳細については、Justware for OLTP のドキュメントを参照してください。

2.5.1 オンラインバッチ機能の概要

オンラインバッチ機能の概要について説明します。

(1) ロット単位の処理

ロットとは、ユーザデータを任意の単位にまとめたものです。オンラインバッチ機能では、ロット単位でバッチ処理を行います。大量のデータをロット単位で処理してデータベースの負荷を分散させることによって、データベースの競合を回避できます。ロットは 1 階層で構成され、一つのロットに対して、一つのトランザクションを起動できます。

(2) トランザクションの種類

オンラインバッチ機能では、次に示すサービストランザクションによってバッチ処理を制御します。

- バッチ処理トランザクション

バッチ処理を実行します。

- バッチ終了トランザクション

次に示す 3 種類があります。

- 正常終了のバッチ終了トランザクション（バッチ処理の正常終了時に起動する）
- 中断のバッチ終了トランザクション（バッチ処理の中断時に起動する）
- 強制終了のバッチ終了トランザクション（バッチ処理の強制終了時に起動する）

(3) オンラインバッチサービスの指定

バッチ処理の開始時に、ロット単位に任意のオンラインバッチサービスを指定して処理できます。バッチ処理開始時のオンラインバッチサービスは、オンラインバッチ機能関連定義の dbqobssrvdef 定義コマンドおよび dbqobslotdef 定義コマンドの -v オプション、または ee_dbq_obsstart 関数【CBLEEDBQ('OBSSTART')】で指定します。

ただし、バッチ処理の終了時の場合は、TP1/EE 全体でオンラインバッチサービスを一つだけ指定できます。バッチ処理の終了時のオンラインバッチサービスは、バッチ機能関連定義の dbqobsrvdef 定義コマンドの -e オプションで指定します。

2.5.2 オンラインバッチ機能の処理形態

オンラインバッチ機能の処理形態には、データ型とイベント型の 2 種類があります。

(1) データ型オンラインバッチ処理

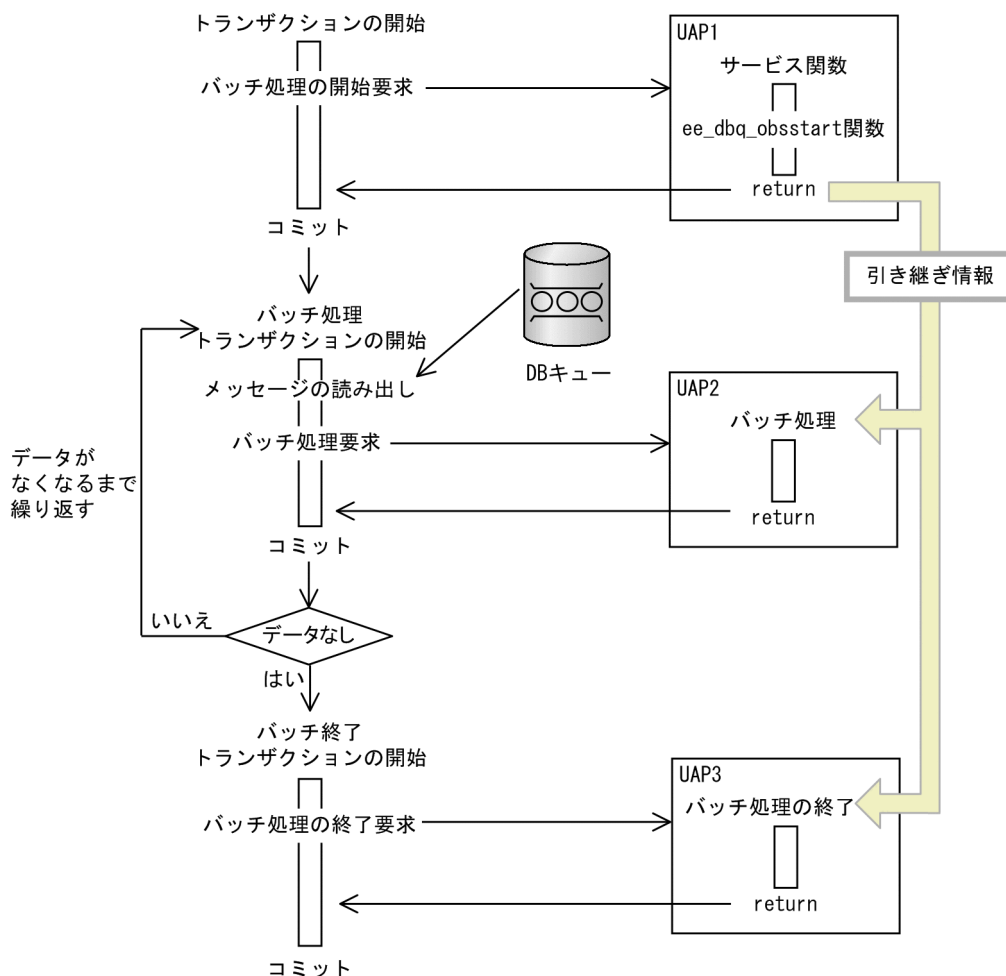
データ型オンラインバッチ処理では、DB キューに登録されている大量のユーザデータをロット単位で UAP に引き渡してバッチ処理を行うことによって、データ処理を管理できます。ロットは、トランザクション起動単位で 1 件ずつ UAP に引き渡されます。

バッチ処理は、任意のトランザクションから ee_dbq_obsstart 関数を呼び出すことによって開始します。バッチ処理の開始要求をしたトランザクションのコミットを契機に、バッチ処理トランザクションを起動します。処理するデータがなくなると、バッチ終了トランザクションを起動してバッチ処理を終了します。

バッチ処理の開始要求をしたトランザクションから、バッチ処理トランザクションおよびバッチ終了トランザクションに、任意のデータを引き継ぐことができます。この任意のデータを引き継ぎ情報と呼びます。引き継ぎ情報は、ee_dbq_obsstart 関数で指定し、ee_dbq_obschdata 関数【CBLEEDBQ('OBSCDT')】で更新できます。引き継ぎ情報を更新できるのは、バッチ処理トランザクションの実行中です。

データ型オンラインバッチ処理の流れを次の図に示します。

図 2-36 データ型オンラインバッチ処理の流れ



(2) イベント型オンラインバッチ処理

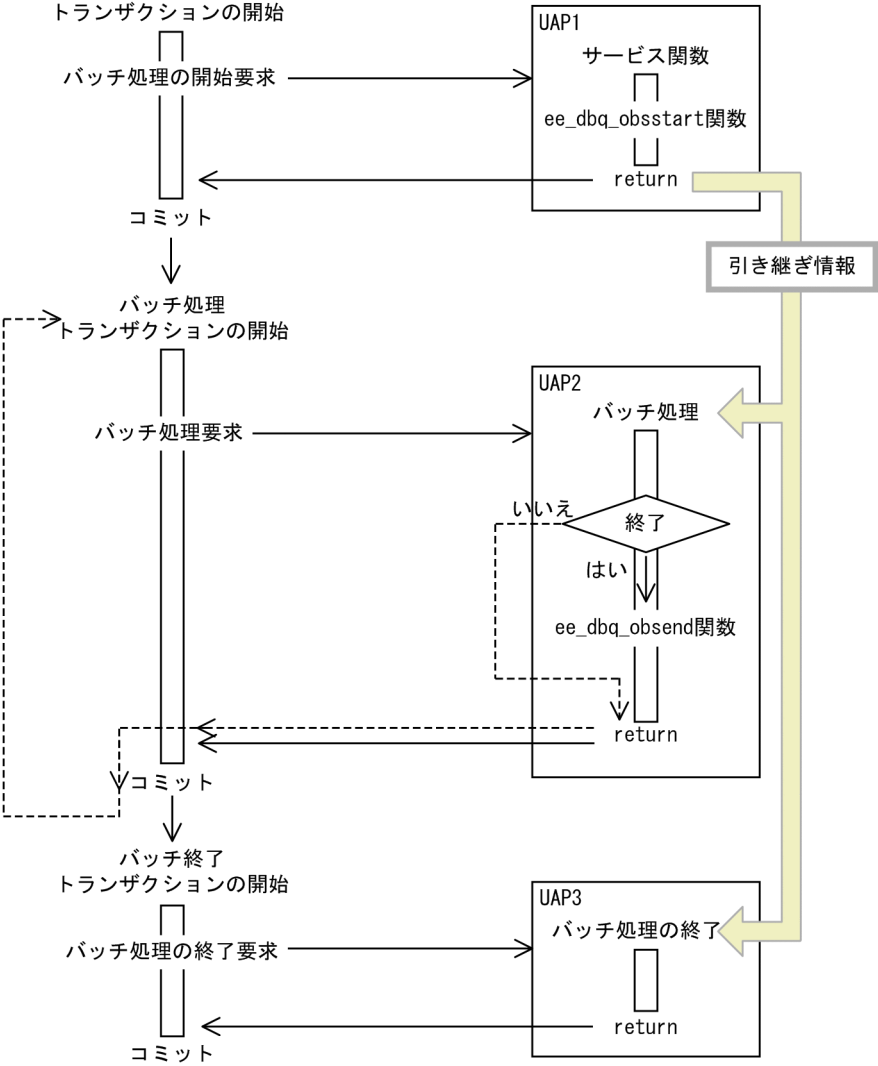
イベント型オンラインバッチ処理では、大量のユーザデータに対して任意の処理を行うことによって、データ処理のスケジュールを管理できます。

バッチ処理は、任意のトランザクションから ee_dbq_obsstart 関数を呼び出すことによって開始します。バッチ処理の開始要求をしたトランザクションのコミットを契機に、バッチ処理トランザクションを起動します。バッチ処理トランザクションから ee_dbq_obsend 関数【CBLEEDBQ('OSEND')】を呼び出すことによって、任意のタイミングでバッチ処理を終了できます。

バッチ処理の開始要求をしたトランザクションから、バッチ処理トランザクションおよびバッチ終了トランザクションに、任意のデータを引き継ぐことができます。この任意のデータを引き継ぎ情報と呼びます。引き継ぎ情報は、ee_dbq_obsstart 関数で指定し、ee_dbq_obschdata 関数で更新できます。引き継ぎ情報を更新できるのは、バッチ処理トランザクションの実行中です。

イベント型オンラインバッチ処理の流れを次の図に示します。

図 2-37 イベント型オンラインバッチ処理の流れ



2.5.3 オンラインバッチ機能で実行できる処理

オンラインバッチ機能で実行できる主な処理の一覧を次の表に示します。

表 2-5 オンラインバッチ機能で実行できる処理

処理	内容
バッチ処理の開始	任意のトランザクションから ee_dbq_obsstart 関数を呼び出すことによって、ロットのバッチ処理の開始要求をします。 バッチ処理の開始要求をしたトランザクションのコミットを契機に、バッチ処理トランザクションを起動します。
バッチ処理の終了	イベント型オンラインバッチ処理のバッチ処理トランザクションから ee_dbq_obsend 関数を呼び出すことによって、ロットのバッチ処理の終了要求をします。 バッチ処理の終了要求は、バッチ処理トランザクションの起動対象ロットに対してできます。

処理	内容
バッチ処理の終了	バッチ処理の終了要求をしたトランザクションのコミットを契機に、正常終了のバッチ終了トランザクションを起動します。
バッチ処理の中断	バッチ処理トランザクションから ee_dbq_obsstop 関数【CBLEEDBQ('OBSSTOP ')】を呼び出すことによって、ロットのバッチ処理の中断要求をします。 バッチ処理の中断要求は、バッチ処理トランザクションの起動対象ロットに対してできます。 バッチ処理の中断要求をしたトランザクションのコミットを契機に、中断のバッチ終了トランザクションを起動します。
バッチ処理の強制終了	バッチ処理トランザクション以外の任意のトランザクションから ee_dbq_obscancel 関数【CBLEEDBQ('OBSCAN ')】を呼び出すことによって、中断中のバッチ処理の強制終了要求をします。 バッチ処理の強制終了要求をしたトランザクションのコミットを契機に、強制終了のバッチ終了トランザクションを起動します。
ロールバック要求時のメッセージの読み出しスキップ	データ型オンラインバッチ処理のバッチ処理トランザクションのロールバック要求時に、ee_dbq_obsrbskip 関数【CBLEEDBQ('OBSRBSP ')】を呼び出すことによって、読み出したメッセージのロールバック処理をしないで、読み出し済みにします。 メッセージの読み出しスキップ要求は、バッチ処理トランザクションの起動対象ロットに対してできます。 次に起動するバッチ処理トランザクションでは、前回読み出し済みにしたメッセージの次のメッセージを読み出します。

2.6 運用コマンド実行機能

運用コマンド実行機能は、ユーザが TP1/EE プロセスの UAP 中で ee_adm_call_command 【CBLEEADM('COMMAND ')】関数を呼び出すことによって運用コマンドを実行することを可能にします。ee_adm_call_command 関数は、処理結果を UAP に返します。

この機能で実行できる運用コマンドは、ユーザが作成したコマンド、TP1/Server Base のコマンド、TP1/EE のコマンド、および XTC のコマンドです。ただし、TP1/Server Base のコマンド、TP1/EE のコマンド、および XTC のコマンドには、この機能で実行できないコマンドが一部あります。また、TP1/EE のコマンドおよび XTC のコマンドのうち、コマンドの処理と同等の処理を関数で提供しているものに関しては運用コマンド実行機能を使用して実行することはお勧めしません。

この機能を使用できるトランザクション種別については、以降の表を参照してください。また、トランザクション種別の詳細については、「付録 A トランザクション種別」を参照してください。

TP1/EE のコマンドおよび XTC のコマンドは、使用するトランザクションによって実行可否が異なります。運用コマンド実行機能での、TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否を次の表に示します。

表 2-6 運用コマンド実行機能での TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否

コマンド	トランザクション種別															
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	JI	C U	UT
eeactsv	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eeadmccsetup	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eeapls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eechgap	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eedbqactsv	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqaltlsh	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eedbqaltput	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqchgr	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqclr	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqclrh	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eedbqcnct	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqdctsv	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqinhibit	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqislt	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○

コマンド	トランザクション種別															
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	JI	C U	UT
eedbqls	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqlscs	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqlsdq	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqlsqg	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqlssv	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqrels	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqrst	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqskip	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqstop	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedbqtblh	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eedbqtblo	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eedbqwtpr	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eedctsv	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eedefchk	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eefilbkup	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilchgrp	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilchmod	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilchown	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefills	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilmkfs	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilrstr	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eefilstatfs	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eelckls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eelspce	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eelsque	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eelssv	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eememls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eenamupdate	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eerasget	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×

コマンド	トランザクション種別															
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	JI	C U	UT
eerpciscn	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eerpcnamdel	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eerpcrapcls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eerpcrapls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eerpcsockcls	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eerpcspndls	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eerpcspndrles	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eerpcstat	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eerpctcps	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eesgrpctl	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eestsclose	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsfills	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsinit	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsls	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsopen	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsrn	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eestsswap	×	×	○	○	○	○	○	○	○	○	×	○	○	×	○	○
eesvstart	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eesvstop*	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrblinedump	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrblined	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbrcvr	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbreport	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbstced	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbstcese	○	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbstcfput	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbstcse	○	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbtaskdump	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbtaskd	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

コマンド	トランザクション種別															
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	JI	C U	UT
eetrbtaskfput	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbtbldump	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbtrcefput	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbuapdump	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbuated	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbuatfput	×	×	○	○	○	○	○	○	○	○	○	○	○	×	○	○
eetrbwtor	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
eetrmcmt	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrmfgt	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrmldump	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrmlsru	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eetrmistr	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrmnkobj	○	○	○	○	○	○	○	○	○	○	×	○	○	○	○	○
eetmrnbk	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
eetmntim	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

(凡例)

- ：実行できます。
- ×

注※

eessvstop コマンドを実行する場合は、コマンド文字列の最後に「&」を付加して、バックグラウンドで実行してください。

運用コマンド実行機能での、XTC コマンドの自 TP1/EE プロセスに対する実行可否を次の表に示します。

表 2-7 運用コマンド実行機能での XTC コマンドの自 TP1/EE プロセスに対する実行可否

コマンド	トランザクション種別												
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
eehamls	○	○	○	○	○	○	○	○	○	○	○	○	○
eelspcenum	○	○	○	○	○	○	○	○	○	○	○	○	○
eemchotqact	○	○	○	○	○	○	○	○	○	○	○	○	○
eemchotqdct	○	○	○	○	○	○	○	○	○	○	○	○	○
eemchotqend	×	×	×	×	×	×	×	×	×	×	×	×	×

コマンド	トランザクション種別												
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
eemchotqls	○	○	○	○	○	○	○	○	○	○	○	○	○
eemchotqskip	○	○	○	○	○	○	○	○	○	○	○	○	○
eepcerefer	×	×	×	×	×	×	×	×	×	×	×	×	×
eepceskip	○	○	○	○	○	○	○	○	○	○	○	○	○
eetrbqueed	○	○	○	○	○	○	○	○	○	○	○	○	○

(凡例)

- ：実行できます。
- ×

TP1/Server Base のコマンドのうち、TP1/Server Base の dc_adm_call_command 関数で実行できないコマンドは、運用コマンド実行機能でも実行できません。詳細については、マニュアル「OpenTP1 プログラム作成の手引」の dc_adm_call_command 関数で実行できるコマンドの説明を参照してください。また、次のコマンドは TP1/Server Base の dc_adm_call_command 関数では実行できますが、運用コマンド実行機能では実行できません。

- scdhold
- scdrles
- scdchprc
- scdrsprc

2.6.1 運用コマンド実行機能の環境設定

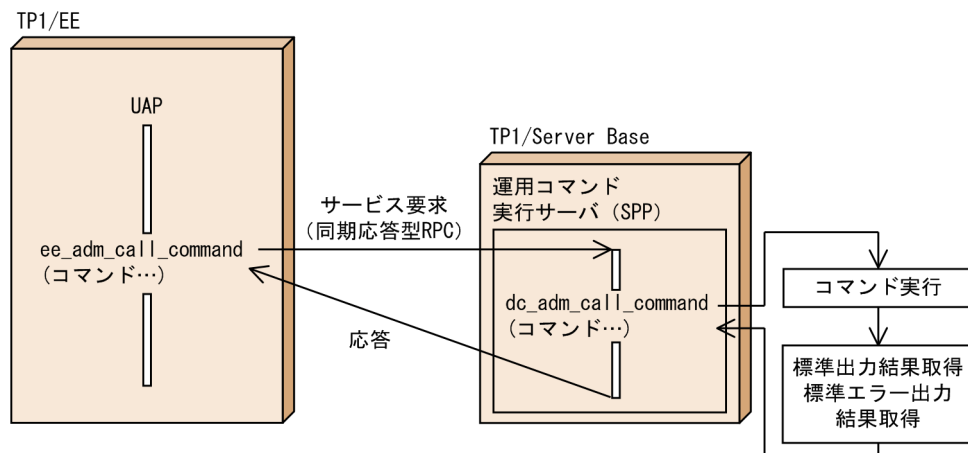
TP1/EE をインストール後、運用コマンド実行機能を使用する前に eeadmccsetup コマンドを実行して、この機能を使用するための環境設定をします。eeadmccsetup コマンドは、初めて運用コマンド実行機能を使用する前に一度だけ実行してください。

2.6.2 運用コマンド実行機能の処理

運用コマンド実行機能を使用すると、あらかじめ起動された TP1/EE とは別の SPP（運用コマンド実行サーバ）に対して同期型 RPC でサービスが要求されます。運用コマンド実行サーバは、コマンドを実行し、処理結果を UAP に返します。

運用コマンド実行機能の処理概要を次の図に示します。

図 2-38 運用コマンド実行機能の処理概要



運用コマンド実行機能は、運用コマンド実行サーバに対して同期型 RPC でサービスを要求するため、コマンド実行の処理結果が UAP に返されます。ただし、&をコマンド文字列の最後に付加したコマンドを、この機能で実行する場合は、バックグラウンドで処理を実行します。バックグラウンドで処理を実行した場合は、コマンドを実行した処理結果が返される前に、関数の処理結果が UAP に返されるため、コマンドを実行した処理結果を UAP が受け取れません。

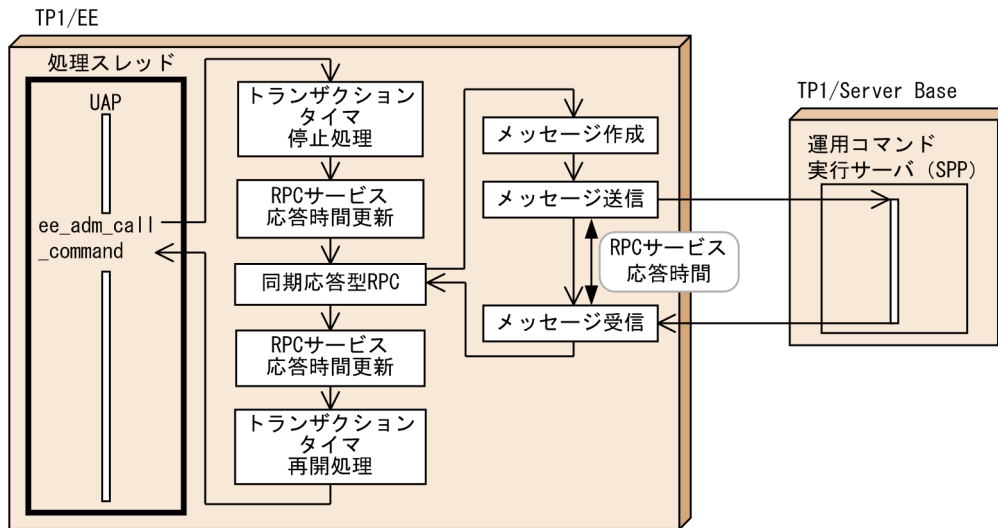
2.6.3 運用コマンド実行機能の時間監視

運用コマンド実行機能の応答時間は、運用コマンド実行サーバに対して同期型 RPC を使用して要求したサービスに対するサービス応答時間と、ほぼ同じです。

運用コマンド実行機能は、ee_adm_call_command 関数の time に指定された監視時間を、サービス要求の応答待ち時間に設定します。サービス要求の応答を運用コマンド実行サーバから受け取ると、サービス要求の応答待ち時間を変更前の値に戻します。サービス要求の応答待ち時間の変更前後に、トランザクションタイマの停止処理および再開処理をします。

運用コマンド実行機能の時間監視による、サービス要求の応答待ち時間の変更の概要を、次の図に示します。

図 2-39 運用コマンド実行機能の時間監視によるサービス要求の応答待ち時間の変更



運用コマンド実行機能の時間監視でタイムアウトが発生した場合は、`ee_adm_call_command` 関数はエラーリターンします。ただし、運用コマンド実行サーバは、コマンドの処理を続行します。運用コマンド実行サーバは、TP1/Server Base のユーザサービス定義の `service_expiration_time` オペランドで指定した時間が経過すると、プロセスを強制停止します。その後、運用コマンド実行サーバは、再起動され、新しいサービス要求を受け付けられるようになります。ただし、サーバプロセスが停止しても、コマンドプロセスは消滅しません。コマンドプロセスは、`kill` コマンドなどで削除してください。

2.6.4 運用コマンド実行サーバ

運用コマンド実行サーバは、運用コマンド実行機能でコマンドの処理を実行する SPP です。運用コマンド実行サーバは、TP1/Server Base と 1 対 1 の関係です。同一の TP1/Server Base にある複数の TP1/EE で運用コマンド実行機能を使用した場合でも、一つの運用コマンド実行サーバで対応します。運用コマンド実行サーバの仕様を次に示します。

サーバ名：`eeadmspp`

実行形式プログラム名：`eeadmspp`

サービスグループ名：ユーザ指定

サービス名：ユーザ指定

エントリポイント名：`ee_call_cmd`

(1) 運用コマンド実行機能の環境設定

TP1/EE をインストールしてから運用コマンド実行機能を使用するまでに、次のコマンドを一度だけ実行して、運用コマンド実行機能の環境設定をする必要があります。

(2) 運用コマンド実行サーバの開始と終了

運用コマンド実行サーバは、運用コマンド実行機能を使用する前に、TP1/Server Base の dcsvstart コマンドを次のように指定して起動してください。

```
dcsvstart -u eeadmspp
```

運用コマンド実行サーバを終了するには、運用コマンド実行機能を使用する UAP がサービス要求を完了したあとに、TP1/Server Base の dcsvstop コマンドを次のように指定して実行してください。

```
dcsvstop eeadmspp
```

dcsvstart コマンドおよび dcsvstop コマンドの詳細については、マニュアル「OpenTP1 運用と操作」を参照してください。

2.6.5 運用コマンド実行機能使用時の定義

運用コマンド実行機能を使用する場合に、運用コマンド実行サーバに指定する必要がある TP1/Server Base のユーザサービス定義および TP1/EE のシステムサービス定義を示します。

(1) TP1/Server Base のユーザサービス定義

運用コマンド実行サーバに指定する必要がある TP1/Server Base のユーザサービス定義のオペランドを次の表に示します。次の表で示すオペランドの詳細については、マニュアル「OpenTP1 システム定義」のユーザサービス定義の説明を参照してください。

なお、運用コマンド実行サーバ特有の説明があるオペランドについては、表のあとに説明します。

表 2-8 運用コマンド実行サーバの TP1/Server Base のユーザサービス定義

形式	オペランド	定義内容	指定値
set	service_group	サービスグループ名	〈必須〉
	module	このサービスグループを実行する実行形式プログラム名	eeadmssp ^{*1}
	service	このサービスグループに属するサービス名とそのサービスを提供するエントリポイント名の組	〈必須〉 =ee_call_cmd ^{*2}
	nice	プロセスの優先順位の変更	0 ^{*1}
	parallel_count	常駐プロセス数と最大プロセス数	1 ^{*3}
	hold	UAP が異常終了した場合にサービスグループまたはサービスを閉塞するかどうかを指定	N ^{*1}

形式	オペランド	定義内容	指定値
set	hold_recovery	サービスグループまたはサービスの閉塞状態をプロセスの回復時に引き継ぐかどうかを指定	N※1
	schedule_priority	スケジュールの優先順位	8※1
	message_buflen	最大メッセージ長	3000※1
	message_store_buflen	メッセージ格納プール長	30000※3
	atomic_update	サービスをトランザクションとして実行するかどうかを指定	N※1
	receive_from	スケジュールキュー, または UNIX ドメインもしくはインターネットドメインを使用するかどうかを指定	queue※1
	uap_trace_max	UAP トレース格納最大数	32※1
	term_watch_time	連続異常終了限界経過時間	30※3
	type	このサービスグループの種別	other※1
	balance_count	1 プロセスが処理するサービス要求数	3※3
	uid	ユーザ識別子	〈デフォルト〉
	auto_restart	UAP が異常終了した場合のサービスグループの処置を指定	Y※1
	critical	UAP が異常終了した場合のシステムの処置	N※1
	service_hold	サービス単位に閉塞管理するかどうかを指定	N※1
	service_priority_control	サービス要求単位の優先順位に従ってスケジュールするかどうかを指定	N※1
	trf_put	トランザクションが出力したジャーナルをトランザクションリカバリジャーナルファイルへ出力するかどうかを指定	N※1
	trn_statistics_item	トランザクションブランチの統計情報項目	nothing※1
	node_down_restart	ユーザサーバを自動起動するかどうかを指定	Y※1
	rpc_response_statistics	レスポンス統計情報を取得するかどうかを指定	N※1
	server_type	サービス関数呼び出し時 OpenTP1 のパラダイムに従うか XATMI のパラダイムに従うか指定	betran※1
	purge_msgget	非常駐サーバでサーバプロセスがない場合 OS のメッセージキューを解放するかどうか指定	N※1
	cancel_normal_terminate	dcsvstop コマンドによる正常終了を抑止するかどうかを指定	N※1
	prc_abort_signal	サーバのアボート用シグナル番号	3※3
	rpc_extend_function	RPC サービスの機能拡張レベル	00000000※1
	max_socket_descriptors	ソケット用ファイル記述子の最大数	100※3

形式	オペランド	定義内容	指定値
set	max_open_fds	UAP プロセスでアクセスするファイルおよびパイプの最大数	100※3
	termed_after_service	サービス終了時非常駐プロセス終了させるかどうか	N※3
	rpc_trace	RPC トレースを取得するかどうかを指定	N※1
	trn_rollback_information_put	ロールバック要因に関する情報をログに取得するかどうか	no※1
	schedule_method	ユーザサーバのスケジューリング方式	msgque※1
	adm_message_option	メッセージ出力指定	F※3
	status_change_when_termining	次回再開始時に最終的な状態の変化を反映するかどうかを指定	Y※1
	service_expiration_time	サービス関数開始から終了までの実行監視時間	60※3
	multi_schedule	マルチスケジューラ機能を使用してスケジューリングするかどうかを指定	N※3
	make_queue_on_starting	非常駐のサービスグループに割り当てるスケジュールキューをサーバ起動時に割り当てるかどうかを指定	Y※3
	ipc_sockctl_highwater	ソケットの一時クローズ処理開始数パーセンテージ	100,0※3
	ipc_sockctl_watchtime	ソケット再利用可能監視時間	60※3
	ipc_conn_interval	コネクション確立監視時間	60※3
	ipc_send_interval	データ送信監視間隔	5※3
	ipc_send_count	データ送信監視回数	5※3
	ipc_header_rcv_time	通信制御データの受信監視時間	10※3
	rpc_send_retry_count	TCP/IP コネクションの接続時にエラーが発生した場合のトライ回数	0※3
	rpc_send_retry_interval	TCP/IP コネクションの接続時にエラーが発生した場合のトライ間隔	0※3
	thdlock_sleep_time	スレッド間で排他が競合した場合のスレッドの待ち時間	15※3
	ipc_rcvbuf_size	TCP/IP の受信バッファサイズ	8192※3
	ipc_sndbuf_size	TCP/IP の送信バッファサイズ	8192※3
	polling_control_data	一時クローズ処理要求が到着していないかどうかを検査	Y※1
	thread_yield_interval	ソケットの再利用指示を受信できる契機を与えるインタバル時間	60※3
	ipc_backlog_count	コネクション確立要求を格納するキューの長さ	0※3
	rpc_buffer_pool_max	プーリングするバッファ数	64※3

形式	オペランド	定義内容	指定値
set	schedule_delay_limit	スケジュール遅延限界経過時間	0※1
	core_shm_suppress	コアファイルへの共用メモリダンプの出力を抑止するかどうか	N※1
dcputenv	PATH	パス	〈デフォルト〉
	LIBPATH※4	ライブラリパス	\$DCDIR/lib※3

(凡例)

〈必須〉：任意の値を指定できるオペランドです。このオペランドは省略できません。

〈デフォルト〉：任意の値を指定できるオペランドです。このオペランドの指定は省略できます。省略した場合は、デフォルト値が設定されます。

注※1

固定の値を指定するオペランドです。指定値の値を指定してください。このオペランドは省略できません。

注※2

サービス名には任意の値を指定してください。エントリポイント名には ee_call_cmd を指定してください。このオペランドは省略できません。

注※3

任意の値を指定できるオペランドです。表中で示している指定値の値はサンプルです。このオペランドを省略すると、TP1/Server Base のユーザサービスデフォルト定義の該当オペランドの指定値が有効になります。

注※4

OS が Linux の場合は、LD_LIBRARY_PATH です。

(a) 運用コマンド実行サーバ特有の説明があるオペランド

表 2-8 中で、運用コマンド実行サーバ特有の説明があるオペランドを、次に示します。

●service_group="サービスグループ名"

運用コマンド実行サーバのサービスグループ名を指定します。ネットワークで接続されるすべての OpenTP1 システムの中で一意になるように指定してください。

●service="サービス名=エントリポイント名"

運用コマンド実行サーバのサービス名およびエントリポイント名を指定します。エントリポイント名には、ee_call_cmd を指定してください。

●max_socket_descriptors=ソケット用ファイル記述子の最大数

OpenTP1 制御下のプロセスで、ソケット用に使用するファイル記述子の最大値を指定します。運用コマンド実行サーバで指定するソケット用ファイル記述子の最大数の計算式を次に示します。

$\uparrow \text{ (運用コマンド実行サーバにサービス要求する TP1/EE の数} + 1 + \text{システムサービスプロセス数)} / 0.8 \uparrow$

↑↑：小数点以下を切り上げます。

●max_open_fds=UAP プロセスでアクセスするファイルおよびパイプの最大数

サーバのプロセスでアクセスするファイルおよびパイプの最大数を指定します。サーバのプロセスでアクセスするファイルおよびパイプの最大数の計算式を、次に示します。

- service_expiration_time=サービス関数開始から終了までの実行監視時間
SPP プロセスでの、サービス関数開始から終了までの実行監視時間を指定します。
ee_adm_call_command 関数の time で指定する応答監視時間より大きい値を指定してください。また、運用コマンド実行サーバにサービス要求する TP1/EE のプロセス関連定義の call_cmd_time オペランドの指定値より大きい値を指定してください。
- ipc_rcvbuf_size=TCP/IP の受信バッファサイズ
コネクションごとに確保される TCP/IP の受信バッファのサイズを指定します。運用コマンド実行サーバにサービス要求する TP1/EE の RPC 関連定義の ipc_sendbuf_size オペランドの指定値と同じ値を指定することをお勧めします。
- ipc_sendbuf_size=TCP/IP の送信バッファサイズ
コネクションごとに確保される TCP/IP の送信バッファのサイズを指定します。運用コマンド実行サーバにサービス要求する TP1/EE の RPC 関連定義の ipc_rcvbuf_size オペランドと同じ値を指定することをお勧めします。
- PATH△運用コマンド実行機能で実行するコマンドの格納ディレクトリ
運用コマンド実行機能で実行するコマンドの格納ディレクトリを指定します。運用コマンド実行機能の実行環境で、実行したいコマンドのパスが設定されていない場合は、このオペランドで該当コマンドの格納ディレクトリのパスを追加してください。
- LIBPATH△OpenTP1 のライブラリパス:TP1/EE のライブラリパス
OpenTP1 のライブラリパスと TP1/EE のライブラリパスを指定します。

(2) TP1/EE のシステムサービス定義

運用コマンド実行機能を使用する場合に指定する必要がある TP1/EE サービス定義のオペランドを、次に示します。

プロセス関連定義

- call_cmd_srvgroup
- call_cmd_service
- call_cmd_time

サービスグループ情報関連定義

- eesvgdef※

注※

運用コマンド実行機能を使用する RPC 通信では、ネームサービスを使用しないため、サービスグループ情報関連定義の eesvgdef 定義コマンドを指定する必要があります。

オペランドの詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」を参照してください。

2.7 エラートランザクション

メッセージ送受信の処理などでエラーや障害が起こった場合、TP1/EE はエラートランザクションを起動します。TP1/EE が起動するエラートランザクションを次の表に示します。

表 2-9 エラートランザクション

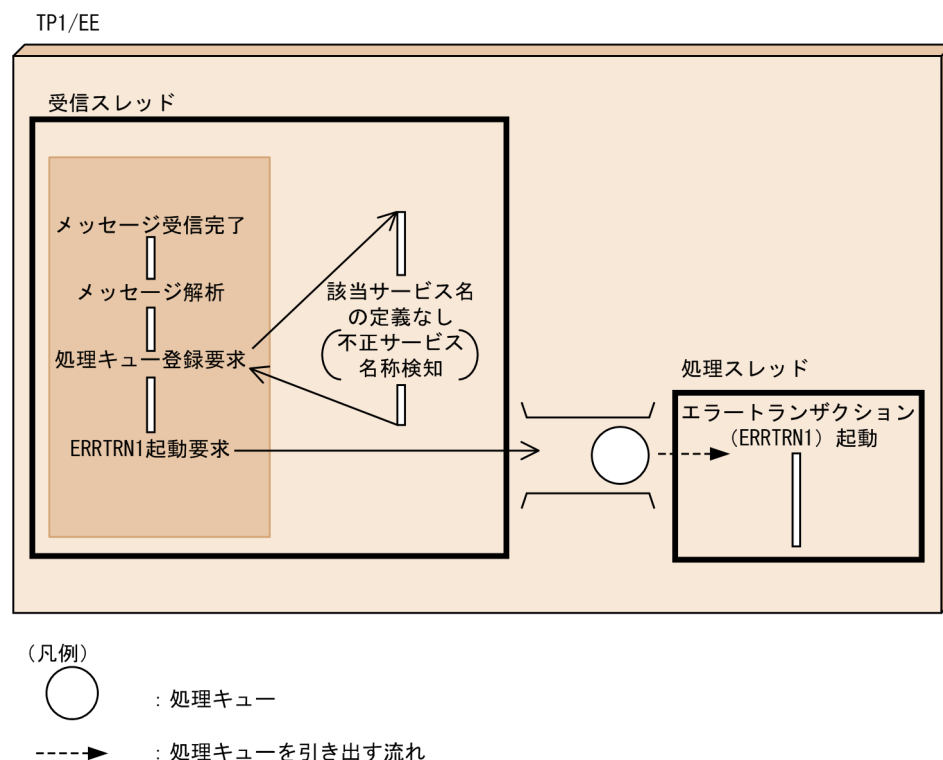
エラーの種類	トランザクション名	エラートランザクションが起動された原因
不正サービス名称検出通知	ERRTRN1	受信したメッセージに設定されたサービス名は定義されていません。
サービス閉塞通知	ERRTRN2	サービス閉塞中のため、トランザクションを起動できません。
UAP 異常終了通知	ERRTRN3	UAP が、メッセージを受信したあとに異常終了しました。
タイマトランザクション起動破棄通知	ERRTRN4	システム終了、または該当サービスが閉塞中のため、タイマトランザクションの起動を破棄しました。
コミット決着失敗通知	ERRTRNR	UAP リターン後のトランザクション同期点処理で、コミット決着できませんでした。
送信障害通知	ERRTRNS	メッセージを送信する途中で通信プロトコル障害が発生しました。

2.7.1 不正サービス名称検出通知（ERRTRN1）

ERRTRN1 は、受信したメッセージに設定されたサービス名がユーザサービス関連定義の service オペランドに定義されていない場合に起動されます。

ERRTRN1 の概要を次の図に示します。

図 2-40 ERRTRN1 の概要



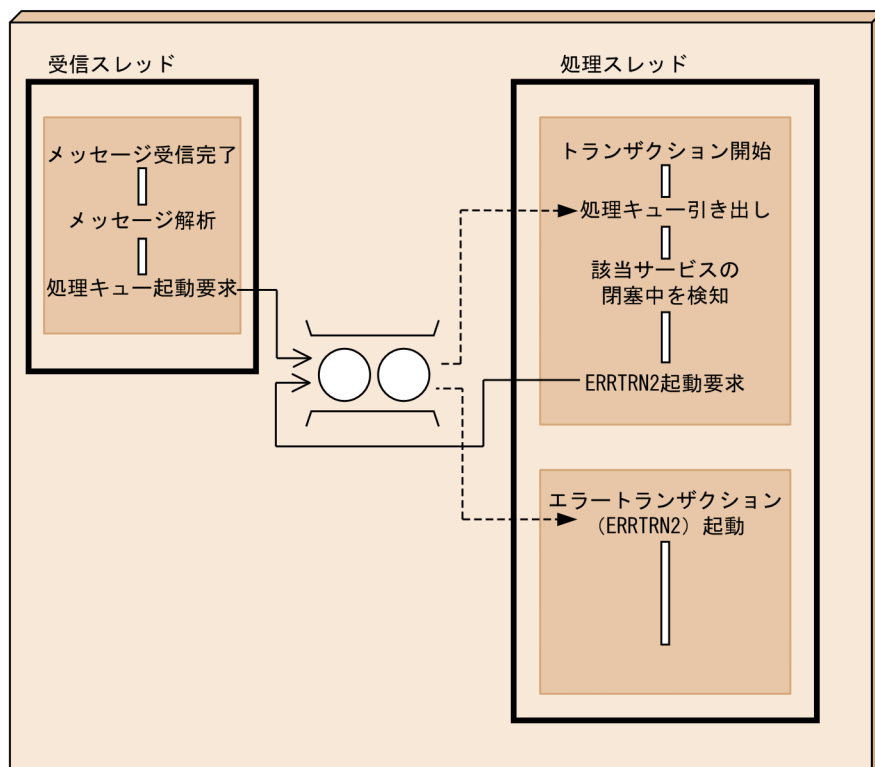
2.7.2 サービス閉塞通知 (ERRTRN2)

ERRTRN2 は、受信したメッセージに設定されたサービス名がユーザサービス関連定義の service オペランドに定義されているが、サービス閉塞中の場合に起動されます。

ERRTRN2 の概要を次の図に示します。

図 2-41 ERRTRN2 の概要

TP1/EE



(凡例)



: 処理キュー



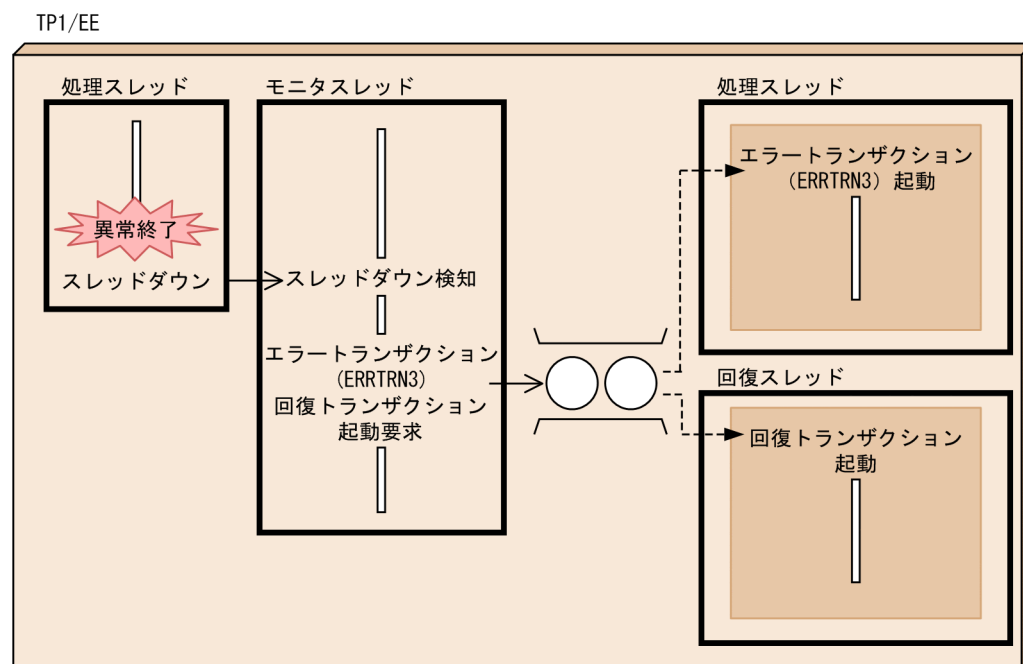
: 処理キューを引き出す流れ

2.7.3 UAP 異常終了通知 (ERRTRN3)

ERRTRN3 は、UAP が異常終了してスレッドダウンした場合に起動されます。

ERRTRN3 の概要を次の図に示します。

図 2-42 ERRTRN3 の概要



(凡例)



: 処理キュー

---▶ : 処理キューを引き出す流れ

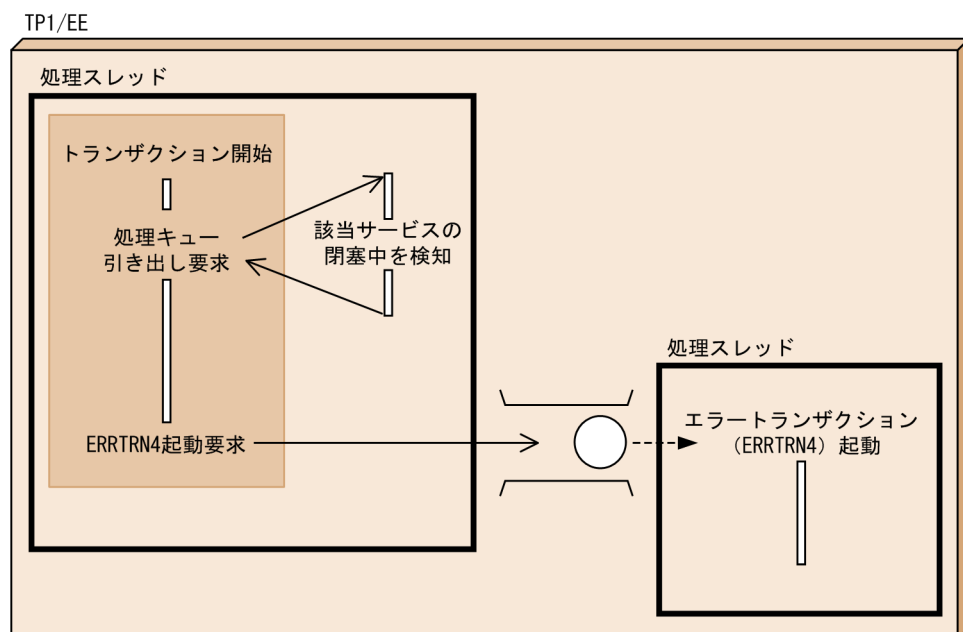
2.7.4 タイマトランザクション起動破棄通知 (ERRTRN4)

ERRTRN4 は、次に示す場合に起動されます。

- サービス閉塞によるタイマトランザクション起動の破棄が発生した場合
- システム終了によるタイマトランザクション起動の破棄が発生した場合

ERRTRN4 の概要を次の図に示します。

図 2-43 ERRTRN4 の概要



(凡例)



: 処理キュー



: 処理キューを引き出す流れ

登録され、起動待ちのタイマトランザクションが存在するタイミングで、システム終了要求された場合は、ERRTRN4 を即時起動します。ERRTRN4 の起動は、プロセス終了形態が、正常終了または計画停止 A の場合だけです。それ以外の場合は、ERRTRN4 を起動しません。

2.7.5 コミット決着失敗通知 (ERRTRNR)

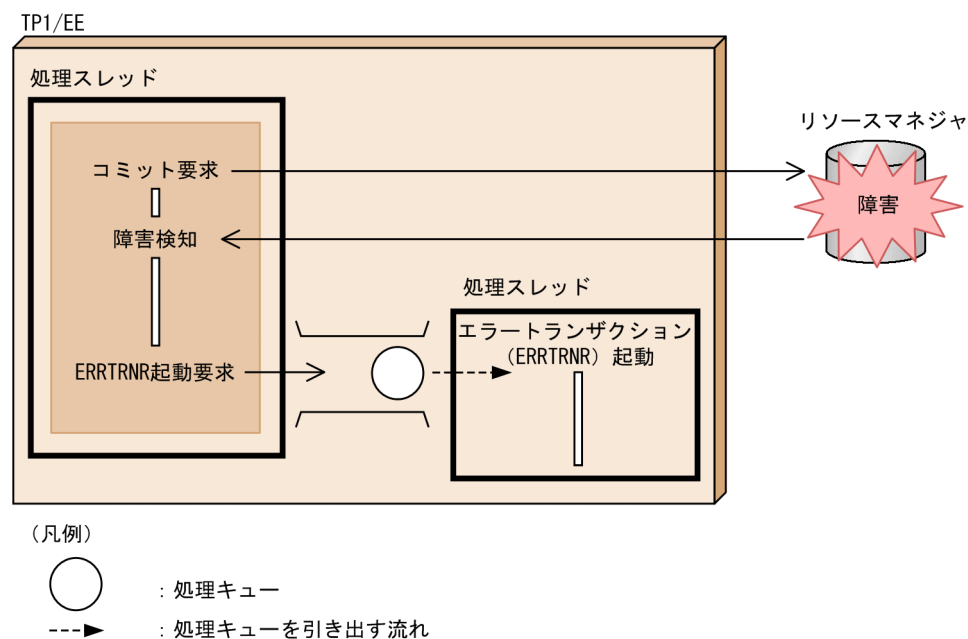
ERRTRNR は、UAP からのリターン後の同期点処理で、リソースマネージャ障害や通信障害などが原因でトランザクションの決着ができなかった場合に起動されます。

ERRTRNR が起動される原因となるトランザクションを次に示します。

- サービストランザクション
- タイマトランザクション
- ユーザコマンドトランザクション
- 再起動トランザクション（登録元が上記トランザクションの場合だけ）

ERRTRNR の概要を次の図に示します。

図 2-44 ERRTRNR の概要



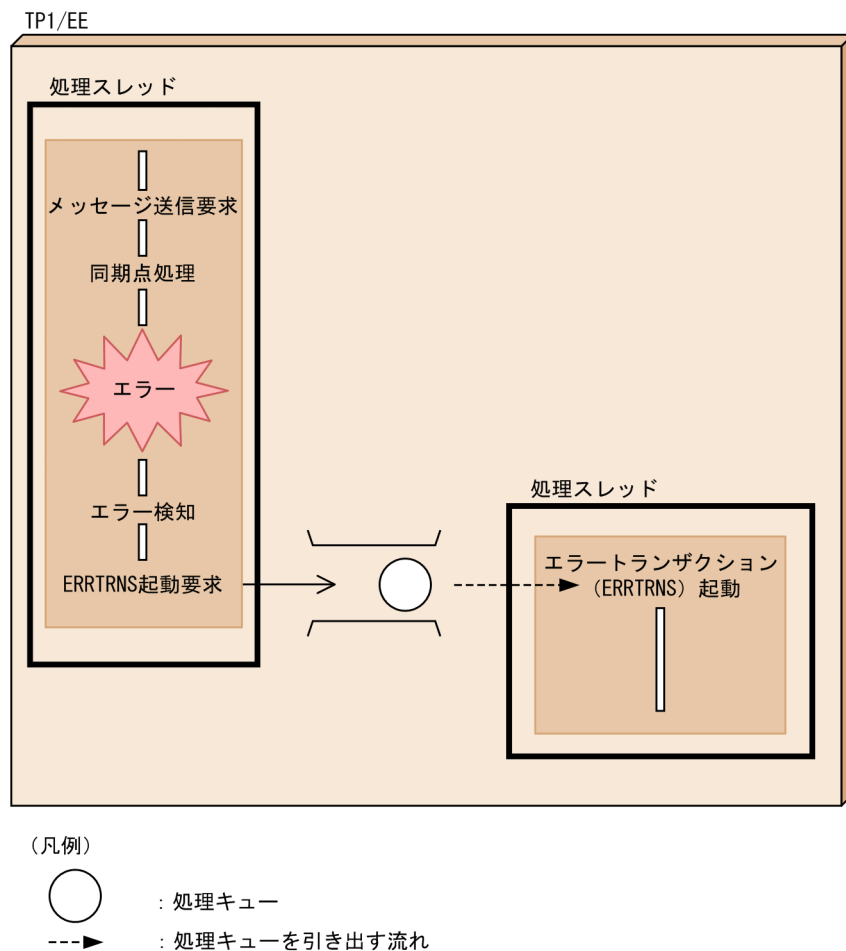
2.7.6 送信障害通知 (ERRTRNS)

ERRTRNS は、メッセージ送信（コミット同期送信）時に通信障害を検知した場合に起動されます。

同期点後のメッセージ送信要求時に、通信障害を検知した場合、ERRTRNS を起動します。

ERRTRNS の概要を次の図に示します。

図 2-45 ERRTRNS の概要



2.7.7 エラートランザクションの登録に失敗した場合の後処理

エラートランザクションの登録時に、エラーが発生した場合は、メッセージを出力して処理キューの登録を破棄します。ただし、オペランドを省略した（エントリポイントを指定していない）ERRTRN2, ERRTRN4 および ERRTRNR の登録時に、エラーが発生した場合は、メッセージを出力しません。

2.8 リモート API 機能

リモート API 機能は、クライアント側が呼び出した API をサーバ側に転送して、サーバ側で代理実行する機能です。代理実行を要求するクライアント側を rap クライアント、代理実行の要求を受けるサーバ側を rap サーバと呼びます。また、rap クライアント、rap サーバのプロトコルを rap プロトコルと呼びます。

リモート API 機能を使用する場合、TP1/EE は、rap クライアントとしても、rap サーバとしても動作します。

リモート API 機能で代理実行できる API を次に示します。

- ee_rpc_call 関数
- dc_rpc_call 関数

TP1/EE が rap サーバとして動作する場合だけ使用できます。TP1/EE が rap クライアントとして動作する場合は使用できません。

rap サーバとして動作する TP1/EE は、サポートしていない API の代理実行を要求された場合、rap クライアントに代理実行エラー応答メッセージを返します。

なお、TP1/Client を使用する場合は、マニュアル「OpenTP1 クライアント使用の手引 TP1/Client/W, TP1/Client/P 編」またはマニュアル「OpenTP1 クライアント使用の手引 TP1/Client/J 編」を参照してください。

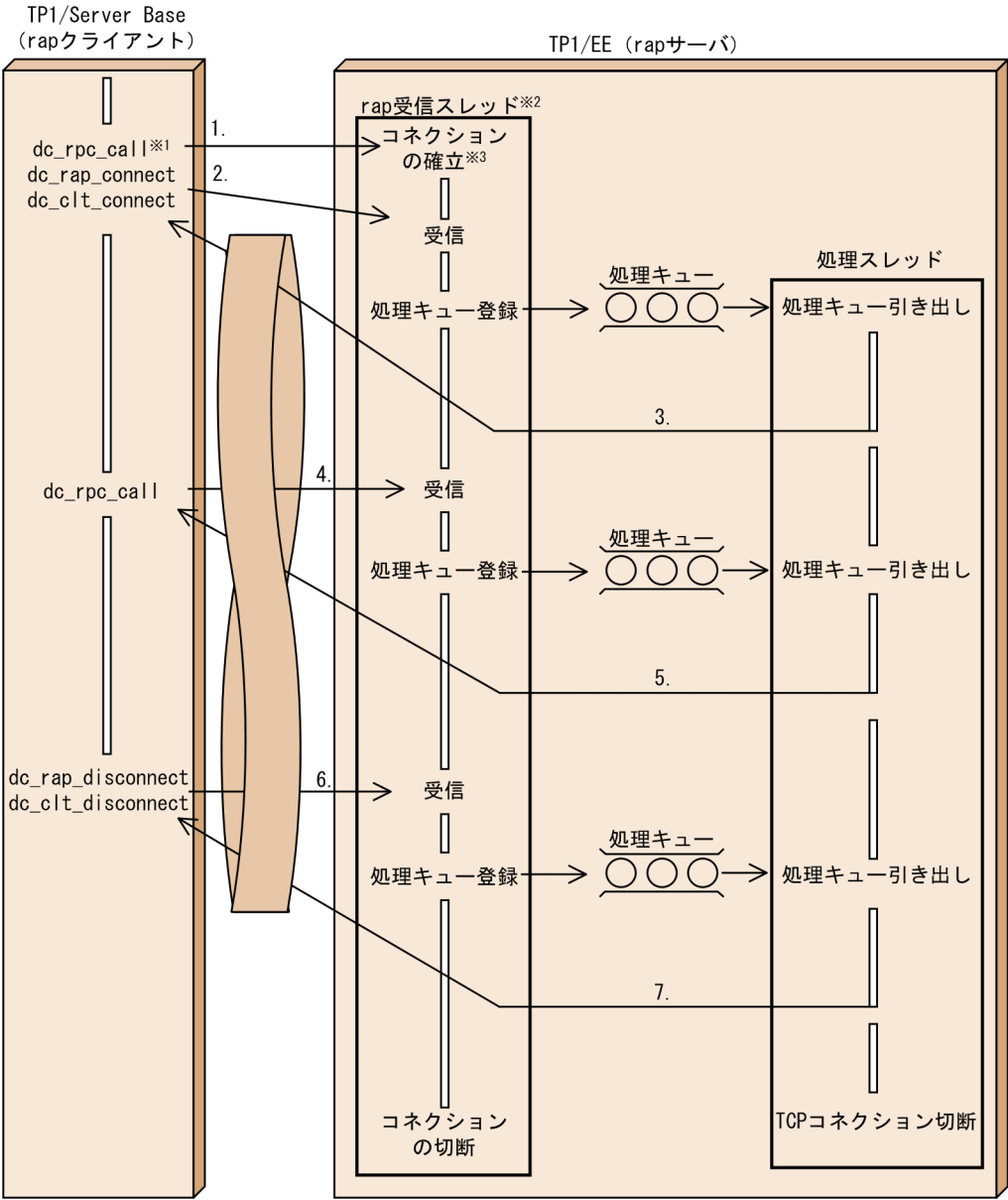
rap クライアントと rap サーバは、論理的な通信路である**常設コネクション**を使用して、リモート API のメッセージを送受信します。常設コネクションは、トランスポート層以下では TCP コネクション (INET ドメイン) を使用します。TCP コネクション制御の詳細については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」のコネクションに関する運用の説明を参照してください。なお、TP1/EE では、常設コネクションの確立・解放を OpenTP1 が管理します (オートコネクトモード)。

rap プロトコルは、次に示す段階で構成されます。各段階で、rap クライアントと rap サーバとの間で要求メッセージと応答メッセージが送受信されます。

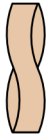
1. 常設コネクションの確立
2. API の代理実行
3. 常設コネクションの解放

TP1/EE が rap サーバとしてリモート API 機能を使用する場合の、コネクション確立からコネクション切断までの流れを、次の図に示します。

図 2-46 リモート API 機能を使用する場合の、コネクション確立からコネクション切断までの流れ



(凡例)



: 常設コネクション

1. TCP コネクション確立要求メッセージ
2. 常設コネクション確立要求メッセージ
3. 常設コネクション確立応答メッセージ
4. API の代理実行要求メッセージ
5. API の代理実行応答メッセージ

6. 常設コネクション解放要求メッセージ

7. 常設コネクション解放応答メッセージ

注※1

常設コネクションがオートコネクトモードのときに、TCP コネクションの確立が要求されます。

注※2

rap 受信スレッドは、プロセスごとに 1 スレッドあります。ソケット API を使用して rap クライアントからのメッセージを受信します。

注※3

TCP コネクション確立要求の受け付けは、オンライン開始（初期化トランザクション終了）からプロセス終了処理開始（終了コマンド受け付け）の間だけ受信できます。

rap プロトコルの段階ごとの処理と、メッセージ受信処理を実行できない場合について説明します。

2.8.1 常設コネクションの確立

図 2-46 で、常設コネクション確立要求受け付けの処理に該当するのは、1.~3.のメッセージ送受信です。

(1) TP1/EE が rap クライアントとして動作する場合の処理

rap クライアントは、該当するサービスグループの rap サーバに対して TCP コネクションを確立して、常設コネクション確立要求メッセージを rap サーバに送信します。

常設コネクションは、次に示すサービスグループに対して ee_rpc_call 関数を呼び出したあと、rap サーバからの常設コネクション確立応答メッセージを受信することによって確立されます。

- 常設コネクションが確立されていないサービスグループ
- 常設コネクション（ほかのスレッドによって使用されている）が確立されているサービスグループ

常設コネクションが確立されると、該当するサービスグループに対する ee_rpc_call 関数は、常設コネクションを経由して rap サーバに送信され、rap サーバ上で代理実行されます。

非同期応答型 RPC は使用できません。非同期応答型 RPC を使用した場合、リモート API 機能ではなく、通常の RPC として動作します。また、トランザクション内からリモート API 機能で ee_rpc_call 関数を呼び出してもトランザクションとして動作しません。

なお、常設コネクション確立要求メッセージを送信したあと、応答待ち時間（RPC 関連定義の rpc_rpc_watch_time オペランドで指定）以内に常設コネクション確立応答メッセージを受信できない場合、ee_rpc_call 関数はエラーリターンします。

(2) TP1/EE が rap サーバとして動作する場合の処理

rap サーバは、rap 受信スレッドで rap クライアントからの TCP コネクション確立要求メッセージ、および常設コネクション確立要求メッセージを受信します。常設コネクション確立要求メッセージは、リモート API 受信用ポート（RPC 関連定義の rap_listen_port オペランドで指定）で受け付けます。確立した常設コネクションは、常設コネクション解放応答メッセージ送信後、または送受信障害発生時に切断されます。

常設コネクション確立要求メッセージを受信したあと、rap 制御用処理キューへメッセージの登録要求をします。rap 制御用処理キューからサービスを引き出した処理スレッド（rap トランザクション）は、常設コネクションを経由して常設コネクション確立応答メッセージを rap クライアントに送信します。

常設コネクション確立エラーが発生する場合

同時に接続する rap クライアントの数が RPC 関連定義の rap_max_client オペランドの指定値を超えた場合、またはプロセス終了処理中に常設コネクション確立要求メッセージを受信した場合は、常設コネクション確立エラー応答メッセージを送信し、TCP コネクションを切断します。

常設コネクション確立状態で TP1/EE プロセスが終了した場合

rap クライアントと常設コネクションを確立している状態で TP1/EE プロセスを正常終了モード、または計画停止 A モードで終了した場合、TP1/EE の終了処理の動作は、RPC 関連定義の rpcRap_remaincon_endwait オペランドの指定に従います。

- RPC 関連定義の rpcRap_remaincon_endwait オペランドの指定値が Y の場合
常設コネクションが解放されるまでプロセス終了を待ち合わせます。
このとき、RPC 関連定義の rap_msg_output_interval オペランドで設定した時間間隔で常設コネクション数を表示します。rap_msg_output_interval オペランドに 0 を設定した場合は、初回だけ常設コネクション数を表示します。rap クライアントから常設コネクション解放要求がない場合は、常設コネクション切断コマンド（eerpRapcls）で切断できます。
- RPC 関連定義の rpcRap_remaincon_endwait オペランドの指定値が「N」の場合
接続中の常設コネクションに対して切断指示を行い切断します。
- RPC 関連定義の rpcRap_remaincon_endwait オペランドの指定値が「F」の場合
接続中の常設コネクションを強制的に切断します。

2.8.2 API の代理実行

図 2-46 で、API の代理実行要求受け付けの処理に該当するのは、4., 5.のメッセージ送受信です。

(1) TP1/EE が rap クライアントとして動作する場合の処理

リモート API 機能を使用するサービスグループに対して ee_rpc_call 関数を呼び出した場合、常設コネクションを経由して代理実行要求メッセージを rap サーバに送信します。サービスの実行結果は、代理実行応答メッセージとして受信します。

トランザクショナル RPC の代理実行要求を送信した場合

リモート API 機能を使用するサービスグループに対してトランザクショナル RPC (dc_rpc_call 関数の flags オプションに DCRPC_TPNOTRAN を指定しない) を実行した場合、非トランザクショナル RPC (ee_rpc_call 関数の flags オプションに EERPC_TPNOTRAN を指定) の代理実行要求として処理されます。

rap クライアントが不正なメッセージを受信した場合、または応答待ちでタイムアウトが発生した場合は、常設コネクションを切断します。

なお、代理実行要求メッセージを送信したあと、応答待ち時間 (RPC 関連定義の watch_time オペランドで指定) 以内に代理実行応答メッセージを受信できない場合、ee_rpc_call 関数はエラーリターンします。

(2) TP1/EE が rap サーバとして動作する場合の処理

rap サーバでの代理実行要求受け付けは、他サービスグループに対する代理実行要求の受け付けか、自サービスグループに対する代理実行要求の受け付けかによって処理が異なります。

(a) 他サービスグループに対する代理実行要求の受け付け

rap 受信スレッドが、rap クライアントからの代理実行要求メッセージを受信します。メッセージを受信したあと、rap 制御用処理キューへメッセージの登録要求をします。rap 制御用処理キューからメッセージを引き出した処理スレッド (rap トランザクション) は、メッセージの処理を実行する他サービスグループに代理実行要求をします。処理が完了すると、常設コネクションを経由して代理実行応答メッセージを rap クライアントに送信します。

同期応答型 RPC を使用した代理実行要求の最大応答待ち時間

同期応答型 RPC を使用した代理実行要求の場合、代理実行要求の最大応答待ち時間は rap クライアント側から継承して、TP1/EE (rap サーバ) が監視します。

トランザクショナル RPC の代理実行要求を受け付けた場合

トランザクショナル RPC (dc_rpc_call 関数の flags オプションに DCRPC_TPNOTRAN を指定しない) の代理実行要求を受け付けた場合、TP1/EE (rap サーバ) は、非トランザクショナル RPC (ee_rpc_call 関数の flags オプションに EERPC_TPNOTRAN を指定) の代理実行要求として処理します。

代理実行要求の処理を同時に実行する処理スレッド数

代理実行要求の処理を同時に実行する処理スレッド数の割合が、RPC 関連定義の rap_max_server_rate オペランドの指定値を超えた場合は、処理中の代理実行要求が終了してから、新しい代理実行要求を処理します。

(b) 自サービスグループに対する代理実行要求の受け付け

rap 受信スレッドが、rap クライアントからの代理実行要求メッセージを受信します。メッセージを受信したあと、該当するサービスのサービス処理キューへメッセージの登録要求をします。

サービス処理キューからメッセージを引き出した処理スレッド（サービストラランザクション）は、メッセージの処理を実行します。処理が完了すると、常設コネクションを経由して代理実行応答メッセージを rap クライアントに送信します。

同期応答型 RPC を使用した代理実行要求の最大応答待ち時間

同期応答型 RPC を使用した代理実行要求の場合、TP1/EE では応答待ち時間を監視しません。そのため、応答待ち時間は、rap クライアント側で監視します。

指定サービス名が不正、またはサービスが閉塞中の場合

代理実行要求のメッセージに指定されたサービス名が不正、またはサービスが閉塞中の場合、エラートラランザクション（ERRTRN1 および ERRTRN2）を起動しないで、rap クライアントにエラーを返します。

トランザクショナル RPC の代理実行要求を受け付けた場合

トランザクショナル RPC（dc_rpc_call 関数の flags オプションに DCRPC_TPNOTRAN を指定しない）の代理実行要求を受け付けた場合、TP1/EE（rap サーバ）は、非トランザクショナル RPC（ee_rpc_call 関数の flags オプションに EERPC_TPNOTRAN を指定）の代理実行要求として処理します。

自プロセスで代理実行要求を受け付けてサービスを実行する場合と通常の RPC でサービスを実行する場合との機能差異を次に示します。

- ダウン時の「トランザクション処理結果に従った RPC 応答メッセージの送信機能（RPC 関連定義の rpc_reply_tplmode_down オペランド）」
自プロセスがプロセスダウンした場合は常設コネクションが切断されるため、rap クライアント側に応答メッセージは返りません。
自プロセスがスレッドダウンした場合は、リターン値 EERPCER_TIMED_OUT でエラーリターンします。
- ERRTRNR による RPC 応答メッセージの送信機能
この機能を有効にしても、応答メッセージ送信の抑止は行いません。この機能を無効にした場合と同じ動作になります。
- 別トランザクションによる RPC 同期応答機能
RPC 応答メッセージ送信の抑止関数（ee_rpc_reply_suspend 関数）を発行した場合、リターン値 EERPCER_PROTO でエラーリターンします。

2.8.3 常設コネクションの解放

図 2-46 で常設コネクション解放要求受け付けの処理に該当するのは、6., 7.のメッセージ送受信です。

(1) TP1/EE が rap クライアントとして動作する場合の処理

rap クライアントは、常設コネクションを経由して常設コネクション解放要求メッセージを rap サーバに送信します。次に常設コネクション解放応答メッセージ受信して、TCP コネクションを切断します。

常設コネクションを解放し、TCP コネクションを切断する場合を次に示します。

- TP1/EE プロセスが正常終了または計画停止 A で終了する場合
rap サーバと常設コネクションを確立している状態で、rap クライアントとして動作している TP1/EE プロセスを正常終了または計画停止 A で終了した場合、終了トランザクションの終了後にすべての常設コネクションを解放します。常設コネクションは、rap サーバに対して常設コネクション解放要求メッセージを送信することで解放されます。複数の常設コネクションが確立されている場合は、rap クライアントとして動作している TP1/EE の終了処理に時間（最大で `rpc_rap_watch_time` オペランドの指定値分）が掛かることがあります。
- 最大数を超えて常設コネクションを確立しようとした場合
常設コネクション最大数（RPC 関連定義の `rpc_rap_connection_count` オペランドで指定）を超えて rap サーバに常設コネクションの確立要求を行った場合は、最も古く、かつ使用されていない常設コネクションを解放したあとで、新しく常設コネクションを確立します。

また、次に示す場合に常設コネクションが確立されているときは、強制的に TCP コネクションを切断します。

- 代理実行要求メッセージを送信したあと、応答待ち時間（RPC 関連定義の `watch_time` オペランドで指定）以内に代理実行応答メッセージを受信できなかった場合
- 代理実行要求メッセージの送信に失敗した場合
- 代理実行応答メッセージの受信で不正なメッセージを受信した場合
- rap サーバから常設コネクション解放指示のメッセージを受信した場合
- 常設コネクション解放要求メッセージを送信したあと、応答待ち時間（RPC 関連定義の `rpc_rap_watch_time` オペランドで指定）以内に常設コネクション解放応答メッセージを受信できない場合

(2) TP1/EE が rap サーバとして動作する場合の処理

rap サーバは、rap 受信スレッドで rap クライアントからの常設コネクション解放要求メッセージを受信します。

常設コネクション解放要求メッセージを受信したあと、rap 制御用処理キューへメッセージの登録要求をします。rap 制御用処理キューからサービスを引き出した処理スレッド（rap トランザクション）は、常設コネクションを経由して常設コネクション解放応答メッセージを rap クライアントに送信し、TCP コネクションを切断します。

rap クライアントから、常設コネクション解放要求メッセージを受信しないで常設コネクションを解放する場合があります。その場合は、rap サーバは、常設コネクションの解放を指示するメッセージを送信します。

2.8.4 メッセージ受信処理を実行できない場合

リモート API 機能では、次の場合はメッセージ受信処理を実行できません。

(1) 不正メッセージを受信した場合

rap クライアントから受信したメッセージのサイズが不正であったり、またはヘッダの設定値が不正であったりした場合は、メッセージ受信処理をしないで次の処理をします。

1. メッセージログを出力する。
2. エラー応答できる場合は、エラー応答メッセージを送信する。
3. TCP コネクションを切断する。

(2) リソース不足を検知した場合

rap クライアントからの代理実行要求メッセージ受信時、受信バッファ不足や処理キュー不足などのリソース不足で、メッセージ受信処理を実行できない場合は、リソースの確保、およびメッセージ受信処理をリトライします。リトライしてもリソース不足でメッセージ受信処理を実行できない場合は、メッセージ受信処理をしないで次の処理をします。

1. メッセージログを出力する。
2. TCP コネクションを切断する。

常設コネクション確立要求または代理実行要求の処理中にリソース不足が発生した場合は、常設コネクション確立要求、または代理実行要求の処理完了時に TCP コネクションを切断します。

2.9 スレッド制御

マルチスレッド構成のプロセスで UAP に異常が発生した場合、同期シグナルが発生して処理が中断されます。この場合、プロセス内の障害の発生したスレッドの処理だけを終了させて、ほかのスレッドの処理を続行させると、処理を続行させたスレッドがハングアップしたり異常終了したりするおそれがあります。そのため、通常は、UAP に異常が発生した場合は、プロセス内の全スレッドを終了させます。

しかし、プロセス内の全スレッドを終了させると、保留中のすべてのメッセージが破棄されてしまいます。これを避けるため、TP1/EE では、UAP を処理内容に応じて保護区（プロセスダウン域）と非保護区（スレッドダウン域）に分けてプロセスの終了を制御します。保護区を実行中に異常が発生した場合は、プロセス内の全スレッドを終了させます。非保護区を実行中に異常が発生した場合は、異常が発生したスレッドだけを終了させます。異常が発生したスレッドだけを終了させることで、プロセスの終了を局所化できます。

UOC は保護区であり、保護区／非保護区の切り替えはできません。

UAP または UOC に異常が発生した場合に発生する同期シグナルを次の表に示します。

表 2-10 UAP または UOC の異常で発生する同期シグナル

同期シグナル名	同期シグナルの発生要因
SIGBUS	バスエラー（アドレス境界不正など）が発生しました。
SIGEMT	EMT シグナルです。
SIGFPE	算術演算の例外（0 除算など）が発生しました。
SIGILL	不正な命令（0 アドレスコールなど）を実行しています。
SIGSEGV	無効なメモリへアクセス（不正アドレスの書き込み、スタックオーバーフローなど）しようとしています。

2.9.1 保護区／非保護区の制御

UAP の保護区／非保護区は、ee_thd_protstart 関数【CBLEETHD('PRTSTART')】および ee_thd_protend 関数【CBLEETHD('PRTEND ')】で切り替えます。デフォルトではすべての UAP が保護区に設定されています。また、保護区／非保護区の設定を有効にするかどうかは、プロセス関連定義の uapabend_downmode オペランドで指定します。プロセス関連定義の uapabend_downmode オペランドの指定については、マニュアル「TP1/Server Base Enterprise Option 使用の手引」の定義の説明を参照してください。

非保護区に設定する場合は、ほかのスレッドへの影響を十分に考慮してください。設定を誤ると、UAP の異常でスレッドを終了させたときに、TP1/EE プロセスがハングアップします。TP1/EE プロセスがハングアップすると、ハングアップ検出によって、TP1/EE プロセスが終了します。

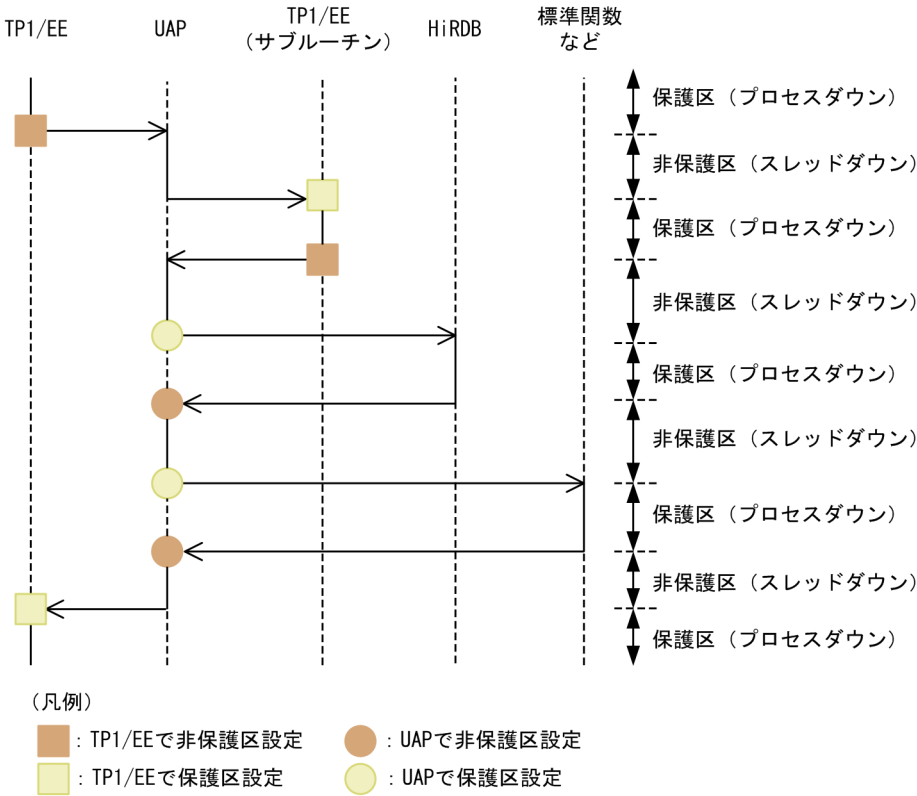
保護区／非保護区に設定する処理と、UAP に異常が発生したときの処理を次の表に示します。

表 2-11 保護区／非保護区に設定する処理と異常発生時の処理

区	処理	異常発生時の処理
保護区	異常が発生したスレッドだけを終了させた場合、ほかのスレッドがハングアップしたり、異常終了したりするおそれがある処理。	プロセスを終了させる。終了させたプロセスは、TP1/Server Base から再起動される。
非保護区	異常が発生したスレッドだけを終了させても、ほかのスレッドに影響がない処理。	異常が発生したスレッドを終了させる。終了させたスレッドは、モニタスレッドから再起動される。

UAP に非保護区を設定した場合に、プロセスを終了させる範囲、およびスレッドを終了させる範囲を次の図に示します。

図 2-47 非保護区を設定した場合の制御



2.10 資源の排他制御

任意の資源を，UAP から排他制御する方法について説明します。任意の資源を確保する場合，UAP から ee_lck_get 関数【CBLEELCK('GET ')】を呼び出します。

資源の排他制御は，トランザクションとして実行している処理で使います。それぞれのトランザクションから排他を指定することで，資源は正しく更新されて，UAP のトランザクション処理同士で排他できるようになります。

2.10.1 排他の対象となる資源

排他の対象になるのは，オペレーティングシステム内で固有の名称を定義してある，ファイルなどの資源です。資源の名称は，TP1/EE プロセス内で一意となる固有の名称を付けてください。排他制御する資源の名称が正しいかどうかは TP1/EE では判断できません。UAP で指定する資源名称は論理的に正しい名称を指定してください。排他指定の有効範囲は，一つの TP1/EE プロセス内で，かつ同じノード内に限ります。ほかの TP1/EE プロセス上の UAP との排他制御はできません。また，排他状態はリラン時に引き継がれません。

2.10.2 排他の種類

排他制御では，TP1/EE プロセス内で固有の資源名称と排他の条件（排他制御モード）を指定します。排他制御モードには，次の 2 種類があります。

参照目的の排他（PR モード Protected Retrieve）

UAP は排他指定した資源の参照だけできます。ほかの UAP には参照だけを許可し，更新を禁止します。

更新目的の排他（EX モード EXclusive）

UAP は排他指定した資源の参照，更新ができます。ほかの UAP には参照と更新の両方を禁止します。

排他を指定しようとした資源が，ほかの UAP ですでに排他指定をされていた場合，互いの排他モードの内容によって，資源を共用できる場合とできない場合があります。

一つの資源に対して複数の UAP から排他制御の指定があった場合の，共用の可否を次の表に示します。資源を共用できない場合に，エラーリターンするか，資源の解放待ちにするかは UAP で指定できます。

表 2-12 排他制御モードの組み合わせと共用の可否

資源を確保中の UAP のモード	排他要求した UAP のモード	
	参照目的の排他（PR モード）	更新目的の排他（EX モード）
参照目的の排他（PR モード）	共用できます。	共用できません。
更新目的の排他（EX モード）	共用できません。	共用できません。

2.10.3 排他待ち限界経過時間の指定

UAP から排他を指定されている資源に対して、ほかの UAP が排他要求した場合、その UAP は資源の解放を待つことができます。さらに続けて解放待ちの UAP がある場合は、排他待ちの優先順位に従って、資源の解放を待ちます。排他待ちの優先順位は資源の排他要求を行った順番によって決まります。資源の排他要求を行う順番がほかの UAP よりも早い場合、排他待ちの優先順位はほかの UAP よりも高くなります。排他待ちの優先順位が高い場合、排他待ちしている、優先順位の低いほかの UAP よりも先に資源を排他できます。

トランザクション関連定義の `lck_wait_timeout` オペランドで排他待ち限界経過時間を指定して、解放待ちの UAP が指定した時間を超えた場合は、その UAP はエラーリターンします。

UAP が排他待ちをしている資源名称、排他待ち限界経過時間などは、`eelckls` コマンドで知ることができます。

2.10.4 排他の解除方法

資源の排他を解除する方法には次の二つがあります。

- 資源を確保している UAP から排他を解除します。解除する資源名称を指定して排他を解除する場合は、`ee_lck_release_byname` 関数【`CBLEELCK('RELNAME ')`】を呼び出します。UAP で確保しているすべての資源の排他を一度に解除する場合は、`ee_lck_release_all` 関数【`CBLEELCK('RELALL ')`】を呼び出します。排他を解除する関数は、その資源の排他を指定した UAP からだけ呼び出せます。排他を解除された資源は、TP1/EE が資源の排他の解除待ちの UAP に割り当てます。
- 資源を確保している UAP の同期点処理後に、その UAP が確保していたすべての資源を TP1/EE で解放します。UAP の終了形態が正常でも異常でも、TP1/EE で自動的に解放します。

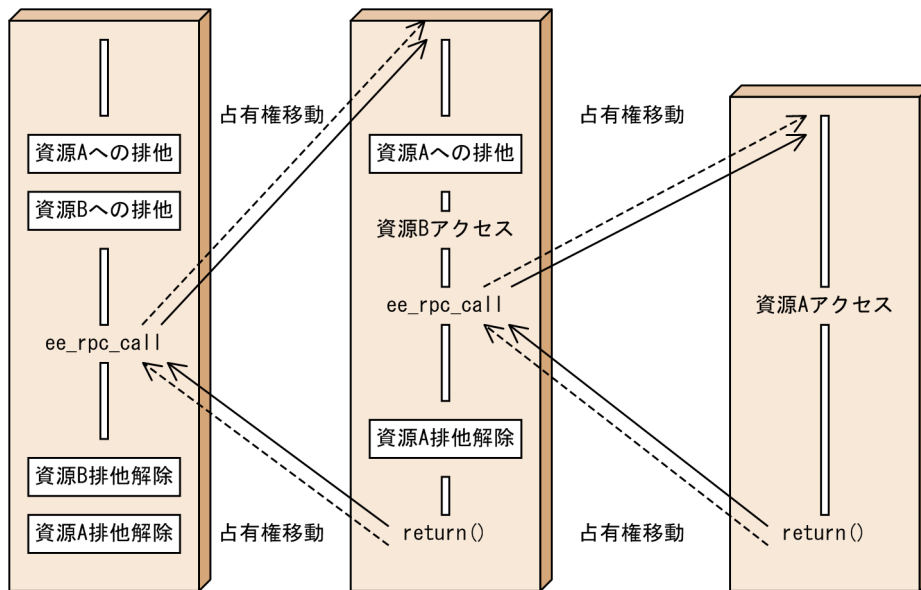
2.10.5 ロックマイグレーション

`ee_lck_get` 関数で資源の排他をする場合、一つのグローバルトランザクション内の各トランザクションブランチに、資源の占有権が順次移動します。この機能をロックマイグレーションといいます。ロックマイグレーションによって、トランザクションブランチ間の排他待ちやデッドロックを防げます。そのため、あるグローバルトランザクションで排他を指定した資源に対しては、その資源が解放されないかぎり、一つのグローバルトランザクション内のどのトランザクションブランチからでもアクセスできます。

ロックマイグレーションは、一つの TP1/EE プロセス内にグローバルトランザクションがある（グローバルトランザクションが複数の TP1/EE プロセスのサービスから構成されていない）場合に保証されます。

ロックマイグレーションの概要を次の図に示します。

図 2-48 ロックマイグレーションの概要



(1) ロックマイグレーションと排他制御モード

ロックマイグレーションでは、PR モードで排他をしても、別のトランザクションブランチで EX モードを指定すれば、それ以降の排他はすべて EX モードになります。一つのグローバルトランザクション内では、一度 EX モードで排他した資源には、PR モードで排他できません。すべて EX モードでの排他となります。

(2) ロックマイグレーションでの資源の解放

ロックマイグレーションの排他は、グローバルトランザクションが終了したときに、自動的に解放されます。グローバルトランザクションの終了を待たないで、排他を解除する方法を次に示します。

- ee_lck_release_byname 関数での解放

ロックマイグレーションの排他をコミット／ロールバックを待たないで解放するときは、資源名称を指定して排他を解除する関数（ee_lck_release_byname 関数）を呼び出してください。排他の解除は、どのトランザクションブランチでもできます。この場合、グローバルトランザクション内でその資源に対して排他を指定した回数だけ、排他解除の関数を呼び出すまでは、資源は解放されません。

- ee_lck_release_all 関数での解放

全資源の排他を解除する関数（ee_lck_release_all 関数）をどこかのトランザクションブランチで呼び出せば、どのトランザクションブランチで何回排他を指定していても、すべての資源の排他を解除できます。

(3) ロックマイグレーションでの注意事項

ロックマイグレーションでの注意事項を次に示します。

- 同期応答型 RPC の ee_rpc_call 関数【CBLEERPC('CALL ')】でロックマイグレーションが起こったあとで、この ee_rpc_call 関数がタイムアウトなどでエラーリターンした場合は、排他した資源に対す

るアクセス（確保済みの資源へのアクセスや新たな排他要求）はしないでください。アクセスした場合の動作は保証されません。

- 非応答型 RPC，または非同期応答型 RPC の ee_rpc_call 関数【CBLEERPC('CALL ')】でロックマイグレーションが起こった場合，ほかのトランザクションブランチで排他した資源に対して排他解除の関数を呼び出しても，解除できないことがあります。この場合，次のときに自動的に資源が解放されます。
 - 資源の排他を行ったトランザクションブランチから，排他解除する関数を呼び出したとき。
 - グローバルトランザクションが終了したとき。

2.10.6 排他のテスト

資源の排他をする関数（ee_lck_get 関数）で，資源を排他できるかどうかをテスト（flags に EELCK_TEST を設定）できます。この場合 ee_lck_get 関数は，資源を確保できる状態でも実際に排他をしないで正常終了します。指定した資源がほかの UAP から排他されている場合は，排他待ちの指定の有無に関係なく EELCKER_WAIT (00450) でエラーリターンとなります。そのほか，テストの排他の場合は次に示すリターン値が戻ります。なお，タイムアウトおよびメモリ不足のエラーリターンはしません。

- EELCKER_PARAM (00401)
引数の設定誤り
- EELCKER_OUTOFTURN (00455)
トランザクション処理でない UAP からの関数呼び出し

注意事項

排他のテストは，排他できる状態かどうかを知るために行います。排他のテストが正常終了したことで，それ以降の排他要求が正常終了するかどうかは保証されません。また，排他のテストが正常終了しても，実際には資源は占有されていません。そのため，排他のテストが正常終了したあとに排他を解除する関数（ee_lck_release_all 関数，ee_lck_release_byname 関数）を呼び出すとエラーリターンします。

3

SPP (TP1/EE) の作成の詳細

この章では、TP1/EE および XTC のアプリケーションプログラムを C 言語、または COBOL 言語でコーディングする場合の詳細について説明します。

3.1 SPP (TP1/EE) の作成

SPP (TP1/EE) の作成手順を次の図に示します。

図 3-1 SPP (TP1/EE) の作成手順

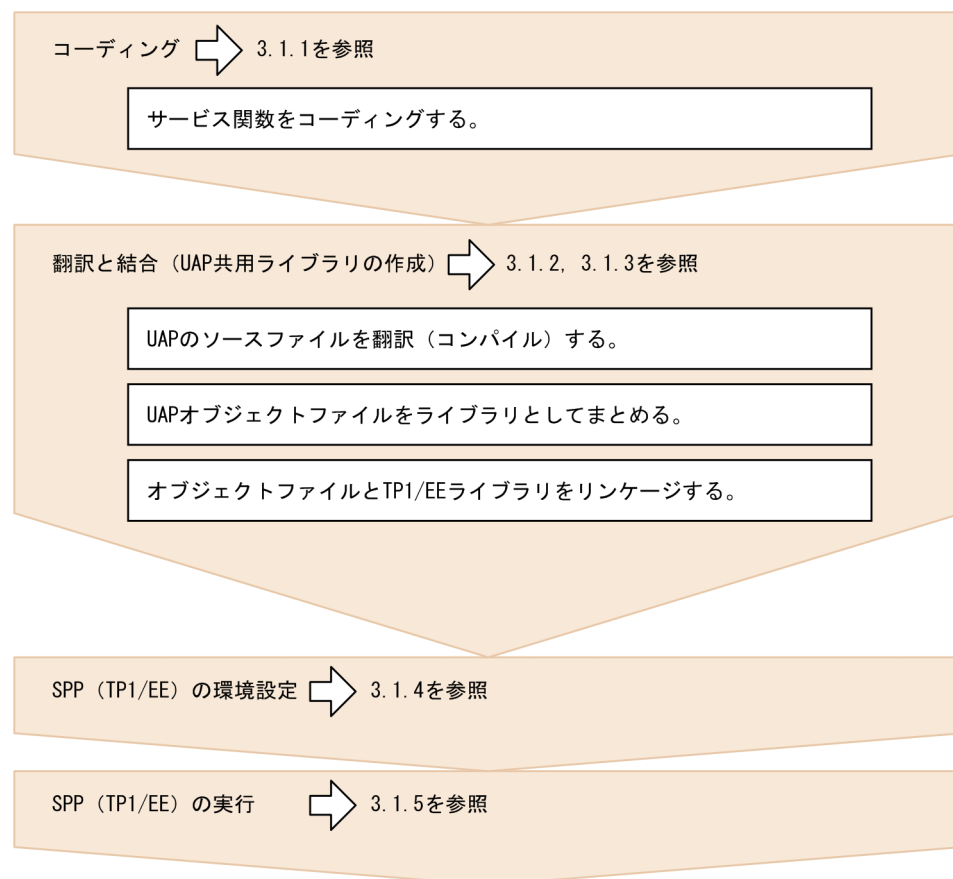
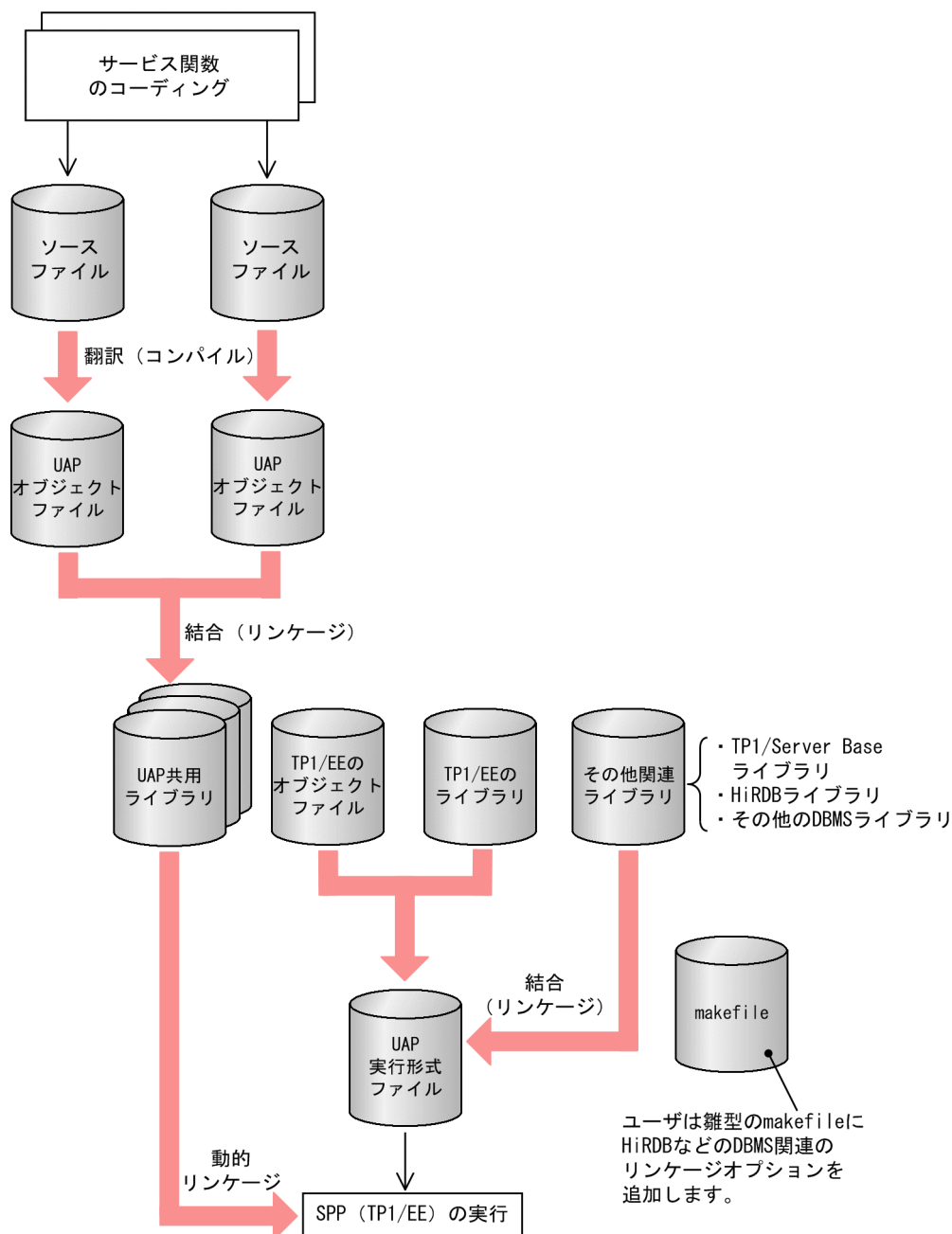


図 3-2 SPP の作成手順



UAP 作成時の注意

システムサービスによっては、古いバージョンの UAP からの関数呼び出しを受け付けないことがあります。旧バージョンの TP1/EE で作成した UAP を使用する場合、現在使用しているバージョンの TP1/EE で、UAP オブジェクトファイルをコンパイル／リンケージし直すことをお勧めします。

以降に、SPP の作成手順の詳細を説明します。

3.1.1 サービス関数のコーディング

サービス関数のコーディングは、サービスグループ単位で行います。

コーディングには、C 言語、C++言語、または COBOL 言語を使います。

サービス関数をコーディングしてソースファイルを作成するとき、TP1/EE の機能のほかにも、OS の標準の機能やデータベース言語 (SQL) を使えます。

(1) C 言語または C++言語でコーディングする場合

(a) C 言語を使うとき

ANSI C の形式に従ってコーディングします。TP1/EE の機能を使うときは、TP1/EE の API 関数を呼び出します。

(b) C++言語を使うとき

ANSI C の形式で C++言語の仕様に従ってコーディングします。TP1/EE の機能を使うときは、TP1/EE の API 関数を呼び出します。TP1/EE の API 関数は、ヘッダファイル (ee×××.h) で C 言語のリンケージを指定しているため、C++言語でコーディングした UAP オブジェクトファイルのリンケージの際には C 言語の関数としてリンケージされ動作します。また、サービス関数の外部参照名は、C 言語の形式にしてください。

(c) TP1/EE の関数の使い方

OS で標準的に提供する関数と同様、関数を呼び出すときには、引数を設定します。

関数が正常に実行されたかどうかは、戻ってくる値 (リターン値) でわかります。

C 言語でコーディングする場合の概要を次の図に示します。

図 3-3 C 言語でコーディングする場合の概要

<pre>#include <stdio.h> #include <string.h> #include <eerp.h> svr01() { /* サービス関数の開始 */ /* リモートプロシジャコール */ rc = ee_rpc_call("svr01","svr01",...); if(rc !=EE_OK){ printf("サービスの要求失敗!!"); goto PROG_END; } PROG_END; printf("svr01:処理を終了します\n"); return; }</pre>	プログラムの処理で使うヘッダファイルを取り込みます。 TP1/EEのヘッダファイルは“ee”で始まります。 ※ サービス関数は関数名で始めます。 サービス関数の終了後は、メイン関数に処理をリターンします。

注※
メイン関数処理はTP1/EEで用意します。
したがって、サービス関数のコーディング時は作成しません。

(2) COBOL 言語でコーディングする場合

COBOL 言語を使うときは、COBOL85、COBOL2002 の形式でコーディングします。ただし、OS が Linux の場合は、次の形式でコーディングしてください。

Linux (IA32)の場合

COBOL 製品として使用できるのは COBOL85 だけです。必ず COBOL85 の形式でコーディングしてください。

Linux (EM64T)の場合

COBOL 製品として使用できるのは COBOL2002 だけです。必ず COBOL2002 の形式でコーディングしてください。

TP1/EE の機能を使うときは、TP1/EE のライブラリにある **COBOL-UAP 作成用プログラム**を使います。COBOL-UAP 作成用プログラムを、COBOL 言語の CALL 文で呼び出して、SPP の処理から TP1/EE のライブラリに制御を移します。

CALL 文の実行結果は、戻ってくる数値（ステータスコード）でわかります。

COBOL 言語でコーディングする場合の概要を次の図に示します。

図 3-4 COBOL 言語でコーディングする場合の概要

<pre>* IDENTIFICATION DIVISION. * PROGRAM-ID. SPP01. * DATA DIVISION. WORKING-STORAGE SECTION. 01 ARG2. 02 REQUEST PIC X(8) VALUE SPACE. 02 STATUS-CODE PIC X(5) VALUE SPACE. : * PROCEDURE DIVISION. * MOVE 'CALL' TO REQUEST OF ARG2. MOVE 'SPP01' TO G-NAME OF ARG2. MOVE 'SVR01' TO S-NAME OF ARG2. : CALL 'CBLEERPC' USING ARG2 ARG3 ARG4. IF STATUS-CODE OF ARG2 NOT = '00000' THEN : END-IF. :</pre>	TP1/EEのCOBOL-UAP作成用プログラムで使う領域を、DATA DIVISIONで定義します。 定義した領域に、必要な値を設定します。 CALL文でCOBOL-UAP作成用プログラムを呼び出します。

(3) コーディング規約

コーディングするときは、マニュアル「OpenTP1 プログラム作成リファレンス」の該当する言語編のコーディング規約に従ってください。SQL のコーディングの規約については、該当するリファレンスマニュアルを参照してください。

マニュアル「OpenTP1 プログラム作成リファレンス」の該当する言語編で示されているコーディング規約以外に、TP1/EE として定められているコーディング規約を次に示します。

- サービス関数でメイン関数を作成しないでください。
- "ee"で始まる名称を使用しないでください。
- "CBLEE"で始まる名称を使用しないでください。
- どのシグナルに対しても操作をしないでください。

3.1.2 UAP 共用ライブラリの作成

ソースファイルをコンパイルして、UAP オブジェクトファイルを作成します。

コンパイルして作成した各 UAP オブジェクトファイルをリンケージして、UAP 共用ライブラリを作成します。UAP 共用ライブラリは、マルチスレッド用ライブラリとして作成する必要があります。コンパイルオプションおよびリンケージオプションについては、TP1/EE のサンプル (makefile) を参照してください。

3.1.3 UAP 実行形式ファイルの作成

TP1/EE のオブジェクトファイルとライブラリをリンケージして、UAP 実行形式ファイルを作成します。UAP 実行形式ファイルは、マルチスレッド用実行形式ファイルとして作成する必要があります。リンケージオプションについては、TP1/EE のサンプル (makefile) を参照してください。

UAP 実行形式ファイルの作成に必要なライブラリは次のとおりです。

- TP1/EE のライブラリ
DB キュー機能を使用しない場合は、DB キュー共通ライブラリをリンケージしないでください。DB キュー機能を使用する場合は、DB キュー共通ライブラリをほかの TP1/EE のライブラリより前に指定してください。
DB キュー HiRDB 用ライブラリおよび DB キュー Oracle 用ライブラリは、リンケージしないでください。この二つのライブラリは、使用するリソースマネージャに合わせて TP1/EE が動的にローディングします。
- TP1/Server Base のライブラリ
- リソースマネージャ連携オブジェクトファイル
XA インタフェースで DBMS にアクセスする場合に必要になります。eetrmkobj コマンドを実行してリソースマネージャ連携オブジェクトファイルを作成します。
- HiRDB ライブラリ

XA インタフェースで DBMS (HiRDB) にアクセスする場合に必要になります。リソースマネージャのオブジェクトファイルのリンケージ方法については、ご使用になるリソースマネージャのリファレンスマニュアルを参照してください。

- その他の DBMS のライブラリ

XA インタフェースで DBMS (HiRDB 以外) にアクセスする場合に必要になります。リソースマネージャのオブジェクトファイルのリンケージ方法については、ご使用になるリソースマネージャのリファレンスマニュアルを参照してください。

3.1.4 SPP (TP1/EE) の環境設定

作成した UAP 共用ライブラリと UAP 実行形式ファイルを、TP1/EE で使えるように、次のとおり環境設定します。

(1) 格納するディレクトリ

作成した UAP 実行形式ファイルは、\$DCDIR/aplib ディレクトリに格納します。UAP 共用ライブラリは、任意のディレクトリに格納します。

(2) TP1/EE への登録

UAP 共用ライブラリと UAP 実行形式ファイルを、TP1/EE に登録します。

登録方法

TP1/EE の UAP 共用ライブラリ名に UAP 共用ライブラリを登録するときに、実行環境を設定します。実行環境は、TP1/Server Base のユーザサービス定義で設定します。

ユーザサーバ名

TP1/Server Base で SPP を操作するときに使う名称をユーザサーバ名といいます。SPP 名は、TP1/Server Base のユーザサービス定義のファイル名です。

(3) SPP の名称

SPP の名称について説明します。

- UAP の共用ライブラリ名
UAP オブジェクトファイルをライブラリにまとめたときの名称です。
- UAP 実行形式ファイル
TP1/EE のオブジェクトファイルを関連ライブラリとリンケージするときに、リンケージのコマンドにオプションで指定した名称です。
- ユーザサーバ名
UAP 実行形式ファイルを TP1/Server Base に登録するときに指定した名称です。dcsvstart コマンド、および eesvstart コマンドの引数に使用します。ユーザサーバ名は、1~8 文字で指定します。

- サービスグループ名，サービス名

TP1/Server Base または TP1/EE のリモートプロシジャコールで，サービスを要求する関数の引数に指定する名称です。ユーザサーバ名を TP1/Server Base のユーザサービス定義，および TP1/EE の TP1/EE サービス定義に登録するとき一緒に指定します。

サービスグループ名は，UAP 実行形式ファイル単位に名称を付けます。

サービス名は，サービス関数の関数名です。

3.1.5 SPP (TP1/EE) の実行

UAP 共用ライブラリと UAP 実行形式ファイルをリンケージして，SPP (TP1/EE) を実行します。

3.2 TP1/EE および XTC のアプリケーションプログラムと API

3.2.1 TP1/EE および XTC の API

TP1/EE および XTC の機能と、C 言語、COBOL 言語の API 関数の対応を次の表に示します。

表 3-1 C 言語、COBOL 言語の API 関数の対応

機能分類	機能詳細	C 言語の API 関数名	COBOL 言語の API 関数名
システム運用の管理	運用コマンドの実行	ee_adm_call_command	CBLEEDADM('COMMAND ')
DB キュー	DB キューのアクセス制御	ee_dbq_acsctl	CBLEEDBQ('ACSCTL ')
	DB キューの通番初期化	ee_dbq_clear	CBLEEDBQ('CLEAR ')
	DB キューの接続制御	ee_dbq_conctl	CBLEEDBQ('CONCTL ')
	DB キュー用サービスの接続照会	ee_dbq_inqsrv	CBLEEDBQ('INQSRV ')
	DB キュー情報の照会	ee_dbq_inquire	CBLEEDBQ('INQUIRE ')
	DB キュー情報の照会 (ユーザキューアクセス機能使用時)	ee_dbq_inquire2	CBLEEDBQ('INQUIRE2')
	メッセージの読み出し	ee_dbq_msgget	CBLEEDBQ('MSGGET ')
	メッセージの書き込み	ee_dbq_msgput	CBLEEDBQ('MSGPUT ')
	メッセージの読み出し (参照読み出し)	ee_dbq_msgrefer	CBLEEDBQ('MSGREFER')
	メッセージの読み出し (参照読み出し) (ユーザキューアクセス機能使用時)	ee_dbq_msgrefer2	CBLEEDBQ('MSGREFE2')
	オンラインバッチ処理の強制終了	ee_dbq_obscancel	CBLEEDBQ('OBSCAN ')
	引き継ぎ情報の更新	ee_dbq_obschdata	CBLEEDBQ('OBSCDT ')
	オンラインバッチ処理の終了	ee_dbq_obsend	CBLEEDBQ('OBSEND ')
	ロットの情報の照会	ee_dbq_obsinquire	CBLEEDBQ('OBSINQ ')
	ロールバック要求時のメッセージの読み出しスキップ	ee_dbq_obsrbskip	CBLEEDBQ('OBSRBSP ')
	オンラインバッチ処理の再開始	ee_dbq_obsrestart	CBLEEDBQ('OBSRST ')
	メッセージの読み出しスキップ	ee_dbq_obsskip	CBLEEDBQ('OBSSKIP ')
	オンラインバッチ処理の開始	ee_dbq_obsstart	CBLEEDBQ('OBSSTART')

機能分類	機能詳細	C 言語の API 関数名	COBOL 言語の API 関数名
DB キュー	オンラインバッチ処理の中断	ee_dbq_obsstop	CBLEEDBQ('OBSSTOP ')
	ロールバック要求時のメッセージの読み出しスキップ	ee_dbq_rbskip	CBLEEDBQ('RBSKIP ')
	メッセージの読み出しスキップ	ee_dbq_skip	CBLEEDBQ('SKIP ')
	DB キュー用サービスの閉塞制御	ee_dbq_srvctl	CBLEEDBQ('SRVCTL ')
資源の排他制御	資源の排他	ee_lck_get	CBLEELCK('GET ')
	全資源の排他の解除	ee_lck_release_all	CBLEELCK('RELALL ')
	資源名称を指定した排他の解除	ee_lck_release_byname	CBLEELCK('RELNAME ')
メッセージログ の出力	メッセージログの出力	ee_logprint	CBLEELOG('PRINT ')
高速メッセージ 送信	トランザクション非同期の一方送信 メッセージの送信 【XTC 限定】	ee_mch_cmtsend	CBLEEMCH('CMTSND ')
	一方送信メッセージ情報の取得 【XTC 限定】	ee_mch_cmtsend_get	CBLEEMCH('CMTSNDGT')
	トランザクション同期の一方送信 メッセージの送信 【XTC 限定】	ee_mch_cmtsend_sync	CBLEEMCH('CMTSND SY')
	出力キュー（OTQ）の閉塞制御 【XTC 限定】	ee_mch_otqbctl	CBLEEMCH('OTQBKCTL')
	出力キュー（OTQ）の情報の 照会 【XTC 限定】	ee_mch_otqinquire	CBLEEMCH('OTQINQIR')
	一方送信メッセージの送信ス キップ 【XTC 限定】	ee_mch_otqskip	CBLEEMCH('OTQSKIP ')
メモリ管理	ワークセグメントの割り当て	ee_mem_getwkseg	CBLEEMEM('GETWK ')
	大量処理用メモリ管理機能の使用 開始 【XTC 限定】	ee_mem_mdpstart	CBLEEMEM('MDPSTART')
	使用済みワークセグメントの返還	ee_mem_rlswkseg	CBLEEMEM('RLSWK ')
リモートプロシ ジャコール	遠隔サービスの要求	ee_rpc_call	CBLEERPC('CALL ')
	通信先を指定した遠隔サービスの 要求	ee_rpc_call_to	CBLEERPC('CALLTO ')
	トランザクションと同期して送信 する非応答型 RPC 要求	ee_rpc_cmtsend	CBLEERPC('CMTSND ')
	通信先を指定したトランザクショ ンと同期して送信する非応答型 RPC 要求	ee_rpc_cmtsend_to	CBLEERPC('CMTSNDTO')
	すべての非同期応答型 RPC の応 答受信拒否	ee_rpc_discard_further_repli es	CBLEERPC('DISCARD F')

機能分類	機能詳細	C 言語の API 関数名	COBOL 言語の API 関数名
リモートプロシ ジャコール	特定の非同期応答型 RPC の応答 受信拒否	ee_rpc_discard_specific_rep ly	CBLEERPC('DISCARDS')
	エラーが発生した非同期応答型 RPC の識別子取得	ee_rpc_get_error_descriptor	CBLEERPC('GETERDES')
	サービスの応答待ち時間の参照	ee_rpc_get_watch_time	CBLEERPC('GETWATCH')
	非同期応答型 RPC の応答受信	ee_rpc_poll_any_replies	CBLEERPC('POLLANYR')
	抑止していた RPC 応答メッセー ジの送信	ee_rpc_reply_send	CBLEERPC('R-SEND ')
	RPC 応答メッセージ送信の抑止	ee_rpc_reply_suspend	CBLEERPC('R-SPEND ')
	サービスの応答待ち時間の更新	ee_rpc_set_watch_time	CBLEERPC('SETWATCH')
滞留メッセージ 受信	ロールバックリトライ回数のク リア 【XTC 限定】	ee_scd_clear_rollback_cnt	CBLEESCD('CLRRBCNT')
	滞留メッセージの受信 【XTC 限 定】	ee_scd_msg_receive	CBLEESCD('MSGRECV ')
	受信メッセージ差し戻し 【XTC 限定】	ee_scd_msg_rollback	CBLEESCD('MSGRLBK ')
	滞留メッセージのスキップ 【XTC 限定】	ee_scd_msg_skip	CBLEESCD('MSGSKIP ')
スレッド制御	サービス閉塞制御	ee_thd_abdctl	CBLEETHD('ABDCTL ')
	現在時刻取得	ee_thd_gettime	CBLEETHD('GETTIME ')
	保護区終了	ee_thd_protend	CBLEETHD('PRTEND ')
	保護区開始	ee_thd_protstart	CBLEETHD('PRTSTART')
	処理スレッド数取得	ee_thd_thdcntget	CBLEETHD('CNTGET ')
	ユーザ IFA 引き継ぎ機能	ee_thd_uifaget	CBLEETHD('UIFAGET ')
タイマトランザ クション	タイマトランザクションの起動	ee_tim_execap	CBLEETIM('EXECAP ')
	タイマトランザクションの起動取 り消し	ee_tim_execap_cancel	CBLEETIM('CANCEL ')
	全タイマトランザクションの起動 取り消し	ee_tim_execap_cancel_all	CBLEETIM('CANCELAL')
トラブルシュー ト	ユーザ固有のトレース取得	ee_trb_ustrace_put	CBLEETRB('UTPUT ')
	ユーザ情報の TASKTM 出力	ee_trb_uinfo_tm	CBLEETRB('UINFOTM ')
トランザクシ ョン制御	連鎖モードのコミット	ee_tm_chained_commit	CBLEETRN('C-COMMIT')
	連鎖モードのロールバック	ee_tm_chained_rollback	CBLEETRN('C-ROLL ')
	現在のトランザクション識別情報 の取得	ee_tm_getinfo	CBLEETRN('GETINFO ')

機能分類	機能詳細	C 言語の API 関数名	COBOL 言語の API 関数名
トランザクション制御	リソースマネージャとのコネクションの解放指示	ee_trn_rm_close_mark	CBLEETRN('RMCLSMK ')
	トランザクションのロールバック指示	ee_trn_rollback_mark	CBLEETRN('ROLLMARK')
	トランザクション処理の監視時間の変更	ee_trn_set_exp_time	CBLEETRN('EXPTIME ')

なお、次の関数はルート以外からトランザクションを発行できません。「ルート」とは、ルートトランザクションブランチ、「ルート以外」とは、ルートトランザクションブランチ以外のトランザクションブランチのことです。

- ee_dbq_obsend, CBLEEDBQ('OBSEND ')
- ee_dbq_obsrbskip, CBLEEDBQ('OBSRBSP ')
- ee_dbq_obsstop, CBLEEDBQ('OBSSTOP ')
- ee_dbq_rbskip, CBLEEDBQ('RBSKIP ')
- ee_rpc_reply_suspend, CBLEERPC('R-SPEND ')
- ee_trn_chained_commit, CBLEETRN('C-COMMIT')

3.2.2 トランザクションで利用できる API 関数

SPP で使用できる TP1/EE および XTC の API は、トランザクション種別によって異なります。それぞれのトランザクションで利用できる API を、製品別に以降の表に示します。

表 3-2 TP1/EE のトランザクションで利用できる API 関数

API 関数	トランザクション種別									
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM
ee_adm_call_command* ¹	○	○	○	○	○	○	○	○	○	○
ee_dbq_acsctl	×	×	○	○	○	○	○	○	○	○
ee_dbq_clear	×	×	○	○	○	○	○	○	×	○
ee_dbq_conctl	×	×	○	○	○	○	○	○	○	○
ee_dbq_inqsrv	×	×	○	○	○	○	○	○	○	○
ee_dbq_inquire	×	×	○	○	○	○	○	○	○	○
ee_dbq_inquire2	×	×	○	○	○	○	○	○	○	○
ee_dbq_msgget	×	×	○	△	△	△	△	△	×	○
ee_dbq_msgput	×	×	○	△	△	△	△	△	×	○

API 関数	トランザクション種別									
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM
ee_dbq_msgrefer	×	×	○	△	△	△	△	△	×	○
ee_dbq_msgrefer2	×	×	○	△	△	△	△	△	×	○
ee_dbq_obscancel	×	×	○	○	○	○	○	○	×	○
ee_dbq_obschdata	×	×	○	○	○	×	○	○	×	○
ee_dbq_obsend	×	×	×	×	×	×	×	×	×	×
ee_dbq_obsinquire	○	○	○	○	○	○	○	○	○	○
ee_dbq_obsrbskip	×	×	×	×	×	×	×	×	×	×
ee_dbq_obsrestart	×	×	○	○	○	○	○	○	×	○
ee_dbq_obsrskip	×	×	○	×	×	×	×	×	×	×
ee_dbq_obsstart	×	×	○	○	○	○	○	○	×	○
ee_dbq_obsstop	×	×	×	×	×	×	×	×	×	×
ee_dbq_rbskip	×	×	○	×	×	×	×	×	×	×
ee_dbq_skip	×	×	○	○	○	○	○	○	×	○
ee_dbq_srvctl	×	×	○	○	○	○	○	○	○	○
ee_lck_get	○	○	○	○	○	○	○	○	○	○
ee_lck_release_all	○	○	○	○	○	○	○	○	○	○
ee_lck_release_byname	○	○	○	○	○	○	○	○	○	○
ee_logprint	○	○	○	○	○	○	○	○	○	○
ee_mch_cmtsend	×	×	○	○	○	○	○	○	○	○
ee_mch_cmtsend_get*2	○	○	○	○	○	○	○	○	○	○
ee_mch_cmtsend_sync	×	×	○	○	○	○	○	○	○	○
ee_mch_otqbctl	○	○	○	○	○	○	○	○	○	○
ee_mch_otqinquire	○	○	○	○	○	○	○	○	○	○
ee_mch_otqskip	○	○	○	○	○	○	○	○	○	○
ee_mem_getwkseg	○	○	○	○	○	○	○	○	○	○
ee_mem_mdpstart	○	○	○	○	○	○	○	○	○	○
ee_mem_rlswkseg	○	○	○	○	○	○	○	○	○	○
ee_rpc_call	△	△	○	○	○	○	○	○	○	○
ee_rpc_call_to	△	△	○	○	○	○	○	○	○	○
ee_rpc_cmtsend	×	×	○	○	○	○	○	○	○	○

API 関数	トランザクション種別									
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM
ee_rpc_cmtsend_to	×	×	○	○	○	○	○	○	○	○
ee_rpc_discard_further_replies	○	○	○	○	○	○	○	○	○	○
ee_rpc_discard_specific_reply	○	○	○	○	○	○	○	○	○	○
ee_rpc_get_error_descriptor	○	○	○	○	○	○	○	○	○	○
ee_rpc_get_watch_time	○	○	○	○	○	○	○	○	○	○
ee_rpc_poll_any_replies	○	○	○	○	○	○	○	○	○	○
ee_rpc_reply_send	○	○	○	○	○	○	○	○	○	○
ee_rpc_reply_suspend	×	×	○	×	×	×	×	×	×	×
ee_rpc_set_watch_time	○	○	○	○	○	○	○	○	○	○
ee_scd_clear_rollback_cnt	×	×	○	○	○	○	○	○	○	○
ee_scd_msg_receive	×	×	×	×	×	×	×	×	×	×
ee_scd_msg_rollback	×	×	×	×	×	×	×	×	×	×
ee_scd_msg_skip	○	○	○	○	○	○	○	○	○	○
ee_thd_abdctl	○	○	○	○	○	○	○	○	○	○
ee_thd_gettime	○	○	○	○	○	○	○	○	○	○
ee_thd_protend	○	○	○	○	○	○	○	○	○	○
ee_thd_protstart	○	○	○	○	○	○	○	○	○	○
ee_thd_thdcntget	○	○	○	○	○	○	○	○	○	○
ee_thd_uifaget	×	×	×	×	×	○	×	×	×	×
ee_tim_execap	○	×	○	○	○	○	○	○	○	○
ee_tim_execap_cancel	○	○	○	○	○	○	○	○	○	○
ee_tim_execap_cancel_all	○	○	○	○	○	○	○	○	○	○
ee_trb_uinfo_tm	○	○	○	○	○	○	○	○	○	○
ee_trb_ustrace_put	○	○	○	○	○	○	○	○	○	○
ee_trn_chained_commit	○	○	○	○	○	○	○	○	○	○
ee_trn_chained_rollback	○	○	○	○	○	○	○	○	○	○
ee_trn_getinfo	○	○	○	○	○	○	○	○	○	○
ee_trn_rm_close_mark	○	○	○	○	○	○	○	○	○	○
ee_trn_rollback_mark	○	○	○	○	○	○	○	○	○	○
ee_trn_set_exp_time	○	○	○	○	○	○	○	○	○	○

(凡例)

○：使用できます。

△：一部使用できます。詳細は各 API の説明を参照してください。

×：使用できません。

注※1

運用コマンド実行機能での、TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否については、「2.6 運用コマンド実行機能」を参照してください。

注※2

トランザクション種別が ES 以外の場合、この関数を発行しても一方送信メッセージ情報はありません。

表 3-3 MCP, XTC, および TP1/FSP のトランザクションで利用できる API 関数

API 関数	トランザクション種別						
	UI	MV	RL	UT	CU	JI	BS
ee_adm_call_command※1	○	○	○	○	○	○	×
ee_dbq_acsctl	×	×	○	×	○	○	×
ee_dbq_clear	×	×	○	×	○	△	×
ee_dbq_conctl	×	×	○	×	○	○	×
ee_dbq_inqsrv	×	×	○	×	○	○	×
ee_dbq_inquire	×	×	○	×	○	○	×
ee_dbq_inquire2	×	×	○	×	○	○	×
ee_dbq_msgget	×	×	△	×	○	△	×
ee_dbq_msgput	×	×	△	×	○	△	×
ee_dbq_msgrefer	×	×	△	×	○	△	×
ee_dbq_msgrefer2	×	×	△	×	○	△	×
ee_dbq_obscancel	×	×	○	×	×	×	×
ee_dbq_obschdata	×	×	×	×	×	×	×
ee_dbq_obsend	×	×	×	×	×	×	×
ee_dbq_obsinquire	×	×	○	×	×	×	×
ee_dbq_obsrbskip	×	×	○	×	×	×	×
ee_dbq_obsrestart	×	×	○	×	×	×	×
ee_dbq_obsstrip	×	×	×	×	×	×	×
ee_dbq_obsstart	×	×	○	×	×	×	×
ee_dbq_obsstop	×	×	×	×	×	×	×
ee_dbq_rbskip	×	×	×	×	×	×	×
ee_dbq_skip	×	×	○	×	○	○	×

API 関数	トランザクション種別						
	UI	MV	RL	UT	CU	JI	BS
ee_dbq_srvctl	×	×	○	×	○	○	×
ee_lck_get	○	○	○	○	○	○	×
ee_lck_release_all	○	○	○	○	○	○	×
ee_lck_release_byname	○	○	○	○	○	○	×
ee_logprint	○	○	○	○	○	○	○
ee_mch_cmtsend	○	○	○	×	×	×	×
ee_mch_cmtsend_get※2	○	○	○	×	×	×	×
ee_mch_cmtsend_sync	○	○	○	×	×	×	×
ee_mch_otqbctl	○	○	○	×	×	×	×
ee_mch_otqinquire	○	○	○	×	×	×	×
ee_mch_otqskip	○	○	○	×	×	×	×
ee_mem_getwkseg	○	○	○	○	○	○	○
ee_mem_mdpstart	○	○	○	×	×	×	×
ee_mem_rlswkseg	○	○	○	○	○	○	○
ee_rpc_call	○	○	○	△	○	△	×
ee_rpc_call_to	○	○	○	△	○	△	×
ee_rpc_cmtsend	○	○	○	×	○	×	×
ee_rpc_cmtsend_to	○	○	○	×	○	×	×
ee_rpc_discard_further_replies	○	○	○	○	○	○	×
ee_rpc_discard_specific_reply	○	○	○	○	○	○	×
ee_rpc_get_error_descriptor	○	○	○	○	○	○	×
ee_rpc_get_watch_time	○	○	○	○	○	○	×
ee_rpc_poll_any_replies	○	○	○	○	○	○	×
ee_rpc_reply_send	○	○	○	○	○	○	×
ee_rpc_reply_suspend	×	×	×	×	×	×	×
ee_rpc_set_watch_time	○	○	○	○	○	○	×
ee_scd_clear_rollback_cnt	×	○	○	×	×	×	×
ee_scd_msg_receive	×	×	×	×	×	×	×
ee_scd_msg_rollback	×	×	×	×	×	×	×
ee_scd_msg_skip	○	○	○	×	×	×	×

API 関数	トランザクション種別						
	UI	MV	RL	UT	CU	JI	BS
ee_thd_abdctl	○	○	○	○	○	○	×
ee_thd_gettime	○	○	○	○	○	○	○
ee_thd_protend	○	○	○	○	○	○	×
ee_thd_protstart	○	○	○	○	○	○	×
ee_thd_thdcntget	○	○	○	○	○	○	×
ee_thd_uifaget	×	×	×	×	×	×	×
ee_tim_execap	○	○	○	○	○	○	×
ee_tim_execap_cancel	○	○	○	○	○	○	×
ee_tim_execap_cancel_all	○	○	○	○	○	○	×
ee_trb_ustrace_put	○	○	○	○	○	○	○
ee_trn_chained_commit	×	×	○	○	○	○	×
ee_trn_chained_rollback	○	○	○	○	○	○	×
ee_trn_getinfo	○	○	○	○	○	○	×
ee_trn_rm_close_mark	○	○	○	○	○	○	×
ee_trn_rollback_mark	○	○	○	○	○	○	×
ee_trn_set_exp_time	○	○	○	○	○	○	×
ee_trb_uinfo_tm	○	○	○	○	○	○	○

(凡例)

○：使用できます。

△：一部使用できます。詳細は各 API の説明を参照してください。

×

注※1

運用コマンド実行機能での、TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否については、「[2.6 運用コマンド実行機能](#)」を参照してください。

注※2

トランザクション種別が ES 以外の場合、この関数を発行しても一方送信メッセージ情報はありません。

4

TP1/EE および XTC のライブラリ関数の文法（C 言語編）

この章では、C 言語の UAP で使用できる TP1/EE および XTC のライブラリ関数の文法について説明します。

関数の説明形式

C 言語の UAP で使用できる TP1/EE および XTC のライブラリ関数の文法を、次の形式で説明します。

本文中で記載している ee_rpc_call 関数、ee_rpc_call_to 関数、ee_rpc_cmtsend 関数、および ee_rpc_cmtsend_to 関数は、クライアント UAP が TP1/EE の場合です。クライアント UAP が TP1/Server Base の場合は、dc_rpc_call 関数、dc_rpc_call_to 関数となります。

形式

TP1/EE および XTC のライブラリ関数の形式と、引数のデータ型を示します。

ANSI C の形式、C++言語でコーディングする場合は「ANSI C、C++の形式」に従ってください。

引数に値を設定するときは、ここで示すデータ型に従ってください。特に断りがないかぎり、引数には固有の名称を任意で付けられます。

機能

ここで説明する関数の機能について説明します。

UAP で値を設定する引数

関数の実行時に、値を指定しておく引数を示します。各引数の説明に従って、値を設定してください。

TP1/EE から値が返される引数

関数を実行したあとに、TP1/EE から値が返される引数（データ領域を含む）です。関数の実行後にこの引数の内容を参照してください。

クライアント UAP から値が渡される引数

サービス関数の場合で、クライアント UAP から値が渡される引数です。この引数の内容を参照して、サービス関数の処理をしてください。

サーバ UAP から値が返される引数

同期応答型 RPC の場合に、サービス関数から値が返される引数です。ee_rpc_call 関数を呼び出した UAP では、ここに示す引数の値を参照できます。

リターン値

関数を実行したときに戻ってくる値を、表形式で説明します。このリターン値によって、関数が正常に実行されたかがわかります。エラーが起こったときは、エラーの内容を示します。

UAP を作成するときは、互換性を保つために、必ずここで示す定数名でリターン値を使ってください。リターン値の定数名は、ヘッダファイルで定義されています。リターンされる値の情報が必要なときは、ヘッダファイルの定義を参照してください。

指定例

指定例が必要な関数には記述します。

注意事項

関数を使うときの注意について記述します。

サービス関数の作成 (SPP)

TP1/EE および XTC の UAP のサービス関数の文法について説明します。SPP はサービス関数を作成します。

形式 (TP1/EE の場合)

ANSI C, C++の形式

32 ビットアーキテクチャの場合

```
#include <eerpcsrv.h>

void 関数名(char *in, EEULONG *in_len, char *out, EEULONG *out_len,
EERPC_INTERFACE_TBL *trninf)
{
    サービスの処理
}

/* EERPC_INTERFACE_TBL構造体の形式 */
EERPC_INTERFACE_TBL {
    EEULONG    trn_len;
    EEULONG    group_len;
    char       servicegroup[32];
    EEULONG    service_len;
    char       service[32];
    char       trn_id[2];
    char       auto_block;
    char       commit_inf;
    EEULONG    thread_no;
    EEULONG    err_code;
    EEULONG    uifa_len;
    char       *uifa;
    EEULONG    usat_len;
    char       *usat;
    char       ans_inf;
    char       msg_inf;
    char       start_inf;
    char       before_end_inf;
    char       end_inf;
    char       thread_down_inf;
    char       uap_errtrn_inf;
    char       trninf_ex;
    EEULONG    rm_no;
    EEULONG    rm_inf;
    EEULONG    es_group_len;
    char       es_servicegroup[32];
    EEULONG    es_service_len;
    char       es_service[32];
    EEULONG    abn_thread_no;
    union {
        TRNINF_SERVICE_EX    *trninf_service_ex;
        TRNINF_DBQ_EX        *trninf_dbq_ex;
        TRNINF_DBQ_OBS_EX    *trninf_dbq_obs_ex;
    } ex_inf;
}
```

```

void          *xtc_pt;
void          *mcp_pt;
char          msg_type;
char          errtrn_factor;
char          run_inf;
char          rm_func;
EEULONG       rerun_cnt;
char          start_time[14];
char          node_id[5];
unsigned char  rtyrbk_cnt;
EEULONG       trnlevel_len;
char          trnlevel[32];
char          yobi_ff[24];
char          rtyrbk_area[32];
}

/* TRNINF_SERVICE_EX構造体の形式 */
TRNINF_SERVICE_EX {
    EEULONG       trn_len;
    EEULONG       service_out_len;
    char          yobi[192];
}

/* TRNINF_DBQ_EX構造体の形式 */
TRNINF_DBQ_EX {
    EEULONG       trn_len;
    char          dbqname[28];
    EEULONG       seq_num;
    unsigned short rollback_cnt;
    char          yobi[162];
}

/* TRNINF_DBQ_OBS_EX構造体の形式 */
TRNINF_DBQ_OBS_EX {
    EEULONG       trn_len;
    EEULONG       trn_kind;
    char          batchque_name[28];
    char          lot_name[24];
    EEULONG       cont_size;
    char          *cont_pt;
    EEULONG       lot_type;
    EEULONG       write_event_num;
    EEULONG       seq_num;
    EEULONG       end_code;
    EEULONG       sts_code;
    EEULONG       read_num;
    EEULONG       skip_num;
    EEULONG       unread_count;
    char          msglen_inf;
    char          yobi[99];
}

```

64 ビットアーキテクチャの場合

```

#include <eerpcsrv.h>

void 関数名(char *in, EEULONG *in_len, char *out, EEULONG *out_len,
EERPC_INTERFACE_TBL *trninf)

```

```

{
サービスの処理
}

/* EERPC_INTERFACE_TBL構造体の形式 */
EERPC_INTERFACE_TBL {
EEULONG    trn_len;
EEULONG    group_len;
char       servicegroup[32];
EEULONG    service_len;
char       service[32];
char       trn_id[2];
char       auto_block;
char       commit_inf;
EEULONG    thread_no;
EEULONG    err_code;
EEULONG    uifa_len;
char       yobi_1[4];
char       *uifa;
EEULONG    usat_len;
char       yobi_2[4];
char       *usat;
char       ans_inf;
char       msg_inf;
char       start_inf;
char       before_end_inf;
char       end_inf;
char       thread_down_inf;
char       uap_errtrn_inf;
char       trninf_ex;
EEULONG    rm_no;
EEULONG    rm_inf;
EEULONG    es_group_len;
char       es_servicegroup[32];
EEULONG    es_service_len;
char       es_service[32];
EEULONG    abn_thread_no;
EEULONG    trnlevel_len;
union {
    TRNINF_SERVICE_EX    *trninf_service_ex;
    TRNINF_DBQ_EX        *trninf_dbq_ex;
    TRNINF_DBQ_OBS_EX    *trninf_dbq_obs_ex;
} ex_inf;
void       *xtc_pt;
void       *mcp_pt;
char       msg_type;
char       errtrn_factor;
char       run_inf;
char       rm_func;
EEULONG    rerun_cnt;
EEULONG64  uifa_large_len;
EEULONG64  usat_large_len;
char       start_time[14];
char       node_id[5];
unsigned char rtyrbk_cnt;
char       trnlevel[32];
char       yobi_ff[4];
char       rtyrbk_area[32];

```

```

}

/* TRNINF_SERVICE_EX構造体の形式 */
TRNINF_SERVICE_EX {
    EEULONG          trn_len;
    EEULONG          service_out_len;
    char             yobi[192];
}

/* TRNINF_DBQ_EX構造体の形式 */
TRNINF_DBQ_EX {
    EEULONG          trn_len;
    char             dbqname[28];
    EEULONG          seq_num;
    unsigned short   rollback_cnt;
    char             yobi[162];
}

/* TRNINF_DBQ_OBS_EX構造体の形式 */
TRNINF_DBQ_OBS_EX {
    EEULONG          trn_len;
    EEULONG          trn_kind;
    char             batchque_name[28];
    char             lot_name[24];
    EEULONG          cont_size;
    char             *cont_pt;
    char             yobi_1[4];
    EEULONG          lot_type;
    EEULONG          write_event_num;
    EEULONG          seq_num;
    EEULONG          end_code;
    EEULONG          sts_code;
    EEULONG          read_num;
    EEULONG          skip_num;
    EEULONG          unread_count;
    char             msglen_inf;
    char             yobi[91];
}

```

形式 (XTC の場合)

ANSI C, C++の形式

64 ビットアーキテクチャの場合

```

#include <eerpcsrv.h>

void 関数名(char *in, EEULONG *in_len, char *out, EEULONG *out_len,
EERPC_INTERFACE_TBL *trninf)
{
    サービスの処理
}

/* EERPC_INTERFACE_TBL構造体の形式 */
EERPC_INTERFACE_TBL {
    EEULONG          trn_len;

```

```

EEULONG    group_len;
char        servicegroup[32];
EEULONG    service_len;
char        service[32];
char        trn_id[2];
char        auto_block;
char        commit_inf;
EEULONG    thread_no;
EEULONG    err_code;
EEULONG    uifa_len;
char        yobi_1[4];
char        *uifa;
EEULONG    usat_len;
char        yobi_2[4];
char        *usat;
char        ans_inf;
char        msg_inf;
char        start_inf;
char        before_end_inf;
char        end_inf;
char        thread_down_inf;
char        uap_errtrn_inf;
char        trninf_ex;
EEULONG    rm_no;
EELONG     rm_inf;
EEULONG    es_group_len;
char        es_servicegroup[32];
EEULONG    es_service_len;
char        es_service[32];
EEULONG    abn_thread_no;
char        yobi_3[4];
union {
    TRNINF_SERVICE_EX    *trninf_service_ex;
} ex_inf;
void        *xtc_pt;_
void        *mcp_pt;_
char        msg_type;
char        errtrn_factor;
char        yobi_4[6];
EEULONG64   uifa_large_len;
EEULONG64   usat_large_len;
char        yobi_ff[56];
}

```

/* TRNINF_SERVICE_EX構造体の形式 */

```

TRNINF_SERVICE_EX {
    EEULONG        trn_len
    EEULONG        service_out_len;
    char            yobi[192];
}

```

/* TRNINF_SERVICE_XTC構造体の形式 */

```

TRNINF_SERVICE_XTC {
    EEULONG        xtc_trn_len;
    char            system_inf;
    char            pre_online_inf;
    char            otq_auto_block;
    char            otq_auto_block_level;
}

```



```

EEULONG      otq_number;
EEULONG      err_code_ex;
EEULONG      itq_number;
unsigned short rollback_cnt;
char          online_inf;
char          yobi_xx[169];
}

```

機能

サービスを実行して結果を返す SPP のサービス関数です。クライアント UAP で ee_rpc_call 関数や ee_rpc_cmtsend 関数を呼び出したとき、TP1/EE プロセスの開始・終了などによってトランザクションが発生したとき、DB キュー機能でキューを読み込んだときに呼び出されます。サービス関数は、上記の形式で任意に作成してください。

サービス関数名は、サービス関数のエントリポイント（入り口点）の名称に対応します。この対応づけは、UAP の実行環境を設定するときに指定します。UAP の実行環境を設定する方法を次に示します。

- TP1/EE および XTC の場合
ユーザサービス関連定義で指定します。

引数の指定

サービス関数には、次の値が引数として渡されます。

- 入力パラメタ (in)
- 入力パラメタ長 (in_len)
- 応答の長さ (out_len)
- トランザクションインタフェース情報 (trninf)

入力パラメタと入力パラメタの長さはクライアント UAP で設定した値が、文字コードや数字の表現形式などを変換しないでサービス関数にそのまま渡されます。応答の長さには、クライアント UAP で設定した長さが渡されます。

ERRTRNR による RPC 応答メッセージ送信機能を使用している場合で、拡張トランザクション種別が EERPC_SERVICE_TRN_EX のときは、次のようになります。

- ERRTRNR のサービス関数には、次の値が引数として渡されます。
 - 入力パラメタ (in)
 - 入力パラメタ長 (in_len)
 - サービス関数の応答 (out)
 - サービス関数で設定した応答の長さ (out_len)
 - トランザクションインタフェース情報 (trninf)

- ERRTRNR のサービス関数の応答には、サービス関数でクライアント UAP が設定したサービス関数の応答 (out)、およびサービス関数で設定した応答の長さ (out_len) が渡されます。

サービス関数に値が渡される引数を次の表に示します。

表 4-1 サービス関数に値が渡される引数

引数	トランザクション種別												
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
入力パラメタ	×	×	○	○	○	○	○	○	○	○	×	×	×
入力パラメタ長	×	×	○	○	○	○	○	○	○	○	×	×	×
サービス関数の応答	×	×	×	×	×	×	×	×	○*	×	×	×	×
応答の長さ	×	×	○	○	○	○	×	×	○	×	×	×	×
トランザクションインタフェース情報	○	○	○	○	○	○	○	○	○	○	○	○	○

(凡例)

- ：参照できます。
- ×：参照できません。

注※

ERRTRNR による RPC 応答メッセージ送信機能を使用している場合で、拡張トランザクション種別が EERPC_SERVICE_TRN_EX のときだけ参照できます。

サービス関数では、次の値を引数に設定します。

- サービス関数の応答 (out)
- サービス関数の応答の長さ (out_len)

サービス関数は、out に応答を設定して、その長さを out_len に設定してからリターンしてください。

サービスのクライアント UAP への応答は、サービス関数がコミットしたかロールバックしたかには無関係に送信されます。必要であればサービス関数でクライアント UAP にエラーを知らせる応答を作成してください。

クライアント UAP から値が渡される引数

●in

クライアント UAP で設定した、入力パラメタが渡されます。

ただし、トランザクション種別が E4、ES、または TM の場合は次の表に示す値が渡されます。

表 4-2 トランザクション種別が E4, ES, または TM の場合に渡される値 (in)

トランザクション種別	渡される値
E4	ee_tim_execap 関数が起動するトランザクションに渡すデータ (data) に設定した値
ES	トランザクションと同期して送信する非同期型のメッセージ送信 API (ee_rpc_cmtsend 関数など) の入力パラメタ (in) に設定した値
TM	ee_tim_execap 関数が起動するトランザクションに渡すデータ (data) に設定した値

●in_len

クライアント UAP で設定した、入力パラメタの長さが渡されます。

ただし、トランザクション種別が E4, ES, または TM の場合は次の表に示す値が渡されます。

表 4-3 トランザクション種別が E4, ES, または TM の場合に渡される値 (in_len)

トランザクション種別	渡される値
E4	ee_tim_execap 関数が起動するトランザクションに渡すデータ (dataleng) に設定した値
ES	トランザクションと同期して送信する非同期型のメッセージ送信 API (ee_rpc_cmtsend 関数など) の入力パラメタ (in_len) に設定した値
TM	ee_tim_execap 関数が起動するトランザクションに渡すデータ (dataleng) に設定した値

●out_len

クライアント UAP で設定した、応答の長さが渡されます。

TP1/EE から値が返される引数

●trninf

トランザクションインタフェース情報が返されます。詳細については、この章の「[トランザクションインタフェース情報](#)」を参照してください。

●TRNINF_SERVICE_XTC

XTC 拡張トランザクションインタフェース情報が返されます。この章の「[トランザクションインタフェース情報](#)」を参照してください。

UAP で値を設定する引数

●out

サービス関数からの応答を設定します。サービス関数の処理では、out にサービス関数での処理結果を設定してからリターンしてください。

●out_len

サービス関数からの、実際の応答の長さを設定します。クライアント UAP から渡された out_len の値以下の数値を設定してください。応答不要（非応答型 RPC）の場合は、0 を設定してください。

サービス関数の処理での注意

1. クライアントから応答不要で呼び出されるサービス関数では、out と out_len は参照できません。クライアントから応答不要で呼び出されるサービス関数を次に示します。
 - 非応答型 RPC で呼び出される RPC のサービス関数
 - DB キューのサービス関数
 - オンラインバッチのサービス関数
2. サービス関数を C 言語で記述した場合は、static 変数には前回のサービス要求時の値が残っているので、必要であれば初期化してから使用してください。
3. サービス関数では exit(), fork(), exec() を使わないでください。これらの関数を使った場合の UAP の動作は保証されません。

トランザクションとサービス関数の関係

グローバルトランザクションのサービス全体は、コミットするかロールバックとなるかのどちらかが保証されています。トランザクションブランチとして実行されるサービス関数がリターン（return）することで、そのトランザクションブランチの正常終了を要求したと見なされます。

リターン値

リターン値はありません。return() で指定した値はクライアント UAP には返りません。TP1/EE でもリターン値を参照しません。リターン値として -1 を指定しても、ロールバックを要求したことにはなりません。

トランザクションインタフェース情報

SPP の引数には、TP1/EE からトランザクションインタフェース情報が渡されます。TP1/EE から SPP に渡されるトランザクションインタフェース情報を次の表に示します。

表 4-4 トランザクションインタフェース情報

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	U T	C U	JI	BS
情報長 (trn_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
サービスグループ名長 (group_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○ ※1
サービスグループ名 (servicegroup)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○ ※1
サービス名長 (service_len)	×	×	○	○	○	○	○	×	○	○	×	○	○	×	○	×	×
サービス名 (service)	×	×	○	○	○	○	○	×	○	○	×	○	○	×	○	×	×
トランザクション種別 (trn_id)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
自動閉塞有無 (auto_block)	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×	×
コミット有無 (commit_inf)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
処理スレッド通番 (thread_no)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
エラー要因コード (err_code)	×	×	×	×	×	○	×	○	○	×	×	×	×	×	×	×	×
ユーザ IFA エリアサイズ (uifa_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ IFA エリアポインタ (*uifa)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ SAT エリアサイズ (usat_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ SAT エリアポインタ (*usat)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
応答要否 (ans_inf)	×	×	○	×	×	×	×	×	○	×	×	×	○	×	○	×	×
メッセージ状態 (msg_inf)	×	×	○	○	○	○	○	○	○	○	×	×	○	×	○	×	×

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL	UT	CU	JI	BS
プロセス起動要因 (start_inf)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
前回プロセス終了要因 (before_end_inf)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
プロセス終了要因 (end_inf)	×	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
スレッドダウン種別 (thread_down_inf)	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×	×
異常発生トランザクション種別 (uap_errtrn_inf)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
拡張トランザクション種別 (trninf_ex)	×	×	○	×	×	○	×	×	○	×	×	×	×	×	○	×	×
リソースマネージャ数 (rm_no)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
リソースマネージャ接続状態 (rm_inf)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×
送信サービスグループ名長 (es_group_len)	×	×	×	×	×	×	×	○	×	×	×	○	×	×	×	×	×
送信サービスグループ名 (es_servicegroup)	×	×	×	×	×	×	×	○	×	×	×	○	×	×	×	×	×
送信サービス名長 (es_service_len)	×	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×
送信サービス名 (es_service)	×	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×
異常発生処理スレッド通番 (abn_thread_no)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
拡張トランザクションインタフェース情報 (ex_inf)	×	×	△	×	×	△	×	×	△	×	×	×	×	×	△	×	×
XTC 拡張トランザクションインタフェース情報 (*xtc_pt) ※2	○	○	○	○	○	○	○	○	○	○	○	○	△	×	×	×	×

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL	UT	CU	JI	BS
MCP 拡張トランザクションインタフェース情報 (*mcp_pt)	×	○※3	×	○※3	○※3	○※3	×	○※3	○※3	×	×	×	○	×	×	×	×
メッセージ種別 (msg_type)	×	×	○	○	○	○	×	○	○	×	×	×	×	×	○	×	×
エラートランザクション要因 (errtm_factor)	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×
拡張ユーザ IFA エリアサイズ (uifa_large_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
拡張ユーザ SAT エリアサイズ (usat_large_len)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
トランザクション起動要因 (run_inf)	×	×	×	○	×	×	×	×	×	×	×	×	×	○	○	○	×
サーバ再起動回数 (rerun_cnt)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
トランザクション起動時刻 (start_time)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
送信元のノード識別子 (node_id)	×	×	○	○	○	○	×	×	○	×	×	×	×	×	×	×	×
リトライ回数 (rtyrbk_cnt)	×	×	○	×	×	×	×	×	×	○	×	×	×	×	○	×	×
トランザクションレベル名長 (trnlevel_len)	×	×	○	○	×	○	×	×	×	○	×	×	○	×	○	×	×
トランザクションレベル名 (trnlevel)	×	×	○	○	×	○	×	×	×	○	×	×	○	×	○	×	×
トランザクション引き継ぎ情報 (rtyrbk_area)	×	×	○	×	×	○	×	×	×	○	×	×	×	×	○	×	×

(凡例)

○：参照できます。

△：拡張トランザクション種別が NULL 以外の場合は参照できます。

×：参照できません。

注※1

サービスグループ名の代わりにジョブ名の情報が設定されます。

注※2

XTC を使用する場合にだけ有効です。

注※3

メッセージ種別 (msg_type) が EEMCP_MSGTYPE_TCP または EEMCP_MSGTYPE_UDP の場合にだけ参照できます。

上記の表のトランザクションインタフェース情報について、次に説明します。なお、TP1/FSP を使用する場合は、ここで説明していない内容が設定されることがあります。詳細については、TP1/FSP の関連ドキュメントを参照してください。

●情報長

トランザクションインタフェース情報の長さが設定されます (単位: バイト)。

情報長 (LL)	その他のトランザクションインタフェース情報
LL	

●サービスグループ名長

サービスグループ名の長さが設定されます (単位: バイト)。

●サービスグループ名

該当するトランザクションのサービスグループ名が設定されます。

●サービス名長

サービス名の長さが設定されます (単位: バイト)。

●サービス名

サービス名が設定されます。

●トランザクション種別

トランザクション種別が設定されます。

EERPC_TRNKIND_MI

初期化トランザクション

EERPC_TRNKIND_ME

終了トランザクション

EERPC_TRNKIND_MN

サービストランザクション

EERPC_TRNKIND_E1

エラートランザクション (ERRTRN1)

EERPC_TRNKIND_E2

エラートランザクション (ERRTRN2)

EERPC_TRNKIND_E3

エラートランザクション (ERRTRN3)

EERPC_TRNKIND_E4

エラートランザクション (ERRTRN4)

EERPC_TRNKIND_ES

エラートランザクション (ERRTRNS)

EERPC_TRNKIND_ER

エラートランザクション (ERRTRNR)

EERPC_TRNKIND_TM

タイマトランザクション

EERPC_TRNKIND_UI

実行系連絡トランザクション

EERPC_TRNKIND_MV

イベント通知トランザクション

EERPC_TRNKIND_RL

MCP 後処理トランザクション

●自動閉塞有無

サービスの自動閉塞の有無が設定されます。

EERPC_SRV_AUTOABD

自動閉塞有

EERPC_SRV_AUTOABD_NONE

自動閉塞無

●コミット有無

エラーになったトランザクションの同期点の有無が設定されます。

EERPC_COMMIT_NONE

コミット無

EERPC_COMMIT

コミット有

エラーになったサービス処理で一度でも同期点を取得している場合は、コミット有が設定されます。

●処理スレッド通番

該当するトランザクションのスレッド通番が設定されます。

●エラー要因コード

エラーの要因コードが設定されます。

スレッドダウンの場合は、その詳細が設定されます。この場合、スレッドダウン種別によって設定情報が異なります。

- 同期シグナルの場合：シグナル番号が設定されます。
- トランザクションタイムの場合：値は設定されません。
- ロールバック要求（スレッド強制終了指示）の場合：ee_trn_chained_rollback 関数の引数 ecode の値が設定されます。
- DB キューの交替用メッセージ表書き込み機能使用時のロールバック要求（スレッド強制終了指示）の場合：次の情報が設定されます。
EEDBQ_ERRINF_WRITE：書き込み処理でエラー(5)
EEDBQ_ERRINF_READ：読み出し処理でエラー(6)
- 上記以外の場合：値は設定されません。

トランザクション種別がエラートランザクション（EERTNR）の場合は、次に示す情報が設定されます。

EERPC_ERRINF_ROLLBACK

UAP リターン後、トランザクションはロールバックしました。

EERPC_ERRINF_HEURISTIC

UAP リターン後、トランザクションはヒューリスティック決着しました。

EERPC_ERRINF_NET_DOWN

UAP リターン後の同期点処理で通信障害が発生したため、トランザクションは未決着です。

EERPC_ERRINF_RM_DOWN

UAP リターン後の同期点処理でリソースマネージャ障害が発生したため、トランザクションは未決着です。

EERPC_ERRINF_XDB_BEFORE_ERR

UAP リターン後、XDB 前処理エラーによってロールバックしました。

EERPC_ERRINF_XDB_ERR

UAP リターン後、XDB 処理エラーによってロールバックしました。

EERPC_ERRINF_SENDSYNC_ERR

UAP リターン後、トランザクション同期のメッセージの送信失敗によってロールバックしました。

トランザクション種別がエラートランザクション（EERTNS）の場合は、次に示す情報が設定されます。

EERPC_ERRINF_SENDSYNC_ERR

UAP リターン後、トランザクション同期のメッセージの送信失敗によってロールバックしました。

EERPC_ERRINF_SEND_ERR

UAP リターン後、トランザクション非同期のメッセージの送信失敗によってロールバックしました。

●ユーザ IFA エリアサイズ

ユーザ IFA エリアのサイズが設定されます（単位：バイト）。

64 ビットアーキテクチャの場合、ユーザ IFA エリアのサイズが 4 ギガバイト以上のときは 0 が設定されます。この場合、uifa_large_len を参照してユーザ IFA エリアのサイズを取得してください。ユーザ IFA エリアのサイズが 4 ギガバイト未満の場合でも、uifa_large_len を参照してユーザ IFA エリアのサイズを取得できます。

●ユーザ IFA エリアポインタ

ユーザ IFA エリアのポインタが設定されます。ユーザ IFA エリアサイズが 0 の場合は、値を参照できません。

●ユーザ SAT エリアサイズ

ユーザ SAT エリアのサイズが設定されます（単位：バイト）。

64 ビットアーキテクチャの場合、ユーザ SAT エリアのサイズが 4 ギガバイト以上のときは 0 が設定されます。この場合、usat_large_len を参照してユーザ SAT エリアのサイズを取得してください。ユーザ SAT エリアのサイズが 4 ギガバイト未満の場合でも、usat_large_len を参照してユーザ SAT エリアのサイズを取得できます。

●ユーザ SAT エリアポインタ

ユーザ SAT エリアのポインタが設定されます。ユーザ SAT エリアサイズが 0 の場合は、値を参照できません。

●応答要否

クライアント UAP からの RPC 要求に対する応答要否が設定されます。

EERPC_REPLY

応答要

EERPC_REPLY_NONE

応答不要

●メッセージ状態

引き渡しメッセージの状態が設定されます。

EERPC_MSGINF_NORMAL

正常

EERPC_MSGINF_OVERFLOW

オーバフロー

オーバフローの場合、SPP に渡される入力パラメタにはメッセージの一部しか設定されません。

●プロセス起動要因

TP1/EE プロセス起動時の要因が設定されます。

EERPC_START_STS_NORMAL

通常起動

EERPC_START_STS_RERUN

再起動

●前回プロセス終了要因

前回の TP1/EE プロセス終了時の要因が設定されます。

EERPC_BEEND_STS_NORMAL

正常終了

EERPC_BEEND_STS_PLANA

計画停止 A

EERPC_BEEND_STS_PLANB

計画停止 B

EERPC_BEEND_STS_FORCE

異常終了

●プロセス終了要因

TP1/EE プロセス終了時の要因が設定されます。

EERPC_END_STS_NORMAL

正常終了

EERPC_END_STS_PLANA

計画停止 A

EERPC_END_STS_PLANB

計画停止 B

EERPC_END_STS_ISOLATEA

実行系孤立モード A

EERPC_END_STS_ISOLATEB

実行系孤立モード B

●スレッドダウン種別

スレッドダウン種別が設定されます。

EERPC_THDDOWN_SIGNAL

同期シグナル

EERPC_THDDOWN_TIMER

トランザクションタイマ

EERPC_THDDOWN_ROLLBACK

ロールバック要求（スレッド強制終了指示）

EERPC_THDDOWN_ROLLBACKLIMITOVER

ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数の呼び出し回数の上限値オーバ

EERPC_THDDOWN_INSTRUCTION

スレッドダウン指示による終了

リトライロールバック前処理 UOC でリターン値 EETRN_UOC_NORETRY_KILL,
EETRN_UOC_NORETRY_ABORT 設定時

EERPC_THDDOWN_COMMITLIMITOVER

ee_trn_chained_commit 関数の呼び出し回数の上限値オーバ

EERPC_THDDOWN_XDBERROR

XDB の障害

EERPC_THDDOWN_UNKNOWN

上記以外

●異常発生トランザクション種別

異常が発生したトランザクション（要求元トランザクション）の種別が設定されます。

EERPC_UAPABN_MN

サービストランザクション

EERPC_UAPABN_TM

タイマトランザクション

EERPC_UAPABN_RM

再起動トランザクション

●拡張トランザクション種別

拡張トランザクション種別が設定されます。

EERPC_SERVICE_TRN_EX

拡張トランザクションインタフェース情報（DB キュー機能およびオンラインバッチ機能を使用しない
場合）

EERPC_TRN_EX_DBQ

拡張トランザクションインタフェース情報（DB キュー機能を使用する場合）

EERPC_TRN_EX_DBQ_OBS

拡張トランザクションインタフェース情報（オンラインバッチ機能を使用する場合）

拡張トランザクション種別が NULL 以外の場合は、トランザクションインタフェース情報の一部として拡張トランザクションインタフェース情報が設定されます。

●リソースマネージャ数

接続しているリソースマネージャの数が設定されます。

●リソースマネージャ接続状態

リソースマネージャとの接続状態が設定されます。

EERPC_RM_CONNECT

リソースマネージャ接続正常

EERPC_RM_ERROR

リソースマネージャ接続失敗によるトランザクション縮退起動

複数のリソースマネージャと接続している場合、一部のリソースマネージャでの接続失敗も含みます。

●送信サービスグループ名長

トランザクション種別が EERPC_TRNKIND_ES の場合

ee_rpc_cmtsend 関数, ee_mch_cmtsend 関数, または ee_mch_cmtsend_sync 関数で指定したサービスグループ名の長さが設定されます (単位: バイト)。

トランザクション種別が EERPC_TRNKIND_MV の場合

サービスグループレベルで出力キュー (OTQ) が自動閉塞していたサービスグループ名の長さが設定されます (単位: バイト)。

●送信サービスグループ名

トランザクション種別が EERPC_TRNKIND_ES の場合

ee_rpc_cmtsend 関数, ee_mch_cmtsend 関数, または ee_mch_cmtsend_sync 関数で指定したサービスグループ名が設定されます。

トランザクション種別が EERPC_TRNKIND_MV の場合

サービスグループレベルで出力キュー (OTQ) が自動閉塞していたサービスグループ名が設定されます。

●送信サービス名長

ee_rpc_cmtsend 関数, ee_mch_cmtsend 関数, または ee_mch_cmtsend_sync 関数で指定したサービス名の長さが設定されます (単位: バイト)。

●送信サービス名

ee_rpc_cmtsend 関数, ee_mch_cmtsend 関数, または ee_mch_cmtsend_sync 関数で指定したサービス名が設定されます。

●異常発生処理スレッド通番

異常が発生したトランザクション（要求元トランザクション）のスレッド通番が設定されます。

●拡張トランザクションインタフェース情報

拡張トランザクションインタフェース情報が設定されます。拡張トランザクションインタフェース情報は、DB キュー機能およびオンラインバッチ機能を使用するかどうかによって設定される情報が異なります。それぞれの場合について説明します。

DB キュー機能およびオンラインバッチ機能を使用しない場合

DB キュー機能およびオンラインバッチ機能を使用しない場合，トランザクションインタフェース情報の一部として，次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 4-5 拡張トランザクションインタフェース情報（DB キュー機能およびオンラインバッチ機能を使用しない場合）

拡張トランザクションインタフェース 情報（変数名）	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長（trn_len）	×	×	×	×	×	×	×	×	○	×	×	×	×
サービス関数起動時の応答の長さ （service_out_len）	×	×	×	×	×	×	×	×	○	×	×	×	×

（凡例）

- ：参照できます。
- ×

・情報長

拡張トランザクションインタフェース情報の長さが設定されます（単位：バイト）。



・サービス関数起動時の応答の長さ

クライアント UAP で設定したサービス関数起動時の応答の長さ（out_len）が設定されます。

DB キュー機能を使用する場合

DB キュー機能を使用する場合，トランザクションインタフェース情報の一部として，次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 4-6 拡張トランザクションインタフェース情報 (DB キュー機能を使用する場合)

拡張トランザクションインタフェース 情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長 (trn_len)	×	×	○	×	×	○	×	×	○	×	×	×	×
読み込み DB キュー名 (dbqname)	×	×	○	×	×	○	×	×	○	×	×	×	×
通番 (読み込み) (seq_num)	×	×	○	×	×	○	×	×	×	×	×	×	×
ロールバック回数 (rollback_cnt)	×	×	○	×	×	○	×	×	○	×	×	×	×

(凡例)

○：参照できます。

×

・ 情報長

拡張トランザクションインタフェース情報の長さが設定されます (単位：バイト)。



・ 読み込み DB キュー名

読み込み時の DB キュー名が設定されます。

・ 通番

読み込みメッセージの通番が設定されます。

・ ロールバック回数

同じメッセージに対して ee_trn_chained_rollback 関数および ee_trn_rollback_mark 関数でロールバックした回数が設定されます。65535 を超えた場合は 65535 が設定されます。

同一メッセージ処理の連続ロールバック監視回数 (DB キュー機能関連定義の dbq_rollback_retry_count オペランドの指定値) を超えたためにスケジュールを停止した場合、ロールバック回数は 0 に戻ります。

オンラインバッチ機能を使用する場合

オンラインバッチ機能を使用する場合、トランザクションインタフェース情報の一部として、次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 4-7 拡張トランザクションインタフェース情報 (オンラインバッチ機能を使用する場合)

拡張トランザクションインタフェース 情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長 (trn_len)	×	×	○	×	×	○	×	×	○	×	×	×	×

拡張トランザクションインタフェース 情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
トランザクション種別 (trn_kind)	×	×	○	×	×	○	×	×	○	×	×	×	×
バッチキュー名 (batchque_name)	×	×	○	×	×	○	×	×	○	×	×	×	×
ロット名 (lot_name)	×	×	○	×	×	○	×	×	○	×	×	×	×
引き継ぎの情報サイズ (cont_size)	×	×	○	×	×	×	×	×	×	×	×	×	×
引き継ぎの情報アドレス (*cont_pt)	×	×	○	×	×	×	×	×	×	×	×	×	×
オンラインバッチ処理形態 (lot_type)	×	×	○	×	×	○	×	×	○	×	×	×	×
該当するロットの全メッセージ数 (write_event_num)	×	×	△	×	×	△	×	×	△	×	×	×	×
読み込んだメッセージの通番 (seq_num)	×	×	△	×	×	△	×	×	×	×	×	×	×
ロット終了要因コード (end_code)	×	×	△	×	×	×	×	×	×	×	×	×	×
終了時のステータスコード (sts_code)	×	×	△	×	×	×	×	×	×	×	×	×	×
読み出し済みメッセージ数 (read_num)	×	×	△	×	×	△	×	×	△	×	×	×	×
スキップメッセージ数 (skip_num)	×	×	△	×	×	△	×	×	△	×	×	×	×
未読み出しメッセージ数 (unread_count)	×	×	△	×	×	△	×	×	△	×	×	×	×
メッセージ長情報 (msglen_inf)	×	×	△	×	×	△	×	×	△ ※	×	×	×	×

(凡例)

○：参照できます。

△：トランザクションの種別や、使用するオンラインバッチ処理の処理方法によって異なります。
詳細については、表 4-6 を参照してください。

×

注※

サービス関数の入力パラメタ長 (in_len) が 0 以外の場合だけ参照できます。

表 4-8 拡張トランザクションインタフェース情報の詳細（オンラインバッチ機能を使用する場合）

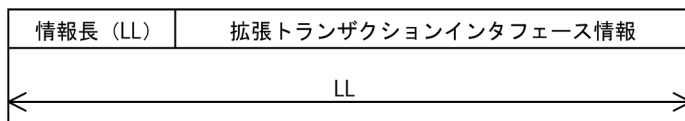
トランザクション種別	バッチ処理トランザクション		バッチ終了トランザクション	
オンラインバッチ処理形態	データ型	イベント型	データ型	イベント型
該当するロットの全メッセージ数	○	×	○	×
読み込んだメッセージの通番	○	×	×	×
ロット終了要因コード	×	×	○	○
終了時のステータスコード	×	×	○	○
読み出し済みメッセージ数	×	×	○	×
スキップメッセージ数	×	×	○	×
未読み出しメッセージ数	×	×	○	×
メッセージ長情報	○	×	×	×

（凡例）

- ：参照できます。
- ×

・ 情報長

拡張トランザクションインタフェース情報の長さが設定されます（単位：バイト）。



・ トランザクション種別

トランザクション種別が設定されます。

EEDBQ_OBS_TRNKIND_CNTL

バッチ処理トランザクション

EEDBQ_OBS_TRNKIND_END

バッチ終了トランザクション

・ バッチキュー名

バッチキュー名が設定されます。

・ ロット名

ロット名が設定されます。

・ 引き継ぎ情報のサイズ

ee_dbq_obsstart 関数、および ee_dbq_obschdata 関数で設定したロットの引き継ぎ情報のサイズが設定されます。

・ 引き継ぎ情報のアドレス

ee_dbq_obsstart 関数、および ee_dbq_obschdata 関数で設定したロットの引き継ぎ情報のエリアアドレスが設定されます。

- オンラインバッチ処理形態

実行中のロットのオンラインバッチ処理形態が設定されます。

EEDBQ_OBS_TYPE_DATA

データ型オンラインバッチ処理

EEDBQ_OBS_TYPE_EVNT

イベント型オンラインバッチ処理

- 該当するロットの全メッセージ数

該当するロットの全メッセージ数が設定されます。該当するロットのバッチ処理トランザクションには、同じ値が設定されます。

回復処理の失敗による中断、および回復処理の失敗による中断の強制終了要求の場合は、設定されません。

- 読み込んだメッセージの通番

該当するロットで読み出したメッセージ通番が設定されます。

- ロット終了要因コード

該当するロットの終了要因が設定されます。ユーザ指定による処理の中断と、障害による処理の中断の両方が発生した場合は、障害による処理の中断が優先して設定されます。

EEDBQ_OBS_END_NORMAL

正常終了

EEDBQ_OBS_USER_SUSPEND

ユーザ指定による中断

EEDBQ_OBS_ABND_SUSPEND

障害による中断

EEDBQ_OBS_END_FORCE

強制終了要求

EEDBQ_OBS_RRN_SUSPEND

回復処理の失敗による中断

EEDBQ_OBS_END_FORCE_RRN

回復処理の失敗による中断中の強制終了要求

- 終了時のステータスコード（理由コード）

処理終了時のステータスコード（理由コード）が設定されます。

障害または回復処理の失敗による処理の中断の場合に設定されます。強制終了要求および回復処理の失敗による中断中の強制終了要求の場合は、以前に設定された値が引き継がれます。複数の障害が重なった場合は、範囲が大きい方の障害が優先して設定されます。そのほかの場合は NULL が設定されます。

EEDBQ_OBS_STATUS_DATAFORM

データフォーマット不正

EEDBQ_OBS_STATUS_OBSERR

オンラインバッチ機能処理中のエラー

EEDBQ_OBS_STATUS_ROLLBACK

次に示す要因以外でのロールバック

- ・ UAP での thkind に EETR_N_KEEP を設定した ee_trn_chained_rollback 関数の呼び出し
- ・ UAP での ee_trn_rollback_mark 関数の呼び出し
- ・ データフォーマット不正
- ・ オンラインバッチ機能処理中のエラー
- ・ DB キューのリソースマネージャへのアクセスエラー
- ・ スレッドダウン
- ・ プロセスダウン

EEDBQ_OBS_STATUS_RMERR

DB キューのリソースマネージャへのアクセスエラー

EEDBQ_OBS_STATUS_THDDOWN

スレッドダウン

EEDBQ_OBS_STATUS_PROCDOWN

プロセスダウン

- 読み出し済みメッセージ数

該当するロットの読み出し済みメッセージ数が設定されます。

回復処理の失敗による中断、および回復処理の失敗による中断の強制終了要求の場合は、設定されません。

- スキップメッセージ数

該当するロットで読み出しをスキップしたメッセージ数が設定されます。

回復処理の失敗による中断、および回復処理の失敗による中断の強制終了要求の場合は、設定されません。

- 未読み出しメッセージ数

該当するロットの未読み出しメッセージ数が設定されます。

回復処理の失敗による中断、および回復処理の失敗による中断の強制終了要求の場合は、設定されません。

- メッセージ長情報

メッセージの前にユーザが設定するメッセージ長と、実際に入力したメッセージ長が一致しているかどうかを確認するための情報が設定されます。

EEDBQ_OBS_MSGLEN_EQUAL

一致しているか、またはオーバフローしています。オーバフローしているかどうかは、トランザクションインタフェース情報のメッセージ状態を参照して確認してください。

EEDBQ_OBS_MSGLEN_DATA

ユーザが設定するメッセージ長が実際に入力したデータ長より大きいため、一致していません。
実際に入力したデータ長によって、入力パラメタ (in) にメッセージが設定されます。

EEDBQ_OBS_MSGLEN_LL

ユーザが設定するメッセージ長が実際に入力したデータ長より小さいため、一致していません。
メッセージの前に設定されたメッセージ長によって、入力パラメタ (in) にメッセージが設定されます。

●XTC 拡張トランザクションインタフェース情報

XTC 拡張トランザクションインタフェース情報が設定されます。次の表に示す XTC 拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 4-9 XTC 拡張トランザクションインタフェース情報

XTC 拡張トランザクションインタフェース情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
系情報 (system_inf)	○	○	×	×	×	×	×	×	×	×	×	×	×
系切り替え前の実行系の状態 (pre_online_inf)	×	×	×	×	×	×	×	×	×	×	○	×	×
出力キュー (OTQ) 自動閉塞 (otq_auto_block)	×	×	×	×	×	×	×	○	×	×	×	×	×
出力キュー (OTQ) 自動閉塞レベル (otq_auto_block_level)	×	×	×	×	×	×	×	○	×	×	×	×	×
出力キュー (OTQ) 通番 (otq_number)	×	×	×	×	×	×	×	○	×	×	×	×	×
エラー要因詳細 (err_code_ex)	×	×	×	×	×	×	×	○	○	×	×	×	×
入力キュー (ITQ) 通番 (itq_number)	×	×	○	×	×	×	×	×	×	○	×	×	×
ロールバックリトライ回数 (rollback_cnt)	×	×	○	×	×	×	×	×	×	○	×	×	×
系切り替え後の実行系の状態 (online_inf)	×	×	×	×	×	×	×	×	×	×	○	×	×

(凡例)

- ：参照できます。
- ×

• 系情報

トランザクションを起動した XTC の系が設定されます。

EERPC_SYSTEM_ONLINE

実行系

EERPC_SYSTEM_STANDBY

待機系

- 系切り替え前の実行系の状態

系切り替えが発生したときの、系切り替え前に実行系だった XTC の状態が設定されます。

EERPC_STATUS_ONLINE

オンライン中

EERPC_STATUS_BEFORE_ME

終了処理中、かつ終了トランザクション起動前、または終了トランザクション起動中

- 出力キュー (OTQ) 自動閉塞有無

メッセージ送信失敗による出力キューの自動閉塞の有無が設定されます。

EERPC_OTQ_AUTOABD

自動閉塞有り

EERPC_OTQ_AUTOABD_NONE

自動閉塞無し

- 出力キュー (OTQ) 自動閉塞レベル

メッセージ送信失敗による出力キューの自動閉塞のレベルが設定されます。

EERPC_OTQ_AUTOABD_NONE

自動閉塞無し

EERPC_OTQ_AUTOABD_GROUP

サービスグループレベル

EERPC_OTQ_AUTOABD_SERVICE

サービスレベル

- 出力キュー (OTQ) 通番

トランザクション非同期のメッセージ送信に失敗したときのメッセージの出力キューの通番が設定されます。

- エラー要因詳細

エラー詳細情報が設定されます。

トランザクション種別が ES、エラー要因コードが EERPC_ERRINF_SENDSYNC_ERR、または EERPC_ERRINF_SEND_ERR の場合

EERPC_ERRINFS_SEND_ERR_PACKET

送信障害 (パケットサイズ不正などの RPC 関連定義の不正)

EERPC_ERRINFS_SEND_ERR_RESOURCE

送信障害 (リソース不足)

EERPC_ERRINFS_SEND_ERR_STATUS

送信障害 (送信先受信不可状態)

EERPC_ERRINFS_SEND_ERR

送信障害（上記以外）

EERPC_ERRINFS_SEND_ERR_ACK_TIMEOUT

送達確認待ちタイムアウト

EERPC_ERRINFS_SEND_ERR_ACK_ERR

送達確認待ち受信障害

EERPC_ERRINFS_SEND_ERR_ACK_RESOURCE

送達確認エラーメッセージ受信（リソース不足）

EERPC_ERRINFS_SEND_ERR_ACK_CLSYNC

送達確認エラーメッセージ受信（CL 同期エラー）

EERPC_ERRINFS_SEND_ERR_ACK_HOLD

送達確認エラーメッセージ受信（送信先サービス閉塞），または自プロセスあて送信時のサービス閉塞

EERPC_ERRINFS_SEND_ERR_ACK_GRP

送達確認エラーメッセージ受信（送信先サービスグループ名不正）

EERPC_ERRINFS_SEND_ERR_ACK_SRV

送達確認エラーメッセージ受信（送信先サービス名不正）

EERPC_ERRINFS_SEND_ERR_ACK_TYPE

送達確認エラーメッセージ受信（メッセージ送信形態不正／オプション不正）

EERPC_ERRINFS_SEND_ERR_ACK

送達確認エラーメッセージ受信（上記以外）

EERPC_ERRINFS_SEND_OTQ_BLOCKADE

出力キュー（OTQ）の閉塞中

EERPC_ERRINFS_SEND_ERR_MYPROC_PUTPCE

処理キュー登録失敗

トランザクション種別が ER，エラー要因コードが EERPC_ERRINF_XDB_BEFORE_ERR の場合

EERPC_ERRINFS_XDB_ERR_RESOURCE

XDB コミット処理時のバッファ不足

トランザクション種別が ER，エラー要因コードが EERPC_ERRINF_XDB_ERR の場合

エラー要因詳細情報（詳細については，マニュアル「TP1/EE/Extended Data Cache 使用の手引」を参照してください）

- 入力キュー（ITQ）通番
入力メッセージの入力キューの通番が設定されます。
- ロールバックリトライ回数

トランザクションがロールバックして同じメッセージを入力するトランザクションをリトライした回数です。回数が 65535 を超えた場合は 65535 を設定します。XTC のロールバックリトライ回数監視によってサービスが閉塞した場合、またはメッセージが読み出し済みとなった場合に 0 に戻ります。

- 系切り替え後の実行系の状態

系切り替えが発生したときの、系切り替え後の XTC の状態が設定されます。

EERPC_STATUS_CHG_ONLINE

オンライン

EERPC_STATUS_CHG_ISOLATEA

実行系孤立モード A

EERPC_STATUS_CHG_ISOLATEB

実行系孤立モード B

●MCP 拡張トランザクションインタフェース情報

マニュアル「TP1/EE/Message Control Extension 使用の手引」を参照してください。

●メッセージ種別

トランザクション起動契機となった入力メッセージの種別が設定されます。

EERPC_MSGTYPE_RPC

RPC メッセージ (TCP/IP)

EERPC_MSGTYPE_TIM

タイマトランザクションメッセージ

EERPC_MSGTYPE_RAP

RAP メッセージ (TCP/IP)

EERPC_MSGTYPE_MCH

一方送信メッセージ (UDP)

EERPC_MSGTYPE_RPC_UDP

RPC メッセージ (UDP)

EEMCP_MSGTYPE_TCP

MCP メッセージ (TCP/IP)

EEMCP_MSGTYPE_UDP

MCP メッセージ (UDP)

EERPC_MSGTYPE_DBQ

DB キューメッセージ

EERPC_MSGTYPE_OBS

オンラインバッチメッセージ

●エラートランザクション要因

ERRTRN4 を起動した要因が設定されます。

EERPC_ERRTRN4_SERVICE_HOLD

サービス閉塞による起動

EERPC_ERRTRN4_ONLINE_END

オンライン終了による起動

●拡張ユーザ IFA エリアサイズ

拡張ユーザ IFA エリアのサイズが設定されます（単位：バイト）。

●拡張ユーザ SAT エリアサイズ

拡張ユーザ SAT エリアのサイズが設定されます（単位：バイト）。

●トランザクション起動要因

TP1/FSP の関連ドキュメントを参照してください。

●サーバ再起動回数

TP1/FSP の関連ドキュメントを参照してください。

●トランザクション起動時刻

トランザクションを開始した時刻が、YYYYMMDDHHMMSS の形式の文字列で設定されます。文字列の末尾に NULL 文字は付きません。

●送信元のノード識別子

TP1/FSP の関連ドキュメントを参照してください。

●リトライ回数

TP1/FSP の関連ドキュメントを参照してください。

●トランザクションレベル名長

TP1/FSP の関連ドキュメントを参照してください。

●トランザクションレベル名

TP1/FSP の関連ドキュメントを参照してください。

●トランザクション引き継ぎ情報

TP1/FSP の関連ドキュメントを参照してください。

システム運用の管理 (ee_adm_～)

UAP から TP1/EE システムの各種機能を使う関数について説明します。システム運用の管理の関数を次に示します。

- ee_adm_call_command – 運用コマンドの実行

ee_adm_call_command

名称

運用コマンドの実行

形式

ANSI C, C++の形式

```
#include <eeadm.h>
int ee_adm_call_command(char *com, int *stat, char *outmsg,
                        EEULONG *outsiz, char **errmsg,
                        EEULONG *errsiz, EELONG time,
                        EELONG flags)
```

機能

オンライン中にコマンドを入力したときと同様に、UAP からコマンドを実行します。コマンドの処理が完了するまで待って、コマンドの実行結果を戻します。コマンドの処理が終了すると、標準出力情報と標準エラー出力情報が返ります。

コマンドの実行形式ファイルを格納したディレクトリをサーチパスに追加しておいてください。

UAP で値を設定する引数

●com

実行する運用コマンドの文字列を設定します。指定できる文字列の最大長は 2048 バイトです（末尾の NULL 文字を含む）。

●outmsg

コマンドの実行結果として標準出力に出力する文字列を受け取る領域の先頭ポインタを設定します。事前にこのアドレスから、(outsiz の値 + errsiz の値) 分の領域を確保しておいてください（単位：バイト）。出力結果を受け取る必要がない場合でも、この長さの領域を確保してください。

●outsiz

コマンドの実行結果として標準出力に出力する文字列（outmsg に返される値）を受け取る長さを設定します（単位：バイト）。指定できる値は 1～32768 です。設定する長さには出力結果の末尾の NULL 文字分も含まれます。

この引数の値は、UAP から実行するコマンドに応じて決めてください。出力結果を受け取る必要がない場合は、1 を設定してください。

処理終了後は、コマンドの実行結果として標準出力に出力した実際の文字列の長さが返されます。

●errsiz

コマンドの実行結果として標準エラー出力に出力する文字列 (errmsg に返される値) を受け取る長さを設定します (単位: バイト)。指定できる値は 13~32780 です。設定する長さには出力結果の末尾の NULL 文字分も含まれます。

この引数の値は、UAP から実行するコマンドに応じて決めてください。出力結果を受け取る必要がない場合は、13 を設定してください。

処理終了後は、コマンドの実行結果として標準エラー出力に出力した実際の文字列の長さが返されます。

●time

応答監視時間を設定します (単位: 秒)。指定できる値は 1~65535, EEADM_TIMEOUT_DEF, または EEADM_TIMEOUT_INF です。

EEADM_TIMEOUT_DEF を設定した場合、プロセス関連定義の call_cmd_time オペランドで指定した値が応答監視時間になります。EEADM_TIMEOUT_INF を設定した場合は、時間監視をしません。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

次に示す引数を参照できるのは、リターン値が EE_OK, EEADMER_STATNOTZERO, EEADMER_MEMORY_OUT, EEADMER_MEMORY_ERR, または EEADMER_MEMORY_OUTERR のときだけです。ほかのリターン値のときは、参照できません。

●stat

実行したコマンドが、正常終了したか異常終了したか (シェルの終了コード) *が返されます。コマンドをバックグラウンド実行した場合は、正常終了か異常終了かに関係なく 0 が返されます。

注※

waitpid(2)で指定した形式で、sh(1)の終了ステータスを示します。

●outmsg

コマンドの実行結果として標準出力に出力した文字列が返されます。

文字列の先頭アドレスは、ee_adm_call_command 関数を呼び出すときに outmsg に設定したアドレスです。文字列の長さは、最大で (ee_adm_call_command 関数を呼び出すときに outsiz に設定した値-1) です (単位: バイト)。

長さの最大を超える文字列は切り捨てられます。また、パイプの容量を超えた分も切り捨てられます。長さの最大を超えなかった場合は、その文字列の長さ分だけ返されます。

文字列の末尾には NULL 文字が付きます。出力結果がない場合は、NULL 文字だけが返されます。

●outsiz

標準出力に出力した文字列の長さが返されます（単位：バイト）。返される長さには出力結果の末尾の NULL 文字分は含まれません。

●errmsg

コマンドの実行結果として標準エラー出力に出力した文字列の先頭ポインタが返されます。

この文字列は、標準出力に出力した文字列に続いて出力されます。文字列の長さは、最大で (ee_adm_call_command 関数を呼び出すときに errsiz に設定した値－13) です（単位：バイト）。

長さの最大を超える文字列は切り捨てられます。また、パイプの容量を超えた分も切り捨てられます。長さの最大を超えなかった場合は、その文字列の長さ分だけ返されます。

文字列の末尾には NULL 文字が付きます。出力結果がない場合は、NULL 文字が 13 バイト返されます。

●errsiz

標準エラー出力に出力した文字列の長さが返されます（単位：バイト）。返される長さには出力結果の末尾の NULL 文字分は含まれません。

リターン値

リターン値	意味
EE_OK	シェルの終了コードは 0（コマンドの実行が正常終了）です。標準出力および標準エラー出力の領域に文字列を格納しました。
EEADMER_ARGUMENT	引数に設定した値が間違っています。
EEADMER_DEF_INVALID	運用コマンド実行サーバ関連の定義が指定されていません。
EEADMER_MEMORY_ERR	標準エラー出力のデータが、領域に収まりません。
EEADMER_MEMORY_OUT	標準出力のデータが、領域に収まりません。
EEADMER_MEMORY_OUTERR	標準出力のデータと標準エラー出力のデータの両方が、領域に収まりません。
EEADMER_MEMORY_SHORT	メモリが不足しました。
EEADMER_RPC_ERR	運用コマンド実行サーバへの通信が失敗しました。
EEADMER_STATNOTZERO	シェルの終了コードは 0 以外（コマンドの実行が異常終了）です。標準出力および標準エラー出力のデータを領域に格納しました。
EEADMER_SYSTEMCALL	システムコール（close, pipe, dup, または read）の呼び出しに失敗しました。
EEADMER_UNRESOLVABLE	そのほかのエラーが発生しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。

注意事項

1. この関数で実行できる TP1/EE のコマンドは、トランザクション種別によって異なります。詳細は「[2.6 運用コマンド実行機能](#)」を参照してください。
2. サーチパスに指定したディレクトリ間で、コマンド名が重複しないように注意してください。コマンド名が重複している場合、正しいコマンドが実行されないおそれがあります。
3. ファイル操作に関連するコマンドを実行する場合、ファイルの I/O などの処理に時間が掛かるため、実行する場合は十分注意してください。

DB キュー制御 (ee_dbq_～)

DB キューによるメッセージ管理に使用する関数について説明します。DB キュー制御の関数を次に示します。

- ee_dbq_acsctl – DB キューの読み出し制御
- ee_dbq_clear – DB キューの通番初期化
- ee_dbq_conctl – DB キューの接続制御
- ee_dbq_inqsrv – DB キューサービスの接続照会
- ee_dbq_inquire – DB キュー情報の照会
- ee_dbq_inquire2 – DB キュー情報の照会
- ee_dbq_msgget – メッセージの読み出し
- ee_dbq_msgput – メッセージの書き込み
- ee_dbq_msgrefer – メッセージの参照読み出し
- ee_dbq_msgrefer2 – メッセージの参照読み出し
- ee_dbq_obscancel – オンラインバッチ処理の強制終了
- ee_dbq_obschdata – 引き継ぎ情報の更新
- ee_dbq_obsend – オンラインバッチ処理の終了
- ee_dbq_obsinquire – ロットの情報の照会
- ee_dbq_obsrbskip – ロールバック要求時のメッセージの読み出しスキップ
- ee_dbq_obsrestart – オンラインバッチ処理の再開始
- ee_dbq_obsskip – メッセージの読み出しスキップ
- ee_dbq_obsstart – オンラインバッチ処理の開始
- ee_dbq_obsstop – オンラインバッチ処理の中断
- ee_dbq_rbskip – ロールバック要求時のメッセージの読み出しスキップ
- ee_dbq_skip – メッセージの読み出しスキップ
- ee_dbq_srvctl – DB キューサービスの閉塞制御

DB キュー機能で提供している関数の一覧と使用できる機能を次の表に示します。

表 4-10 DB キュー関数の一覧と使用できる機能

項番	関数名	DB キュー		オンラインバッチ (OBS)
		UAP 間通信	ユーザキュー	
1	ee_dbq_acsctl	○	○	×

項番	関数名	DB キュー		オンラインバッチ (OBS)
		UAP 間通信	ユーザキュー	
2	ee_dbq_clear	○	○	×
3	ee_dbq_conctl	○	×	×
4	ee_dbq_inqsrv	○	×	×
5	ee_dbq_inquire	○	×	×
6	ee_dbq_inquire2	○	○	×
7	ee_dbq_msgget	×	○	×
8	ee_dbq_msgput	○	○	×
9	ee_dbq_msgrefer	○	○	×
10	ee_dbq_msgrefer2	○	○	×
11	ee_dbq_obscancel	×	×	○
12	ee_dbq_obschdata	×	×	○
13	ee_dbq_obsend	×	×	○
14	ee_dbq_obsinquire	×	×	○
15	ee_dbq_obsrbskip	×	×	○
16	ee_dbq_obsrestart	×	×	○
17	ee_dbq_obsskip	×	×	○
18	ee_dbq_obsstart	×	×	○
19	ee_dbq_obsstop	×	×	○
20	ee_dbq_rbskip	○	×	×
21	ee_dbq_skip	○	○	×
22	ee_dbq_srvctl	○	×	×

(凡例)

○：使用できます。

×

ee_dbq_acsctl

名称

DB キューの読み出し制御

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_acsctl(EEULONG access_flg, char *dbque_name,
                  EELONG flags)
```

機能

DB キューの読み出しを停止，および再開します。

UAP で値を設定する引数

●access_flg

次のフラグを設定します。

EEDBQ_ACSCTL_STOP

DB キューの読み出しを停止します。

EEDBQ_ACSCTL_RESTART

DB キューの読み出しを再開します。

●dbque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

オプションを次に示す形式で設定します。

```
{EENOFLLAGS | EEDBQ_ACSCTL_ANY}
```

EENOFLLAGS

dbque_name に指定した自サーバ読み出しの DB キューを停止，再開します。

EEDBQ_ACSCTL_ANY

dbque_name に指定した自サーバ読み出しまたは他サーバ読み出しの DB キューを停止，再開します。

dbque_name に指定した DB キューが他サーバ読み出しの場合、メッセージを読み出している TP1/EE プロセスが 07-80 以降かつ DB キューのフォーマットタイプが 1 であれば、RPC 通信を行って DB キューの読み出しを停止、再開します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDBQER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDBQER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EEDBQER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
EEDBQER_FUNC_OPT	引数に設定した DB キューのフォーマットタイプが不正です。※1
EEDBQER_FWD_CHGR	変更要求処理を実行しましたが、変更要求処理の実行後に読み出し先サーバの変更を検知しました。※1, ※2
EEDBQER_FWD_DEST	変更要求処理に失敗しました。 対象の DB キューを読み出しているサーバが存在しません。※1
EEDBQER_FWD_SEND	変更要求処理に失敗しました。※1
EEDBQER_FWD_TIMED_OUT	変更要求は実行しましたが、変更要求結果の確認処理が時間切れになりました。※1
EEDBQER_FWD_UN_FWD_DEST	変更要求処理を実行しましたが、変更要求処理の結果確認で EEDBQER_FWD_DEST を検知しました。※1, ※2
EEDBQER_FWD_UN_FWD_VER	変更要求処理を実行しましたが、変更要求処理の結果確認で EEDBQER_FWD_VER を検知しました。※1, ※2
EEDBQER_FWD_UN_RMACCESS	変更要求処理を実行しましたが、変更要求処理の結果確認で EEDBQER_RM_ACCESS が発生しました。※1, ※2
EEDBQER_FWD_UN_ROLLBACK_RESERVE	変更要求処理を実行しましたが、変更要求処理の結果確認で EEDBQER_ROLLBACK_RESERVE を検知しました。※1, ※2
EEDBQER_FWD_VER	変更要求処理に失敗しました。 対象の DB キューを読み出しているサーバが変更要求に対応していません。※1
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_QUE_TYPE	設定された DB キューのタイプは機能対象外です。※3

リターン値	意味
EEDBQER_QUEUE_TYPE	相手読み出し DB キュー
EEDBQER_RM_ACCESS	リソースマネージャアクセス時にエラーが発生しました。※1
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。※1
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。※1 HiRDB の暗黙的ロールバックについては、HiRDB シリーズのマニュアルを参照してください。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注※1

flags に EEDBQ_ACSCCTL_ANY を指定した場合に返されます。

注※2

DB キューが HiRDB の場合にだけ発生するリターン値です。

注※3

flags に EENOFLLAGS を指定した場合に返されます。

注意事項

- DB キューの読み出しトランザクションで、該当する DB キューに対して ee_dbq_acsctl 関数を呼び出した場合、読み出しトランザクション完了後に読み出しを停止します。この状態で ee_dbq_inquire 関数を呼び出した場合、DB キューの状態は EEDBQ_QUEUE_STATUS_CSTOP（読み出し停止中）が返されます。
- 次の場合、ee_dbq_inquire 関数はリターン値 EE_OK でリターンします。
 - 読み出し停止状態の DB キューに対して読み出し停止を要求した場合
 - 読み出し開始状態（再開を含む）の DB キューに対して読み出し再開を要求した場合
- DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した DB キューに対して ee_dbq_acsctl 関数を呼び出した場合、DB キューのステータスが他サーバ読み出しのときでも、リターン値が EEDBQER_QUEUE_TYPE とならないで、EE_OK となることがあります。
- flags に EEDBQ_ACSCCTL_ANY を指定した場合に、DB キュー読み出し先サーバ変更機能と ee_dbq_acsctl 関数が競合すると、リターン値が EE_OK でも DB キューの読み出し状態が読み出し停止、再開状態にならないことがあります。
- ee_dbq_acsctl 関数のエラーリターン時は、リターン値別に次のように対処してください。
 - EEDBQER_FWD_TIMED_OUT
ee_dbq_acsctl 関数の発行元の TP1/EE プロセスが出力するメッセージで要求先の TP1/EE プロセスを特定します。
要求先の TP1/EE プロセスが出力するエラーメッセージを基に障害を取り除いてください。

このリターン値が返ったときは読み出し停止，再開要求の結果は不定です。

実行結果を確認したい場合は，ee_dbq_inquire2 関数または eedbqls コマンドで実行結果を確認してください。

- EEDBQER_FWD_DEST

DB キュー読み出し先サーバ変更機能を使用して読み出し先サーバを決定するか，該当の DB キューを読み出す TP1/EE プロセスを開始してください。

- EEDBQER_FWD_VER

メッセージを読み出しているサーバが 07-80 以降の TP1/EE であることを確認してください。

- EEDBQER_FWD_SEND

ee_dbq_acsctl 関数の発行元の TP1/EE プロセスが出力する次のメッセージを参照して，障害を取り除いてください。

KFSB40320-E, KFSB40322-E, KFSB40350-E, KFSB40352-E

- EEDBQER_FWD_CHGR

このリターン値が返ったときは読み出し停止，再開要求の結果は不定です。

実行結果を確認したい場合は，ee_dbq_inquire2 関数または eedbqls コマンドで実行結果を確認してください。

- EEDBQER_FWD_UN_RMACCESS, EEDBQER_FWD_UN_FWD_VER,
EEDBQER_FWD_UN_FWD_DEST, EEDBQER_FWD_UN_ROLLBACK_RESERVE

これらのリターン値が返ったときは読み出し停止，再開要求の結果は不定です。

実行結果を確認したい場合は，変更要求結果処理確認で発生したエラーの障害要因を取り除いてから，ee_dbq_inquire2 関数または eedbqls コマンドで実行結果を確認してください。

ee_dbq_clear

名称

DB キューの通番初期化

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_clear(char *dbque_name, EELONG flags)
```

機能

通番初期化属性がある DB キューの書き込み通番、読み出し通番を 0 で初期化します。未読み出しのメッセージがある場合は破棄します。初期化した通番を元に戻したり、破棄したメッセージを読み出したりはできません。

通番を初期化するための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

通番を初期化するには、次の条件をすべて満たす必要があります。満たさない場合は、エラーリターンします。() 内は条件を満たさない場合のリターン値です。

- 自プロセスが対象の DB キューから読み出す (EEDBQER_QUE_TYPE)
- 読み出し停止中 (EEDBQER_TIMING)
- 書き込み抑止中 (EEDBQER_TIMING)
- 他スレッドでメッセージの読み出し、スキップ、初期化を実行していない (EEDBQER_TIMING)

これらの条件に関係なく、強制的に通番を初期化することもできます。強制的に初期化する場合に、強制初期化と次のトランザクションが競合すると、次のトランザクションがロールバックすることがあります。

- DB キューからメッセージを読み出し中のトランザクション
- DB キューのメッセージをスキップ中のトランザクション

【TP1/FSP 限定】 TP1/FSP 使用時は、最初に起動したサーバの JI トランザクションから、初期化属性があるすべての DB キューを初期化できます。

UAP で値を設定する引数

●dbque_name

DB キュー名を、27 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

flags に EEDBQ_MSGCLR_FLAGS_ALL を指定した場合は、NULL 文字を設定してください。

●flags

次のフラグを設定します。

EENOFLLAGS

関数の呼び出し条件をすべて満たしているかチェックして初期化する場合に指定します。

EEDBQ_MSGCLR_FLAGS_FORCE

関数の呼び出し条件に関係なく強制初期化する場合に指定します。

EEDBQ_MSGCLR_FLAGS_ALL **【TP1/FSP 限定】**

定義されているすべての DB キューの通番を初期化します。TP1/FSP 使用時の最初に起動したサーバの JI トランザクションでだけ指定できます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDCOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDCOMER_CNDOUC	UOC から発行しているため、この機能は使用できません。
EEDCOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。 トランザクション種別不正
EEDBQER_FUNC_OPT	DB キューの属性が対応していません。 - DB キューに通番初期化属性が付与されていません。 - DB キューのフォーマットタイプに 1 以上が指定されていません。
EEDBQER_MEMORY_SHORT*	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_QUE_TYPE	設定された DB キューのタイプは機能対象外です。

リターン値	意味
EEDBQER_QUEUE_TYPE	相手読み出し DB キュー
EEDBQER_RM_ACCESS※	リソースマネージャアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT※	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_STOP※	DB キューが使用できない状態です。
EEDBQER_TIMING※	この API 関数を呼び出せる状態ではありません。 - 読み出し停止中ではありません。 - 書き込み抑止中ではありません。 - 他スレッドで DB キューにアクセスしています。
EEDBQER_UNRESOLVABLE※	その他のエラーが発生しました。

注※

flags に EEDBQ_MSGCLR_FLAGS_ALL を設定した場合、エラーが発生した DB キューでエラーリターンします。一部の DB キューの通番初期化が実行されているおそれがあります。無効にしたい場合、ロールバックしてください。

注意事項

1. 通番の初期化は、サービス関数のリターン、または ee_trn_chained_commit 関数発行時に行われます。
2. 同一グローバルトランザクションのトランザクションブランチ間で、同一 DB キューに対して ee_dbq_msgput 関数の発行、または初期化を行うと、デッドロックが発生する場合があります。
3. DB キューからメッセージを読み出したトランザクションで該当する DB キューを ee_dbq_acsctl 関数で読み出し停止にして、該当する DB キューに初期化要求を行っても受け付けません。初期化要求は、別トランザクションまたはコマンドで実行してください。
4. flags に EEDBQ_MSGCLR_FLAGS_ALL を設定する場合に、他プロセスで DB キューにアクセスしているとデッドロックするおそれがあります。EEDBQ_MSGCLR_FLAGS_ALL を指定する場合は、すべてのサーバで同じ共有情報表を使用するか、同じ共有情報表を使用していないプロセスからは DB キューにアクセスしないようにしてください。
5. ee_dbq_clear 関数の受け付け後に、eedbqchgr コマンドで読み出し先サーバを 07-80 より前のバージョンのサーバに変更した場合、ee_dbq_clear 関数を受け付けたトランザクションがロールバックします。
6. flags に EENOFLLAGS を設定した ee_dbq_clear 関数の受け付け後に、eedbqchgr コマンドで読み出し先サーバを変更した場合、ee_dbq_clear 関数を受け付けたトランザクションがロールバックします。

ee_dbq_conctl

名称

DB キューの接続制御

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_conctl(EEULONG conctl_flg, char *dbque_name,
                  char *service_name, EELONG flags)
```

機能

DB キューと DB キューサービスとを接続，および接続解除します。

UAP で値を設定する引数

●conctl_flg

次のフラグを設定します。

EEDBQ_CONCTL_CONNECT

DB キューと DB キューサービスとを接続します。

EEDBQ_CONCTL_RELEASE

DB キューと DB キューサービスとの接続を解除します。

●dbque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●service_name

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。

リターン値	意味
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_CONNECT_LIMIT	該当する DB キューサービスで使用する最大 DB キュー数を超えるため接続できません。
EEDBQER_GROUP	要求のあった DB キューサービスの属する DB キューサービスグループと DB キューの属する DB キューグループは DB キュー機能関連定義で関連づけられていません。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_NO_SERVICE	引数に設定した DB キューサービス名は存在しません。
EEDBQER_NOT_CONNECT	要求のあった DB キューと DB キューサービスは接続されていません。
EEDBQER_NOT_UAPQUE	AP 間通信機能を使用する DB キューではありません。
EEDBQER_OTHER_CONNECT	すでに、別 DB キューサービスと接続済みのため接続できません。
EEDBQER_QUE_TYPE	設定された DB キューのタイプは機能対象外です。 相手読み出し DB キュー
EEDBQER_SERVICE_STOP	DB キューサービスが永久閉塞中（エントリポイントなし）です。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 該当する DB キューの読み出し処理中に接続解除、接続を続けて行えません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. DB キュー機能関連定義の dbqgrpdef 定義コマンドに -c オプションを指定していない相手システムの読み出し DB キューとの接続、および接続解除はできません。
2. 該当する DB キューの読み出し処理中に接続解除と接続とを続けて実行できません。
3. 最初の AP 起動時には、定義によって任意の DB キューサービスに接続されているか、またはどの DB キューサービスにも接続されていない場合があります。DB キューをほかの DB キューサービスに接続変更する場合は、現在接続されている DB キューサービスを指定して接続を解除してから、変更先の DB キューサービスを指定して接続してください。
4. 次の場合、ee_dbq_conctl 関数はリターン値 EE_OK でリターンします。

- 接続されている DB キューと DB キューサービスに対して接続を要求した場合
- 接続されていない DB キューと DB キューサービスに対して接続解除を要求した場合

ee_dbq_inqsrv

名称

DB キューサービスの接続照会

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_inqsrv(char *svname, EEULONG *srv_sts, void *dbq_list,
                  EEULONG in_cnt, EEULONG *out_cnt, EEULONG flags)
```

機能

DB キューサービスと接続されている DB キュー名を照会します。

UAP で値を設定する引数

●svname

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●srv_sts

DB キューサービスの閉塞状態を格納する領域のアドレスを設定します。

●dbq_list

DB キューサービスと接続されている DB キュー名を格納する領域のアドレスを設定します。この領域には、DB キュー名の配列を格納します。一つの DB キュー名の長さは 28 バイトです。

●in_cnt

DB キュー名の配列を格納する領域の配列要素数を設定します。dbq_list で設定した格納領域は、(28×in_cnt の値) 分の長さが必要です (単位: バイト)。

●out_cnt

DB キュー名の個数を格納する領域のアドレスを設定します。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●srv_sts

DB キューサービスの閉塞状態が返されます。

EEDBQ_INQSRV_ACT

閉塞解除中

EEDBQ_INQSRV_PCT

障害による閉塞中

EEDBQ_INQSRV_CCT

コマンドまたは API 関数による閉塞中

EEDBQ_INQSRV_ECT

永久閉塞中（エントリポイントなし）

●dbq_list

DB キューサービスと接続されている DB キュー名の配列が返されます。

●out_cnt

DB キューサービスと接続されている DB キュー名の個数が返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDCOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDCOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EEDCOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_LIST_SZ	DB キュー名リストサイズが不足しています。このとき、引数 out_cnt には該当する DB キュー数が設定されていますので、引数 in_cnt の設定値を再考する必要があります。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンクされていないためです。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_NO_SERVICE	引数に設定したサービス名は存在しません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

DB キュー名の配列を格納する領域が不足した場合（リターン値 EEDBQER_LIST_SZ）でも，引数 in_cnt に設定した要素数分の DB キュー名は返されます。

ee_dbq_inquire

名称

DB キュー情報の照会

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_inquire(char *dbque_name, EEDBQINF *dbq_inf,
                  EELONG flags)
```

機能

DB キューの情報を照会します。ユーザキューアクセス機能を使用する DB キューでは、使用できません。ee_dbq_inquire2 関数を使用してください。

UAP で値を設定する引数

●dbque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●dbq_inf

DB キュー情報を格納するための構造体ポインタのアドレスを設定します。

●flags

次のフラグを設定します。

EEDBQ_INF_FLAGS_0200

DB キュー読み出しサーバ変更機能または DB キュー書き込み抑止機能を使用する場合に設定します。このフラグを設定すると、最新の DB キュー情報を照会します。

EENOFLLAGS

DB キュー読み出しサーバ変更機能および DB キュー書き込み抑止機能以外の機能を使用する場合に設定します。このフラグを設定すると、DB キュー情報を照会します。

TP1/EE から値が返される引数

●dbq_inf

DB キュー情報が構造体 EEDBQINF で返されます。構造体の形式は次のとおりです。

```

struct ee_dbq_inf_t{
    char    service_name[32];
    char    dbq_grp_name[32];
    char    dbque_name[28];
    char    read_srvgrp_name[32];
    EEULONG sts_code;
    EEULONG write_event_num;
    EEULONG read_num;
    EEULONG write_max_num;
    EEULONG unread_count;
}EEDBQINF;

```

- service_name (DB キューサービス名)
指定した DB キューに接続している DB キューサービス名を返します。接続していない場合は NULL 文字を設定します。
- dbq_grp_name (DB キューグループ名)
指定した DB キューが属する DB キューグループ名を返します。
- dbque_name (DB キュー名)
ee_dbq_inquire 関数を呼び出すときに dbque_name に設定した DB キュー名をそのまま返します。
- read_srvgrp_name (DB キューを読み出すサービスグループ名) ※1
指定した DB キューを読み出すサービスグループ名を返します。
DB キュー機能関連定義の dbqgrpdef 定義コマンドに -f オプションを指定した場合、または、-k オプションに 1 を指定した場合で、DB キューからメッセージを読み出すサーバを確認できないとき、および、DB キューからメッセージを読み出すサーバにサービスグループ名がないときは、値は NULL 文字を設定します。
- sts_code (ステータスコード)
 - ステータスコード 1
DB キューの状態を返します。 ※1
 - EEDBQ_QUEUE_STATUS_NORMAL
正常
 - EEDBQ_QUEUE_STATUS_PSTOP
障害による読み出し停止中
 - EEDBQ_QUEUE_STATUS_CSTOP
コマンド、TP1/EE サービス定義、または API 関数による読み出し停止中
 - EEDBQ_QUEUE_STATUS_ESTOP
DB 構成不正または DB キューの内容矛盾による使用不可
 - EEDBQ_QUEUE_STATUS_NOTRM
リソースマネージャ未接続で使用不可
 - ステータスコード 2
DB キューの状態を返します。

・ EEDBQ_QUE_STATUS_WTPR

書き込み抑止※1, ※2

このステータスは、ステータスコード 1 が EEDBQ_QUE_STATUS_ESTOP または EEDBQ_QUE_STATUS_NOTRM の場合は無効となります。

- write_event_num (メッセージ書き込み通番)
DB キューへ最後に書き込んだメッセージの通番を返します。※3
- read_num (メッセージ読み出し通番)
DB キューから最後に読み出したメッセージの通番を返します。※3
- write_max_num (最大書き込みメッセージ数)
DB キュー機能関連定義の dbqdef 定義コマンドで指定した最大書き込みメッセージ数と同一の値を返します。
- unread_count (未読み出しメッセージ数)
未読み出しメッセージ数を返します。※3

注※1

flags に EEDBQ_INF_FLAGS_0200 を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、DB キューから情報を読み込みます。

注※2

flags に EEDBQ_INF_FLAGS_0200 を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、ステータスコード EEDBQ_QUE_STATUS_WTPR は、ほかのコードとの論理和を返します。

注※3

ステータスコードが次の場合、値は参照できません。

EEDBQ_QUE_STATUS_ESTOP

DB 構成不正または DB キューの内容矛盾による使用不可

EEDBQ_QUE_STATUS_NOTRM

リソースマネージャ未接続で使用不可

また、該当する DB キューのタイプが相手読み出し DB キューの場合、値は返りません。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。

リターン値	意味
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_NOT_UAPQUE	AP 間通信機能を使用する DB キューではありません。
EEDBQER_RM_ACCESS	リソースマネージャへのアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 通番の初期化中です。

注意事項

- DB キュー読み出しサーバ変更機能または DB キュー書き込み抑止機能を使用する場合、次に示す点に注意してください。
 - flags に EEDBQ_INF_FLAGS_0200 を設定してください。ただし、EEDBQ_INF_FLAGS_0200 を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、DB アクセスが発生するため、TP1/EE の性能が低下します。
 - flags に EENOFLAGS を設定しないでください。EENOFLAGS を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、read_srvgrp_name や sts_code のステータスコード 1 に最新の DB キュー情報が返らないことがあります。
- ほかの TP1/EE で、ee_dbq_clear 関数または eedbqtblh コマンドで通番の強制初期化を行った場合、ee_dbq_inquire 関数は初期化前の通番を返すことがあります。強制初期化後の通番を知りたい場合は、ee_dbq_inquire2 関数を使用してください。

ee_dbq_inquire2

名称

DB キュー情報の照会

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_inquire2(char *dbque_name, EEDBQINF2 *dbq_inf, EELONG flags)
```

機能

DB キューの情報内容を照会します。

DB キューユーザ要求転送機能を使用した場合は、指定した DB キューを読み出しているサーバの情報を取得できます。

DB キューのフォーマットタイプが 1 の DB キューに対して実行できます。

UAP で値を設定する引数

●dbque_name

照会する DB キュー名を、27 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●dbq_inf

照会した DB キュー情報を格納するための構造体ポインタのアドレスを設定します。

●flags

次のフラグを設定します。

EENOFLAGS

メッセージを読み出している TP1/EE プロセスの読み出し状態の照会が不要な場合に設定します。

メッセージを読み出している TP1/EE プロセスの DB キューの読み出し状態を照会するための RPC 通信は行いません。

EEDBQ_REQ_READSTATUS

メッセージを読み出している TP1/EE プロセスの読み出し状態の照会が必要な場合に設定します。

dbque_name に指定した DB キューを読み出している TP1/EE プロセスが API 発行元のプロセスと異なる場合、RPC 通信を行って DB キューの読み出し状態を照会します。

flags に EEDBQ_REQ_READSTATUS を指定した場合、メッセージを読み出している TP1/EE のバージョンが 07-80 以降である必要があります。

TP1/EE から値が返される引数

●dbq_inf

DB キュー情報が構造体 EEDBQINF2 で返されます。構造体の形式は次のとおりです。

```
typedef struct ee_dbq_inf2_t{
    char    service_name[32];
    char    dbq_grp_name[32];
    char    dbque_name[28];
    char    read_srvgrp_name[32];
    EEULONG sts_code;
    EEULONG write_event_num;
    EEULONG read_num;
    EEULONG write_max_num;
    EEULONG unread_count;
    char    read_base_id[8];
    char    read_system_id[16];
    EEULONG dbq_kind;
    EEULONG attribute;
    char    node_id[5];
    char    write_inhibit_sts;
    char    yobi[74];
}EEDBQINF2;
```

- service_name (サービス名)
指定した DB キューに現在接続しているサービス名を返します。接続していない場合、NULL 文字を設定します。
- dbq_grp_name (DB キューグループ名)
指定した DB キューが属する DB キューグループ名を返します。
- dbque_name (DB キュー名)
dbque_name に設定した DB キュー名をそのまま返します。
- read_srvgrp_name (DB キューを読み出すサービスグループ名)
指定した DB キューを読み出すサービスグループ名を返します。
読み出すサーバにサービスグループ名がない場合、NULL 文字を設定します。
- sts_code (DB キューの読み出し状態)
flags に EEDBQ_REQ_READSTATUS を指定した場合の、DB キューの読み出し状態を返します。
flags に EENOFLAGS を指定した場合、DB キューを読み出している TP1/EE プロセスが ee_dbq_inquire2 関数の発行元のプロセスと異なっていると、DB キュー状態 1 が読み出し停止中のときでも EEDBQ_QUE_STATUS_NORMAL (正常) を返します。
DB キューの読み出し状態は DB キューの状態 1 と DB キューの状態 2 の和で返します。
 - DB キューの状態 1 :
DB キューの状態 1 を次のコードで返します。
EEDBQ_QUE_STATUS_NORMAL : 正常

EEDBQ_QUE_STATUS_PSTOP：障害による読み出し停止中

EEDBQ_QUE_STATUS_CSTOP：コマンド，定義，または API 関数による読み出し停止中

EEDBQ_QUE_STATUS_ESTOP：DB 構成不正または DB キューの内容矛盾による使用不可

EEDBQ_QUE_STATUS_NOTRM：リソースマネージャ未接続で使用不可

- DB キューの状態 2：

DB キューの状態 2 を次のコードで返します。

EEDBQ_QUE_STATUS_WTPR：書き込み抑止中

DB キューの状態 1 が EEDBQ_QUE_STATUS_ESTOP または EEDBQ_QUE_STATUS_NOTRM のときは加算しません。

このとき，書き込み抑止状態は不明となります。

- write_event_num（書き込み通番）

DB キューへ最後に書き込んだメッセージの通番を返します。※

- read_num（読み出し通番）

DB キューから最後に読み出したメッセージの通番を返します。※

- write_max_num（最大書き込みメッセージ数）

定義に指定した最大書き込みメッセージ数と同一の値を返します。

- unread_count（未読み出し数）

未読み出し数を返します。※

- read_base_id（該当する DB キューを読み出すサーバの基盤種別）

読み出すサーバの基盤種別を返します。※

TP1EE：TP1/EE

CosmiRM：uCosminexus Reliable Messaging

- read_system_id（該当する DB キューを読み出すシステム種別）

該当する DB キューを読み出すシステム種別を返します。※

TP1/EE の場合，ノード識別子を返します。

uCosminexus Reliable Messaging の場合，RMSystemName で指定したシステム名称を返します。

- dbq_kind（DB キュー種別）

DB キューの種別を返します。

EEDBQ_QUE_KIND_UAPFUNC：AP 間通信機能を使用した DB キュー

EEDBQ_QUE_KIND_USERQUE：ユーザキューアクセス機能を使用した DB キュー

- attribute（DB キュー属性）

DB キューに付与した属性を論理和で返します。※

0：付与した属性なし

1：上書き抑止属性

2：通番初期化属性

4：データ連携支援書き込み

8：交替用メッセージ表書き込み機能を使用

- node_id (TP1/Server Base のノード識別子)

該当する DB キューを読み出す TP1/EE プロセスの TP1/Server Base のノード識別子を返します。文字列の最後は NULL 文字です。※

次の条件をすべて満たす場合に TP1/Server Base のノード識別子を返します。

- 該当する DB キューを読み出すサーバの基盤種別が TP1/EE
- 該当する DB キューを読み出す TP1/EE プロセスでネーム機能が有効
- 該当する DB キューを読み出す TP1/EE プロセスの TP1/EE バージョンが 07-80 以降

TP1/Server Base のノード識別子を返さない場合は、NULL 文字を返します。

- write_inhibit_sts (DB キューの書き込み禁止状態)

該当する DB キューに対する書き込み禁止状態を返します。

EEDBQ_WRITE_INHIBIT_OFF：書き込み可能状態 (X'00')

EEDBQ_WRITE_INHIBIT_ON：書き込み禁止状態 (X'01')

注※

DB キュー状態 1 が EEDBQ_QUE_STATUS_ESTOP または EEDBQ_QUE_STATUS_NOTRM の場合は値を参照できません。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。 • トランザクション種別が不正です。
EEDBQER_FUNC_OPT	引数に設定した DB キューのフォーマットタイプが不正です。※1
EEDBQER_FWD_CHGR	情報照会処理は実行しましたが、情報照会処理の実行後に読み出し先サーバの変更を検知しました。※1, ※2
EEDBQER_FWD_DEST	情報照会処理に失敗しました。 対象の DB キューを読み出しているサーバが存在しません。※1
EEDBQER_FWD_SEND	情報照会処理が失敗しました。※1

リターン値	意味
EEDBQER_FWD_TIMED_OUT	情報照会処理が時間切れになりました。※1
EEDBQER_FWD_VER	情報照会処理に失敗しました。 対象の DB キューを読み出しているサーバが情報照会処理に対応していません。※1
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_RM_ACCESS	リソースマネージャアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 • 通番の初期化中です。
EEDBQER_UNRESOLVABLE	その他のエラーが発生しました。

注※1

flags に EEDBQ_REQ_READSTATUS を指定した場合にだけ発生するリターン値です。

注※2

DB キューが HiRDB の場合にだけ発生するリターン値です。

注意事項

1. flags に EEDBQ_REQ_READSTATUS を指定し、かつ DB キューが Oracle の場合に DB キュー読み出し先サーバ変更機能と ee_dbq_inquire2 関数が競合すると、リターン値が EE_OK でも DB キューの読み出し状態が最新の状態にならないことがあります。
2. ee_dbq_inquire2 関数のエラーリターン時は、リターン値別に次のように対処してください。
 - EEDBQER_FWD_TIMED_OUT
ee_dbq_inquire2 関数の発行元の TP1/EE プロセスが出力するメッセージで、照会先の TP1/EE プロセスを特定します。
照会先の TP1/EE プロセスが出力するエラーメッセージを基に障害を取り除いてください。
 - EEDBQER_FWD_DEST
DB キュー読み出し先サーバ変更機能を使用して読み出し先サーバを決定するか、該当の DB キューを読み出す TP1/EE プロセスを開始してください。
 - EEDBQER_FWD_VER
メッセージを読み出しているサーバが 07-80 以降の TP1/EE であることを確認してください。
 - EEDBQER_FWD_SEND

ee_dbq_inquire2 関数の発行元の TP1/EE プロセスが出力する次のメッセージを参照して、障害を取り除いてください。

KFSB40320-E, KFSB40322-E, KFSB40350-E, KFSB40352-E

- EEDBQER_FWD_CHGR

ee_dbq_inquire2 関数を再発行するか、または eedbqls コマンドで実行結果を確認してください。

ee_dbq_msgget

名称

メッセージの読み出し

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_msgget (char *dbque_name, EEULONG buf_len, void *buf_pt,
                  EEULONG *out_len, EEULONG *seq_num, EEULONG flags)
```

機能

メッセージを読み出します。

メッセージを DB キューから読み出すための SQL の実行は、この関数内で行います。このときの SQL は、SQL 実行中だけ排他を取ります。

メッセージを読み出し済みに更新するための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定する引数

●dbque_name

読み出し対象の DB キュー名を設定します。27 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●buf_len

読み出したメッセージを格納する領域の長さを設定します。65 以上の値を指定してください。

●buf_pt

読み出したメッセージを格納する領域のアドレスを設定します。読み出したいメッセージの長さ + 64 バイトの領域を用意してください。先頭から 64 バイトは TP1/EE で使用します。

buf_pt には、8 バイトで境界調整した領域のアドレスを指定してください。

●out_len

読み出したメッセージの長さを格納する領域のアドレスを設定します。

●seq_num

読み出したメッセージの通番を格納する領域のアドレスを設定します。

●flags

EENOFLLAGS を設定します。

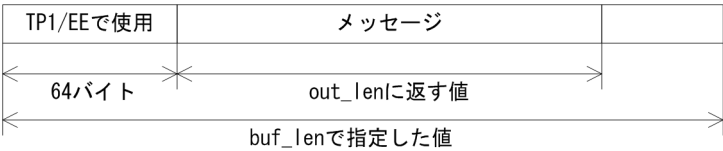
TP1/EE から値が返される引数

●buf_pt

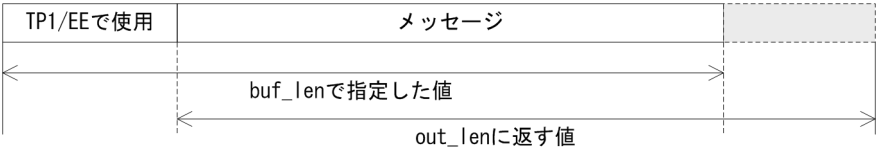
読み出したメッセージが返されます。読み出したメッセージの先頭は buf_pt + 64 バイトのアドレスです。

●out_len

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（リターン値 EEDBQER_BUF_OVERFLOW）でも、buf_len で指定した長さにかかわらず、読み出したメッセージの長さが返されます。この場合、オーバーフローしたメッセージは読み出し成功として動作します。



●seq_num

読み出したメッセージの通番が返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。

リターン値	意味
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_BUF_LEN	バッファ長が指定範囲外です。
EEDBQER_BUF_OVERFLOW	読み出したメッセージがバッファをオーバーフローしました。読み出し成功として動作します。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	要求された DB キューは存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_MESSAGE	未読み出しのメッセージはありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_NOT_USERQUE	ユーザキューアクセス機能を使用する DB キューではありません。
EEDBQER_QUE_TYPE	設定された DB キューのタイプは機能対象外です。 相手読み出し DB キュー
EEDBQER_QUE_TYPE_ROLLBACK	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、設定された DB キューのタイプの属性が変更されたため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_READ_STOP	読み出し停止中です。
EEDBQER_READ_TIMING	他サービスでリード処理中です。
EEDBQER_RM_ACCESS	リソースマネージャアクセス時にエラーが発生しました。
EEDBQER_RM_ACCESS_ROLLBACK	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、RM で障害が発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 - 通番の初期化中です。 - スキップ処理中です。
EEDBQER_TIMING_ROLLBACK	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、呼び出せる状態ではないため、ee_trn_rollback_mark 関数を呼び出しました。 通番の初期化中です。
EEDBQER_UNRESOLVABLE	その他のエラーが発生しました。

注意事項

1. DB キューサービス（ルートトランザクションブランチ）から呼び出された同一グローバルトランザクションのトランザクションブランチで、同一 DB キューに対してメッセージの読み出しを行うと、デッドロックが発生する場合があります。
2. 同一グローバルトランザクション内のトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだり読み出したりスキップしたりすると、デッドロックが発生する場合があります。
3. 同一トランザクションで同一の DB キューに対して複数回 ee_dbq_msgget 関数を実行すると、一回目以外の呼び出しでは、前回読み出したメッセージの次の通番のメッセージを読み出します。同じ通番のメッセージの再読み出しはしません。
4. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション (RM)
5. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション (ERRTRN3) のときは、リトライロールバックしません。

名称

メッセージの書き込み

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_msgput(char *dbque_name, char *msg_pt, EEULONG msg_len,
                  EELONG flags)
```

機能

送信メッセージを DB キューへ書き込みます。

DB キューに書き込むための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定する引数

●dbque_name

DB キュー名または DB キューグループ名を設定します。

- flags に EENOFLLAGS を設定する場合、DB キュー名を 27 バイト以内のアスキー文字列で設定します。
- flags に EEDBQ_MSGPUT_FLAGS_DBQGRP を設定する場合、DB キューグループ名を 31 バイト以内のアスキー文字列で設定します。

文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●msg_pt

メッセージを格納する領域のアドレスを設定します。

●msg_len

メッセージを格納する領域の長さを設定します（単位：バイト）。1～メモリ関連定義の dbq_max_message_size オペランドの値の範囲の値を指定できます。

●flags

次のフラグを設定します。

EEDBQ_MSGPUT_FLAGS_DBQGRP

dbque_name に DB キューグループ名を設定してメッセージを書き込みます。

EENOFLAGS

dbque_name に DB キュー名を設定してメッセージを書き込みます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_BUF_LEN	バッファ長が指定範囲外です。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_DBQGRP_ALLSTOP	要求された DB キューグループ内のすべての DB キューが次のどれかの状態のため、要求を受け付けられません。 - DB 構成不正による使用不可 - リソースマネージャ未接続で使用不可 - 縮退中 - 書き込み禁止状態
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQGRP	要求された DB キューグループは存在しません。
EEDBQER_NO_DBQUE	要求された DB キューは存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_STOP	DB キューが使用できない状態です。次のどちらかの状態となっています。 - DB 構成不正または DB キューの内容矛盾による使用不可 - リソースマネージャ未接続で使用不可
EEDBQER_WRITE_INHIBIT	DB キューが書き込み禁止状態です。

注意事項

1. DB キューサービス（ルートトランザクションブランチ）から呼び出された同一グローバルトランザクションのトランザクションブランチで、同一 DB キューに対してメッセージを書き込むと、デッドロックが発生する場合があります。
2. 同一グローバルトランザクション内のトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだりスキップしたりすると、デッドロックが発生する場合があります。
3. eedbqwtpr コマンドで書き込みを抑止している DB キューに対して、ee_dbq_msgput 関数で DB キューまたは DB キューグループを指定してメッセージの書き込み要求を行った場合、ee_dbq_msgput 関数は正常に終了し、メッセージ書き込み要求を行ったトランザクションをロールバックしてからエラートランザクション（ERRTRNR）を起動します。ただし、ee_trn_chained_commit 関数を呼び出して連鎖モードのコミット要求をした場合、ee_trn_chained_commit 関数はエラーリターンし、エラートランザクション（ERRTRNR）を起動しません。
4. DB キューグループを指定してメッセージの書き込み要求を行った場合は、eedbqwtpr コマンドで書き込みを抑止している DB キューに対して eedbqislt コマンドを実行して縮退させることによって、DB キューグループ指定の対象外とする必要があります。
5. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション（RM）
6. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数実行後の同期点処理（UAP リターン, ee_trn_chained_commit 関数実行）のメッセージ表へのアクセスで障害が発生すると、トランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション（ERRTRN3）のときは、リトライロールバックしません。

ee_dbq_msgrefer

名称

メッセージの参照読み出し

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_msgrefer (char *dbque_name, EEULONG read_num,
                    EEULONG buf_len, void *buf_pt,
                    EEULONG *out_len, EEULONG flags)
```

機能

読み出し済みメッセージを読み出します（参照読み出し）。

UAP で値を設定する引数

●dbque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●read_num

読み出すメッセージの通番を設定します。

●buf_len

読み出したメッセージを格納する領域の長さを設定します（単位：バイト）。1 以上の値を指定してください。

●buf_pt

読み出したメッセージを格納する領域のアドレスを設定します。

●out_len

読み出したメッセージの長さを格納する領域のアドレスを設定します。

●flags

EENOFLLAGS を設定します。

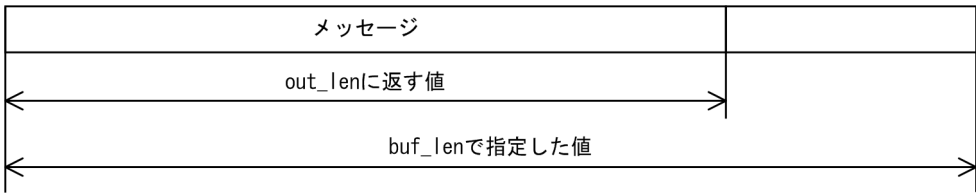
TP1/EE から値が返される引数

●buf_pt

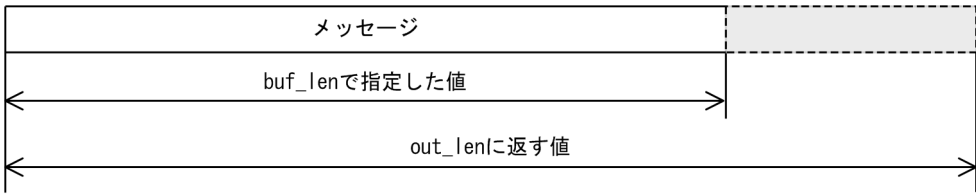
読み出したメッセージが返されます。

●out_len

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（リターン値 EEDBQER_BUF_OVERFLOW）は次のようになります。



リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_BUF_LEN	バッファ長が指定範囲外です。
EEDBQER_BUF_OVERFLOW	読み出したメッセージがバッファをオーバーフローしました。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	要求された DB キューは存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_QUEUE_TYPE	設定された DB キューのタイプは DB キュー機能で使用できません。

リターン値	意味
EEDBQER_QUEUE_TYPE	相手読み出し DB キュー
EEDBQER_READ_NUM	引数 read_num に指定した値は不当な値です。 1～最大メッセージ数の指定範囲外の通番です。 - 読み出し済みの通番ではありません。
EEDBQER_RM_ACCESS	リソースマネージャへのアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 通番の初期化中です。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

- 【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
 MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション (RM)
- 【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション (ERRTRN3) のときは、リトライロールバックしません。

ee_dbq_msgrefer2

名称

メッセージの参照読み出し

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_msgrefer2 (char *dbque_name, EEULONG read_num,
                     EEULONG buf_len, void *buf_pt,
                     EEULONG *out_len, EEULONG flags)
```

機能

通番指定で、メッセージを読み出します（参照読み出し）。

他サーバ読み出しの DB キューのメッセージや、未読み出しのメッセージを読み出せます。

UAP で値を設定する引数

●dbque_name

読み出し対象の DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●read_num

読み出し対象のメッセージの通番を設定します。

●buf_len

読み出したメッセージを格納する領域の長さを設定します。65 以上の値を指定してください。

●buf_pt

読み出したメッセージを格納する領域のアドレスを設定します。読み出したいメッセージの長さ + 64 バイトの領域を用意してください。先頭から 64 バイトは TP1/EE で使用します。

buf_pt には、8 バイトで境界調整した領域のアドレスを指定してください。

●out_len

読み出したメッセージの長さを格納する領域のアドレスを設定します。

●flags

EENOFLLAGS を設定します。

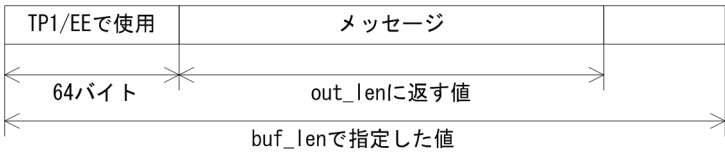
TP1/EE から値が返される引数

●buf_pt

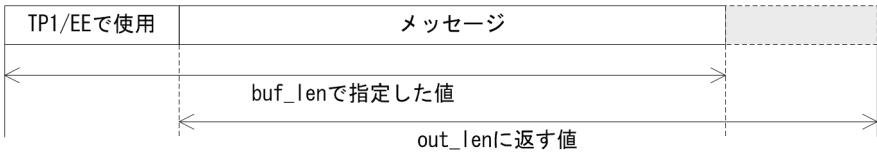
読み出したメッセージが返されます。読み出したメッセージの先頭は buf_pt + 64 バイトのアドレスです。

●out_len

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（リターン値 EEDBQER_BUF_OVERFLOW）は次のようになります。



リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_BUF_LEN	バッファ長が指定範囲外です。
EEDBQER_BUF_OVERFLOW	読み出したメッセージがバッファをオーバーフローしました。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	要求された DB キューは存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_READ_NUM	read_num に指定した値は不当な値です。

リターン値	意味
EEDBQER_READ_NUM	1～最大メッセージ数の指定範囲外の通番 - 書き込まれていない通番
EEDBQER_RM_ACCESS	リソースマネージャアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 通番の初期化中です。
EEDBQER_UNRESOLVABLE	その他のエラーが発生しました。

注意事項

1. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション (RM)
2. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション (ERRTRN3) のときは、リトライロールバックしません。

ee_dbq_obscancel

名称

オンラインバッチ処理の強制終了

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obscancel(char *batchque_name, char *lot_name,
                    EELONG flags)
```

機能

オンラインバッチ処理を中断中のロットに対して、オンラインバッチ処理の強制終了要求を行います。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。

リターン値	意味
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンクされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 - 障害による読み出し中断中 - 運用による読み出し中断中 - 回復処理失敗による中断中 - 正常終了トランザクション障害による中断中 - 回復処理失敗による正常終了トランザクション中断中 - 正常終了トランザクション障害による中断中の回復処理失敗 - 該当するロットに対して API 関数を呼び出したトランザクションが、コミットまたはロールバックしている
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は強制終了します。
2. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの API 関数は、ee_dbq_obsinquire 関数および ee_dbq_obschdata 関数だけです。ほかのオンラインバッチの API 関数は受け付けません。

名称

引き継ぎ情報の更新

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obschdata(char *batchque_name, char *lot_name,
                    char *cont_pt, EEULONG cont_size,
                    EEULONG flags)
```

機能

オンラインバッチ処理中のロットの引き継ぎ情報を更新します。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●cont_pt

引き継ぎ情報を格納している領域のアドレスを設定します。

●cont_size

引き継ぎ情報のサイズを設定します。1～256 バイトの範囲で設定できます。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。

リターン値	意味
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_CONT_LEN	引き継ぎ情報を格納している領域のサイズが指定範囲外です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 • 実行中 • 障害による読み出し中断中 • 運用による読み出し中断中 • 回復処理失敗による中断中 • 正常終了トランザクション障害による中断中 • 回復処理失敗による正常終了トランザクション中断中 • 正常終了トランザクション障害による中断中の回復処理失敗 • 終了トランザクション実行中 • 回復処理失敗による終了トランザクション実行中
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. バッチ処理の開始時に引き継ぎ情報が指定されなかった場合は、この関数で指定された引き継ぎ情報を追加します。
2. 呼び出し元トランザクションの決着結果がコミットかまたはロールバックかに関係なく、引き継ぎ情報が更新されます。
ただし、強制停止中の場合、引き継ぎ情報は呼び出し元トランザクションの決着を待たないで即時更新されます。
3. トランザクション内で複数の要求があった場合は、最新の要求が有効となります。

ee_dbq_obsend

名称

オンラインバッチ処理の終了

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsend(char *batchque_name, char *lot_name,
                  EELONG flags)
```

機能

イベント型オンラインバッチ処理の場合に、オンラインバッチ処理の終了要求を行います。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。

リターン値	意味
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_OBJECT_TRN	対象ロットのサービストランザクションから呼び出されていません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 • 実行中 • 該当するロットに対して API 関数を呼び出したトランザクションが、コミットまたはロールバックしている
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. この API 関数は、対象ロットのサービストランザクションからだけ呼び出せます。
2. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は終了します。
3. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの API 関数は、ee_dbq_obsinquire 関数および ee_dbq_obschdata 関数だけです。ほかのオンラインバッチの API 関数は受け付けません。
4. この API 関数の動作はオンラインバッチ機能関連定義の dbq_obs_trn_end_api オペランドの指定値に依存します。dbq_obs_trn_end_api オペランドの指定値が Y の場合、ee_trn_chained_commit 関数または ee_trn_chained_rollback 関数の呼び出し以降、この API 関数は受け付けません。

ee_dbq_obsinquire

名称

ロットの情報の照会

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsinquire(char *batchque_name, char *lot_name,
                      EEDBQOINF *dbqobs_inf, EELONG flags)
```

機能

ロットの情報を照会します。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●dbqobs_inf

照会した DB キュー情報を格納するための構造体ポインタのアドレスを設定します。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●dbqobs_inf

DB キュー情報が構造体 EEDBQOINF で返されます。構造体の形式は次のとおりです。

```
struct ee_dbq_obsinf_t{
    char    service_name[32];
    char    dbque_name[28];
    char    lot_name[24];
    EELONG  sts_code;
    EELONG  lot_type;
    EELONG  write_event_num;
```

```

EEULONG read_num;
EEULONG skip_num;
EEULONG unread_count;
}EEDBQOINF;

```

- service_name (サービス名)
指定したロットに現在接続しているサービス名を返します。※1
- dbque_name (DB キュー名)
batchque_name に設定した DB キュー名をそのまま返します。
- lot_name (ロット名)
lot_name に設定したロット名をそのまま返します。
- sts_code (ステータスコード)
ロットの状態を返します。

EEDBQ_OBS_STATUS_NORMAL

実行中

EEDBQ_OBS_STATUS_PSTOP

障害による読み出し中断中

EEDBQ_OBS_STATUS_CSTOP

運用による読み出し中断中

EEDBQ_OBS_STATUS_DEACT

未起動

EEDBQ_OBS_STATUS_RSTOP

回復処理失敗による中断中

EEDBQ_OBS_STATUS_NSTOP

正常終了トランザクション障害による中断中

EEDBQ_OBS_STATUS_RNSTOP

回復処理失敗による正常終了トランザクション中断中、または正常終了トランザクション障害による中断中の回復処理失敗

EEDBQ_OBS_STATUS_END_EXEC

正常、中断、または強制終了トランザクション実行中

EEDBQ_OBS_STATUS_RRN_EXEC

回復処理失敗による中断の終了トランザクション実行中

- lot_type (該当するロットのオンラインバッチ処理形態)

ロットのオンラインバッチ処理形態を返します。※1

EEDBQ_OBS_TYPE_DATA

データ型オンラインバッチ処理

EEDBQ_OBS_TYPE_EVNT

イベント型オンラインバッチ処理

- write_event_num (該当するロットの全メッセージ数)
DB キュー内の該当するロットのメッセージ数を返します。※1, ※2
- read_num (読み出し済みメッセージ数)
該当するロットの読み出し済みメッセージ数を返します。※1, ※2
- skip_num (スキップメッセージ数)
該当するロットのスキップメッセージ数を返します。※1, ※2
- unread_count (未読み出しメッセージ数)
未読み出しメッセージ数を返します。※1, ※2

注※1

ステータスコードが EEDBQ_OBS_STATUS_DEACT 以外のときに値が返ります。

注※2

該当するロットの lot_type が EEDBQ_OBS_TYPE_EVNT で、ステータスコードが EEDBQ_OBS_STATUS_RSTOP, EEDBQ_OBS_STATUS_RNSTOP, または EEDBQ_OBS_STATUS_RRN_EXEC のときは値が返りません。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDCOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDCOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EEDCOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

名称

ロールバック要求時のメッセージの読み出しスキップ

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsrbskip (EELONG flags)
```

機能

データ型オンラインバッチ処理の場合で、トランザクションのロールバック要求時に、読み出したメッセージのロールバック処理（メッセージ読み出し通番の戻し）をしないようにします。コミット要求があった場合には、この機能は無効となります。

UAP で値を設定する引数

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_OBJECT_TRN	対象ロットのサービストランザクションから呼び出されていません。
EEDBQER_RETRY_OVER	次にロールバック要求すると、同一メッセージ処理のロールバック監視回数を超えるため、要求を受け付けません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。

注意事項

1. この API 関数は、対象ロットのサービストランザクションからだけ呼び出せます。
2. この API 関数を呼び出したトランザクションの実行中に計画停止 B でシステムを停止すると、API 関数は正常に終了しますが、メッセージの読み出しはスキップされない場合があります。また、読み出しスキップ処理中に障害が発生すると、メッセージの読み出しがスキップされない場合があります。この場合、同一メッセージを読み出したときに、再度この API 関数を呼び出してください。
3. 呼び出し元トランザクションがロールバックしたときに、メッセージの読み出しがスキップされます。
4. この API 関数の動作はオンラインバッチ機能関連定義の `dbq_obs_trn_end_api` オペランドの指定値に依存します。`dbq_obs_trn_end_api` オペランドの指定値が Y の場合、`ee_trn_chained_commit` 関数または `ee_trn_chained_rollback` 関数の呼び出し以降、この API 関数は受け付けません。

ee_dbq_obsrestart

名称

オンラインバッチ処理の再開始

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsrestart(char *batchque_name, char *lot_name,
                     EELONG flags)
```

機能

オンラインバッチ処理の再開始要求を行います。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。

リターン値	意味
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 • 障害による読み出し中断中 • 運用による読み出し中断中 • 回復処理失敗による中断中 • 正常終了トランザクション障害による中断中 • 回復処理失敗による正常終了トランザクション中断中 • 正常終了トランザクション障害による中断中の回復処理失敗 • 該当するロットに対して API 関数を呼び出したトランザクションが、コミットまたはロールバックしている
EEDBQER_RM_ACCESS	リソースマネージャへのアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HiRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は再開始します。
2. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの API 関数は、ee_dbq_obsinquire 関数および ee_dbq_obschdata 関数だけです。ほかのオンラインバッチの API 関数は受け付けません。

ee_dbq_obsskip

名称

メッセージの読み出しスキップ

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsskip(EEULONG skip_flg, char *batchque_name,
                  char *lot_name, EELONG flags)
```

機能

データ型オンラインバッチ処理の場合に、指定されたロットのメッセージを読み出し済みにします。

UAP で値を設定する引数

●skip_flg

次のフラグを設定します。

EEDBQ_OBS_SKIP_ALL

指定されたロットのすべてのメッセージを読み出し済みにします。

EEDBQ_OBS_SKIP_ONE

未読み出しの先頭にある 1 メッセージだけを読み出し済みにします。

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。

リターン値	意味
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_LOT_TYPE	該当するロットのタイプはこの要求では受け付けません。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 - 障害による読み出し中断中 - 運用による読み出し中断中 - 該当するロットに対して API 関数を呼び出したトランザクションが、コミットまたはロールバックしている
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_SKIPMSG_NOTHING	この要求で読み出し済みにするメッセージはありません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

- スキップの処理中（コミットされるまでの間）は別のスキップを受け付けません。
- 運用または障害によって、該当するロットが中断中の場合だけ呼び出せます。
- 呼び出し元トランザクションがコミットしたときに、メッセージが読み出し済みになります。
- 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの API 関数は、ee_dbq_obsinquire 関数、ee_dbq_obschdata 関数だけです。ほかのオンラインバッチの API 関数は受け付けません。

ee_dbq_obsstart

名称

オンラインバッチ処理の開始

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsstart(char *batchque_name, char *lot_name,
                    EEDBQOBSST *start_inf, EELONG flags)
```

機能

オンラインバッチ処理の開始要求を行います。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

データ型オンラインバッチ処理を起動する場合は、定義ファイルで定義されている DB キュー名を指定してください。イベント型オンラインバッチ処理を起動する場合は、任意の DB キュー名を指定してください。オンラインバッチ処理開始以降の関数を呼び出すときに指定する DB キュー名は、この関数で指定した DB キュー名を指定します。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●start_inf

DB キュー情報を構造体 EEDBQOBSST で設定します。構造体の形式は次のとおりです。

32 ビットアーキテクチャの場合

```
struct ee_dbq_obsstart_t{
    char    service_name[32];
    EELONG lot_type;
    char    *cont_pt;
    EELONG cont_sz;
}EEDBQOBSST;
```

64 ビットアーキテクチャの場合

```
struct ee_dbq_obsstart_t{
    char    service_name[32];
    EEULONG lot_type;
    char    yobi_1[4];
    char    *cont_pt;
    EEULONG cont_sz;
    char    yobi_2[4];
}EEDBQOBSST;
```

- service_name (サービス名)

該当するロットでの読み出しトランザクションを起動するサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。文字列の先頭に NULL 文字を設定した場合、定義の指定を有効にします。

- lot_type (ロットのオンラインバッチ処理形態)

次のフラグを設定します。

EEDBQ_OBS_TYPE_DATA

データ型オンラインバッチ処理を起動します。

EEDBQ_OBS_TYPE_EVNT

イベント型オンラインバッチ処理を起動します。

- cont_pt (引き継ぎ情報を格納している領域のアドレス)

引き継ぎ情報を格納している領域のアドレスを設定します。アドレスを設定する必要がない場合は、NULL 文字を設定してください。

- cont_sz (引き継ぎ情報を格納している領域のサイズ)

引き継ぎ情報を格納している領域のサイズを設定します。0~256 バイトの範囲で設定できます。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。

リターン値	意味
EEDBQER_BEFORE_ROLLBACK	同一トランザクションで以前に行われた開始要求がエラーリターンしたあと、ロールバック処理が行われていないため、要求を受け付けません。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_CONT_LEN	引き継ぎ情報を格納している領域のサイズが指定範囲外です。
EEDBQER_ENDSERVICE_STOP	オンラインバッチ処理を終了するためのサービスにエントリポイントが存在しないため、要求を受け付けません。
EEDBQER_LOT_SHORT	ロット管理テーブル面数が不足しました。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	オンラインバッチ機能は使用できません。
EEDBQER_NO_SERVICE	引数に設定したサービス名は存在しません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 - 初期状態 - 該当するロットに対して API 関数を呼び出したトランザクションが、コミットまたはロールバックしている
EEDBQER_RM_ACCESS	リソースマネージャへのアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_SERVICE_STOP	引数に設定したサービス名はエントリポイントが存在しないため使用できません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. リターン値が EE_OK 以外の場合、必ずロールバックを行ってください。
2. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は開始します。
3. 呼び出し元トランザクションがロールバックまたはコミットするまでの間、該当するロットが受け付けられるオンラインバッチの API 関数は、ee_dbq_obsinquire 関数だけです。ほかのオンラインバッチの API 関数は受け付けません。

ee_dbq_obsstop

名称

オンラインバッチ処理の中断

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_obsstop(char *batchque_name, char *lot_name,
                  EELONG flags)
```

機能

オンラインバッチ処理の中断要求を行います。

UAP で値を設定する引数

●batchque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●lot_name

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	オンラインバッチ機能のライブラリがリンケージされていません。

リターン値	意味
EEDBQER_NO_LOT	引数に設定したロット名は存在しません。または、対象のロットではありません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_OBJECT_TRN	対象ロットのサービストランザクションから呼び出されていません。
EEDBQER_PROHIBIT	該当するロットが次に示す状態ではないため、要求を受け付けられません。 • 実行中
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. この関数は、対象ロットのサービストランザクションからだけ呼び出せます。
2. 呼び出し元トランザクションの決着結果がコミットかまたはロールバックかに関係なく、オンラインバッチ処理が中断します。

名称

ロールバック要求時のメッセージの読み出しスキップ

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_rbskip(EELONG flags)
```

機能

トランザクションのロールバック要求時に、読み出したメッセージのロールバック処理（メッセージ読み出し通番の戻し）をしないようにします。トランザクションのコミット要求があった場合、この機能は無効になります。この関数を呼び出せるのは、DB キューサービストランザクションだけです。

UAP で値を設定する引数

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_RETRY_OVER	次にロールバック要求すると、同一メッセージ処理のロールバック監視回数を超えるため、要求を受け付けません。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。

注意事項

この関数を呼び出したトランザクションの実行中に計画停止 B でシステムを停止すると、関数は正常に終了しますが、メッセージの読み出しはスキップされない場合があります。また、メッセージの読み出しスキップ処理中に障害が発生すると、読み出しはスキップされない場合があります。同一メッセージを読み出したときに、この関数を再度呼び出してください。

ee_dbq_skip

名称

メッセージの読み出しスキップ

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_skip(EEULONG skip_flg, EEULONG read_num,
               char *dbque_name, EELONG flags)
```

機能

DB キューのメッセージを読み出し済みにして、読み出しをスキップします。

メッセージを読み出し済みに更新するための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定する引数

●skip_flg

次のフラグを設定します。

EEDBQ_SKIP_ALL

読み出し通番が書き込み通番と同じになるまで、すべてのメッセージを読み出し済みにします。

EEDBQ_SKIP_ONE

未読み出しの先頭にある 1 メッセージだけを読み出し済みにします。

EEDBQ_SKIP_NUM

read_num で設定された読み出し通番まで、メッセージを読み出し済みにします。

●read_num

メッセージ読み出し通番を設定します。

skip_flg で EEDBQ_SKIP_NUM を設定した場合に、この引数が有効になります。

●dbque_name

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDCOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDCOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EEDCOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_MEMORY_SHORT	メモリが不足しました。
EEDBQER_NO_DBQUE	引数に設定した DB キュー名は存在しません。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンケージされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_QUE_TYPE	設定された DB キューのタイプは DB キュー機能で使用できません。 相手読み出し DB キュー
EEDBQER_READ_NUM	引数 read_num に指定した値は不当な値です。
EEDBQER_RM_ACCESS	リソースマネージャへのアクセス時にエラーが発生しました。
EEDBQER_RM_CONNECT	該当するスレッドではリソースマネージャに接続されていません。
EEDBQER_ROLLBACK_RESERVE	HIRDB の暗黙的ロールバックが発生したため、ee_trn_rollback_mark 関数を呼び出しました。
EEDBQER_SKIPMSG_NOTHING	この要求で読み出し済みにするメッセージはありません。
EEDBQER_STOP	DB キューが使用できない状態です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。 - 読み出し停止中ではありません。 - 通番の初期化中です。

注意事項

- スキップの対象になるのは、トランザクションのコミットによって書き込みが完了したメッセージです。コミットしていない場合は、メッセージをスキップできません。

2. メッセージ読み出し通番は、トランザクションのコミット時に更新されます。
3. メッセージをスキップしたあと、トランザクションのコミットまでは、さらにスキップすることはできません。
4. 該当する DB キューが読み出し停止の状態（コマンド、TP1/EE サービス定義、API 関数で指定するか、または障害発生による）にある場合だけ、メッセージをスキップできます。
5. 同一グローバルトランザクションのトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだりスキップしたりすると、デッドロックが発生する場合があります。
6. DB キューの読み出しトランザクションで、該当する DB キューを読み出し停止状態にしている場合、メッセージをスキップできません。別のトランザクションで `ee_dbq_skip` 関数を呼び出すか、またはコマンドを実行してメッセージをスキップしてください。

名称

DB キューサービスの閉塞制御

形式

ANSI C, C++の形式

```
#include <eedbq.h>
int ee_dbq_srvctl(EELONG mode, char *svname, EELONG flags)
```

機能

DB キューサービスを閉塞，および閉塞解除します。

UAP で値を設定する引数

●mode

次のフラグを設定します。

EEDBQ_SRVCTL_DCT

DB キューサービスを閉塞します。

EEDBQ_SRVCTL_ACT

DB キューサービスの閉塞を解除します。

●svname

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEDBQER_ARGUMENT	引数に設定した値が間違っています。

リターン値	意味
EEDBQER_CONDITION	関数の呼び出し条件が不正です。
EEDBQER_NO_LIBRARY	DB キュー機能のライブラリがリンクされていません。
EEDBQER_NO_OPTION	DB キュー機能は使用できません。
EEDBQER_NO_SERVICE	引数に設定した DB キューサービス名は存在しません。
EEDBQER_SERVICE_STOP	DB キューサービスが永久閉塞中（エントリポイントなし）です。
EEDBQER_TIMING	この API 関数を呼び出せる状態ではありません。
EEDBQER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

次の場合、ee_dbq_srvctl 関数はリターン値 EE_OK でリターンします。

- 閉塞状態の DB キューに対して閉塞を要求した場合
- 閉塞解除状態の DB キューに対して閉塞解除を要求した場合

資源の排他制御 (ee_lck_～)

任意の資源を排他制御する関数について説明します。資源の排他制御の関数を次に示します。

- ee_lck_get – 資源の排他
- ee_lck_release_all – 全資源の排他の解除
- ee_lck_release_byname – 資源名称を指定した排他の解除

ee_lck_get

名称

資源の排他

形式

ANSI C, C++の形式

```
#include <eelck.h>
int ee_lck_get(char *name, EELONG lockmode, EELONG ownerflag,
               EELONG flags)
```

機能

UAP で使用する資源の排他を指定します。排他の管理は、TP1/EE のトランザクションマネージャで管理するグローバルトランザクション単位で処理されます。

ここで指定した排他は、排他を解除する関数（ee_lck_release_all 関数または ee_lck_release_byname 関数）を呼び出すか、または ee_lck_get 関数を呼び出したグローバルトランザクションの同期点取得後に解除されます。

UAP で値を設定する引数

●name

排他する資源の名称を、16 バイトの英数字で設定します。ここで設定した資源名称を基に、ロックサービスで排他を管理します。16 バイト未満で、NULL 文字がある場合は、そこまでの値が資源名称と見なされます。16 バイトを超えた値を設定した場合は、16 バイト目までが資源名称と見なされて、超えた部分が切り捨てられます。ロックサービスでは、文字列の内容についてはチェックしません。論理的に正しい名称を設定してください。資源名称に英数字以外の値を使った場合は、タイムアウト情報、および eelckls コマンドの表示が乱れることがあります。

●lockmode

排他制御モードとして、次のどちらかを設定します。重複して指定することはできません。

EELCK_PR

資源を参照します。ほかの UAP には参照だけを許可し、更新を禁止します。

EELCK_EX

資源を更新します。ほかの UAP には参照と更新の両方を禁止します。

●ownerflag

EELCK_OWNER_MIGRATE を設定します。

●flags

資源の排他に関するフラグを設定します。設定できる値を次に示します。

EELCK_WAIT

ほかの UAP と資源を競合した場合に、資源を排他できるまで待ちます。競合した場合にこのフラグが設定されていないときは、エラーリターンします。EELCK_TEST と同時に設定したときは、このフラグの設定は無効となります。

EELCK_TEST

資源が使えるかどうかをテストする場合に設定します。このフラグを設定したときは、ee_lck_get 関数が正常終了しても、name に設定した資源は確保されていないので注意してください。

EENOFLLAGS

フラグを設定しません。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EELCKER_CONDITION	この機能は使用できません。
EELCKER_MEMORY	排他制御用のテーブルが不足しています。
EELCKER_OUTOFTRN	トランザクション処理でない UAP から指定しています。
EELCKER_PARAM	引数に設定した値が間違っています。
EELCKER_TIMEOUT	排他待ち限界経過時間（トランザクション関連定義の lck_wait_timeout オペランドの指定値）を超えたため、資源を確保できませんでした。
EELCKER_WAIT	ほかの UAP が name に設定した資源を使っています。

ee_lck_release_all

名称

全資源の排他解除

形式

ANSI C, C++の形式

```
#include <eelck.h>
int ee_lck_release_all(EELONG ownerflag, EELONG flags)
```

機能

ee_lck_get 関数で指定した資源の排他をすべて解除します。同期点取得前に排他を解除するときに、ee_lck_release_all 関数を呼び出します。

排他をしたグローバルトランザクションが終了した場合、TP1/EE のロックサービスによって自動的に排他は解除されます。このときは、UAP で排他を解除する必要はありません。

UAP で値を設定する引数

●ownerflag

EELCK_OWNER_MIGRATE を設定します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EELCKER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EELCKER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EELCKER_ENVIRON	TP1/EE の環境下ではありません。
EELCKER_CONDITION	この機能は使用できません。
EELCKER_NOTHING	この関数を呼び出したトランザクションでは、資源を確保していません。
EELCKER_OUTOFTRN	トランザクション処理でない UAP から ee_lck_release_all 関数を呼び出しています。
EELCKER_PARAM	引数に設定した値が間違っています。

ee_lck_release_byname

名称

資源名称を指定した排他解除

形式

ANSI C, C++の形式

```
#include <eelck.h>
int ee_lck_release_byname(char *name, EELONG ownerflag,
                          EELONG flags)
```

機能

ee_lck_get 関数で指定した資源の排他を、資源名称を指定して解除します。同期点取得前に排他を解除するときに、ee_lck_release_byname 関数を呼び出します。

排他をしたグローバルトランザクションが終了したときに、TP1/EE のロックサービスによって自動的に排他は解除されます。このときは、UAP で排他を解除する必要はありません。

UAP で値を設定する引数

●name

排他の指定を解除する資源名称を設定します。資源名称は ee_lck_get 関数で設定した名称と同じ値を設定してください。

●ownerflag

EELCK_OWNER_MIGRATE を設定します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EELCKER_CONDITION	この機能は使用できません。
EELCKER_NOTHING	解除を指定した資源名称に該当する資源がありません。

リターン値	意味
EELCKER_OUTOFTRN	トランザクション処理でない UAP から ee_lck_release_byname 関数を呼び出しています。
EELCKER_PARAM	引数に設定した値が間違っています。

メッセージログの出力 (ee_log～)

UAP からメッセージログを出力する関数について説明します。メッセージログの出力の関数を次に示します。

- ee_logprint – メッセージログの出力

ee_logprint

名称

メッセージログの出力

形式

ANSI C, C++の形式

```
#include <eelog.h>
int ee_logprint(char *msgid, char *pgm_id, char *string, char *info,
                EELONG flags)
```

機能

引数に設定した文字列に、行ヘッダ、メッセージ ID、および要求元プログラム ID を付けて、メッセージログファイル、syslog、標準出力に出力します。メッセージログファイルの行ヘッダは、プロセス ID、メッセージ通番、サービス名、中央処理通番、年月日、時刻です。syslog の行ヘッダは定義で設定します。syslog 出力は、flags に設定した出力レベルが定義に指定した出力レベルと同じか高い場合です。flags に出力レベルを設定しなかった場合の出力レベルは、1 です。

標準出力への出力有無は、次の表のようになります。定義：log_usermsg_console に all を指定している場合、flags に設定する標準出力への出力フラグの設定に関係なく、標準出力へ出力します。

定義： log_msg_console	定義： log_usermsg_console	API : flags 標準出力への出力フラグの設定	標準出力への出力有無
Y (デフォルト)	all	有り	○
		無し (デフォルト)	○
	api (デフォルト)	有り	○
		無し (デフォルト)	×
N	—	—	×

(凡例)

- ：メッセージを標準出力に出力します。
- ×

TP1/EE では、ee_logprint 関数で使うメッセージ ID 用に、05000 から 06999 までの範囲の番号を割り当てています。UAP から出力するメッセージ ID の番号には、05000 から 06999 までの範囲の値を付けてください。

メッセージテキストに復改文字「¥n」を含むメッセージログは、次のように扱います。

メッセージの出力先	復改文字「¥n」の扱い
メッセージログファイル	空白に置き換えて 1 行で出力する
syslog	空白に置き換えて 1 行で出力する
標準出力	そのまま出力する（「¥n」で改行する）

障害が発生して UAP からメッセージログが出力できない場合でも、ee_logprint 関数がリターン値 EE_OK で正常に終了することがあります。そのため、メッセージログが抜ける場合があります。メッセージログの抜けは、メッセージログに付けるメッセージログ通番で確認できます。

一つのスレッドから複数回 ee_logprint 関数を呼び出した場合、メッセージログファイルへの出力順序は保証されます。複数のスレッドから別々に ee_logprint 関数を呼び出した場合は、呼び出した順にメッセージログファイルに出力されないことがあります。

エラーを検出した場合、エラーの原因を示すメッセージログに ee_logprint 関数に指定したメッセージを付けて、標準エラー出力に出力します。

UAP で値を設定する引数

●msgid

メッセージログごとに付ける識別子（メッセージ ID）を設定します。「KF SBnnnnnn-x」の形式（11 文字）で、末尾に NULL 文字を付けてください。

TP1/EE では、ee_logprint 関数で使うメッセージ ID の通番（nnnnnn の部分）用に、05000～06999 の範囲の番号を割り当てています。UAP から出力するメッセージ ID の通番は、05000～06999 の範囲の値にしてください。

●pgm_id

ee_logprint 関数を呼び出した UAP を識別する任意の値（要求元プログラム ID）を設定します。英数字 2 文字で、末尾に NULL 文字を付けてください。

●string

メッセージログファイルにメッセージログとして出力したい任意の文字列を設定します。最大 222 文字で、末尾に NULL 文字を付けてください。

文字列に改行「¥n」が含まれる場合、1 行で出力されます。

●info

NULL を設定します。

●flags

syslog 出力レベルと標準出力への出力フラグを設定します。

syslog 出力レベルだけを設定する場合

syslog 出力レベルを表す次のフラグを設定します。

EE_NOFLAGS

特定のフラグは設定しません。syslog 出力レベルはデフォルトの 1 になります。

EELOG_FLAG_SYSLOG_L1

syslog 出力レベルを 1 にします。

EELOG_FLAG_SYSLOG_L2

syslog 出力レベルを 2 にします。

EELOG_FLAG_SYSLOG_L3

syslog 出力レベルを 3 にします。

EELOG_FLAG_SYSLOG_L4

syslog 出力レベルを 4 にします。

EELOG_FLAG_SYSLOG_L5

syslog 出力レベルを 5 にします。

EELOG_FLAG_SYSLOG_L6

syslog 出力レベルを 6 にします。

EELOG_FLAG_SYSLOG_NOPRINT

メッセージを syslog に出力しません。

syslog 出力レベルは、1 が最も高く、6 が最も低くなります。flags に設定した出力レベルがメッセージログ関連定義の log_syslog_out オペランドに指定した出力レベルと同じか高い場合、syslog にメッセージログが出力されます。flags に出力レベルを設定しなかった場合の出力レベルは 1 です。

syslog 出力レベルと標準出力への出力フラグを設定する場合

syslog 出力レベルフラグとの論理和で設定します。

EELOG_FLAG_CONSOLE

メッセージを標準出力に出力します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EELOGER_ARGUMENT	引数に設定した値が間違っています。
EELOGER_CONDITION	メッセージを出力できる状態ではありません。

リターン値	意味
EELOGER_EXCLOCK	メッセージログ機能内の排他設定に失敗しました。
EELOGER_EXCUNLOCK	メッセージログ機能内の排他解除に失敗しました。
EELOGER_FILE_OVER	すべてのメッセージログファイル切り替えに失敗しました。
EELOGER_FILEOUT	メッセージログファイルへの出力に失敗しました。
EELOGER_SNPRINTF	メッセージの組み立てに失敗しました。
EELOGER_STDERR	標準エラー出力に失敗しました。
EELOGER_SYSLOG	syslog への出力に失敗しました。

注意事項

1. ログ出力量が多い場合は、ee_logprint 関数のリターンが遅くなります。例えば、障害発生時にメッセージ出力量が著しく多くなると、トランザクション処理時間が長くなります。これはスローダウンの要因になるので、注意してください。
2. ee_logprint 関数で標準出力に非 ASCII 文字を含むメッセージを出力する場合、標準出力に複数の文字コードが混在する可能性があります。この場合は次のどちらかの方法で回避してください。
 - 出力メッセージを ASCII 文字だけにする
 - flags に EELOG_FLAG_CONSOLE を指定しない

高速メッセージ送信 (ee_mch_～)

クライアントとサーバの間で UDP プロトコルで高速通信するときに使用する関数について説明します。
高速メッセージ送信の関数を次に示します。

- ee_mch_cmtsend 【XTC 限定】 – トランザクション非同期の一方送信メッセージの送信
- ee_mch_cmtsend_get 【XTC 限定】 – 一方送信メッセージ情報の取得
- ee_mch_cmtsend_sync 【XTC 限定】 – トランザクション同期の一方送信メッセージの送信
- ee_mch_otqbkctl 【XTC 限定】 – 出力キュー (OTQ) の閉塞制御
- ee_mch_otqinquire 【XTC 限定】 – 出力キュー (OTQ) の情報の照会
- ee_mch_otqskip 【XTC 限定】 – 一方送信メッセージの送信スキップ

ee_mch_cmtsend 【XTC 限定】

名称

トランザクション非同期の一方送信メッセージの送信

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_cmtsend(char *group, char *service,
                  char *in, EEULONG *in_len,
                  EEULONG priority, EEULONG flags)
```

機能

トランザクションのコミットが確定したとき、出力キュー（OTQ）の通番を採番して出力キューにメッセージを格納します。その後、送信用のスレッドを使用して出力キューの通番順に一方送信メッセージをUDP プロトコルで送信します。これによって、SPP に対してサービスを要求します。トランザクションがロールバックとなった場合、メッセージは破棄されます。

サービスを要求するときには、「サービスグループ名」および「サービス名」を引数に設定します。この名称に該当するサービス関数へサービスが要求されます。

この関数を使う場合、サーバ UAP があるノードの TP1/EE が稼働していることが前提です。

UAP で値を設定する引数

●group

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。

文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメタ長を設定します。1～EEMCH_MAX_MESSAGE_SIZE※の範囲の値を設定できます。EEMCH_MAX_MESSAGE_SIZE は、eemch.h で定義してあります。

注※

EEMCH_MAX_MESSAGE_SIZE の値は、将来拡張される場合があります。

●priority

メッセージ送信先で登録された処理キューの中で優先的に起動するかどうかを設定します。

EEMCH_HI

優先的に起動します。

EEMCH_LOW

優先的に起動しないで、登録した順序で起動します。

●flags

メッセージ送信形態とオプションを次に示す形式で設定します。

```
{EENOFLAGS | EEMCH_MULTI_CL | EEMCH_MYPROC} [ | EEMCH_PERMANENCE]
```

EENOFLAGS

ユニキャストで一方送信メッセージを送信する場合に設定します。

EEMCH_MULTI_CL

マルチキャストで CL サーバに一方送信メッセージを送信する場合に設定します。

EEMCH_MYPROC

自プロセスあてに一方送信メッセージを送信する場合に設定します。

EEMCH_PERMANENCE

メッセージを永続属性とする場合に設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_MEMORY_SHORT	メモリが不足しました。
EEMCHER_MESSAGE_TOO_BIG	in_len に設定した入力パラメタ長が、最大値を超えています。
EEMCHER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。
EEMCHER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。

リターン値	意味
EEMCHER_OTQ_SHUT_DOWN	出力キュー（OTQ）が閉塞中、または閉塞解除処理中です。
EEMCHER_PARAM_FLAGS	flags に設定した値が間違っています。
EEMCHER_PARAM_GROUP	group に設定した値が間違っています。
EEMCHER_PARAM_IN	in に設定した値が間違っています。
EEMCHER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EEMCHER_PARAM_PRIORITY	priority に設定した値が間違っています。
EEMCHER_PARAM_SERVICE	service に設定した値が間違っています。
EEMCHER_SYSERR	システムエラーが起きました。

ee_mch_cmtsend 関数がエラーになる場合

ee_mch_cmtsend 関数がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合、送信先サービス関連定義を指定していない場合、トランザクション処理でない UAP から要求された場合、終了処理中に要求された場合、初期化トランザクション（MI）もしくは終了トランザクション（ME）から要求された場合、または ee_trn_rollback_mark 関数が発行済みである場合

ee_mch_cmtsend 関数は EEMCHER_CONDITION のリターン値でエラーリターンします。

flags に EEMCH_MYPROC を設定して自プロセス送信を行う場合で、送信先サービス関連定義に自サービス名（自サービスグループ名）が設定されていないとき

ee_mch_cmtsend 関数は EEMCHER_NO_SUCH_SERVICE (EEMCHER_NO_SUCH_SERVICE_GROUP) のリターン値でエラーリターンします。

メモリ関連定義の udp_send_message_buf_cnt オペランドで指定した UDP 用送信バッファ面数 (UOBF) が不足した場合、または flags に EEMCH_MYPROC を設定して自プロセス送信を行う場合にメモリ関連定義の pce_no オペランドで指定する処理キュー数が不足したとき

ee_mch_cmtsend 関数は EEMCHER_MEMORY_SHORT のリターン値でエラーリターンします。

注意事項

1. 設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で指定しておいてください。
2. メッセージ送信時にエラーが発生した場合、高速メッセージ送信関連定義の mch_send_retry_count オペランドに従い、再送信します。再送信がエラーとなった場合は、ERRTRNS にメッセージを引き継ぐために UDP 用送信バッファ（実行系から待機系に転送されたメッセージのときは UDP 用受信バッファ (UIBF)）を確保し、ERRTRNS を起動します。
3. メッセージ送信は、自プロセスあてを含め UDP 用送信バッファを使用します。

4. 自プロセスあてのサービス要求によるサービス実行と他サービス要求（サービス要求受信やタイマトランザクション起動要求など）によるサービス実行との順序性は保証されません。
5. この関数を発行したトランザクションでは、トランザクションの最適化は行われません。
6. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。

ee_mch_cmtsend_get【XTC 限定】

名称

一方送信メッセージ情報の取得

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_cmtsend_get(EEULONG *msg_cnt, EEMCHMSG *eemchmsg,
                      EELONG type, EELONG flags)

typedef struct ee_mch_msg_t{           /*メッセージ情報取得構造体          */
    EEULONG group_size;                /*送信先サービスグループ名長        */
    char group[32];                    /*送信先サービスグループ名          */
    EEULONG service_size;              /*送信先サービス名長                */
    char service[32];                  /*送信先サービス名                  */
    char otq_dct;                      /*出力キュー（OTQ）の自動閉塞有無    */
    char dummy1[3];                    /*予備                              */
    EEULONG send_sts;                  /*送信結果                          */
    char dummy2[4];                    /*予備                              */
    EEULONG send_size;                 /*送信メッセージサイズ              */
    char *msg;                         /*送信メッセージポインタ            */
    char dummy3[160];                  /*予備                              */
} EEMCHMSG;
```

機能

メッセージ送信に失敗（未送信も含む）したトランザクション同期の一方送信メッセージとトランザクション非同期の一方送信メッセージのメッセージ情報を取得します。リターン値が EE_OK（次のメッセージ情報なし）になるまで、この関数を連続して呼び出すことによって、同一トランザクションで一方送信メッセージの送信要求を行ったすべてのメッセージ情報を取得できます。リターン値が EE_OK の状態でこの関数を呼び出した場合は、最初からメッセージ情報を取得します。

一方送信メッセージのメッセージ情報は、ERRTRNR のタイミングで取得できます。

UAP で値を設定する引数

●msg_cnt

参照できるメッセージ情報の件数を格納する変数アドレスを設定します。

●eemchmsg

メッセージ情報を格納するための構造体ポインタのアドレスを設定します。

●type

参照したい一方送信メッセージの要求種別を設定します。

EEMCH_CMTSEND

ee_mch_cmtsend 関数

EEMCH_CMTSEND_SYNC

ee_mch_cmtsend_sync 関数

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●msg_cnt

参照できるメッセージ情報の件数が設定されます。0 の場合は、eemchmsg 構造体の内容は参照できません。

●eemchmsg

情報格納構造体に次の情報が設定されます。

- 送信先サービスグループ名長
送信先サービスグループ名の長さが返されます。
- 送信先サービスグループ名
送信先サービスグループ名が返されます。
- 送信先サービス名長
送信先サービス名の長さが返されます。
- 送信先サービス名
送信先サービス名が返されます。
- 出力キュー（OTQ）の自動閉塞有無
出力キューの自動閉塞有無が次のコードで返されます。

EEMCH_OTQ_NOBLOCKADE

出力キューの自動閉塞なし

EEMCH_OTQ_BLOCKADE_GROUP

サービスグループレベルの出力キューの自動閉塞あり

EEMCH_OTQ_BLOCKADE_SERVICE

サービスレベルの出力キューの自動閉塞あり

- 送信結果
送信結果が次のコードで返されます。

EEMCH_SEND_NORMAL

送信正常

EEMCH_SEND_ERR_PACKET

送信障害（パケットサイズ不正などの RPC 関連定義の不正）

EEMCH_SEND_ERR_RESOURCE

送信障害（リソース不足）

EEMCH_SEND_ERR_STATUS

送信障害（送信先受信不可状態）

EEMCH_SEND_ERR

送信障害（上記以外）

EEMCH_SEND_ERR_ACK_TIMEOUT

送達確認待ちタイムアウト

EEMCH_SEND_ERR_ACK_ERR

送達確認待ち受信障害

EEMCH_SEND_ERR_ACK_RESOURCE

送達確認エラーメッセージ受信（リソース不足）

EEMCH_SEND_ERR_ACK_CLSYNC

送達確認エラーメッセージ受信（CL 同期エラー）

EEMCH_SEND_ERR_ACK_HOLD

送達確認エラーメッセージ受信（送信先サービス閉塞），または自プロセスのサービス閉塞

EEMCH_SEND_ERR_ACK_GRP

送達確認エラーメッセージ受信（送信先サービスグループ名の不正）

EEMCH_SEND_ERR_ACK_SRV

送達確認エラーメッセージ受信（送信先サービス名の不正）

EEMCH_SEND_ERR_ACK_TYPE

送達確認エラーメッセージ受信（メッセージ送信形態不正またはオプションの不正）

EEMCH_SEND_ERR_ACK

送達確認エラーメッセージ受信（上記以外）

EEMCH_SEND_OTQ_BLOCKADE

出力キューの閉塞中

EEMCH_SEND_MYPROC_PUTPCE

処理キューの登録失敗

EEMCH_SEND_BEFORE

未送信

- 送信メッセージサイズ

送信メッセージのサイズが返されます。

- 送信メッセージポインタ
送信メッセージのポインタが返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました（次のメッセージ情報なし）。
EE_CONTINUE	正常に終了しました（次のメッセージ情報あり）。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMCHER_ARGUMENT	引数に設定した値が間違っています。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_MEMORY_SHORT	メモリが不足しました。

ee_mch_cmtsend_get 関数がエラーになる場合

メモリ関連定義の system_work_size オペランドで指定したシステムワーク領域が不足した場合

ee_mch_cmtsend_get 関数は EEMCHER_MEMORY_SHORT のリターン値でエラーリターンします。

ee_mch_cmtsend_sync 【XTC 限定】

名称

トランザクション同期の一方送信メッセージの送信

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_cmtsend_sync(char *group, char *service,
                        char *in, EEULONG *in_len,
                        EEULONG priority, EEULONG flags)
```

機能

トランザクションのコミットが確定したとき、トランザクション処理と同じスレッドから一方送信メッセージを UDP プロトコルで送信します。これによって、SPP に対してサービスを要求します。トランザクションがロールバックとなった場合、メッセージは破棄されます。

サービスを要求するときには、「サービスグループ名」および「サービス名」を引数に設定します。この名称に該当するサービス関数へサービスが要求されます。

この関数を使う場合、サーバ UAP があるノードの TP1/EE が稼働していることが前提です。

UAP で値を設定する引数

●group

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。

文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメタ長を設定します。1～EEMCH_MAX_MESSAGE_SIZE※の範囲の値を設定できます。EEMCH_MAX_MESSAGE_SIZE は、eemch.h で定義してあります。

注※

EEMCH_MAX_MESSAGE_SIZE の値は、将来拡張される場合があります。

●priority

メッセージ送信先で登録された処理キューの中で優先的に起動するかどうかを設定します。

EEMCH_HI

優先的に起動します。

EEMCH_LOW

優先的に起動しないで、登録した順序で起動します。

●flags

メッセージ送信形態とオプションを次に示す形式で設定します。

{EENOFLLAGS EEMCH_MULTI_CL EEMCH_MYPROC} [EEMCH_PERMANENCE]

EENOFLLAGS

ユニキャストで一方送信メッセージを送信する場合に設定します。

EEMCH_MULTI_CL

マルチキャストで CL サーバに一方送信メッセージを送信する場合に設定します。

EEMCH_MYPROC

自プロセスあてに一方送信メッセージを送信する場合に設定します。

EEMCH_PERMANENCE

メッセージを永続属性とする場合に設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下にありません。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_MEMORY_SHORT	メモリが不足しました。
EEMCHER_MESSAGE_TOO_BIG	in_len に設定した入力パラメタ長が、最大値を超えています。
EEMCHER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。
EEMCHER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。

リターン値	意味
EEMCHER_OTQ_SHUT_DOWN	出力キュー（OTQ）が閉塞中、または閉塞解除処理中です。
EEMCHER_PARAM_FLAGS	flags に設定した値が間違っています。
EEMCHER_PARAM_GROUP	group に設定した値が間違っています。
EEMCHER_PARAM_IN	in に設定した値が間違っています。
EEMCHER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EEMCHER_PARAM_PRIORITY	priority に設定した値が間違っています。
EEMCHER_PARAM_SERVICE	service に設定した値が間違っています。
EEMCHER_SYSERR	システムエラーが起きました。

ee_mch_cmtsend_sync 関数がエラーになる場合

ee_mch_cmtsend_sync 関数がエラーリターンする場合について説明します。

クラスタ連携関連定義の cluster_mode オペランドに Y を指定している場合

ee_mch_cmtsend_sync 関数は EEMCHER_CONDITION のリターン値でエラーリターンします。その他の理由については、ee_mch_cmtsend 関数の説明を参照してください。

注意事項

1. 設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で指定しておいてください。
2. メッセージ送信時にエラーが発生した場合、高速メッセージ送信関連定義の mch_send_retry_count オペランドに従い、再送信します。再送信がエラーとなった場合は、ERRTRNS*を起動してメッセージを破棄します。ただし、要求元のトランザクションがエラートランザクションの場合、ERRTRNS は起動されません。

注※

高速メッセージ送信関連定義の mch_send_err_rollback オペランドで Y を指定した場合は、ERRTRNR が起動されます。ただし、要求元のトランザクション種別が MN または TM 以外の場合、ERRTRNR は起動されません。

3. CL サーバでは使用できません。
4. メッセージ送信は、自プロセスあてを含め UDP 用送信バッファ（UOBF）を使用します。
5. 高速メッセージ送信関連定義の mch_send_err_rollback オペランドで Y を指定した場合は、自プロセスあて用の UDP 用送信バッファは 2 倍必要です。
6. この関数を発行したトランザクションでは、トランザクションの最適化は行われません。
7. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。

ee_mch_otqbkctl 【XTC 限定】

名称

出力キュー（OTQ）の閉塞制御

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_otqbkctl(char * group, char *service, EELONG mode,
                    EELONG flags)
```

機能

指定したサービスグループ下のサービス名に対応する出力キュー（OTQ）の閉塞/閉塞解除，またはサービスグループ下のすべてのサービスに対応する出力キューの閉塞/閉塞解除を行います。

UAP で値を設定する引数

●group

サービスグループ名を，31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

サービス名を，31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

また，すべてのサービスを対象にしたい場合，サービス名を設定しないでください（先頭を NULL 文字にしてください）。

サービス名を設定しない場合は，サービスグループレベルの出力キューの閉塞/閉塞解除，サービス名を設定した場合は，サービスレベルの出力キューの閉塞/閉塞解除となります。

●mode

閉塞か閉塞解除を設定します。

EEMCH_OTQBKSET

出力キューを閉塞します。

出力キューの状態がすでに自動閉塞中の場合，関数による閉塞中の状態に変わります。

EEMCH_OTQBKRST

出力キューの閉塞を解除します。

サービスグループレベルの場合，通信を行って送信先サービスグループの起動状態を確認します。

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMCHER_ARGUMENT	引数に設定した値が間違っています。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。
EEMCHER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。
EEMCHER_SERVICE_GROUP_DCT	出力キューを閉塞解除できる状態ではありません。
EEMCHER_SYSERR	システムエラーが起きました。
EEMCHER_TIMING	この関数を発行できる状態ではありません。出力キューの閉塞処理中、または出力キューの閉塞解除処理です。

注意事項

1. サービスグループレベルで出力キューが閉塞中（出力キューの自動閉塞中含む）の場合、サービスレベルの出力キューの閉塞解除はできません。EEMCHER_CONDITION のリターン値でエラーリターンします。
2. サービスグループレベルの出力キューの自動閉塞中にサービスグループレベルの出力キューの閉塞要求を行った場合、出力キューの閉塞監視を終了します。ただし、サービスレベルの出力キューの閉塞要求を行った場合は、出力キューの閉塞監視は続行します。
3. サービスグループレベルの出力キューの自動閉塞中にサービスグループレベルの出力キューの閉塞解除要求を行った場合、出力キューの閉塞監視を終了します。

ee_mch_otqinquire 【XTC 限定】

名称

出力キュー（OTQ）の情報の照会

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_otqinquire (char * group, char *service,
                      EEMCH_OTQINF *otq_inf, EELONG flags)

typedef struct ee_mch_otq_inf_t{ /*出力キュー（OTQ）の情報取得構造体 */
    EEULONG sts_code;           /*閉塞状態 */
    EEULONG request_num;        /*送信要求済み出力キューの通番 */
    EEULONG send_num;           /*送信済み出力キューの通番 */
    EEULONG unwrite_count;      /*未送信メッセージ数 */
    char dct_level;             /*閉塞レベル */
    char dummy[239];            /*予備 */
}EEMCH_OTQINF;
```

機能

指定したサービスグループ下のサービス名に対応する出力キュー（OTQ）の情報を照会します。照会対象のメッセージは、トランザクション非同期の一方送信メッセージです。

UAP で値を設定する引数

●group

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●otq_inf

照会した出力キューの情報を格納するための構造体ポインタのアドレスを設定します。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●otq_inf

照会した出力キューの情報格納構造体に次の情報が設定されます。

- 閉塞状態
出力キューの状態が次のコードで返されます。
EEMCH_OTQ_STATUS_NORMAL
正常
EEMCH_OTQ_STATUS_BLOCKADE_AUTO
自動閉塞中
EEMCH_OTQ_STATUS_BLOCKADE_API
関数による閉塞中
EEMCH_OTQ_STATUS_BLOCKADE_CMD
コマンドによる閉塞中
- 送信要求済み出力キューの通番
出力キューへ最後に送信要求した，トランザクション非同期の一方送信メッセージの出力キューの通番が返されます。
- 送信済み出力キューの通番
出力キューから最後に送信した，トランザクション非同期の一方送信メッセージの出力キューの通番が返されます。
- 未送信メッセージ数
未送信（送信失敗含む）のメッセージ数が返されます。
- 閉塞レベル
出力キューの閉塞レベルが返されます。
EEMCH_DCT_LEVEL_G
サービスグループレベル
EEMCH_DCT_LEVEL_S
サービスレベル
EEMCH_DCT_LEVEL_N
閉塞中でない

リターン値

リターン値	意味
EE_OK	正常に終了しました。

リターン値	意味
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMCHER_ARGUMENT	引数に設定した値が間違っています。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。
EEMCHER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。

ee_mch_otqskip 【XTC 限定】

名称

一方送信メッセージの送信スキップ

形式

ANSI C, C++の形式

```
#include <eemch.h>
int ee_mch_otqskip(char * group, char *service, EEULONG skip_flg,
                  EEULONG number, EEULONG flags)
```

機能

指定した通番までのトランザクション非同期の一方送信メッセージを送信済み状態として、出力キュー (OTQ) から破棄します。

指定したサービスグループ下のサービス名に対応する出力キューのメッセージを送信済み状態とします。一方送信メッセージの送信スキップ要求は、該当の出力キューが閉塞している場合だけ受け付けられます。

UAP で値を設定する引数

●group

サービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

サービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●skip_flg

次のフラグを設定します。

EEMCH_SKIP_ALL

出力キューに格納しているすべてのメッセージを送信済みにします。

EEMCH_SKIP_ONE

出力キューに格納している未送信（送信失敗含む）の 1 メッセージを送信済みにします。

EEMCH_SKIP_NUM

number で設定された出力キューの通番までのメッセージを送信済みにします。

●number

送信済みとする出力キューの通番を設定します。

設定した出力キューの通番と一致する出力キューの通番の未送信メッセージがある場合は、設定した通番まで送信をスキップします。

設定した出力キューの通番と一致する出力キューの通番の未送信メッセージがない場合は、送信をスキップしないで正常終了します。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMCHER_ARGUMENT	引数に設定した値が間違っています。
EEMCHER_CONDITION	発行条件が不正です。
EEMCHER_MEMORY_SHORT	メモリが不足しました。
EEMCHER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。
EEMCHER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。
EEMCHER_SYSERR	システムエラーが起きました。
EEMCHER_TIMING	この関数を発行できる状態ではありません。

注意事項

送信スキップ対象のメッセージがない場合は、正常扱いとなります。

メモリ管理 (ee_mem_～)

メモリ上のワークセグメントの割り当ておよび返還ならびに大量処理用メモリ管理機能の関数について説明します。メモリ管理の関数を次に示します。

- ee_mem_getwkseg – ワークセグメントの割り当て
- ee_mem_mdpstart 【XTC 限定】 – 大量処理用メモリ管理機能の使用開始
- ee_mem_rlswkseg – ワークセグメントの返還

ee_mem_getwkseg

名称

ワークセグメントの割り当て

形式

ANSI C, C++の形式

```
#include <eemem.h>
int ee_mem_getwkseg(EEULONG seg_type, EEULONG seg_sz,
                    void **seg_pt, EELONG flags)
```

機能

ユーザが使用するワークセグメントを割り当てます。

UAP で値を設定する引数

●seg_type

ワークセグメントの利用形態によって、次のように設定します。

EE_SEGTYPE_UAPWORK

TASK 属性セグメント（ワークセグメントのうち、該当するトランザクションでだけ使用できるセグメント）

EE_SEGTYPE_UAPFREE

FREE 属性セグメント（ワークセグメントのうち、複数のトランザクション間で使用できるセグメント）

●seg_sz

割り当てるワークセグメントの長さを設定します（単位：バイト）。1～16777216 の範囲で指定してください。

●seg_pt

ワークセグメントのアドレスを格納する領域のアドレスを設定します。

●flags

EENOFLAGS を設定します。

TP1/EE から値が返される引数

●seg_pt

ワークセグメントのアドレスが返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMEMER_ARGUMENT	引数に設定した値が間違っています。
EEMEMER_CONDITION	関数の呼び出し条件が不正です。
EEMEMER_NOSEGMENT	セグメントが不足しました。

注意事項

1. 関数が正常に終了したときだけ、ワークセグメントを使用できます。
2. ワークセグメントの割り当て時に領域破壊を検知した場合は、プロセスダウンします。

ee_mem_mdpstart 【XTC 限定】

名称

大量処理用メモリ管理機能の使用開始

形式

ANSI C, C++の形式

```
#include <eemem.h>
int ee_mem_mdpstart(EELONG flags)
```

機能

大量処理用メモリ管理機能を有効にします。

UAP で値を設定する引数

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMEMER_ARGUMENT	引数に設定した値が間違っています。
EEMEMER_CONDITION	関数の呼び出し条件が不正です。

ee_mem_mdpstart 関数がエラーになる場合

ee_mem_mdpstart 関数がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合

ee_mem_mdpstart 関数は EEMEMER_CONDITION のリターン値でエラーリターンします。

メモリ関連定義の memory_mdpsys_area_size オペランドまたは memory_mdpusr_area_size オペランドのどちらも指定していない場合

ee_mem_mdpstart 関数は EEMEMER_CONDITION のリターン値でエラーリターンします。

注意事項

1. UAP 終了後のトランザクションが終了するまで、大量処理用メモリ管理機能は継続します。
2. 大量処理用メモリ管理機能を使う UAP 内で、大量処理用ユーザ領域ではなくユーザワーク領域から FREE 属性のセグメントを取得したい場合、この関数の前に `ee_mem_getwkseg` 関数を発行する必要があります。

ee_mem_rlswkseg

名称

ワークセグメントの返還

形式

ANSI C, C++の形式

```
#include <eemem.h>
int ee_mem_rlswkseg(void *seg_pt, EELONG flags)
```

機能

使用済みのワークセグメントを返還します。

UAP で値を設定する引数

●seg_pt

返還するワークセグメントのアドレスを設定します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EEMEMER_ARGUMENT	引数に設定した値が間違っています。
EEMEMER_CONDITION	関数の呼び出し条件が不正です。
EEMEMER_INVALID	無効なセグメントを設定しています。

注意事項

ワークセグメントの返還時に領域破壊を検知した場合は、プロセスダウンします。

リモートプロシジャコール (ee_rpc_～)

クライアント／サーバ形態の通信をするときに使う，TP1/EE のリモートプロシジャコールの関数について説明します。リモートプロシジャコールの関数を次に示します。

- ee_rpc_call – 遠隔サービスの要求
- ee_rpc_call_to – 通信先を指定した遠隔サービスの要求
- ee_rpc_cmtsend – トランザクションと同期して送信する非応答型 RPC 要求
- ee_rpc_cmtsend_to – 通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求
- ee_rpc_discard_further_replies – すべての非同期応答型 RPC の応答受信拒否
- ee_rpc_discard_specific_reply – 特定の非同期応答型 RPC の応答受信拒否
- ee_rpc_get_error_descriptor – エラーが発生した非同期応答型 RPC の識別子取得
- ee_rpc_get_watch_time – サービスの応答待ち時間の参照
- ee_rpc_poll_any_replies – 非同期応答型 RPC の応答受信
- ee_rpc_reply_send – 抑止していた RPC 応答メッセージの送信
- ee_rpc_reply_suspend – RPC 応答メッセージ送信の抑止
- ee_rpc_set_watch_time – サービスの応答待ち時間の更新

ee_rpc_call

名称

遠隔サービスの要求

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_call(char *group, char *service, char *in,
                EEULONG *in_len, char *out, EEULONG *out_len,
                EELONG flags)
```

機能

SPP にサービスを要求します。この関数を使うときは、要求するサービスがどのノードにあるかを意識する必要はありません。

サービスを要求するときには、「サービスグループ名」と「サービス名」を引数に設定します。この名称に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

この関数を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定する引数

●group

SPP のサービスグループ名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメータ長を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の IMA サイズ以下の値を設定してください。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●out

サービス関数から返ってくる応答を格納する領域を設定します。out に設定した領域に、サービス関数からの応答が設定されます。

●out_len

サービスの応答の長さを設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の OMA サイズ以下の値を設定してください。

非応答型 RPC の場合も、サービスの応答の長さを設定した領域のアドレスを指定する必要があります。この場合、サービスの応答の長さには、0 を設定します。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●flags

RPC の形態とオプションを次に示す形式で設定します。

`{EENOFLLAGS | EERPC_NOWAIT | EERPC_NOREPLY} [ | EERPC_TPNOTRAN] [ | EERPC_CALL_MYPROC] [ | EERPC_NOWAIT_OUTLEN] [ | EERPC_UDP]`

EENOFLLAGS

同期応答型 RPC

EERPC_NOWAIT

非同期応答型 RPC

EERPC_NOREPLY

非応答型 RPC

EERPC_TPNOTRAN

トランザクション処理からのサービス要求で、要求先の処理をトランザクションにしない（要求先の処理を要求元と同一のグローバルトランザクションにしない）場合に設定します。

この値は RPC の形態との論理和で設定してください。

EERPC_CALL_MYPROC

group に自サービスグループ名を設定したときに自プロセス送信（通信レス）を行う場合に指定します。EERPC_CALL_MYPROC を指定する場合は、必ず EERPC_TPNOTRAN と論理和で設定してください。

EERPC_UDP と論理和で設定することはできません。

EERPC_NOWAIT_OUTLEN

非同期応答型 RPC で、サービス関数が設定した応答の長さを参照したい場合に設定します。

EERPC_NOWAIT_OUTLEN を設定した場合、ee_rpc_poll_any_replies 関数の正常終了時、ee_rpc_call 関数で設定した out と out_len に、サービス関数が設定した応答と応答の長さがそれぞれ設定されます。EERPC_NOWAIT_OUTLEN を設定しない場合、out だけ設定されます。

EERPC_NOWAIT_OUTLEN は EERPC_NOWAIT と論理和で指定してください。

EERPC_NOWAIT を指定していない場合、EERPC_NOWAIT_OUTLEN は無視されます。

EERPC_UDP 【XTC 限定】

サーバとの通信をコネクションレス（UDP プロトコル）で行う場合に設定します。

この値は RPC の形態との論理和で設定してください。EERPC_UDP を設定する場合は必ず EERPC_TPNOTRAN との論理和で設定してください。

サーバ UAP から値が返される引数

●out

サービス関数が設定した応答が返されます。

●out_len

サービス関数が設定した応答の長さが返されます（単位：バイト）。

サーバ UAP から返される値

サービス関数の処理が終了して応答が返ってくると、次の値を参照できます。

- サービス関数の応答（out）
- サービス関数の応答の長さ（out_len）

out または out_len を参照できる場合を次に示します。

- 同期応答型 RPC の場合
ee_rpc_call 関数がリターンしたあと、out および out_len を参照できます。
- 非同期応答型 RPC の場合
ee_rpc_poll_any_replies 関数が該当する応答を受け取ってリターンしたあと、out を参照できます。
EERPC_NOWAIT_OUTLEN を指定していない場合、out_len は参照できません。

- 非応答型 RPC の場合

out および out_len は参照できません。

ee_rpc_call 関数、または ee_rpc_poll_any_replies 関数がエラーリターンした場合は、out および out_len は参照できません。

返ってきた応答が、クライアント UAP で用意した応答の領域 (out) よりも大きい場合は、EERPCER_REPLY_TOO_BIG でエラーリターンします。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK または正の整数	正常に終了しました。非同期応答型 RPC の場合は、正の整数は記述子です。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下にあります。
EERPCER_AGAIN	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられません。
EERPCER_CONDITION	rpc_tcp_communication_use オペランドに N を指定しているため、TCP/IP を使用して RPC 要求できません。
	終了トランザクション (ME) からトランザクショナル RPC でサービスを要求しています。
	初期化トランザクション (MI)、または終了トランザクション (ME) から自プロセス通信 (通信レス) でサービスを要求しています。
EERPCER_INVALID_REPLY	サービス関数が OpenTP1 に返した応答の長さが、1 ~ EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
EERPCER_LIMITS_OVER	非同期応答型 RPC (flags に EERPC_NOWAIT を設定) の場合は、rpc_nowait_cnt オペランドで指定した同時要求最大数をオーバーしました。
EERPCER_MEMORY_SHORT	メモリが不足しました。要求先が TP1/EE の場合は、out_len の設定値が OMA サイズを超過しているおそれがあります。
EERPCER_MESSAGE_TOO_BIG	in_len に設定した入力パラメータ長が、最大値を超えています。
EERPCER_NET_DOWN	ネットワークに障害が起きました。
EERPCER_NO_BUFS_AT_SERVER	設定したサービスで、メモリが不足しました。
	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を設定) の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランドの指定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。次に示す原因が考えられます。

リターン値	意味
EERPCER_NO_BUFS_AT_SERVER	- UAP リターン後の同期点処理でトランザクション決着できなかった、かつ、RPC 関連定義の <code>rpc_replymsg_save</code> オペランドに N を指定した - UAP リターン後の同期点処理以降にプロセスダウンした
EERPCER_NO_BUFS_RB	メモリが不足しました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_NO_SERVICE_SOCKET	ネットワークの障害などによってサービス要求できるコネクションの取得に失敗しました。
EERPCER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。 group に設定したサービスグループ名が、サービス要求先プロセスのサービスグループ名と異なります。次の条件をすべて満たす場合に発生します。 - サービスグループ情報関連定義の <code>eesvgdef</code> 定義コマンドで登録したサービスグループを指定している - RPC の形態として同期応答型 RPC を指定している - サービスの要求先が TP1/EE で、かつ要求先の転送機能が無効（RPC 関連定義の <code>rpc_loadbalance</code> オペランド、および <code>rpc_transfer_othersvg</code> オペランドに N を指定）になっている
EERPCER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。
EERPCER_OLTF_INITIALIZING	サービスを要求されたノードにある TP1/Server Base または TP1/EE は、開始処理中です。
EERPCER_OLTF_NOT_UP	service に設定したサービスがあるノードの TP1/Server Base または TP1/EE が稼働していません。異常終了、停止中、終了処理中、および通信障害が起こったことが考えられます。
EERPCER_PARAM_FLAGS	flags に設定した値が間違っています。または、値の組み合わせが不正です。
EERPCER_PARAM_GROUP	group に設定した値が間違っています。
EERPCER_PARAM_IN	in に設定した値が間違っています。
EERPCER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EERPCER_PARAM_OUT	out に設定した値が間違っています。
EERPCER_PARAM_OUT_LEN	out_len に設定した値が間違っています。
EERPCER_PARAM_SERVICE	service に設定した値が間違っています。
EERPCER_REPLY_TOO_BIG	返ってきた応答が、クライアント UAP で用意した領域に収まりません。
EERPCER_REPLY_TOO_BIG_RB	返ってきた応答が、クライアント UAP で用意した領域に収まりません。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SECCHK	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
EERPCER_SERVER_BUSY	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。

リターン値	意味
EERPCER_SERVICE_CLOSED	service に設定したサービス名があるサービスグループは、閉塞しています。
EERPCER_SERVICE_NOT_UP	service に設定したサービスの UAP プロセスが、稼働していません。
	サービス要求の応答待ち時間に 0 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。
EERPCER_SERVICE_TERMINATED	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
EERPCER_SERVICE_TERMINATING	service に設定したサービスは、終了処理中です。
EERPCER_SYSERR	システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER	設定したサービスで、システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER_RB	設定したサービスで、システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SYSERR_RB	システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_TESTMODE	オンラインテストを使っている環境で、テストモードの UAP からテストモードではない SPP へサービスを要求しています。または、テストモードではない UAP からテストモードの SPP へサービスを要求しています。
EERPCER_TIMED_OUT	ee_rpc_call 関数の処理が時間切れ（タイムアウト）になりました。
	サービスを要求された SPP が、処理を完了する前に異常終了しました。
	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を設定) の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の rpc_reply_tplmode_down オペランドの指定値が J の場合、サーバ側のトランザクション決着結果がロールバック以外になりました。
EERPCER_TRNCHK	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある TP1/Server Base または TP1/EE のバージョンが、クライアントの TP1/Server Base または TP1/EE のバージョンよりも古いため、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。 このリターン値は、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
EERPCER_TRNCHK_EXTEND	同時に起動できるトランザクションブランチの数を越えたため、トランザクションブランチを開始できません。
	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を越えたため、トランザクションブランチを開始できません。
	リソースマネージャでエラーが発生したため、トランザクションブランチを開始できません。
	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を設定) の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の

リターン値	意味
EERPCER_TRNCHK_EXTEND	rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランドの指定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を設定) の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の rpc_reply_tplmode_down オペランドの指定値が J の場合、サーバ側のトランザクションがロールバックしました。

ee_rpc_call 関数がエラーになる場合

ee_rpc_call 関数がエラーリターンする場合について説明します。

サービス要求先が稼働していない場合、または送信できるコネクションがない場合

ee_rpc_call 関数は EERPCER_NO_SERVICE_SOCKET のリターン値でエラーリターンします。

サービス要求先コネクション確立後、サービス要求時にネットワーク障害を検知した場合

ee_rpc_call 関数は EERPCER_NET_DOWN のリターン値でエラーリターンします。

ネームサービスを使用しない場合で、サービスグループ情報関連定義の eesvgdef 定義コマンドにサーバ UAP のサービスグループ名が設定されていないとき

ネームサービスを使用しない場合は、サービスグループ情報関連定義の eesvgdef 定義コマンドの指定値を基にサービス要求先を取得します。eesvgdef 定義コマンドにサーバ UAP のサービスグループ名が設定されていない場合、ee_rpc_call 関数は EERPCER_NO_SUCH_SERVICE_GROUP のリターン値でエラーリターンします。

ネームサービスを使用する場合で、該当するサービスグループ名が見つからなかったとき

ネームサービスを使用する場合は、RPC 関連定義の rpc_destination_mode オペランドの指定値に従ってサービス要求先を取得します。該当するサービスグループ名が見つからなかった場合、ee_rpc_call 関数は EERPCER_NO_SUCH_SERVICE_GROUP のリターン値でエラーリターンします。

サービス要求時にメッセージ格納領域 (TCP/IP バッファ) 不足がリトライ回数を超えて連続で発生した場合

ee_rpc_call 関数は EERPCER_AGAIN のリターン値でエラーリターンします。サービス要求のリトライ回数は、RPC 関連定義の rpc_router_retry_count オペランドで指定します。サービス要求のリトライ間隔は、RPC 関連定義の rpc_router_retry_interval オペランドで指定します。そのほか、RPC 関連定義の ipc_sendbuf_size オペランドやサービス要求先の TCP/IP バッファの定義 (RPC 関連定義, TP1/Server Base のユーザサービス定義またはシステム共通定義の ipc_recvbuf_size オペランド) などを見直してください。

注意事項

1. 引数に設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で設定しておいてください。

2. 次のトランザクションの場合は、自 TP1/EE のサービスグループ名を指定した RPC はできません。
 - 初期化トランザクション (MI)
 - 終了トランザクション (ME)
 - 共有リソース初期化トランザクション (JI)
 - システム開始時のユーザ初期化トランザクション (UT)
 - 実行系連絡トランザクション (UI) (ただし、flags に EERPC_CALL_MYPROC を指定している場合だけは、自 TP1/EE のサービスグループ名を指定した RPC ができます)
3. 終了トランザクション (ME) では、トランザクショナル RPC はできません。
4. 初期化トランザクション (MI) からのトランザクショナル RPC は実行しないでください。通常起動 (再起動ではない起動) で TP1/EE を起動した場合で、初期化トランザクションからのトランザクショナル RPC の処理中にプロセスダウンしたとき、TP1/EE は再び通常起動で起動されます。このとき、プロセスダウン前の仕掛けり中トランザクションは、回復処理の対象外となります。回復処理の対象外となったトランザクションは、サービス要求先でコマンドを実行することによって強制的に決着させることしかできなくなる場合があります。
5. 非同期応答型 RPC を連続して使用したあと応答を受け取る場合は、非同期応答型 RPC のサービスを要求するときに、それぞれ異なる応答格納領域 (out) および応答の長さを格納する領域 (out_len) を設定してください。同じ領域を設定すると、あとから来た応答に上書きされてしまい、応答を正しく受け取れなくなります。
6. **【TP1/FSP 限定】** 次のリターン値の場合、出力メッセージ (OJ) は取得されません。
EECOMER_ENVIRON, EECOMER_CNDUOC, EECOMER_CNDBPP,
EERPCER_PARAM_GROUP, EERPCER_PARAM_SERVICE, EERPCER_PARAM_IN,
EERPCER_PARAM_IN_LEN, EERPCER_PARAM_OUT, EERPCER_PARAM_OUT_LEN,
EERPCER_PARAM_FLAGS, EERPCER_MESSAGE_TOO_BIG
7. 非同期応答型 RPC の応答格納領域 (out) および応答の長さを格納する領域 (out_len) は、次に示す方法で非同期応答型 RPC を終了するまで解放しないでください。
 - ee_rpc_poll_any_replies 関数で応答を受け取る
 - ee_rpc_discard_further_replies 関数または ee_rpc_discard_specific_reply 関数で応答の受信を拒否する
 - トランザクショナル RPC の場合、同期点処理の関数でコミットまたはロールバックする

ee_rpc_call_to

名称

通信先を指定した遠隔サービスの要求

形式

ANSI C, C++の形式

```
#include <eerrpc.h>
int ee_rpc_call_to(struct EERPC_BINDING_TBL *direction,
                  char *group, char *service,
                  char *in, EEULONG *in_len,
                  char *out, EEULONG *out_len, EELONG flags)
```

機能

特定の SPP にサービスを要求します。サービスを要求するには、ホスト名またはノード識別子を指定した EERPC_BINDING_TBL 構造体に加え、「サービスグループ名」と「サービス名」を引数に設定します。EERPC_BINDING_TBL 構造体に指定したホスト名またはノード識別子は、サービス要求先を特定する検索のキーとして使用します。この設定に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

この関数を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義 service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定する引数

●direction

サービス要求先を特定する検索キーを格納する EERPC_BINDING_TBL 構造体のアドレスを指定します。検索のキーはホスト名またはノード識別子のどちらかです。

EERPC_BINDING_TBL 構造体の形式を次に示します。

```
struct EERPC_BINDING_TBL {
    char *nid ;                /*ノード識別子格納アドレス*/
    char *hostnm ;            /*ホスト名格納アドレス */
    unsigned short portno ;    /*ポート番号 */
    short filler1 ;           /*予備 */
    EELONG flags ;            /*属性 */
```



```
EELONG filler2[4] ;      /*予備          */  
};
```

- nid

ノード識別子を検索キーにする場合、サービス要求先のノード識別子を格納した領域のアドレスを指定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えませんが、

ノード識別子は、OpenTP1 システム共通定義の node_id オペランドに指定した名称で、かつサービス要求先ノードのホスト名がグローバルドメイン内にあることが前提です。相手先が TP1/EE の場合も親 TP1/Server Base のノード識別子を指定してください。

ノード識別子を検索のキーにしない場合には、nid にアドレス 0 を指定してください。

EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定した場合は、nid にアドレス 0 を指定してください。

- hostnm

ホスト名を検索キーにする場合、サービス要求先ノードのホスト名を格納した領域のアドレスを指定します。EERPC_BINDING_TBL 構造体の flags に EENOFLLAGS 以外を指定した場合は、hostnm を必ず指定してください。

指定できるホスト名は 1～255 文字の文字列です。文字列に使用できる文字は、英数字、ピリオド、およびハイフンです（ただし、IP アドレス形式は除く）。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えませんが、指定するホスト名は、/etc/hosts ファイルまたは DNS など、IP アドレスとのマッピングができる名称です。

なお、サービス要求先ノードのホスト名はグローバルドメインに指定されていても、指定されていなくてもかまいません。

ホスト名を検索キーにしない場合には、hostnm にアドレス 0 を指定してください。

- portno

ホスト名を検索キーとする場合のポート番号（1～65535）を指定します。

EERPC_BINDING_TBL 構造体の flags に EERPC_NAMPORT を指定した場合は、サービス要求先ノードのネームサービスのポート番号（TP1/Server Base のシステム共通定義の name_port オペランドに指定した値）を指定します。

EERPC_BINDING_TBL 構造体の flags に EENOFLLAGS を指定した場合は、サービス要求元のネームサービスのポート番号と、サービス要求先のネームサービスのポート番号が同じであることが前提となります。

EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定した場合は、サービス要求先が TP1/Server Base の SPP のときは、要求先のスケジュールサービスのポート番号を指定します。サービス要求先が TP1/EE のときは、要求先の自サービスグループ定義の mysvgdef オペランドで指定したポート番号を指定します。

EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定し、portno に 0 を指定した場合は、送信先ポート番号の省略値として、最初に定義した自サービスグループ情報の mysvgdef 定義コマンドに指定したポート番号の値を仮定します。最初に定義した mysvgdef 定義コマンドのポート番号に 0 を指定している場合は、portno に 0 を指定できません。

ノード識別子を検索のキーとする場合は、portno に指定した値は無視されます。

- filler1
0 を指定します。

- flags

EERPC_NAMPORT

portno にネームサービスのポート番号を指定した場合に指定してください。

EERPC_SCDPORT

サービス要求先 (TP1/Server Base) のスケジュールサービスのポート、またはサービス要求先 (TP1/EE) の mysvgdef 定義コマンドで指定したポートに対してサービスを要求する場合に指定してください。

EENOFLLAGS

上記以外の場合に指定してください。

- filler2
すべて 0 を指定します。

●group

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメタ長を設定します (単位: バイト)。1 ~ EERPC_MAX_MESSAGE_SIZE_EX[※] の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.c.h で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の IMA サイズ以下の値を設定してください。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.c.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●out

サービス関数から返ってくる応答を格納する領域を設定します。out で指す領域に、サービス関数からの応答が設定されます。

●out_len

サービスの応答の長さを設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX[※]の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

非応答型 RPC の場合も、サービスの応答の長さを設定した領域のアドレスを指定する必要があります。この場合、サービスの応答の長さには、0 を設定します。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●flags

RPC の形態とオプションを次に示す形式で設定します。

`{EENOFLLAGS | EERPC_NOWAIT | EERPC_NOREPLY} [ | EERPC_TPNOTRAN] [ | EERPC_NOWAIT_OUTLEN]`

EENOFLLAGS

同期応答型 RPC

EERPC_NOWAIT

非同期応答型 RPC

EERPC_NOREPLY

非応答型 RPC

EERPC_TPNOTRAN

トランザクション処理からのサービス要求で、要求先の処理をトランザクションにしない（要求先の処理を要求元と同一のグローバルトランザクションにしない）場合に設定します。

この値は RPC の形態との論理和で設定してください。

EERPC_NOWAIT_OUTLEN

非同期応答型 RPC で、サービス関数が設定した応答の長さを参照したい場合に設定します。

EERPC_NOWAIT_OUTLEN を設定した場合、ee_rpc_poll_any_replies 関数の正常終了時、ee_rpc_call_to 関数で設定した out と out_len に、サービス関数が設定した応答と応答の長さがそれぞれ設定されます。EERPC_NOWAIT_OUTLEN を設定しない場合、out だけ設定されます。

EERPC_NOWAIT_OUTLEN は EERPC_NOWAIT と論理和で指定してください。

EERPC_NOWAIT を指定していない場合、EERPC_NOWAIT_OUTLEN は無視されます。

サーバ UAP から値が返される引数

●out

サービス関数が設定した応答が返されます。

●out_len

サービス関数が設定した応答の長さが返されます。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK または正の整数	正常に終了しました。非同期応答型 RPC の場合は、正の整数は記述子です。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EERPCER_AGAIN	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられませんでした。
EERPCER_ARGUMENT	引数に設定した値が間違っています。 EERPC_BINDING_TBL 構造体の hostnm に指定されたホスト名が、/etc/hosts ファイルまたは DNS などで、IP アドレスとのマッピングができません。
EERPCER_CONDITION	関数の呼び出し条件が不正です。 RPC 関連定義の name_use オペランドに Y が設定されていません。
EERPCER_INVALID_REPLY	サービス関数が OpenTP1 に返した応答の長さが、1～EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
EERPCER_LIMITS_OVER	非同期応答型 RPC (flags に EERPC_NOWAIT 指定) の場合は、rpc_nowait_cnt オペランドで指定した同時要求最大数をオーバーしました。
EERPCER_MEMORY_SHORT	メモリが不足しました。要求先が TP1/EE の場合は、out_len の設定値が OMA サイズを超過しているおそれがあります。
EERPCER_MESSAGE_TOO_BIG	in_len に設定した入力パラメタ長が、最大値を超えています。
EERPCER_NET_DOWN	ネットワークに障害が起きました。
EERPCER_NO_BUFS_AT_SERVER	設定したサービスで、メモリが不足しました。 非トランザクショナル RPC (EERPC_TPNOTRAN 指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tp1mode オペランドまたは rpc_reply_tp1mode_down オペランド設定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。次に示す原因が考えられます。 - UAP リターン後の同期点処理でトランザクション決着できない状態で、かつ rpc_replymsg_save オペランド設定値が N - UAP リターン後の同期点処理以降にプロセスダウン
EERPCER_NO_BUFS_RB	メモリが不足しました。このリターン値が返った場合は、トランザクションブランチをコミットできません。

リターン値	意味
EERPCER_NO_SERVICE_SOCKET	ネットワークの障害などによってサービス要求できるコネクションの取得に失敗しました。
EERPCER_NO_SUCH_SERVICE	service に設定したサービス名は、定義されていません。 サービスの要求先が TP1/EE であり、かつ要求先の転送機能が無効 (rpc_loadbalance オペランド設定値が N、かつ rpc_transfer_othersvg オペランド設定値が N) の場合は、サービスグループ名が不正のおそれがあります。
EERPCER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。 EERPC_BINDING_TBL 構造体の nid に指定されたノード識別子が、グローバルドメイン (システム共通定義の all_node オペランドで指定したノード名の集合) 内にありません。
EERPCER_OLTF_INITIALIZING	サービスを要求されたノードにある TP1/Server Base または TP1/EE は、開始処理中です。
EERPCER_OLTF_NOT_UP	service に設定したサービスがあるノードの TP1/Server Base または TP1/EE が稼働していません。異常終了、停止中、終了処理中、および通信障害が起こったことが考えられます。
EERPCER_PARAM_FLAGS	flags に設定した値が間違っています。
EERPCER_PARAM_GROUP	group に設定した値が間違っています。
EERPCER_PARAM_IN	in に設定した値が間違っています。
EERPCER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EERPCER_PARAM_OUT	out に設定した値が間違っています。
EERPCER_PARAM_OUT_LEN	out_len に設定した値が間違っています。
EERPCER_PARAM_SERVICE	service に設定した値が間違っています。
EERPCER_REPLY_TOO_BIG	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。
EERPCER_REPLY_TOO_BIG_RB	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。 このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SECCHK	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
EERPCER_SERVER_BUSY	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。
EERPCER_SERVICE_CLOSED	service に設定したサービス名があるサービスグループは、閉塞しています。
EERPCER_SERVICE_NOT_UP	service に設定したサービスの UAP プロセスが、稼働していません。 サービス要求の応答待ち時間に 0 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。

リターン値	意味
EERPCER_SERVICE_TERMINATED	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
EERPCER_SERVICE_TERMINATING	service に設定したサービスは、終了処理中です。
EERPCER_SYSERR	システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER	設定したサービスで、システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER_RB	設定したサービスで、システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SYSERR_RB	システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_TESTMODE	オンラインテストを使っている環境で、テストモードの UAP からテストモードでない SPP ヘサービスを要求しています。または、テストモードでない UAP からテストモードの SPP ヘサービスを要求しています。
EERPCER_TIMED_OUT	ee_rpc_call_to 関数の処理が時間切れ（タイムアウト）になりました。
	サービスを要求された SPP が、処理を完了する前に異常終了しました。
	非トランザクショナル RPC (flags に EERPC_TPNOTTRAN を指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランド設定値が J の場合、サーバ側のトランザクション決着結果がロールバック以外です。
EERPCER_TRNCHK	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある TP1/Server Base または TP1/EE のバージョンが、クライアントの TP1/Server Base または TP1/EE のバージョンよりも古い場合、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。 このリターン値は、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
EERPCER_TRNCHK_EXTEND	同時に起動できるトランザクションブランチの数を越えたため、トランザクションブランチを開始できません。
	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を越えたため、トランザクションブランチを開始できません。
	EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定し、非トランザクション属性（ユーザサービス定義の atomic_update オペランドに 'N' を指定、またはシステム共通定義の jnl_fileless_option オペランドに 'Y' を指定）の TP1/Server Base ユーザサーバに対し、ee_rpc_call_to 関数の flags に EERPC_TPNOTTRAN との論理和を指定しないでサービスを要求しました。
	リソースマネージャでエラーが発生したため、トランザクションブランチを開始できません。

リターン値	意味
EERPCER_TRNCHK_EXTEND	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランド設定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	非トランザクショナル RPC (flags に EERPC_TPNOTRAN を指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランド設定値が J の場合、サーバ側のトランザクションがロールバックしました。

注意事項

1. EERPC_BINDING_TBL 構造体にホスト名とノード識別子を同時に指定した場合、ホスト名の指定が有効になり、ノード識別子の指定は無視されます。
2. EERPC_BINDING_TBL 構造体にホスト名、ノード識別子ともに 0 を指定した場合、ee_rpc_call 関数と同様に動作します。
3. EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定し、ソケット受信型（ユーザーサービス定義の receive_from オペランドに socket を指定）の TP1/Server Base ユーザサーバへサービスを要求した場合、ee_rpc_call_to 関数は EERPCER_SERVICE_NOT_UP でエラーリターンします。
4. EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定して、ee_rpc_call_to 関数を呼び出した場合、rpc_retry オペランドは無効になります。
5. RPC 関連定義の name_use オペランドに N を設定して、ee_rpc_call_to 関数を呼び出した場合、EERPCER_CONDITION でエラーリターンします。
6. リモート API 機能を使用した RPC はできません。
7. **【TP1/FSP 限定】** 次のリターン値の場合、出力メッセージ (OJ) は取得されません。
EECOMER_ENVIRON, EECOMER_CNDUOC, EECOMER_CNDBPP,
EERPCER_ARGUMENT, EERPCER_PARAM_GROUP, EERPCER_PARAM_SERVICE,
EERPCER_PARAM_IN, EERPCER_PARAM_IN_LEN, EERPCER_PARAM_OUT,
EERPCER_PARAM_OUT_LEN, EERPCER_PARAM_FLAGS, EERPCER_MESSAGE_TOO_BIG
8. その他の注意事項は、ee_rpc_call 関数の注意事項を参照してください。

ee_rpc_cmtsend

名称

トランザクションと同期して送信する非応答型 RPC 要求

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_cmtsend(char *group, char *service,
                  char *in, EEULONG *in_len,
                  EELONG flags)
```

機能

SPP にサービスを要求します。この関数を使うときは、要求するサービスがどのノードにあるかを意識する必要はありません。

サービスを要求するときには、「サービスグループ名」と「サービス名」を引数に設定します。この名称に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

この関数を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定する引数

●group

SPP のサービスグループ名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメタ長を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerpch で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の IMA サイズ以下の値を設定してください。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerpch で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●flags

次のフラグを設定します。

EERPC_CMTSEND_MYPROC

ネームサービスを使用している場合、group に自サービスグループ名を設定したときに明示的に自プロセス送信（通信レス）をします。

EENOFLLAGS

上記以外

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下にありません。
EERPCER_CONDITION	関数の呼び出し条件が不正です。 rpc_tcp_communication_use オペランドに N を指定しているため、TCP/IP を使用して RPC 要求できません。
EERPCER_MEMORY_SHORT	メモリが不足しました。
EERPCER_MESSAGE_TOO_BIG	in_len に設定した入力パラメタ長が、最大値を超えています。
EERPCER_NO_SUCH_SERVICE	自プロセスの TP1/EE へのサービス要求時、service に設定したサービス名は、定義されていません。
EERPCER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。
EERPCER_PARAM_FLAGS	flags に設定した値が間違っています。
EERPCER_PARAM_GROUP	group に設定した値が間違っています。

リターン値	意味
EERPCER_PARAM_IN	in に設定した値が間違っています。
EERPCER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EERPCER_PARAM_SERVICE	service に設定した値が間違っています。
EERPCER_SYSERR	システムエラーが起きました。

ee_rpc_cmtsend 関数がエラーになる場合

ee_rpc_cmtsend 関数がエラーリターンする場合について説明します。

初期化トランザクション (MI), 終了トランザクション (ME), 共有リソース初期化トランザクション (JI), またはユーザ初期化トランザクション (UT) から RPC 要求された場合

ee_rpc_cmtsend 関数は EERPCER_CONDITION のリターン値でエラーリターンします。

すでに ee_trn_rollback_mark 関数を呼び出している場合

ee_rpc_cmtsend 関数は EERPCER_CONDITION のリターン値でエラーリターンします。

ネームサービスを使用しない場合, または, ネームサービスを使用して自プロセス送信をする場合で, サービスグループ情報関連定義の mysvgdef 定義コマンドまたは eesvgdef 定義コマンドにサーバ UAP のサービスグループ名が設定されていないとき

ネームサービスを使用しない場合, または, ネームサービスを使用して自プロセス送信をする (flags に EERPC_CMTSEND_MYPROC を設定) 場合は, サービスグループ情報関連定義の mysvgdef 定義コマンドまたは eesvgdef 定義コマンドの指定値を基にサービス要求先を取得します。mysvgdef 定義コマンドまたは eesvgdef 定義コマンドにサーバ UAP のサービスグループ名が設定されていない場合, ee_rpc_cmtsend 関数は EERPCER_NO_SUCH_SERVICE_GROUP のリターン値でエラーリターンします。

ネームサービスを使用する場合で, 該当するサービスグループ名が見つからなかったとき

ネームサービスを使用する場合は, RPC 関連定義の rpc_destination_mode オペランドの指定値に従ってサービス要求先を取得します。該当するサービスグループ名が見つからなかった場合, ee_rpc_cmtsend 関数は EERPCER_NO_SUCH_SERVICE_GROUP のリターン値でエラーリターンします。

ユーザサービス関連定義の service オペランドに自 TP1/EE のサービス名を指定しないで自プロセスの TP1/EE へ RPC 要求をした場合

ee_rpc_cmtsend 関数は EERPCER_NO_SUCH_SERVICE のリターン値でエラーリターンします。

注意事項

1. 引数に設定するサービスグループ名およびサービス名は, サーバ UAP の環境設定で設定しておいてください。

2. SPP へのサービス要求時にエラーが発生した場合、プロセス関連定義の `rpc_cmtsend_retry` オペランドの指定に従って、別処理スレッドによる再送信、またはエラートランザクション (ERRTRNS) の起動のどちらかの処理が実行されます。ただし、サービス要求元のトランザクションがエラートランザクションの場合は、エラーメッセージ出力後、これらの処理は実行されません。また、別処理スレッドによる再送信でエラーとなった場合は、エラートランザクション (ERRTRNS) が起動されます。
3. この関数を呼び出したトランザクションでは、トランザクションは最適化されません。
4. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。
5. リモート API 機能を使用した RPC はできません。
6. **【TP1/FSP 限定】** 次のリターン値の場合、出力メッセージ (OJ) は取得されません。
EECOMER_ENVIRON, EECOMER_CNDUOC, EECOMER_CNDBPP,
EERPCER_ARGUMENT, EERPCER_PARAM_GROUP, EERPCER_PARAM_SERVICE,
EERPCER_PARAM_IN, EERPCER_PARAM_IN_LEN, EERPCER_PARAM_FLAGS,
EERPCER_MESSAGE_TOO_BIG
7. 次の条件をすべて満たす場合は、自 TP1/EE のサービスグループ名を指定した RPC はできません。
 - 実行系連絡トランザクション (UI) である
 - `name_use` オペランドに Y を指定
 - `flags` に `EERPC_CMTSEND_MYPROC` を指定していない

ee_rpc_cmtsend_to

名称

通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求

形式

ANSI C, C++の形式

```
#include <eerrpc.h>
int ee_rpc_cmtsend_to(struct EERPC_BINDING_TBL *direction,
                     char *group, char *service,
                     char *in, EEULONG *in_len,
                     EELONG flags)
```

機能

SPP にサービスを要求します。サービスを要求するには、ホスト名またはノード識別子を指定した EERPC_BINDING_TBL 構造体に加え、「サービスグループ名」と「サービス名」を引数に設定します。EERPC_BINDING_TBL 構造体に指定したホスト名またはノード識別子は、サービス要求先を特定する検索のキーとして使用します。この設定に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

この関数を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定する引数

●direction

サービス要求先を特定する検索キーを格納する EERPC_BINDING_TBL 構造体のアドレスを指定します。検索のキーはホスト名またはノード識別子のどちらかです。

direction の指定値は ee_rpc_call_to 関数を参照してください。

●group

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●service

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●in

サービスの入力パラメタを設定します。

●in_len

サービスの入力パラメタ長を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerc.h で定義してあります。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerc.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●flags

EENOFLLAGS を設定します。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EERPCER_ARGUMENT	引数に設定した値が間違っています。 EERPC_BINDING_TBL 構造体の hostnm に指定されたホスト名が、/etc/hosts ファイルまたは DNS などで、IP アドレスとのマッピングができません。
EERPCER_CONDITION	関数の呼び出し条件が不正です。 RPC 関連定義の name_use オペランドに Y が設定されていません。
EERPCER_MEMORY_SHORT	メモリが不足しました。
EERPCER_MESSAGE_TOO_BIG	in_len に設定した入力パラメタ長が、最大値を超えています。
EERPCER_NO_SUCH_SERVICE_GROUP	group に設定したサービスグループ名は、定義されていません。
EERPCER_PARAM_FLAGS	flags に設定した値が間違っています。
EERPCER_PARAM_GROUP	group に設定した値が間違っています。
EERPCER_PARAM_IN	in に設定した値が間違っています。

リターン値	意味
EERPCER_PARAM_IN_LEN	in_len に設定した値が間違っています。
EERPCER_PARAM_SERVICE	service に設定した値が間違っています。
EERPCER_SYSERR	システムエラーが起きました。

注意事項

1. EERPC_BINDING_TBL 構造体にホスト名とノード識別子を同時に指定した場合、ホスト名の指定が有効になり、ノード識別子の指定は無視されます。
2. EERPC_BINDING_TBL 構造体にホスト名、ノード識別子ともに 0 を指定した場合、ee_rpc_cmtsend 関数と同様に動作します。
3. EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定し、ソケット受信型（ユーザーサービス定義の receive_from オペランドに socket を指定）の TP1/Server Base ユーザサーバへサービスを要求した場合、送信エラーとなります。
4. EERPC_BINDING_TBL 構造体の flags に EERPC_SCDPORT を指定して、ee_rpc_cmtsend_to 関数を呼び出した場合、rpc_retry オペランドは無効になります。
5. RPC 関連定義の name_use オペランドに N を設定して、ee_rpc_cmtsend_to 関数を呼び出した場合、EERPCER_CONDITION でエラーリターンします。
6. **【TP1/FSP 限定】** 次のリターン値の場合、出力メッセージ (OJ) は取得されません。
EECOMER_ENVIRON, EECOMER_CNDUOC, EECOMER_CNDBPP,
EERPCER_ARGUMENT, EERPCER_PARAM_GROUP, EERPCER_PARAM_SERVICE,
EERPCER_PARAM_IN, EERPCER_PARAM_IN_LEN, EERPCER_PARAM_FLAGS,
EERPCER_MESSAGE_TOO_BIG
7. その他の注意事項は、ee_rpc_cmtsend 関数の注意事項を参照してください。

ee_rpc_discard_further_replies

名称

すべての非同期応答型 RPC の応答受信拒否

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_discard_further_replies(EELONG flags)
```

機能

非同期応答型 RPC（ee_rpc_call 関数の flags に EERPC_NOWAIT を設定）で、返ってきていない応答を、これ以上受信しないことを示す関数です。この関数を呼び出したあとは、応答が返ってきても受信しないで捨てられます。

非同期応答型 RPC の結果をこれ以上受信しない場合は、必ず ee_rpc_discard_further_replies 関数を呼び出してください。呼び出さないと、ee_rpc_poll_any_replies 関数が不要な応答を受信してしまうことがあります。

ee_rpc_discard_further_replies 関数を使う場合を次に示します。

- 応答待ち時間切れになったあと、結果を保持しておくバッファを解放する場合
- 非同期応答型 RPC を複数回使って、そのうちの最初の応答だけ必要な場合

UAP で値を設定する引数

●flags

EENOFLLAGS を設定します。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EERPCEER_ARGUMENT	引数に設定した値が間違っています。

リターン値	意味
EERPCER_CONDITION	発行条件が不正です。

ee_rpc_discard_specific_reply

名称

特定の非同期応答型 RPC の応答受信拒否

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_discard_specific_reply(int des, EELONG flags)
```

機能

非同期応答型 RPC（ee_rpc_call 関数の flags に EERPC_NOWAIT を設定）で、返ってきていない特定の応答を、これ以上受信しないことを示す関数です。受信を拒否する非同期応答を特定するには、des に非同期応答型 RPC がリターンした際に返した記述子を設定します。この関数を呼び出したあとに返ってきた応答の中で、設定した記述子と同じ記述子を持つ応答は受信しないで捨てられます。

UAP で値を設定する引数

●des

非同期応答型 RPC の ee_rpc_call 関数（flags に EERPC_NOWAIT を設定）が正常に終了したときに返された、記述子を設定します。

●flags

EENOFLLAGS を設定します。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	発行条件が不正です。
EERPCER_INVALID_DES	des に設定した記述子は存在しません。要因として次のことが考えられます。 設定した記述子に対応する非同期応答型 RPC を行っていない。

リターン値	意味
EERPCER_INVALID_DES	非同期応答型 RPC の応答がすでに受信されている、または受信が拒否されている。
EERPCER_SYSERR	システムエラーが起きました。

ee_rpc_get_error_descriptor

名称

エラーが発生した非同期応答型 RPC の識別子取得

形式

ANSI C, C++の形式

```
#include <eerrpc.h>
int ee_rpc_get_error_descriptor(EELONG flags)
```

機能

非同期応答を特定しない ee_rpc_poll_any_replies 関数をエラーリターンした直後に呼び出すことで、エラーが発生した非同期応答型 RPC 要求に対応する記述子を取得します。

記述子を取得できるのは、サーバ UAP 側でエラーが発生した場合だけです。

ee_rpc_poll_any_replies 関数の呼び出し側でエラーが発生した場合には、この関数で記述子を取得できません。

UAP で値を設定する引数

●flags

EENOFLLAGS を設定します。

リターン値

ここで示すリターン値は、TP1/EE が返す値です。サービス関数から返される値ではありません。

リターン値	意味
正の整数	ee_rpc_poll_any_replies 関数が戻したエラーに対応する非同期応答型 RPC 要求の記述子を取得しました。
EE_OK	ee_rpc_poll_any_replies 関数が戻したエラーに対応する非同期応答型 RPC 要求の記述子を取得できませんでした。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	発行条件が不正です。

ee_rpc_get_watch_time

名称

サービスの応答待ち時間の参照

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_get_watch_time(void)
```

機能

現在のサービス要求の応答待ち時間を参照します（単位：秒）。この関数は、ee_rpc_set_watch_time 関数で応答待ち時間を一時的に変更する前に、元の値を退避するために使います。

この関数は、ee_rpc_set_watch_time 関数で変更したサービス応答待ち時間をリターンします。ee_rpc_set_watch_time 関数でサービス応答待ち時間を変更していない場合は RPC 関連定義の watch_time オペランドの値をリターンします。

この関数で得られる値は、TP1/EE の ee_rpc_call 関数に対して有効です。

リターン値

リターン値	意味
正の整数	現在のサービス応答待ち時間（秒）を示します。
EE_OK	サービス応答待ち時間は、無制限に待ち続ける指定です。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EERPCER_CONDITION	関数の呼び出し条件が不正です。

ee_rpc_poll_any_replies

名称

非同期応答型 RPC の応答受信

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_poll_any_replies(int des,
                           EELONG timeout, EELONG flags)
```

機能

非同期応答型 RPC (ee_rpc_call 関数の flags に EERPC_NOWAIT を設定) でサービス要求した結果を受信します。

受信する非同期応答を特定する場合は、flags に EERPC_SPECIFIC_MSG を設定します。このフラグを設定した場合は、des に設定した記述子をリターンした非同期応答型 RPC の応答を受信します。

受信する非同期応答を特定しない場合は、flags に EENOFLAGS を設定します。このとき、des に設定した値は無視されます。flags に EENOFLAGS を設定した ee_rpc_poll_any_replies 関数が正常に終了すると、受信した非同期応答の記述子と同じ値をリターンします。

ee_rpc_poll_any_replies 関数は、次のどちらかの場合にリターンします。

- 非同期応答型 RPC の応答を受信した場合
- timeout に設定した応答待ち時間が終了した場合

UAP で値を設定する引数

●des

非同期応答型 RPC の ee_rpc_call 関数 (flags に EERPC_NOWAIT を設定) が正常に終了したときに返される記述子を設定します。flags に EENOFLAGS を設定した場合は、ここに設定した値は無視されます。

●timeout

非同期応答型 RPC の ee_rpc_call 関数の結果が返ってくるまでの待ち時間を、秒単位またはミリ秒単位で設定します。設定できる値の範囲は、-1 から EELONG 型で表せる最大の数までです。

ee_rpc_poll_any_replies 関数で非同期応答を受信する場合は、RPC 関連定義の watch_time オペランドまたは ee_rpc_set_watch_time 関数で設定した応答待ち時間を参照しません。

0を設定した場合で、flagsにEERPC_WAIT_MILLISECを設定していないとき、応答が返ってこない
EERPCER_TIMED_OUTですぐにエラーリターンします。flagsにEERPC_WAIT_MILLISECを設定し
たときは50ミリ秒として処理します。

-1を設定した場合は、応答が返るまで待ち続けます。

【TP1/FSP 限定】 次のすべての条件を満たす場合、応答待ち時間は2,147,483秒となります。

- このオプションの設定値が2147484以上
- 待ち時間を秒単位で設定（flagsオプションにEERPC_WAIT_MILLISEC設定なし）
- RPC関連定義のrpc_reply_watch_intervalオペランドの設定値が1以上

●flags

RPCの形態とオプションを次に示す形式で設定します。

```
{EENOFLLAGS | EERPC_SPECIFIC_MSG}  [ | EERPC_WAIT_MILLISEC]
```

EENOFLLAGS

このee_rpc_poll_any_replies関数で受信する非同期の応答を特定しません。

EERPC_SPECIFIC_MSG

desに設定した記述子をリターンした、非同期応答型RPCの応答を受信します。

EERPC_WAIT_MILLISEC

timeoutで設定した待ち時間の単位をミリ秒にします。

リターン値

ここで示すリターン値は、TP1/EEが返す値です。サービス関数から返される値ではありません。

リターン値	意味
正の整数	受信した非同期応答の記述子を示します。正の整数は、flagsにEENOFLLAGSを設定したee_rpc_poll_any_replies関数が正常に終了した場合に返されます。
EE_OK	正常に終了しました。EE_OKは、flagsにEERPC_SPECIFIC_MSGを設定したee_rpc_poll_any_replies関数が正常に終了した場合に返されます。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOCから発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EEの環境下ではありません。
EERPCER_ALL_RECEIVED	非同期応答型RPCで要求したサービスの処理結果は、すべて受信しました。

リターン値	意味
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	発行条件が不正です。
EERPCER_INVALID_DES	des に指定した記述子は存在しません。このリターン値は、flags に EERPC_SPECIFIC_MSG を設定した場合に返されます。
EERPCER_INVALID_REPLY	サービス関数によって OpenTP1 に返された応答の長さが、1～EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
EERPCER_MEMORY_SHORT	メモリが不足しました。
EERPCER_NET_DOWN	ネットワークに障害が起こりました。
EERPCER_NO_BUFS_AT_SERVER	ee_rpc_call 関数の設定したサービスで、メモリが不足しました。 ee_rpc_call 関数で非トランザクショナル RPC (EERPC_TPNOTRAN 指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランドの設定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。応答を返せなかった原因を次に示します。 • UAP リターン後の同期点処理でトランザクション決着できない状態で、かつ rpc_replymsg_save オペランドの設定値が N • UAP リターン後の同期点処理以降にプロセスダウン
EERPCER_NO_BUFS_RB	メモリが不足しました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_NO_SUCH_SERVICE	ee_rpc_call 関数の service に設定したサービス名は、定義されていません。 サービスの要求先が TP1/EE であり、かつ要求先の転送機能が無効 (rpc_loadbalance オペランドの設定値が N、かつ rpc_transfer_othersvg オペランドの設定値が N) の場合は、サービスグループ名不正のおそれがあります。
EERPCER_NO_SUCH_SERVICE_GROUP	ee_rpc_call 関数の group に設定したサービスグループ名は、定義されていません。
EERPCER_OLTF_INITIALIZING	サービスを要求されたノードにある OpenTP1 は、開始処理中です。
EERPCER_OLTF_NOT_UP	ee_rpc_call 関数の service に設定したサービスがあるノードの OpenTP1 が稼働していません。異常終了、停止中、終了処理中、または通信障害が起こったことが考えられます。
EERPCER_TIMED_OUT	ee_rpc_call 関数の処理が時間切れ (タイムアウト) になりました。

リターン値	意味
EERPCER_TIMED_OUT	サービスを要求された SPP が、処理を完了する前に異常終了しました。 ee_rpc_call 関数の非トランザクショナル RPC (EERPC_TPNOTRAN 指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランドの設定値が J の場合、サーバ側のトランザクションがロールバック以外です。
EERPCER_SERVICE_NOT_UP	ee_rpc_call 関数の service に設定したサービスの UAP プロセスが、稼働していません。 timeout に -1 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。
EERPCER_REPLY_TOO_BIG	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。
EERPCER_REPLY_TOO_BIG_RB	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SERVER_BUSY	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。
EERPCER_SERVICE_CLOSED	ee_rpc_call 関数の service に設定したサービス名があるサービスグループは、閉塞しています。
EERPCER_SERVICE_TERMINATING	ee_rpc_call 関数の service に設定したサービスは、終了処理中です。
EERPCER_SYSERR	システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER	ee_rpc_call 関数の設定したサービスで、システムエラーが起きました。
EERPCER_SYSERR_AT_SERVER_RB	設定したサービスで、システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_SYSERR_RB	システムエラーが起きました。このリターン値が返った場合は、トランザクションブランチをコミットできません。
EERPCER_TESTMODE	オンラインテストを使っている環境で、テストモードの UAP からテストモードでない SPP へサービスを要求しています。または、テストモードでない UAP からテストモードの SPP へサービスを要求しています。
EERPCER_TRNCHK	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある OpenTP1 のバージョンが、クライアントの OpenTP1 のバージョンよりも低いため、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。

リターン値	意味
EERPCER_TRNCHK	このリターン値は、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
EERPCER_TRNCHK_EXTEND	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を超えたため、トランザクションブランチを開始できません。
	ee_rpc_call 関数で非トランザクショナル RPC (EERPC_TPNOTRAN 指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランドの設定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	ee_rpc_call 関数で非トランザクショナル RPC (EERPC_TPNOTRAN 指定) の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランドの設定値が J の場合、サーバ側のトランザクションがロールバックしました。

注意事項

1. 記述子を特定しない ee_rpc_poll_any_replies 関数がエラーリターンした場合、エラーとなった応答の記述子を特定できません。ee_rpc_poll_any_replies 関数がエラーリターンしたときに該当する記述子ができるようにしておきたい場合は、flags に EERPC_SPECIFIC_MSG を設定しておいてください。または、ee_rpc_get_error_descriptor 関数を使用することで、エラーリターンした記述子を取得することもあります。
2. 別スレッドで送信した非同期応答型 RPC (ee_rpc_call 関数の flags に EERPC_NOWAIT を設定) は、この関数で受信できません。非同期応答型 RPC は同一スレッドで ee_rpc_poll_any_replies 関数を使用して受信する必要があります。
3. 非同期応答型 RPC でサービスを要求した UAP が次に示す関数を呼び出すと、ee_rpc_poll_any_replies 関数で応答を受け取れません。
 - ee_rpc_discard_further_replies 関数で、非同期応答の受信を拒否した場合
 - トランザクショナル RPC の場合、同期点処理の関数でコミットまたはロールバックしたとき
上記の関数を使ったあとで返ってきた応答は、破棄されます。非同期応答型 RPC では、上記の関数を呼び出す前に、必要な非同期の応答を ee_rpc_poll_any_replies 関数ですべて受け取ってください。

ee_rpc_reply_send

名称

抑止していた RPC 応答メッセージの送信

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_reply_send(EEULONG id, char *out, EEULONG out_len,
                     EELONG flags)
```

機能

ee_rpc_reply_suspend 関数で抑止していた RPC 応答メッセージを送信します。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定する引数

●id

抑止識別子を設定します。ee_rpc_reply_suspend 関数で割り当てた抑止識別子を設定してください。

●out

サービス関数からの応答を設定します。

●out_len

サービス関数からの、実際の応答の長さを設定します（単位：バイト）。クライアント UAP から渡された out_len の値以下の数値を設定してください。

●flags

次のフラグを指定します。

EENOFLLAGS

この API の延長で RPC 応答メッセージを送信します。

EERPC_RSEND_CMT

この API を発行したトランザクションがコミット決着した時点で RPC 応答メッセージを送信します。
トランザクションがロールバック決着した場合は RPC 応答メッセージを送信せずに破棄します。

EERPC_RSEND_CMTERR

この API を発行したトランザクションがコミット決着した時点で RPC 応答メッセージを送信します。トランザクションがロールバック決着した場合は、RPC エラー応答メッセージ (EERPCER_TRNCHK_EXTEND) を送信します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EERPCER_AGAIN	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられません。
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	関数の呼び出し条件が不正です。
EERPCER_MEMORY_SHORT	メモリが不足しました。
EERPCER_NET_DOWN	ネットワークに障害が起きました。
EERPCER_NO_SERVICE_SOCKET	コネクションの取得に失敗しました。
EERPCER_NOTHING	id に設定した値は未登録か、すでにタイムアウトなどで応答が返されています。
EERPCER_PARAM_OUT_LEN	out_len に設定した値が間違っています。
EERPCER_SYSERR	システムエラーが起きました。

注意事項

1. 一つの抑止識別子で複数の RPC 応答メッセージの送信を抑止している場合、一度の ee_rpc_reply_send 関数で複数の RPC 応答メッセージが送信されます。
2. **【TP1/FSP 限定】** 次のリターン値の場合、出力メッセージ (OJ) は取得されません。
EECOMER_ENVIRON, EECOMER_CNDUOC, EECOMER_CNDBPP,
EERPCER_ARGUMENT, EERPCER_PARAM_OUT_LEN
3. ee_rpc_reply_send() の flags に EERPC_RSEND_CMT を指定し、かつ、そのトランザクションがロールバック決着した場合、RPC 応答メッセージは送信されません。そのため、RPC 要求元のクライアント UAP は、応答待ち時間を指定している場合は応答待ちタイムアウト、応答待ち時間を指定していない場合は無限に応答待ちを続けます。
4. ee_rpc_reply_send() の flags に EERPC_RSEND_CMT または EERPC_RSEND_CMTERR を指定し、かつ、そのトランザクションがプロセスリラン後にコミット決着した場合、RPC エラー応答メッセージ (EERPCER_NO_BUFS_AT_SERVER) を送信します。ロールバック決着した場合、

EERPC_RSEND_CMT は RPC 応答メッセージを破棄, EERPC_RSEND_CMTERR は RPC エラー応答メッセージ (EERPCER_TRNCHK_EXTEND) を送信します。

5. ee_trn_rollback_mark()発行済みの状態で, ee_rpc_reply_send()の flags に EERPC_RSEND_CMT または EERPC_RSEND_CMTERR を指定しても, ee_rpc_reply_send()は正常に終了します。その後, トランザクションがロールバック決着した時点で, EERPC_RSEND_CMT は RPC 応答メッセージを破棄, EERPC_RSEND_CMTERR は RPC エラー応答メッセージ (EERPCER_TRNCHK_EXTEND) を送信します。

ee_rpc_reply_suspend

名称

RPC 応答メッセージ送信の抑止

形式

ANSI C, C++の形式

```
#include <eerrpc.h>
int ee_rpc_reply_suspend(EEULONG type, EEULONG *id,
                        EELONG timeout, EELONG flags)
```

機能

サービス関数終了時の RPC 応答メッセージ送信を抑止します。抑止した RPC 応答メッセージは、ee_rpc_reply_send 関数を使って明示的に送信してください。トランザクションブランチ内では RPC 応答メッセージを抑止できません。

UAP で値を設定する引数

●type

抑止識別子の設定方法を設定します。

EERPC_IDTYPE_SYSTEM

TP1/EE が抑止識別子を割り当てます。

EERPC_IDTYPE_USER

ユーザが抑止識別子を指定します。

●id

type に EERPC_IDTYPE_USER を設定した場合に、抑止識別子を設定します。抑止識別子として 1～2147483647 の範囲の値を格納したポインタのアドレスを設定してください。

●timeout

RPC 応答メッセージ抑止限界時間を設定します（単位：秒）。指定できる値は 1～65535, EERPC_TIMEOUT_DEF, または EERPC_TIMEOUT_INF です。

EERPC_TIMEOUT_DEF を設定した場合、RPC 関連定義の rpc_reply_suspend_time オペランドで指定した値が抑止限界時間になります。EERPC_TIMEOUT_INF を設定した場合は、抑止限界時間は無制限になります。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●id

type に EERPC_IDTYPE_SYSTEM を設定した場合に、抑止識別子が返されます。抑止識別子は、TP1/EE が自動的に割り当てた、プロセス内で一意の識別子です。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下がありません。
EERPCER_ALREADY	RPC 応答メッセージ送信は応答抑止済みです。
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	関数の呼び出し条件が不正です。
EERPCER_LIMITS_OVER	定義で指定した同時抑止最大数を超えました。
EERPCER_PROTO	トランザクションブランチ内で関数を呼び出しています。
	非応答型 RPC で関数を呼び出しています。
	リモート API 機能による自サービスグループへの代理実行要求中に関数を呼び出しています。
	コネクションレス（UDP プロトコル）で起動されたトランザクション内で関数を発行しています。
	RPC メッセージの永続化機能が有効な状態で関数を発行しています。
EERPCER_SYSERR	システムエラーが起きました。

注意事項

1. この関数を呼び出したあと、RPC 応答メッセージ抑止限界時間以内に ee_rpc_reply_send 関数を呼び出さなかった場合、RPC エラー応答メッセージ（EERPCER_TIMED_OUT）が自動的に送信されます。そのあとで ee_rpc_reply_send 関数を呼び出しても、関数はエラーになります。
2. 同一の抑止識別子を複数の ee_rpc_reply_suspend 関数で指定できます。この場合、一度の ee_rpc_reply_send 関数で複数の RPC 応答メッセージが送信されます。
3. 非応答型 RPC でこの関数を呼び出した場合は、エラーリターンします。
4. **【XTC 限定】** コネクションレス（UDP プロトコル）で起動されたトランザクション内でこの関数を発行した場合は、エラーリターンします。

ee_rpc_set_watch_time

名称

サービスの応答待ち時間の更新

形式

ANSI C, C++の形式

```
#include <eerp.h>
int ee_rpc_set_watch_time(int var)
```

機能

サービス要求の応答待ち時間を変更します（単位：秒）。この関数で変更した値は、サービス関数終了時まで有効です。

サービス要求の応答待ち時間をこの関数を呼び出す前の値に戻すときは、ee_rpc_get_watch_time 関数で返された元の値を、この関数で再設定してください。

この関数は、RPC 関連定義の watch_time オペランドに指定した値を変更しません。この関数で設定する値は、あとから呼び出す ee_rpc_call 関数にだけ影響します。

UAP で値を設定する引数

●var

変更後のサービス応答待ち時間を設定します（単位：秒）。1～65535 の範囲で指定してください。無制限に待ち続ける場合は、0 を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EEDCOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EEDCOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EEDCOMER_ENVIRON	TP1/EE の環境下ではありません。
EERPCER_ARGUMENT	引数に設定した値が間違っています。
EERPCER_CONDITION	関数の呼び出し条件が不正です。

注意事項

サービス応答待ち時間に 0 を指定した場合、TP1/EE が終了しないことがあります。

滞留メッセージ受信 (ee_scd_～)

滞留メッセージを UAP 起動中に受信するときに使用する関数について説明します。滞留メッセージ受信の関数を次に示します。

- ee_scd_clear_rollback_cnt 【XTC 限定】 – ロールバックリトライ回数のクリア
- ee_scd_msg_receive 【XTC 限定】 – 滞留メッセージの受信
- ee_scd_msg_rollback 【XTC 限定】 – 受信メッセージの差し戻し
- ee_scd_msg_skip 【XTC 限定】 – 滞留メッセージのスキップ

ee_scd_clear_rollback_cnt 【XTC 限定】

名称

ロールバックリトライ回数のクリア

形式

ANSI C, C++の形式

```
#include <eescd.h>
int ee_scd_clear_rollback_cnt(char *service,
                             EELONG flags)
```

機能

ロールバックリトライ回数を 0 にクリアします。

UAP で値を設定する引数

●service

サービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EESCDER_ARGUMENT	引数に設定した値が間違っています。
EESCDER_CONDITION	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
EESCDER_SRVC_VALUE	service に設定したサービス名は、定義されていません。
EESCDER_TIMING	この関数を発行できる状態ではありません。

ee_scd_clear_rollback_cnt 関数がエラーになる場合

ee_scd_clear_rollback_cnt 関数がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合

ee_scd_clear_rollback_cnt 関数は EESCDER_CONDITION のリターン値でエラーリターンします。

service に設定したサービスのサービス属性定義に、service_attr -e serial を指定していない場合

ee_scd_clear_rollback_cnt 関数は EESCDER_CONDITION のリターン値でエラーリターンします。

この関数を発行したサービスと service に設定したサービスが異なり、他スレッドで service に設定したサービスのトランザクションが実行中の場合

ee_scd_clear_rollback_cnt 関数は EESCDER_TIMING のリターン値でエラーリターンします。

注意事項

1. この関数を発行したサービスと service に設定したサービスが同じで、トランザクションがロールバック決着した場合、ロールバックリトライ回数はカウントしないで、0 になります。XTC のロールバックリトライ回数監視によるサービス閉塞は行いません。
2. この関数を発行したサービスと service に設定したサービスが異なる場合、またはこの関数を発行したトランザクションがエラートランザクションの場合は、処理キュー全体の排他を取ります。そのため、この関数を頻繁に使用するとスループットが悪くなるおそれがあります。

ee_scd_msg_receive 【XTC 限定】

名称

滞留メッセージの受信

形式

ANSI C, C++の形式

```
#include <eescd.h>
int ee_scd_msg_receive(char **in, EEULONG *in_len, char **msg_inf,
                      EEULONG *msg_no, EELONG flags)
```

機能

入力キュー（ITQ）に滞留しているメッセージを受信します。

UAP で値を設定する引数

●in

入力メッセージ引き渡しエリアのアドレスを格納する領域アドレスです。

●in_len

入力メッセージ長を格納する領域アドレスです。

●msg_no

メッセージ通番を格納する領域アドレスです。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●in

入力メッセージ引き渡しエリアのアドレスが返されます。

●in_len

入力メッセージ長が返されます。

●msg_no

受信したメッセージのメッセージ通番が返されます。

●msg_inf

入力メッセージ情報が返されます。

ただし、受信メッセージが MCP メッセージの場合だけアドレスが返されます。

メッセージ情報を参照する場合は TRNINF_SERVICE_MCP にキャストしてください。

また、トランザクション起動時に通知する TRNINF_SERVICE_MCP の領域を使用するため、以前の情報は参照できなくなります。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EESCDER_ARGUMENT	引数に設定した値が間違っています。
EESCDER_CONDITION	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
EESCDER_INVALID_MESSAGE	トランザクション起動時のメッセージが滞留メッセージ読み出し対象メッセージではありません。
EESCDER_INVALID_TRNTYPE	この関数を発行できるサービスではありません（サービス属性定義で service_attr -e serial を指定していないサービスでこの関数を発行）。または、トランザクション種別が MN または TM ではありません。
EESCDER_NO_DATA	滞留メッセージはありません。※1
EESCDER_NO_MESSAGE	滞留メッセージ中に同種のメッセージがありません。
EESCDER_OVERFLOW	入力メッセージ引き渡しエリア（IMA）がオーバフローしました。※2
EESCDER_TIMING	スキップ処理中のため読み出せません。
EESCDER_TRN_CHANGE	中間同期点処理を行ったため、滞留メッセージの受信はできません。
EESCDER_UPPER_LIMIT	読み出し上限を超えました。

注※1

受信処理中の入力キュー（優先/通常）には滞留メッセージがない場合に設定されます。優先メッセージキューの受信処理中に優先メッセージをすべて受信処理した場合、通常メッセージキューに滞留メッセージがあっても EESCDER_NO_DATA となります。

注※2

メッセージ格納領域に格納できるサイズ分のメッセージは格納されています。
in_len には格納されているメッセージサイズが設定されています。

注意事項

1. 滞留メッセージを受信できるのは、次のメッセージ種別がトランザクション起動時に渡されたメッセージ種別と同じメッセージの場合です。
なお、タイマトランザクションの場合は、永続タイマと非永続タイマは同じメッセージ種別とは見なされません。
2. トランザクション起動時に渡されたメッセージ種別が RPC 受信メッセージであり、かつ応答型 RPC またはトランザクショナル RPC の場合は、滞留しているメッセージを受信できません。
また、リモート API 機能 (RAP) で送信されたメッセージも受信できません。
3. トランザクション起動時に引き出したキュー (優先キューまたは通常キュー) に滞留しているメッセージだけ受信できます。
例えば最初に引き出したキューが優先キューの場合、通常キューにメッセージが滞留していてもこの関数では受信できません。
4. 入力メッセージ引き渡しエリアおよび入力メッセージサイズ格納領域はトランザクション起動時に使用した領域を再使用します。この関数発行後に前のメッセージおよびメッセージサイズを参照する場合は領域の内容を退避したあと、この関数を発行してください。
5. トランザクション起動時のトランザクション種別が MN または TM の場合だけ、この関数を発行できます。
6. 該当サービスの同時引き出しの可否 (service_attr -e) に serial を指定した場合だけ、この関数を発行できます。
7. 滞留メッセージを受信できる最大数は、トランザクション起動時に滞留していたメッセージ数です。
8. 中間同期点処理 (ee_tm_chained_rollback 関数発行) を行ったあとは、この関数を発行できません。

ee_scd_msg_rollback 【XTC 限定】

名称

受信メッセージの差し戻し

形式

ANSI C, C++の形式

```
#include <eescd.h>
int ee_scd_msg_rollback(EEULONG msg_no, EELONG flags)
```

機能

トランザクションの終了状態に関係なく、滞留メッセージの受信関数（ee_scd_msg_receive 関数）で読み出したメッセージを差し戻します（ロールバックさせます）。

UAP で値を設定する引数

●msg_no

差し戻したいメッセージのメッセージ通番を設定します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EESCDER_ARGUMENT	引数に設定した値が間違っています。
EESCDER_CONDITION	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
EESCDER_INVALID_MESSAGE	トランザクション起動時のメッセージが滞留メッセージ読み出し対象メッセージではありません。
EESCDER_INVALID_NUMBER	指定したメッセージ通番が不正です（通番範囲外、または ee_scd_msg_receive 関数未発行）。

リターン値	意味
EESCDER_INVALID_TRNTYPE	この関数を発行できるサービスではありません（サービス属性定義で service_attr -e serial を指定していないサービスでこの関数を発行）。または、トランザクション種別が MN または TM ではありません。

注意事項

1. この関数は、トランザクション実行中に複数回発行できますが、最後に発行した関数で指定したメッセージ通番だけが有効となります。
2. ee_scd_msg_receive 関数を複数回発行した場合、この関数で指定したメッセージ通番以降の全メッセージが差し戻されます。
3. トランザクション起動時に渡されたメッセージはこの関数で差し戻すことはできません。差し戻したい場合はトランザクションをロールバックさせてください。

ee_scd_msg_skip 【XTC 限定】

名称

滞留メッセージのスキップ

形式

ANSI C, C++の形式

```
#include <eescd.h>
int ee_scd_msg_skip(char *service, EEULONG priority,
                    EEULONG skip_flg, EEULONG number,
                    EEULONG flags)
```

機能

入力キュー（ITQ）に滞留しているメッセージを読み出し済みとして入力キューから破棄します。

指定したサービス名に対応する入力キューのメッセージを読み出し済みとします。スキップ要求は、該当サービスの処理キューが引き出し禁止状態、かつ該当サービスのトランザクションが動作していない場合に受け付けられます。

UAP で値を設定する引数

●service

サービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には NULL 文字を設定してください。この NULL 文字は文字列の長さに数えません。

●priority

EENOFLLAGS を設定します。

EENOFLLAGS

優先キューと通常キューに滞留しているメッセージを読み出し済みにします。

●skip_flg

次のフラグを設定します。

EESCD_SKIP_ALL

入力キューに滞留しているすべてのメッセージを読み出し済みにします。

●number

0 を設定します。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EESCDER_ARGUMENT	引数に設定した値が間違っています。
EESCDER_CONDITION	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
EESCDER_MEMORY_SHORT	リソースが不足しました。
EESCDER_SRVC_VALUE	service に設定したサービス名は、定義されていません。
EESCDER_SYSERR	システムエラーが起きました。
EESCDER_TIMING	この関数を発行できる状態ではありません。

注意事項

スキップの対象となるメッセージがない場合は、正常扱いになります。

スレッド制御 (ee_thd_～)

スレッドを制御する関数について説明します。スレッド制御の関数を次に示します。

- ee_thd_abdctl – サービス閉塞の制御
- ee_thd_gettime – 現在時刻の取得
- ee_thd_protend – 保護区の終了
- ee_thd_protstart – 保護区の開始
- ee_thd_thdcntget – 処理スレッド数の取得
- ee_thd_uifaget – ユーザ IFA 情報格納領域アドレスの取得

ee_thd_abdctl

名称

サービス閉塞の制御

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_abdctl(char *svname, EELONG mode, EELONG flags)
```

機能

サービスを閉塞、および閉塞解除します。

UAP で値を設定する引数

●svname

SPP のサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

全サービスを対象にする場合は、サービス名を設定しないで先頭を NULL 文字にしてください。

●mode

閉塞か閉塞解除を設定します。

EETHD_ABDSET

サービスを閉塞します。

EETHD_ABDRST

サービスの閉塞を解除します。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。

リターン値	意味
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。
EETHDER_MEMORY_SHORT	メモリが不足しました。
EETHDER_NO_SERVICE	svname に設定したサービス名が間違っています。
EETHDER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. svname に、TP1/FSP（全銀 RC）で自動生成するサービスは指定できません。
2. すべてのサービスを対象にしてこの API を発行する場合、TP1/FSP（全銀 RC）で自動生成するサービスは閉塞/閉塞解除の対象外となります。

ee_thd_gettime

名称

現在時刻の取得

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_gettime(EELONG form, EELONG figure, void *time, EELONG flags)
```

機能

現在時刻を取得します。

ee_thd_gettime 関数で取得する時刻は、TP1/EE が管理している現在時刻、またはマシンの現在時刻になります。

TP1/EE が管理している現在時刻は、プロセス関連定義の clock_time_interval オペランドの指定値の間隔で更新し、60 秒ごとにマシン時刻と同期を取ります。マシン時刻との同期の際には、時刻の進みが発生することがあります。また、時刻の同期に失敗した場合、および同期時点でのマシン時刻との誤差が大きい場合はメッセージが出力されます。

取得時刻のタイムゾーンは、TP1/Server Base のユーザサービス定義に指定した TZ 環境変数に依存します。

UAP で値を設定するデータ領域

●form

時刻の取得形式を設定します。

EETHD_BINARY

BINARY 形式で取得します。

EETHD_DATE

DATE 形式で取得します。

EETHD_BINARY_TEXT

BINARY_TEXT 形式で取得します。

EETHD_DATE_TEXT

DATE_TEXT 形式で取得します。

●figure

取得時刻の精度を設定します。

EETHD_FIG_SEC

秒精度で取得します。この値を設定した場合は、TP1/EE が管理している現在時刻を取得します。

EETHD_FIG_MICROSEC

マイクロ秒精度で取得します。この値を設定した場合は、システムコールを使用してマシンの現在時刻を取得します。

●time

取得した時刻を格納する領域のアドレスを設定します。

格納する領域に必要なサイズは、form、および figure に設定した値によって異なります。次の表に示すサイズの領域を設定してください。

表 4-11 格納領域に必要なサイズ（単位：バイト）

form 指定値	figure 指定値	
	EETHD_FIG_SEC	EETHD_FIG_MICROSEC
EETHD_BINARY	4※1	8※2
EETHD_DATE	6	9
EETHD_BINARY_TEXT	7	11
EETHD_DATE_TEXT	12	18

注※1

領域のアドレスは 4 バイトのバウンダリを保証してください。

注※2

領域のアドレスは 8 バイトのバウンダリを保証してください。

●flags

EENOFLAGS を設定します。

TP1/EE から値が返されるデータ領域

●time

取得した時刻が設定されます。

取得形式のイメージを、次に示します。

- BINARY 形式・秒精度で取得する場合（form に EETHD_BINARY、figure に EETHD_FIG_SEC を設定）

図 4-1 BINARY 形式・秒精度の現在時刻表示

0	1	2	3	バイト
ss	ss	ss	ss	

SSSSSSSS

当日の 00:00:00 からの通算時刻です。0～8640000 がホストバイトオーダの符号なし整数で設定されます。

単位は 1/100 秒です。

(例) 通算時刻が 01:23:45 の場合、「0007AAE4」が設定されます。

- BINARY 形式・マイクロ秒精度で取得する場合 (form に EETHD_BINARY, figure に EETHD_FIG_MICROSEC を設定)

図 4-2 BINARY 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	バイト
ss	ss	ss	ss	ss	ss	ss	ss	

SSSSSSSSSSSSSSSS

当日の 00:00:00 からの通算時刻です。0～86400999999 がホストバイトオーダの符号なし整数で設定されます。

単位はマイクロ秒です。

(例) 通算時刻が 01:23:45.567890 の場合、「000000012B8C1492」が設定されます。

- DATE 形式・秒精度で取得する場合 (form に EETHD_DATE, figure に EETHD_FIG_SEC を設定)

図 4-3 DATE 形式・秒精度の現在時刻表示

0	1	2	3	4	5	バイト
YY	MM	DD	hh	mm	ss	

次の各値が 10 進数パック形式で設定されます。

- YY：当日の西暦年下 2 けた (00～99)
- MM：当日の月 (01～12)
- DD：当日の日 (01～31)
- hh：当日 0 時からの通算時 (00～23)
- mm：分 (00～59)
- ss：秒 (00～60)

(例) 時刻が 2010/12/31 23:59:59 の場合、「101231235959」が設定されます。

- DATE 形式・マイクロ秒精度で取得する場合 (form に EETHD_DATE, figure に EETHD_FIG_MICROSEC を設定)

図 4-4 DATE 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	バイト
YY	MM	DD	hh	mm	ss	ss	ss	ss	

次の各値が 10 進数パック形式で設定されます。

- YY：当日の西暦年下 2 けた (00～99)
- MM：当日の月 (01～12)
- DD：当日の日 (01～31)
- hh：当日 0 時からの通算時 (00～23)
- mm：分 (00～59)
- ssssssss：マイクロ秒 (00000000～60999999)

(例) 時刻が 2010/12/31 23:59:59.999999 の場合, 「101231235959999999」が設定されます。

- BINARY_TEXT 形式・秒精度で取得する場合 (form に EETHD_BINARY_TEXT, figure に EETHD_FIG_SEC を設定)

図 4-5 BINARY_TEXT 形式・秒精度の現在時刻表示

0	1	2	3	4	5	6	バイト
s	s	s	s	s	s	s	

SSSSSSS

当日の 00:00:00 からの通算時刻です。0000000～8640000 が 10 進数の文字列で設定されます。
下 2 けたは常に「00」となります。

単位は 1/100 秒です。

(例) 通算時刻が 01:23:45 の場合, 「0502500」が設定されます。

- BINARY_TEXT 形式・マイクロ秒精度で取得する場合 (form に EETHD_BINARY_TEXT, figure に EETHD_FIG_MICROSEC を設定)

図 4-6 BINARY_TEXT 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	バイト
s	s	s	s	s	s	s	s	s	s	s	

SSSSSSSSSS

当日の 00:00:00 からの通算時刻です。00000000000～86400999999 が 10 進数の文字列で設定されます。

単位はマイクロ秒です。

(例) 通算時刻が 01:23:45.567890 の場合, 「05025567890」が設定されます。

- DATE_TEXT 形式・秒精度で取得する場合 (form に EETHD_DATE_TEXT, figure に EETHD_FIG_SEC を設定)

図 4-7 DATE_TEXT 形式・秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	11	バイト
Y	Y	M	M	D	D	h	h	m	m	s	s	

次の各値が 10 進数の文字列で設定されます。

- YY：当日の西暦年下 2 けた (00～99)

- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）
- mm：分（00～59）
- ss：秒（00～60）

（例）時刻が 2010/12/31 23:59:59 の場合、「101231235959」が設定されます。

- DATE_TEXT 形式・マイクロ秒精度で取得する場合（form に EETHD_DATE_TEXT, figure に EETHD_FIG_MICROSEC を設定）

図 4-8 DATE_TEXT 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	バイト
Y	Y	M	M	D	D	h	h	m	m	s	s	s	s	s	s	s	s	

次の各値が 10 進数の文字列で設定されます。

- YY：当日の西暦年下 2 けた（00～99）
- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）
- mm：分（00～59）
- ssssssss：マイクロ秒（00000000～60999999）

（例）時刻が 2010/12/31 23:59:59.999999 の場合、「101231235959999999」が設定されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。
EETHDER_GETFAILURE	時刻取得に失敗しました。
EETHDER_LIMITS_OVER	取得した時刻が、TP1/EE 内で制御可能な上限値を超過しました。
EETHDER_UNRESOLVABLE	その他のエラーが発生しました。

注意事項

リターン値が EE_OK 以外の場合、time の内容は参照できません。

figure に EETHD_FIG_MICROSEC を指定した場合は、システムコールを使用してマシン時刻を取得します。そのため、figure に EETHD_FIG_MICROSEC を指定した ee_thd_gettime 関数を多量に呼び出すと、性能が低下するおそれがあります。

KFSB45310-E または KFSB45311-W メッセージが出力された前後の 60 秒間では、figure に EETHD_FIG_SEC を指定した場合の取得時刻とマシン時刻がずれていることがあります。

ee_thd_protend

名称

保護区の終了

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_protend(EELONG flags)
```

機能

保護区を終了します。保護区の外で異常が発生した場合は、スレッドダウンします。

UAP で値を設定する引数

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。
EETHDER_UNRESOLVABLE	そのほかのエラーが発生しました。

ee_thd_protstart

名称

保護区の開始

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_protstart(EELONG flags)
```

機能

保護区を開始します。保護区の中で異常が発生した場合は、プロセスダウンします。

UAP で値を設定する引数

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETHDER_ALREADY	すでに保護区になっています。
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。
EETHDER_UNRESOLVABLE	そのほかのエラーが発生しました。

ee_thd_thdcntget

名称

処理スレッド数の取得

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_thdcntget(EEULONG *pthd_cnt, EEULONG *ethd_cnt)
```

機能

処理スレッド数を取得します。

UAP で値を設定する引数

●pthd_cnt

通常処理スレッド数を格納する変数アドレスを設定します。

●ethd_cnt

予備処理スレッド数を格納する変数アドレスを設定します。

TP1/EE から値が返される引数

●pthd_cnt

通常処理スレッド数が返されます。

●ethd_cnt

予備処理スレッド数が返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。

リターン値	意味
EETHER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

リターン値が EE_OK 以外の場合、pthd_cnt および ethd_cnt に返された値を参照しても、内容は保証されません。

ee_thd_uifaget

名称

ユーザ IFA 情報格納領域アドレスの取得

形式

ANSI C, C++の形式

```
#include <eethd.h>
int ee_thd_uifaget(char **abnuser_ptr)
```

機能

異常終了したスレッドのユーザ IFA 情報が格納された領域のアドレスを取得します。

UAP で値を設定する引数

●abnuser_ptr

ユーザ IFA 情報格納領域アドレスを格納する変数アドレスを設定します。

TP1/EE から値が返される引数

●abnuser_ptr

ユーザ IFA 情報格納領域アドレスが格納された領域の先頭アドレスが返されます。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETHDER_ARGUMENT	引数に設定した値が間違っています。
EETHDER_CONDITION	関数の呼び出し条件が不正です。
EETHDER_UNRESOLVABLE	そのほかのエラーが発生しました。

注意事項

1. この関数はエラートランザクション (ERRTRN3) のトランザクションでだけ呼び出せます。エラートランザクション (ERRTRN3) のトランザクション内で、ee_trn_chained_commit 関数または ee_trn_chained_rollback 関数を呼び出したあとは、この関数を呼び出せません。
2. 処理スレッドの再生成が成功したあとで、この関数を呼び出した場合、異常終了スレッドのユーザ IFA 情報格納領域アドレスと、自スレッドのユーザ IFA 情報格納領域アドレスとは、同じになります。自スレッドのユーザ IFA 情報格納領域で引き継いだ情報が破壊されるおそれがあるので、領域のアクセスには注意してください。
この状態を確認するには、処理スレッド通番と異常発生元トランザクションの処理スレッド通番とを比較してください。これらが同じ場合は、同一スレッドで動作しています。
3. リターン値が EE_OK 以外の場合、abnuser_ptr に返された値を参照しても、内容は保証されません。

タイマトランザクション (ee_tim_～)

時刻指定または経過時間指定のトランザクション（タイマトランザクション）を制御する関数について説明します。タイマトランザクションの関数を次に示します。

- ee_tim_execap – タイマトランザクションの起動
- ee_tim_execap_cancel – タイマトランザクションの起動取り消し
- ee_tim_execap_cancel_all – 全タイマトランザクションの起動取り消し

ee_tim_execap

名称

タイマトランザクションの起動

形式

ANSI C, C++の形式

```
#include <eetim.h>
int ee_tim_execap(EEULONG acttype, EEULONG action, EEULONG active,
                  EEULONG reqid, char *service, EEULONG priority,
                  char *data, EEULONG dataleng, EEULONG flags)
```

機能

UAP からトランザクション（タイマトランザクション）を起動します。すぐに起動することも、経過時間または時刻を指定して起動することもできます。

UAP で値を設定する引数

●acttype

タイマトランザクションの起動方式を設定します。

EETIM_INST

即時登録方式です。コミットとは無関係に、action に設定した起動タイミングに従ってタイマトランザクションを起動します。

EETIM_COMT

トランザクション同期登録方式です。コミット後、action に設定した起動タイミングに従ってタイマトランザクションを起動します。コミット前には起動されません。

●action

タイマトランザクションの起動タイミングを設定します。

EETIM_JUST

タイマトランザクションを即時起動します。

EETIM_INTV

経過時間を指定してタイマトランザクションを起動します。active に設定した時間が経過したあとに起動します。

EETIM_TIME

時刻を指定してタイマトランザクションを起動します。active に設定した時刻に起動します。

●active

action に設定した起動タイミングに応じて次のように設定します。

- action に EETIM_JUST を設定した場合（即時起動）
設定値は無視されます。
- action に EETIM_INTV を設定した場合（経過時間指定の起動）
タイマトランザクションを起動するまでの経過時間を設定します（単位：100 ミリ秒）。1～3600000（100 ミリ秒から 100 時間）の範囲で指定してください。
acttype に EETIM_INST（即時登録方式）を設定した場合は、ee_tim_execap 関数を呼び出してから経過時間です。acttype に EETIM_COMT（トランザクション同期登録方式）を設定した場合は、コミットからの経過時間です。
- action に EETIM_TIME を設定した場合（時刻指定の起動）
タイマトランザクションを起動する時刻を設定します。0 時 0 分 0 秒を基準にして、起動時刻を秒数で算出します。0（0 時 0 分 0 秒に開始）～86399（23 時 59 分 59 秒に開始）の範囲で指定してください。
(例)

午後 2 時 30 分 30 秒に起動させる場合、次のように 52230 を設定します。

$$14 \times 3600 + 30 \times 60 + 30 = 52230$$

設定した時刻が現在の時刻より前またはあとだった場合、タイマトランザクションの起動処理は、プロセス関連定義の execap_permission_time オペランドで指定した先行許容時間または後退許容時間に従います。

詳細は「2.3.3 時刻または経過時間の指定方法」および「2.3.6 時刻または経過時間の指定と TP1/EE サービス定義との関係」を参照してください。

●reqid

ee_tim_execap_cancel 関数でタイマトランザクションを取り消す場合のキーとなるリクエスト識別子を設定します。

タイマトランザクションを取り消す必要がない場合は 0 を設定します。0 を設定した場合でも、acttype に EETIM_COMT（即時登録方式）を設定して登録したトランザクションは、コミット前であれば ee_tim_execap_cancel_all 関数で取り消せます。

●service

タイマトランザクションとして起動するトランザクションのサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には NULL 文字を付けてください。この NULL 文字は文字列の長さに数えません。

●priority

タイマトランザクションが起動タイミングを迎えたとき、その時点で登録されている処理キューの中で優先的に起動するかどうかを設定します。

【TP1/FSP 限定】 トランザクションレベル方式の処理キュー制御機能を使用している場合、この引数の指定値は有効にならないで、タイマトランザクションは優先的に起動しません。優先的に起動したい場合は、トランザクションレベル名判定 UOC でプライオリティのいちばん高いトランザクションレベル名を指定してください。

EETIM_HI
優先的に起動します。

EETIM_LOW
優先的に起動しないで、登録した順序で起動します。

●data

起動するトランザクションに渡すデータを設定します。

●dataleng

起動するトランザクションに渡すデータの長さを設定します（単位：バイト）。メモリ関連定義の time_message_size オペランドで指定した値以下の値を指定してください。渡すデータがない場合は 0 を設定します。

●flags

オプションを次に示す形式で設定します。

```
{EENOFLAGS} [ | EETIM_PERMANENCE ] [ |EETIM_SCDL_HOLD]
```

EENOFLAGS
必ず設定してください。

EETIM_PERMANENCE **【XTC 限定】**
タイマトランザクションを永続属性とする場合に設定します。acttype に EETIM_COMT を設定した場合だけ有効です。

EETIM_SCDL_HOLD **【XTC 限定】**
自トランザクションが終了するまで次のトランザクションのスケジュールを停止したい場合（UAP リターン後の次のトランザクションのスケジュールを停止したい場合）に設定します。このオプションを設定すると、待機系への転送処理終了後に、次のトランザクションのスケジュールを行います。
acttype に EETIM_COMT を設定した場合だけ有効です。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。

リターン値	意味
EETIMER_ENVIRON	TP1/EE の環境下ではありません。
EETIMER_ARGUMENT	引数に設定した値が間違っています。
EETIMER_CONDITION	関数の呼び出し条件が不正です。
EETIMER_DATA_OVER	dataleng に設定したデータ長が大き過ぎます。
EETIMER_NO_ENTRY	指定したサービス名に対するエントリポイントがありません。
EETIMER_NO_SERVICE	service に設定したサービス名が間違っています。
EETIMER_PROHIBIT	TP1/EE システムが終了処理中のため、登録できません。または、CL サーバでの待機系のため登録できません。
EETIMER_QUEUE_FULL	指定したサービスのキューが満杯のため登録できません。
EETIMER_RESOURCE_SHORT	次に示す要因によって、資源不足で登録できません。 - メモリ関連定義の icb_no の指定値を超えています。 - メモリ関連定義の time_message_no の指定値を超えています。 - メモリ関連定義の pce_no の指定値を超えています。 - UDP 用送信バッファ（UOBF）面数が不足しています。
EETIMER_TIME_INJUSTICE	指定した時刻が先行および後退許容時間範囲内にありません。

注意事項

- acttype に EETIM_COMT を設定した場合、コミット前にトランザクションが異常終了したとき、またはロールバックしたときは、この関数の要求は無効になります。同一のトランザクション内で ee_trn_rollback_mark 関数がすでに呼び出されている場合は、リターン値 EETIMER_CONDITION でエラーリターンします。
- この関数は、トランザクションのコミット時に実行されます。acttype に EETIM_COMT を設定した場合、トランザクションの同期点を取得する処理でエラーが起これば、この関数の処理は無効となります。
- acttype に EETIM_COMT を設定してこの関数を呼び出したトランザクションでは、トランザクションは最適化されません。
- acttype に EETIM_COMT、action に EETIM_TIME を設定した場合、関数呼び出し時に先行許容時間および後退許容時間がチェックされます。この時点で許容時間外の場合はエラーになり、コミット時には許容時間はチェックされません。
- acttype に EETIM_COMT、action に EETIM_TIME を設定した場合、コミット時に後退許容時間外となっていたときは、タイマトランザクションが即時に起動します。
- acttype に EETIM_INST を設定した場合、トランザクションが異常終了したとき、またはロールバックしたときも、この関数の要求は有効になります。
- システム終了（正常終了、計画停止 A）の場合、登録済みのタイマトランザクションがあるときは、タイマ破棄通知のエラートランザクション（ERRTRN4）が起動します。計画停止 B の場合は、登録済みのタイマトランザクションがあっても、ERRTRN4 は起動しません。

8. **【XTC 限定】** CL サーバがない場合、flags に EETIM_PERMANENCE を設定しても永続属性となりません。
9. **【XTC 限定】** 永続属性の場合、UDP 用送信バッファを使用します。
10. **【XTC 限定】** CL サーバでの待機系ではこの関数は使用しないでください。

ee_tim_execap_cancel

名称

タイマトランザクションの起動取り消し

形式

ANSI C, C++の形式

```
#include <eetim.h>
int ee_tim_execap_cancel(EEULONG acttype, EEULONG reqid, EELONG flags)
```

機能

タイマトランザクションを取り消します。ee_tim_execap 関数でタイマトランザクションを登録するときに設定した起動タイミングより前であれば、ee_tim_execap_cancel 関数でタイマトランザクションを取り消せます。

UAP で値を設定する引数

●acttype

どの起動方式のタイマトランザクションを取り消すかを設定します。

EETIM_INST

次のタイマトランザクションを、コミットとは無関係に直ちに取り消します。

- acttype で EETIM_INST（即時登録方式）を設定した、ee_tim_execap 関数のタイマトランザクション
- acttype で EETIM_COMT（トランザクション同期登録方式）を設定した、コミット後の ee_tim_execap 関数のタイマトランザクション

EETIM_COMT

ee_tim_execap 関数で acttype に EETIM_COMT（トランザクション同期登録方式）を設定したタイマトランザクションを取り消します。コミット前であれば、ee_tim_execap 関数を呼び出したトランザクションからタイマトランザクションを取り消せます。

EETIM_CMTC

次のタイマトランザクションを、コミット後に取り消します。

- acttype で EETIM_INST（即時登録方式）を設定した、ee_tim_execap 関数のタイマトランザクション
- acttype で EETIM_COMT（トランザクション同期登録方式）を設定した、コミット後の ee_tim_execap 関数のタイマトランザクション

●reqid

タイマトランザクションのリクエスト識別子を設定します。ee_tim_execap 関数でタイマトランザクションを登録したときに reqid で設定したリクエスト識別子と同じものを設定してください。

同一のリクエスト識別子のタイマトランザクションが複数登録されている場合、acttype で EETIM_INST または EETIM_CMTC を設定したときは、起動時刻が最も早いタイマトランザクションが取り消されます。acttype で EETIM_COMT を設定したときは、最も早く登録要求されたタイマトランザクションが取り消されます。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETIMER_ARGUMENT	引数に設定した値が間違っています。
EETIMER_CONDITION	関数の呼び出し条件が不正です。
EETIMER_NOTHING	設定されたリクエスト識別子を持つタイマトランザクションは未登録、起動中、または起動済みです。

注意事項

1. タイマトランザクションを登録するときに ee_tim_execap 関数で acttype に EETIM_COMT を設定した場合、ee_tim_execap 関数を呼び出したトランザクションがコミットしたあとは、ee_tim_execap_cancel 関数で acttype に EETIM_COMT を設定してもタイマトランザクションを取り消せません。コミット後に取り消したい場合は、ee_tim_execap_cancel 関数で acttype に EETIM_INST または EETIM_CMTC を設定してください。
2. ee_tim_execap 関数で設定した起動タイミングに達しているタイマトランザクションは取り消せません。
3. acttype で EETIM_CMTC を設定した場合、対象のタイマトランザクションが同期点処理前に起動時刻に達しても、タイマトランザクションを起動しません。
4. acttype に EETIM_CMTC を設定した場合、コミット前にトランザクションが異常終了したとき、またはロールバックしたときは、この関数の要求は無効になります。このとき、起動時刻に達している場合はタイマトランザクションを即時起動します。同一のトランザクション内で ee_trn_rollback_mark 関数がすでに呼び出されている場合は、リターン値 EETIMER_CONDITION でエラーリターンします。

ee_tim_execap_cancel_all

名称

全タイマトランザクションの起動取り消し

形式

ANSI C, C++の形式

```
#include <eetim.h>
int ee_tim_execap_cancel_all(EELONG flags)
```

機能

トランザクション同期登録方式で登録したすべてのタイマトランザクションを取り消します。ee_tim_execap関数を呼び出したトランザクションがコミット前であれば、そのトランザクションでee_tim_execap関数のacttypeにEETIM_COMTを設定して登録したすべてのタイマトランザクションを、ee_tim_execap_cancel_all関数で取り消せます。

UAP で値を設定する引数

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETIMER_ARGUMENT	引数に設定した値が間違っています。
EETIMER_NOTHING	このトランザクションではタイマトランザクションは未登録です。

注意事項

- 1.ee_tim_execap関数でacttypeにEETIM_INSTを設定して登録したタイマトランザクションは、ee_tim_execap_cancel_all関数では取り消せません。
- 2.ee_tim_execap関数でタイマトランザクションを登録したトランザクションからee_tim_execap_cancel_all関数を呼び出してください。別のトランザクションで

ee_tim_execap_cancel_all 関数を呼び出しても、タイマトランザクションを取り消すことはできません。

3. ee_tim_execap 関数を呼び出したトランザクションのコミット後にタイマトランザクションを取り消したい場合は、ee_tim_execap_cancel 関数を使用してください。

トラブルシュート (ee_trb_～)

トラブルシュートに使用する関数について説明します。トラブルシュートの関数を次に示します。

- ee_trb_uinfo_tm – ユーザ情報の TASKTM 出力
- ee_trb_utrace_put – ユーザ固有のトレースの取得

名称

ユーザ情報の TASKTM 出力

形式

ANSI C, C++の形式

```
#include <eetrb.h>
int ee_trb_uinfo_tm(char *uinfo_pt, EELONG type, EELONG flags)
```

機能

指定されたユーザ情報を TASKTM の「ユーザ情報」列に出力します。

UAP で値を設定する引数

●uinfo_pt

出力するユーザ情報格納領域アドレスを設定します。ユーザ情報は 63 バイトで指定してください。63 バイトに満たない場合は後続の領域に、キャラクタ形式の場合は半角スペース、バイナリ形式の場合は 0x00 を設定してください。TASKTM には 63 バイト固定で出力します。

●type

指定したユーザ情報の属性を次の値で指定します。

EETRB_UINFO_BINARY：バイナリ属性で出力

指定したユーザ情報を TASKTM 編集時に 16 進数形式で出力します。

EETRB_UINFO_CHARACTER：キャラクタ属性で出力

指定したユーザ情報を TASKTM 編集時にキャラクタ形式で出力します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。

リターン値	意味
EETRBER_ARGUMENT	引数に設定した値が間違っています。
EETRBER_CONDITION	発行条件が不正です。
EETRBER_STOP	トラブルシュート機能停止中です。

注意事項

1. トラブルシュート関連定義の `trb_tasktm_type` に 1, 2, 3 を指定した場合に有効です。
1, 2, 3 以外を指定した場合は出力されません。また, 1, 2, 3 以外を指定した場合でも, この API はエラーにはなりません。
2. 同一トランザクション内で複数発行した場合は, 最後に要求したユーザ情報が有効となります。
3. 文字コードのチェックは行いません。

ee_trb_utrace_put

名称

ユーザ固有のトレースの取得

形式

ANSI C, C++の形式

```
#include <eetrb.h>
int ee_trb_utrace_put(unsigned short event_id,
                     unsigned short datalen,
                     char *buffaddr, EELONG flags)
```

機能

ユーザ固有のトレースデータを回線トレースに出力します。出力したトレースデータは、eetrblined コマンドまたは eetrblinedump コマンドで編集できます。

UAP で値を設定する引数

●event_id

トレースを取得するイベントのイベント ID を設定します。0x0700~0x0740 の範囲のイベント ID を設定してください。

●datalen

トレースのデータ長を設定します（単位：バイト）。4~180 の範囲のデータ長を設定してください。

●buffaddr

トレースデータが格納されているバッファの先頭ポインタを設定します。

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRBER_ARGUMENT	引数に設定した値が間違っています。
EETRBER_BUF_LEN	引数 datalen に設定したデータ長が設定できる範囲外です。

リターン値	意味
EETRBER_STOP	トラブルシュート機能が縮退中です。

トランザクション制御 (ee_trn_～)

TP1/EE 独自のトランザクション制御をする関数について説明します。トランザクション制御の関数を次に示します。

- ee_trn_chained_commit – 連鎖モードのコミット
- ee_trn_chained_rollback – 連鎖モードのロールバック
- ee_trn_getinfo – 現在のトランザクション識別情報の取得
- ee_trn_rm_close_mark – リソースマネージャとの接続の解放指示
- ee_trn_rollback_mark – トランザクションのロールバック指示
- ee_trn_set_exp_time – トランザクション処理の監視時間の変更

ee_trn_chained_commit

名称

連鎖モードのコミット

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_chained_commit(void)
```

機能

トランザクションの同期点を取得します。グローバルトランザクションのルートトランザクションブランチとして、処理が正常に終了したこと（コミット）を、トランザクションを構成するトランザクションブランチのUAP、トランザクションサービス、およびリソースマネージャに知らせます。

ee_trn_chained_commit 関数が正常に終了すると、新しいグローバルトランザクションが開始します。関数を呼び出したスレッドは、このトランザクションの範囲内です。

グローバルトランザクションが複数のトランザクションブランチから構成されるとき（関数を呼び出したUAP だけではないとき）は、それぞれのトランザクションブランチの処理結果がコミットとならない限りコミットされません。

ee_trn_chained_commit 関数を呼び出せるのはルートトランザクションブランチだけです。それ以外のUAP から呼び出した場合は、リターン値 EETRNER_CONDITION でエラーリターンします。

ee_trn_chained_commit 関数を呼び出すスレッドは、このマニュアルの記述に従って正しく作成されたUAP を稼働させたものでなければなりません。

ee_trn_chained_commit 関数は、同期点処理が完了したときに正常に終了、またはエラーリターンのどちらかで返ります。

サービス関数リターン時に明示的に ee_trn_chained_commit 関数または ee_trn_chained_rollback 関数を呼び出さない場合は、トランザクションの同期点を取得します。

トランザクション起動後に UAP をリターンするまでの、ee_trn_chained_commit 関数の呼び出し回数の上限値を、トランザクション関連定義の trn_max_commit_count オペランドで指定できます。

ee_trn_chained_commit 関数の呼び出し回数が、trn_max_commit_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_commit_downmode オペランドの指定に従って、異常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

リターン値

リターン値	意味
EE_OK※	正常に終了しました。ee_trn_chained_commit 関数が終了したあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRNER_COMPLETION	通信障害やリソースマネージャ障害などによってトランザクション決着に失敗したため、トランザクション回復処理を行います。このリターン値が返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
EETRNER_CONDITION	ee_trn_chained_commit 関数を不正なコンテキスト（ルートトランザクションではないなど）で呼び出しています。トランザクションモードに対する影響はありません。
EETRNER_HAZARD	グローバルトランザクションのトランザクションブランチがヒューリスティックに決着しました。しかし、障害のため、ヒューリスティックに決着したトランザクションブランチの同期点の結果がわかりません。このリターン値が返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
EETRNER_HEURISTIC	ee_trn_chained_commit 関数を呼び出したグローバルトランザクションは、ヒューリスティック決着のため、あるトランザクションブランチはコミットとなり、あるトランザクションブランチはロールバックとなりました。このリターン値は、ヒューリスティック決着の結果が、グローバルトランザクションの同期点の結果と一致しなかった場合にリターンします。このリターン値が返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
EETRNER_ROLLBACK_OK	コミット処理に失敗した、または ee_trn_rollback_mark 関数呼び出し済みのため、ロールバックを行いました。

注※

次の要求がある場合は、要求を実行します。

- ee_mch_cmtsend 関数によるトランザクション非同期の一方送信メッセージの送信要求（TP1 キャッシュ機能を使用する場合）
- ee_mch_cmtsend_sync 関数によるトランザクション同期の一方送信メッセージの送信要求（TP1 キャッシュ機能を使用する場合）

- ee_mcp_send 関数による非同期一方送信メッセージの送信要求（MCP を使用する場合）
- ee_rpc_cmtsend 関数によるトランザクションと同期して送信する非応答型 RPC 要求
- ee_rpc_cmtsend_to 関数による通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求
- ee_tim_execap 関数（acttype に EETIM_COMT を設定）によるタイマトランザクション起動要求

指定例

```
if(ee_trn_chained_commit() != EE_OK)
    fputs("cannot commit transaction\n",stderr);
```

注意事項

1. 次の状態のとき、ee_trn_chained_commit 関数は正常終了します。

- リソースマネージャ接続失敗によるトランザクション縮退起動
- 新しいグローバルトランザクション生成時にリソースマネージャ接続失敗

必要に応じて ee_trn_getinfo 関数を呼び出してリソースマネージャの状態を確認してください。

2. **【TP1/FSP 限定】** DB キューの交替用メッセージ表書き込み機能使用時に、この関数内で行う DB キューのメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。

ee_trn_chained_rollback

名称

連鎖モードのロールバック

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_chained_rollback(EEULONG thkind, EEULONG ecode)
```

機能

トランザクションをロールバックさせます。thkind に EETR_N_KEEP を設定した場合、ee_trn_chained_rollback 関数を呼び出したあとには、続けてトランザクションが開始します。thkind に EETR_N_KEEP 以外を設定した場合、新たなトランザクションを開始しないで強制終了します。

ee_trn_chained_rollback 関数を呼び出すことで、ルートトランザクションブランチから、トランザクションブランチ、トランザクションサービス、およびリソースマネージャにロールバックを知らせます。

thkind に EETR_N_KEEP を設定した ee_trn_chained_rollback 関数が正常に終了すると、関数を呼び出したスレッドはロールバックしてリターンします。そのあとで新しいグローバルトランザクションが開始します。関数を呼び出したスレッドはこのトランザクションの範囲内です。ただし、この関数を呼び出した UAP 以外のトランザクションモードに対しての指定を意味しません。

ee_trn_chained_rollback 関数を呼び出せるのは、ルートトランザクションブランチからだけです。それ以外の UAP から呼び出した場合は、リターン値 EETRNER_CONDITION でエラーリターンします。ただし、thkind に EETR_N_PRCS, EETR_N_KILL, または EETR_N_ABRT を指定した場合は、トランザクションブランチから呼び出せます。

ee_trn_chained_rollback 関数を呼び出すスレッドは、このマニュアルの記述に従って正しく作成された UAP を稼働させたものでなければなりません。

ee_trn_chained_rollback 関数は、同期点処理が完了したときに正常に終了、エラーリターン、または強制終了します。

サービス関数リターン時に明示的に ee_trn_chained_commit 関数または ee_trn_chained_rollback 関数を呼び出さない場合は、トランザクションの同期点を取得します。

トランザクション起動後に UAP をリターンするまでの、ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数の呼び出し回数の上限値を、トランザクション関連定義の trn_max_rollback_count オペランドで指定できます。ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数の呼び出し回数が、trn_max_rollback_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_rollback_downmode オペランドの指定に従って、

異常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

UAP で値を設定する引数

●thkind

ロールバック時、プロセスまたはスレッドを強制終了させるかどうかを設定します。

EETR_N_PRC

コアファイルと COBOL 異常終了時要約リスト（COBOL プログラムがある場合）を出力してプロセスを強制終了させます。

EETR_N_KILL

スレッドを強制終了させます。

EETR_N_ABRT

メモリダンプファイルと COBOL 異常終了時要約リスト（COBOL プログラムがある場合）を出力してスレッドを強制終了させます。

EETR_N_KEE

プロセスおよびスレッドを強制終了させません。

●ecode

トランザクション情報としてエラー トランザクション（ERRTRN3）へ引き渡すエラー要因コードを設定します。

thkind に EETR_N_KILL または EETR_N_ABRT を設定したときだけ有効です。

リターン値

リターン値	意味
EE_OK	正常に終了しました。ee_trn_chained_rollback 関数が終了したあとも、このスレッドはトランザクション下にあって、グローバル トランザクションの範囲内です。
EECOMER_CNDBPP	オフライン バッチ プロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRNER_ARGUMENT	引数に設定した値が間違っています。
EETRNER_COMPLETION	通信障害やリソース マネージャ 障害などによって トランザクション 決着に失敗したため、トランザクション 回復処理を行います。このリターン値が返る原因になった UAP、リソース マネージャ、および グローバル トランザクションの同期点の結果は、メッセージ ログ ファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下にあって、グローバル トランザクションの範囲内です。

リターン値	意味
EETRNER_CONDITION	ee_trn_chained_rollback 関数を不正なコンテキスト（ルートトランザクションブランチではないなど）で呼び出しています。トランザクションモードに対する影響はありません。
EETRNER_HAZARD	グローバルトランザクションのトランザクションブランチがヒューリスティックに決着しました。しかし、障害のため、ヒューリスティックに決着したトランザクションブランチの同期点の結果がわかりません。このリターン値が返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
EETRNER_HEURISTIC	ee_trn_chained_rollback 関数を呼び出したグローバルトランザクションは、ヒューリスティック決着のため、あるトランザクションブランチはコミットとなり、あるトランザクションブランチはロールバックとなりました。このリターン値は、ヒューリスティック決着の結果が、グローバルトランザクションの同期点の結果と一致しなかった場合にリターンします。このリターン値が返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このリターン値が返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。

指定例

```
if (ee_trn_chained_rollback(EETRNER_KILL,100) != EE_OK)
    fputs("cannot rollback transaction\n",stderr);
```

注意事項

- トランザクション種別が MI（初期化トランザクション）、ME（終了トランザクション）、JI（共有リソース初期化トランザクション）、および UT（ユーザ初期化トランザクション）の場合、thkind に EETRNER_ABRT を設定してもメモリダンプファイルは出力されません。また、thkind に EETRNER_KILL または EETRNER_ABRT を設定してもスレッドは強制終了しないでプロセスダウンになり、コアファイルが出力されます。
- 次の状態のとき、ee_trn_chained_rollback 関数は正常終了します。
 - リソースマネージャ接続失敗によるトランザクション縮退起動
 - 新しいグローバルトランザクション生成時にリソースマネージャ接続失敗
必要に応じて ee_trn_getinfo 関数を呼び出してリソースマネージャの状態を確認してください。

ee_trn_getinfo

名称

現在のトランザクション識別情報の取得

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_getinfo(EETRNIID2 *eetrnid, EELONG flags)
```

機能

実行中のトランザクションのトランザクション識別情報をリソースマネージャごとに取得します。リターン値が EE_OK（次のリソースマネージャ情報なし）になるまで、この関数を連続して呼び出すことによって、接続中のすべてのリソースマネージャの情報を取得できます。リターン値が EE_OK の状態でこの関数を呼び出した場合は、最初からトランザクション識別情報を取得します。

UAP で値を設定する引数

●eetrnid

トランザクション識別情報を格納するための構造体ポインタのアドレスを設定します。

●flags

EENOFLLAGS を設定します。

TP1/EE から値が返される引数

●eetrnid

トランザクション識別情報が構造体 EETRNIID2 で返されます。構造体の形式は次のとおりです。

```
struct ee_trn_id_t2{
    EELONG processid;
    EEULONG centraltransactionid;
    EELONG resume;
    int rmid;
    int size;
    char name[EETRNI_RM_NAMESZ];
    int opensize;
    char openname[OPENNAMESZ];
    int xidsize;
    char xid[XIDSZ];
    char dummy[1024];
}EETRNIID2;
```

- processid (プロセス ID)
プロセス ID が返されます。
 - centraltransactionid (中央処理通番)
該当するトランザクションの中央処理通番が返されます。
 - resume (リソースマネージャ接続状態)
リソースマネージャとの接続状態が返されます。
- EETRN_RM_CONNECT
リソースマネージャ接続正常
- EETRN_RM_ERROR
リソースマネージャ接続失敗によるトランザクション縮退起動
- rmid (リソースマネージャ ID)
リソースマネージャ ID が返されます。
 - size (リソースマネージャ名長)
リソースマネージャ名 (リソースマネージャ拡張子を含む) の長さが返されます (単位: バイト)。
 - name (リソースマネージャ名)
リソースマネージャ名 (リソースマネージャ拡張子を含む) が返されます。領域の長さ (EETRN_RM_NAMESZ) は 36 バイトです。
 - opensize (xa_open 関数用文字列長)
xa_open 関数用文字列の長さが返されます (単位: バイト)。
 - openname (xa_open 関数用文字列)
xa_open 関数用文字列が返されます。領域の長さ (OPENNAMESZ) は 256 バイトです。
 - xidsize (XID 長)
XID の長さが返されます (単位: バイト)。
 - xid (XID)
XID が返されます。領域の長さ (XIDSZ) は 256 バイトです。
 - dummy
予備の領域です。

リターン値

リターン値	意味
EE_OK	正常に終了しました (次のリソースマネージャ情報なし)。
EE_CONTINUE	正常に終了しました (次のリソースマネージャ情報あり)。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。

リターン値	意味
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRNER_ARGUMENT	引数に設定したポインタが NULL のため値を設定できません。

指定例

```
int      returncode;
EETRnid2 eetrnid;
      :
memset(&eetrnid, '¥0', sizeof(EETRnid2));
      :
returncode = EE_CONTINUE;
while(returncode == EE_CONTINUE){
    returncode = ee_trn_getinfo(&eetrnid, EENOFARGS);
    if(returncode == EE_OK || returncode == EE_CONTINUE){
        printf("プロセスID = %d¥n ", eetrnid.processid);
        printf("中央処理通番 = %u¥n ", eetrnid.centraltransactionid);
        :
    }
    else{
        printf("cannot get transaction information¥n");
    }
}
```

注意事項

リソースマネジャがない場合はリソースマネジャ情報以外の値が設定され、リターン値 EE_OK（次のリソースマネジャ情報なし）でリターンします。

ee_trn_rm_close_mark

名称

リソースマネージャとのコネクションの解放指示

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_rm_close_mark(EELONG flags)
```

機能

ee_trn_rm_close_mark 関数が正常に終了すると、ee_trn_rm_close_mark 関数を呼び出したスレッドのトランザクションの同期点処理が完了した時点でスレッドにリンクされているすべてのリソースマネージャに対してコネクションの解放要求（xa_close 関数）が指示されます。同期点処理の完了時にコネクション確立済みのリソースマネージャが存在しない場合は、ee_trn_rm_close_mark 関数を呼び出してもコネクションの解放要求（xa_close 関数）を指示しません。

リソースマネージャとのコネクションの再確立（xa_open 関数）は、ee_trn_rm_close_mark 関数を呼び出したスレッドでの次のトランザクション開始時に TP1/EE が行います。

UAP で値を設定する引数

●flags

EENOFLLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRNER_ARGUMENT	引数に設定した値が間違っています。

指定例

```
if (ee_trn_rm_close_mark(EENOFLLAGS) != EE_OK){
    fputs("cannot close connection\n",stderr);
}
```

ee_trn_rollback_mark

名称

トランザクションのロールバック指示

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_rollback_mark(void)
```

機能

トランザクションのロールバックを指示します。ee_trn_rollback_mark 関数が正常に終了すると、関数を呼び出したスレッドのトランザクションはロールバックされます。

トランザクション起動後に UAP をリターンするまでの、ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数の呼び出し回数の上限値を、トランザクション関連定義の trn_max_rollback_count オペランドで指定できます。ee_trn_chained_rollback 関数または ee_trn_rollback_mark 関数の呼び出し回数が、trn_max_rollback_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_rollback_downmode オペランドの指定に従って、異常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

リターン値

リターン値	意味
EE_OK	同期点処理をロールバックさせます。また、ロールバックが終了したあとも、このスレッドはトランザクション下において、グローバルトランザクションの範囲内です。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。

指定例

```
if (ee_trn_rollback_mark() != EE_OK)
    fputs("cannot rollback transaction\n",stderr);
```


注意事項

ee_trn_rollback_mark 関数が正常終了したあとは、ee_rpc_cmtsend 関数によるサービス要求、および ee_tim_execap 関数（acttype に EETIM_COMT を設定）によるタイマトランザクション起動要求を実行してもエラーになります。

ee_trn_set_exp_time

名称

トランザクション処理の監視時間の変更

形式

ANSI C, C++の形式

```
#include <eetrn.h>
int ee_trn_set_exp_time(EEULONG time,EELONG flags)
```

機能

トランザクション処理の監視時間を変更します。

ee_trn_set_exp_time 関数が正常に終了すると、関数を呼び出したトランザクション処理の監視が、time に設定した秒数で再開始されます。このとき、関数呼び出し前の処理時間監視は無効となります。

ee_trn_set_exp_time 関数によるトランザクション処理の監視時間の変更は、呼び出し元トランザクションにだけ反映されます。

ee_trn_set_exp_time 関数は、同一トランザクション内で 2 回以上呼び出すことはできません。

同一トランザクション内で ee_trn_watch 関数を発行済みの場合、この関数は使用できません。

UAP で値を設定する引数

●time

変更後のトランザクション処理の監視時間を設定します（単位：秒）。指定できる値は 1～65535 です。0 を設定した場合は、時間監視をしません。

●flags

EENOFLAGS を設定します。

リターン値

リターン値	意味
EE_OK	正常に終了しました。
EECOMER_CNDBPP	オフラインバッチプロセスから発行しているため、この機能は使用できません。
EECOMER_CNDUOC	UOC から発行しているため、この機能は使用できません。
EECOMER_ENVIRON	TP1/EE の環境下ではありません。
EETRNER_ARGUMENT	引数に設定した値が間違っています。

リターン値	意味
EETRNER_CALLAGAIN	ee_trn_set_exp_time 関数を同一トランザクション内で 2 回以上呼び出しました。
EETRNER_UNRESOLVABLE	その他のエラーが発生しました。

指定例

```
if (ee_trn_set_exp_time(100, EENOFLLAGS) != EE_OK)
    fputs("cannot change trn_expiration_time¥n", stderr);
```

注意事項

ee_trn_set_exp_time 関数によるトランザクション処理の監視時間の変更は、呼び出し元のトランザクションにだけ反映されるため、eetrntim コマンドの-l オプションで表示するトランザクション処理の監視時間には反映されません。

5

TP1/EE および XTC のライブラリ関数の文法 (COBOL 言語編)

この章では、COBOL 言語の UAP で使用できる TP1/EE および XTC のライブラリ関数の文法について説明します。

COBOL-UAP 作成用プログラムの説明形式

TP1/EE の UAP を COBOL 言語で作成するときは、TP1/EE のライブラリにある関数に対応した COBOL-UAP 作成用プログラムを CALL 文で呼び出します。

COBOL-UAP 作成用プログラムを次の形式で説明します。

形式

ライブラリにある関数と対応する COBOL-UAP 作成用プログラムを、CALL 文で呼び出す形式と、領域の指定方法を示します。

データ名に値を指定するときは、ここで示す PICTURE 句のデータ形式に従ってください。指定する値が決まっているときは VALUE 句で記述してあります。

一意名で示すファイル名とデータ名には、断りがないかぎり、固有の名称を任意に付けてください。

データ名として指定する文字の長さなどは、コーディングで使う COBOL 言語および COBOL コンパイラの仕様に従ってください。

COBOL 言語でコーディングするときは、TP1/EE のサンプルにある COBOL 言語用テンプレートを使えます。この COBOL 言語用テンプレートを、コーディングするプログラムに合わせて修正すれば、DATA DIVISION を最初からコーディングする手間が省けます。COBOL 言語用テンプレートは、インストールディレクトリ/examples/tp1ee/aplib/cobol/ディレクトリの下に、システムサービスごとのファイル名で格納してあります。ファイル名は、次に示す規則で付けてあります。

EEXXX.cbl (XXX は、COBOL-UAP 作成用プログラムの下 3 文字)

機能

COBOL-UAP 作成用プログラムの機能について説明します。以降、COBOL-UAP 作成用プログラムを、次に示す形式で表記します。

CBLEEXXX (' XXXXXXXX ')
└──────────┘ └──────────┘
 要求コードを示します。
└──────────────────────────┘
 COBOL-UAP 作成用プログラムのプログラム名を示します。

UAP で値を設定するデータ領域

DATA DIVISION に指定するデータのうち、COBOL-UAP 作成用プログラムの呼び出し時にデータ領域に値を指定しておくデータ名です。各データ名の説明に従って、値を設定してください。

TP1/EE から値が返されるデータ領域

DATA DIVISION に指定するデータのうち、CALL 文を実行したあとで、TP1/EE から値が返されるデータ名です。CALL 文の実行後に、データ名で示すデータ領域の内容を参照してください。

クライアント UAP から値が渡されるデータ領域

サービスプログラムの場合で、クライアント UAP から値が渡されるデータ領域です。このデータ領域の内容を参照して、サービスプログラムの処理をしてください。

サーバ UAP から値が返されるデータ領域

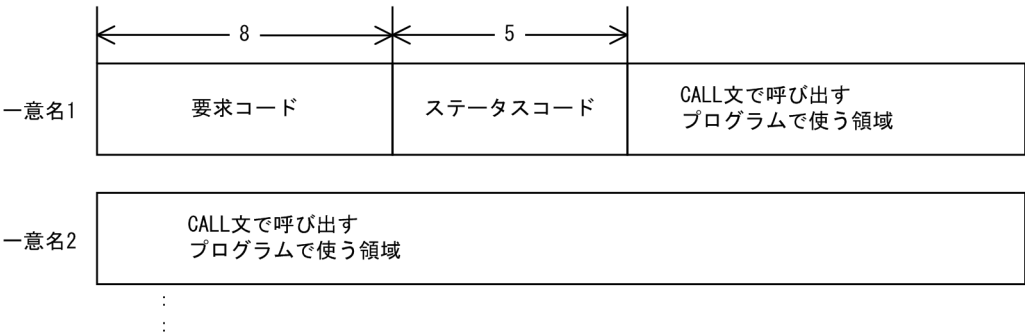
同期応答型 RPC の場合に、サービスプログラムから値が返されるデータ名です。CBLEERPC('CALL ') を呼び出した UAP では、ここに示すデータ領域の値を参照できます。

ステータスコード

CALL 文を実行したときに返される値を、表形式で説明します。ステータスコードによって、COBOL-UAP 作成用プログラムが正常に実行されたかがわかります。エラーが起こったときは、エラーの内容を示します。

COBOL 言語のステータスコードは 5 けたの数字列で、USING 句で指定する最初の一意名に含まれます。CALL 文での USING 句で指定する一意名とステータスコードの関係を次に示します。

CALL '呼び出すプログラム名' USING 一意名 1 一意名 2 ………



指定例

指定例が必要な場合に記述します。

注意事項

COBOL-UAP 作成用プログラムを使うときの注意を記述します。

サービスプログラムの作成 (SPP)

TP1/EE および XTC の UAP のサービスプログラムの文法について説明します。SPP はサービスプログラムを作成します。

形式 (TP1/EE の場合)

32 ビットアーキテクチャの場合

```
PROGRAM-ID. プログラム名.
LINKAGE SECTION.
01一意名1.
    02 データ名A PIC X(n).
01一意名2.
    02 データ名B PIC S9(9) COMP.
01一意名3.
    02 データ名C PIC X(n).
01一意名4.
    02 データ名D PIC S9(9) COMP.
01一意名5.
    02 データ名E PIC 9(9) COMP. /*情報長 */
    02 データ名F PIC 9(9) COMP. /*サービスグループ名長 */
    02 データ名G PIC X(32). /*サービスグループ名 */
    02 データ名H PIC 9(9) COMP. /*サービス名長 */
    02 データ名I PIC X(32). /*サービス名 */
    02 データ名J PIC X(2). /*トランザクション種別 */
    02 データ名K PIC X(1). /*自動閉塞有無 */
    02 データ名L PIC X(1). /*コミット有無 */
    02 データ名M PIC 9(9) COMP. /*処理スレッド通番 */
    02 データ名N PIC 9(9) COMP-X. /*エラー要因コード */
    02 データ名O PIC 9(9) COMP. /*ユーザIFAエリアサイズ */
    02 データ名P ADDRESS. /*ユーザIFAエリアポインタ */
    02 データ名Q PIC 9(9) COMP. /*ユーザSATエリアサイズ */
    02 データ名R ADDRESS. /*ユーザSATエリアポインタ */
    02 データ名S PIC X(1). /*応答要否 */
    02 データ名T PIC X(1). /*メッセージ状態 */
    02 データ名U PIC X(1). /*プロセス起動要因 */
    02 データ名V PIC X(1). /*前回プロセス終了要因 */
    02 データ名W PIC X(1). /*プロセス終了要因 */
    02 データ名X PIC X(1). /*スレッドダウン種別 */
    02 データ名Y PIC X(1). /*異常発生トランザクション種別 */
    02 データ名Z PIC X(1). /*拡張トランザクション種別 */
    02 データ名AA PIC 9(9) COMP. /*リソースマネージャ数 */
    02 データ名AB PIC S9(9) COMP. /*リソースマネージャ接続状態 */
    02 データ名AC PIC 9(9) COMP. /*送信サービスグループ名長 */
    02 データ名AD PIC X(32). /*送信サービスグループ名 */
    02 データ名AE PIC 9(9) COMP. /*送信サービス名長 */
    02 データ名AF PIC X(32). /*送信サービス名 */
    02 データ名AG PIC 9(9) COMP. /*異常発生処理スレッド通番 */
    02 データ名AH ADDRESS. /*拡張トランザクションインタフェース情報*/
    02 データ名AI ADDRESS. /*XTC拡張トランザクションインタフェース情報*/
    02 データ名AJ ADDRESS. /*MCP拡張トランザクションインタフェース情報*/
    02 データ名AK PIC X(1). /*メッセージ種別 */
    02 データ名AL PIC X(1). /*エラートランザクション要因 */
    02 データ名AO PIC X(1). /*トランザクション起動要因 */
```

```

02 データ名AU PIC X(1). /*リソースマネージャ連携有無 */
02 データ名AP PIC 9(9) COMP. /*サーバ再起動回数 */
02 データ名AQ PIC X(14). /*トランザクション起動時刻 */
02 データ名AR PIC X(5). /*ノード識別子 */
02 データ名AX PIC X(1). /*リトライ回数 */
02 データ名AS PIC 9(9) COMP. /*トランザクションレベル名長 */
02 データ名AT PIC X(32). /*トランザクションレベル名 */
02 FILLER PIC X(24). /*予備 */
02 データ名AY PIC X(32). /*トランザクション引き継ぎ情報 */

```

/* DBキュー機能およびオンラインバッチ機能を使用しない場合の形式 */

```

01 一意名6.
02 データ名DA PIC 9(9) COMP. /*情報長 */
02 データ名DB PIC S9(9) COMP. /*サービスプログラム起動時の応答の長さ*/
02 データ名DC PIC X(192). /*予備 */

```

/* DBキュー機能を使用する場合の形式 */

```

01 一意名6.
02 データ名BA PIC 9(9) COMP. /*情報長 */
02 データ名BB PIC X(28). /*読み込みDBキュー名*/
02 データ名BC PIC 9(9) COMP. /*通番 */
02 データ名BD PIC 9(4) COMP. /*ロールバック回数 */
02 データ名BE PIC X(162). /*予備 */

```

/* オンラインバッチ機能を使用する場合の形式 */

```

01 一意名6.
02 データ名CA PIC 9(9) COMP. /*情報長 */
02 データ名CB PIC 9(9) COMP. /*トランザクション種別 */
02 データ名CC PIC X(28). /*バッチキュー名 */
02 データ名CD PIC X(24). /*ロット名 */
02 データ名CE PIC 9(9) COMP. /*引き継ぎ情報のサイズ */
02 データ名CF ADDRESS. /*引き継ぎ情報のアドレス */
02 データ名CG PIC 9(9) COMP. /*ロットのバッチ処理形態 */
02 データ名CH PIC 9(9) COMP. /*該当するロットの全メッセージ数*/
02 データ名CI PIC 9(9) COMP. /*読み込んだメッセージの通番 */
02 データ名CJ PIC 9(9) COMP. /*ロット終了要因コード */
02 データ名CK PIC 9(9) COMP. /*終了時のステータスコード */
02 データ名CL PIC 9(9) COMP. /*読み出し済みメッセージ数 */
02 データ名CM PIC 9(9) COMP. /*スキップメッセージ数 */
02 データ名CN PIC 9(9) COMP. /*未読み出しメッセージ数 */
02 データ名CO PIC X(1). /*メッセージ長情報 */
02 データ名CR PIC X(99). /*予備 */

```

PROCEDURE DIVISION USING 一意名1 一意名2 一意名3 一意名4 一意名5

```

:
サービスの処理
:
EXIT PROGRAM.

```

注

データ名 BB, データ名 CC, およびデータ名 CD の文字列の最後は LOW-VALUE です。

64 ビットアーキテクチャの場合

```

PROGRAM-ID. プログラム名.
LINKAGE SECTION.

```


01一意名1.		
02データ名A	PIC X(n).	
01一意名2.		
02データ名B	PIC S9(9) COMP.	
01一意名3.		
02データ名C	PIC X(n).	
01一意名4.		
02データ名D	PIC S9(9) COMP.	
01一意名5.		
02データ名E	PIC 9(9) COMP.	/*情報長 */
02データ名F	PIC 9(9) COMP.	/*サービスグループ名長 */
02データ名G	PIC X(32).	/*サービスグループ名 */
02データ名H	PIC 9(9) COMP.	/*サービス名長 */
02データ名I	PIC X(32).	/*サービス名 */
02データ名J	PIC X(2).	/*トランザクション種別 */
02データ名K	PIC X(1).	/*自動閉塞有無 */
02データ名L	PIC X(1).	/*コミット有無 */
02データ名M	PIC 9(9) COMP.	/*処理スレッド通番 */
02データ名N	PIC 9(9) COMP-X.	/*エラー要因コード */
02データ名O	PIC 9(9) COMP.	/*ユーザIFAエリアサイズ */
02 FILLER	PIC X(4).	/*予備 */
02データ名P	ADDRESS.	/*ユーザIFAエリアポインタ */
02データ名Q	PIC 9(9) COMP.	/*ユーザSATエリアサイズ */
02 FILLER	PIC X(4).	/*予備 */
02データ名R	ADDRESS.	/*ユーザSATエリアポインタ */
02データ名S	PIC X(1).	/*応答要否 */
02データ名T	PIC X(1).	/*メッセージ状態 */
02データ名U	PIC X(1).	/*プロセス起動要因 */
02データ名V	PIC X(1).	/*前回プロセス終了要因 */
02データ名W	PIC X(1).	/*プロセス終了要因 */
02データ名X	PIC X(1).	/*スレッドダウン種別 */
02データ名Y	PIC X(1).	/*異常発生トランザクション種別 */
02データ名Z	PIC X(1).	/*拡張トランザクション種別 */
02データ名AA	PIC 9(9) COMP.	/*リソースマネージャ数 */
02データ名AB	PIC S9(9) COMP.	/*リソースマネージャ状態 */
02データ名AC	PIC 9(9) COMP.	/*送信サービスグループ名長 */
02データ名AD	PIC X(32).	/*送信サービスグループ名 */
02データ名AE	PIC 9(9) COMP.	/*送信サービス名長 */
02データ名AF	PIC X(32).	/*送信サービス名 */
02データ名AG	PIC 9(9) COMP.	/*異常発生処理スレッド通番 */
02データ名AS	PIC 9(9) COMP.	/*トランザクションレベル名長 */
02データ名AH	ADDRESS.	/*拡張トランザクションインタフェース情報*/
02データ名AI	ADDRESS.	/*XTC拡張トランザクションインタフェース情報*/
02データ名AJ	ADDRESS.	/*MCP拡張トランザクションインタフェース情報*/
02データ名AK	PIC X(1).	/*メッセージ種別 */
02データ名AL	PIC X(1).	/*エラートランザクション要因 */
02データ名AO	PIC X(1).	/*トランザクション起動要因 */
02データ名AU	PIC X(1).	/*リソースマネージャ連携有無 */
02データ名AP	PIC 9(9) COMP.	/*サーバ再起動回数 */
02データ名AM	PIC 9(18) COMP.	/*拡張ユーザIFAエリアサイズ */
02データ名AN	PIC 9(18) COMP.	/*拡張ユーザSATエリアサイズ */
02データ名AQ	PIC X(14).	/*トランザクション起動時刻 */
02データ名AR	PIC X(5).	/*ノード識別子 */
02データ名AX	PIC X(1).	/*リトライ回数 */
02データ名AT	PIC X(32).	/*トランザクションレベル名 */
02 FILLER	PIC X(4).	/*予備 */
02データ名AY	PIC X(32).	/*トランザクション引き継ぎ情報 */

```

/* DBキュー機能およびオンラインバッチ機能を使用しない場合の形式 */
01 一意名6.
  02 データ名DA PIC 9(9) COMP. /*情報長 */
  02 データ名DB PIC S9(9) COMP. /*サービスプログラム起動時の応答の長さ*/
  02 データ名DC PIC X(192). /*予備 */

```

```

/* DBキュー機能を使用する場合の形式 */
01 一意名6.
  02 データ名BA PIC 9(9) COMP. /*情報長 */
  02 データ名BB PIC X(28). /*読み込みDBキュー名*/
  02 データ名BC PIC 9(9) COMP. /*通番 */
  02 データ名BD PIC 9(4) COMP. /*ロールバック回数 */
  02 データ名BE PIC X(162). /*予備 */

```

```

/* オンラインバッチ機能を使用する場合の形式 */
01 一意名6.
  02 データ名CA PIC 9(9) COMP. /*情報長 */
  02 データ名CB PIC 9(9) COMP. /*トランザクション種別 */
  02 データ名CC PIC X(28). /*バッチキュー名 */
  02 データ名CD PIC X(24). /*ロット名 */
  02 データ名CE PIC 9(9) COMP. /*引き継ぎ情報のサイズ */
  02 データ名CF ADDRESS. /*引き継ぎ情報のアドレス */
  02 FILLER PIC X(4). /*予備 */
  02 データ名CG PIC 9(9) COMP. /*ロットのバッチ処理形態 */
  02 データ名CH PIC 9(9) COMP. /*該当するロットの全メッセージ数*/
  02 データ名CI PIC 9(9) COMP. /*読み込んだメッセージの通番 */
  02 データ名CJ PIC 9(9) COMP. /*ロット終了要因コード */
  02 データ名CK PIC 9(9) COMP. /*終了時のステータスコード */
  02 データ名CL PIC 9(9) COMP. /*読み出し済みメッセージ数 */
  02 データ名CM PIC 9(9) COMP. /*スキップメッセージ数 */
  02 データ名CN PIC 9(9) COMP. /*未読み出しメッセージ数 */
  02 データ名CO PIC X(1). /*メッセージ長情報 */
  02 データ名CR PIC X(91). /*予備 */

```

```

PROCEDURE DIVISION USING 一意名1 一意名2 一意名3 一意名4 一意名5
:
サービスの処理
:
EXIT PROGRAM.

```

注

データ名 BB, データ名 CC, およびデータ名 CD の文字列の最後は LOW-VALUE です。

形式 (XTC の場合)

64 ビットアーキテクチャの場合

```

PROGRAM-ID. プログラム名.
LINKAGE SECTION.
01一意名1.
  02 データ名A PIC X(n).
01 一意名2.
  02 データ名B PIC S9(9) COMP.
01 一意名3.
  02 データ名C PIC X(n).

```

```

01 一意名4.
02 データ名D PIC S9(9) COMP.
01 一意名5.
02 データ名E PIC 9(9) COMP. /*情報長 */
02 データ名F PIC 9(9) COMP. /*サービスグループ名長 */
02 データ名G PIC X(32). /*サービスグループ名 */
02 データ名H PIC 9(9) COMP. /*サービス名長 */
02 データ名I PIC X(32). /*サービス名 */
02 データ名J PIC X(2). /*トランザクション種別 */
02 データ名K PIC X(1). /*自動閉塞有無 */
02 データ名L PIC X(1). /*コミット有無 */
02 データ名M PIC 9(9) COMP. /*処理スレッド通番 */
02 データ名N PIC 9(9) COMP-X. /*エラー要因コード */
02 データ名O PIC 9(9) COMP. /*ユーザIFAエリアサイズ */
02 FILLER PIC X(4). /*予備 */
02 データ名P ADDRESS. /*ユーザIFAエリアポイント */
02 データ名Q PIC 9(9) COMP. /*ユーザSATエリアサイズ */
02 FILLER PIC X(4). /*予備 */
02 データ名R ADDRESS. /*ユーザSATエリアポイント */
02 データ名S PIC X(1). /*応答要否 */
02 データ名T PIC X(1). /*メッセージ状態 */
02 データ名U PIC X(1). /*プロセス起動要因 */
02 データ名V PIC X(1). /*前回プロセス終了要因 */
02 データ名W PIC X(1). /*プロセス終了要因 */
02 データ名X PIC X(1). /*スレッドダウン種別 */
02 データ名Y PIC X(1). /*異常発生トランザクション種別 */
02 データ名Z PIC X(1). /*拡張トランザクション種別 */
02 データ名AA PIC 9(9) COMP. /*リソースマネージャ数 */
02 データ名AB PIC S9(9) COMP. /*リソースマネージャ状態 */
02 データ名AC PIC 9(9) COMP. /*送信サービスグループ名長 */
02 データ名AD PIC X(32). /*送信サービスグループ名 */
02 データ名AE PIC 9(9) COMP. /*送信サービス名長 */
02 データ名AF PIC X(32). /*送信サービス名 */
02 データ名AG PIC 9(9) COMP. /*異常発生処理スレッド通番 */
02 FILLER PIC X(4). /*予備 */
02 データ名AH ADDRESS. /*拡張トランザクションインタフェース情報 */
02 データ名AI ADDRESS. /*XTC拡張トランザクションインタフェース情報 */
02 データ名AJ ADDRESS. /*MCP拡張トランザクションインタフェース情報 */
02 データ名AK PIC X(1). /*メッセージ種別 */
02 データ名AL PIC X(1). /*ERRTRN4起動要因 */
02 FILLER PIC X(6). /*予備 */
02 データ名AM PIC 9(18) COMP. /*拡張ユーザIFAエリアサイズ */
02 データ名AN PIC 9(18) COMP. /*拡張ユーザSATエリアサイズ */
02 FILLER PIC X(56). /*予備 */

```

/* 拡張トランザクションインタフェース情報 */

```

01 一意名6.
02 データ名EA PIC 9(9) COMP. /*情報長 */
02 データ名EB PIC S9(9) COMP. /*サービスプログラム起動時の応答の長さ */
02 データ名EC PIC X(192). /*予備 */

```

/* XTC拡張トランザクションインタフェース情報 */

```

01 一意名7.
02 データ名EA PIC 9(9) COMP. /*情報長 */
02 データ名EB PIC X(1). /*系情報 */
02 データ名EC PIC X(1). /*系切り替え前の実行系の状態 */
02 データ名ED PIC X(1). /*出力キュー (OTQ) 自動閉塞 */
02 データ名EE PIC X(1). /*出力キュー自動閉塞レベル */

```

02	データ名EF	PIC 9(9) COMP.	/*出力キュー通番	*/
02	データ名EG	PIC 9(9) COMP.	/*エラー要因通番	*/
02	データ名EH	PIC 9(9) COMP.	/*入力キュー (ITQ) 通番	*/
02	データ名EI	PIC 9(4) COMP.	/*ロールバックリトライ回数	*/
02	データ名EJ	PIC X(1).	/*系切り替え後の実行系の状態	*/
02	データ名EX	PIC X(169).	/*予備	*/

機能

サービスを実行して結果を返す SPP のサービスプログラムです。クライアント UAP で CBLEERPC('CALL')や CBLEERPC('CMTSND')を呼び出したとき、TP1/EE プロセスの開始・終了などによってトランザクションが発生したとき、DB キュー機能でキューを読み込んだときにサービスを要求されます。

サービスプログラムの処理は、上記の形式で任意に作成してください。サービス名は、サービスプログラムのプログラム ID に対応させます。この対応づけは、UAP の実行環境を設定するときに指定します。UAP の実行環境を設定する方法を次に示します。

- TP1/EE の場合
ユーザサービス関連定義で指定します。

データ領域の設定

サービスプログラムには、次の値がデータ領域に渡されます。

- 入力パラメタ (データ名 A)
- 入力パラメタ長 (データ名 B)
- 応答の長さ (データ名 D)
- トランザクションインタフェース情報 (一意名 5 のデータ名、および一意名 6 のデータ名)

入力パラメタと入力パラメタの長さは、クライアント UAP で設定した値が、文字コードや数字の表現形式などを変換しないで、サービスプログラムにそのまま渡されます。応答の長さには、クライアント UAP で設定した長さが渡されます。

ERRTRNR による RPC 応答メッセージ送信機能を使用している場合で、拡張トランザクション種別 (データ名 Z) が EERPC_SERVICE_TRN_EX のときは、次のようになります。

- ERRTRNR のサービスプログラムには、次の値がデータ領域に渡されます。
 - 入力パラメタ (データ名 A)
 - 入力パラメタ長 (データ名 B)
 - サービスプログラムの応答 (データ名 C)
 - サービスプログラムの応答の長さ (データ名 D)
 - トランザクションインタフェース情報 (一意名 5 のデータ名、および一意名 6 のデータ名)

- ERRTRNR のサービスプログラムの応答には、サービスプログラムでクライアント UAP が設定したサービスプログラムの応答（データ名 C）、およびサービスプログラムの応答の長さ（データ名 D）が渡されます。

サービスプログラムに値が渡されるデータ領域を次の表に示します。

表 5-1 サービスプログラムに値が渡されるデータ領域

引数	トランザクション種別												
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
入力パラメタ	×	×	○	○	○	○	○	○	○	○	×	×	×
入力パラメタ長	×	×	○	○	○	○	○	○	○	○	×	×	×
サービス関数の応答	×	×	×	×	×	×	×	×	○※	×	×	×	×
応答の長さ	×	×	○	○	○	○	×	×	○	×	×	×	×
トランザクションインタフェース情報	○	○	○	○	○	○	○	○	○	○	○	○	○

(凡例)

- ：参照できます。
- ×

注※

ERRTRNR による RPC 応答メッセージ送信機能を使用している場合で、拡張トランザクション種別が EERPC_SERVICE_TRN_EX のときだけ参照できます。

サービスプログラムには、次の値をデータ領域に設定します。

- サービスプログラムの応答（データ名 C）
- サービスプログラムの応答の長さ（データ名 D）

サービスプログラムは、データ名 C に応答を設定して、その長さをデータ名 D に設定してからリターンしてください。

サービスのクライアント UAP への応答は、サービスプログラムがコミットしたかロールバックしたかには無関係に送信されます。必要ならばサービスプログラムでクライアント UAP にエラー発生を知らせる応答を作成してください。

クライアント UAP から値が渡されるデータ領域

●データ名 A

クライアント UAP で設定した、入力パラメタが渡されます。

ただし、トランザクション種別が E4, ES, または TM の場合は次の表に示す値が渡されます。

表 5-2 トランザクション種別が E4, ES, または TM の場合に渡される値 (データ名 A)

トランザクション種別	渡される値
E4	CBLEETIM('EXECAP ')が起動するトランザクションに渡すデータ (データ名 J) に設定した値
ES	トランザクションと同期して送信する非同期型のメッセージ送信 API (CBLEERPC('CMTSND ')など) の入力パラメタ (データ名 H) に設定した値
TM	CBLEETIM('EXECAP ')が起動するトランザクションに渡すデータ (データ名 J) に設定した値

●データ名 B

クライアント UAP で設定した、入力パラメタの長さが渡されます。

ただし、トランザクション種別が E4, ES, または TM の場合は次の表に示す値が渡されます。

表 5-3 トランザクション種別が E4, ES, または TM の場合に渡される値 (データ名 B)

トランザクション種別	渡される値
E4	CBLEETIM('EXECAP ')が起動するトランザクションに渡すデータ (データ名 I) に設定した値
ES	トランザクションと同期して送信する非同期型のメッセージ送信 API (CBLEERPC('CMTSND ')など) の入力パラメタ (データ名 G) に設定した値
TM	CBLEETIM('EXECAP ')が起動するトランザクションに渡すデータ (データ名 I) に設定した値

●データ名 D

クライアント UAP で設定した、応答の長さが渡されます。

TP1/EE から値が返されるデータ領域

●一意名 5

トランザクションインタフェース情報が返されます。詳細については、この章の「[トランザクションインタフェース情報](#)」を参照してください。

ただし、C 言語と次の差分があります。

- データ名 O
ユーザ IFA エリアのサイズを設定します (単位: バイト)。
ユーザ IFA エリアのサイズが 2GB 以上の場合、データ名 AM の値を参照してください。
- データ名 Q
ユーザ SAT エリアのサイズを設定します (単位: バイト)。
ユーザ SAT エリアのサイズが 2GB 以上の場合、データ名 AN の値を参照してください。

●一意名 7

XTC 拡張トランザクションインタフェース情報が返されます。詳細については、この章の「[トランザクションインタフェース情報](#)」を参照してください。

UAP で値を設定するデータ領域

●データ名 C

サービスプログラムからの応答を設定します。サービスプログラムの処理では、データ名 D に処理結果を設定してからリターンしてください。

●データ名 D

サービスプログラムからの、実際の応答の長さを設定します。クライアント UAP から渡されたデータ名 D の値以下の数値を設定してください。

サービスプログラムの処理での注意

- クライアントから応答不要で呼び出されるサービスプログラムでは、サービス関数の応答（データ名 C）とサービス関数の応答の長さ（データ名 D）は参照できません。クライアントから応答不要で呼び出されるサービスプログラムを次に示します。
 - 非応答型 RPC で呼び出される RPC のサービスプログラム
 - DB キューのサービスプログラム
 - オンラインバッチのサービスプログラム
- サービスプログラムが、COBOL の初期化プログラム（プログラムの見出し部に INITIAL 句を指定）である場合、データ項目などの値は、サービス要求 1 回ごとに初期化されます。プログラムの見出し部に INITIAL 句を指定しない場合、データ項目には前回のサービスプログラムが設定した値が残っているので、必要であれば初期化してから使ってください。プロセス関連定義の `cobol_uap_cblend` オペランド設定値が Y の場合、サービスプログラムの実行前に COBOL 実行空間を削除して、毎回新たな COBOL 実行空間でサービスプログラムを実行するため、データ項目などの値はサービス要求 1 回ごとに初期化されます。
- サービスプログラムからプログラムの処理を抜ける命令文は使わないでください。また、子プロセスを生成するシステムコールをサービスプログラムで使わないでください。このような命令文を使った場合の UAP の動作は保証されません。
- トランザクション情報として設定する次に示すデータ名は、それぞれのサイズで参照してください。
 - サービスグループ名（データ名 G）
 - サービス名（データ名 I）
 - 送信サービスグループ名（データ名 AD）
 - 送信サービス名（データ名 AF）

- DB キュー機能を使用する場合、DB キュー名（データ名 BB）の文字列の最後は LOW-VALUE です。オンラインバッチ機能を使用する場合、バッチキュー名（データ名 CC）およびロット名（データ名 CD）の文字列の最後は LOW-VALUE です。

トランザクションとサービスプログラムの関係

グローバルトランザクションのそれぞれのサービスは、コミットするか、ロールバックとなるかのどちらかが保証されています。トランザクションブランチとして実行されるサービスプログラムが EXIT PROGRAM で終わることで、そのトランザクションブランチを正常終了させることを要求したことになります。

ステータスコード

ステータスコードはありません。TP1/EE でもステータスコードを参照しません。サービスプログラムから -1 をリターンしても、ロールバックを要求したことにはなりません。

トランザクションインタフェース情報

SPP のデータ領域には、TP1/EE からトランザクションインタフェース情報が渡されます。TP1/EE から SPP に渡されるトランザクションインタフェース情報を次の表に示します。

表 5-4 トランザクションインタフェース情報

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	M E	M N	E1	E2	E3	E4	ES	ER	T M	UI	M V	RL	U T	C U	JI	BS
情報長 (データ名 E)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
サービスグループ名長 (データ名 F)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○※1
サービスグループ名 (データ名 G)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○※1
サービス名長 (データ名 H)	×	×	○	○	○	○	○	×	○	○	×	○	○	×	○	×	×
サービス名 (データ名 I)	×	×	○	○	○	○	○	×	○	○	×	○	○	×	○	×	×
トランザクション種別 (データ名 J)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
自動閉塞有無 (データ名 K)	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×	×
コミット有無 (データ名 L)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
処理スレッド通番 (データ名 M)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
エラー要因コード (データ名 N)	×	×	×	×	×	○	×	○	○	×	×	×	×	×	×	×	×
ユーザ IFA エリアサイズ (データ名 O)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ IFA エリアポインタ (データ名 P)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ SAT エリアサイズ (データ名 Q)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ユーザ SAT エリアポインタ (データ名 R)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
応答要否 (データ名 S)	×	×	○	×	×	×	×	×	○	×	×	×	○	×	○	×	×

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL	UT	CU	JI	BS
メッセージ状態 (データ名 T)	×	×	○	○	○	○	○	○	○	○	×	×	○	×	○	×	×
プロセス起動要因 (データ名 U)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
前回プロセス終了要因 (データ名 V)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
プロセス終了要因 (データ名 W)	×	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
スレッドダウン種別 (データ名 X)	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×	×
異常発生トランザクション種別 (データ名 Y)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
拡張トランザクション種別 (データ名 Z)	×	×	○	×	×	○	×	×	○	×	×	×	×	×	○	×	×
リソースマネージャ数 (データ名 AA)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
リソースマネージャ接続状態 (データ名 AB)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×
送信サービスグループ名長 (データ名 AC)	×	×	×	×	×	×	×	○	×	×	×	○	×	×	×	×	×
送信サービスグループ名 (データ名 AD)	×	×	×	×	×	×	×	○	×	×	×	○	×	×	×	×	×
送信サービス名長 (データ名 AE)	×	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×
送信サービス名 (データ名 AF)	×	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×
異常発生処理スレッド通番 (データ名 AG)	×	×	×	×	×	○	×	×	○	×	×	×	×	×	×	×	×
拡張トランザクションインタフェース情報 (データ名 AH)	×	×	△	×	×	△	×	×	△	×	×	×	×	×	△	×	×
XTC 拡張トランザクションインタフェース情報 (データ名 AI) ※ 2	○	○	○	○	○	○	○	○	○	○	○	○	△	×	×	×	×

トランザクションインタフェース情報 (変数名)	トランザクション種別																
	MI	ME	MN	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL	UT	CU	JI	BS
MCP 拡張トランザクションインタフェース情報 (データ名 AJ)	×	○ ※3	×	○ ※3	○ ※3	○ ※3	×	○ ※3	○ ※3	×	×	×	○	×	×	×	×
メッセージ種別 (データ名 AK)	×	×	○	○	○	○	×	○	○	×	×	×	×	×	○	×	×
エラートランザクション要因 (データ名 AL)	×	×	×	×	×	×	○	×	×	×	×	×	×	×	×	×	×
拡張ユーザ IFA エリアサイズ (データ名 AM)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
拡張ユーザ SAT エリアサイズ (データ名 AN)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
トランザクション起動要因 (データ名 AO)	×	×	×	○	×	×	×	×	×	×	×	×	×	○	○	○	×
サーバ再起動回数 (データ名 AP)	○	×	×	×	×	×	×	×	×	×	×	×	×	×	×	○	×
トランザクション起動時刻 (データ名 AQ)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
送信元のノード識別子 (データ名 AR)	×	×	○	○	○	○	×	×	○	×	×	×	×	×	×	×	×
リトライ回数 (データ名 AX)	×	×	○	×	×	×	×	×	×	○	×	×	×	×	○	×	×
トランザクションレベル名長 (データ名 AS)	×	×	○	○	×	○	×	×	×	○	×	×	○	×	○	×	×
トランザクションレベル名 (データ名 AT)	×	×	○	○	×	○	×	×	×	○	×	×	○	×	○	×	×
トランザクション引き継ぎ情報 (データ名 AY)	×	×	○	×	×	○	×	×	×	○	×	×	×	×	○	×	×

(凡例)

○：参照できます。

△：拡張トランザクション種別が NULL 以外の場合は参照できます。

×：参照できません。

注※1

サービスグループ名の代わりにジョブ名の情報が設定されます。

注※2

XTC を使用する場合にだけ有効です。

注※3

メッセージ種別（データ名 AK）が X'05'または X'06'の場合にだけ参照できます。

上記の表のトランザクションインタフェース情報について、次に説明します。なお、TP1/FSP を使用する場合は、ここで説明していない内容が設定されることがあります。詳細については、TP1/FSP の関連ドキュメントを参照してください。

●情報長

トランザクションインタフェース情報の長さが設定されます（単位：バイト）。

情報長（LL）	その他のトランザクションインタフェース情報
LL	

●サービスグループ名長

サービスグループ名の長さが設定されます（単位：バイト）。

●サービスグループ名

該当するトランザクションのサービスグループ名が設定されます。

●サービス名長

サービス名の長さが設定されます（単位：バイト）。

●サービス名

サービス名が設定されます。

●トランザクション種別

トランザクション種別が設定されます。

'MI'
初期化トランザクション

'ME'
終了トランザクション

'MN'
サービストランザクション

'E1'
エラートランザクション（ERRTRN1）

'E2'
エラートランザクション（ERRTRN2）

'E3'

エラートランザクション (ERRTRN3)

'E4'

エラートランザクション (ERRTRN4)

'ES'

エラートランザクション (ERRTRNS)

'ER'

エラートランザクション (ERRTRNR)

'TM'

タイマトランザクション

'UI'

実行系連絡トランザクション

'MV'

イベント通知トランザクション

'RL'

MCP 後処理トランザクション

●自動閉塞有無

サービスの自動閉塞の有無が設定されます。

X'01'

自動閉塞有

X'00'

自動閉塞無

●コミット有無

エラーになったトランザクションの同期点の有無が設定されます。

X'00'

コミット無

X'01'

コミット有

エラーになったサービス処理で一度でも同期点を取得している場合は、コミット有が設定されます。

●処理スレッド通番

該当するトランザクションのスレッド通番が設定されます。

●エラー要因コード

エラーの要因コードが設定されます。

スレッドダウンの場合は、その詳細が設定されます。この場合、スレッドダウン種別によって設定情報が異なります。

同期シグナルの場合

シグナル番号が設定されます。

トランザクションタイマの場合

値は設定されません。

ロールバック要求（スレッド強制終了指示）の場合

CBLEETRN('C-ROLL ')のデータ名 D の値が設定されます。

DB キューの交替用メッセージ表書き込み機能使用時のロールバック要求（スレッド強制終了指示）の場合
次の情報が設定されます。

- EEDBQ_ERRINF_WRITE：書き込み処理でエラー(5)
- EEDBQ_ERRINF_READ：読み出し処理でエラー(6)

上記以外の場合

値は設定されません。

トランザクション種別がエラートランザクション（EERTNR）の場合は、次に示す情報が設定されます。

1

UAP リターン後、トランザクションはロールバックしました。

2

UAP リターン後、トランザクションはヒューリスティック決着しました。

3

UAP リターン後の同期点処理で通信障害が発生したため、トランザクションは未決着です。

4

UAP リターン後の同期点処理でリソースマネージャ障害が発生したため、トランザクションは未決着です。

10

UAP リターン後、XDB 前処理エラーによってロールバックしました。

11

UAP リターン後、XDB 処理エラーによってロールバックしました。

20

UAP リターン後、トランザクション同期のメッセージの送信失敗によってロールバックしました。

トランザクション種別がエラートランザクション（EERTNS）の場合は、次に示す情報が設定されます。

20

UAP リターン後、トランザクション同期のメッセージの送信失敗によってロールバックしました。

21

UAP リターン後、トランザクション非同期のメッセージの送信失敗によってロールバックしました。

●ユーザ IFA エリアサイズ

ユーザ IFA エリアのサイズが設定されます（単位：バイト）。

64 ビットアーキテクチャの場合、ユーザ IFA エリアのサイズが 4 ギガバイト以上のときは 0 が設定されます。この場合、データ名 AM を参照してユーザ IFA エリアのサイズを取得してください。ユーザ IFA エリアのサイズが 4 ギガバイト未満の場合でも、データ名 AM を参照してユーザ IFA エリアのサイズを取得できます。

●ユーザ IFA エリアポインタ

ユーザ IFA エリアのポインタが設定されます。ユーザ IFA エリアサイズが 0 の場合は、値を参照できません。

●ユーザ SAT エリアサイズ

ユーザ SAT エリアのサイズが設定されます（単位：バイト）。

64 ビットアーキテクチャの場合、ユーザ SAT エリアのサイズが 4 ギガバイト以上のときは 0 が設定されます。この場合、データ名 AN を参照してユーザ SAT エリアのサイズを取得してください。ユーザ SAT エリアのサイズが 4 ギガバイト未満の場合でも、データ名 AN を参照してユーザ SAT エリアのサイズを取得できます。

●ユーザ SAT エリアポインタ

ユーザ SAT エリアのポインタが設定されます。ユーザ SAT エリアサイズが 0 の場合は、値を参照できません。

●応答要否

クライアント UAP からの RPC 要求に対する応答要否が設定されます。

'Y'

応答要

'N'

応答不要

●メッセージ状態

引き渡しメッセージの状態が設定されます。

'N'

正常

'O'
オーバーフロー

オーバーフローの場合、SPP に渡される入力パラメタにはメッセージの一部しか設定されません。

●プロセス起動要因

TP1/EE プロセス起動時の要因が設定されます。

'N'
通常起動

'R'
再起動

●前回プロセス終了要因

前回の TP1/EE プロセス終了時の要因が設定されます。

'N'
正常終了

'A'
計画停止 A

'B'
計画停止 B

'E'
異常終了

●プロセス終了要因

TP1/EE プロセス終了時の要因が設定されます。

'N'
正常終了

'A'
計画停止 A

'B'
計画停止 B

'C'
実行系孤立モード A

'D'
実行系孤立モード B

●スレッドダウン種別

スレッドダウン種別が設定されます。

X'01'

同期シグナル

X'02'

トランザクションタイマ

X'04'

ロールバック要求（スレッド強制終了指示）

X'08'

CBLEETRN('C-ROLL ')または CBLEETRN('ROLLMARK')の呼び出し回数の上限値オーバ

X'09'

スレッドダウン指示による終了

リトライロールバック前処理 UOC でリターン値 3, 4 設定時

X'20'

CBLEETRN('C-COMMIT')の呼び出し回数の上限値オーバ

X'44'

XDB の障害

X'40'

上記以外

●異常発生トランザクション種別

異常が発生したトランザクション（要求元トランザクション）の種別が設定されます。

X'01'

サービストランザクション

X'02'

タイマトランザクション

X'04'

再起動トランザクション

●拡張トランザクション種別

拡張トランザクション種別が設定されます。

'S'

拡張トランザクションインタフェース情報（DB キュー機能およびオンラインバッチ機能を使用しない場合）

'D'

拡張トランザクションインタフェース情報（DB キュー機能を使用する場合）

'O'

拡張トランザクションインタフェース情報（オンラインバッチ機能を使用する場合）

拡張トランザクション種別が NULL 以外の場合は、トランザクションインタフェース情報の一部として拡張トランザクションインタフェース情報が設定されます。

●リソースマネージャ数

接続しているリソースマネージャの数が設定されます。

●リソースマネージャ接続状態

リソースマネージャとの接続状態が設定されます。

0

リソースマネージャ接続正常

1

リソースマネージャ接続失敗によるトランザクション縮退起動

複数のリソースマネージャと接続している場合、一部のリソースマネージャでの接続失敗も含みます。

●送信サービスグループ名長

トランザクション種別が EERPC_TRNKIND_ES の場合

CBLEERPC('CMTSND ')関数, CBLEEMCH('CMTSND ')関数, または
CBLEEMCH('CMTSND SY')関数で指定したサービスグループ名の長さが設定されます（単位：バイト）。

トランザクション種別が EERPC_TRNKIND_MV の場合

サービスグループレベルで出力キュー（OTQ）が自動閉塞していたサービスグループ名の長さが設定
されます（単位：バイト）。

●送信サービスグループ名

トランザクション種別が EERPC_TRNKIND_ES の場合

CBLEERPC('CMTSND '), CBLEEMCH('CMTSND '), または CBLEEMCH('CMTSND SY')で設定
したサービスグループ名が設定されます。

トランザクション種別が EERPC_TRNKIND_MV の場合

サービスグループレベルで出力キュー（OTQ）が自動閉塞していたサービスグループ名が設定されます。

●送信サービス名長

CBLEERPC('CMTSND ')関数, CBLEEMCH('CMTSND ')関数, または CBLEEMCH('CMTSND SY')
関数で指定したサービス名の長さが設定されます（単位：バイト）。

●送信サービス名

CBLEERPC('CMTSND ')関数，CBLEEMCH('CMTSND ')関数，またはCBLEEMCH('CMTSND SY')関数で指定したサービス名が設定されます。

●異常発生処理スレッド通番

異常が発生したトランザクション（要求元トランザクション）のスレッド通番が設定されます。

●拡張トランザクションインタフェース情報

拡張トランザクションインタフェース情報が設定されます。拡張トランザクションインタフェース情報は、DB キュー機能およびオンラインバッチ機能を使用するかどうかによって設定される情報が異なります。それぞれの場合について説明します。

DB キュー機能およびオンラインバッチ機能を使用しない場合

DB キュー機能およびオンラインバッチ機能を使用しない場合，トランザクションインタフェース情報の一部として，次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 5-5 拡張トランザクションインタフェース情報（DB キュー機能およびオンラインバッチ機能を使用しない場合）

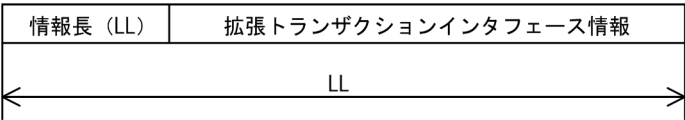
拡張トランザクションインタフェース 情報（変数名）	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長（データ名 DA）	×	×	×	×	×	×	×	×	○	×	×	×	×
サービスプログラム起動時の応答の長さ（データ名 DB）	×	×	×	×	×	×	×	×	○	×	×	×	×

（凡例）

- ：参照できます。
- ×

・情報長

拡張トランザクションインタフェース情報の長さが設定されます（単位：バイト）。



・サービスプログラム起動時の応答の長さ

クライアント UAP で設定したサービスプログラム起動時の応答の長さ（データ名 D）が設定されます。

DB キュー機能を使用する場合

DB キュー機能を使用する場合，トランザクションインタフェース情報の一部として，次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 5-6 拡張トランザクションインタフェース情報 (DB キュー機能を使用する場合)

拡張トランザクションインタフェース 情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長 (データ名 BA)	×	×	○	×	×	○	×	×	○	×	×	×	×
読み込み DB キュー名 (データ名 BB)	×	×	○	×	×	○	×	×	○	×	×	×	×
通番 (読み込み) (データ名 BC)	×	×	○	×	×	○	×	×	×	×	×	×	×
ロールバック回数 (データ名 BD)	×	×	○	×	×	○	×	×	○	×	×	×	×

(凡例)

○：参照できます。

×

・ 情報長

拡張トランザクションインタフェース情報の長さが設定されます (単位：バイト)。



・ 読み込み DB キュー名

読み込み時の DB キュー名が設定されます。

・ 通番

読み込みメッセージの通番が設定されます。

・ ロールバック回数

同じメッセージに対して CBLEETRN('C-ROLL ')および CBLEETRN('ROLLMARK')でロールバックした回数が設定されます。65535 を超えた場合は 65535 が設定されます。

同一メッセージ処理の連続ロールバック監視回数 (DB キュー機能関連定義の dbq_rollback_retry_count オペランドの指定値) を超えたためにスケジュールを停止した場合、ロールバック回数は 0 に戻ります。

オンラインバッチ機能を使用する場合

オンラインバッチ機能を使用する場合、トランザクションインタフェース情報の一部として、次の表に示す拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 5-7 拡張トランザクションインタフェース情報 (オンラインバッチ機能を使用する場合)

拡張トランザクションインタフェース 情報 (変数名)	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
情報長 (データ名 CA)	×	×	○	×	×	○	×	×	○	×	×	×	×

拡張トランザクションインタフェース 情報（変数名）	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
トランザクション種別（データ名 CB）	×	×	○	×	×	○	×	×	○	×	×	×	×
バッチキュー名（データ名 CC）	×	×	○	×	×	○	×	×	○	×	×	×	×
ロット名（データ名 CD）	×	×	○	×	×	○	×	×	○	×	×	×	×
引き継ぎ情報のサイズ（データ名 CE）	×	×	○	×	×	×	×	×	×	×	×	×	×
引き継ぎ情報のアドレス（データ名 CF）	×	×	○	×	×	×	×	×	×	×	×	×	×
オンラインバッチ処理形態（データ名 CG）	×	×	○	×	×	○	×	×	○	×	×	×	×
該当するロットの全メッセージ数 （データ名 CH）	×	×	△	×	×	△	×	×	△	×	×	×	×
読み込んだメッセージの通番（データ名 CI）	×	×	△	×	×	△	×	×	×	×	×	×	×
ロット終了要因コード（データ名 CJ）	×	×	△	×	×	×	×	×	×	×	×	×	×
終了時のステータスコード（データ名 CK）	×	×	△	×	×	×	×	×	×	×	×	×	×
読み出し済みメッセージ数（データ名 CL）	×	×	△	×	×	△	×	×	△	×	×	×	×
スキップメッセージ数（データ名 CM）	×	×	△	×	×	△	×	×	△	×	×	×	×
未読み出しメッセージ数（データ名 CN）	×	×	△	×	×	△	×	×	△	×	×	×	×
メッセージ長情報（データ名 CO）	×	×	△	×	×	△	×	×	△ ※	×	×	×	×

（凡例）

○：参照できます。

△：トランザクションの種別や、使用するオンラインバッチ処理の処理方法によって異なります。
詳細については、表 5-6 を参照してください。

×：参照できません。

注※

サービスプログラムの入力パラメタ長（データ名 B）が 0 以外の場合だけ参照できます。

表 5-8 拡張トランザクションインタフェース情報の詳細（オンラインバッチ機能を使用する場合）

トランザクション種別	バッチ処理トランザクション		バッチ終了トランザクション	
オンラインバッチ処理形態	データ型	イベント型	データ型	イベント型
該当するロットの全メッセージ数	○	×	○	×
読み込んだメッセージの通番	○	×	×	×
ロット終了要因コード	×	×	○	○
終了時のステータスコード	×	×	○	○
読み出し済みメッセージ数	×	×	○	×
スキップメッセージ数	×	×	○	×
未読み出しメッセージ数	×	×	○	×
メッセージ長情報	○	×	×	×

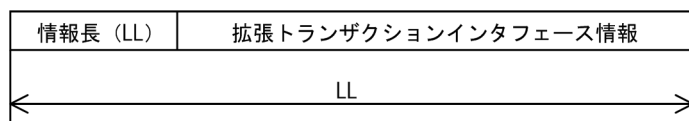
（凡例）

○：参照できます。

×：参照できません。

- 情報長

拡張トランザクションインタフェース情報の長さが設定されます（単位：バイト）。



- トランザクション種別

トランザクション種別が設定されます。

1

バッチ処理トランザクション

2

バッチ終了トランザクション

- バッチキュー名

バッチキュー名が設定されます。

- ロット名

ロット名が設定されます。

- 引き継ぎ情報のサイズ

CBLEEDBQ('OBSSTART'), および CBLEEDBQ('OBSCDT ')で設定したロットの引き継ぎ情報のサイズが設定されます。

- 引き継ぎ情報のアドレス

CBLEEDBQ('OBSSTART'), および CBLEEDBQ('OBSCDT ')で設定したロットの引き継ぎ情報のエリアアドレスが設定されます。

- オンラインバッチ処理形態

実行中のロットのオンラインバッチ処理形態が設定されます。

1

データ型オンラインバッチ処理

2

イベント型オンラインバッチ処理

- 該当するロットの全メッセージ数

該当するロットの全メッセージ数が設定されます。該当するロットのバッチ処理トランザクションには、同じ値が設定されます。

回復処理の失敗による中断、および回復処理の失敗による中断の強制終了要求の場合は、設定されません。

- 読み込んだメッセージの通番

該当するロットで読み出したメッセージ通番が設定されます。

- ロット終了要因コード

該当するロットの終了要因が設定されます。ユーザ指定による処理の中断と、障害による処理の中断の両方が発生した場合は、障害による処理の中断が優先して設定されます。

0

正常終了

1

ユーザ指定による中断

2

障害による中断

3

強制終了要求

4

回復処理の失敗による中断

5

回復処理の失敗による中断中の強制終了要求

- 終了時のステータスコード（理由コード）

処理終了時のステータスコード（理由コード）が設定されます。

障害または回復処理の失敗による処理の中断の場合に設定されます。強制終了要求および回復処理の失敗による中断中の強制終了要求の場合は、以前に設定された値が引き継がれます。複数の障害が重なった場合は、ステータスコードの値が大きい方の障害が優先して設定されます。そのほかの場合は NULL が設定されます。

4

データフォーマット不正

8

オンラインバッチ機能処理中のエラー

10

次に示す要因以外でのロールバック

- ・ UAP でのデータ名 C に VALUE 'KEEP'を設定した CBLEETRN('C-ROLL ')の呼び出し
- ・ UAP での CBLEETRN('ROLLMARK')の呼び出し
- ・ データフォーマット不正
- ・ オンラインバッチ機能処理中のエラー
- ・ DB キューのリソースマネージャへのアクセスエラー
- ・ スレッドダウン
- ・ プロセスダウン

12

DB キューのリソースマネージャへのアクセスエラー

15

スレッドダウン

16

プロセスダウン

- 読み出し済みメッセージ数

該当するロットの読み出し済みメッセージ数が設定されます。

回復処理の失敗による中断，および回復処理の失敗による中断の強制終了要求の場合は，設定されません。

- スキップメッセージ数

該当するロットで読み出しをスキップしたメッセージ数が設定されます。

回復処理の失敗による中断，および回復処理の失敗による中断の強制終了要求の場合は，設定されません。

- 未読み出しメッセージ数

該当するロットの未読み出しメッセージ数が設定されます。

回復処理の失敗による中断，および回復処理の失敗による中断の強制終了要求の場合は，設定されません。

- メッセージ長情報

メッセージの前にユーザが設定するメッセージ長と，実際に入力したメッセージ長が一致しているかどうかを確認するための情報が設定されます。

'E'

一致しているか，またはオーバフローしています。オーバフローしているかどうかは，トランザクションインタフェース情報のメッセージ状態を参照して確認してください。

'D'

ユーザが設定するメッセージ長が実際に入力したデータ長より大きいため、一致していません。実際に入力したデータ長によって、入力パラメタ（データ名 A）にメッセージが設定されます。

⌈

ユーザが設定するメッセージ長が実際に入力したデータ長より小さいため、一致していません。メッセージの前に設定されたメッセージ長によって、入力パラメタ（データ名 A）にメッセージが設定されます。

●XTC 拡張トランザクションインタフェース情報

XTC 拡張トランザクションインタフェース情報が設定されます。次の表に示す XTC 拡張トランザクションインタフェース情報が TP1/EE から渡されます。

表 5-9 XTC 拡張トランザクションインタフェース情報

XTC 拡張トランザクションインタフェース情報（変数名）	トランザクション種別												
	MI	ME	M N	E1	E2	E3	E4	ES	ER	TM	UI	MV	RL
系情報（データ名 EB）	○	○	×	×	×	×	×	×	×	×	×	×	×
系切り替え前の実行系の状態（データ名 EC）	×	×	×	×	×	×	×	×	×	×	○	×	×
出力キュー（OTQ）自動閉塞の有無（データ名 ED）	×	×	×	×	×	×	×	○	×	×	×	×	×
出力キュー（OTQ）自動閉塞レベル（データ名 EE）	×	×	×	×	×	×	×	○	×	×	×	×	×
出力キュー（OTQ）通番（データ名 EF）	×	×	×	×	×	×	×	○	×	×	×	×	×
エラー要因詳細（データ名 EG）	×	×	×	×	×	×	×	○	○	×	×	×	×
入力キュー（ITQ）通番（データ名 EH）	×	×	○	×	×	×	×	×	×	○	×	×	×
ロールバックリトライ回数（データ名 EI）	×	×	○	×	×	×	×	×	×	○	×	×	×
系切り替え後の実行系の状態（データ名 EJ）	×	×	×	×	×	×	×	×	×	×	○	×	×

（凡例）

- ：参照できます。
- ×

×：参照できません。

• 系情報

トランザクションを起動した XTC の系が設定されます。

'O'

実行系

'S'

待機系

- 系切り替え前の実行系の状態

系切り替えが発生したときの、系切り替え前に実行系だった XTC の状態が設定されます。

X'01'

オンライン中

X'02'

終了処理中、かつ終了トランザクション起動前、または終了トランザクション起動中

- 出力キュー（OTQ）自動閉塞の有無

メッセージ送信失敗による出力キューの自動閉塞の有無が設定されます。

X'01'

自動閉塞有り

X'00'

自動閉塞無し

- 出力キュー（OTQ）自動閉塞レベル

メッセージ送信失敗による出力キューの自動閉塞のレベルが設定されます。

X'00'

自動閉塞無し

X'01'

サービスグループレベル

X'02'

サービスレベル

- 出力キュー（OTQ）通番

トランザクション非同期のメッセージ送信に失敗したときのメッセージの出力キューの通番が設定されます。

- エラー要因詳細

エラー詳細情報を設定します。

トランザクション種別が ES、エラー要因コードが EERPC_ERRINF_SENDSYNC_ERR、または EERPC_ERRINF_SEND_ERR の場合

1

送信障害（パケットサイズ不正などの RPC 関連定義の不正）

2

送信障害（リソース不足）

- 4
送信障害（送信先受信不可状態）
- 3
送信障害（上記以外）
- 16
送達確認待ちタイムアウト
- 17
送達確認待ち受信障害
- 32
送達確認エラーメッセージ受信（リソース不足）
- 33
送達確認エラーメッセージ受信（CL 同期エラー）
- 34
送達確認エラーメッセージ受信（送信先サービス閉塞），または自プロセスあて送信時のサービス閉塞
- 64
送達確認エラーメッセージ受信（送信先サービスグループ名不正）
- 65
送達確認エラーメッセージ受信（送信先サービス名不正）
- 66
送達確認エラーメッセージ受信（メッセージ送信形態不正／オプション不正）
- 128
送達確認エラーメッセージ受信（上記以外）
- 256
出力キューの閉塞中
- 512
処理キュー登録失敗

トランザクション種別が ER，エラー要因コードが EERPC_ERRINF_XDB_BEFORE_ERR の場合

EERPC_ERRINF_XDB_ERR_RESOURCE

XDB コミット処理時のバッファ不足

トランザクション種別が ER，エラー要因コードが EERPC_ERRINF_XDB_ERR の場合

エラー要因詳細情報（詳細については，マニュアル「TP1/EE/Extended Data Cache 使用の手引」を参照）

- 入力キュー（ITQ）通番

入力メッセージの入力キューの通番が設定されます。

- ロールバックリトライ回数

トランザクションがロールバックして同じメッセージを入力するトランザクションをリトライした回数です。回数が 65535 を超えた場合は 65535 が設定されます。XTC のロールバックリトライ回数監視によってサービスが閉塞した場合、またはメッセージが読み出し済みとなった場合には 0 に戻ります。

- 系切り替え後の実行系の状態

系切り替えが発生したときの、系切り替え後の XTC の状態が設定されます。

'N'

オンライン

'A'

実行系孤立モード A

'B'

実行系孤立モード B

●MCP 拡張トランザクションインタフェース情報

マニュアル「TP1/EE/Message Control Extension 使用の手引」を参照してください。

●メッセージ種別

トランザクション起動契機となった入力メッセージの種別が設定されます。

X'00'

RPC メッセージ (TCP/IP)

X'01'

タイマトランザクションメッセージ

X'02'

RAP メッセージ (TCP/IP)

X'03'

一方送信メッセージ (UDP)

X'04'

RPC メッセージ (UDP)

X'05'

MCP メッセージ (TCP/IP)

X'06'

MCP メッセージ (UDP)

X'11'

DB キューメッセージ

X'12'

オンラインバッチメッセージ

●エラーランザクション要因

ERRTRN4 を起動した要因が設定されます。

X'01'

サービス閉塞による起動

X'02'

オンライン終了による起動

●拡張ユーザ IFA エリアサイズ

拡張ユーザ IFA エリアのサイズが設定されます（単位：バイト）。

●拡張ユーザ SAT エリアサイズ

拡張ユーザ SAT エリアのサイズが設定されます（単位：バイト）。

●ランザクション起動要因

TP1/FSP の関連ドキュメントを参照してください。

●サーバ再起動回数

TP1/FSP の関連ドキュメントを参照してください。

●ランザクション起動時刻

ランザクションを開始した時刻が、YYYYMMDDHHMMSS の型式の文字列で設定されます。文字列の末尾に LOW-VALUE は付きません。

●送信元のノード識別子

TP1/FSP の関連ドキュメントを参照してください。

●リトライ回数

TP1/FSP の関連ドキュメントを参照してください。

●ランザクションレベル名長

TP1/FSP の関連ドキュメントを参照してください。

●ランザクションレベル名

TP1/FSP の関連ドキュメントを参照してください。

●ランザクション引き継ぎ情報

TP1/FSP の関連ドキュメントを参照してください。

システム運用の管理 (CBLEEADM)

UAP から TP1/EE システムの各種機能を使うプログラムについて説明します。システム運用の管理の COBOL-UAP 作成用プログラムを次に示します。

- CBLEEADM('COMMAND ') – 運用コマンドの実行

CBLEEADM('COMMAND ')

名称

運用コマンドの実行

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEADM' USING 一意名1 一意名2 一意名3 一意名4
```

DATA DIVISION の指定

32 ビットアーキテクチャの場合

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'COMMAND'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC S9(9) COMP.  
01 一意名2.  
  02 データ名E PIC 9(9) COMP.  
  02 データ名G PIC X(n).  
01 一意名3.  
  02 データ名H PIC 9(9) COMP.  
  02 データ名J PIC X(n).  
01 一意名4.  
  02 データ名K PIC 9(9) COMP.  
  02 データ名M ADDRESS.
```

64 ビットアーキテクチャの場合

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'COMMAND'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC S9(9) COMP.  
01 一意名2.  
  02 データ名E PIC 9(9) COMP.  
  02 データ名G PIC X(n).  
01 一意名3.  
  02 データ名H PIC 9(9) COMP.  
  02 データ名J PIC X(n).  
01 一意名4.  
  02 データ名K PIC 9(9) COMP.  
  02 FILLER PIC X(4).  
  02 データ名M ADDRESS.
```

機能

オンライン中にコマンドを入力したときと同様に、UAP からコマンドを実行します。コマンドの処理が完了するまで待ち、コマンドの実行結果を戻します。コマンドの処理が終了すると、標準出力情報と標準エラー出力情報が返ります。

コマンドの実行形式ファイルを格納したディレクトリをサーチパスに追加しておいてください。

UAP で値を設定するデータ領域

●データ名 A

コマンドの実行を示す要求コードを「VALUE 'COMMAND△」と設定します。

●データ名 Z

0 を設定します。

●データ名 D

応答監視時間を設定します（単位：秒）。指定できる範囲は-1、0～65535 です。

-1 を指定した場合、プロセス関連定義の call_cmd_time オペランドで指定した値が応答監視時間になります。0 を設定した場合は、時間監視をしません。

●データ名 E

データ名 G に設定したコマンドの文字列の長さを設定します（単位：バイト）。2047 を超えて指定できません。

●データ名 G

実行する運用コマンドの文字列を設定します。指定できる文字列の最大長は 2048 バイトです（末尾の空白を含む）。

●データ名 H

コマンドの実行結果として標準出力に出力する文字列（データ名 J に返される値）の長さに 1 を加えた値を設定します（単位：バイト）。指定できる値は 1～32768 です。

データ名 H の値は、UAP から実行するコマンドに応じて決めてください。出力結果を受け取る必要がない場合は、1 を設定してください。

処理終了後は、データ名 J に格納された実際の文字列の長さが返されます。データ名 H 自身の長さは含みません。

●データ名 J

コマンドの実行結果として標準出力に出力する文字列を受け取る領域です。UAP で値を設定しておく必要はありませんが、事前にこのデータ名 J が指すアドレスから、（データ名 H の値＋データ名 K の値）分の

領域を確保しておいてください（単位：バイト）。出力結果を受け取る必要がない場合でも、この長さの領域を確保してください。

●データ名 K

コマンドの実行結果として標準エラー出力に出力する文字列（データ名 M に返される値）の長さに 13 を加えた値を設定します（単位：バイト）。指定できる値は 13～32780 です。

データ名 K の値は、UAP から実行するコマンドに応じて決めてください。出力結果を受け取る必要がない場合は、13 を設定してください。

処理終了後は、データ名 M に格納した実際の文字列の長さが返されます。データ名 K 自身の長さは含みません。

TP1/EE から値が返されるデータ領域

データ名 B 以外を参照できるのは、ステータスコードが、00000, 01811, 01813, 01814, または 01815 のときだけです。ほかのステータスコードのときは、参照できません。

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 C

実行したコマンドが、正常終了したか異常終了したか（シェルの終了コード）※が返されます。コマンドをバックグラウンド実行した場合は、正常終了か異常終了かに関係なく 0 が返されます。

注※

waitpid(2)で指定した形式で、sh(1)の終了ステータスを示します。

●データ名 H

処理終了後に、データ名 J に格納された文字列の長さが返されます（単位：バイト）。データ名 H 自身の長さは含みません。

●データ名 J

コマンドの実行結果として標準出力に出力した文字列が返されます。

文字列の長さは、最大で（CBLEEDM('COMMAND')を呼び出すときにデータ名 H に設定した値－1）です（単位：バイト）。

長さの最大を超える文字列は切り捨てられます。また、パイプの容量を超えた分も切り捨てられます。

●データ名 K

処理終了後に、データ名 M に格納された文字列の長さが返されます（単位：バイト）。データ名 K 自身の長さは含みません。

●データ名 M

コマンドの実行結果として標準エラー出力に出力した文字列を格納した領域の先頭アドレスが返されます。

文字列の長さは、最大で (CBLEEDM('COMMAND ')) を呼び出すときにデータ名 K に設定した値-13) です (単位: バイト)。

長さの最大を超える文字列は切り捨てられます。また、パイプの容量を超えた分も切り捨てられます。

ステータスコード

ステータスコード	意味
00000	シェルの終了コードは 0 (コマンドの実行が正常終了) です。標準出力および標準エラー出力の領域に文字列を格納しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
01811	シェルの終了コードは 0 以外 (コマンドの実行が異常終了) です。標準出力および標準エラー出力のデータを領域に格納しました。
01812	データ名に設定した値が間違っています。要求コード (データ名 A) が間違っている場合も含まれます。
01813	標準出力のデータが、領域に収まりません。
01814	標準エラー出力のデータが、領域に収まりません。
01815	標準出力のデータと標準エラー出力のデータの両方が、領域に収まりません。
01816	システムコール (close, pipe, dup, または read) の呼び出しに失敗しました。
01817	運用コマンド実行サーバ関連の定義が指定されていません。
01818	運用コマンド実行サーバへの通信が失敗しました。
01819	メモリが不足しました。
01820	そのほかのエラーが発生しました。

注意事項

1. このプログラムで実行できる TP1/EE のコマンドは、トランザクション種別によって異なります。詳細は「[2.6 運用コマンド実行機能](#)」を参照してください。
2. サーチパスに指定したディレクトリ間で、コマンド名が重複しないように注意してください。コマンド名が重複している場合、正しいコマンドが実行されないおそれがあります。
3. ファイル操作に関連するコマンドを実行する場合、ファイルの I/O などの処理に時間が掛かるため、実行する場合は十分注意してください。

DB キュー制御 (CBLEEDBQ)

DB キューによるメッセージ管理に使用するプログラムについて説明します。DB キュー制御の COBOL-UAP 作成用プログラムを次に示します。

- CBLEEDBQ('ACSCCTL ') – DB キューの読み出し制御
- CBLEEDBQ('CLEAR ') – DB キューの通番初期化
- CBLEEDBQ('CONCTL ') – DB キューの接続制御
- CBLEEDBQ('INQSRV ') – DB キューサービスの接続照会
- CBLEEDBQ('INQUIRE ') – DB キュー情報の照会
- CBLEEDBQ('INQUIRE2') – DB キュー情報の照会
- CBLEEDBQ('MSGGET ') – メッセージの読み出し
- CBLEEDBQ('MSGPUT ') – メッセージの書き込み
- CBLEEDBQ('MSGREFE2') – メッセージの参照読み出し
- CBLEEDBQ('MSGREFER') – メッセージの参照読み出し
- CBLEEDBQ('OBSCAN ') – オンラインバッチ処理の強制終了
- CBLEEDBQ('OBSCDT ') – 引き継ぎ情報の更新
- CBLEEDBQ('OBSEND ') – オンラインバッチ処理の終了
- CBLEEDBQ('OBSINQ ') – ロットの情報の照会
- CBLEEDBQ('OBSRBSP ') – ロールバック要求時のメッセージの読み出しスキップ
- CBLEEDBQ('OBSRST ') – オンラインバッチ処理の再開始
- CBLEEDBQ('OBSSKIP ') – メッセージの読み出しスキップ
- CBLEEDBQ('OBSSTART') – オンラインバッチ処理の開始
- CBLEEDBQ('OBSSTOP ') – オンラインバッチ処理の中断
- CBLEEDBQ('RBSKIP ') – ロールバック要求時のメッセージの読み出しスキップ
- CBLEEDBQ('SKIP ') – メッセージの読み出しスキップ
- CBLEEDBQ('SRVCTL ') – DB キューサービスの閉塞制御

DB キュー制御の COBOL-UAP 作成用プログラムで利用できる機能を次の表に示します。

表 5-10 DB キュー制御の COBOL-UAP 作成用プログラムで利用できる機能

項番	プログラム名	DB キュー		オンラインバッチ (OBS)
		UAP 間通信	ユーザキュー	
1	CBLEEDBQ('ACSCCTL ')	○	○	×

項番	プログラム名	DB キュー		オンラインバッチ (OBS)
		UAP 間通信	ユーザキュー	
2	CBLEEDBQ('CLEAR ')	○	○	×
3	CBLEEDBQ('CONCTL ')	○	×	×
4	CBLEEDBQ('INQSRV ')	○	×	×
5	CBLEEDBQ('INQUIRE ')	○	×	×
6	CBLEEDBQ('INQUIRE2')	○	○	×
7	CBLEEDBQ('MSGGET ')	×	○	×
8	CBLEEDBQ('MSGPUT ')	○	○	×
9	CBLEEDBQ('MSGREFE2')	○	○	×
10	CBLEEDBQ('MSGREFER')	○	○	×
11	CBLEEDBQ('OBSCAN ')	×	×	○
12	CBLEEDBQ('OBSCDT ')	×	×	○
13	CBLEEDBQ('OBSEND ')	×	×	○
14	CBLEEDBQ('OBSINQ ')	×	×	○
15	CBLEEDBQ('OBSRBSP ')	×	×	○
16	CBLEEDBQ('OBSRST ')	×	×	○
17	CBLEEDBQ('OBSSKIP ')	×	×	○
18	CBLEEDBQ('OBSSTART')	×	×	○
19	CBLEEDBQ('OBSSTOP ')	×	×	○
20	CBLEEDBQ('RBSKIP ')	○	×	×
21	CBLEEDBQ('SKIP ')	○	○	×
22	CBLEEDBQ('SRVCTL ')	○	×	×

(凡例)

○：使用できます。

×

CBLEEDBQ('ACSCCTL ')

名称

DB キューの読み出し制御

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'ACSCCTL '  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(28).
```

機能

DB キューの読み出しを停止、および再開します。

UAP で値を設定するデータ領域

●データ名 A

DB キューの読み出し制御を示す要求コードを「VALUE 'ACSCCTL△△」と設定します。

●データ名 C

オプションを数値で設定します。

0
データ名 E に指定した自サーバ読み出しの DB キューを停止、再開します。

1
データ名 E に指定した自サーバ読み出しまたは他サーバ読み出しの DB キューを停止、再開します。
データ名 E に指定した DB キューが他サーバ読み出しの場合、メッセージを読み出している TP1/EE プロセスが 07-80 以降かつ DB キューのフォーマットタイプが 1 であれば、RPC 通信を行って DB キューの読み出しを停止、再開します。

●データ名 D

要求種別を設定します。

VALUE 'S'

DB キューの読み出しを停止します。

VALUE 'R'

DB キューの読み出しを再開します。

●データ名 E

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。※ ¹
05607	DB キューが使用できない状態です。
05609	リソースマネージャアクセス時にエラーが発生しました。※ ¹
05615	そのほかのエラーが発生しました。
05616	設定された DB キューのタイプは機能対象外です。※ ² 相手読み出し DB キュー
05621	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。※ ¹

ステータスコード	意味
05651	データ名 E に設定した DB キューのフォーマットタイプが不正です。※1
05652	変更要求は実行しましたが、変更要求結果の確認処理が時間切れになりました。※1
05653	変更要求処理に失敗しました。 対象の DB キューを読み出しているサーバが存在しません。※1
05654	変更要求処理に失敗しました。 対象の DB キューを読み出しているサーバが変更要求に対応していません。※1
05655	変更要求処理に失敗しました。※1
05691	変更要求は実行しましたが、変更要求処理の実行後に読み出し先サーバの変更を検知しました。※1※3
05692	変更要求処理を実行しましたが、変更要求処理の結果確認で 05609 のエラーを検知しました。※1※3
05693	変更要求処理を実行しましたが、変更要求処理の結果確認で 05653 のエラーを検知しました。※1※3
05694	変更要求処理を実行しましたが、変更要求処理の結果確認で 05654 のエラーを検知しました。※1※3
05695	変更要求処理を実行しましたが、変更要求処理の結果確認で 05639 のエラーを検知しました。※1※3

注※1

データ名 C に 1 を指定した場合に返されます。

注※2

データ名 C に 0 を指定した場合に返されます。

注※3

DB キューが HiRDB の場合だけ発生するステータスコードです。

注意事項

- DB キューの読み出しトランザクションで、該当する DB キューに対して CBLEEDBQ('ACSCCTL ') を呼び出した場合、読み出しトランザクション完了後に読み出しを停止します。この状態で CBLEEDBQ('INQUIRE ') を呼び出した場合、DB キューの状態は「2」（読み出し停止中）が返されます。
- 次の場合、CBLEEDBQ('ACSCCTL ') はステータスコード「00000」でリターンします。
 - 読み出し停止状態の DB キューに対して読み出し停止を要求した場合。
 - 読み出し開始状態（再開を含む）の DB キューに対して読み出し再開を要求した場合。
- DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した DB キューに対して CBLEEDBQ('ACSCCTL ') を呼び出した場合、DB キューのステータスが他サーバ読み出しのときでも、ステータスコードが「05616」とならないで、「00000」となることがあります。

4. データ名 C に 1 を指定した場合に、DB キュー読み出し先サーバ変更機能と CBLEEDBQ('ACSCTL ')が競合すると、ステータスコードが「00000」でも、DB キューの読み出し状態が読み出し停止、再開状態にならないことがあります。
5. CBLEEDBQ('ACSCTL ')のエラーリターン時は、ステータスコード別に次のように対処してください。
- 「05652」
CBLEEDBQ('ACSCTL ')の発行元の TP1/EE プロセスが出力するメッセージで要求先の TP1/EE プロセスを特定します。
要求先の TP1/EE プロセスが出力するエラーメッセージを基に障害を取り除いてください。
このステータスコードが返ったときは読み出し停止、再開要求の結果は不定です。
実行結果を確認したい場合は、CBLEEDBQ('INQUIRE2')または eedbqls コマンドで実行結果を確認してください。
 - 「05653」
DB キュー読み出し先サーバ変更機能を使用して読み出し先サーバを決定するか、該当の DB キューを読み出す TP1/EE プロセスを開始してください。
 - 「05654」
メッセージを読み出しているサーバが 07-80 以降の TP1/EE であることを確認してください。
 - 「05655」
CBLEEDBQ('ACSCTL ')の発行元の TP1/EE プロセスが出力する次のメッセージを参照して、障害を取り除いてください。
KFSB40320-E, KFSB40322-E, KFSB40350-E, KFSB40352-E
 - 「05691」
このステータスコードが返ったときは読み出し停止、再開要求の結果は不定です。
実行結果を確認したい場合は、CBLEEDBQ('INQUIRE2')または eedbqls コマンドで実行結果を確認してください。
 - 「05692」, 「05693」, 「05694」, 「05695」
これらのステータスコードが返ったときは読み出し停止、再開要求の結果は不定です。
実行結果を確認したい場合は、変更要求結果処理確認で発生したエラーの障害要因を取り除いてから、CBLEEDBQ('INQUIRE2')または eedbqls コマンドで実行結果を確認してください。

CBLEEDBQ('CLEAR ')

名称

DB キューの通番初期化

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CLEAR'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
01 一意名2.  
  02 データ名D PIC X(28).
```

機能

通番初期化属性がある DB キューの書き込み通番，読み出し通番を 0 で初期化します。未読み出しのメッセージがある場合は破棄します。初期化した通番を元に戻したり，破棄したメッセージを読み出したりはできません。

通番を初期化するための SQL の実行は，トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は，API 関数の実行順序に関係なく，DB キュー名の昇順に SQL を実行します。そのため，複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても，実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると，デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

通番を初期化するには，次の条件をすべて満たす必要があります。満たさない場合は，エラーリターンします。（ ）内は条件を満たさない場合のステータスコードです。

- 自プロセスが対象の DB キューから読み出す (05616)
- 読み出し停止中 (05603)
- 書き込み抑止中 (05603)
- 他スレッドでメッセージの読み出し，スキップ，初期化を実行していない (05603)

これらの条件に関係なく，強制的に通番を初期化することもできます。強制的に初期化する場合に，強制初期化と次のトランザクションが競合すると，次のトランザクションがロールバックすることがあります。

- DB キューからメッセージを読み出し中のトランザクション
- DB キューのメッセージをスキップ中のトランザクション

【TP1/FSP 限定】 TP1/FSP 使用時は、最初に起動したサーバの JI トランザクションから、初期化属性があるすべての DB キューを初期化できます。

UAP で値を設定するデータ領域

●データ名 A

DB キューの通番初期化要求を示す要求コードを「VALUE 'CLEAR△△△」と設定します。

●データ名 C

要求種別を設定します。

- 0
CBLEEDBQ('CLEAR ')の呼び出し条件をすべて満たしているかチェックして初期化する場合に指定します。
- 1
CBLEEDBQ('CLEAR ')の呼び出し条件に関係なく強制初期化する場合に指定します。
- 2 **【TP1/FSP 限定】**
定義されているすべての DB キューの通番を初期化します。TP1/FSP 使用時の最初に起動したサーバの JI トランザクションでだけ指定できます。

●データ名 D

DB キュー名を、27 バイト以内のアスキー文字列で設定します。文字列の最後には、空白を設定してください。この空白は文字列の長さに数えません。

データ名 C に 2 を指定した場合は、空白を設定してください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。

ステータスコード	意味
05601	データ名に設定した値が間違っています。
05604	データ名に設定した DB キュー名は存在しません。
05602	DB キュー機能は使用できません。
05603※	この関数を呼び出せる状態ではありません。 - 読み出し停止中でない - 書き込み抑止中でない - 他スレッドで DB キューにアクセスしている
05605※	該当するスレッドではリソースマネージャに接続されていません。
05607※	DB キューが使用できない状態です。
05609※	リソースマネージャアクセス時にエラーが発生しました。
05615※	その他のエラーが発生しました。
05616	設定された DB キューのタイプは機能対象外です。 相手読み出し DB キュー
05621	関数の呼び出し条件が不正です。 トランザクション種別不正
05608※	メモリが不足しました。
05623	DB キュー機能のライブラリがリンケージされていません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。
05651	DB キューの属性が対応していません。 - DB キューに通番初期化属性が付与されていない - DB キューのフォーマットタイプに 1 以上が指定されていない

注※

データ名 C に 2 を設定した場合、エラーが発生した DB キューでエラーリターンします。一部の DB キューの通番初期化が実行されているおそれがあります。無効にしたい場合、ロールバックしてください。

注意事項

- 通番の初期化は、サービス関数のリターン、または CBLEETRN('C-COMMIT')の呼び出し時に行われます。
- 同一グローバルトランザクションのトランザクションブランチ間で、同一 DB キューに対して CBLEEDBQ('MSGPUT ')の呼び出し、または初期化を行うと、デッドロックが発生する場合があります。
- DB キューからメッセージを読み出したトランザクションで、該当する DB キューに対して CBLEEDBQ('ACSCCTL ')を呼び出して読み出し停止にしたあと、該当する DB キューに初期化要求を行っても受け付けません。初期化要求は、別トランザクションまたはコマンドで実行してください。

4. データ名 C に 2 を設定する場合に、他プロセスで DB キューにアクセスしているとデッドロックするおそれがあります。データ名 C に 2 を指定する場合は、すべてのサーバで同じ共有情報表を使用するか、同じ共有情報表を使用していないプロセスからは DB キューにアクセスしないようにしてください。
5. CBLEEDBQ('CLEAR ')の受け付け後に、eedbqchgr コマンドで読み出し先サーバを 07-80 より前のバージョンのサーバに変更した場合、CBLEEDBQ('CLEAR ')を受け付けたトランザクションがロールバックします。
6. データ名 C に 0 を設定した CBLEEDBQ('CLEAR ')の受け付け後に、eedbqchgr コマンドで読み出し先サーバを変更した場合、CBLEEDBQ('CLEAR ')を受け付けたトランザクションがロールバックします。

CBLEEDBQ('CONCTL ')

名称

DB キューの接続制御

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CONCTL '  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(28).  
  02 データ名F PIC X(32).
```

機能

DB キューと DB キューサービスとを接続，および接続解除します。

UAP で値を設定するデータ領域

●データ名 A

DB キューの接続制御を示す要求コードを「VALUE 'CONCTL△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

要求種別を設定します。

VALUE 'C'

DB キューと DB キューサービスとを接続します。

VALUE 'R'

DB キューと DB キューサービスとの接続を解除します。

●データ名 E

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 • 該当する DB キューの読み出し処理中に接続解除、接続を続けて行えません。
05604	データ名に設定した DB キュー名は存在しません。
05607	DB キューが使用できない状態です。
05610	すでに、別 DB キューサービスと接続済みのため接続できません。
05611	データ名に設定した DB キューサービス名は存在しません。
05612	要求のあった DB キューと DB キューサービスは接続されていません。
05613	要求のあった DB キューサービスの属する DB キューサービスグループと DB キューの属する DB キューグループは DB キュー機能関連定義で関連づけられていません。
05615	そのほかのエラーが発生しました。
05616	設定された DB キューのタイプは機能対象外です。 • 相手読み出し DB キュー
05617	DB キューサービスが永久閉塞中（エントリポイントなし）です。
05621	関数の呼び出し条件が不正です。

ステータスコード	意味
05622	該当する DB キューサービスで使用する最大 DB キュー数を超えるため接続できません。
05623	DB キュー機能のライブラリがリンケージされていません。
05627	AP 間通信機能を使用する DB キューではありません。

注意事項

1. DB キュー機能関連定義の dbqgrpdef 定義コマンドに -c オプションを指定していない相手システムの読み出し DB キューとの接続、および接続解除はできません。
2. 該当する DB キューの読み出し処理中に接続解除と接続とを続けて実行できません。
3. 最初の AP 起動時には、定義によって任意の DB キューサービスに接続されているか、またはどの DB キューサービスにも接続されていない場合があります。DB キューをほかの DB キューサービスに接続変更する場合は、現在接続されている DB キューサービスを指定して接続を解除してから、変更先の DB キューサービスを指定して接続してください。
4. 次の場合、CBLEEDBQ('CONCTL ')はステータスコード「00000」でリターンします。
 - 接続されている DB キューと DB キューサービスに対して接続を要求した場合
 - 接続されていない DB キューと DB キューサービスに対して接続解除を要求した場合

CBLEEDBQ('INQSRV ')

名称

DB キューサービスの接続照会

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'INQSRV ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(32).  
  02 データ名E PIC 9(9) COMP.  
01 一意名3.  
  02 データ名F PIC 9(9) COMP-X.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).
```

機能

DB キューサービスと接続されている DB キュー名を照会します。

UAP で値を設定するデータ領域

●データ名 A

DB キューサービスの接続照会を示す要求コードを「VALUE 'INQSRV△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

DB キュー名の配列を格納する領域の配列要素数を設定します。1～64 の値を設定してください。

TP1/EE から値が返されるデータ領域

●データ名 E

DB キューサービスの閉塞状態が返されます。

- 0
閉塞解除中
- 1
障害による閉塞中
- 2
コマンドまたは関数による閉塞中
- 4
永久閉塞中（エントリポイントなし）

●データ名 G

DB キューサービスと接続されている DB キュー名の個数が返されます。

●データ名 H

DB キューサービスと接続されている DB キュー名の配列が返されます。文字列の最後は LOW-VALUE です。

一つの DB キュー名の長さは 28 バイトです。この領域は、(28×データ名 F に設定した値) 分の長さが必要です (単位: バイト)。

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下がありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05611	データ名に設定した DB キューサービス名は存在しません。

ステータスコード	意味
05615	そのほかのエラーが発生しました。
05619	DB キュー名リストサイズが不足しています。この時、データ名 G には該当する DB キュー数が設定されていますので、データ名 F の設定値を再考する必要があります。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。

注意事項

DB キュー名の配列を格納する領域が不足した場合（ステータスコード 05619）でも、データ名 F に設定した要素数分の DB キュー名は返されます。

CBLEEDBQ('INQUIRE ')

名称

DB キュー情報の照会

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'INQUIRE'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC X(28).  
01 一意名2.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).  
  02 データ名G PIC X(28).  
  02 データ名H PIC X(32).  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC 9(9) COMP.  
  02 データ名K PIC 9(9) COMP.  
  02 データ名L PIC 9(9) COMP.  
  02 データ名M PIC 9(9) COMP.
```

機能

DB キューの情報を照会します。ユーザキューアクセス機能を使用する DB キューでは、使用できません。CBLEEDBQ('INQUIRE2')を使用してください。

UAP で値を設定するデータ領域

●データ名 A

DB キュー情報の照会を示す要求コードを「VALUE 'INQUIRE△」と設定します。

●データ名 C

要求種別を設定します。

1

DB キュー読み出しサーバ変更機能または DB キュー書き込み抑止機能を使用する場合に設定します。このフラグを設定すると、最新の DB キュー情報を照会します。

DB キュー読み出しサーバ変更機能および DB キュー書き込み抑止機能以外の機能を使用する場合に設定します。このフラグを設定すると、DB キュー情報を照会します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 E

指定した DB キューに接続している DB キューサービス名が返されます。接続していない場合は LOW-VALUE を設定します。

●データ名 F

指定した DB キューが属する DB キューグループ名が返されます。文字列の最後は LOW-VALUE です。

●データ名 G

CBLEEDBQ('INQUIRE ')を呼び出すときにデータ名 D に設定した DB キュー名が返されます。文字列の最後は LOW-VALUE です。

●データ名 H※1

指定した DB キューを読み出すサービスグループ名が返されます。文字列の最後は LOW-VALUE です。

DB キュー機能関連定義の dbqgrpdef 定義コマンドに -f オプションを指定した場合、または、-k オプションに 1 を指定した場合で、DB キューからメッセージを読み出すサーバを確認できないとき、および DB キューからメッセージを読み出すサーバにサービスグループ名がないときは、LOW-VALUE を設定します。

●データ名 I

ステータスコード 1

DB キューの状態が返されます。※1

0

正常

1

障害による読み出し停止中

2

コマンド、TP1/EE サービス定義、または関数による読み出し停止中

4

DB 構成不正または DB キューの内容矛盾による使用不可

リソースマネージャ未接続で使用不可

ステータスコード 2

DB キューの状態が返されます。

4096

書き込み抑止※1, ※2

このステータスは、ステータスコード 1 が 4 または 8 の場合は無効となります。

●データ名 J

DB キューへ最後に書き込んだメッセージの通番が返されます。※3

●データ名 K

DB キューから最後に読み出したメッセージの通番が返されます。※3

●データ名 L

DB キュー機能関連定義の dbqdef 定義コマンドで指定した最大書き込みメッセージ数と同一の値が返されます。

●データ名 M

未読み出しメッセージ数が返されます。※3

注※1

データ名 C に 1 を指定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの-k オプションに 1 を指定した場合、DB キューから情報を読み込みます。

注※2

データ名 C に 1 を指定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの-k オプションに 1 を指定した場合、ほかの状態と加算した値が返されます。コマンド、TP1/EE サービス定義、または関数による読み出し停止中 (2) で、かつ書き込み抑止 (4096) の場合、4098 になります。

注※3

ステータスコードが次の場合、値は参照できません。

VALUE 'E'

DB 構成不正または DB キューの内容矛盾による使用不可

VALUE 'R'

リソースマネージャ未接続で使用不可

また、該当する DB キューの形態が相手読み出し DB キューの場合、値は返されません。

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 • 通番の初期化中です。
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05609	リソースマネージャへのアクセス時にエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンクされていない。
05627	AP 間通信機能を使用する DB キューではありません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。

注意事項

- DB キュー読み出しサーバ変更機能または DB キュー書き込み抑止機能を使用する場合、次に示す点に注意してください。
 - データ名 C に 1 を設定してください。ただし、1 を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、DB アクセスが発生するため、TP1/EE の性能が低下します。
 - データ名 C に 0 を設定しないでください。0 を設定し、かつ DB キュー機能関連定義の dbqgrpdef 定義コマンドの -k オプションに 1 を指定した場合、データ名 H やデータ名 I のステータスコード 1 に最新の DB キュー情報が返らないことがあります。
- ほかの TP1/EE で CBLEEDBQ('CLEAR ')または eedbqtblh コマンドで通番の強制初期化を行った場合、CBLEEDBQ('INQUIRE ')は初期化前の通番を返すことがあります。強制初期化後の通番を知りたい場合は、CBLEEDBQ('INQUIRE2')を使用してください。

CBLEEDBQ('INQUIRE2')

名称

DB キュー情報の照会

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'INQUIRE2'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC X(28).  
01 一意名2.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).  
  02 データ名G PIC X(28).  
  02 データ名H PIC X(32).  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC 9(9) COMP.  
  02 データ名K PIC 9(9) COMP.  
  02 データ名L PIC 9(9) COMP.  
  02 データ名M PIC 9(9) COMP.  
  02 データ名N PIC X(8).  
  02 データ名O PIC X(16).  
  02 データ名P PIC 9(9) COMP.  
  02 データ名Q PIC 9(9) COMP.  
  02 データ名R PIC X(5).  
  02 データ名S PIC X(1).  
  02 FILLER PIC X(74).
```

機能

DB にアクセスして DB キューの情報内容を照会します。

DB キューのフォーマットタイプが 1 の DB キューに対して実行できます。

UAP で値を設定するデータ領域

●データ名 A

DB キュー情報の照会要求を示す要求コードを「VALUE 'INQUIRE2'」と設定します。

●データ名 C

要求種別を、数値で設定します。

0

メッセージを読み出している TP1/EE プロセスの読み出し状態（データ名 I）の照会が不要な場合に設定します。

メッセージを読み出している TP1/EE プロセスの DB キューの読み出し状態を照会するための RPC 通信は行いません。

1

メッセージを読み出している TP1/EE プロセスの読み出し状態（データ名 I）の照会が必要な場合に設定します。

データ名 D に指定した DB キューを読み出している TP1/EE プロセスが API 発行元のプロセスと異なる場合、RPC 通信を行って DB キューの読み出し状態を照会します。

データ名 C に 1 を指定した場合、メッセージを読み出している TP1/EE のバージョンが 07-80 以降である必要があります。

●データ名 D

照会する DB キュー名を、27 バイト以内のアスキー文字列で設定します。文字列の最後には、空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 E

指定した DB キューに現在接続している DB キューサービス名が返されます。接続していない場合、およびユーザキューアクセス機能を使用した DB キューの場合は LOW-VALUE が設定されます。

●データ名 F

指定した DB キューが属する DB キューグループ名が返されます。文字列の最後は LOW-VALUE です。

●データ名 G

データ名 D に設定した DB キュー名が返されます。文字列の最後は LOW-VALUE です。

●データ名 H

指定した DB キューを読み出すサービスグループ名が返されます。文字列の最後は LOW-VALUE です。読み出すサーバにサービスグループ名がない場合、LOW-VALUE が設定されます。

●データ名 I

データ名 C に 1 を指定した場合の、DB キューの読み出し状態が返されます。

データ名 C に 0 を指定した場合、DB キューを読み出している TP1/EE プロセスが CBLEEDBQ('INQUIRE2')の発行元のプロセスと異なっていると、DB キュー状態 1 が読み出し停止中のときでも 0（正常）が返されます。

DB キューの読み出し状態は、DB キューの状態 1 と DB キューの状態 2 の和で返されます。

DB キューの状態 1：

DB キューの状態 1 が次のどれかで返されます。

0：正常

1：障害による読み出し停止中

2：コマンド、定義、または API 関数による読み出し停止中

4：DB 構成不正または DB キューの内容矛盾による使用不可

8：リソースマネージャ未接続で使用不可

DB キューの状態 2：

DB キューの状態 2 が次で返されます。

4096：書き込み抑止中

DB キューの状態 1 が 4 または 8 のときは加算しません。

このとき、書き込み抑止状態は不明となります。

●データ名 J

DB キューへ最後に書き込んだメッセージの通番が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

●データ名 K

DB キューから最後に読み出したメッセージの通番が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

●データ名 L

定義に指定した最大書き込みメッセージ数と同一の値が設定されます。

●データ名 M

未読み出し数が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

●データ名 N

読み出すサーバの基盤種別が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

文字列の最後は LOW-VALUE です。

TP1EE：TP1/EE

CosmiRM：uCosminexus Reliable Messaging

●データ名 O

該当する DB キューの読み出しを行うシステム種別が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

文字列の最後は LOW-VALUE です。

TP1/EE の場合、ノード識別子が返されます。

uCosminexus Reliable Messaging の場合、RMSystemName で指定したシステム名称が返されます。

●データ名 P

DB キューの種別が返されます。

0：AP 間通信機能を使用した DB キュー

1：ユーザキューアクセス機能を使用した DB キュー

●データ名 Q

DB キューに付与した属性の論理和が返されます。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

0：付与した属性なし

1：上書き抑止属性

2：通番初期化属性

4：データ連携支援書き込み

8：交替用メッセージ表書き込み機能を使用

●データ名 R

該当する DB キューを読み出す TP1/EE プロセスの TP1/Server Base のノード識別子が返されます。文字列の最後は LOW-VALUE です。データ名 I の DB キュー状態 1 が 4 または 8 の場合は参照できません。

次の条件をすべて満たす場合に TP1/Server Base のノード識別子が返されます。

- ・ 該当する DB キューを読み出すサーバの基盤種別が TP1/EE
- ・ 該当する DB キューを読み出す TP1/EE プロセスでネーム機能が有効
- ・ 該当する DB キューを読み出す TP1/EE プロセスの TP1/EE バージョンが 07-80 以降

TP1/Server Base のノード識別子を返さない場合は、LOW-VALUE が返されます。

●データ名 S

該当する DB キューに対する書き込み禁止状態を返します。

X'00'：書き込み可能状態

X'01'：書き込み禁止状態

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 通番の初期化中です。
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05609	リソースマネージャアクセス時にエラーが発生しました。
05615	その他のエラーが発生しました。
05621	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。
05651	データ名 D に設定した DB キューのフォーマットタイプが不正です。
05652	情報照会処理が時間切れになりました。※1
05653	情報照会処理に失敗しました。 対象の DB キューを読み出しているサーバが存在しません。※1
05654	情報照会処理に失敗しました。 対象の DB キューを読み出しているサーバが情報照会処理に対応していません。※1
05655	情報照会処理が失敗しました。※1
05691	情報照会処理は実行しましたが、情報照会処理の実行後に読み出し先サーバの変更を検知しました。※1, ※2

注※1

データ名 C に 1 を指定した場合にだけ発生するステータスコードです。

注※2

DB キューが HiRDB の場合にだけ発生するステータスコードです。

注意事項

1. データ名 C に 1 を指定し、かつ DB キューが Oracle の場合に DB キュー読み出し先サーバ変更機能と CBLEEDBQ('INQUIRE2')が競合すると、ステータスコードが「00000」でも DB キューの読み出し状態が最新の状態にならないことがあります。
2. CBLEEDBQ('INQUIRE2')のエラーリターン時は、ステータスコード別に次のように対処してください。
 - 「05652」
CBLEEDBQ('INQUIRE2')の発行元の TP1/EE プロセスが出力するメッセージで、照会先の TP1/EE プロセスを特定します。
照会先の TP1/EE プロセスが出力するエラーメッセージを基に障害を取り除いてください。
 - 「05653」
DB キュー読み出し先サーバ変更機能を使用して読み出し先サーバを決定するか、該当の DB キューを読み出す TP1/EE プロセスを開始してください。
 - 「05654」
メッセージを読み出しているサーバが 07-80 以降の TP1/EE であることを確認してください。
 - 「05655」
CBLEEDBQ('INQUIRE2')の発行元の TP1/EE プロセスが出力する次のメッセージを参照して、障害を取り除いてください。
KFSB40320-E, KFSB40322-E, KFSB40350-E, KFSB40352-E
 - 「05691」
CBLEEDBQ('INQUIRE2')を再発行するか、または eedbqls コマンドで実行結果を確認してください。

CBLEEDBQ('MSGGET ')

名称

メッセージの読み出し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGGET ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC X(28).  
01 一意名2.  
  02 データ名E PIC 9(9) COMP-X.  
  02 FILLER PIC X(68).  
  02 データ名F PIC X(n).  
01 一意名3.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC 9(9) COMP.
```

機能

メッセージを読み出します。

メッセージを DB キューから読み出すための SQL の実行は、この関数内で行います。このときの SQL は、SQL 実行中だけ排他を取ります。

メッセージを読み出し済みに更新するための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定するデータ領域

●データ名 A

メッセージの読み出し要求を示す要求コードを「VALUE 'MSGGET△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

読み出し対象となる DB キュー名を、27 バイト以内のアスキー文字列で設定します。文字列の最後には、空白を設定してください。この空白は文字列の長さに数えません。

●データ名 E

読み出したメッセージを格納する領域の長さを設定します。1～8388608 の値を指定してください。

●データ名 F

読み出したメッセージを格納する領域を設定します。

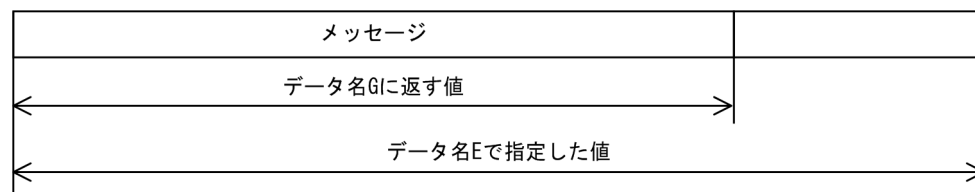
TP1/EE から値が返されるデータ領域

●データ名 F

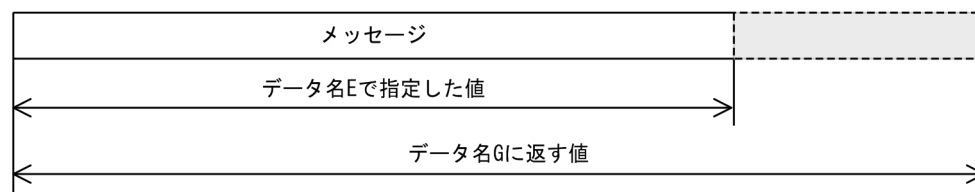
読み出したメッセージが返されます。

●データ名 G

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（ステータスコード 05620）でも、データ名 E で指定した長さにかかわらず、読み出したメッセージの長さが返されます。この場合、オーバーフローしたメッセージは読み出し成功として動作します。



●データ名 H

読み出したメッセージの通番が返されます。

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 - 通番の初期化中です。 - スキップ処理中です。
05604	要求された DB キューは存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05606	バッファ長が指定範囲外です。
05607	DB キューが使用できない状態です。
05609	リソースマネージャアクセス時にエラーが発生しました。
05616	設定された DB キューのタイプは機能対象外です。 相手読み出し DB キュー
05620	読み出したメッセージがバッファをオーバフローしました。読み出し成功として動作します。
05621	関数の呼び出し条件が不正です。 トランザクション種別が不正です。
05608	メモリが不足しました。
05615	その他のエラーが発生しました。
05623	DB キュー機能のライブラリがリンクageされていません。
05628	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、設定された DB キューのタイプの属性が変更されたため、CBLEETRN('ROLLMARK') を呼び出しました。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK') を呼び出しました。
05656	読み出し停止中です。
05657	他サービスでリード処理中です。
05658	ユーザキューアクセス機能を使用する DB キューではありません。
05659	未読み出しのメッセージはありません。

ステータスコード	意味
05697	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、RM で障害が発生したため、CBLEETRN('ROLLMARK')を呼び出しました。
05698	同じトランザクションブランチ内でこの関数が正常に終了したあと、再度呼び出されましたが、呼び出せる状態ではないため、CBLEETRN('ROLLMARK')を呼び出しました。 通番の初期化中です。

注意事項

1. DB キューサービス（ルートトランザクションブランチ）から呼び出された同一グローバルトランザクションのトランザクションブランチで、同一 DB キューに対してメッセージの読み出しを行うと、デッドロックが発生する場合があります。
2. 同一グローバルトランザクション内のトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだり、読み出したり、スキップしたりすると、デッドロックが発生する場合があります。
3. 同一トランザクションで同一の DB キューに対して複数回 CBLEEDBQ('MSGGET ')を呼び出すと、一回目以外の呼び出しでは、前回読み出したメッセージの次の通番のメッセージを読み出します。同じ通番のメッセージの再読み出しはしません。
4. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時にこの関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション（RM）
5. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション（ERRTRN3）のときは、リトライロールバックしません。

CBLEEDBQ('MSGPUT ')

名称

メッセージの書き込み

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3 一意名4
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGPUT ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC X(32).  
01 一意名2.  
  02 データ名E PIC 9(9) COMP.  
  02 データ名F PIC X(n).
```

機能

送信メッセージを DB キューへ書き込みます。

DB キューに書き込むための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定するデータ領域

●データ名 A

メッセージの書き込みを示す要求コードを「VALUE 'MSGPUT△△」と設定します。

●データ名 C

要求種別を設定します。

0

データ名 D に DB キュー名を設定してメッセージを書き込みます。

データ名 D に DB キューグループ名を設定してメッセージを書き込みます。

●データ名 D

DB キュー名または DB キューグループ名を設定します。

- データ名 C に 0 を設定する場合、DB キュー名を 27 バイト以内のアスキー文字列で設定します。
- データ名 C に 1 を設定する場合、DB キューグループ名を 31 バイト以内のアスキー文字列で設定します。

文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

メッセージを格納する領域の長さを設定します（単位：バイト）。1～メモリ関連定義の dbq_max_message_size オペランドの値の範囲の値を指定できます。

●データ名 F

メッセージを格納する領域のアドレスを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05604	要求された DB キューは存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05606	バッファ長が指定範囲外です。
05607	DB キューが使用できない状態です。次のどれかの状態となっています。 - DB 構成不正による使用不可 - リソースマネージャ未接続で使用不可

ステータスコード	意味
05608	メモリが不足しました。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。
05624	要求された DB キューグループは存在しません。
05625	要求された DB キューグループ内のすべての DB キューが次のどれかの状態のため、要求を受け付けられません。 • DB 構成不正または DB キューの内容矛盾による使用不可 • リソースマネージャ未接続で使用不可 • 縮退中 • 書き込み禁止状態
05629	DB キューが書き込み禁止状態です。

注意事項

1. DB キューサービス（ルートトランザクションブランチ）から呼び出された同一グローバルトランザクションのトランザクションブランチで、同一 DB キューに対してメッセージを書き込むと、デッドロックが発生する場合があります。
2. 同一グローバルトランザクション内のトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだり、スキップしたりすると、デッドロックが発生する場合があります。
3. eedbqwtpr コマンドで書き込みを抑止している DB キューに対して、CBLEEDBQ('MSGPUT ')で DB キューまたは DB キューグループを指定してメッセージの書き込み要求を行った場合、CBLEEDBQ('MSGPUT ')は正常に終了し、メッセージ書き込み要求を行ったトランザクションをロールバックしてからエラートランザクション（ERRTRNR）を起動します。ただし、CBLEETRN('C-COMMIT')を呼び出して連鎖モードのコミット要求をした場合、CBLEETRN('C-COMMIT')はエラーリターンし、エラートランザクション（ERRTRNR）を起動しません。
4. DB キューグループを指定してメッセージの書き込み要求を行った場合は、eedbqwtpr コマンドで書き込みを抑止している DB キューに対して eedbqislt コマンドを実行して縮退させることによって、DB キューグループ指定の対象外とする必要があります。
5. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時にこの関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション（RM）
6. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数実行後の同期点処理（UAP リターン, CBLEETRN('C-COMMIT')関数実行）のメッセージ表へのアクセスで障害が発生すると、トランザクションがロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション（ERRTRN3）の場合は、リトライロールバックしません。

CBLEEDBQ('MSGREFE2')

名称

メッセージの参照読み出し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGREFE2'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC X(28).  
  02 データ名E PIC 9(9) COMP-X.  
01 一意名2.  
  02 データ名F PIC 9(9) COMP-X.  
  02 FILLER PIC X(68).  
  02 データ名G PIC X(n).  
01 一意名3.  
  02 データ名H PIC 9(9) COMP.
```

機能

通番指定で、メッセージを読み出します（参照読み出し）。

他サーバ読み出しの DB キューのメッセージ、未読み出しのメッセージを読み出せます。

UAP で値を設定するデータ領域

●データ名 A

メッセージの読み出し要求を示す要求コードを「VALUE 'MSGREFE2'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

読み出し対象の DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 E

読み出し対象のメッセージの読み出し通番を設定します。1～2147483647 の値を設定してください。

●データ名 F

読み出したメッセージを格納する領域の長さを設定します。1～8388608 の値を指定してください。

●データ名 G

読み出したメッセージを格納する領域を設定します。

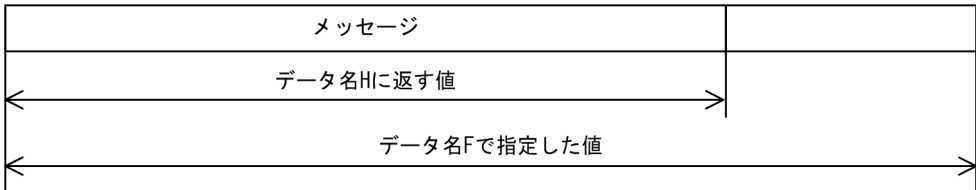
TP1/EE から値が返されるデータ領域

●データ名 G

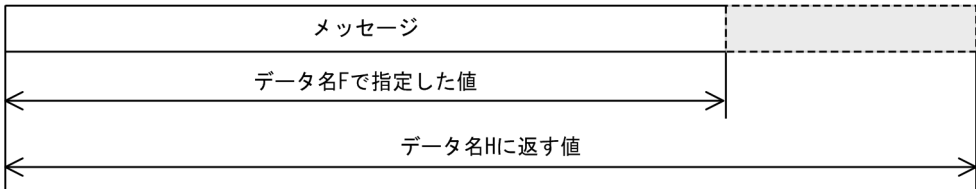
読み出したメッセージが返されます。

●データ名 H

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（ステータスコード 05620）は次のようになります。



●データ名 B

ステータスコードが，5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため，この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため，この機能は使用できません。
05601	データ名に設定した値が間違っています。

ステータスコード	意味
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 • 通番の初期化中です。
05604	要求された DB キューは存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05606	バッファ長が指定範囲外です。
05607	DB キューが使用できない状態です。
05609	リソースマネージャアクセス時にエラーが発生しました。
05619	データ名 E に指定した値は不当な値です。 • 1～最大メッセージ数の範囲外の通番 • 書き込まれていない通番
05620	読み出したメッセージがバッファをオーバーフローしました。
05621	関数の呼び出し条件が不正です。 • トランザクション種別が不正です。
05608	メモリが不足しました。
05615	その他のエラーが発生しました。
05623	DB キュー機能のライブラリがリンケージされていません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。

注意事項

1. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時にこの関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらの トランザクションの再起動トランザクション (RM)
2. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション (ERRTRN3) のときは、リトライロールバックしません。

CBLEEDBQ('MSGREFER')

名称

メッセージの参照読み出し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGREFER'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC X(28).  
  02 データ名E PIC 9(9) COMP-X.  
01 一意名2.  
  02 データ名F PIC 9(9) COMP-X.  
  02 データ名G PIC X(n).  
01 一意名3.  
  02 データ名H PIC 9(9) COMP.
```

機能

読み出し済みメッセージを読み出します（参照読み出し）。

UAP で値を設定するデータ領域

●データ名 A

メッセージの参照読み出しを示す要求コードを「VALUE 'MSGREFER'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

読み出すメッセージの通番を設定します。1～2147483647 の値を設定してください。

●データ名 F

読み出したメッセージを格納する領域の長さを設定します（単位：バイト）。1～8388608 の値を指定してください。

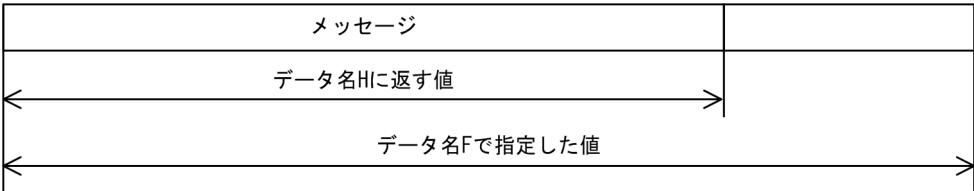
TP1/EE から値が返されるデータ領域

●データ名 G

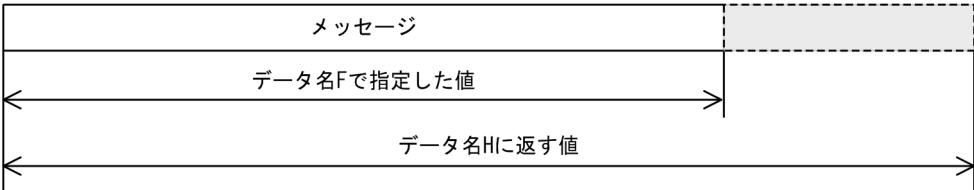
読み出したメッセージが返されます。

●データ名 H

読み出したメッセージの長さが返されます（単位：バイト）。



読み出したメッセージがオーバーフローしてしまう場合（ステータスコード 05620）は次のようになります。



●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 通番の初期化中です。
05604	要求された DB キューは存在しません。

ステータスコード	意味
05605	該当するスレッドではリソースマネージャに接続されていません。
05606	バッファ長が指定範囲外です。
05607	DB キューが使用できない状態です。
05608	メモリが不足しました。
05609	リソースマネージャへのアクセス時にエラーが発生しました。
05615	そのほかのエラーが発生しました。
05616	設定された DB キューのタイプは DB キュー機能で使用できません。 相手読み出し DB キュー
05619	データ名 E に指定した値は不当な値です。 1～最大メッセージ数の範囲外の通番です。 - 読み出し済みの通番ではありません。
05620	読み出したメッセージがバッファをオーバフローしました。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。

注意事項

1. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時にこの関数を使用できるトランザクション種別を、次に示します。ほかのトランザクション種別で実行した場合は、エラーリターンします。
MN, TM, CU, E3, およびこれらのトランザクションの再起動トランザクション (RM)
2. **【TP1/FSP 限定】** 交替用メッセージ表書き込み機能使用時に、この関数内で行うメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。ただし、障害が発生したトランザクションが UAP 異常終了通知トランザクション (ERRTRN3) のときは、リトライロールバックしません。

CBLEEDBQ('OBSCAN ')

名称

オンラインバッチ処理の強制終了

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSCAN '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).
```

機能

オンラインバッチ処理を中断中のロットに対して、オンラインバッチ処理の強制終了要求を行います。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の強制終了要求を示す要求コードを「VALUE 'OBSCAN△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05608	メモリが不足しました。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンクされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05632	該当するロットが次に示す状態でないため、要求を受け付けられません。 • 障害による読み出し中断中 • 運用による読み出し中断中 • 回復処理失敗による中断中 • 正常終了トランザクション障害による中断中 • 回復処理失敗による正常終了トランザクション中断中 • 正常終了トランザクション障害による中断中の回復処理失敗 • 該当するロットに対して関数を発行したトランザクションがコミットまたはロールバックしている

注意事項

1. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は強制終了します。
2. 発行元トランザクションがコミットまたはロールバックするまでは、該当するロットが受け付けられるオンラインバッチの関数は、CBLEEDBQ('OBSINQ ')および CBLEEDBQ('OBSCDT ')だけです。ほかのオンラインバッチの関数は受け付けません。

CBLEEDBQ('OBSCDT ')

名称

引き継ぎ情報の更新

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSCDT ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).  
01 一意名3.  
  02 データ名F PIC 9(9) COMP.  
  02 データ名G PIC X(n).
```

機能

オンラインバッチ処理中のロットの引き継ぎ情報を更新します。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の開始要求を示す要求コードを「VALUE 'OBSCDT△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

引き継ぎ情報のサイズを設定します。1～256 バイトの範囲で設定できます。

●データ名 G

引き継ぎ情報を格納している領域を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下がありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05608	メモリが不足しました。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05634	該当するロットが次に示す状態でないため、要求を受け付けられません。 • 実行中 • 障害による読み出し中断中 • 運用による読み出し中断中 • 回復処理失敗による中断中 • 正常終了トランザクション障害による中断中 • 回復処理失敗による正常終了トランザクション中断中 • 正常終了トランザクション障害による中断中の回復処理失敗 • 終了トランザクション実行中 • 回復処理失敗による終了トランザクション実行中
05635	引き継ぎ情報を格納している領域のサイズが指定範囲外です。

注意事項

1. バッチ処理の開始時に引き継ぎ情報が指定されなかった場合は、この関数で指定された引き継ぎ情報を追加します。
2. 呼び出し元トランザクションの決着結果がコミットかまたはロールバックかに関係なく、引き継ぎ情報が更新されます。
ただし、強制停止中の場合、引き継ぎ情報は呼び出し元トランザクションの決着を待たないで即時更新されます。
3. トランザクション内で複数の要求があった場合は、最新の要求が有効となります。

CBLEEDBQ('OBSEND ')

名称

オンラインバッチ処理の終了

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSEND '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).
```

機能

イベント型オンラインバッチ処理の場合に、オンラインバッチ処理の終了要求を行います。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の終了要求を示す要求コードを「VALUE 'OBSEND△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05634	該当するロットが次に示す状態でないため、要求を受け付けられません。 ・ 実行中 ・ 該当するロットに対して関数を発行したトランザクションがコミットまたはロールバックしている
05638	対象ロットのサービストランザクションから呼び出されていません。

注意事項

1. この関数は、対象ロットのサービストランザクションからだけ呼び出せます。
2. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は終了します。
3. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの関数は、CBLEEDBQ('OBSINQ ')および CBLEEDBQ('OBSCDT ')だけです。ほかのオンラインバッチの関数は受け付けません。
4. この関数の動作はオンラインバッチ機能関連定義の dbq_obs_trn_end_api オペランドの指定値に依存します。dbq_obs_trn_end_api オペランドの指定値が Y の場合、CBLEETRN('C-COMMIT')または CBLEETRN('C-ROLL ')の呼び出し以降、この関数は受け付けません。

CBLEEDBQ('OBSINQ ')

名称

ロットの情報の照会

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSINQ ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).  
01 一意名3.  
  02 データ名F PIC X(32).  
  02 データ名G PIC X(28).  
  02 データ名H PIC X(24).  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC 9(9) COMP.  
  02 データ名K PIC 9(9) COMP.  
  02 データ名L PIC 9(9) COMP.  
  02 データ名M PIC 9(9) COMP.  
  02 データ名N PIC 9(9) COMP.
```

機能

ロットの情報を照会します。

UAP で値を設定するデータ領域

●データ名 A

DB キューの情報の照会要求を示す要求コードを「VALUE 'OBSINQ△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 F

指定したロットに現在接続しているサービス名が返されます。文字列の最後は LOW-VALUE です。※1

●データ名 G

データ名 D に設定した DB キュー名が返されます。文字列の最後は LOW-VALUE です。

●データ名 H

データ名 E に設定したロット名が返されます。文字列の最後は LOW-VALUE です。

●データ名 I

ロットの状態が返されます。

- 1
実行中
- 2
障害による読み出し中断中
- 4
運用による読み出し中断中
- 8
未起動
- 16
回復処理失敗による中断中
- 32
正常終了トランザクション障害による中断中
- 48
回復処理失敗による正常終了トランザクション中断中、または正常終了トランザクション障害による中断中の回復処理失敗
- 64
正常、中断、または強制終了トランザクション実行中
- 80
回復処理失敗による中断の終了トランザクション実行中

●データ名 J

ロットのオンラインバッチ処理形態が返されます。※1

1

データ型オンラインバッチ処理

2

イベント型オンラインバッチ処理

●データ名 K

DB キュー内の該当するロットのメッセージ数が返されます。※1, ※2

●データ名 L

該当するロットの読み出し済みメッセージ数が返されます。※1, ※2

●データ名 M

該当するロットのスキップメッセージ数が返されます。※1, ※2

●データ名 N

未読み出しメッセージ数が返されます。※1, ※2

注※1

データ名 I が 8 以外のときに値が返されます。

注※2

該当するロットのデータ名 J が 1 で、データ名 I が 16 以外のときに、値が返されます。

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05615	そのほかのエラーが発生しました。

ステータスコード	意味
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。

CBLEEDBQ('OBSRBSP ')

名称

ロールバック要求時のメッセージの読み出しスキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSRBSP'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

データ型オンラインバッチ処理の場合で、トランザクションのロールバック要求時に、読み出したメッセージのロールバック処理（メッセージ読み出し通番の戻し）をしないようにします。コミット要求があった場合は、この機能は無効となります。

UAP で値を設定するデータ領域

●データ名 A

ロールバック要求時のメッセージの読み出しスキップ要求を示す要求コードを「VALUE 'OBSRBSP△」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。

ステータスコード	意味
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05614	同一メッセージ処理のロールバック監視回数を越えたため、要求を受け付けません。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05638	対象ロットのサービストラランザクションから発行されていません。

注意事項

1. この関数は、対象ロットのサービストラランザクションからだけ呼び出せます。
2. この関数を呼び出したトラランザクションの実行中に計画停止 B でシステムを停止すると、関数は正常に終了しますが、メッセージの読み出しはスキップされない場合があります。また、読み出しスキップ処理中に障害が発生すると、メッセージの読み出しがスキップされない場合があります。この場合、同一メッセージを読み出したときに、再度この関数を呼び出してください。
3. 呼び出し元トラランザクションがロールバックしたときに、メッセージの読み出しがスキップされます。
4. この関数の動作はオンラインバッチ機能関連定義の dbq_obs_trn_end_api オペランドの指定値に依存します。dbq_obs_trn_end_api オペランドの指定値が Y の場合、CBLEETRN('C-COMMIT')またはCBLEETRN('C-ROLL ')の呼び出し以降、この関数は受け付けません。

CBLEEDBQ('OBSRST ')

名称

オンラインバッチ処理の再開始

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSRST ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).
```

機能

オンラインバッチ処理の再開始要求を行います。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の再開始要求を示す要求コードを「VALUE 'OBSRST△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05608	メモリが不足しました。
05609	リソースマネージャへのアクセス時にエラーが発生しました。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05632	該当するロットが次に示す状態でないため、要求を受け付けられません。 • 障害による読み出し中断中 • 運用による読み出し中断中 • 回復処理失敗による中断中 • 正常終了トランザクション障害による中断中 • 回復処理失敗による正常終了トランザクション中断中 • 正常終了トランザクション障害による中断中の回復処理失敗 • 該当するロットに対して関数を発行したトランザクションがコミットまたはロールバックしている
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。

注意事項

1. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は再開始します。

2. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの関数は、CBLEEDBQ('OBSINQ ')およびCBLEEDBQ('OBSCDT ')だけです。ほかのオンラインバッチの関数は受け付けません。

CBLEEDBQ('OBSSKIP ')

名称

メッセージの読み出しスキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSSKIP'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).  
  02 データ名F PIC X(1).
```

機能

データ型オンラインバッチ処理の場合に、指定されたロットのメッセージを読み出し済みにします。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の終了要求を示す要求コードを「VALUE 'OBSSKIP△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

要求種別を設定します。

VALUE'A'

指定されたロットのすべてのメッセージを読み出し済みにします。

VALUE'O'

未読み出しの先頭にある 1 メッセージだけを読み出し済みにします。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下がありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05608	メモリが不足しました。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05626	この要求で読み出し済みにするメッセージはありません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05634	該当するロットが次に示す状態でないため、要求を受け付けられません。 • 障害による読み出し中断中 • 運用による読み出し中断中 • 該当するロットに対して関数を発行したトランザクションがコミットまたはロールバックしている
05636	該当するロットのタイプはこの要求では受け付けません。

注意事項

1. スキップの処理中（コミットされるまで）は別のスキップを受け付けません。
2. 運用または障害によって、該当するロットが中断中の場合だけ呼び出せます。

3. 呼び出し元トランザクションがコミットしたときに、メッセージが読み出し済みになります。
4. 呼び出し元トランザクションがコミットまたはロールバックするまでの間、該当するロットが受け付けられるオンラインバッチの関数は、CBLEEDBQ('OBSINQ '), または CBLEEDBQ('OBSCDT ')だけです。ほかのオンラインバッチの関数は受け付けません。

CBLEEDBQ('OBSSTART')

名称

オンラインバッチ処理の開始

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSSTART'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).  
  02 データ名F PIC X(32).  
  02 データ名G PIC X(1).  
01 一意名3.  
  02 データ名H PIC 9(9) COMP.  
  02 データ名I PIC X(n).
```

機能

オンラインバッチ処理の開始要求を行います。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の開始要求を示す要求コードを「VALUE 'OBSSTART'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

データ型オンラインバッチ処理を起動する場合は、定義ファイルで定義されている DB キュー名を指定してください。イベント型オンラインバッチ処理を起動する場合は、任意の DB キュー名を指定してください。

い。オンラインバッチ処理開始以降の関数を呼び出すときに指定する DB キュー名は、この関数で指定した DB キュー名を指定します。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

該当するロットでの読み出しトランザクションを起動するサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 G

オンラインバッチ処理形態を設定します。

VALUE 'D'

データ型オンラインバッチ処理を起動します。

VALUE 'E'

イベント型オンラインバッチ処理を起動します。

●データ名 H

引き継ぎ情報のサイズを設定します。0～256 バイトの範囲で設定できます。

●データ名 I

引き継ぎ情報を格納している領域を設定します。不要な場合（データ名 H に 0 を設定した場合）、設定する必要はありません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	オンラインバッチ機能は使用できません。
05603	この関数を呼び出せる状態ではありません。

ステータスコード	意味
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05608	メモリが不足しました。
05609	リソースマネージャへのアクセス時にエラーが発生しました。
05611	データ名に設定したサービス名は存在しません。
05615	そのほかのエラーが発生しました。
05617	データ名に設定したサービス名はエントリポイントが存在しないため使用できません。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05632	該当するロットが次に示す状態でないため、要求を受け付けられません。 - 初期状態 - 該当するロットに対して関数を発行したトランザクションがコミットまたはロールバックしている
05633	ロット管理テーブルの面数が不足しました。
05634	同一トランザクションで以前に行われた開始要求がエラーリターンしたあと、ロールバック処理が行われていないため、要求を受け付けません。
05635	引き継ぎ情報を格納している領域のサイズが指定範囲外です。
05637	オンラインバッチ処理終了のためのサービスにエントリポイントが存在しないため、要求を受け付けません。

注意事項

1. ステータスコードが 00000 以外のときは必ずロールバックを行ってください。
2. 呼び出し元トランザクションがコミットしたときに、オンラインバッチ処理は開始します。
3. 呼び出し元トランザクションがロールバックまたはコミットするまでの間、該当するロットが受け付けられるオンラインバッチの関数は、CBLEEDBQ('OBSINQ ')だけです。ほかのオンラインバッチの関数は受け付けません。

CBLEEDBQ('OBSSTOP ')

名称

オンラインバッチ処理の中断

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OBSSTOP'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(28).  
  02 データ名E PIC X(24).
```

機能

オンラインバッチ処理の中断要求を行います。

UAP で値を設定するデータ領域

●データ名 A

オンラインバッチ処理の中断要求を示す要求コードを「VALUE 'OBSSTOP△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 E

ロット名を 23 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05604	データ名に設定した DB キュー名は存在しません。
05615	そのほかのエラーが発生しました。
05621	関数の呼び出し条件が不正です。
05623	オンラインバッチ機能のライブラリがリンケージされていません。
05631	データ名に設定したロット名は存在しません。または、対象のロットではありません。
05632	該当するロットが次に示す状態でないため、要求を受け付けられません。 • 実行中
05638	対象ロットのサービストランザクションから発行されていません。

注意事項

1. この関数は、対象ロットのサービストランザクションからだけ発行できます。
2. 呼び出し元トランザクションの決着結果がコミットかまたはロールバックかに関係なく、オンラインバッチ処理が中断します。

CBLEEDBQ('RBSKIP ')

名称

ロールバック要求時のメッセージの読み出しスキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A  PIC X(8) VALUE 'RBSKIP ' .  
  02 データ名B  PIC X(5).  
  02 FILLER     PIC X(3).  
  02 データ名C  PIC S9(9) COMP VALUE ZERO.
```

機能

トランザクションのロールバック要求時に、読み出したメッセージのロールバック処理（メッセージ読み出し通番の戻し）をしないようにします。トランザクションのコミット要求があった場合、この機能は無効になります。この関数を呼び出せるのは、DB キューサービストランザクションだけです。

UAP で値を設定するデータ領域

●データ名 A

ロールバック要求時のメッセージの読み出しスキップを示す要求コードを「VALUE 'RBSKIP△△'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。

ステータスコード	意味
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05614	次にロールバック要求すると、同一メッセージ処理のロールバック監視回数を超えるため、要求を受け付けません。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。

注意事項

この関数を呼び出したトランザクションの実行中に計画停止 B でシステムを停止すると、関数は正常に終了しますが、メッセージの読み出しはスキップされない場合があります。また、メッセージの読み出しスキップ処理中に障害が発生すると、読み出しはスキップされない場合があります。同一メッセージを読み出したときに、この関数を再度呼び出してください。

CBLEEDBQ('SKIP ')

名称

メッセージの読み出しスキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'SKIP  ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(28).  
01 一意名3.  
  02 データ名F PIC 9(9) COMP-X.
```

機能

DB キューのメッセージを読み出し済みにして、読み出しをスキップします。

メッセージを読み出し済みに更新するための SQL の実行は、トランザクションの同期点処理内で行います。1 トランザクションで複数の DB キューに対してこの関数を実行した場合は、API 関数の実行順序に関係なく、DB キュー名の昇順に SQL を実行します。そのため、複数の DB キューに対してこの関数を実行するトランザクション間でこの関数の実行順序が異なっても、実行順序が原因でデッドロックすることはありません。

同一グローバルトランザクションのトランザクションブランチ間でこの API 関数を実行すると、デッドロックが発生する場合があります。詳細は「注意事項」を参照してください。

UAP で値を設定するデータ領域

●データ名 A

メッセージの読み出しスキップを示す要求コードを「VALUE 'SKIP△△△△」と設定します。

●データ名 C

0 を設定します。

●データ名D

要求種別を設定します。

VALUE 'A'

読み出し通番が書き込み通番と同じになるまで、すべてのメッセージを読み出し済みにします。

VALUE 'O'

未読み出しの先頭にある 1 メッセージだけを読み出し済みにします。

VALUE 'N'

データ名 F で設定された読み出し通番まで、メッセージを読み出し済みにします。

●データ名E

DB キュー名を 27 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名F

メッセージ読み出し通番を設定します。

データ名 D に VALUE 'N'を設定した場合は、1～2147483647 の値を設定してください。

データ名 D に VALUE 'N'以外を設定した場合は、0 を設定してください。

0 以外を設定しても 0 として扱います。

TP1/EE から値が返されるデータ領域

●データ名B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。 - 読み出し停止中ではありません。 - 通番の初期化中です。

ステータスコード	意味
05604	データ名に設定した DB キュー名は存在しません。
05605	該当するスレッドではリソースマネージャに接続されていません。
05607	DB キューが使用できない状態です。
05608	メモリが不足しました。
05609	リソースマネージャへのアクセス時にエラーが発生しました。
05616	設定された DB キューのタイプは DB キュー機能で使用できません。 • 相手読み出し DB キュー
05620	データ名 F に指定した値は不当な値です。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。
05626	この要求で読み出し済みにするメッセージはありません。
05639	HiRDB の暗黙的ロールバックが発生したため、CBLEETRN('ROLLMARK')を呼び出しました。

注意事項

- スキップの対象になるのは、トランザクションのコミットによって書き込みが完了したメッセージです。コミットしていない場合は、メッセージをスキップできません。
- メッセージ読み出し通番は、トランザクションのコミット時に更新されます。
- メッセージをスキップしたあと、トランザクションのコミットまでは、さらにスキップすることはできません。
- 該当する DB キューが読み出し停止の状態（コマンド、TP1/EE サービス定義、関数で指定するか、または障害発生による）にある場合だけ、メッセージをスキップできます。
- 同一グローバルトランザクションのトランザクションブランチ間で、同一 DB キューに対してメッセージを書き込んだりスキップしたりすると、デッドロックが発生する場合があります。
- DB キューの読み出しトランザクションで、該当する DB キューを読み出し停止状態にしている場合、メッセージをスキップできません。別のトランザクションで CBLEEDBQ('SKIP ')を呼び出すか、またはコマンドを実行してメッセージをスキップしてください。

CBLEEDBQ('SRVCTL ')

名称

DB キューサービスの閉塞制御

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEDBQ' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'SRVCTL '  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(32).
```

機能

DB キューサービスを閉塞、および閉塞解除します。

UAP で値を設定するデータ領域

●データ名 A

DB キューサービスの閉塞制御を示す要求コードを「VALUE 'SRVCTL△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

要求種別を設定します。

VALUE 'D'

DB キューサービスを閉塞します。

VALUE 'A'

DB キューサービスの閉塞を解除します。

●データ名 E

DB キューサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05601	データ名に設定した値が間違っています。
05602	DB キュー機能は使用できません。
05603	この関数を呼び出せる状態ではありません。
05611	データ名に設定した DB キューサービス名は存在しません。
05615	そのほかのエラーが発生しました。
05618	DB キューサービスが永久閉塞中（エントリポイントなし）です。
05621	関数の呼び出し条件が不正です。
05623	DB キュー機能のライブラリがリンケージされていません。

注意事項

次の場合、CBLEEDBQ('SRVCTL ')はステータスコード「00000」でリターンします。

- 閉塞状態の DB キューに対して閉塞を要求した場合
- 閉塞解除状態の DB キューに対して閉塞解除を要求した場合

資源の排他制御 (CBLEELCK)

任意の資源を排他制御するプログラムについて説明します。資源の排他制御の COBOL-UAP 作成用プログラムを次に示します。

- CBLEELCK('GET ') – 資源の排他
- CBLEELCK('RELALL ') – 全資源の排他の解除
- CBLEELCK('RELNAME ') – 資源名称を指定した排他の解除

CBLEELCK('GET ')

名称

資源の排他

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEELCK' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GET      '.  
  02 データ名B PIC X(5).  
  02 FILLER     PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(16).  
  02 データ名D PIC X(2).  
  02 FILLER     PIC X(6).  
  02 データ名E PIC X(6).  
  02 FILLER     PIC X(2).  
  02 データ名F PIC X(6).  
  02 FILLER     PIC X(2).  
  02 データ名G PIC X(7) VALUE 'MIGRATE'.  
  02 FILLER     PIC X(1).
```

機能

UAP で使う資源の排他を指定します。排他の管理は、TP1/EE のトランザクションマネージャで管理するグローバルトランザクション単位で処理されます。

ここで指定した排他は、排他を解除する COBOL-UAP 作成用プログラム (CBLEELCK('RELALL ')または CBLEELCK('RELNAME ')) を呼び出すか、または CBLEELCK('GET ')を呼び出したグローバルトランザクションの同期点取得後に解除されます。

UAP で値を設定するデータ領域

●データ名 A

資源の排他を示す要求コードを「VALUE 'GET△△△△△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

排他する資源の名称を、16 バイトの英数字で設定します。ここで設定した資源名称を基に、ロックサービスで排他を管理します。ロックサービスでは、文字列の内容についてはチェックしません。論理的に正しい名称を設定してください。資源名称に英数字以外の値を使った場合は、タイムアウト情報、および eelckls コマンドの表示が乱れることがあります。

●データ名 D

排他制御モードとして、次のどちらかを設定します。重複して指定することはできません。

VALUE 'PR'

資源を参照します。ほかの UAP には参照だけを許可し、更新を禁止します。

VALUE 'EX'

資源を更新します。ほかの UAP には参照と更新の両方を禁止します。

●データ名 E

ほかの UAP で使用する資源と競合した場合に、資源の解放待ちとするか、エラーリターンとするかを設定します。次に示すどちらかの値を設定します。

VALUE 'WAIT△△'

ほかの UAP で使用する資源と競合した場合、資源を排他できるまで待ちます。

VALUE 'NOWAIT'

ほかの UAP で使用する資源と競合した場合、エラーリターンします。

●データ名 F

このプログラムで指定する排他をテストとしないかどうかを設定します。次に示すどちらかの値を設定します。

VALUE 'TEST△△'

排他をテストとします。CBLEELCK('GET ')が正常に終了しても、データ名 C に設定した資源は確保されませんので注意してください。

VALUE 'NOTEST'

排他をテストとしません。CBLEELCK('GET ')が正常に終了した場合、データ名 C に設定した資源は確保されます。

●データ名 G

VALUE 'MIGRATE'を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00401	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合もこのステータスコードが返されます。
00402	この機能は使用できません。
00450	ほかの UAP が指定した資源を使っています。
00453	排他待ち限界経過時間（トランザクション関連定義の lck_wait_timeout オペランドの指定値）を超えたため、資源を確保できませんでした。
00454	排他制御用のテーブルが不足しています。
00455	トランザクション処理ではない UAP から指定しています。

CBLEELCK('RELALL ')

名称

全資源の排他解除

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEELCK' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'RELALL ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(7) VALUE 'MIGRATE' .  
  02 FILLER PIC X(1).
```

機能

CBLEELCK('GET ')で指定した資源の排他をすべて解除します。同期点取得前に排他を解除するときに、CBLEELCK('RELALL ')を呼び出します。

排他をしたグローバルトランザクションが終了した場合、TP1/EE のロックサービスによって自動的に排他は解除されます。このときは、UAP で排他を解除する必要はありません。

UAP で値を設定するデータ領域

●データ名 A

資源の排他解除を示す要求コードを「VALUE 'RELALL△△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

「VALUE 'MIGRATE」を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00401	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合もこのステータスコードが返されます。
00402	この機能は使用できません。
00455	トランザクション処理ではない UAP から呼び出しています。
00456	このプログラムを呼び出したトランザクションは、資源を確保していません。

CBLEELCK('RELNAME ')

名称

資源名称を指定した排他解除

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEELCK' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'RELNAME'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(16).  
  02 データ名D PIC X(7) VALUE 'MIGRATE'.  
  02 FILLER PIC X(1).
```

機能

CBLEELCK('GET ')で指定した資源の排他を、資源名称を指定して解除します。同期点取得前に排他を解除するときに、CBLEELCK('RELNAME ')を呼び出します。

排他をしたグローバルトランザクションが終了したときに、TP1/EE のロックサービスによって排他は自動的に解除されます。このときは、UAP で排他を解除する必要はありません。

UAP で値を設定するデータ領域

●データ名 A

資源の排他解除を示す要求コードを「VALUE 'RELNAME△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

排他の指定を解除する資源名称を設定します。資源名称は CBLEELCK('GET ')で設定した名称と同じ値を設定してください。

●データ名 D

「VALUE 'MIGRATE」を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00401	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合もこのステータスコードが返されます。
00402	この機能は使用できません。
00455	トランザクション処理ではない UAP から呼び出しています。
00456	解除を指定した資源名称に該当する資源がありません。

メッセージログの出力 (CBLEELOG)

UAP からメッセージログを出力するプログラムについて説明します。メッセージログの出力の COBOL-UAP 作成用プログラムを次に示します。

- CBLEELOG('PRINT ') – メッセージログの出力

CBLEELLOG('PRINT ')

名称

メッセージログの出力

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEELLOG' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'PRINT ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP.  
  02 データ名C PIC X(12).  
  02 データ名D PIC X(3).  
01 一意名2.  
  02 データ名F PIC 9(9) COMP.  
  02 データ名G PIC X(n).  
01 一意名3.  
  02 データ名H PIC S9(9) COMP VALUE ZERO.
```

機能

データ名 G に設定した文字列に、行ヘッダ、メッセージ ID、および要求元プログラム ID を付けて、メッセージログファイル、syslog、標準出力に出力します。メッセージログファイルの行ヘッダは、プロセス ID、メッセージ通番、サービス名、中央処理通番、年月日、時刻です。syslog の行ヘッダは定義で設定します。syslog 出力は、データ名 Z に設定した出力レベルが定義に指定した出力レベルと同じか高い場合です。データ名 Z に出力レベルを設定しなかった場合の出力レベルは、1 です。

標準出力への出力有無は、次の表のようになります。定義：log_usermsg_console に all を指定している場合、データ名 Z に設定する標準出力への出力有の設定に関係なく、標準出力へ出力します。

定義： log_msg_console	定義： log_usermsg_console	API：データ名 Z 標準出力への出力有の設定	標準出力への出力有無
Y（デフォルト）	all	有り	○
		無し（デフォルト）	○
	api（デフォルト）	有り	○
		無し（デフォルト）	×
N	—	—	×

(凡例)

- ：メッセージを標準出力に出力します。
- ×：メッセージを標準出力に出力しません。
- －：該当しません。

TP1/EE では、CBLEELOG('PRINT ')で使うメッセージ ID 用に、05000 から 06999 までの範囲の番号を割り当てています。UAP から出力するメッセージ ID の番号には、05000 から 06999 までの範囲の値を付けてください。

メッセージテキストに復改文字「¥n」を含むメッセージログは、次のように扱います。

メッセージの出力先	復改文字「¥n」の扱い
メッセージログファイル	空白に置き換えて 1 行で出力する
syslog	空白に置き換えて 1 行で出力する
標準出力	そのまま出力する（「¥n」で改行する）

障害が発生して UAP からメッセージログが出力できない場合でも、CBLEELOG('PRINT ')がステータスコード「00000」で正常に終了することがあります。そのため、メッセージログが抜ける場合があります。メッセージログの抜けは、メッセージログに付けるメッセージログ通番で確認できます。

一つのスレッドから CBLEELOG('PRINT ')を呼び出した場合、メッセージログファイルへの出力順序は保証されます。複数のスレッドから別々に CBLEELOG('PRINT ')を呼び出した場合は、呼び出した順にメッセージログファイルに出力されないことがあります。

エラーを検出した場合、エラーの原因を示すメッセージログに CBLEELOG('PRINT ')に指定したメッセージを付けて、標準エラー出力に出力します。

UAP で値を設定するデータ領域

●データ名 A

メッセージログの出力を示す要求コードを「VALUE 'PRINT△△△」と設定します。

●データ名 Z

syslog 出力レベルと標準出力への出力フラグを設定します。

syslog 出力レベルだけを設定する場合

syslog 出力レベルを表す次のフラグを設定します。

0

特定のレベルは設定しません。syslog 出力レベルはデフォルトの 1 になります。

1

syslog 出力レベルを 1 にします。

2

syslog 出力レベルを 2 にします。

3

syslog 出力レベルを 3 にします。

4

syslog 出力レベルを 4 にします。

5

syslog 出力レベルを 5 にします。

6

syslog 出力レベルを 6 にします。

7

メッセージを syslog に出力しません。

syslog 出力レベルは、1 が最も高く、6 が最も低くなります。データ名 Z に設定した出力レベルがメッセージログ関連定義の log_syslog_out オペランドに指定した出力レベルと同じか高い場合、syslog にメッセージログが出力されます。データ名 Z に出力レベルを設定しなかった場合の出力レベルは 1 です。

syslog 出力レベルと標準出力への出力フラグを設定する場合

syslog 出力レベルフラグとの論理和で設定します。

8

メッセージを標準出力に出力します。

●データ名 C

メッセージログごとに付ける識別子（メッセージ ID）を設定します。「KFSBnnnnnn-x」の形式（11 文字）で、末尾に空白を付けてください。

TP1/EE では、CBLEELOG('PRINT ')で使うメッセージ ID の通番（nnnnnn の部分）用に、05000～06999 の範囲の番号を割り当てています。UAP から出力するメッセージ ID の通番は、05000～06999 の範囲の値にしてください。

●データ名 D

CBLEELOG('PRINT ')を呼び出した UAP を識別する任意の値（要求元プログラム ID）を設定します。英数字 2 文字で、末尾に空白を付けてください。

●データ名 F

メッセージログファイルにメッセージログとして出力する文字列の長さを設定します（単位：バイト）。222 までの値を指定してください。

●データ名 G

メッセージログファイルにメッセージログとして出力したい任意の文字列を設定します。最大 222 バイトで、末尾に空白を付けてください。

文字列に改行「¥n」が含まれる場合、1 行で出力されます。

●データ名 H

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
01900	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含みます。
01907	メッセージログ機能内の排他設定に失敗しました。
01908	メッセージログ機能内の排他解除に失敗しました。
01914	メッセージの組み立てに失敗しました。
01915	メッセージを出力できる状態にありません。
01916	syslog 出力に失敗しました。
01918	標準エラー出力に失敗しました。
01919	メッセージログファイルへの出力に失敗しました。
01920	すべてのメッセージログファイル切り替えに失敗しました。

注意事項

1. ログ出力量が多い場合は、CBLEelog('PRINT ')のリターンが遅くなります。例えば、障害発生時にメッセージ出力量が著しく多くなると、トランザクション処理時間が長くなります。これはスローダウンの要因になるので、注意してください。
2. CBLEelog('PRINT ')で標準出力に非 ASCII 文字を含むメッセージを出力する場合、標準出力に複数の文字コードが混在する可能性があります。この場合は次のどちらかの方法で回避してください。
 - 出力メッセージを ASCII 文字だけにする

- データ名 Z に 8 を指定しない

高速メッセージ送信 (CBLEEMCH)

クライアントとサーバの間で UDP プロトコルで高速通信するときに使用するプログラムについて説明します。高速メッセージ送信の COBOL-UAP 作成用プログラムを次に示します。

- CBLEEMCH('CMTSND ')【XTC 限定】 – トランザクション非同期の一方送信メッセージの送信
- CBLEEMCH('CMTSNDGT')【XTC 限定】 – 一方送信メッセージ情報の取得
- CBLEEMCH('CMTSND SY')【XTC 限定】 – トランザクション同期の一方送信メッセージの送信
- CBLEEMCH('OTQBKCTL')【XTC 限定】 – 出力キュー (OTQ) の閉塞制御
- CBLEEMCH('OTQINQIR')【XTC 限定】 – 出力キュー (OTQ) の情報の照会
- CBLEEMCH('OTQSKIP ')【XTC 限定】 – 一方送信メッセージの送信スキップ

CBLEEMCH('CMTSND ')【XTC 限定】

名称

トランザクション非同期の一方送信メッセージの送信

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CMTSND ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC X(4).  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).  
01 一意名2.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).
```

機能

トランザクションのコミットが確定したとき、出力キュー（OTQ）の通番を採番して出力キューにメッセージを格納します。その後、送信用のスレッドを使用して出力キューの通番順に一方送信メッセージをUDP プロトコルで送信します。これによって、SPP に対してサービスを要求します。トランザクションがロールバックとなった場合、メッセージは破棄されます。

サービスを要求するときには、「サービスグループ名」と「サービス名」をデータ領域に設定します。この名称に該当するサービスプログラムへサービスが要求されます。

CBLEEMCH('CMTSND ')を使う場合、サーバ UAP があるノードの TP1/EE が稼働していることが前提です。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'CMTSND△△」と設定します。

●データ名 C

メッセージ送信形態とオプションを数値で設定します。

0

ユニキャストで一方送信メッセージを送信する場合に設定します。

1

マルチキャストで CL サーバに一方送信メッセージを送信する場合に設定します。

65536

自プロセスあてに一方送信メッセージを送信する場合に設定します。

一方送信メッセージを永続属性とする場合は、「16」を加算してください。

●データ名 D

メッセージ送信先で登録された処理キューの中で優先的に起動するかどうかを設定します。

VALUE 'HI△△'

優先的に起動します。

VALUE 'LOW△'

優先的に起動しないで、登録した順序で起動します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します。1～EEMCH_MAX_MESSAGE_SIZE※の範囲の値を設定できます。EEMCH_MAX_MESSAGE_SIZE は、eemch.h で定義してあります。

注※

EEMCH_MAX_MESSAGE_SIZE の値は、将来拡張される場合があります。

●データ名 H

サービスの入力パラメタを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	要求コード（データ名 A）が間違っています。
08002	データ名 F に設定した値が間違っています。
08003	データ名 E に設定した値が間違っています。
08004	データ名 G に設定した値が間違っています。
08005	データ名 D に設定した値が間違っています。
08006	データ名 C に設定した値が間違っています。
08008	データ名 G に設定した入力パラメタ長が、最大値を超えています。
08010	データ名 F に設定したサービスグループ名は、定義されていません。
08011	データ名 E に設定したサービス名は、定義されていません。
08030	発行条件が不正です。
08040	出力キュー（OTQ）が閉塞中、または閉塞解除処理中です。
08050	メモリが不足しました。
08099	システムエラーが起きました。

CBLEEMCH('CMTSND ')がエラーになる場合

CBLEEMCH('CMTSND ')がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合、送信先サービス関連定義を指定していない場合、トランザクション処理でない UAP から要求された場合、終了処理中に要求された場合、初期化トランザクション（MI）もしくは終了トランザクション（ME）から要求された場合、または ee_trn_rollback_mark が発行済みである場合

CBLEEMCH('CMTSND ')は「08030」でエラーリターンします。

データ名 C に 65536 を指定して自プロセス送信を行う場合で、送信先サービス関連定義に自サービス名（自サービスグループ名）が設定されていないとき

CBLEEMCH('CMTSND ')は「08010」（「08011」）でエラーリターンします。

メモリ関連定義の `udp_send_message_buf_cnt` オペランドで指定した UDP 用送信バッファ面数 (UOBF) が不足した場合、またはデータ名 C に 65536 を指定して自プロセス送信を行うときにメモリ関連定義の `pce_no` オペランドで指定する処理キュー数が不足した場合

CBLEEMCH('CMTSND ')は「08050」でエラーリターンします。

注意事項

1. 設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で指定しておいてください。
2. メッセージ送信時にエラーが発生した場合、高速メッセージ送信関連定義の `mch_send_retry_count` オペランドに従い、再送信を行います。再送信がエラーとなった場合は、ERRTRNS にメッセージを引き継ぐために UDP 用送信バッファ（実行系から待機系に転送されたメッセージのときは UDP 用受信バッファ）を確保し、ERRTRNS を起動します。
3. メッセージ送信は、自プロセスあてを含め UDP 用送信バッファを使用します。
4. 自プロセスあてのサービス要求によるサービス実行と他サービス要求（サービス要求受信やタイマトランザクション起動要求など）によるサービス実行との順序性は保証されません。
5. この関数を発行したトランザクションでは、トランザクションの最適化は行われません。
6. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。

CBLEEMCH('CMTSNDGT')【XTC 限定】

名称

一方送信メッセージ情報の取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CMTSNDGT'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC 9(9) COMP.  
  02 データ名D PIC 9(9) COMP.  
  02 FILLER PIC X(3).  
01 一意名3.  
  02 データ名E PIC 9(9) COMP.  
  02 データ名F PIC X(32).  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(32).  
  02 データ名I PIC X(1).  
  02 FILLER PIC X(3).  
  02 データ名J PIC 9(9) COMP.  
  02 FILLER PIC X(4).  
  02 データ名K PIC 9(9) COMP.  
  02 データ名L ADDRESS.  
  02 FILLER PIC X(160).
```

機能

メッセージ送信に失敗（未送信も含む）したトランザクション同期の一方送信メッセージとトランザクション非同期の一方送信メッセージのメッセージ情報を取得します。ステータスコードが 00000（次のメッセージ情報なし）になるまで、この関数を連続して呼び出すことによって、同一トランザクションで一方送信メッセージの送信要求を行ったすべてのメッセージ情報を取得できます。ステータスコードが 00000 の状態でこの関数を呼び出した場合は、最初からメッセージ情報を取得します。

一方送信メッセージのメッセージ情報は、ERRTRNR のタイミングで取得できます。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'CMTSNDGT'」と設定します。

●データ名 C

参照したい一方送信メッセージの要求種別を設定します。

0

CBLEEMCH('CMTSND ')要求

1

CBLEEMCH('CMTSND SY')要求

●データ名 Z

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

参照できるメッセージ情報の件数が返されます。0 の場合は、データ名 E、データ名 F、データ名 G、データ名 H、データ名 I、データ名 J、データ名 K、およびデータ名 L の内容は参照できません。

●データ名 E

送信先サービスグループ名の長さが返されます。

●データ名 F

送信先サービスグループ名が返されます。

●データ名 G

送信先サービス名の長さが返されます。

●データ名 H

送信先サービス名が返されます。

●データ名 I

出力キュー（OTQ）の自動閉塞の有無が次のコードで返されます。

0

出力キューの自動閉塞なし

1

サービスグループレベルの出力キューの自動閉塞あり

2

サービスレベルの出力キューの自動閉塞あり

●データ名 J

送信結果が次のコードで返されます。

0

送信正常

1

送信障害（パケットサイズ不正などの RPC 関連定義の不正）

2

送信障害（リソース不足）

4

送信障害（送信先受信不可状態）

3

送信障害（上記以外）

16

送達確認待ちタイムアウト

17

送達確認待ち受信障害

32

送達確認エラーメッセージ受信（リソース不足）

33

送達確認エラーメッセージ受信（CL 同期エラー）

34

送達確認エラーメッセージ受信（送信先サービス閉塞）、または自プロセスのサービス閉塞

64

送達確認エラーメッセージ受信（送信先サービスグループ名の不正）

65

送達確認エラーメッセージ受信（送信先サービス名の不正）

66

送達確認エラーメッセージ受信（メッセージ送信形態の不正／オプションの不正）

128

送達確認エラーメッセージ受信（上記以外）

256
出力キューの閉塞中

512
処理キューの登録失敗

65536
未送信

●データ名K

送信メッセージのサイズが返されます。

●データ名L

送信メッセージのアドレスが返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました（次のメッセージ情報なし）。
00001	TP1/EE の環境下ではありません。
00003	正常に終了しました（次のメッセージ情報あり）。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含みます。
08030	発行条件が不正です。
08050	メモリが不足しました。

CBLEEMCH('CMTSNDGT')がエラーになる場合

CBLEEMCH('CMTSNDGT')がエラーリターンする場合について説明します。

メモリ関連定義の system_work_size オペランドで指定したシステムワーク領域が不足した場合

CBLEEMCH('CMTSNDGT')は「08050」でエラーリターンします。

CBLEEMCH('CMTSNDY')【XTC 限定】

名称

トランザクション同期の一方送信メッセージの送信

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CMTSNDY'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC X(4).  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).  
01 一意名2.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).
```

機能

トランザクションのコミットが確定したとき、トランザクション処理と同じスレッドから一方送信メッセージを UDP プロトコルで送信します。これによって、SPP に対してサービスを要求します。トランザクションがロールバックとなった場合、メッセージは破棄されます。

サービスを要求するときには、「サービスグループ名」と「サービス名」をデータ領域に設定します。この名称に該当するサービスプログラムへサービスが要求されます。

CBLEEMCH('CMTSNDY')を使う場合、サーバ UAP があるノードの TP1/EE が稼働していることが前提です。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'CMTSNDY'」と設定します。

●データ名 C

メッセージ送信形態とオプションを数値で設定します。

0

ユニキャストで一方送信メッセージを送信する場合に設定します。

1

マルチキャストで CL サーバに一方送信メッセージを送信する場合に設定します。

65536

自プロセスあてに一方送信メッセージを送信する場合に設定します。

一方送信メッセージを永続属性とする場合は、「16」を加算してください。

●データ名 D

メッセージ送信先で登録された処理キューの中で優先的に起動するかどうかを設定します。

VALUE 'HI△△'

優先的に起動します。

VALUE 'LOW△'

優先的に起動しないで、登録した順序で起動します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します。1～EEMCH_MAX_MESSAGE_SIZE※の範囲の値を設定できます。EEMCH_MAX_MESSAGE_SIZE は、eemch.h で定義してあります。

注※

EEMCH_MAX_MESSAGE_SIZE の値は、将来拡張される場合があります。

●データ名 H

サービスの入力パラメタを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	要求コード（データ名 A）が間違っています。
08002	データ名 F に設定した値が間違っています。
08003	データ名 E に設定した値が間違っています。
08004	データ名 G に設定した値が間違っています。
08005	データ名 D に設定した値が間違っています。
08006	データ名 C に設定した値が間違っています。
08008	データ名 G に設定した入力パラメタ長が、最大値を超えています。
08010	データ名 F に設定したサービスグループ名は、定義されていません。
08011	データ名 E に設定したサービス名は、定義されていません。
08030	発行条件が不正です。
08040	出力キュー（OTQ）が閉塞中、または閉塞解除処理中です。
08050	メモリが不足しました。
08099	システムエラーが起きました。

CBLEEMCH('CMTSND SY')がエラーになる場合

CBLEEMCH('CMTSND SY')がエラーリターンする場合について説明します。

クラスタ連携関連定義の cluster_mode オペランドに Y を指定している場合

CBLEEMCH('CMTSND SY')は「08030」でエラーリターンします。その他の理由については CBLEEMCH('CMTSND ')の説明を参照してください。

注意事項

1. 設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で指定しておいてください。
2. メッセージ送信時にエラーが発生した場合、高速メッセージ送信関連定義の mch_send_retry_count オペランドに従い、再送信を行います。再送信がエラーとなった場合は、ERRTRNS*が起動されます。ただし、要求元のトランザクションがエラートランザクションの場合、ERRTRNS は起動されません。

注※

高速メッセージ送信関連定義の `mch_send_err_rollback` オペランドで Y を指定した場合は、ERRTRNR が起動されます。ただし、要求元のトランザクション種別が MN または TM 以外の場合、ERRTRNR は起動されません。

3. CL サーバでは使用できません。
4. メッセージ送信は、自プロセスあてを含め UDP 用送信バッファ (UOBF) を使用します。
5. 高速メッセージ送信関連定義 `mch_send_err_rollback` オペランドで Y を指定した場合は、自プロセスあて用の UDP 用送信バッファは 2 倍必要です。
6. この関数を発行したトランザクションでは、トランザクションの最適化は行われません。
7. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。

CBLEEMCH('OTQBKCTL')【XTC 限定】

名称

出力キュー（OTQ）の閉塞制御

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OTQBKCTL'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).
```

機能

指定したサービスグループ下のサービス名に対応する出力キューの閉塞/閉塞解除，または送信先サービスグループ下のすべてのサービスに対応する出力キューの閉塞/閉塞解除を行います。

UAP で値を設定するデータ領域

●データ名 A

サービスの閉塞制御要求を示す要求コードを「VALUE 'OTQBKCTL'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

閉塞か閉塞解除かを設定します。

VALUE 'S'

出力キューを閉塞します。

出力キューの状態がすでに自動閉塞中の場合，関数による閉塞中の状態に変わります。

VALUE 'R'

出力キューの閉塞を解除します。

サービスグループレベルの場合、通信を行って送信先サービスグループの起動状態を確認します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

また、全サービスを対象にしたい場合、サービス名を設定しないで、先頭を空白にしてください。

サービス名を設定しない場合は、サービスグループレベルの出力キューの閉塞/閉塞解除、サービス名を設定した場合は、サービスレベルの出力キューの閉塞/閉塞解除となります。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
08010	データ名 F に設定したサービスグループ名は、定義されていません。
08011	データ名 E に設定したサービス名は、定義されていません。
08030	発行条件が不正です。
08041	この関数を発行できる状態ではありません。出力キューの閉塞処理中、または出力キューの閉塞解除処理中です。
08042	出力キューの閉塞解除をできる状態ではありません。
08099	システムエラーが起きました。

注意事項

1. サービスグループレベルで出力キューが閉塞中（出力キューの自動閉塞中含む）の場合、サービスレベルの出力キューの閉塞解除はできません。「08030」でエラーリターンします。

2. サービスグループレベルの出力キューの自動閉塞中にサービスグループレベルの出力キューの閉塞要求を行った場合、出力キューの閉塞監視を終了します。ただし、サービスレベルの出力キューの閉塞要求を行った場合は、出力キューの閉塞監視は続行します。
3. サービスグループレベルの出力キューの自動閉塞中にサービスグループレベルの出力キューの閉塞解除要求を行った場合、出力キューの閉塞監視を終了します。

CBLEEMCH('OTQINQIR')【XTC 限定】

名称

出力キュー（OTQ）の情報の照会

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OTQINQIR'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC 9(9) COMP.  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC 9(9) COMP.  
  02 データ名K PIC X(1).  
  02 FILLER PIC X(239).
```

機能

指定したサービスグループ下のサービス名に対応する出力キューの情報を照会します。照会対象のメッセージは、トランザクション非同期の一方送信メッセージです。

UAP で値を設定する引数

●データ名 A

サービスの要求を示す要求コードを「VALUE 'OTQINQIR'」と設定します。

●データ名 Z

0 を設定します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 G

出力キューの状態が次のコードで返されます。

- 0
正常
- 1
自動閉塞中
- 2
関数による閉塞中
- 3
コマンドによる閉塞中

●データ名 H

出力キューへ最後に送信要求した、トランザクション非同期の一方送信メッセージの出力キューの通番が返されます。

●データ名 I

出力キューから最後に送信した、トランザクション非同期の一方送信メッセージの出力キューの通番が返されます。

●データ名 J

未送信（送信失敗を含む）のメッセージ数が返されます。

●データ名 K

出力キューの閉塞レベルが返されます。

*

閉塞中でない

G

サービスグループレベル

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
08010	データ名 F に設定したサービスグループ名は、定義されていません。
08011	データ名 E に設定したサービス名は、定義されていません。
08030	発行条件が不正です。

CBLEEMCH('OTQSKIP ')【XTC 限定】

名称

一方送信メッセージの送信スキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMCH' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'OTQSKIP'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC 9(9) COMP.  
  02 データ名D PIC 9(9) COMP.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(32).
```

機能

指定した通番までのトランザクション非同期の一方送信メッセージを送信済み状態として、出力キュー（OTQ）から破棄します。

指定したサービスグループ下のサービス名に対応する出力キューのメッセージを送信済み状態とします。一方送信メッセージの送信スキップ要求は、該当の出力キューが閉塞している場合だけ受け付けられます。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'OTQSKIP△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

送信済みとする対象となるメッセージを設定します。

0

出力キューに格納しているすべてのメッセージを送信済みにします。

1

出力キューに格納している未送信（送信失敗を含む）の 1 メッセージを送信済みにします。

2

データ名 D で設定された出力キューの通番までのメッセージを送信済みにします。

●データ名 D

送信済みとする出力キューの通番を設定します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
08001	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
08010	データ名 F に設定したサービスグループ名は、定義されていません。
08011	データ名 E に設定したサービス名は、定義されていません。
08030	発行条件が不正です。
08041	この関数を発行できる状態ではありません。
08050	メモリが不足しました。
08099	システムエラーが起きました。

注意事項

送信スキップ対象のメッセージがない場合は、正常扱いとなります。

メモリ管理 (CBLEEMEM)

メモリ上のワークセグメントの割り当ておよび返還ならびに大量処理用メモリ管理機能の関数について説明します。メモリ管理の COBOL-UAP 作成用プログラムを次に示します。

- CBLEEMEM('GETWK ') – ワークセグメントの割り当て
- CBLEEMEM('MDPSTART') 【XTC 限定】 – 大量処理用メモリ管理機能の使用開始
- CBLEEMEM('RLSWK ') – ワークセグメントの返還

CBLEEMEM('GETWK ')

名称

ワークセグメントの割り当て

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMEM' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GETWK  '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 FILLER PIC X(3).  
  02 データ名E PIC 9(9) COMP.  
  02 データ名F ADDRESS.
```

機能

ユーザが使用するワークセグメントを割り当てます。

UAP で値を設定するデータ領域

●データ名 A

ワークセグメントの割り当てを示す要求コードを「VALUE 'GETWK△△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

ワークセグメントの利用形態によって、次のように設定します。

VALUE 'U'

TASK 属性セグメント（ワークセグメントのうち、該当するトランザクションでだけ使用できるセグメント）

VALUE 'F'

FREE 属性セグメント（ワークセグメントのうち、複数のトランザクション間で使用できるセグメント）

●データ名 E

割り当てるワークセグメントの長さを設定します（単位：バイト）。1～16777216 の範囲で指定してください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 F

ワークセグメントのアドレスが返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
05101	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05102	関数の呼び出し条件が不正です。
05106	セグメントが不足しました。

注意事項

1. 関数が正常に終了したときだけ、ワークセグメントを使用できます。
2. ワークセグメントの割り当て時に領域破壊を検知した場合は、プロセスダウンします。

CBLEEMEM('MDPSTART')【XTC 限定】

名称

大量処理用メモリ管理機能の使用開始

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMEM' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MDPSTART'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

大量処理用メモリ管理機能を有効にします。

UAP で値を設定するデータ領域

●データ名 A

大量処理用メモリ管理機能の開始要求を示す要求コードを「VALUE 'MDPSTART'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。

ステータスコード	意味
05101	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05102	関数の呼び出し条件が不正です。

CBLEEMEM('MDPSTART')がエラーになる場合

CBLEEMEM('MDPSTART')がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合

CBLEEMEM('MDPSTART')は「05102」でエラーリターンします。

メモリ関連定義の memory_mdpsys_area_size オペランドまたは memory_mdpsr_area_size オペランドのどちらも指定していない場合

CBLEEMEM('MDPSTART')は「05102」でエラーリターンします。

注意事項

1. UAP 終了後のトランザクションが終了するまで、大量処理用メモリ管理機能は継続します。
2. 大量処理用メモリ管理機能を使う UAP 内で、大量処理用ユーザ領域ではなくユーザワーク領域から FREE 属性のセグメントを取得したい場合、CBLEEMEM('MDPSTART')の前に CBLEEMEM('GETWK ')を発行する必要があります。

CBLEEMEM('RLSWK ')

名称

ワークセグメントの返還

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEEMEM' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'RLSWK ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D ADDRESS.
```

機能

使用済みのワークセグメントを返還します。

UAP で値を設定するデータ領域

●データ名 A

ワークセグメントの返還を示す要求コードを「VALUE 'RLSWK△△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

返還するワークセグメントのアドレスを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
05101	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05102	関数の呼び出し条件が不正です。
05109	無効なセグメントを設定しています。

注意事項

ワークセグメントの返還時に領域破壊を検知した場合は、プロセスダウンします。

リモートプロシジャコール (CBLEERPC)

クライアント／サーバ形態の通信をするときに使う、TP1/EE のリモートプロシジャコールのプログラムについて説明します。リモートプロシジャコールの COBOL-UAP 作成用プログラムを次に示します。

- CBLEERPC('CALL ') – 遠隔サービスの要求
- CBLEERPC('CALLTO ') – 通信先を指定した遠隔サービスの要求
- CBLEERPC('CMTSND ') – トランザクションと同期して送信する非応答型 RPC 要求
- CBLEERPC('CMTSNDTO') – 通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求
- CBLEERPC('DISCARD') – すべての非同期応答型 RPC の応答受信拒否
- CBLEERPC('DISCARDS') – 特定の非同期応答型 RPC の応答受信拒否
- CBLEERPC('GETERDES') – エラーが発生した非同期応答型 RPC の識別子取得
- CBLEERPC('GETWATCH') – サービスの応答待ち時間の参照
- CBLEERPC('POLLANYR') – 非同期応答型 RPC の応答受信
- CBLEERPC('R-SEND ') – 抑止していた RPC 応答メッセージの送信
- CBLEERPC('R-SPEND ') – RPC 応答メッセージ送信の抑止
- CBLEERPC('SETWATCH') – サービスの応答待ち時間の更新

CBLEERPC('CALL ')

名称

遠隔サービスの要求

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CALL ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC S9(9) COMP.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(n).  
01 一意名2.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).  
01 一意名3.  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC X(n).
```

機能

SPP にサービスを要求します。CBLEERPC('CALL ')を使うときは、要求するサービスがどのノードにあるかを意識する必要はありません。

サービスを要求するときには、「サービスグループ名」と「サービス名」を CBLEERPC('CALL ')のデータ領域に設定します。この名称に該当するサービスプログラムへサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

CBLEERPC('CALL ')を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定するデータ領域

●データ名 A

遠隔サービスの要求を示す要求コードを「VALUE 'CALL△△△△」と設定します。

●データ名 C

RPC の形態とオプションを数値で設定します。

0

同期応答型 RPC

1

非応答型 RPC

2

非同期応答型 RPC

トランザクション処理からのサービス要求で、要求先の処理をトランザクションにしない（要求先の処理を要求元と同一のグローバルトランザクションにしない）場合は、RPC の形態を示す数値に「32」を加算します。「32」を加算すると、トランザクションの処理からのサービス要求でも、サービスプログラムの処理はトランザクションになりません。

自プロセス送信（通信レス）を行う場合、RPC の形態を示す数値に「131072」を加算します。「131072」を加算する場合は、必ず「32」も加算してください。

非同期応答型 RPC で、サービス関数が設定した応答の長さを参照したい場合は、RPC の形態を示す数値に「512」を加算します。その場合は、CBLEERPC('POLLANYR')の正常終了時に、CBLEERPC('CALL')のサービス応答とサービス応答の長さに、サービス関数が設定した応答と応答の長さがそれぞれ設定されます。「512」を加算しない場合は、サービス応答だけ設定されます。

【XTC 限定】 サーバとの通信を、コネクションレス（UDP プロトコル）で行う場合は、RPC の形態を示す数値に「4096」を加算します。「4096」を加算する場合は、必ず「32」も加算してください。

(例 1)

非応答型 RPC でトランザクションにしないサービス要求をする場合

データ名 C には $1 + 32 = 33$ を設定します。

(例 2)

非同期応答型 RPC で自プロセス宛てでサービス要求をする場合

データ名 C には $2 + 32 + 131072 = 131106$ を設定します。

(例 3)

非同期応答型 RPC でコネクションレス（UDP プロトコル）を使用して、サービス要求をする場合

データ名 C には $2 + 4096 + 32 = 4130$ を設定します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の IMA サイズ以下の値を設定してください。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●データ名 H

サービスの入力パラメタを設定します。

●データ名 I

サービスの応答の長さ（データ名 J の長さ）を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

サービス要求先が TP1/EE の場合、相手先の TP1/EE の OMA サイズ以下の値を設定してください。

非応答型 RPC の場合も、サービスの応答の長さを設定した領域のアドレスを指定する必要があります。この場合、サービスの応答の長さには、0 を設定します。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

非同期応答型 RPC の場合に、記述子が返されます。

サーバUAP から値が返されるデータ領域

●データ名I

サービスプログラムが設定した応答の長さ（データ名Jの長さ）が返されます（単位：バイト）。

●データ名J

サービスプログラムが設定した応答が返されます。

サーバUAP から返される値

サービスプログラムの処理が終了して応答が戻ってくると、次の値を参照できます。

- ・ サービスプログラムの応答（データ名J）
- ・ サービスプログラムの応答の長さ（データ名I）

データ名Jまたはデータ名Iを参照できる場合を次に示します。

- ・ 同期応答型 RPC の場合
CBLEERPC('CALL ')がリターンしたあと、データ名Jおよびデータ名Iを参照できます。
- ・ 非同期応答型 RPC の場合
CBLEERPC('POLLANYR')が該当する応答を受け取ってリターンしたあと、データ名Jを参照できます。RPC の形態に「512」を加算していない場合、データ名Iは参照できません。
- ・ 非応答型 RPC の場合
データ名Jおよびデータ名Iは参照できません。

CBLEERPC('CALL '), または CBLEERPC('POLLANYR')がエラーリターンした場合は、データ名Jおよびデータ名Iは参照できません。

返ってきた応答が、クライアントUAP で用意した応答の領域（データ名J）よりも大きい場合は、ステータスコードが「00309」でエラーリターンします。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。非同期応答型 RPC の場合は、データ名 D に記述子が設定されました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	要求コード（データ名 A）が間違っています。

ステータスコード	意味
00304	メモリが不足しました。要求先が TP1/EE の場合は、データ名 I の設定値が OMA サイズを超過しているおそれがあります。
00305	ネットワークの障害などによってサービス要求できるコネクションの取得に失敗しました。
00306	ネットワークに障害が起きました。
00307	処理が時間切れ（タイムアウト）になりました。
	サービスを要求された SPP が、処理を完了する前に異常終了しました。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の <code>rpc_reply_tp1mode_down</code> オペランドの指定値が J の場合、サーバ側のトランザクション決着結果がロールバック以外になりました。
00308	データ名 G に設定した入力パラメタ長が、最大値を超えています。
00309	返ってきた応答が、クライアント UAP で用意した領域に収まりません。
00310	データ名 F に設定したサービスグループ名は、定義されていません。
00311	データ名 E に設定したサービス名は、定義されていません。
	データ名 F に設定したサービスグループ名が、サービス要求先プロセスのサービスグループ名と異なります。次の条件をすべて満たす場合に発生します。 - サービスグループ情報関連定義の <code>eesvgdef</code> 定義コマンドで登録したサービスグループを指定している - RPC の形態として同期応答型 RPC を指定している - サービスの要求先が TP1/EE で、かつ要求先の転送機能が無効（RPC 関連定義の <code>rpc_loadbalance</code> オペランド、および <code>rpc_transfer_othersvg</code> オペランドに N を指定）になっている
00312	データ名 E に設定したサービス名があるサービスグループは、閉塞しています。
00313	データ名 E に設定したサービスは、終了処理中です。
00314	データ名 E に設定したサービスの UAP プロセスが、稼働していません。
	サービス要求の応答待ち時間に 0 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。
00315	データ名 E に設定したサービスがあるノードの TP1/Server Base または TP1/EE が稼働していません。異常終了、停止中、終了処理中、および通信障害が起こったことが考えられます。
00316	設定したサービスで、システムエラーが起きました。
00317	設定したサービスで、メモリが不足しました。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の <code>rpc_reply_tp1mode</code> オペランドまたは <code>rpc_reply_tp1mode_down</code> オペランドの指定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。次に示す原因が考えられます。 - UAP リターン後の同期点処理でトランザクション決着できなかった、かつ、RPC 関連定義の <code>rpc_replymsg_save</code> オペランドに N を指定した - UAP リターン後の同期点処理以降にプロセスダウンした

ステータスコード	意味
00318	システムエラーが起きました。
00319	サービスプログラムが TP1/EE に返した応答の長さが、1～EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
00320	サービスを要求されたノードにある TP1/Server Base または TP1/EE は、開始処理中です。
00323	メモリが不足しました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00324	システムエラーが起きました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00325	設定したサービスで、システムエラーが起きました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00326	返ってきた応答が、クライアント UAP で用意した領域に収まりません。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00327	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある TP1/Server Base または TP1/EE のバージョンが、クライアントの TP1/Server Base または TP1/EE のバージョンよりも古い場合、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。 このステータスコードは、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
00331	データ名 F に設定した値が間違っています。
00332	データ名 E に設定した値が間違っています。
00334	データ名 G に設定した値が間違っています。
00336	データ名 I に設定した値が間違っています。
00337	データ名 C に設定した値が間違っています。または、値の組み合わせが不正です。
00338	rpc_tcp_communication_use オペランドに N を指定しているため、TCP/IP を使用して RPC 要求できません。
	終了トランザクション (ME) からトランザクショナル RPC でサービスを要求しています。
	初期化トランザクション (MI)、または終了トランザクション (ME) から自プロセス通信 (通信レス) でサービスを要求しています。
00339	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられません。
00340	非同期応答型 RPC の場合は、rpc_nowait_cnt オペランドで指定した、同時要求最大数をオーバーしました。
00356	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。
00366	オンラインテストを使っている環境で、テストモードの UAP からテストモードではない SPP へサービスを要求しています。または、テストモードではない UAP からテストモードの SPP へサービスを要求しています。

ステータスコード	意味
00370	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
00372	同時に起動できるトランザクションブランチの数を越えたため、トランザクションブランチを開始できません。
	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を越えたため、トランザクションブランチを開始できません。
	リソースマネージャでエラーが発生したため、トランザクションブランチを開始できません。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の <code>rpc_reply_tplmode</code> オペランドまたは <code>rpc_reply_tplmode_down</code> オペランドの指定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の RPC 関連定義の <code>rpc_reply_tplmode_down</code> オペランド指定値が J の場合、サーバ側のトランザクションがロールバックしました。
00378	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。

CBLEERPC('CALL ')がエラーになる場合

CBLEERPC('CALL ')がエラーリターンする場合について説明します。

サービス要求先が稼働していない場合、または送信できるコネクションがない場合

CBLEERPC('CALL ')は「00305」でエラーリターンします。

サービス要求先コネクション確立後、サービス要求時にネットワーク障害を検知した場合

CBLEERPC('CALL ')は「00306」でエラーリターンします。

ネームサービスを使用しない場合で、サービスグループ情報関連定義の `eesvgdef` 定義コマンドにサーバ UAP のサービスグループ名が設定されていないとき

ネームサービスを使用しない場合は、サービスグループ情報関連定義の `eesvgdef` 定義コマンドの指定値を基にサービス要求先を取得します。`eesvgdef` 定義コマンドにサーバ UAP のサービスグループ名が設定されていない場合、CBLEERPC('CALL ')は「00310」でエラーリターンします。

ネームサービスを使用する場合で、該当するサービスグループ名が見つからなかったとき

ネームサービスを使用する場合は、RPC 関連定義の `rpc_destination_mode` オペランドの指定値に従ってサービス要求先を取得します。該当するサービスグループ名が見つからなかった場合、CBLEERPC('CALL ')は「00310」でエラーリターンします。

サービス要求時にメッセージ格納領域（TCP/IP バッファ）不足がリトライ回数を越えて連続で発生した場合

CBLEERPC('CALL ')は「00339」でエラーリターンします。サービス要求のリトライ回数は、RPC 関連定義の `rpc_router_retry_count` オペランドで指定します。サービス要求のリトライ間隔は、RPC 関連定

義の `rpc_router_retry_interval` オペランドで指定します。そのほか、RPC 関連定義の `ipc_sendbuf_size` オペランドやサービス要求先の TCP/IP バッファの定義（RPC 関連定義、TP1/Server Base のユーザサービス定義またはシステム共通定義の `ipc_recvbuf_size` オペランド）などを見直してください。

注意事項

1. データ領域に設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で設定しておいてください。
2. 次のトランザクションの場合は、自 TP1/EE のサービスグループ名を指定した RPC はできません。
 - 初期化トランザクション (MI)
 - 終了トランザクション (ME)
 - 共有リソース初期化トランザクション (JI)
 - システム開始時のユーザ初期化トランザクション (UT)
 - 実行系連絡トランザクション (UI) (ただし、データ名 C に 131072 を指定している場合だけ、自 TP1/EE のサービスグループ名を指定した RPC ができます)
3. 終了トランザクション (ME) では、トランザクショナル RPC はできません。
4. 初期化トランザクション (MI) からのトランザクショナル RPC は実行しないでください。通常起動（再起動ではない起動）で TP1/EE を起動した場合で、初期化トランザクションからのトランザクショナル RPC の処理中にプロセスダウンしたとき、TP1/EE は再び通常起動で起動されます。このとき、プロセスダウン前の仕掛けり中トランザクションは、回復処理の対象外となります。回復処理の対象外となったトランザクションは、サービス要求先でコマンドを実行することによって強制的に決着させることしかできなくなる場合があります。
5. 非同期応答型 RPC を連続して使用したあと応答を受け取る場合は、非同期応答型 RPC のサービスを要求するときに、それぞれ異なる応答格納領域（データ名 J）および応答の長さを格納する領域（`out_len`）を設定してください。同じ領域を設定すると、あとから来た応答に上書きされてしまい、応答を正しく受け取れなくなります。
6. **【TP1/FSP 限定】** 次のステータスコードの場合、出力メッセージ (OJ) は取得されません。
00001, 00004, 00005, 00301, 00308, 00331, 00332, 00334, 00336, 00337
7. 非同期応答型 RPC の応答格納領域は、次に示す方法で非同期応答型 RPC を終了するまで解放しないでください。
 - `CBLEERPC('POLLANYR')`関数で応答を受け取る
 - `CBLEERPC('DISCARD')`関数または `CBLEERPC('DISCARDS')`関数で応答の受信を拒否する
 - トランザクショナル RPC の場合、同期点処理の関数でコミットまたはロールバックする

CBLEERPC('CALLTO ')

名称

通信先を指定した遠隔サービスの要求

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2 一意名3 一意名4
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CALLTO ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC S9(9) COMP.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(n).  
01 一意名2.  
  02 データ名G PIC S9(9) COMP.  
  02 データ名H PIC X(n).  
01 一意名3.  
  02 データ名I PIC S9(9) COMP.  
  02 データ名J PIC X(n).  
01 一意名4.  
  02 データ名K PIC X(1).  
  02 FILLER PIC X(1).  
  02 データ名L PIC 9(4) COMP-X.  
  02 データ名M PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名N PIC X(n).
```

機能

特定の SPP のサービスを要求します。サービスを要求するには、ホスト名またはノード識別子に加え、「サービスグループ名」と「サービス名」を CBLEERPC('CALLTO ')のデータ領域に設定します。指定したホスト名またはノード識別子は、サービス要求先を特定する検索のキーとして使用します。この設定に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

CBLEERPC('CALLTO ')を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義 service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定するデータ領域

●データ名 A

遠隔サービスの要求を示す要求コードを「VALUE 'CALLTO△△」と設定します。

●データ名 C

RPC の形態とオプションを、数値で設定します。

0

同期応答型 RPC

1

非応答型 RPC

2

非同期応答型 RPC

トランザクション処理からのサービス要求で、要求先の処理をトランザクションにしない（要求先の処理を要求元と同一のグローバルトランザクションにしない）場合は、RPC の形態を示す数値に「32」を加算します。「32」を加算すると、トランザクションの処理からのサービス要求でも、サービスプログラムの処理はトランザクションになりません。

非同期応答型 RPC で、サービス関数が設定した応答の長さを参照したい場合は、RPC の形態を示す数値に「512」を加算します。その場合は、CBLEERPC('POLLANYR')の正常終了時に、CBLEERPC('CALLTO')のサービス応答とサービス応答の長さに、サービス関数が設定した応答と応答の長さがそれぞれ設定されます。「512」を加算しない場合は、サービス応答だけ設定されます。

(例)

非応答型 RPC でトランザクションにしないサービス要求をする場合

データ名 C には $1 + 32 = 33$ を設定します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerpch で定義してあります。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerpch で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●データ名 H

サービスの入力パラメタを設定します。

●データ名 I

サービスの応答の長さ（データ名 J の長さ）を設定します（単位：バイト）。1～EERPC_MAX_MESSAGE_SIZE_EX※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerpch で定義してあります。

非応答型 RPC の場合も、サービスの応答の長さを設定した領域のアドレスを指定する必要があります。この場合、サービスの応答の長さには、0 を設定します。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerpch で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●データ名 K

VALUE 'N'

データ名 L にネームサービスのポート番号を指定した場合に指定してください。

VALUE 'S'

サービス要求先（TP1/Server Base）のスケジュールサービスのポート、またはサービス要求先（TP1/EE）の mysvgdef オペランドで指定したポートに対してサービスを要求する場合に指定してください。

VALUE 'D'

上記以外の場合に指定してください。

●データ名 L

ホスト名を検索キーとする場合のポート番号（1～65535）を指定します。

データ名 K に VALUE 'N' を指定した場合、サービス要求先ノードのネームサービスのポート番号（TP1/Server Base のシステム共通定義の name_port オペランドに指定した値）を指定します。

データ名 K に VALUE 'D' を指定した場合は、サービス要求元のネームサービスのポート番号と、サービス要求先のネームサービスのポート番号が同じであることが前提となります。

データ名 K に VALUE 'S' を指定した場合、サービス要求先が TP1/Server Base の SPP のときは、要求先のスケジュールサービスのポート番号を指定します。サービス要求先が TP1/EE のときは、要求先の自サービスグループ定義の mysvgdef 定義コマンドで指定したポート番号を指定します。

データ名 K に VALUE 'S' を指定し、データ名 L に 0 を指定した場合は、送信先ポート番号の省略値として、最初に定義した自サービスグループ情報の mysvgdef 定義コマンドに指定したポート番号の値を仮定します。最初に定義した mysvgdef 定義コマンドのポート番号に 0 を指定している場合は、データ名 L に 0 を指定できません。

ノード識別子を検索のキーとする場合は、データ名 L に指定した値は無視されます。

●データ名 M

ノード識別子を検索のキーにする場合、サービス要求先のノード識別子を指定します。文字列の最後には空白文字を設定してください。この空白文字は文字列の長さに数えません。

ノード識別子は、OpenTP1 システム共通定義の node_id オペランドに指定した名称で、かつサービス要求先ノードのホスト名がグローバルドメイン内にあることが前提です。相手先が TP1/EE の場合も親 TP1/Server Base のノード識別子を指定してください。

ノード識別子を検索のキーにしない場合には、データ名 M に空白を指定してください。

データ名 K に VALUE 'S' を指定した場合は、データ名 M に空白を指定してください。

●データ名 N

ホスト名を検索のキーにする場合、サービス要求先ノードのホスト名を指定します。データ名 K に VALUE 'D' 以外を指定した場合は、データ名 N を必ず指定してください。

指定できるホスト名は 1～255 文字の文字列です。文字列に使用できる文字は、英数字、ピリオド、およびハイフンです。ただし、IP アドレス形式は除きます。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。指定するホスト名は、/etc/hosts ファイル、DNS など、IP アドレスとのマッピングができる名称です。

なお、サービス要求先ノードのホスト名はグローバルドメインに指定されていても、指定されていなくても、どちらでもかまいません。

ホスト名を検索キーにしない場合には、データ名 N に空白を指定してください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

非同期応答型 RPC の場合に、記述子が返されます。

サーバ UAP から値が返されるデータ領域

●データ名 I

サービスプログラムが設定した応答の長さ（データ名 J の長さ）が返されます。

●データ名 J

サービスプログラムが設定した応答が返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムから返される値ではありません。

ステータスコード	意味
00000	正常に終了しました。非同期応答型 RPC の場合は、データ名 D に記述子が設定されました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。 要求コード（データ名 A）が間違っている場合も含みます。 データ名 N に指定されたホスト名が、/etc/hosts ファイル、DNSなどで、IP アドレスとのマッピングができません。
00304	メモリが不足しました。要求先が TP1/EE の場合は、データ名 I の設定値が OMA サイズを超過しているおそれがあります。
00305	ネットワークの障害などによってサービス要求できるコネクションの取得に失敗しました。
00306	ネットワークに障害が起きました。
00307	処理が時間切れ（タイムアウト）になりました。
	サービスを要求された SPP が、処理を完了する前に異常終了しました。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tp1mode_down オペランド設定値が J の場合、サーバ側のトランザクション決着結果がロールバック以外です。
00308	データ名 G に設定した入力パラメタ長が、最大値を超えています。
00309	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。
00310	データ名 F に設定したサービスグループは、定義されていません。
	データ名 N で指定されたノード識別子が、グローバルドメイン（システム共通定義の all_node オペランドで指定したノード名の集合）内にありません。

ステータスコード	意味
00311	データ名 E に設定したサービス名は、定義されていません。 サービスの要求先が TP1/EE であり、かつ要求先の転送機能が無効（rpc_loadbalance オペランド設定値が N、かつ rpc_transfer_othersvg オペランド設定値が N）の場合は、サービスグループ名が不正のおそれがあります。
00312	データ名 E に設定したサービス名があるサービスグループは、閉塞しています。
00313	データ名 E に設定したサービスは、終了処理中です。
00314	データ名 E に設定したサービスの UAP プロセスが、稼働していません。 サービス要求の応答待ち時間に 0 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。
00315	データ名 E に設定したサービスがあるノードの TP1/Server Base または TP1/EE が稼働していません。異常終了、停止中、終了処理中、または通信障害が起こったことが考えられます。
00316	設定したサービスで、システムエラーが起こりました。
00317	設定したサービスで、メモリが不足しました。 非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランド設定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。次に示す原因が考えられます。 - UAP リターン後の同期点処理でトランザクション決着できない状態で、かつ rpc_replymsg_save オペランド設定値が N - UAP リターン後の同期点処理以降にプロセスダウン
00318	システムエラーが起こりました。
00319	サービスプログラムが TP1/EE に返した応答の長さが、1～EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
00320	サービスを要求されたノードにある TP1/Server Base または TP1/EE は、開始処理中です。
00323	メモリが不足しました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00324	システムエラーが起こりました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00325	設定したサービスで、システムエラーが起こりました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00326	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00327	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある TP1/Server Base または TP1/EE のバージョンが、クライアントの TP1/Server Base または TP1/EE のバージョンよりも古いため、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。このステータスコードは、ノード間負荷バランス機能およ

ステータスコード	意味
00327	びノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
00331	データ名 F に設定した値が間違っています。
00332	データ名 E に設定した値が間違っています。
00334	データ名 G に設定した値が間違っています。
00336	データ名 I に設定した値が間違っています。
00337	データ名 C に設定した値が間違っています。
00338	関数の呼び出し条件が不正です。 RPC 関連定義の name_use オペランドに Y が設定されていません。
00339	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられませんでした。
00340	非同期応答型 RPC（データ名 C に 2 を設定）の場合は、rpc_nowait_cnt オペランドで指定した同時要求最大数をオーバーしました。
00356	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。
00366	オンラインテストを使っている環境で、テストモードの UAP からテストモードでない SPP へサービスを要求しています。または、テストモードでない UAP からテストモードの SPP へサービスを要求しています。
00370	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。
00372	同時に起動できるトランザクションブランチの数を越えたため、トランザクションブランチを開始できません。
	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を越えたため、トランザクションブランチを開始できません。
	データ名 K に VALUE 'S'を指定し、非トランザクション属性（ユーザサービス定義の atomic_update オペランドに N を指定、またはシステム共通定義の jnl_fileless_option オペランドに Y を指定）の TP1/Server Base ユーザサーバに対し、データ名 C に 32 を加算しないでサービスを要求しました。
	リソースマネージャでエラーが発生したため、トランザクションブランチを開始できません。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tp1mode オペランドまたは rpc_reply_tp1mode_down オペランド設定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tp1mode_down オペランド設定値が J の場合、サーバ側のトランザクションがロールバックしました。
00378	リモート API 機能を使用した場合に、代理実行要求先で発生したエラーリターンです。エラーリターンの原因は、代理実行要求先のマニュアルを参照してください。

注意事項

1. ホスト名（データ名 N）とノード識別子（データ名 M）を同時に指定した場合、ホスト名の指定が有効になり、ノード識別子の指定は無視されます。
2. ホスト名（データ名 N）、ノード識別子（データ名 M）ともに空白を指定した場合、CBLEERPC('CALL ')と同様に動作します。
3. データ名 K に VALUE 'S'を指定し、ソケット受信型（ユーザサービス定義の receive_from オペランドに socket を指定）の TP1/Server Base ユーザサーバへサービスを要求した場合、CBLEERPC('CALLTO ')はステータスコード「00314」でリターンします。
4. データ名 K に VALUE 'S'を指定して CBLEERPC('CALLTO ')を呼び出した場合、rpc_retry オペランドは無効になります。
5. RPC 関連定義の name_use オペランドに N を設定して、CBLEERPC('CALLTO ')を呼び出した場合、ステータスコード「00338」でリターンします。
6. リモート API 機能を使用した RPC はできません。
7. **【TP1/FSP 限定】** 次のステータスコードの場合、出力メッセージ（OJ）は取得されません。
00001, 00004, 00005, 00301, 00308, 00331, 00332, 00334, 00336, 00337
8. その他の注意事項は、CBLEERPC('CALL ')の注意事項を参照してください。

CBLEERPC('CMTSND ')

名称

トランザクションと同期して送信する非応答型 RPC 要求

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CMTSND ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 FILLER PIC S9(9) COMP.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(n).  
01 一意名2.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).
```

機能

SPP にサービスを要求します。CBLEERPC('CMTSND ')を使うときは、要求するサービスがどのノードにあるかを意識する必要はありません。

サービスを要求するときには、「サービスグループ名」と「サービス名」を CBLEERPC('CMTSND ')のデータ領域に設定します。この名称に該当するサービスプログラムへサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

CBLEERPC('CMTSND ')を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定するデータ領域

●データ名 A

トランザクションと同期して送信する非応答型 RPC 要求を示す要求コードを「VALUE 'CMTSND△△」と設定します。

●データ名 C

次のフラグを設定します。

65536

ネームサービスを使用している場合、データ名 F に自サービスグループ名を設定したときに明示的に自プロセス送信（通信レス）をします。

0

上記以外

●データ名 E

SPP のサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します。1～EERPC_MAX_MESSAGE_SIZE_EX ※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●データ名 H

サービスの入力パラメタを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	要求コード（データ名 A）が間違っています。
00304	メモリが不足しました。
00308	データ名 G に設定した入力パラメタ長が、最大値を超えています。
00310	データ名 F に設定したサービスグループ名は、定義されていません。
00311	自プロセスの TP1/EE へのサービス要求時、データ名 E に設定したサービス名は、定義されていません。
00318	システムエラーが起きました。
00331	データ名 F に設定した値が間違っています。
00332	データ名 E に設定した値が間違っています。
00334	データ名 G に設定した値が間違っています。
00337	データ名 C に設定した値が間違っています。
00338	関数の呼び出し条件が不正です。
	rpc_tcp_communication_use オペランドに N を指定しているため、TCP/IP を使用して RPC 要求できません。

CBLEERPC('CMTSND ')がエラーになる場合

CBLEERPC('CMTSND ')がエラーリターンする場合について説明します。

初期化トランザクション (MI)、終了トランザクション (ME)、共有リソース初期化トランザクション (JI)、またはユーザ初期化トランザクション (UT) から RPC 要求された場合

CBLEERPC('CMTSND ')は「00338」でエラーリターンします。

すでに CBLEETRN('ROLLMARK')を呼び出している場合

CBLEERPC('CMTSND ')は「00338」でエラーリターンします。

ネームサービスを使用しない場合、または、ネームサービスを使用して自プロセス送信をする場合で、サービスグループ情報関連定義の mysvgdef 定義コマンドまたは eesvgdef 定義コマンドにサーバ UAP のサービスグループ名が設定されていないとき

ネームサービスを使用しない場合、または、ネームサービスを使用して自プロセス送信をする（データ名 C に 65536 を設定）場合は、サービスグループ情報関連定義の mysvgdef 定義コマンドまたは eesvgdef 定義コマンドの指定値を基にサービス要求先を取得します。mysvgdef 定義コマンドまたは eesvgdef 定

義コマンドにサーバ UAP のサービスグループ名が設定されていない場合、CBLEERPC('CMTSND ')は「00310」でエラーリターンします。

ネームサービスを使用する場合で、該当するサービスグループ名が見つからなかったとき

ネームサービスを使用する場合は、RPC 関連定義の rpc_destination_mode オペランドの指定値に従ってサービス要求先を取得します。該当するサービスグループ名が見つからなかった場合、CBLEERPC('CMTSND ')は「00310」でエラーリターンします。

ユーザサービス関連定義の service オペランドに自 TP1/EE のサービス名を指定しないで自プロセスの TP1/EE へ RPC 要求をした場合

CBLEERPC('CMTSND ')は「00311」でエラーリターンします。

注意事項

1. データ領域に設定するサービスグループ名およびサービス名は、サーバ UAP の環境設定で設定してください。
2. SPP へのサービス要求時にエラーが発生した場合、プロセス関連定義の rpc_cmtsend_retry オペランドの指定に従って、別処理スレッドによる再送信、またはエラートランザクション (ERRTRNS) の起動のどちらかの処理が実行されます。ただし、サービス要求元のトランザクションがエラートランザクションの場合は、エラーメッセージ出力後、これらの処理は実行されません。また、別処理スレッドによる再送信でエラーとなった場合は、エラートランザクション (ERRTRNS) が起動されます。
3. CBLEERPC('CMTSND ')を呼び出したトランザクションでは、トランザクションは最適化されません。
4. CBLEERPC('CMTSND ')の要求は、プロセスダウン時やスレッドダウン時の回復対象にはなりません。CBLEERPC('CMTSND ')の要求後に次の状態になったときは、回復処理でのトランザクション決着結果に関係なく、CBLEERPC('CMTSND ')の要求は破棄されます。
 - TP1/EE プロセスダウン
 - UAP 異常発生によるスレッドダウン
 - 通信障害やリソースマネージャ障害などによるトランザクション未決着
5. この関数は、トランザクションのコミット時に実行されます。したがって、トランザクションの同期点を取得する処理でエラーが起こった場合、この関数の処理は無効となります。
6. リモート API 機能を使用した RPC はできません。
7. **【TP1/FSP 限定】** 次のステータスコードの場合、出力メッセージ (OJ) は取得されません。
00001, 00004, 00005, 00301, 00308, 00331, 00332, 00334, 00337
8. 次の条件をすべて満たす場合は、自 TP1/EE のサービスグループ名を指定した RPC はできません。
 - 実行系連絡トランザクション (UI) である
 - name_use オペランドに Y を指定
 - データ名 C に 65536 を指定していない

CBLEERPC('CMTSNDTO')

名称

通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CMTSNDTO'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.  
  02 FILLER PIC S9(9) COMP.  
  02 データ名E PIC X(32).  
  02 データ名F PIC X(n).  
01 一意名2.  
  02 データ名G PIC 9(9) COMP.  
  02 データ名H PIC X(n).  
01 一意名3.  
  02 データ名K PIC X(1).  
  02 FILLER PIC X(1).  
  02 データ名L PIC 9(4) COMP-X.  
  02 データ名M PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名N PIC X(n).
```

機能

SPP のサービスを要求します。サービスを要求するには、ホスト名またはノード識別子に加え、「サービスグループ名」と「サービス名」を CBLEERPC('CMTSNDTO') のデータ領域に設定します。指定したホスト名またはノード識別子は、サービス要求先を特定する検索のキーとして使用します。この設定に該当するサービス関数へサービスが要求されます。ドメイン修飾をしてサービスを要求することはできません。

CBLEERPC('CMTSNDTO') を使う場合、サーバ UAP があるノードの TP1/Server Base または TP1/EE が稼働していることが前提です。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、または、ユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定するデータ領域

●データ名 A

遠隔サービスの要求を示す要求コードを「VALUE 'CMTSNDTO'」と設定します。

●データ名 C

0 を設定します。

●データ名 E

SPP のサービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 F

SPP のサービスグループ名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

●データ名 G

サービスの入力パラメタ長（データ名 H の長さ）を設定します。1～EERPC_MAX_MESSAGE_SIZE_EX ※の範囲の値を指定できます。EERPC_MAX_MESSAGE_SIZE_EX は、eerp.h で定義してあります。

注※

従来の EERPC_MAX_MESSAGE_SIZE は、eerp.h で定義してありますが、
EERPC_MAX_MESSAGE_SIZE_EX の使用をお勧めします。

●データ名 H

サービスの入力パラメタを設定します。

●データ名 K

指定値は CBLEERPC('CALLTO ')のデータ名 K を参照してください。

●データ名 L

指定値は CBLEERPC('CALLTO ')のデータ名 L を参照してください。

●データ名 M

指定値は CBLEERPC('CALLTO ')のデータ名 M を参照してください。

●データ名 N

指定値は CBLEERPC('CALLTO ')のデータ名 N を参照してください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムから返される値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	要求コード（データ名 A）が間違っています。 データ名 N に指定されたホスト名が、/etc/hosts ファイル、DNSなどで、IP アドレスとのマッピングができません。
00304	メモリが不足しました。
00308	データ名 G に設定した入力パラメタ長が、最大値を超えています。
00310	データ名 F に設定したサービスグループ名は、定義されていません。
00318	システムエラーが起きました。
00331	データ名 F に設定した値が間違っています。
00332	データ名 E に設定した値が間違っています。
00334	データ名 G に設定した値が間違っています。
00337	データ名 C に設定した値が間違っています。
00338	関数の呼び出し条件が不正です。 RPC 関連定義の name_use に Y が設定されていません。

注意事項

1. ホスト名（データ名 M）とノード識別子（データ名 N）を同時に指定した場合、ホスト名の指定が有効になり、ノード識別子の指定は無視されます。
2. ホスト名（データ名 N）、ノード識別子（データ名 M）ともに空白を指定した場合は、CBLEERPC('CMTSND ')と同様に動作します。
3. データ名 K に VALUE 'S'を指定し、ソケット受信型（ユーザサービス定義の receive_from オペランドに socket を指定）の TP1/Server Base ユーザサーバへサービスを要求した場合、CBLEERPC('CMTSNDTO')は、送信エラーとなります。

4. データ名 K に VALUE 'S'を指定して CBLEERPC('CMTSNDTO')を呼び出した場合、rpc_retry オペランドは無効になります。
5. RPC 関連定義の name_use オペランドに N を設定して、CBLEERPC('CMTSNDTO')を呼び出した場合、ステータスコード「00338」でリターンします。
6. **【TP1/FSP 限定】** 次のステータスコードの場合、出力メッセージ（OJ）は取得されません。
00001, 00004, 00005, 00301, 00308, 00331, 00332, 00334, 00337
7. その他の注意事項は、CBLEERPC('CMTSND ')の注意事項を参照してください。

CBLEERPC('DISCARDF')

名称

すべての非同期応答型 RPC の応答受信拒否

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'DISCARDF'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

非同期応答型 RPC を使った場合に、まだ返ってきていない応答を、これ以上受信しないことを宣言します。CBLEERPC('DISCARDF')を呼び出したあとは、応答が返ってきても受信しないで捨てられます。

非同期応答型 RPC の応答待ちのデータを受信しない場合は、必ず CBLEERPC('DISCARDF')を呼び出してください。呼び出さないと、CBLEERPC('POLLANYR')が不要な応答を受信してしまうことがあります。

CBLEERPC('DISCARDF')は、次に示す場合に呼び出してください。

- ・ 応答待ち時間切れになったあと、結果を保持しておくバッファを解放する場合
- ・ 非同期応答型 RPC を複数回使って、そのうちの最初の応答だけ必要な場合

UAP で値を設定するデータ領域

●データ名 A

処理結果の受信を拒否することを示す要求コードを「VALUE 'DISCARDF'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00338	発行条件が不正です。

CBLEERPC('DISCARDS')

名称

特定の非同期応答型 RPC の応答受信拒否

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'DISCARDS'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC S9(9) COMP.
```

機能

非同期応答型 RPC を使った場合、まだ返ってきていない特定の応答を、これ以上受信しないことを宣言します。受信を拒否する非同期応答を特定するには、データ名 D に、非同期応答型 RPC がリターンした際に返した記述子を設定します。CBLEERPC('DISCARDS')を呼び出したあとに返ってきた応答の中で、設定した記述子と同じ記述子を持つ応答は受信しないで捨てられます。

UAP で値を設定するデータ領域

●データ名 A

特定の処理結果の受信を拒否することを示す要求コードを「VALUE 'DISCARDS'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

非同期応答型 RPC の CBLEERPC('CALL ')（データ名 C に 2 を設定）が正常に終了したときに返された、記述子を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00318	システムエラーが起きました。
00322	データ名 D に指定した記述子は存在しません。要因として次のことが考えられます。 - 設定した記述子に対応する非同期応答型 RPC を行っていない。 - 非同期応答型 RPC の応答がすでに受信されている、または受信が拒否されている。
00338	発行条件が不正です。

CBLEERPC('GETERDES')

名称

エラーが発生した非同期応答型 RPC の識別子取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GETERDES'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC S9(9) COMP.
```

機能

非同期応答を特定しない CBLEERPC('POLLANYR')がエラーリターンした直後に呼び出すことで、エラーが発生した非同期応答型 RPC 要求に対応する記述子を取得します。

記述子を取得できるのは、サーバ UAP 側でエラーが発生した場合だけです。

CBLEERPC('POLLANYR')の呼び出し側でエラーが発生した場合は、CBLEERPC('GETERDES')で記述子を取得できません。

UAP で値を設定するデータ領域

●データ名 A

記述子取得を示す要求コードを「VALUE 'GETERDES'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

記述子が返されます。記述子を取得できない場合は、0 が設定されます。

ステータスコード

ここで示すステータスコードは，TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため，この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため，この機能は使用できません。
00301	データ名に設定した値が間違っています。
00338	発行条件が不正です。

CBLEERPC('GETWATCH')

名称

サービスの応答待ち時間の参照

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GETWATCH'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP.
```

機能

現在のサービス要求の応答待ち時間を参照します（単位：秒）。CBLEERPC('GETWATCH')は、CBLEERPC('SETWATCH')で応答待ち時間を一時的に変更する前に、元の値を退避するために使います。

CBLEERPC('GETWATCH')は、CBLEERPC('SETWATCH')で変更したサービス応答待ち時間をリターンします。CBLEERPC('SETWATCH')でサービス応答待ち時間を変更していない場合は RPC 関連定義の watch_time オペランドの値をリターンします。

CBLEERPC('GETWATCH')で得られる値は、TP1/EE のリモートプロシジャコール（CBLEERPC('CALL ')）に対してだけ有効です。

UAP で値を設定するデータ領域

●データ名 A

サービスの応答待ち時間の参照を示す要求コードを「VALUE 'GETWATCH'」と設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 C

現在のサービスの応答待ち時間が返されます（単位：秒）。0 が返された場合は、無制限に応答を待ち続ける指定であることを示します。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00338	関数の呼び出し条件が不正です。

CBLEERPC('POLLANYR')

名称

非同期応答型 RPC の応答受信

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'POLLANYR'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
  02 データ名D PIC S9(9) COMP.  
  02 データ名E PIC S9(9) COMP.  
  02 データ名F PIC S9(9) COMP.
```

機能

非同期応答型 RPC でサービス要求した結果を受信します。

受信する非同期応答を特定する場合は、データ名 C に 1, または 17 を設定します。この値を設定した場合は、データ名 E に設定した記述子をリターンした、非同期応答型 RPC の応答を受信します。

受信する非同期応答を特定しない場合は、データ名 C に 0, または 16 を設定します。このとき、データ名 E に設定した値は無視されます。データ名 C に 0, または 16 を設定した CBLEERPC('POLLANYR') が正常に終了すると、受信した非同期応答の記述子と同じ値がリターンされます。

CBLEERPC('POLLANYR')は、次のどちらかの場合にリターンします。

- ・ 非同期応答型 RPC の応答を受信した場合
- ・ データ名 F に設定した応答待ち時間が終了した場合

CBLEERPC('POLLANYR')が正常に終了すると、非同期応答型 RPC を使った CBLEERPC('CALL ')の応答を格納する領域に、値が設定されます。

UAP で値を設定するデータ領域

●データ名 A

処理結果を非同期に受信することを示す要求コードを「VALUE 'POLLANYR'」と設定します。

●データ名 C

次のどれかを設定します。

0

待ち時間を設定します（単位：秒）。CBLEERPC('POLLANYR')で受信する非同期の応答を特定しません。

1

待ち時間を設定します（単位：秒）。データ名 E に設定された記述子をリターンした、非同期応答型 RPC の応答を受信します。

16

待ち時間を設定します（単位：ミリ秒）。CBLEERPC('POLLANYR')で受信する非同期の応答を特定しません。

17

待ち時間を設定します（単位：ミリ秒）。データ名 E に設定された記述子をリターンした、非同期応答型 RPC の応答を受信します。

●データ名 E

非同期応答型 RPC の CBLEERPC('CALL ')（データ名 C に 2 を設定）が正常に終了したときに返された、記述子を設定します。データ名 C に 0、または 16 を設定した場合は、ここに設定した値は無視されます。

●データ名 F

CBLEERPC('POLLANYR')を呼び出してから、応答が返ってくるまでの待ち時間を設定します（単位：秒またはミリ秒）。設定できる値は-1 から S9(9) COMP で表せる最大の数までです。

CBLEERPC('POLLANYR')で非同期応答を受信する場合は、RPC 関連定義の watch_time オペランドまたは CBLEERPC('SETWATCH')で設定した応答待ち時間を参照しません。

0 を設定した場合は、データ名 C が 0、または 1 を設定したとき応答が返ってこないと「00307」ですぐにエラーリターンします。データ名 C に 16、または 17 を設定したとき、50 ミリ秒として処理されます。

-1 を設定した場合は、応答が返るまで待ち続けます。

【TP1/FSP 限定】 次のすべての条件を満たす場合、応答待ち時間は 2,147,483 秒となります。

- このオプションの設定値が 2147484 以上
- 待ち時間を秒単位で指定（データ名 C に 17 の設定なし）
- RPC 関連定義の rpc_reply_watch_interval 設定値が 1 以上

●データ名 D

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

受信した非同期応答の記述子が返されます。この記述子は、データ名 C に 0、または 16 を設定した CBLEERPC('POLLANYR')が正常に終了した場合に返されます。データ名 C に 1、または 17 を設定した CBLEERPC('POLLANYR')が正常に終了した場合は、0 が設定されます。

ステータスコード

ここで示すステータスコードは、TP1/EE が返す値です。サービスプログラムが返す値ではありません。

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00304	メモリが不足しました。
00306	ネットワークに障害が起きました。
00307	CBLEERPC('CALL ')の処理が時間切れ（タイムアウト）になりました。
	サービスを要求された SPP が、処理を完了する前に異常終了しました。
	CBLEERPC('CALL ')の非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランドの設定値が J の場合、サーバ側のトランザクションがロールバック以外です。
00309	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。
00310	CBLEERPC('CALL ')のデータ名 F に設定したサービスグループ名は定義されていません。
00311	CBLEERPC('CALL ')のデータ名 E に設定したサービス名は、定義されていません。
	サービスの要求先が TP1/EE であり、かつ要求先の転送機能が無効（rpc_loadbalance オペランドの設定値が N、かつ rpc_transfer_othersvg オペランドの設定値が N）の場合は、サービスグループ名不正のおそれがあります。
00312	CBLEERPC('CALL ')のデータ名 E に設定したサービス名があるサービスグループは、閉塞しています。
00313	CBLEERPC('CALL ')のデータ名 E に設定したサービスは、終了処理中です。
00314	CBLEERPC('CALL ')のデータ名 E に設定したサービスの UAP プロセスが、稼働していません。

ステータスコード	意味
00314	データ名 F に-1 を指定した場合に、サービスを要求された SPP が、処理を完了する前に異常終了しました。
00315	CBLEERPC('CALL ')のデータ名 E に設定したサービスがあるノードの OpenTP1 が稼働していません。異常終了、停止中、終了処理中、および通信障害が起こったことが考えられます。
00316	CBLEERPC('CALL ')に設定したサービスで、システムエラーが起こりました。
00317	CBLEERPC('CALL ')に設定したサービスで、メモリが不足しました。 CBLEERPC('CALL ') で非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランド設定値が N の場合、サーバ側のトランザクションがコミットしましたが、応答を返せませんでした。応答を返せなかった原因を次に示します。 • UAP リターン後の同期点処理でトランザクション決着できない状態であり、かつ rpc_replymsg_save オペランドの設定値が N • UAP リターン後の同期点処理以降にプロセスダウン
00318	システムエラーが起こりました。
00319	サービス関数が OpenTP1 に返した応答の長さが、1～EERPC_MAX_MESSAGE_SIZE_EX で定義されている値の範囲にありません。
00320	サービスを要求されたノードにある OpenTP1 は、開始処理中です。
00321	非同期応答型 RPC で要求したサービスの処理結果は、すべて受信しました。
00322	データ名 E に設定した記述子は存在しません。このステータスコードは、データ名 C に 1 または 17 を設定した場合に返されます。
00323	メモリが不足しました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00324	システムエラーが起こりました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00325	設定したサービスで、システムエラーが起こりました。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00326	返ってきた応答が、クライアント UAP で用意した領域に入り切りません。このステータスコードが返った場合は、トランザクションブランチをコミットできません。
00327	ノード間負荷バランス機能およびノード間負荷バランス拡張機能の環境で、複数の SPP のトランザクション属性が一致していません。または、負荷を分散する先のノードにある OpenTP1 のバージョンが、クライアントの OpenTP1 のバージョンよりも低いため、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を実行できません。 このステータスコードは、ノード間負荷バランス機能およびノード間負荷バランス拡張機能を使っている SPP にサービスを要求した場合にだけリターンされます。
00338	発行条件が不正です。
00356	サービスを要求されたソケット受信型サーバが、サービス要求を受け取れません。

ステータスコード	意味
00366	オンラインテストを使っている環境で、テストモードの UAP からテストモードでない SPP へサービスを要求しています。または、テストモードでない UAP からテストモードの SPP へサービスを要求しています。
00372	一つのトランザクションブランチから開始できる子トランザクションブランチの最大数を超えたため、トランザクションブランチを開始できません。
	CBLEERPC('CALL ')で非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode オペランドまたは rpc_reply_tplmode_down オペランドの設定値が N の場合、サーバ側のトランザクションがコミットできませんでした。
	CBLEERPC('CALL ')で非トランザクショナル RPC（データ名 C に 32 以上を設定）の要求先サーバが TP1/EE であり、かつサーバ側の rpc_reply_tplmode_down オペランドの設定値が J の場合、サーバ側のトランザクションがロールバックしました。

注意事項

1. 記述子を特定しない CBLEERPC('POLLANYR')がエラーリターンした場合、エラーとなった応答の記述子を特定できません。CBLEERPC('POLLANYR')がエラーリターンしたときに該当する記述子がかかるようにしておきたい場合は、データ名 C に 1、または 17 を設定しておいてください。または、CBLEERPC('GETERDES')を使用することで、エラーリターンした記述子を取得できることもあります。
2. 別スレッドで送信した非同期応答型 RPC（CBLEERPC('CALL ')（データ名 C に 2 を設定））は、この COBOL-UAP 作成用プログラムで受信できません。非同期応答型 RPC は同一スレッドで CBLEERPC('POLLANYR')を使用して受信する必要があります。
3. 非同期応答型 RPC でサービスを要求した UAP が次に示す COBOL-UAP 作成用プログラムを呼び出すと、CBLEERPC('POLLANYR')で応答を受け取れません。
 - CBLEERPC('DISCARD')で、非同期応答の受信を拒否した場合
 - トランザクショナル RPC の場合、同期点処理の COBOL-UAP 作成用プログラムでコミットまたはロールバックした場合

上記の COBOL-UAP 作成用プログラムを使ったあとで返ってきた応答は、破棄されます。非同期応答型 RPC では、上記の COBOL-UAP 作成用プログラムを呼び出す前に、必要な非同期の応答を CBLEERPC('POLLANYR')ですべて受け取ってください。

CBLEERPC('R-SEND ')

名称

抑止していた RPC 応答メッセージの送信

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'R-SEND '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC 9(9) COMP-X.  
01 一意名3  
  02 データ名D PIC 9(9) COMP.  
  02 データ名E PIC X(n).
```

機能

CBLEERPC('R-SEND ')で抑止していた RPC 応答メッセージを送信します。

【TP1/FSP 限定】 RPC 関連定義の rpc_output_message オペランドが設定されている場合、またはユーザサービス関連定義の service_attr 定義コマンドが設定されている場合に、送信データを UAP 履歴情報のデータ種別「OJ（出力メッセージ）」として取得します。

オペランドについては、TP1/FSP の関連ドキュメントを参照してください。

UAP で値を設定するデータ領域

●データ名 A

抑止していた RPC 応答メッセージの送信を示す要求コードを「VALUE 'R-SEND△△'」と設定します。

●データ名 Z

次の値を設定します。

0

この API の延長で RPC 応答メッセージを送信します。

1

この API を発行したトランザクションがコミット決着した時点で RPC 応答メッセージを送信します。
トランザクションがロールバック決着した場合は RPC 応答メッセージを送信せずに破棄します。

2

この API を発行したトランザクションがコミット決着した時点で RPC 応答メッセージを送信します。
トランザクションがロールバック決着した場合は、RPC エラー応答メッセージ
(EERPCER_TRNCHK_EXTEND) を送信します。

●データ名 C

抑止識別子を設定します。CBLEERPC('R-SPEND ')で割り当てた抑止識別子を設定してください。

●データ名 D

サービスプログラムからの応答を設定します。

●データ名 E

サービスプログラムからの、実際の応答の長さを設定します（単位：バイト）。クライアント UAP から渡されたサービスの応答の長さ以下の数値を設定してください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00304	メモリが不足しました。
00305	コネクションの取得に失敗しました。
00306	ネットワークに障害が起きました。
00318	システムエラーが起きました。
00336	データ名 D に設定した値が間違っています。
00338	関数の呼び出し条件が不正です。

ステータスコード	意味
00339	サービスを要求された SPP のメッセージ格納領域に十分な空きがないため、サービス要求を受け付けられません。
00341	データ名 C に設定した値は未登録か、すでにタイムアウトなどで応答が返されています。

注意事項

1. 一つの抑止識別子で複数の RPC 応答メッセージの送信を抑止している場合、一度の CBLEERPC('R-SEND ')で複数の RPC 応答メッセージが送信されます。
2. **【TP1/FSP 限定】** 次のステータスコードの場合、出力メッセージ (OJ) は取得されません。
00001, 00004, 00005, 00301, 00336
3. CBLEERPC('R-SEND ')のデータ名 Z に 1 を指定し、かつ、そのトランザクションがロールバック決着した場合、RPC 応答メッセージは送信されません。そのため、RPC 要求元のクライアント UAP は、応答待ち時間を指定している場合は応答待ちタイムアウト、応答待ち時間を指定していない場合は応答待ちを無限に続けます。
4. CBLEERPC('R-SEND ')のデータ名 Z に 1 または 2 を指定し、かつ、そのトランザクションがリラン後にコミット決着した場合、RPC エラー応答メッセージ (EERPCER_NO_BUFS_AT_SERVER) を送信します。ロールバック決着した場合、データ名 Z に 1 を指定したときは RPC 応答メッセージを破棄し、データ名 Z に 2 を指定したときは RPC エラー応答メッセージ (EERPCER_TRNCHK_EXTEND) を送信します。
5. CBLEETRAN('ROLLMARK')発行済みの状態で、CBLEERPC('R-SEND')のデータ名 Z に 1 または 2 を指定しても、CBLEERPC('R-SEND')は正常に終了します。その後、トランザクションがロールバック決着した時点で、データ名 Z に 1 を指定したときは RPC 応答メッセージを破棄し、データ名 Z に 2 を指定したときは RPC エラー応答メッセージ (EERPCER_TRNCHK_EXTEND) を送信します。

CBLEERPC('R-SPEND ')

名称

RPC 応答メッセージ送信の抑止

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'R-SPEND'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(1).  
  02 FILLER PIC X(3).  
  02 データ名D PIC 9(9) COMP-X.  
  02 データ名E PIC S9(9) COMP.
```

機能

サービスプログラム終了時の RPC 応答メッセージ送信を抑止します。抑止した RPC 応答メッセージは、CBLEERPC('R-SEND ')を使って明示的に送信してください。トランザクションブランチ内では RPC 応答メッセージを抑止できません。

UAP で値を設定するデータ領域

●データ名 A

RPC 応答メッセージ送信の抑止を示す要求コードを「VALUE 'R-SPEND△'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

抑止識別子の設定方法を設定します。

VALUE 'S'

TP1/EE が抑止識別子を割り当てます。

VALUE 'U'

ユーザが抑止識別子を指定します。

●データ名 D

データ名 C に VALUE 'U'を設定した場合に、抑止識別子を設定します。抑止識別子は、1～2147483647 の値を指定してください。

●データ名 E

RPC 応答メッセージ抑止限界時間を設定します。指定できる範囲は-1，0～65535 です。

-1 を指定した場合、RPC 関連定義の rpc_reply_suspend_time オペランドで指定した値が抑止限界時間になります。0 を設定した場合は、抑止限界時間は無制限になります。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

データ名 C に VALUE 'S'を設定した場合に、抑止識別子が返されます。抑止識別子は、TP1/EE が自動的に割り当てた、プロセス内で一意の識別子です。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下がありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00302	トランザクションブランチ内で CBLEERPC('R-SPEND ')を呼び出しています。
	非応答型 RPC で CBLEERPC('R-SPEND ')を呼び出しています。
	リモート API 機能による自サービスグループへの代理実行要求中に CBLEERPC('R-SPEND ')を呼び出しています。
	コネクションレス（UDP プロトコル）で起動されたトランザクション内で関数を発行しています。
	RPC メッセージの永続化機能を有効にした状態で関数を発行しています。
00318	システムエラーが起きました。
00338	関数の呼び出し条件が不正です。
00340	定義で指定した同時抑止最大数を超えました。
00342	RPC 応答メッセージ送信は応答抑止済みです。

注意事項

1. CBLEERPC('R-SPEND ')を呼び出したあと、RPC 応答メッセージ抑止限界時間以内に CBLEERPC('R-SEND ')を呼び出さなかった場合、RPC エラー応答メッセージ「00307」が自動的に送信されます。
そのあとで CBLEERPC('R-SEND ')を呼び出しても、CBLEERPC('R-SEND ')はエラーになります。
2. 同一の抑止識別子を複数の CBLEERPC('R-SPEND ')で指定できます。この場合、一度の CBLEERPC('R-SEND ')で複数の RPC 応答メッセージが送信されます。
3. 非応答型 RPC で CBLEERPC('R-SPEND ')を呼び出した場合は、エラーリターンします。
4. **【XTC 限定】** コネクションレス（UDP プロトコル）で起動されたトランザクション内でこの関数を発行した場合は、エラーリターンします。

CBLEERPC('SETWATCH')

名称

サービスの応答待ち時間の更新

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEERPC' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'SETWATCH'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

サービス要求の応答待ち時間を変更します（単位：秒）。CBLEERPC('SETWATCH')で変更した値は、サービスプログラム終了時まで有効です。

サービス要求の応答待ち時間を CBLEERPC('SETWATCH')を呼び出す前の値に戻すときは、事前に呼び出した CBLEERPC('GETWATCH')で返された元の値を、CBLEERPC('SETWATCH')で再設定してください。

CBLEERPC('SETWATCH')を呼び出しても、RPC 関連定義の watch_time の値に指定した値は変更されません。CBLEERPC('SETWATCH')に設定した値は、あとから呼び出す CBLEERPC('CALL ')にだけ影響します。

UAP で値を設定するデータ領域

●データ名 A

サービスの応答待ち時間の変更を示す要求コードを「VALUE 'SETWATCH'」と設定します。

●データ名 C

変更後のサービス応答待ち時間を設定します（単位：秒）。1～65535 の範囲で指定してください。無制限に待ち続ける場合は、0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00301	データ名に設定した値が間違っています。
00338	関数の呼び出し条件が不正です。

注意事項

応答待ち時間に 0 を指定した場合、TP1/EE が終了しないことがあります。

滞留メッセージ受信 (CBLEESCD)

滞留メッセージを UAP 起動中に受信するときに使用するプログラムについて説明します。滞留メッセージ受信の COBOL-UAP 作成用プログラムを次に示します。

- CBLEESCD('CLRRBCNT')【XTC 限定】 – ロールバックリトライ回数のクリア
- CBLEESCD('MSGRECV ')【XTC 限定】 – 滞留メッセージの受信
- CBLEESCD('MSGRLBK ')【XTC 限定】 – 受信メッセージの差し戻し
- CBLEESCD('MSGSKIP ')【XTC 限定】 – 滞留メッセージのスキップ

CBLEESCD('CLRRBCNT')【XTC 限定】

名称

ロールバックリトライ回数のクリア

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEESCD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CLRRBCNT'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(32).
```

機能

ロールバックリトライ回数を 0 にクリアします。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'CLRRBCNT'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

サービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00801	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00802	データ名 C に設定したサービス名は、定義されていません。
00804	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
00881	この関数を発行できる状態ではありません。

CBLEESCD('CLRRBCNT')がエラーになる場合

CBLEESCD('CLRRBCNT')がエラーリターンする場合について説明します。

プロセス関連定義の xtc_use オペランドに Y を指定していない場合

CBLEESCD('CLRRBCNT')は「00804」でエラーリターンします。

service に設定したサービスのサービス属性定義に、service_attr -e serial を指定していない場合

CBLEESCD('CLRRBCNT')は「00804」でエラーリターンします。

この関数を発行したサービスと service に設定したサービスが異なり、他スレッドで service に設定したサービスのトランザクションが実行中の場合

CBLEESCD('CLRRBCNT')は「00881」でエラーリターンします。

注意事項

1. この関数を発行したサービスとデータ名 C に設定したサービスが同じで、トランザクションがロールバック決着した場合、ロールバックリトライ回数はカウントしないで、0 になります。XTC のロールバックリトライ回数監視によるサービス閉塞は行いません。
2. この関数を発行したサービスとデータ名 C に設定したサービスが異なる場合、またはこの関数を発行したトランザクションがエラートランザクションの場合は、処理キュー全体の排他を取ります。そのため、この関数を頻繁に使用するとスループットが悪くなるおそれがあります。

CBLEESCD('MSGRECV ')【XTC 限定】

名称

滞留メッセージの受信

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEESCD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGRECV'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC 9(9) COMP VALUE ZERO.  
  02 データ名D PIC 9(9) COMP VALUE ZERO.  
  02 データ名E ADDRESS.  
  02 データ名F ADDRESS.
```

機能

入力キュー (ITQ) に滞留しているメッセージを受信します。

UAP で値を設定するデータ領域

●データ名 A

滞留しているメッセージの受信を示す要求コードを「VALUE 'MSGRECV△」と設定します。

●データ名 Z

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 C

入力メッセージ長が返されます。

●データ名D

受信したメッセージのメッセージ通番が返されます。

●データ名E

入力メッセージ引き渡しエリアのアドレスが返されます。

●データ名F

入力メッセージ情報が返されます。

ただし、受信メッセージが MCP メッセージの場合だけアドレスが返されます。

アドレスが示す領域トランザクション起動時に MCP エリアポインタが示す構造体を参照してください。

また、トランザクション起動時に MCP エリアポインタが示す領域を使用するため、以前の情報は参照できなくなります。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00801	引数に設定した値が間違っています。
00804	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
00806	入力メッセージ引き渡しエリア（IMA）がオーバーフローしました。※ ¹
00860	滞留メッセージはありません。※ ²
00861	滞留メッセージ中に同種のメッセージがありません。
00862	読み出し上限を超えました。
00870	中間同期点処理を行ったため、滞留メッセージの受信はできません。
00875	この関数を発行できるサービスではありません（サービス属性定義で service_attr -e serial を指定していないサービスでこの関数を発行）。または、トランザクション種別が MN もしくは TM ではありません。
00876	トランザクション起動時のメッセージが滞留メッセージ読み出し対象メッセージではありません。
00881	スキップ処理中のため読み出せません。

注※1

メッセージ格納領域に格納できるサイズ分のメッセージは格納されています。

データ名 C には格納されているメッセージサイズが設定されています。

注※2

受信処理中の入力キュー（優先/通常）には滞留メッセージがない場合に設定します。優先キューの受信処理中に優先メッセージをすべて受信処理した場合、通常キューに滞留メッセージがあってもステータスコードは「00860」となります。

注意事項

1. 滞留メッセージを受信できるのは、次のメッセージ種別がトランザクション起動時に渡されたメッセージ種別と同じメッセージの場合です。
なお、タイマトランザクションの場合は、永続タイマと非永続タイマは同じメッセージ種別とは見なされません。
2. トランザクション起動時に渡されたメッセージ種別が RPC 受信メッセージであり、かつ応答型 RPC またはトランザクショナル RPC の場合は、滞留しているメッセージを受信できません。
また、リモート API 機能（RAP）で送信されたメッセージも受信できません。
3. トランザクション起動時に引き出したキュー（優先キューまたは通常キュー）に滞留しているメッセージだけ受信できます。
例えば最初に引き出したキューが優先キューの場合、通常キューにメッセージが滞留していてもこの関数では受信できません。
4. 入力メッセージ引き渡しエリアおよび入力メッセージサイズ格納領域はトランザクション起動時に使用した領域を再使用します。この関数発行後に前のメッセージおよびメッセージサイズを参照する場合は領域の内容を退避したあと、この関数を発行してください。
5. トランザクション起動時のトランザクション種別が MN または TM の場合だけ、この関数を発行できます。
6. 該当サービスの同時引き出しの可否（service_attr -e）に serial を指定した場合だけ、この関数を発行できます。
7. 滞留メッセージを受信できる最大数は、トランザクション起動時に滞留していたメッセージ数です。
8. 中間同期点処理（CBLEETRN('C-ROLL ')発行）を行ったあとは、この関数を発行できません。

CBLEESCD('MSGRLBK ')【XTC 限定】

名称

受信メッセージの差し戻し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEESCD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A   PIC X(8)  VALUE 'MSGRLBK'.  
  02 データ名B   PIC X(5).  
  02 FILLER      PIC X(3).  
  02 データ名Z   PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C   PIC 9(9)  COMP VALUE ZERO.
```

機能

トランザクションの終了状態に関係なく、滞留メッセージの受信関数（CBLEESCD('MSGRECV ')関数）で読み出したメッセージを差し戻します（ロールバックさせます）。

UAP で値を設定するデータ領域

●データ名 A

受信メッセージ差し戻しを示す要求コードを「VALUE 'MSGRLBK△'」と設定します。

●データ名 C

差し戻しを行いたいメッセージのメッセージ通番を設定します。

●データ名 Z

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00801	引数に設定した値が間違っています。
00804	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
00871	指定したメッセージ通番が不正です（通番範囲外、または CBLEESCD('MSGRECV ')関数未発行）。
00875	この関数を発行できるサービスではありません（サービス属性定義で service_attr -e serial を指定していないサービスでこの関数を発行）。または、トランザクション種別が MN もしくは TM ではありません。
00876	トランザクション起動時のメッセージが滞留メッセージ読み出し対象メッセージではありません。

注意事項

1. この関数は、トランザクション実行中に複数回発行できますが、最後に発行した関数で指定したメッセージ通番だけが有効となります。
2. CBLEESCD('MSGRECV ')関数を複数回発行した場合、この関数で指定したメッセージ通番以降の全メッセージが差し戻されます。
3. トランザクション起動時に渡されたメッセージはこの関数で差し戻すことはできません。差し戻しを行いたい場合はトランザクションをロールバックさせてください。

CBLEESCD('MSGSKIP ')【XTC 限定】

名称

滞留メッセージのスキップ

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEESCD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'MSGSKIP'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC 9(9) COMP VALUE ZERO.  
  02 データ名D PIC X(4) VALUE ' '.  
  02 データ名E PIC X(1) VALUE 'A'.  
  02 データ名F PIC X(32).
```

機能

入力キュー (ITQ) に滞留しているメッセージを読み出し済みとして入力キューから破棄します。

指定したサービス名に対応する入力キューのメッセージを読み出し済みとします。スキップ要求は、該当サービスの処理キューが引き出し禁止状態、かつ該当サービスのトランザクションが動作していない場合に受け付けられます。

UAP で値を設定するデータ領域

●データ名 A

サービスの要求を示す要求コードを「VALUE 'MSGSKIP△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

0 を設定します。

●データ名 D

VALUE '△△△△'を設定します。

●データ名 E

次の値を設定します。

VALUE 'A'

入力キューに滞留しているすべてのメッセージを読み出し済みにします。

●データ名 F

サービス名を、31 バイト以内のアスキー文字列で設定します。文字列の最後には空白を設定してください。この空白は文字列の長さに数えません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00801	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00802	データ名 F に設定したサービス名は、定義されていません。
00803	リソースが不足しました。
00804	発行条件が不正です（プロセス関連定義の xtc_use オペランドに N を指定してこの関数を発行）。
00881	この関数を発行できる状態ではありません。
00899	システムエラーが起きました。

注意事項

スキップの対象となるメッセージがない場合は、正常扱いになります。

スレッド制御 (CBLEETHD)

スレッドを制御するプログラムについて説明します。スレッド制御の COBOL-UAP 作成用プログラムを次に示します。

- CBLEETHD('ABDCTL ') – サービス閉塞の制御
- CBLEETHD('CNTGET ') – 処理スレッド数の取得
- CBLEETHD('GETTIME ') – 現在時刻の取得
- CBLEETHD('PRTEND ') – 保護区の終了
- CBLEETHD('PRTSTART') – 保護区の開始
- CBLEETHD('UIFAGET ') – ユーザ IFA 情報格納領域アドレスの取得

CBLEETHD('ABDCTL ')

名称

サービス閉塞の制御

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'ABDCTL '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(1).  
  02 データ名E PIC X(32).
```

機能

サービスを閉塞、および閉塞解除します。

UAP で値を設定するデータ領域

●データ名 A

サービス閉塞の制御を示す要求コードを「VALUE 'ABDCTL△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

閉塞か閉塞解除かを設定します。

VALUE 'S'

サービスを閉塞します。

VALUE 'R'

サービスの閉塞を解除します。

●データ名 E

SPP のサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

全サービスを対象にする場合は、サービス名を設定しないで先頭を空白にしてください。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05301	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05302	関数の呼び出し条件が不正です。
05303	メモリが不足しました。
05305	データ名 E に設定したサービス名が間違っています。
05399	そのほかのエラーが発生しました。

注意事項

1. データ名 E に、TP1/FSP（全銀 RC）で自動生成するサービスは指定できません。
2. すべてのサービスを対象にしてこの API を発行する場合、TP1/FSP（全銀 RC）で自動生成するサービスは閉塞/閉塞解除の対象外となります。

CBLEETHD('CNTGET ')

名称

処理スレッド数の取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CNTGET ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC 9(9) COMP.  
  02 データ名E PIC 9(9) COMP.
```

機能

処理スレッド数を取得します。

UAP で値を設定するデータ領域

●データ名 A

処理スレッド数の取得を示す要求コードを「VALUE 'CNTGET△△」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

通常処理スレッド数が返されます。

●データ名 E

予備処理スレッド数が返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05301	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05399	そのほかのエラーが発生しました。

注意事項

ステータスコードが「00000」以外の場合、データ名 D およびデータ名 E に返された値を参照しても、内容は保証されません。

CBLEETHD('GETTIME ')

名称

現在時刻の取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GETTIME '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC S9(9) COMP.  
  02 データ名E PIC S9(9) COMP.  
  02 データ名F ADDRESS.
```

機能

現在時刻を取得します。

CBLEETHD('GETTIME ')で取得する時刻は、TP1/EE が管理している現在時刻、またはマシンの現在時刻になります。

TP1/EE が管理している現在時刻は、プロセス関連定義の clock_time_interval オペランドの指定値の間隔で更新し、60 秒ごとにマシン時刻と同期を取ります。マシン時刻との同期の際には、時刻の進みが発生することがあります。また、時刻の同期に失敗した場合、および同期時点でのマシン時刻との誤差が大きい場合はメッセージが出力されます。

取得時刻のタイムゾーンは、TP1/Server Base のユーザサービス定義に指定した TZ 環境変数に依存します。

UAP で値を設定するデータ領域

●データ名 A

現在時刻取得要求を示す要求コードを「VALUE 'GETTIME△'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

時刻の取得形式を設定します。

- 1
BINARY 形式で取得します。
- 2
DATE 形式で取得します。
- 16
BINARY_TEXT 形式で取得します。
- 32
DATE_TEXT 形式で取得します。

●データ名 E

取得時刻の精度を設定します。

- 1
秒精度で取得します。この値を設定した場合は、TP1/EE が管理している現在時刻を取得します。
- 2
マイクロ秒精度で取得します。
この値を設定した場合は、システムコールを使用してマシンの現在時刻を取得します。

●データ名 F

取得した時刻を格納する領域のアドレスを設定します。

格納する領域に必要な PICTURE 句は、データ名 D、およびデータ名 E に設定した値によって異なります。次の表に示すサイズの領域を設定してください。

表 5-11 格納領域の PICTURE 句

データ名 D 指定値	データ名 E 指定値	
	1 (秒精度)	2 (マイクロ秒精度)
1 (BINARY 形式)	PIC 9(9) SYNC COMP	PIC 9(18) SYNC COMP
2 (DATE 形式)	PIC X(6)	PIC X(9)
16 (BINARY_TEXT 形式)	PIC X(7)	PIC X(11)
32 (DATE_TEXT 形式)	PIC X(12)	PIC X(18)

TP1/EE から値が返されるデータ領域

●データ名B

ステータスコードが，5 けたの数字で返されます。

●データ名F

取得した時刻が設定されます。

取得形式のイメージを，次に示します。

- BINARY 形式・秒精度で取得する場合（データ名 D に 1，データ名 E に 1 を設定）

図 5-1 BINARY 形式・秒精度の現在時刻表示

0	1	2	3	バイト
ss	ss	ss	ss	

SSSSSSSS

当日の 00:00:00 からの通算時刻です。0～8640000 がホストバイトオーダの符号なし整数で設定されます。

単位は 1/100 秒です。

(例) 通算時刻が 01:23:45 の場合，「0007AAE4」が設定されます。

- BINARY 形式・マイクロ秒精度で取得する場合（データ名 D に 1，データ名 E に 2 を設定）

図 5-2 BINARY 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	バイト
ss	ss	ss	ss	ss	ss	ss	ss	

SSSSSSSSSSSSSSSS

当日の 00:00:00 からの通算時刻です。0～86400999999 がホストバイトオーダの符号なし整数で設定されます。

単位はマイクロ秒です。

(例) 通算時刻が 01:23:45.567890 の場合，「0000000012B8C1492」が設定されます。

- DATE 形式・秒精度で取得する場合（データ名 D に 2，データ名 E に 1 を設定）

図 5-3 DATE 形式・秒精度の現在時刻表示

0	1	2	3	4	5	バイト
YY	MM	DD	hh	mm	ss	

次の各値が 10 進数パック形式で設定されます。

- YY：当日の西暦年下 2 けた（00～99）
- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）

- mm：分（00～59）
- ss：秒（00～60）

（例）時刻が 2010/12/31 23:59:59 の場合、「101231235959」が設定されます。

- DATE 形式・マイクロ秒精度で取得する場合（データ名 D に 2，データ名 E に 2 を設定）

図 5-4 DATE 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	バイト
YY	MM	DD	hh	mm	ss	ss	ss	ss	

次の各値が 10 進数パック形式で設定されます。

- YY：当日の西暦年下 2 けた（00～99）
- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）
- mm：分（00～59）
- sssssss：マイクロ秒（00000000～60999999）

（例）時刻が 2010/12/31 23:59:59.999999 の場合、「101231235959999999」が設定されます。

- BINARY_TEXT 形式・秒精度で取得する場合（データ名 D に 16，データ名 E に 1 を設定）

図 5-5 BINARY_TEXT 形式・秒精度の現在時刻表示

0	1	2	3	4	5	6	バイト
s	s	s	s	s	s	s	

SSSSSSS

当日の 00:00:00 からの通算時刻です。00000000～86400000 が 10 進数の文字列で設定されます。
下 2 けたは常に「00」となります。

単位は 1/100 秒です。

（例）通算時刻が 01:23:45 の場合、「0502500」が設定されます。

- BINARY_TEXT 形式・マイクロ秒精度で取得する場合（データ名 D に 16，データ名 E に 2 を設定）

図 5-6 BINARY_TEXT 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	バイト
s	s	s	s	s	s	s	s	s	s	s	

SSSSSSSSSS

当日の 00:00:00 からの通算時刻です。000000000000～864009999999 が 10 進数の文字列で設定されます。

単位はマイクロ秒です。

（例）通算時刻が 01:23:45.567890 の場合、「05025567890」が設定されます。

- DATE_TEXT 形式・秒精度で取得する場合（データ名 D に 32，データ名 E に 1 を設定）

図 5-7 DATE_TEXT 形式・秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	11	バイト
Y	Y	M	M	D	D	h	h	m	m	s	s	

次の各値が 10 進数の文字列で設定されます。

- YY：当日の西暦年下 2 けた（00～99）
- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）
- mm：分（00～59）
- ss：秒（00～60）

（例）時刻が 2010/12/31 23:59:59 の場合、「101231235959」が設定されます。

- DATE_TEXT 形式・マイクロ秒精度で取得する場合（データ名 D に 32，データ名 E に 2 を設定）

図 5-8 DATE_TEXT 形式・マイクロ秒精度の現在時刻表示

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	バイト
Y	Y	M	M	D	D	h	h	m	m	s	s	s	s	s	s	s	s	

次の各値が 10 進数の文字列で設定されます。

- YY：当日の西暦年下 2 けた（00～99）
- MM：当日の月（01～12）
- DD：当日の日（01～31）
- hh：当日 0 時からの通算時（00～23）
- mm：分（00～59）
- ssssssss：マイクロ秒（00000000～60999999）

（例）時刻が 2010/12/31 23:59:59.999999 の場合、「101231235959999999」が設定されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
05301	データ名に設定した値が間違っています。 要求コード（データ名 A）が間違っている場合も含まれます。
05302	関数の呼び出し条件が不正です。
05304	取得した時刻が、TP1/EE 内で制御可能な上限値を超過しました。
05319	時刻取得に失敗しました。

ステータスコード	意味
05399	その他のエラーが発生しました。

注意事項

ステータスコードが 00000 以外の場合、データ名 F は参照できません。

データ名 E に 2 を指定した場合は、システムコールを使用してマシン時刻を取得します。そのため、データ名 E に 2 を指定した CBLEETHD('GETTIME ')を多量に呼び出すと、性能が低下するおそれがあります。

KFSB45310-E または KFSB45311-W メッセージが出力された前後の 60 秒間では、データ名 E に 1 を指定した場合の取得時刻とマシン時刻がずれていることがあります。

CBLEETHD('PRTEND ')

名称

保護区の終了

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'PRTEND ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

保護区を終了します。保護区の外で異常が発生した場合は、スレッドダウンします。

UAP で値を設定するデータ領域

●データ名 A

保護区の終了を示す要求コードを「VALUE 'PRTEND△△'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。

ステータスコード	意味
05301	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05302	関数の呼び出し条件が不正です。
05399	そのほかのエラーが発生しました。

CBLEETHD('PRTSTART')

名称

保護区の開始

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'PRTSTART'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

保護区を開始します。保護区の中で異常が発生した場合は、プロセスダウンします。

UAP で値を設定するデータ領域

●データ名 A

保護区の開始を示す要求コードを「VALUE 'PRTSTART'」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。

ステータスコード	意味
05301	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05302	関数の呼び出し条件が不正です。
05306	すでに保護区になっています。
05399	そのほかのエラーが発生しました。

CBLEETHD('UIFAGET ')

名称

ユーザ IFA 情報格納領域アドレスの取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETHD' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'UIFAGET'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D ADDRESS.
```

機能

異常終了したスレッドのユーザ IFA 情報が格納された領域のアドレスを取得します。

UAP で値を設定するデータ領域

●データ名 A

ユーザ IFA 情報格納領域アドレスの取得を示す要求コードを「VALUE 'UIFAGET△」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 D

ユーザ IFA 情報格納領域アドレスが返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05301	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
05302	関数の呼び出し条件が不正です。
05399	そのほかのエラーが発生しました。

注意事項

1. CBLEETHD('UIFAGET ')はエラートランザクション（ERRTRN3）のトランザクションでだけ呼び出せます。エラートランザクション（ERRTRN3）のトランザクション内で、CBLEETRNL('C-COMMIT')またはCBLEETRNL('C-ROLL ')を呼び出したあとは、CBLEETHD('UIFAGET ')を呼び出せません。
2. 処理スレッドの再生成が成功したあとで、CBLEETHD('UIFAGET ')を呼び出した場合、異常終了スレッドのユーザ IFA 情報格納領域アドレスと、自スレッドのユーザ IFA 情報格納領域アドレスとは、同じになります。自スレッドのユーザ IFA 情報格納領域で引き継いだ情報が破壊されるおそれがあるので、領域のアクセスには注意してください。
この状態を確認するには、処理スレッド通番と異常発生元トランザクションの処理スレッド通番とを比較してください。これらが同じ場合は、同一スレッドで動作しています。
3. ステータスコードが「00000」以外の場合、データ名 D に返された値を参照しても、内容は保証されません。

タイマトランザクション (CBLEETIM)

時刻指定または経過時間指定のトランザクション（タイマトランザクション）を制御するプログラムについて説明します。タイマトランザクションの COBOL-UAP 作成用プログラムを次に示します。

- CBLEETIM('CANCEL ') – タイマトランザクションの起動取り消し
- CBLEETIM('CANCELAL') – 全タイマトランザクションの起動取り消し
- CBLEETIM('EXECAP ') – タイマトランザクションの起動

CBLEETIM('CANCEL ')

名称

タイマトランザクションの起動取り消し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETIM' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'CANCEL '  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(4).  
  02 データ名D PIC 9(9) COMP-X.
```

機能

タイマトランザクションを取り消します。CBLEETIM('EXECAP ')でタイマトランザクションを登録するときに設定した起動タイミングより前であれば、CBLEETIM('CANCEL ')でタイマトランザクションを取り消せます。

UAP で値を設定するデータ領域

●データ名 A

タイマトランザクションの起動取り消しを示す要求コードを「VALUE 'CANCEL△△」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

どの起動方式のタイマトランザクションを取り消すかを設定します。

VALUE 'INST'

次のタイマトランザクションを、コミットとは無関係に直ちに取り消します。

- CBLEETIM('EXECAP ')のデータ名 C に'INST' (即時登録方式) を設定したタイマトランザクション
- CBLEETIM('EXECAP ')のデータ名 C に'COMT' (トランザクション同期登録方式) を設定したタイマトランザクション

VALUE 'COMT'

CBLEETIM('EXECAP ')でデータ名 C に'COMT'（トランザクション同期登録方式）を設定したタイマトランザクションを取り消します。コミット前であれば、CBLEETIM('EXECAP ')を呼び出したトランザクションからタイマトランザクションを取り消せます。

VALUE 'CMTC'

次のタイマトランザクションを、コミット後に取り消します。

- CBLEETIM('EXECAP ')のデータ名 C に'INST'（即時登録方式）を設定したタイマトランザクション
- CBLEETIM('EXECAP ')のデータ名 C に'COMT'（トランザクション同期登録方式）を設定したタイマトランザクション

●データ名 D

タイマトランザクションのリクエスト識別子を設定します。CBLEETIM('EXECAP ')でタイマトランザクションを登録したときにデータ名 F で設定したリクエスト識別子と同じものを設定してください。

同一のリクエスト識別子のタイマトランザクションが複数登録されている場合、データ名 C で'INST'または'CMTC'を設定したときは、起動時刻が最も早いタイマトランザクションが取り消されます。データ名 C で'COMT'を設定したときは、最も早く登録要求されたタイマトランザクションが取り消されます。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00501	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00505	設定されたリクエスト識別子を持つタイマトランザクションは未登録、起動中、または起動済みです。
00509	関数の呼び出し条件が不正です。

注意事項

1. タイマトランザクションを登録するときに CBLEETIM('EXECAP ')でデータ名 C に'COMT'を設定した場合、CBLEETIM('EXECAP ')を呼び出したトランザクションがコミットしたあとは、

CBLEETIM('CANCEL ')でデータ名 C に'COMT'を設定してもタイマトランザクションを取り消せません。コミット後に取り消したい場合は、CBLEETIM('CANCEL ')でデータ名 C に'INST'または'CMTC'を設定してください。

2. CBLEETIM('EXECAP ')で設定した起動タイミングに達しているタイマトランザクションは取り消せません。
3. トランザクション起動タイミング (CBLEETIM('EXECAP ')のデータ名 C) で'CMTC'を設定した場合、対象のタイマトランザクションが同期点処理前に起動時刻に達しても、タイマトランザクションを起動しません。
4. トランザクション起動タイミング (CBLEETIM('EXECAP ')のデータ名 C) で'CMTC'を設定した場合、コミット前にトランザクションが異常終了したとき、またはロールバックしたときは、CBLEETIM('CANCEL ')の要求は無効になります。このとき、起動時刻に達している場合はタイマトランザクションを即時起動します。同一のトランザクション内で CBLEETR('ROLLMARK')がすでに呼び出されている場合は、ステータスコード「00509」でエラーリターンします。

CBLEETIM('CANCELAL')

名称

全タイマトランザクションの起動取り消し

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETIM' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A  PIC X(8) VALUE 'CANCELAL'.  
  02 データ名B  PIC X(5).  
  02 FILLER      PIC X(3).  
  02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

トランザクション同期登録方式で登録したすべてのタイマトランザクションを取り消します。
CBLEETIM('EXECAP ')を呼び出したトランザクションがコミット前であれば、そのトランザクションで CBLEETIM('EXECAP ')のデータ名 C に'COMT'を設定して登録したすべてのタイマトランザクションを、CBLEETIM('CANCELAL')で取り消せます。

UAP で値を設定するデータ領域

●データ名 A

全タイマトランザクションの起動取り消しを示す要求コードを「VALUE 'CANCELAL'」と設定します。

●データ名 Z

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。

ステータスコード	意味
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00501	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00505	このトランザクションでは、タイマトランザクションは、未登録です。

注意事項

1. CBLEETIM('EXECAP ')でデータ名 C に'INST'を設定して登録したタイマトランザクションは、CBLEETIM('CANCELAL')では取り消せません。
2. CBLEETIM('EXECAP ')でタイマトランザクションを登録したトランザクションから CBLEETIM('CANCELAL')を呼び出してください。別のトランザクションで CBLEETIM('CANCELAL')を呼び出しても、タイマトランザクションを取り消すことはできません。
3. CBLEETIM('EXECAP ')を呼び出したトランザクションのコミット後にタイマトランザクションを取り消したい場合は、CBLEETIM('CANCEL ')を使用してください。

CBLEETIM('EXECAP ')

名称

タイマトランザクションの起動

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETIM' USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'EXECAP '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名C PIC X(4).  
  02 データ名D PIC X(4).  
  02 データ名E PIC 9(9) COMP.  
  02 データ名F PIC 9(9) COMP-X.  
  02 データ名G PIC X(32).  
  02 データ名H PIC X(4).  
01 一意名3.  
  02 データ名I PIC 9(9) COMP.  
  02 データ名J PIC X(n).
```

機能

UAP からトランザクション（タイマトランザクション）を起動します。すぐに起動することも、経過時間または時刻を指定して起動することもできます。

UAP で値を設定するデータ領域

●データ名 A

タイマトランザクションの起動を示す要求コードを「VALUE 'EXECAP△△」と設定します。

●データ名 Z

オプションを数値で設定します。

0

タイマトランザクションを起動する場合に設定します。

1 【XTC 限定】

タイマトランザクションを永続属性とする場合に設定します。データ名 C に'COMT'を設定した場合だけ有効です。

2 【XTC 限定】

自トランザクションが終了するまで次のトランザクションのスケジュールを停止したい場合（UAP リターン後の次のトランザクションのスケジュールを停止したい場合）に設定します。このオプションを設定すると、待機系への転送処理終了後に、次のトランザクションのスケジュールが行われます。データ名 C に'COMT'を設定した場合だけ有効です。

●データ名 C

タイマトランザクションの起動方式を設定します。

VALUE 'INST'

即時登録方式です。コミットとは無関係に、データ名 D に設定した起動タイミングに従ってタイマトランザクションを起動します。

VALUE 'COMT'

トランザクション同期登録方式です。コミット後、データ名 D に設定した起動タイミングに従ってタイマトランザクションを起動します。コミット前には起動されません。

●データ名 D

タイマトランザクションの起動タイミングを設定します。

VALUE 'JUST'

タイマトランザクションを即時起動します。

VALUE 'INTV'

経過時間を指定してタイマトランザクションを起動します。データ名 E に設定した時間が経過したあとに起動します。

VALUE 'TIME'

時刻を指定してタイマトランザクションを起動します。データ名 E に設定した時刻に起動します。

●データ名 E

データ名 D に設定した起動タイミングに応じて次のように設定します。

- データ名 D に'JUST'を指定した場合（即時起動）
設定値は無視されます。
- データ名 D に'INTV'を指定した場合（経過時間指定の起動）
タイマトランザクションを起動するまでの経過時間を設定します（単位：100 ミリ秒）。1～3600000（100 ミリ秒から 100 時間）の範囲で指定してください。

データ名 C に 'INST' (即時登録方式) を設定した場合は、CBLEETIM('EXECAP ') を呼び出してからの経過時間です。データ名 C に 'COMT' (トランザクション同期登録方式) を設定した場合は、コミットからの経過時間です。

- データ名 D に 'TIME' を指定した場合 (時刻指定の起動)

タイマトランザクションを起動する時刻を設定します。0 時 0 分 0 秒を基準にして、起動時刻を秒数で算出します。0 (0 時 0 分 0 秒に開始) ~ 86399 (23 時 59 分 59 秒に開始) の範囲で指定してください。

(例)

午後 2 時 30 分 30 秒に起動させる場合、次のように 52230 を設定します。

$$14 \times 3600 + 30 \times 60 + 30 = 52230$$

設定した時刻が現在の時刻より前またはあとだった場合、タイマトランザクションの起動処理は、プロセス関連定義の `execap_permission_time` オペランドで指定した先行許容時間または後退許容時間に従います。

また、経過時間指定か時刻指定かに関係なく、タイマ起動時に誤差 (遅れ) が発生することがあります。

詳細は「[2.3.3 時刻または経過時間の指定方法](#)」および「[2.3.6 時刻または経過時間の指定と TP1/EE サービス定義との関係](#)」を参照してください。

●データ名 F

CBLEETIM('CANCEL ') でタイマトランザクションを取り消す場合のキーとなるリクエスト識別子を設定します。

タイマトランザクションを取り消す必要がない場合は 0 を設定します。0 を設定した場合でも、データ名 C に 'COMT' (即時登録方式) を設定して登録したトランザクションは、コミット前であれば CBLEETIM('CANCELAL') で取り消せます。

●データ名 G

タイマトランザクションとして起動するトランザクションのサービス名を 31 バイト以内のアスキー文字列で設定します。文字列の末尾には空白を付けてください。この空白は文字列の長さに数えません。

●データ名 H

タイマトランザクションが起動タイミングを迎えたとき、その時点で登録されている処理キューの中で優先的に起動するかどうかを設定します。

【TP1/FSP 限定】 トランザクションレベル方式の処理キュー制御機能を使用している場合、このデータの指定値は有効にならないで、タイマトランザクションは優先的に起動しません。優先的に起動したい場合は、トランザクションレベル名判定 UOC でプライオリティのいちばん高いトランザクションレベル名を指定してください。

VALUE 'HI△△'

優先的に起動します。

VALUE 'LOW△'

優先的に起動しないで、登録した順序で起動します。

●データ名 I

起動するトランザクションに渡すデータの長さを設定します（単位：バイト）。メモリ関連定義の time_message_size オペランドで指定した値以下の値を指定してください。渡すデータがない場合は 0 を設定します。

●データ名 J

起動するトランザクションに渡すデータを設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00501	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00502	データ名 G に設定したサービス名が間違っています。
00503	データ名 I に設定したデータ長が大き過ぎます。
00504	次に示す要因によって、資源不足で登録できません。 - メモリ関連定義の icb_no の指定値を超えています。 - メモリ関連定義の time_message_no の指定値を超えています。 - メモリ関連定義の pce_no の指定値を超えています。 - UDP 用送信バッファ（UOBF）面数が不足しています。
00506	TP1/EE システムが終了処理中のため、登録できません。または、CL サーバでの待機系のため登録できません。
00507	指定したサービス名に対するエントリポイントがありません。
00508	指定した時刻が先行および後退許容時間範囲内にありません。
00509	関数の呼び出し条件が不正です。
00520	指定したサービスのキューが満杯のため登録できません。

注意事項

1. データ名 C に'COMT'を指定した場合、コミット前にトランザクションが異常終了したとき、またはロールバックしたときは、CBLEETIM('EXECAP ')の要求は無効になります。同一のトランザクション内で CBLEETRN('ROLLMARK')がすでに呼び出されている場合は、ステータスコード「00509」でエラーリターンします。
2. この関数は、トランザクションのコミット時に実行されます。データ名 C に'COMT'を指定した場合、トランザクションの同期点を取得する処理でエラーが起こったときは、この関数の処理は無効となります。
3. データ名 C に'COMT'を設定して CBLEETIM('EXECAP ')を呼び出たトランザクションでは、トランザクションは最適化されません。
4. データ名 C に'COMT'、データ名 D に'TIME'を設定した場合、CBLEETIM('EXECAP ')呼び出し時に先行許容時間および後退許容時間がチェックされます。この時点で許容時間外の場合はエラーになり、コミット時には許容時間はチェックされません。
5. データ名 C に'COMT'、データ名 D に'TIME'を設定した場合、コミット時に後退許容時間外となっていたときは、タイマトランザクションが即時に起動します。
6. データ名 C に'INST'を設定した場合、トランザクションが異常終了したとき、またはロールバックしたときも、CBLEETIM('EXECAP ')の要求は有効になります。
7. システム終了（正常終了、計画停止 A）の場合、登録済みのタイマトランザクションがあるときは、タイマ破棄通知のエラートランザクション（ERRTRN4）が起動します。計画停止 B の場合は、登録済みのタイマトランザクションがあっても、ERRTRN4 は起動しません。
8. **【XTC 限定】** CL サーバがない場合、データ名 Z に 1 を設定しても永続属性となりません。
9. **【XTC 限定】** 永続属性の場合、UDP 用送信バッファ（UOBF）を使用します。
10. **【XTC 限定】** CL サーバでの待機系ではこの関数は使用しないでください。

トラブルシュート (CBLEETRB)

トラブルシュートに使用するプログラムについて説明します。トラブルシュートの COBOL-UAP 作成用プログラムを次に示します。

- CBLEETRB('UINFOTM ') – ユーザ情報の TASKTM 出力
- CBLEETRB('UTPUT ') – ユーザ固有のトレースの取得

CBLEETRB('UINFOTM ')

名称

ユーザ情報の TASKTM 出力

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRB' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'UINFOTM'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC X(63).  
  02 FILLER PIC X(1).  
  02 データ名E PIC S9(9) COMP.
```

機能

指定されたユーザ情報を TASKTM の「ユーザ情報」列に出力します。

UAP で値を設定するデータ領域

●データ名 A

ユーザ情報の TASKTM 出力を示す要求コードを「VALUE 'UINFOTM△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

出力するユーザ情報格納領域を設定します。63 バイトに満たない場合は後続の領域に、キャラクタ形式の場合は半角スペース、バイナリ形式の場合は 0x00 を設定してください。TASKTM には 63 バイト固定で出力します。

●データ名 E

指定したユーザ情報の属性を次の値で指定します。

1：バイナリ属性で出力

指定したユーザ情報を TASKTM 編集時にバイナリ形式で出力します。

2: キャラクタ属性で出力

指定したユーザ情報を TASKTM 編集時にキャラクタ形式で出力します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

リターン値	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
05401	引数に設定した値が間違っています。
05403	発行条件が不正です。
05407	トラブルシュート機能停止中です。

注意事項

1. トラブルシュート関連定義の `trb_tasktm_type` に 1, 2, 3 を指定した場合に有効です。
1, 2, 3 以外を指定した場合は出力されません。また、1, 2, 3 以外を指定した場合でも、この API はエラーにはなりません。
2. 同一トランザクション内で複数発行した場合は、最後に要求したユーザ情報が有効となります。
3. 文字コードのチェックは行いません。

CBLEETRB('UTPUT ')

名称

ユーザ固有のトレースの取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRB' USING 一意名1 一意名2
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'UTPUT ' .  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.  
01 一意名2.  
  02 データ名D PIC 9(9) COMP.  
  02 データ名E PIC 9(9) COMP.  
  02 データ名F PIC X(n).
```

機能

ユーザ固有のトレースデータを回線トレースに出力します。出力したトレースデータは、eetrblined コマンドまたは eetrblinedump コマンドで編集できます。

UAP で値を設定するデータ領域

●データ名 A

ユーザ固有のトレースの取得を示す要求コードを「VALUE 'UTPUT△△△」と設定します。

●データ名 C

0 を設定します。

●データ名 D

トレースを取得するイベントのイベント ID を設定します。0x0700~0x0740 の範囲のイベント ID を設定してください。

●データ名 E

トレースのデータ長を設定します（単位：バイト）。4~180 の範囲のデータ長を設定してください。

●データ名 F

トレースデータを取得する領域を設定します。トレースデータとして有効なデータは、データ名 E で設定した長さまでです。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
05401	データ名に設定した値が間違っています。
05406	データ名 E に設定したデータ長が設定できる範囲外です。
05407	トラブルシュート機能が縮退中です。

トランザクション制御 (CBLEETRN)

TP1/EE 独自のトランザクション制御をするプログラムについて説明します。トランザクション制御の COBOL-UAP 作成用プログラムを次に示します。

- CBLEETRN('C-COMMIT') – 連鎖モードのコミット
- CBLEETRN('C-ROLL ') – 連鎖モードのロールバック
- CBLEETRN('EXPTIME ') – トランザクション処理の監視時間の変更
- CBLEETRN('GETINFO ') – 現在のトランザクション識別情報の取得
- CBLEETRN('RMCLSMK ') – リソースマネージャとの接続の解放指示
- CBLEETRN('ROLLMARK') – トランザクションのロールバック指示

CBLEETRN('C-COMMIT')

名称

連鎖モードのコミット

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRN' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'C-COMMIT'.  
  02 データ名B PIC X(5).
```

機能

トランザクションの同期点を取得します。ルートトランザクションブランチから、処理が正常に終了したこと（コミット）を、トランザクションを構成するトランザクションブランチ、トランザクションサービス、リソースマネージャに知らせます。

CBLEETRN('C-COMMIT')が正常に終了すると、新しいグローバルトランザクションが開始します。CBLEETRN('C-COMMIT')を呼び出したスレッドは、このトランザクションの範囲内です。

グローバルトランザクションが複数のトランザクションブランチから構成される（CBLEETRN('C-COMMIT')を呼び出した UAP だけではない）場合は、それぞれのトランザクションブランチの処理結果がコミットにならないかぎり、コミットされません。

CBLEETRN('C-COMMIT')を呼び出せるのは、ルートトランザクションブランチだけです。それ以外の UAP から呼び出した場合は、ステータスコード「00941」でエラーリターンします。

CBLEETRN('C-COMMIT')を呼び出すスレッドは、このマニュアルの記述に従って正しく作成された UAP を稼働させたものでなければなりません。

CBLEETRN('C-COMMIT')は、同期点処理が完了したときに正常に終了、またはエラーリターンのどちらかで返ります。

サービスプログラムリターン時に明示的に CBLEETRN('C-COMMIT')、または CBLEETRN('C-ROLL ') を呼び出さない場合は、トランザクションの同期点を取得します。

トランザクション起動後に UAP をリターンするまでの、CBLEETRN('C-COMMIT')の呼び出し回数の上限値を、トランザクション関連定義の trn_max_commit_count オペランドで指定できます。

CBLEETRN('C-COMMIT')の呼び出し回数が、trn_max_commit_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_commit_downmode オペランドの指定に従って、異

常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

UAP で値を設定するデータ領域

●データ名 A

連鎖モードのコミットを示す要求コードを「VALUE 'C-COMMIT'」と設定します。連鎖モードのコミットをした処理以降も、この内容は変わりません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000※	正常に終了しました。CBLEETRN('C-COMMIT')が終了したあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00001	TP1/EE の環境下にありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00901	要求コード（データ名 A）が間違っています。
00903	CBLEETRN('C-COMMIT')を呼び出したグローバルトランザクションは、ヒューリスティック決着のため、あるトランザクションブランチはコミットとなり、あるトランザクションブランチはロールバックとなりました。このステータスコードは、ヒューリスティック決着の結果が、グローバルトランザクションの同期点の結果と一致しなかった場合にリターンします。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00904	グローバルトランザクションのトランザクションブランチがヒューリスティックに決着しました。しかし、障害のため、ヒューリスティックに決着したトランザクションブランチの同期点の結果がわかりません。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00932	コミット処理に失敗した、または CBLEETRN('ROLLMARK')を呼び出し済みのため、ロールバックを行いました。
00941	CBLEETRN('C-COMMIT')を不正なコンテキスト（例えば、すでにトランザクション中にいない）で呼び出しています。トランザクションモードに対する影響はありません。

ステータスコード	意味
00942	通信障害やリソースマネージャ障害などによってトランザクション決着に失敗したため、トランザクション回復処理を行います。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。

注※

次の要求がある場合は、要求を実行します。

- CBLEEMCH('CMTSND ')によるトランザクション非同期の一方送信メッセージの送信要求（TP1 キャッシュ機能を使用する場合）
- CBLEEMCH('CMTSND SY')によるトランザクション同期の一方送信メッセージの送信要求（TP1 キャッシュ機能を使用する場合）
- CBLEEMCP('SEND ')による非同期一方送信メッセージの送信要求（MCP を使用する場合）
- CBLEERPC('CMTSND ')によるトランザクションと同期して送信する非応答型 RPC 要求
- CBLEERPC('CMTSND TO')による通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求
- CBLEETIM('EXECAP ')（データ名 C に'COMT'を設定）によるタイマトランザクション起動要求

指定例

```
01 MISC.
  02 CMD-CODE      PIC X(8).
  02 STATUS-CODE   PIC X(5).
  :
  :
MOVE 'C-COMMIT' TO CMD-CODE OF MISC.
CALL 'CBLEETRN' USING MISC.
IF STATUS-CODE OF MISC NOT EQUAL TO '00000' THEN
  MOVE 'CANNOT COMMIT TRANSACTION' TO ERRMSG OF OUT-ERROR-REC
  WRITE OUT-ERROR-REC
END IF.
```

注意事項

1. 次の状態のとき、CBLEETRN('C-COMMIT')は正常終了します。

- リソースマネージャ接続失敗によるトランザクション縮退起動
- 新しいグローバルトランザクション生成時にリソースマネージャ接続失敗

必要に応じて CBLEETRN('GETINFO ')を呼び出してリソースマネージャの状態を確認してください。

2. **【TP1/FSP 限定】** DB キューの交替用メッセージ表書き込み機能使用時に、この関数内で行う DB キューのメッセージ表へのアクセスで障害が発生すると、この関数はリターンしないでトランザクションがリトライロールバックする場合があります。

CBLEETR('C-ROLL ')

名称

連鎖モードのロールバック

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETR' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'C-ROLL '.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC X(4).  
  02 データ名D PIC 9(9) COMP-X.
```

機能

トランザクションをロールバックします。データ名 C に'KEEP'を設定した場合、連鎖モードでロールバックしたあとは続けてトランザクションが開始します。データ名 C に'KEEP'以外を設定した場合、新たなトランザクションを開始しないで強制終了します。

CBLEETR('C-ROLL ')を呼び出すことで、ルートトランザクションブランチから、トランザクションブランチ、トランザクションサービス、およびリソースマネージャにロールバックを知らせます。

データ名 C に'KEEP'を設定した CBLEETR('C-ROLL ')が正常に終了すると、CBLEETR('C-ROLL ')を呼び出したスレッドはロールバックしてリターンします。そのあとで新しいグローバルトランザクションが開始します。CBLEETR('C-ROLL ')を呼び出したスレッドは、トランザクションの範囲内です。

CBLEETR('C-ROLL ')を呼び出せるのは、ルートトランザクションブランチだけです。それ以外の UAP から呼び出した場合は、ステータスコード「00941」でエラーリターンします。ただし、データ名 C に'PRCS'、'KILL'、または'ABRT'を指定した場合は、トランザクションブランチから呼び出せます。

CBLEETR('C-ROLL ')を呼び出すスレッドは、このマニュアルの記述に従って正しく作成された UAP を稼働させたものでなければなりません。

CBLEETR('C-ROLL ')は、同期点処理が完了したときに正常に終了、エラーリターン、または強制終了します。

サービスプログラムリターン時に明示的に CBLEETR('C-COMMIT')または CBLEETR('C-ROLL ')を呼び出さない場合は、トランザクションの同期点を取得します。

トランザクション起動後に UAP をリターンするまでの、CBLEETRN('C-ROLL ')または CBLEETRN('ROLLMARK')の呼び出し回数の上限値を、トランザクション関連定義の trn_max_rollback_count オペランドで指定できます。CBLEETRN('C-ROLL ')または CBLEETRN('ROLLMARK')の呼び出し回数が、trn_max_rollback_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_rollback_downmode オペランドの指定に従って、異常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

UAP で値を設定するデータ領域

●データ名 A

連鎖モードのロールバックを示す要求コードを「VALUE 'C-ROLL△△」と設定します。連鎖モードのロールバックをした処理以降も、この内容は変わりません。

●データ名 C

ロールバック時、プロセスまたはスレッドを強制終了するかどうかを設定します。

VALUE 'PRCS'

コアファイルと COBOL 異常終了時要約リストを出力してプロセスを強制終了させます。

VALUE 'KILL'

スレッドを強制終了させます。

VALUE 'ABRT'

メモリダンプファイルと COBOL 異常終了時要約リストを出力してスレッドを強制終了させます。

VALUE 'KEEP'

プロセスおよびスレッドを強制終了させません。

●データ名 D

トランザクション情報としてエラートランザクション (ERRTRN3) へ引き渡すエラー要因コードを設定します。

データ名 C に 'KILL' または 'ABRT' を設定したときだけ有効です。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。CBLEETRN('C-ROLL ')が終了したあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00901	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00903	CBLEETRN('C-ROLL ')を呼び出したグローバルトランザクションは、ヒューリスティック決着のため、あるトランザクションブランチはコミットとなり、あるトランザクションブランチはロールバックとなりました。このステータスコードは、ヒューリスティック決着の結果が、グローバルトランザクションの同期点の結果と一致しなかった場合にリターンします。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00904	グローバルトランザクションのトランザクションブランチがヒューリスティックに決着しました。しかし、障害のため、ヒューリスティックに決着したトランザクションブランチの同期点の結果がわかりません。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00941	CBLEETRN('C-ROLL ')を不正なコンテキスト（例えば、すでにトランザクション中にいない）で呼び出しています。トランザクションモードに対する影響はありません。
00942	通信障害やリソースマネージャ障害などによってトランザクション決着に失敗したため、トランザクション回復処理を行います。このステータスコードが返る原因になった UAP、リソースマネージャ、およびグローバルトランザクションの同期点の結果は、メッセージログファイルを参照してください。このステータスコードが返ったあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。

指定例

```

01 MISC.
  02 CMD-CODE      PIC X(8).
  02 STATUS-CODE   PIC X(5).
  :
  :
MOVE  'C-ROLL ' TO CMD-CODE OF MISC.
CALL  'CBLEETRN' USING MISC.
IF STATUS-CODE OF MISC NOT EQUAL TO '00000' THEN
  MOVE 'CANNOT ROLLBACK TRANSACTION' TO ERRMSG OF OUT-ERROR-REC
  WRITE OUT-ERROR-REC
END IF.

```

注意事項

1. トランザクション種別が MI（初期化トランザクション）、ME（終了トランザクション）、JI（共有リソース初期化トランザクション）、および UT（ユーザ初期化トランザクション）の場合、データ名 C に 'ABRT' を設定してもメモリダンプファイルは出力されません。また、データ名 C に 'KILL' または 'ABRT' を設定してもスレッドは強制終了しないでプロセスダウンになり、コアファイルが出力されます。
2. 次の状態のとき、CBLEETRN('C-ROLL ') は正常終了します。
 - リソースマネージャ接続失敗によるトランザクション縮退起動
 - 新しいグローバルトランザクション生成時にリソースマネージャ接続失敗必要に応じて CBLEETRN('GETINFO ') を呼び出してリソースマネージャの状態を確認してください。

CBLEETRN('EXPTIME ')

名称

トランザクション処理の監視時間の変更

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRN' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'EXPTIME'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC 9(9) COMP.  
  02 データ名D PIC S9(9) COMP.
```

機能

トランザクション処理の監視時間を変更します。

CBLEETRN('EXPTIME ')が正常に終了すると、CBLEETRN('EXPTIME ')を呼び出したトランザクション処理の監視が、データ名 C に設定した秒数で再開始されます。このとき、CBLEETRN('EXPTIME ')呼び出し前の処理時間監視は無効となります。

CBLEETRN('EXPTIME ')によるトランザクション処理の監視時間の変更は、呼び出し元トランザクションにだけ反映されます。

CBLEETRN('EXPTIME ')は、同一トランザクション内で 2 回以上呼び出すことはできません。

同一トランザクション内で CBLEETRN('WATCH ')を発行済みの場合、この関数は使用できません。

UAP で値を設定するデータ領域

●データ名 A

トランザクション処理の監視時間の変更を示す要求コードを「VALUE 'EXPTIME△」と設定します。

●データ名 C

変更後のトランザクション処理の監視時間を設定します（単位：秒）。指定できる値は 1～65535 です。0 を設定した場合は、時間監視をしません。

●データ名 D

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00901	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。
00943	CBLEETR('EXPTIME ')を同一トランザクション内で 2 回以上呼び出しました。
00944	その他のエラーが発生しました。

指定例

```
01 MISC.
  02 CMD-CODE      PIC X(8).
  02 STATUS-CODE   PIC X(5).
  02 FILLER        PIC  X(3).
  02 TIME          PIC  9(9) COMP.
  02 FLAGS         PIC  S9(9) COMP.
  :
  :
MOVE  'EXPTIME ' TO CMD-CODE OF MISC.
MOVE  100 TO TIME OF MISC.
CALL  'CBLEETR' USING MISC.
IF STATUS-CODE OF MISC NOT EQUAL TO  '00000' THEN
  MOVE  'CANNOT CHANGE TRN_EXPIRATION_TIME' TO ERRMSG OF OUT-ERROR-REC
  WRITE OUT-ERROR-REC
END IF.
```

注意事項

CBLEETR('EXPTIME ')によるトランザクション処理の監視時間の変更は、呼び出し元のトランザクションにだけ反映されるため、eetrntim コマンドの-I オプションで表示するトランザクション処理の監視時間には反映されません。

CBLEETRN('GETINFO ')

名称

現在のトランザクション識別情報の取得

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRN' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'GETINFO'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名Z PIC S9(9) COMP VALUE ZERO.  
  02 データ名C PIC S9(9) COMP.  
  02 データ名D PIC 9(9) COMP-X.  
  02 データ名E PIC S9(9) COMP.  
  02 データ名F PIC S9(9) COMP.  
  02 データ名G PIC S9(9) COMP.  
  02 データ名H PIC X(36).  
  02 データ名I PIC S9(9) COMP.  
  02 データ名J PIC X(256).  
  02 データ名K PIC S9(9) COMP.  
  02 データ名L PIC X(256).  
  02 FILLER PIC X(1024).
```

機能

実行中のトランザクションのトランザクション識別情報をリソースマネージャごとを取得します。ステータスコードが「00000」（次のリソースマネージャ情報なし）になるまで、CBLEETRN('GETINFO ')を連続して呼び出すことによって、接続中のすべてのリソースマネージャの情報を取得できます。ステータスコードが「00000」の状態で CBLEETRN('GETINFO ')を呼び出した場合は、最初からトランザクション識別情報を取得します。

UAP で値を設定するデータ領域

●データ名 A

現在のトランザクション識別情報の取得を示す要求コードを「VALUE 'GETINFO△'」と設定します。

●データ名 Z

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

●データ名 C

プロセス ID が返されます。

●データ名 D

該当するトランザクションの中央処理通番が返されます。

●データ名 E

リソースマネージャとの接続状態が返されます。

0

リソースマネージャ接続正常

1

リソースマネージャ接続失敗によるトランザクション縮退起動

●データ名 F

リソースマネージャ ID が返されます。

●データ名 G

リソースマネージャ名（リソースマネージャ拡張子を含む）の長さが返されます（単位：バイト）。

●データ名 H

リソースマネージャ名（リソースマネージャ拡張子を含む）が返されます。

●データ名 I

xa_open 関数用文字列の長さが返されます（単位：バイト）。

●データ名 J

xa_open 関数用文字列が返されます。

●データ名 K

XID の長さが返されます（単位：バイト）。

●データ名 L

XID が返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。(次のリソースマネージャ情報なし)
00001	TP1/EE の環境下ではありません。
00003	正常に終了しました。(次のリソースマネージャ情報あり)
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00901	データ名に設定した値が間違っています。要求コード (データ名 A) が間違っている場合も含まれます。

指定例

```

01 MISC.
  02 CMD-CODE      PIC X(8).
  02 STATUS-CODE   PIC X(5).
  02 FILLER        PIC X(3).
  02 FLAGS         PIC S9(9) COMP VALUE ZERO.
  02 PROCESSID     PIC S9(9) COMP.
  02 CTRANID       PIC 9(9) COMP-X.
  02 RESUME        PIC S9(9) COMP.
  02 RMID          PIC S9(9) COMP.
  02 SIZE          PIC S9(9) COMP.
  02 NAME          PIC X(36) COMP.
  02 OPENSIZ       PIC S9(9) COMP.
  02 OPENNAME      PIC X(256).
  02 XIDSIZ        PIC S9(9) COMP.
  02 XID           PIC X(256).

MOVE 'GETINFO' TO CMD-CODE OF MISC.

MOVE '00003' TO STATUS-CODE OF MISC.
PERFORM UNTIL STATUS-CODE OF MISC NOT = '00003'
  CALL 'CBLEETR' USING MISC
  IF STATUS-CODE OF MISC EQUAL TO '00000' THEN
    DISPLAY PROCESSID OF MISC
    DISPLAY CTRANID OF MISC
  ELSE
    IF STATUS-CODE OF MISC EQUAL TO '00003' THEN
      DISPLAY PROCESSID OF MISC
      DISPLAY CTRANID OF MISC
    ELSE
      DISPLAY 'CANNOT GET TRANSACTION INFORMATION'
    END-IF
  END-IF
END-UNTIL
END-PERFORM.

```

注意事項

リソースマネージャがない場合はリソースマネージャ情報以外の値が設定され、ステータスコード「00000」(次のリソースマネージャ情報なし) でリターンします。

CBLEETR('RMCLSMK ')

名称

リソースマネージャとの接続の解放指示

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETR' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'RMCLSMK'.  
  02 データ名B PIC X(5).  
  02 FILLER PIC X(3).  
  02 データ名C PIC S9(9) COMP VALUE ZERO.
```

機能

CBLEETR('RMCLSMK ')が正常に終了すると、CBLEETR('RMCLSMK ')を呼び出したスレッドのトランザクションの同期点処理が完了した時点でスレッドにリンクされているすべてのリソースマネージャに対して接続の解放要求（xa_close 関数）が指示されます。同期点処理の完了時に接続確立済みのリソースマネージャが存在しない場合は、CBLEETR('RMCLSMK ')を呼び出しても接続の解放要求（xa_close 関数）を指示しません。

リソースマネージャとの接続の再確立（xa_open 関数）は、CBLEETR('RMCLSMK ')を呼び出したスレッドでの次のトランザクション開始時に TP1/EE が行います。

UAP で値を設定するデータ領域

●データ名 A

リソースマネージャとの接続の解放指示を示す要求コードを「VALUE 'RMCLSMK△」と設定します。

●データ名 C

0 を設定します。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。
00901	データ名に設定した値が間違っています。要求コード（データ名 A）が間違っている場合も含まれます。

指定例

```
01 MISC.  
  02 CMD-CODE      PIC X(8).  
  02 STATUS-CODE   PIC X(5).  
  :  
  :  
MOVE 'RMCLSMK ' TO CMD-CODE OF MISC.  
CALL 'CBLEETR' USING MISC.  
IF STATUS-CODE OF MISC NOT EQUAL TO '00000' THEN  
  MOVE 'CANNOT CLOSE CONNECTION' TO ERRMSG OF OUT-ERROR-REC  
  WRITE OUT-ERROR-REC  
END IF.
```

CBLEETRN('ROLLMARK')

名称

トランザクションのロールバック指示

形式

PROCEDURE DIVISION の指定

```
CALL 'CBLEETRN' USING 一意名1
```

DATA DIVISION の指定

```
01 一意名1.  
  02 データ名A PIC X(8) VALUE 'ROLLMARK'.  
  02 データ名B PIC X(5).
```

機能

トランザクションのロールバックを指示します。CBLEETRN('ROLLMARK')が正常に終了すると、関数を呼び出したスレッドのトランザクションはロールバックされます。

トランザクション起動後に UAP をリターンするまでの、CBLEETRN('C-ROLL ')または CBLEETRN('ROLLMARK')の呼び出し回数の上限値を、トランザクション関連定義の trn_max_rollback_count オペランドで指定できます。CBLEETRN('C-ROLL ')または CBLEETRN('ROLLMARK')の呼び出し回数が、trn_max_rollback_count オペランドで指定した上限値に達した場合、トランザクション関連定義の trn_max_rollback_downmode オペランドの指定に従って、異常終了するか、または KFSB50927-E メッセージを出力します。異常終了時にサービスを閉塞するかどうかは、プロセス関連定義の service_hold オペランドで指定します。

UAP で値を設定するデータ領域

●データ名 A

トランザクションのロールバック指示を示す要求コードを「VALUE 'ROLLMARK'」と設定します。トランザクションがロールバックした処理以降も、この内容は変わりません。

TP1/EE から値が返されるデータ領域

●データ名 B

ステータスコードが、5 けたの数字で返されます。

ステータスコード

ステータスコード	意味
00000	同期点処理をロールバックさせます。また、ロールバックが終了したあとも、このスレッドはトランザクション下であって、グローバルトランザクションの範囲内です。
00001	TP1/EE の環境下ではありません。
00004	UOC から発行しているため、この機能は使用できません。
00005	オフラインバッチプロセスから発行しているため、この機能は使用できません。

指定例

```
01 MISC.  
  02 CMD-CODE      PIC X(8).  
  02 STATUS-CODE   PIC X(5).  
    :  
    :  
MOVE 'ROLLMARK' TO CMD-CODE OF MISC.  
CALL 'CBLEETRN' USING MISC.  
IF STATUS-CODE OF MISC NOT EQUAL TO '00000' THEN  
  MOVE 'CANNOT ROLLBACK TRANSACTION' TO ERRMSG OF OUT-ERROR-REC  
  WRITE OUT-ERROR-REC  
END IF.
```

注意事項

CBLEETRN('ROLLMARK')が正常終了したあとは、CBLEERPC('CMTSND ')によるサービス要求、および CBLEETIM('EXECAP ')（データ名 C に'COMT'を設定）によるタイマトランザクション起動要求を実行してもエラーになります。

6

ユーザOWNコーディング (UOC)

この章では、XTC のユーザOWNコーディングの使用方法について説明します。

6.1 UOC インタフェース

メッセージの送受信をする UAP をより多様な業務に対応させるために補助するプログラムを、UOC といいます。UOC は保護区です。

XTC で使用できる UOC を次に示します。

- オンライン開始 UOC

UOC を使用する場合は、次のとおりです。

1. C 言語で UOC を作成する（COBOL 言語では作成できません）。
2. XTC のユーザサービス関連定義の `uoc_func` 定義コマンドに UOC 関数を登録する。
3. UOC 関数のオブジェクトファイル、およびその他の UAP オブジェクトファイルをリンケージして UAP 共用ライブラリを作成する。

6.2 UOC で使用できる関数

UOC では、次に示す関数だけを使用できます。次に示す以外の関数を使用した場合、正常に動作しないことがあるためご注意ください。

- メッセージログ出力関数 (ee_logprint)
- ワークセグメントの割り当て関数 (ee_mem_getwkseg)
- 使用済みワークセグメントの返還関数 (ee_mem_rlswkseg)
- 現在時刻取得関数 (ee_thd_gettime)

オンライン開始 UOC から TASK 属性のセグメント要求を行った場合、オンライン開始 UOC のリターン前に、使用済みワークセグメントの返還関数で必ずワークセグメントを解放してください。ワークセグメントの解放漏れがあった場合、オンライン開始 UOC のリターン後に自動解放されません。

6.3 オンライン開始 UOC インタフェース

オンライン開始 UOC は、XTC の開始時に前処理をします。

6.3.1 形式

ANSI C, C++の形式

```
#include <eemch_uoc.h>
int uoc_func(struct eemch_uoc_parm *parm)
```

6.3.2 説明

オンライン開始 UOC を呼び出すとき、次に示す所定のパラメタを parm に設定します。

6.3.3 パラメタの内容

(1) eemch_uoc_parm の内容

64 ビットアーキテクチャの場合

```
typedef struct eemch_uoc_parm {
    char    *usat;           ... ユーザSATエリアポインタ
    EEULONG usat_len;        ... ユーザSATエリアサイズ
    char    reserve1[4];     ... 予備
    EEULONG64 usat_large_len; ... 拡張ユーザSATエリアサイズ
    char    reserve[40];     ... 予備
} EEMCHUOCPARM;
```

32 ビットアーキテクチャの場合

```
typedef struct eemch_uoc_parm {
    char    *usat;           ... ユーザSATエリアポインタ
    EEULONG usat_len;        ... ユーザSATエリアサイズ
    char    reserve[56];     ... 予備
} EEMCHUOCPARM;
```

6.3.4 XTC が値を設定する項目

(1) eemch_uoc_parm の内容

●usat

ユーザ SAT エリアのアドレスが設定されます。ユーザ SAT エリアサイズが 0 の場合は、値を参照できません。

●usat_len

ユーザ SAT エリアサイズが設定されます（単位：バイト）。

64 ビットアーキテクチャの場合、ユーザ SAT エリアのサイズが 4 ギガバイト以上のときは 0 が設定されます。この場合、usat_large_len を参照してユーザ SAT エリアのサイズを取得してください。

ユーザ SAT エリアのサイズが 4 ギガバイト未満の場合でも、usat_large_len を参照してユーザ SAT エリアのサイズを取得できます。

●usat_large_len

拡張ユーザ SAT エリアサイズが設定されます（単位：バイト）。

6.3.5 リターン値

オンライン開始 UOC は、次のコードでリターンしてください。

リターン値	意味
EEMCH_UOC_OK	正常リターン
EEMCH_UOC_NG	エラーリターン

EEMCH_UOC_OK でリターンした場合

XTC の開始処理を続行します。

EEMCH_UOC_NG でリターンした場合

XTC の開始処理を中止します。

7

コーディング例

この章では、TP1/EE の UAP 共用ライブラリのコーディング例について説明します。

7.1 C 言語のコーディング例

C 言語の UAP 共用ライブラリのコーディング例を示します。

```
/* C標準ヘッダファイルのインクルード */
#include <stdio.h>
#include <string.h>
/* TP1/EE 提供ヘッダファイルのインクルード */
#include <eerpcsrv.h>
#include <eerpc.h>
#include <eelog.h>
#include <eetrn.h>
#include <eetim.h>
#include <eemem.h>
#include <eedbq.h>
/* 自作関数 ヘッダファイルのインクルード */
#include <sv_dba.h>

/*****
** 目的      : サービストランザクション（非応答型RPC用）
** 機能      : (1)受信データから取得する情報を格納するデータ領域を確保する
               (2)受信データから、必要な情報を取得する
               (3)データ領域を用いて、タイマトランザクションを発行
               (4)確保した領域を解放
               (5)タイマトランザクション失敗時には、ロールバックする
*****/
```

入力データ形式

データ名	データ型	意味
name_len	unsigned int	名前の文字列長
name	char(20)	名前
sex_len	unsigned int	性別の文字列長
sex	char(4)	性別
age_len	unsigned int	年齢の文字列長
age	char(4)	年齢
sale	int	売上

トランザクション処理に必要な情報

データ名	データ型	意味
name	char(20)	名前
sex	char(4)	性別
age	char(4)	年齢

sale	int	売上
srv	char(32)	サービス名

*****/

```

void SAMPLE_UAP(
    char *in_data,
    EEULONG *in_len,
    char *out_data,
    EEULONG *out_len,
    EERPC_INTERFACE_TBL *trninf
)
{
    /******
    /*          変数の宣言          */
    /******
    in_customer_t    in_custm;        /*クライアントから渡される顧客情報(44byte)*/
    customer_t       custm;           /*トランザクションに必要な顧客情報(64byte)*/
    EEULONG          send_len;        /*送信データ長*/
    int              wrc;              /*関数リターンコード*/
    void             *seg_pt;         /*確保領域の先頭アドレス*/

    /******
    /*          データ領域の確保          */
    /******
    send_len = sizeof(customer_t);
    wrc = ee_mem_getwkseg(EE_SEGTYPE_UAPWORK, send_len, &seg_pt, EENOFLLAGS);
    if(wrc != EE_OK){
        wrc = ee_logprint("KFSB05200-E", "SP", "SERVER:領域確保に失敗しました。¥n",
        NULL, EENOFLLAGS);
        return;
    }

    /******
    /*          必要なデータ取得          */
    /******
    memcpy(&in_custm, in_data, *in_len);
    memcpy(custm.name, in_custm.name, 20);
    memcpy(custm.sex, in_custm.sex, 4);
    memcpy(custm.age, in_custm.age, 4);
    custm.sale = in_custm.sale;
    memcpy(custm.srv, (trninf->service), 32);

    /******
    /*          TMトランザクション起動          */
    /******
    memcpy(seg_pt, &custm, send_len);
    wrc = ee_tim_execap(EETIM_INST, EETIM_JUST, 0, 0, "srv02", EETIM_HI, (char*)seg_pt, send_
n, EENOFLLAGS);

    /* TMトランザクション起動失敗の場合、ロールバック終了 */
    if(wrc != EE_OK){
        wrc = ee_logprint("KFSB05300-E", "SP", "SERVER:タイマトランザクション起動に失
敗しました。¥n", NULL, EENOFLLAGS);
        wrc = ee_mem_rlsbkseg(seg_pt, EENOFLLAGS);
        wrc = ee_trn_chained_rollback(EETRN_KILL, 0);
    }
}

```

```

        /** ロールバック失敗の場合、エラー表示して終了 **/
        if(wrc != EE_OK){
            wrc = ee_logprint("KFSB05400-E", "SP", "SERVER:ROLLBACKに失敗しまし
た。¥n", NULL, EENOFLLAGS);
            return;
        }

        /*          領域の解放          */
        wrc = ee_mem_rlswkseg(seg_pt, EENOFLLAGS);

        /*          DBキューに書込み          */
        wrc = ee_dbq_msgput("DBQ001", in_data, *in_leng, EENOFLLAGS);
        if(wrc != EE_OK){
            wrc = ee_logprint("KFSB06000-E", "SP", "SERVER:DBキューのメッセージ書込みに失
敗しました。¥n", NULL, EENOFLLAGS);
            wrc = ee_trn_chained_rollback(EETRN_KILL, 0);

            /** ロールバック失敗の場合、エラー表示して終了 **/
            if(wrc != EE_OK){
                wrc = ee_logprint("KFSB05400-E", "SP", "SERVER:ROLLBACKに失敗しまし
た。¥n", NULL, EENOFLLAGS);
                return;
            }
        }

        return;
    }

}

/*****
** 目的      : エラートランザクション 1
** 機能      : エラートランザクション 1 を実行する
*****/
void SAMPLE_E1(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /*******
    /*          変数の宣言          */
    /*******
    int      wrc;                                /*関数リターンコード*/

    /*******
    /*          表示          */
    /*******
    wrc = ee_logprint("KFSB05600-E", "SP", "SERVER:エラートランザクション1を起動します。¥n
", NULL, EENOFLLAGS);

    return;
}

/*****
** 目的      : エラートランザクション 2
** 機能      : エラートランザクション 2 を実行する
*****/

```

```

*****/
void SAMPLE_E2(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /******
    /*          変数の宣言          */
    /******
    int          wrc;                /*関数リターンコード*/

    /******
    /*          表示          */
    /******
    wrc = ee_logprint("KFSB05700-E", "SP", "SERVER:エラーランザクション2を起動します。¥n", NULL, EENOFLLAGS);

    return;
}

/*****
** 目的      : エラーランザクション3
** 機能      : エラーランザクション3を実行する
*****/
void SAMPLE_E3(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /******
    /*          変数の宣言          */
    /******
    int          wrc;                /*関数リターンコード*/
    char          rcv_data[8];        /*RPC応答格納領域*/
    EEULONG       rcv_len=8;          /*RPC応答データ長*/

    /******
    /*          表示          */
    /******
    wrc = ee_logprint("KFSB05800-E", "SP", "SERVER:エラーランザクション3を起動します。¥n", NULL, EENOFLLAGS);

    /******
    /*          RPC 送信          */
    /******
    memset(rcv_data, 0, 8);
    wrc = ee_rpc_call("outSPP", "outSRV", in_data, in_leng, rcv_data, &rcv_len, EENOFLLAGS);
    if(wrc != EE_OK){
        wrc = ee_logprint("KFSB05500-E", "SP", "SERVER:RPC処理に失敗しました。¥n", NULL, EENOFLLAGS);
    }
}

```



```

        return;
    }

/*****
** 目的      : エラートランザクション4
** 機能      : エラートランザクション4を実行する
*****/
void SAMPLE_E4(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /*****/
    /*          変数の宣言          */
    /*****/
    int          wrc;                /*関数リターンコード*/

    /*****/
    /*          表示          */
    /*****/
    wrc = ee_logprint("KFSB05900-E", "SP", "SERVER:エラートランザクション4を起動します。¥n", NULL, EENOF
LAGS);

    return;
}

/*****
** 目的      : MIトランザクション
** 機能      : MIトランザクションを実行する
*****/
void SAMPLE_MI(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /*****/
    /*          変数の宣言          */
    /*****/
    int          wrc;                /*関数リターンコード*/

    /*****/
    /*          表示          */
    /*****/
    wrc = ee_logprint("KFSB06200-I", "SP", "SERVER:サンプルUAPを開始します。¥n", NULL, EENOF
LAGS);

    return;
}

/*****
** 目的      : MEトランザクション
** 機能      : MEトランザクションを実行する
*****/

```

```

*****/
void SAMPLE_ME(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /***/
    /*          変数の宣言          */
    /***/
    int          wrc;                /*関数リターンコード*/

    /***/
    /*          表示          */
    /***/
    wrc = ee_logprint("KFSB06300-I", "SP", "SERVER:サンプルUAPを終了します。¥n", NULL, EENOF
LAGS);

    return;
}

*****/
** 目的      : タイマトランザクション
** 機能      : タイマトランザクションを実行する
*****/
void SAMPLE_TM(
    char *in_data,
    EEULONG *in_leng,
    char *out_data,
    EEULONG *out_leng,
    EERPC_INTERFACE_TBL *trninf
)
{
    /***/
    /*          変数の宣言          */
    /***/
    int          wrc;                /*関数リターンコード*/
    char          rcv_data[8];        /*RPC応答格納領域*/
    EEULONG       rcv_len=8;          /*RPC応答データ長*/

    /***/
    /*          表示          */
    /***/
    wrc = ee_logprint("KFSB06400-I", "SP", "SERVER:タイマトランザクションを起動します。¥n"
, NULL, EENOF LAGS);

    /***/
    /*          RPC 送信          */
    /***/
    memset(rcv_data, 0, 8);
    wrc = ee_rpc_call("outSPP", "outSRV", in_data, in_leng, rcv_data, &rcv_len, EENOF LAGS);
    if(wrc != EE_OK){
        wrc = ee_logprint("KFSB05500-E", "SP", "SERVER:RPC処理に失敗しました。¥n", NULL
, EENOF LAGS);
    }
}

```

```

        return;
    }

/*****
** 目的      : DBキューの読出しトランザクション
** 機能      : DBキューの読出しトランザクションを実行する
*****/
void SAMPLE_DBQ(
    char *in_data,
    EEULONG *in_len,
    char *out_data,
    EEULONG *out_len,
    EERPC_INTERFACE_TBL *trninf
)
{
    /*****/
    /*          変数の宣言          */
    /*****/
    int      wrc;                                /*関数リターンコード*/
    char      rcv_data[8];                       /*RPC応答格納領域*/
    EEULONG    rcv_len=8;                        /*RPC応答データ長*/

    /*****/
    /*          表示          */
    /*****/
    wrc = ee_logprint("KFSB06400-I", "SP", "SERVER:DBキューの読出しトランザクションを起動
します。¥n", NULL, EENOFLLAGS);

    /*****/
    /*          RPC 送信          */
    /*****/
    memset(rcv_data, 0, 8);
    wrc = ee_rpc_call("outSPP", "outSRV", in_data, in_len, rcv_data, &rcv_len, EENOFLLAGS);
    if(wrc != EE_OK){
        wrc = ee_logprint("KFSB05500-E", "SP", "SERVER:RPC処理に失敗しました。¥n", NULL, EENOFLLA
GS);
    }

    return;
}

```

7.2 COBOL 言語のコーディング例

COBOL 言語の UAP 共用ライブラリのコーディング例を示します。

```
*****
**      All Rights Reserved. Copyright (C) 2004, Hitachi, Ltd. P-1M64-2A1
**      目的      :   TP1/EE サンプルプログラム(COBOL言語版)
*****
* spp.cbl
*****
**      目的      :   サービストランザクション処理 (非応答型RPC用)
**      機能      :   サービストランザクション処理の実行
**                  (1)受信データから取得する情報を格納するデータ領域を確保する
**                  (2)受信データから、必要な情報を取得する
**                  (3)データ領域を用いて、タイマトランザクションを発行
**                  (4)確保した領域の解放
**                  (5)タイマトランザクション失敗時には、ロールバックする
**
**      入力データ形式
**
**      +-----+
**      | データ名 | データ型 | 意味 |
**      |-----|
**      | NAM-LEN  | PIC S9(9) | 名前の文字列長 |
**      |-----|
**      | NAM-ARG  | PIC X(20) | 名前 |
**      |-----|
**      | SEX-LEN  | PIC S9(9) | 性別の文字列長 |
**      |-----|
**      | SEX-ARG  | PIC X(4)  | 性別 |
**      |-----|
**      | AGE-LEN  | PIC S9(9) | 年齢の文字列長 |
**      |-----|
**      | AGE-ARG  | PIC X(4)  | 年齢 |
**      |-----|
**      | SALE-ARG | PIC S9(9) | 売上 |
**      |-----|
**
**      トランザクション処理に必要な情報
**
**      +-----+
**      | データ名 | データ型 | 意味 |
**      |-----|
**      | NAM-ARG  | PIC X(20) | 名前 |
**      |-----|
**      | SEX-ARG  | PIC X(4)  | 性別 |
**      |-----|
**      | AGE-ARG  | PIC X(4)  | 年齢 |
**      |-----|
**      | SALE-ARG | PIC S9(9) | 売上 |
**      |-----|
**      | SRV-ARG  | PIC X(32) | サービス名 |
**      |-----|
**
*****
*****
* 見出し
*****
IDENTIFICATION DIVISION.
```

```

*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_UAP.
*
*****
*   環境部
*****
ENVIRONMENT      DIVISION.
*
*****
*   変数宣言部
*****
DATA             DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** 送信データ使用変数 *****
*** 引数(IN-DATA)情報格納構造体
01 SQL-ARG1.
    02 NAM-LEN     PIC S9(9) COMP.
    02 NAM-ARG     PIC X(20).
    02 SEX-LEN     PIC S9(9) COMP.
    02 SEX-ARG     PIC X(4).
    02 AGE-LEN     PIC S9(9) COMP.
    02 AGE-ARG     PIC X(4).
    02 SALE-ARG    PIC S9(9) COMP.
*** 送信データ構造体
01 SQL-ARG2.
    02 NAM-ARG     PIC X(20).
    02 SEX-ARG     PIC X(4).
    02 AGE-ARG     PIC X(4).
    02 SALE-ARG    PIC S9(9) COMP.
    02 SRV-ARG     PIC X(32).
*
*** LOGPRINTメッセージ設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.
*** サンプルメッセージ(領域確保失敗)
01 MSGID52        PIC X(12) VALUE 'KFSB05200-E '.
01 MSG52          PIC X(64) VALUE
    'SERVER:領域確保に失敗しました.'.
01 MSGLEN52       PIC 9(9)  COMP VALUE 64.
*** サンプルメッセージ(タイマトラン起動失敗)
01 MSGID53        PIC X(12) VALUE 'KFSB05300-E '.
01 MSG53          PIC X(128) VALUE
    'SERVER:タイマトランザクション起動に失敗しました.'.
01 MSGLEN53       PIC 9(9)  COMP VALUE 128.
*** サンプルメッセージ(ロールバック失敗)
01 MSGID54        PIC X(12) VALUE 'KFSB05400-E '.
01 MSG54          PIC X(64) VALUE
    'SERVER:ROLLBACKに失敗しました.'.
01 MSGLEN54       PIC 9(9)  COMP VALUE 64.
*** サンプルメッセージ(DBキュー書込み失敗)

```

```

01 MSGID55      PIC X(12) VALUE 'KFSB06000-E '.
01 MSG55        PIC X(128) VALUE
' SERVER:DBキューのメッセージ書込みに失敗しました. '.
01 MSGLEN55     PIC 9(9)  COMP VALUE 128.
*
*** API各種引数設定 *****
*** CBLEELOG(' PRINT  ')
01 EELOG-PRINT-ARG1.
02 EELOG-PRINT-REQUEST      PIC X(8) VALUE ' PRINT  '.
02 EELOG-PRINT-STATUS-CODE  PIC X(5).
02 FILLER                   PIC X(3).
02 EELOG-PRINT-FLAGS        PIC S9(9) COMP VALUE ZERO.
02 EELOG-PRINT-MSG-ID       PIC X(12).
02 EELOG-PRINT-PGM-ID       PIC X(3).
01 EELOG-PRINT-ARG2.
02 EELOG-PRINT-MSG-LEN      PIC 9(9) COMP.
02 EELOG-PRINT-MSG         PIC X(128).
01 EELOG-PRINT-ARG3.
02 EELOG-PRINT-INFO        PIC S9(9) COMP VALUE ZERO.
*
*** CBLEETRN(' C-ROLL  ')
01 EETRN-C-ROLL-ARG.
02 EETRN-C-ROLL-REQUEST     PIC X(8) VALUE ' C-ROLL  '.
02 EETRN-C-ROLL-STATUS-CODE PIC X(5).
02 FILLER                   PIC X(3).
02 EETRN-C-ROLL-THKIND      PIC X(4) VALUE ' KILL '.
02 EETRN-C-ROLL-ECODE       PIC S9(9) COMP.
*
*** CBLEEMEM(' GETWK  ')
01 EEMEM-GETWK-ARG1.
02 EEMEM-GETWK-REQUEST      PIC X(8) VALUE ' GETWK  '.
02 EEMEM-GETWK-STATUS-CODE  PIC X(5).
02 FILLER                   PIC X(3).
02 EEMEM-GETWK-FLAGS        PIC S9(9) COMP VALUE ZERO.
01 EEMEM-GETWK-ARG2.
02 EEMEM-GETWK-SEG-TYPE     PIC X(1) VALUE ' U '.
02 FILLER                   PIC X(3).
02 EEMEM-GETWK-SEG-SIZE     PIC 9(9) COMP VALUE 128.
02 EEMEM-GETWK-SEG-PT       ADDRESS.
*
*** CBLEEMEM(' RLSWK  ')
01 EEMEM-RLSWK-ARG1.
02 EEMEM-RLSWK-REQUEST      PIC X(8) VALUE ' RLSWK  '.
02 EEMEM-RLSWK-STATUS-CODE  PIC X(5).
02 FILLER                   PIC X(3).
02 EEMEM-RLSWK-FLAGS        PIC S9(9) COMP VALUE ZERO.
01 EEMEM-RLSWK-ARG2.
02 EEMEM-RLSWK-SEG-PT       ADDRESS.
*
*** CBLEETIM(' EXECAP  ')
01 EETIM-EXECAP-ARG1.
02 EETIM-EXECAP-REQUEST     PIC X(8) VALUE ' EXECAP  '.
02 EETIM-EXECAP-STATUS-CODE PIC X(5).
02 FILLER                   PIC X(3).
02 EETIM-EXECAP-FLAGS       PIC S9(9) COMP VALUE 0.
01 EETIM-EXECAP-ARG2.
02 EETIM-EXECAP-ACTTYPE     PIC X(4) VALUE ' INST '.
02 EETIM-EXECAP-ACTION      PIC X(4) VALUE ' INTV '.

```

```

02 EETIM-EXECAP-ACTIVE      PIC 9(9) COMP VALUE 10.
02 EETIM-EXECAP-REQID      PIC 9(9) COMP VALUE 0.
02 EETIM-EXECAP-SERVICE    PIC X(32) VALUE 'srv02 '.
02 EETIM-EXECAP-PRIORITY   PIC X(4) VALUE 'HI '.
01 EETIM-EXECAP-ARG3 ADDRESSED BY TIM-PT.
02 EETIM-EXECAP-DATA_LEN    PIC 9(9) COMP.
02 EETIM-EXECAP-DATA        PIC X(64).
*
*** CBLEEDBQ('MSGPUT ')
01 EEDBQ-MSGPUT-ARG1.
02 EEDBQ-MSGPUT-REQUEST     PIC X(8) VALUE 'MSGPUT '.
02 EEDBQ-MSGPUT-STATUS-CODE PIC X(5).
02 FILLER                   PIC X(3).
02 EEDBQ-MSGPUT-FLAGS       PIC S9(9).
02 EEDBQ-MSGPUT-DBQNAME     PIC X(32) VALUE 'DBQ001'.
01 EEDBQ-MSGPUT-ARG2.
02 EEDBQ-MSGPUT-DATA_LEN    PIC 9(9) COMP.
02 EEDBQ-MSGPUT-DATA        PIC X(128).
*
*****
*  引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
02 ARG-INDATA  PIC X(128).
01 IN-LEN.
02 ARG-INLEN   PIC S9(9) COMP.
01 OUT-DATA.
02 ARG-OUTDATA PIC X(8).
01 OUT-LEN.
02 ARG-OUTLEN  PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*  プログラム開始
*****
PROCEDURE DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*****
*  IN-DATAをタイマトラン送信用データに変更
*****
MOVE IN-DATA TO SQL-ARG1.
*
MOVE NAM-ARG OF SQL-ARG1 (1:NAM-LEN OF SQL-ARG1)
    TO NAM-ARG OF SQL-ARG2.
MOVE SEX-ARG OF SQL-ARG1 (1:SEX-LEN OF SQL-ARG1)
    TO SEX-ARG OF SQL-ARG2.
MOVE AGE-ARG OF SQL-ARG1 (1:AGE-LEN OF SQL-ARG1)
    TO AGE-ARG OF SQL-ARG2.
MOVE SALE-ARG OF SQL-ARG1 TO SALE-ARG OF SQL-ARG2.
MOVE SERVICE_NAME OF EERPC_INTERFACE_TBL
    (1:SERVICE_LEN OF EERPC_INTERFACE_TBL)
    TO SRV-ARG OF SQL-ARG2 .
*
*****
*  GETWKを発行してタイマトラン送信用の領域を確保
*****

```

```

*** GETWK発行
CALL 'CBLEEMEM'
    USING EEMEM-GETWK-ARG1 EEMEM-GETWK-ARG2.
*** GETWK成否判定
    IF EEMEM-GETWK-STATUS-CODE OF EEMEM-GETWK-ARG1
        NOT = EE_OK THEN
*****
*   GETWK失敗
*****
*** 領域確保失敗メッセージの表示
*** LOGPRINT引数設定
    MOVE PG MID      TO
        EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1
    MOVE MSGID52     TO
        EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1
    MOVE MSG52       TO
        EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
    MOVE MSGLEN52    TO
        EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2
    CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
        EELOG-PRINT-ARG3
    GO TO MN-END
    END-IF.
*
*****
*   GETWK成功>確保した領域を用い送信データを設定
*****
*** アドレス受け渡し
    COMPUTE TIM-PT = EEMEM-GETWK-SEG-PT OF EEMEM-GETWK-ARG2.
*** 送信DATA設定
    MOVE 64 TO EETIM-EXECAP-DATA LEN OF EETIM-EXECAP-ARG3.
    MOVE SQL-ARG2 TO EETIM-EXECAP-DATA OF EETIM-EXECAP-ARG3.
*
*****
*   EXECAP発行
*****
    CALL 'CBLEETIM' USING EETIM-EXECAP-ARG1
        EETIM-EXECAP-ARG2 EETIM-EXECAP-ARG3.
*****
*   RLSWK発行(確保領域の解放)
*****
*** 解放する領域の設定
    COMPUTE EEMEM-RLSWK-SEG-PT OF EEMEM-RLSWK-ARG2
        = EEMEM-GETWK-SEG-PT OF EEMEM-GETWK-ARG2.
*** RLSWK発行
    CALL 'CBLEEMEM'
        USING EEMEM-RLSWK-ARG1 EEMEM-RLSWK-ARG2.
*
*****
*   EXECAP成否判定
*****
    IF EETIM-EXECAP-STATUS-CODE OF EETIM-EXECAP-ARG1
        NOT = EE_OK THEN
*****
*   EXECAP:失敗>LOG出力/ROLLBACK発行
*****
*** タイマトラン失敗メッセージの表示
*** LOGPRINT引数設定

```



```

MOVE PGMID      TO
  EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1
MOVE MSGID53    TO
  EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1
MOVE MSG53      TO
  EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
MOVE MSGLEN53   TO
  EELOG-PRINT-MSG-LEN  OF EELOG-PRINT-ARG2
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
  EELOG-PRINT-ARG3
END-CALL

*
*** ROLLBACK(KILL)発行
CALL 'CBLEETR'N' USING EETR'N-C-ROLL-ARG
*** ROLLBACK成否判定
  IF EETR'N-C-ROLL-STATUS-CODE OF EETR'N-C-ROLL-ARG
    NOT = EE_OK THEN
*****
*   ROLLBACK失敗
*****
*** ロールバック失敗メッセージの表示
*** LOGPRINT引数設定
  MOVE PGMID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1
  MOVE MSGID54    TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1
  MOVE MSG54      TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
  MOVE MSGLEN54   TO
    EELOG-PRINT-MSG-LEN  OF EELOG-PRINT-ARG2
  CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3
  END-IF
GO TO MN-END
END-IF.

*
*** 送信DATA設定
MOVE 0 TO EEDBQ-MSGPUT-FLAGS OF EEDBQ-MSGPUT-ARG1.
MOVE 124 TO EEDBQ-MSGPUT-DATA_LEN OF EEDBQ-MSGPUT-ARG2.
MOVE IN-DATA TO EEDBQ-MSGPUT-DATA OF EEDBQ-MSGPUT-ARG2.
*** MSGPUT発行
CALL 'CBLEEDBQ'
  USING EEDBQ-MSGPUT-ARG1 EEDBQ-MSGPUT-ARG2.

*
*****
*   MSGPUT成否判定
*****
  IF EEDBQ-MSGPUT-STATUS-CODE OF EEDBQ-MSGPUT-ARG1
    NOT = EE_OK THEN
*****
*   MSGPUT:失敗>LOG出力/ROLLBACK発行
*****
*** タイマトラン失敗メッセージの表示
*** LOGPRINT引数設定
  MOVE PGMID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1
  MOVE MSGID55    TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1

```

```

MOVE MSG55      TO
  EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
MOVE MSGLEN55 TO
  EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
  EELOG-PRINT-ARG3
END-CALL

*
*** ROLLBACK(KILL)発行
CALL 'CBLEETR'N' USING EETR'N-C-ROLL-ARG
*** ROLLBACK成否判定
  IF EETR'N-C-ROLL-STATUS-CODE OF EETR'N-C-ROLL-ARG
    NOT = EE_OK THEN
*****
*   ROLLBACK失敗
*****
*** ロールバック失敗メッセージの表示
*** LOGPRINT引数設定
  MOVE PGMID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1
  MOVE MSGID54    TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1
  MOVE MSG54      TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
  MOVE MSGLEN54   TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2
  CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3
  END-IF
END-IF.

*
MN-END.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_UAP.

*
*
*****
**   目的      : エラートランザクション1処理
**   機能      : エラートランザクション1処理を実行
*****
*****
*   見出し
*****
IDENTIFICATION DIVISION.
*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_E1.
*
*****
*   環境部
*****
ENVIRONMENT      DIVISION.

```

```

*
*****
*   変数宣言部
*****
DATA          DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID56        PIC X(12) VALUE 'KFSB05600-E '.
01 MSG56          PIC X(64) VALUE
'SERVER:エラートランザクション1を起動します.'.
01 MSGLEN56       PIC 9(9)  COMP VALUE 64.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
02 EELOG-PRINT-REQUEST      PIC X(8) VALUE 'PRINT '.
02 EELOG-PRINT-STATUS-CODE  PIC X(5).
02 FILLER                   PIC X(3).
02 EELOG-PRINT-FLAGS        PIC S9(9) COMP VALUE ZERO.
02 EELOG-PRINT-MSG-ID        PIC X(12).
02 EELOG-PRINT-PGM-ID        PIC X(3).
01 EELOG-PRINT-ARG2.
02 EELOG-PRINT-MSG-LEN        PIC 9(9) COMP.
02 EELOG-PRINT-MSG            PIC X(128).
01 EELOG-PRINT-ARG3.
02 EELOG-PRINT-INFO           PIC S9(9) COMP VALUE ZERO.
*
*****
*   引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
02 ARG-INDATA  PIC X(128).
01 IN-LEN.
02 ARG-INLEN   PIC S9(9) COMP.
01 OUT-DATA.
02 ARG-OUTDATA PIC X(8).
01 OUT-LEN.
02 ARG-OUTLEN   PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
PROCEDURE          DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****

```

```

*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PG MID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1.
MOVE MSGID56     TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1.
MOVE MSG56       TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2.
MOVE MSGLEN56    TO
    EELOG-PRINT-MSG-LEN  OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.

*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_E1.

*
*****
**   目的      : エラートランザクション2処理
**   機能      : エラートランザクション2処理を実行
*****
*****
*   見出し
*****
IDENTIFICATION DIVISION.

*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_E2.

*
*****
*   環境部
*****
ENVIRONMENT      DIVISION.

*
*****
*   変数宣言部
*****
DATA             DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5) VALUE '00000'.

*
*** LOGPRINT設定 *****
01 PG MID         PIC X(3) VALUE 'SP '.

*
*** サンプルメッセージ(トラン起動)
01 MSGID57        PIC X(12) VALUE 'KFSB05700-E '.
01 MSG57          PIC X(64) VALUE
    'SERVER:エラートランザクション2を起動します.'.
01 MSGLEN57       PIC 9(9)  COMP VALUE 64.

```

```

*
*** API引数設定 *****
*** CBLELOG('PRINT ')
01 EELOG-PRINT-ARG1.
    02 EELOG-PRINT-REQUEST      PIC X(8) VALUE 'PRINT '.
    02 EELOG-PRINT-STATUS-CODE  PIC X(5).
    02 FILLER                   PIC X(3).
    02 EELOG-PRINT-FLAGS        PIC S9(9) COMP VALUE ZERO.
    02 EELOG-PRINT-MSG-ID       PIC X(12).
    02 EELOG-PRINT-PGM-ID       PIC X(3).
01 EELOG-PRINT-ARG2.
    02 EELOG-PRINT-MSG-LEN      PIC 9(9) COMP.
    02 EELOG-PRINT-MSG         PIC X(128).
01 EELOG-PRINT-ARG3.
    02 EELOG-PRINT-INFO        PIC S9(9) COMP VALUE ZERO.
*
*****
*   引数宣言
*****
    LINKAGE SECTION.
*
01 IN-DATA.
    02 ARG-INDATA    PIC X(128).
01 IN-LEN.
    02 ARG-INLEN     PIC S9(9) COMP.
01 OUT-DATA.
    02 ARG-OUTDATA   PIC X(8).
01 OUT-LEN.
    02 ARG-OUTLEN    PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
    PROCEDURE        DIVISION
        USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
    MOVE PGMID      TO
        EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1.
    MOVE MSGID57    TO
        EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1.
    MOVE MSG57      TO
        EELOG-PRINT-MSG    OF EELOG-PRINT-ARG2.
    MOVE MSGLEN57   TO
        EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
    CALL 'CBLELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
        EELOG-PRINT-ARG3.
*
*****
*   プログラム終了
*****
    EXIT PROGRAM.
    END PROGRAM SAMPLE_E2.
*

```

```

*****
**      目的      : エラートランザクション3処理
**      機能      : エラートランザクション3処理を実行
*****
**      見出し
*****
IDENTIFICATION DIVISION.
*
*****
**      関数名
*****
PROGRAM-ID.      SAMPLE_E3.
*
*****
**      環境部
*****
ENVIRONMENT DIVISION.
*
*****
**      変数宣言部
*****
DATA DIVISION.
*****
**      変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID58        PIC X(12) VALUE 'KFSB05800-E '.
01 MSG58          PIC X(64) VALUE
'SERVER:エラートランザクション3を起動します.'.
01 MSGLEN58       PIC 9(9)  COMP VALUE 64.
*** サンプルメッセージ(RPC処理失敗)
01 MSGID55        PIC X(12) VALUE 'KFSB05500-E '.
01 MSG55          PIC X(64) VALUE
'SERVER:RPC処理に失敗しました.'.
01 MSGLEN55       PIC 9(9)  COMP VALUE 64.
*
*** RPC設定 *****v0101**
01 RPC-FLAGS      PIC S9(9) COMP VALUE 0.
01 RPC-SVNAME      PIC X(32) VALUE 'outSRV '.
01 RPC-SVGROUP     PIC X(32) VALUE 'outSPP '.
01 RPC-OUTDATALEN  PIC 9(9)  COMP VALUE 8.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
02 EELOG-PRINT-REQUEST PIC X(8) VALUE 'PRINT '.
02 EELOG-PRINT-STATUS-CODE PIC X(5).
02 FILLER              PIC X(3).
02 EELOG-PRINT-FLAGS    PIC S9(9) COMP VALUE ZERO.
02 EELOG-PRINT-MSG-ID   PIC X(12).

```

```

    02 EELOG-PRINT-PGM-ID      PIC X(3).
01 EELOG-PRINT-ARG2.
    02 EELOG-PRINT-MSG-LEN     PIC 9(9) COMP.
    02 EELOG-PRINT-MSG        PIC X(128).
01 EELOG-PRINT-ARG3.
    02 EELOG-PRINT-INFO       PIC S9(9) COMP VALUE ZERO.
*
*** CBLEERPC('CALL      ')
01 EERPC-CALL-ARG1.
    02 EERPC-CALL-REQUEST      PIC X(8) VALUE 'CALL      '.
    02 EERPC-CALL-STATUS-CODE  PIC X(5).
    02 FILLER                  PIC X(3).
    02 EERPC-CALL-FLAGS        PIC S9(9) COMP VALUE ZERO.
    02 EERPC-CALL-DESCRIPTOR   PIC S9(9) COMP.
    02 EERPC-CALL-SVNAME       PIC X(32).
    02 EERPC-CALL-SVGROUP      PIC X(32).
01 EERPC-CALL-ARG2.
    02 EERPC-CALL-INDATALEN    PIC 9(9) COMP.
    02 EERPC-CALL-INDATA      PIC X(128).
01 EERPC-CALL-ARG3.
    02 EERPC-CALL-OUTDATALEN   PIC 9(9) COMP.
    02 EERPC-CALL-OUTDATA     PIC X(8).
*
*****
*   引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
    02 ARG-INDATA    PIC X(128).
01 IN-LEN.
    02 ARG-INLEN     PIC S9(9) COMP.
01 OUT-DATA.
    02 ARG-OUTDATA   PIC X(8).
01 OUT-LEN.
    02 ARG-OUTLEN    PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
PROCEDURE      DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PGMID      TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1.
MOVE MSGID58    TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1.
MOVE MSG58      TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2.
MOVE MSGLEN58   TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.

```

```

*
*****
*   RPC送信
*****
*** RPC送信データの設定
MOVE ARG-INDATA OF IN-DATA TO
    EERPC-CALL-INDATA OF EERPC-CALL-ARG2.
MOVE ARG-INLEN OF IN-LEN TO
    EERPC-CALL-INDATALEN OF EERPC-CALL-ARG2.
MOVE RPC-FLAGS      TO EERPC-CALL-FLAGS      OF EERPC-CALL-ARG1.
MOVE RPC-SVNAME     TO EERPC-CALL-SVNAME     OF EERPC-CALL-ARG1.
MOVE RPC-SVGROUP    TO EERPC-CALL-SVGROUP    OF EERPC-CALL-ARG1.
MOVE RPC-OUTDATALEN TO EERPC-CALL-OUTDATALEN OF EERPC-CALL-ARG3.
*** RPC発行
CALL 'CBLEERPC'
    USING EERPC-CALL-ARG1 EERPC-CALL-ARG2 EERPC-CALL-ARG3.
*** RPC成否判定
    IF EERPC-CALL-STATUS-CODE OF EERPC-CALL-ARG1
        NOT = EE_OK THEN
*****
*   RPC送信失敗
*****
*** RPC処理失敗メッセージの表示
*** LOGPRINT引数設定
MOVE PGMID      TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1
MOVE MSGID55    TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1
MOVE MSG55      TO
    EELOG-PRINT-MSG    OF EELOG-PRINT-ARG2
MOVE MSGLEN55   TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3
END-IF.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_E3.
*
*****
**   目的      : エラートランザクション4処理
**   機能      : エラートランザクション4処理を実行
*****
*****
*   見出し
*****
IDENTIFICATION DIVISION.
*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_E4.
*
*****
*   環境部

```



```

*****
ENVIRONMENT      DIVISION.
*
*****
*   変数宣言部
*****
DATA            DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID59        PIC X(12) VALUE 'KFSB05900-E '.
01 MSG59          PIC X(64) VALUE
'SERVER:エラートランザクション4を起動します.'.
01 MSGLEN59       PIC 9(9)  COMP VALUE 64.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
02 EELOG-PRINT-REQUEST      PIC X(8) VALUE 'PRINT '.
02 EELOG-PRINT-STATUS-CODE  PIC X(5).
02 FILLER                   PIC X(3).
02 EELOG-PRINT-FLAGS        PIC S9(9) COMP VALUE ZERO.
02 EELOG-PRINT-MSG-ID       PIC X(12).
02 EELOG-PRINT-PGM-ID       PIC X(3).
01 EELOG-PRINT-ARG2.
02 EELOG-PRINT-MSG-LEN       PIC 9(9) COMP.
02 EELOG-PRINT-MSG           PIC X(128).
01 EELOG-PRINT-ARG3.
02 EELOG-PRINT-INFO         PIC S9(9) COMP VALUE ZERO.
*
*****
*   引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
02 ARG-INDATA  PIC X(128).
01 IN-LEN.
02 ARG-INLEN   PIC S9(9) COMP.
01 OUT-DATA.
02 ARG-OUTDATA PIC X(8).
01 OUT-LEN.
02 ARG-OUTLEN  PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
PROCEDURE      DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.

```

```

*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PGMID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1.
MOVE MSGID59    TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1.
MOVE MSG59      TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2.
MOVE MSGLEN59   TO
    EELOG-PRINT-MSG-LEN  OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_E4.
*
*****
**   目的      :   初期化トランザクション処理
**   機能      :   初期化トランザクション処理を実行
*****
*   見出し
*****
IDENTIFICATION DIVISION.
*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_MI.
*
*****
*   環境部
*****
ENVIRONMENT DIVISION.
*
*****
*   変数宣言部
*****
DATA DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK      PIC X(5) VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID      PIC X(3) VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID62    PIC X(12) VALUE 'KFSB06200-I '.
01 MSG62      PIC X(64) VALUE

```

```

      'SERVER:サンプルUAPを開始します。'.
01 MSGLEN62      PIC 9(9)  COMP VALUE 64.
*
*** API引数設定 *****
*** CBLEELOG('PRINT  ')
01 EELOG-PRINT-ARG1.
    02 EELOG-PRINT-REQUEST      PIC X(8) VALUE 'PRINT  '.
    02 EELOG-PRINT-STATUS-CODE  PIC X(5).
    02 FILLER                   PIC X(3).
    02 EELOG-PRINT-FLAGS       PIC S9(9) COMP VALUE ZERO.
    02 EELOG-PRINT-MSG-ID      PIC X(12).
    02 EELOG-PRINT-PGM-ID      PIC X(3).
01 EELOG-PRINT-ARG2.
    02 EELOG-PRINT-MSG-LEN      PIC 9(9) COMP.
    02 EELOG-PRINT-MSG         PIC X(128).
01 EELOG-PRINT-ARG3.
    02 EELOG-PRINT-INFO        PIC S9(9) COMP VALUE ZERO.
*
*****
*   引数宣言
*****
      LINKAGE SECTION.
*
01 IN-DATA.
    02 ARG-INDATA      PIC X(128).
01 IN-LEN.
    02 ARG-INLEN      PIC S9(9) COMP.
01 OUT-DATA.
    02 ARG-OUTDATA    PIC X(8).
01 OUT-LEN.
    02 ARG-OUTLEN     PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
      PROCEDURE      DIVISION
      USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
      MOVE PGMID      TO
          EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1.
      MOVE MSGID62    TO
          EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1.
      MOVE MSG62      TO
          EELOG-PRINT-MSG     OF EELOG-PRINT-ARG2.
      MOVE MSGLEN62   TO
          EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
      CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
          EELOG-PRINT-ARG3.
*
*****
*   プログラム終了
*****
      EXIT PROGRAM.

```

```

END PROGRAM SAMPLE_MI.
*
*****
**      目的      :   終了ランザクション
**      機能      :   終了ランザクション処理を実行
*****
*      見出し
*****
IDENTIFICATION   DIVISION.
*
*****
*      関数名
*****
PROGRAM-ID.      SAMPLE_ME.
*
*****
*      環境部
*****
ENVIRONMENT      DIVISION.
*
*****
*      変数宣言部
*****
DATA             DIVISION.
*****
*      変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID63        PIC X(12) VALUE 'KFSB06300-I '.
01 MSG63          PIC X(64) VALUE
'SERVER:サンプルUAPを終了します.'.
01 MSGLEN63       PIC 9(9)  COMP VALUE 64.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
    02 EELOG-PRINT-REQUEST      PIC X(8) VALUE 'PRINT '.
    02 EELOG-PRINT-STATUS-CODE  PIC X(5).
    02 FILLER                   PIC X(3).
    02 EELOG-PRINT-FLAGS        PIC S9(9) COMP VALUE ZERO.
    02 EELOG-PRINT-MSG-ID       PIC X(12).
    02 EELOG-PRINT-PGM-ID       PIC X(3).
01 EELOG-PRINT-ARG2.
    02 EELOG-PRINT-MSG-LEN      PIC 9(9) COMP.
    02 EELOG-PRINT-MSG         PIC X(128).
01 EELOG-PRINT-ARG3.
    02 EELOG-PRINT-INFO        PIC S9(9) COMP VALUE ZERO.
*
*****
*      引数宣言

```

```

*****
LINKAGE SECTION.
*
01 IN-DATA.
    02 ARG-INDATA    PIC X(128).
01 IN-LEN.
    02 ARG-INLEN     PIC S9(9) COMP.
01 OUT-DATA.
    02 ARG-OUTDATA   PIC X(8).
01 OUT-LEN.
    02 ARG-OUTLEN    PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
PROCEDURE          DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PG MID       TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1.
MOVE MSGID63      TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1.
MOVE MSG63        TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2.
MOVE MSGLEN63     TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_ME.
*
*****
**   目的          :   タイマトランザクション処理
**   機能          :   タイマトランザクション処理を実行
*****
*****
*   見出し
*****
IDENTIFICATION DIVISION.
*
*****
*   関数名
*****
PROGRAM-ID.        SAMPLE_TM.
*
*****
*   環境部
*****
ENVIRONMENT DIVISION.

```

```

*
*****
*   変数宣言部
*****
DATA          DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK      PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID      PIC X(3)  VALUE 'SP '.
*
*** サンプルメッセージ(トラン起動)
01 MSGID64    PIC X(12) VALUE 'KFSB06400-I '.
01 MSG64      PIC X(64) VALUE
               'SERVER:タイマトランザクションを起動します.'.
01 MSGLEN64   PIC 9(9)  COMP VALUE 64.
*
*** サンプルメッセージ(RPC処理失敗)
01 MSGID55    PIC X(12) VALUE 'KFSB05500-E '.
01 MSG55      PIC X(64) VALUE
               'SERVER:RPC処理に失敗しました.'.
01 MSGLEN55   PIC 9(9)  COMP VALUE 64.
*
*** RPC設定 *****v0101***
01 RPC-FLAGS  PIC S9(9) COMP VALUE 0.
01 RPC-SVNAME PIC X(32) VALUE 'outSRV '.
01 RPC-SVGROUP PIC X(32) VALUE 'outSPP '.
01 RPC-OUTDATALEN PIC 9(9) COMP VALUE 8.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
   02 EELOG-PRINT-REQUEST PIC X(8) VALUE 'PRINT '.
   02 EELOG-PRINT-STATUS-CODE PIC X(5).
   02 FILLER PIC X(3).
   02 EELOG-PRINT-FLAGS PIC S9(9) COMP VALUE ZERO.
   02 EELOG-PRINT-MSG-ID PIC X(12).
   02 EELOG-PRINT-PGM-ID PIC X(3).
01 EELOG-PRINT-ARG2.
   02 EELOG-PRINT-MSG-LEN PIC 9(9) COMP.
   02 EELOG-PRINT-MSG PIC X(128).
01 EELOG-PRINT-ARG3.
   02 EELOG-PRINT-INFO PIC S9(9) COMP VALUE ZERO.
*
*** CBLEERPC('CALL ')
01 EERPC-CALL-ARG1.
   02 EERPC-CALL-REQUEST PIC X(8) VALUE 'CALL '.
   02 EERPC-CALL-STATUS-CODE PIC X(5).
   02 FILLER PIC X(3).
   02 EERPC-CALL-FLAGS PIC S9(9) COMP VALUE ZERO.
   02 EERPC-CALL-DESCRIPTOR PIC S9(9) COMP.
   02 EERPC-CALL-SVNAME PIC X(32).
   02 EERPC-CALL-SVGROUP PIC X(32).
01 EERPC-CALL-ARG2.

```

```

02 EERPC-CALL-INDATALEN      PIC 9(9) COMP.
02 EERPC-CALL-INDATA        PIC X(64).
01 EERPC-CALL-ARG3.
02 EERPC-CALL-OUTDATALEN    PIC 9(9) COMP.
02 EERPC-CALL-OUTDATA      PIC X(8).
*
*****
*   引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
02 ARG-INDATA      PIC X(128).
01 IN-LEN.
02 ARG-INLEN      PIC S9(9) COMP.
01 OUT-DATA.
02 ARG-OUTDATA    PIC X(8).
01 OUT-LEN.
02 ARG-OUTLEN     PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
*   プログラム開始
*****
PROCEDURE      DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
*   トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PGMID      TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1.
MOVE MSGID64    TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1.
MOVE MSG64      TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2.
MOVE MSGLEN64   TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.
*
*****
*   RPC送信
*****
*** RPC送信データの設定
MOVE ARG-INDATA OF IN-DATA TO
    EERPC-CALL-INDATA OF EERPC-CALL-ARG2.
MOVE ARG-INLEN OF IN-LEN TO
    EERPC-CALL-INDATALEN OF EERPC-CALL-ARG2.
MOVE RPC-FLAGS      TO EERPC-CALL-FLAGS      OF EERPC-CALL-ARG1.
MOVE RPC-SVNAME     TO EERPC-CALL-SVNAME     OF EERPC-CALL-ARG1.
MOVE RPC-SVGROUP    TO EERPC-CALL-SVGROUP    OF EERPC-CALL-ARG1.
MOVE RPC-OUTDATALEN TO EERPC-CALL-OUTDATALEN OF EERPC-CALL-ARG3.
*** RPC発行
CALL 'CBLEERPC'
    USING EERPC-CALL-ARG1 EERPC-CALL-ARG2 EERPC-CALL-ARG3.
*** RPC成否判定

```

```

IF EERPC-CALL-STATUS-CODE OF EERPC-CALL-ARG1
  NOT = EE_OK THEN
*****
*   RPC送信失敗
*****
*** RPC処理失敗メッセージの表示
*** LOGPRINT引数設定
  MOVE PGMID      TO
    EELOG-PRINT-PGM-ID  OF EELOG-PRINT-ARG1
  MOVE MSGID55     TO
    EELOG-PRINT-MSG-ID  OF EELOG-PRINT-ARG1
  MOVE MSG55       TO
    EELOG-PRINT-MSG      OF EELOG-PRINT-ARG2
  MOVE MSGLEN55    TO
    EELOG-PRINT-MSG-LEN  OF EELOG-PRINT-ARG2
  CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3
END-IF.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_TM.
*
*
*****
**   目的      :   DBキューの読出しトランザクション
**   機能      :   DBキューの読出しトランザクションを実行する
*****
*****
*   見出し
*****
IDENTIFICATION  DIVISION.
*
*****
*   関数名
*****
PROGRAM-ID.      SAMPLE_DBQ.
*
*****
*   環境部
*****
ENVIRONMENT      DIVISION.
*
*****
*   変数宣言部
*****
DATA              DIVISION.
*****
*   変数/定数宣言
*****
WORKING-STORAGE SECTION.
*** リターンコード値 *****
01 EE_OK          PIC X(5)  VALUE '00000'.
*
*** LOGPRINT設定 *****
01 PGMID          PIC X(3)  VALUE 'SP '.

```



```

*
*** サンプルメッセージ(DBQトラン起動)
01 MSGID65      PIC X(12) VALUE 'KFSB06500-I '.
01 MSG65        PIC X(128) VALUE
' SERVER:DBキューの読み出しトランザクションを起動します.'
01 MSGLEN65     PIC 9(9)  COMP VALUE 128.
*** サンプルメッセージ(RPC処理失敗)
01 MSGID55      PIC X(12) VALUE 'KFSB05500-E '.
01 MSG55        PIC X(64) VALUE
' SERVER:RPC処理に失敗しました.'
01 MSGLEN55     PIC 9(9)  COMP VALUE 64.
*
*** RPC設定 *****v0101***
01 RPC-FLAGS    PIC S9(9) COMP VALUE 0.
01 RPC-SVNAME   PIC X(32) VALUE 'outSRV '.
01 RPC-SVGROUP  PIC X(32) VALUE 'outSPP '.
01 RPC-OUTDATALEN PIC 9(9)  COMP VALUE 8.
*
*** API引数設定 *****
*** CBLEELOG('PRINT ')
01 EELOG-PRINT-ARG1.
02 EELOG-PRINT-REQUEST PIC X(8) VALUE 'PRINT '.
02 EELOG-PRINT-STATUS-CODE PIC X(5).
02 FILLER PIC X(3).
02 EELOG-PRINT-FLAGS PIC S9(9) COMP VALUE ZERO.
02 EELOG-PRINT-MSG-ID PIC X(12).
02 EELOG-PRINT-PGM-ID PIC X(3).
01 EELOG-PRINT-ARG2.
02 EELOG-PRINT-MSG-LEN PIC 9(9) COMP.
02 EELOG-PRINT-MSG PIC X(128).
01 EELOG-PRINT-ARG3.
02 EELOG-PRINT-INFO PIC S9(9) COMP VALUE ZERO.
*
*** CBLEERPC('CALL ')
01 EERPC-CALL-ARG1.
02 EERPC-CALL-REQUEST PIC X(8) VALUE 'CALL '.
02 EERPC-CALL-STATUS-CODE PIC X(5).
02 FILLER PIC X(3).
02 EERPC-CALL-FLAGS PIC S9(9) COMP VALUE ZERO.
02 EERPC-CALL-DESCRIPTOR PIC S9(9) COMP.
02 EERPC-CALL-SVNAME PIC X(32).
02 EERPC-CALL-SVGROUP PIC X(32).
01 EERPC-CALL-ARG2.
02 EERPC-CALL-INDATALEN PIC 9(9) COMP.
02 EERPC-CALL-INDATA PIC X(64).
01 EERPC-CALL-ARG3.
02 EERPC-CALL-OUTDATALEN PIC 9(9) COMP.
02 EERPC-CALL-OUTDATA PIC X(8).
*
*****
* 引数宣言
*****
LINKAGE SECTION.
*
01 IN-DATA.
02 ARG-INDATA PIC X(128).
01 IN-LEN.
02 ARG-INLEN PIC S9(9) COMP.

```

```

01 OUT-DATA.
    02 ARG-OUTDATA PIC X(8).
01 OUT-LEN.
    02 ARG-OUTLEN PIC S9(9) COMP.
COPY EERPCSRV.
*
*****
* プログラム開始
*****
PROCEDURE DIVISION
    USING IN-DATA IN-LEN OUT-DATA OUT-LEN EERPC_INTERFACE_TBL.
*
*****
* トランザクション起動メッセージの表示
*****
*** LOGPRINT引数設定
MOVE PGMID TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1.
MOVE MSGID65 TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1.
MOVE MSG65 TO
    EELOG-PRINT-MSG OF EELOG-PRINT-ARG2.
MOVE MSGLEN65 TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2.
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
    EELOG-PRINT-ARG3.
*
*****
* RPC送信
*****
*** RPC送信データの設定
MOVE ARG-INDATA OF IN-DATA TO
    EERPC-CALL-INDATA OF EERPC-CALL-ARG2.
MOVE ARG-INLEN OF IN-LEN TO
    EERPC-CALL-INDATALEN OF EERPC-CALL-ARG2.
MOVE RPC-FLAGS TO EERPC-CALL-FLAGS OF EERPC-CALL-ARG1.
MOVE RPC-SVNAME TO EERPC-CALL-SVNAME OF EERPC-CALL-ARG1.
MOVE RPC-SVGROUP TO EERPC-CALL-SVGROUP OF EERPC-CALL-ARG1.
MOVE RPC-OUTDATALEN TO EERPC-CALL-OUTDATALEN OF EERPC-CALL-ARG3.
*** RPC発行
CALL 'CBLEERPC'
    USING EERPC-CALL-ARG1 EERPC-CALL-ARG2 EERPC-CALL-ARG3.
*** RPC成否判定
IF EERPC-CALL-STATUS-CODE OF EERPC-CALL-ARG1
    NOT = EE_OK THEN
*****
* RPC送信失敗
*****
*** RPC処理失敗メッセージの表示
*** LOGPRINT引数設定
MOVE PGMID TO
    EELOG-PRINT-PGM-ID OF EELOG-PRINT-ARG1
MOVE MSGID55 TO
    EELOG-PRINT-MSG-ID OF EELOG-PRINT-ARG1
MOVE MSG55 TO
    EELOG-PRINT-MSG OF EELOG-PRINT-ARG2
MOVE MSGLEN55 TO
    EELOG-PRINT-MSG-LEN OF EELOG-PRINT-ARG2

```

```
CALL 'CBLEELOG' USING EELOG-PRINT-ARG1 EELOG-PRINT-ARG2
      EELOG-PRINT-ARG3
END-IF.
*
*****
*   プログラム終了
*****
EXIT PROGRAM.
END PROGRAM SAMPLE_DBQ.
*
```

付録

付録 A トランザクション種別

トランザクションマネージャが管理、実行するトランザクション種別の一覧を次の表に示します。

表 A-1 トランザクション種別一覧

トランザクション種別	トランザクション名称	UAP で実行
AH	UAP 履歴情報取得トランザクション	×
BS	オフラインバッチサービストランザクション	○
CM	コマンドトランザクション	×
CU	ユーザコマンドトランザクション	○
DQ	DB キューシステムトランザクション	×
DT	DB キュータイマトランザクション	×
E1	ERRTRN1	○
E2	ERRTRN2	○
E3	ERRTRN3	○
E4	ERRTRN4	○
ER	ERRTRNR	○
ES	ERRTRNS	○
FO	ファイル出力トランザクション	×
JI	共有リソース初期化トランザクション	○
ME	終了トランザクション	○
MI	初期化トランザクション	○
MN	サービストランザクション	○
MV	イベント通知トランザクション	○
OB	OBS システムトランザクション	×
RC	系切り替え制御トランザクション	×
RD	キュー・ログ回復トランザクション	×
RE	RPC エラー応答トランザクション	×
RF	待機系メモリ反映トランザクション	×
RL	MCP 後処理トランザクション	○
RM	再起動トランザクション	○
RP	rap トランザクション	×
RR	回復トランザクション	×

トランザクション種別	トランザクション名称	UAP で実行
SB	SDB トランザクション	×
SD	OTQ/送達確認送信トランザクション	×
SE	プロセス終了監視トランザクション	×
SI	オンライン開始監視トランザクション	×
SN	エラー再送トランザクション	×
ST	統計情報取得タイマトランザクション	×
TM	タイマトランザクション	○
TT	インタバルタイマトランザクション	×
TU	永続タイマ処理トランザクション	×
TW	永続タイマ監視トランザクション	×
UI	実行系連絡トランザクション	○
UT	ユーザ初期化トランザクション	○
XB	XDB コマンドトランザクション	×

(凡例)

○：UAP で実行できます。

×：UAP で実行できません。

付録 B 各バージョンの変更内容

各バージョンの変更内容を示します。

付録 B.1 変更内容 (3000-3-F52-70)

次の製品の変更内容 (3000-3-F52-70) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-91

追加・変更内容
次のコマンドを追加した。 • eedbqaltlsh • eedbqaltput • eedbqinhibit • eelsque • eenamupdate • eesgrpctl
次の API 関数でトランザクション種別の使用可否を変更した。 • ee_dbq_msgget • ee_dbq_msgput • ee_dbq_msgrefer • ee_dbq_msgrefer2
エラー要因コードに交替用メッセージ表書き込み機能使用時の説明を追加した。
参照できる DB キューの属性として、交替用メッセージ表書き込み機能の使用を指定できるようにした。
【TP1/FSP 限定】 交替用メッセージ表書き込み機能に関する注意事項を追加した。

次の製品の変更内容 (3000-3-F52-70) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-90

追加・変更内容
トランザクション引き継ぎ情報の領域、およびリトライ回数を追加した。
DB キューの API のデッドロックに関する注意事項を追加、変更した。

付録 B.2 変更内容 (3000-3-F52-60)

次の製品の変更内容 (3000-3-F52-60) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-89

追加・変更内容
適用 OS に AIX V7.2 を追加した。
ee_dbq_inquire2 の説明を変更した。
ee_dbq_msgput のリターン値を追加、およびリターン値の説明を変更した。
CBLEEDBQ('INQUIRE2')の説明を変更した。
CBLEEDBQ('MSGPUT ')のステータスコードを変更、およびステータスコードの説明を変更した。

次の製品の変更内容（3000-3-F52-60）を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-88

追加・変更内容
ee_rpc_reply_send の説明を変更した。
CBLEERPC('R-SEND ')の説明を変更した。

付録 B.3 変更内容（3000-3-F52-50）

次の製品の変更内容（3000-3-F52-50）を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-87

追加・変更内容
ee_logprint の説明を変更した。
CBLEELOG('PRINT ')の説明を変更した。

次の製品の変更内容（3000-3-F52-50）を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-86

追加・変更内容
EERPC_INTERFACE_TBL 構造体に rm_func を追加した。
一意名 5 にデータ名 AU(リソースマネージャ連携有無)を追加した。

付録 B.4 変更内容（3000-3-F52-40）

次の製品の変更内容（3000-3-F52-40）を表に示します。

- uCosminexus TP1/Server Base Enterprise Option(64) 07-85

追加・変更内容
非同期応答型 RPC で、サービス関数が設定した応答の長さを参照できるようにした。
• ee_rpc_call • ee_rpc_call_to • CBLEERPC('CALL ') • CBLEERPC('CALLTO ')
各バージョンの変更内容を追加した。

付録 B.5 変更内容 (3000-3-F52-30)

次の製品の変更内容 (3000-3-F52-30) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-84
- uCosminexus TP1/Server Base Enterprise Option(64) 07-84

追加・変更内容
各バージョンの変更内容を追加した。

次の製品の変更内容 (3000-3-F52-30) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-83
- uCosminexus TP1/Server Base Enterprise Option(64) 07-83

追加・変更内容
トラブルシュート情報として、トランザクション単位のユーザ情報を TASKTM ファイルに出力できるようにした。 これに伴い、次の関数を追加した。 • ee_trb_uinfo_tm • CBLEETRB('UINFOTM ')
メッセージログを標準出力に出力できるようにした。 これに伴い、次の関数の説明を変更した。 • ee_logprint • CBLEELOG('PRINT ')

次の製品の変更内容 (3000-3-F52-30) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-81
- uCosminexus TP1/Server Base Enterprise Option(64) 07-81

追加・変更内容
システム構築時に使用するコネクション数を制限できる一時クローズ機能を追加した。 これに伴い、次の運用コマンドを追加した。

追加・変更内容
• eerpcstat
自サービスグループ名を設定したときに自プロセス送信をできるようにした。 これに伴い、次の関数を変更した。 • ee_rpc_call • ee_rpc_cmtsend • CBLEERPC('CALL ') • CBLEERPC('CMTSND ')
次の関数のリターン値またはステータスコードの意味を変更した。 リターン値 EESCDER_INVALID_TRNTYPE を変更した関数 - ee_scd_msg_receive 【XTC 限定】 - ee_scd_msg_rollback 【XTC 限定】 ステータスコード 00875 の説明を変更した関数 - CBLEESCD('MSGRECV ') 【XTC 限定】 - CBLEESCD('MSGRLBK ') 【XTC 限定】
次の関数の注意事項を変更した。 • ee_thd_abdctl • CBLEETHD('ABDCTL ')
C リターン値 EEMCHER_PARAM_IN の COBOL ステータスコードを変更した。

付録 B.6 変更内容（3000-3-F52-20）

次の製品の変更内容（3000-3-F52-20）を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-80
- uCosminexus TP1/Server Base Enterprise Option(64) 07-80

追加・変更内容
通信先を指定したサービスの要求をできるようにした。 これに伴い、次の API 関数を追加した。 • ee_rpc_call_to • ee_rpc_cmtsend_to • CBLEERPC('CALLTO ') • CBLEERPC('CMTSNDTO') また、次の API 関数を変更した。 • ee_trn_chained_commit • CBLEETRNC('C-COMMIT')
l 相最適化の注意事項を追加した。
タイマトランザクションの取り消し方法を変更した。

追加・変更内容
これに伴い、次の API 関数を変更した。 • ee_tim_execap_cancel • CBLEETIM('CANCEL ')
タイマトランザクション起動時の誤差について、説明を変更した。
DB キューに書き込んだメッセージを読み出すトランザクションを、AP 間通信をすることなくユーザが独自にスケジュールする、ユーザキューアクセスをできるようにした。 これに伴い、次の API 関数を追加した。 • ee_dbq_msgget • CBLEEDBQ('MSGGET ')
DB キューにメッセージを書き込むときの順番を通番管理できるようにした。 これに伴い、次の API 関数を追加した。 • ee_dbq_clear • ee_dbq_msgrefer2 • CBLEEDBQ('CLEAR ') • CBLEEDBQ('MSGREFE2')
AP 間通信をする DB キューのスケジュール状態を読み出し停止にする条件を指定できるようにした。
DB キューから不正なメッセージを読み出したり、メッセージの読み出し時に障害が発生したりしたときに、メッセージをスキップすることで読み出しを停止することなく次のメッセージを読み出せるようにした。
自システムの TP1/EE が読み出すすべての DB キューのスケジュール状態を、1 回のコマンドの実行で読み出し停止にしたり、再開したりできるようにした。
運用コマンド実行機能で、TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否を変更した。
TP1/EE のオプション製品として、TP1/FSP をサポートした。 これに伴い、次の API 関数を変更した。 • ee_rpc_call • ee_rpc_cmtsend • ee_rpc_reply_send • ee_trn_chained_rollback • CBLEERPC('CALL ') • CBLEERPC('CMTSND ') • CBLEERPC('R-SEND ') • CBLEETRN('C-ROLL ') 次のトランザクションインタフェース情報を追加した。 • トランザクション起動要因 • サーバ再起動回数 • トランザクション起動時刻 • 送信元のノード識別子 • トランザクションレベル名長 • トランザクションレベル名 次のトランザクション種別を追加した。 AH, BS, CU, JI, SB, TU, TW, UT

追加・変更内容
別プロセスによるトランザクション回復機能を削除した。これに伴い、関連するコマンド、トランザクション種別を削除した。
DB キューのフォーマットタイプが 1 の DB キューに対して、DB キューの情報内容を照会できるようにした。これに伴い、次の API 関数を追加した。 • ee_dbq_inquire2 • CBLEEDBQ('INQUIRE2')
TP1/EE またはマシンが管理している現在時刻を取得できるようにした。 これに伴い、次の API 関数を追加した。 • ee_thd_gettime • CBLEETHD('GETTIME ')
トランザクションでの API 関数の使用可否を変更した。
次の API 関数に、リターン値 EECOMER_CNDBPP、またはステータスコード 00005 を追加した。 • ee_adm_call_command • ee_dbq_acsctl • ee_dbq_conctl • ee_dbq_inqsrv • ee_dbq_inquire • ee_dbq_msgput • ee_dbq_msgrefer • ee_dbq_obscancel • ee_dbq_obschdata • ee_dbq_obsend • ee_dbq_obsinquire • ee_dbq_obsrbskip • ee_dbq_obsrestart • ee_dbq_obsskip • ee_dbq_obsstart • ee_dbq_obsstop • ee_dbq_rbskip • ee_dbq_skip • ee_dbq_srvctl • ee_lck_get • ee_lck_release_all • ee_lck_release_byname • ee_mch_cmtsend • ee_mch_cmtsend_get • ee_mch_cmtsend_sync • ee_mch_otqbctl • ee_mch_otqinquire • ee_mch_otqskip • ee_mem_mdpstart • ee_rpc_call

追加・変更内容

- ee_rpc_cmtsend
- ee_rpc_discard_further_replies
- ee_rpc_discard_specific_reply
- ee_rpc_get_error_descriptor
- ee_rpc_get_watch_time
- ee_rpc_poll_any_replies
- ee_rpc_reply_send
- ee_rpc_reply_suspend
- ee_rpc_set_watch_time
- ee_scd_clear_rollback_cnt
- ee_scd_msg_receive
- ee_scd_msg_rollback
- ee_scd_msg_skip
- ee_thd_abdctl
- ee_thd_protend
- ee_thd_protstart
- ee_thd_thdcntget
- ee_thd_uifaget
- ee_tim_execap
- ee_tim_execap_cancel
- ee_tim_execap_cancel_all
- ee_trn_chained_commit
- ee_trn_chained_rollback
- ee_trn_getinfo
- ee_trn_rm_close_mark
- ee_trn_rollback_mark
- ee_trn_set_exp_time
- CBLEEADM('COMMAND ')
- CBLEEDBQ('ACSCCTL ')
- CBLEEDBQ('CONCTL ')
- CBLEEDBQ('INQSRV ')
- CBLEEDBQ('INQUIRE ')
- CBLEEDBQ('MSGPUT ')
- CBLEEDBQ('MSGREFER')
- CBLEEDBQ('OBSCAN ')
- CBLEEDBQ('OBSCDT ')
- CBLEEDBQ('OBSEND ')
- CBLEEDBQ('OBSINQ ')
- CBLEEDBQ('OBSRBSP ')
- CBLEEDBQ('OBSRST ')
- CBLEEDBQ('OBSSKIP ')
- CBLEEDBQ('OBSSTART')
- CBLEEDBQ('OBSSTOP ')

追加・変更内容

- CBLEEDBQ('RBSKIP ')
- CBLEEDBQ('SKIP ')
- CBLEEDBQ('SRVCTL ')
- CBLEELCK('GET ')
- CBLEELCK('RELALL ')
- CBLEELCK('RELNAME ')
- CBLEEMCH('CMTSND ')
- CBLEEMCH('CMTSNDGT')
- CBLEEMCH('CMTSND SY')
- CBLEEMCH('OTQBKCTL')
- CBLEEMCH('OTQINQIR')
- CBLEEMCH('OTQSKIP ')
- CBLEEMEM('MDPSTART')
- CBLEERPC('CALL ')
- CBLEERPC('CMTSND ')
- CBLEERPC('DISCARD F')
- CBLEERPC('DISCARDS')
- CBLEERPC('GETERDES')
- CBLEERPC('GETWATCH')
- CBLEERPC('POLLANYR')
- CBLEERPC('R-SEND ')
- CBLEERPC('R-SPEND ')
- CBLEERPC('SETWATCH')
- CBLEESCD('CLRRBCNT')
- CBLEESCD('MSGRECV ')
- CBLEESCD('MSGRLBK ')
- CBLEESCD('MSGSKIP ')
- CBLEETHD('ABDCTL ')
- CBLEETHD('CNTGET ')
- CBLEETHD('PRTEND ')
- CBLEETHD('PRTSTART')
- CBLEETHD('UIFAGET ')
- CBLEETIM('CANCEL ')
- CBLEETIM('CANCELAL')
- CBLEETIM('EXECAP ')
- CBLEETRN('C-COMMIT')
- CBLEETRN('C-ROLL ')
- CBLEETRN('EXPTIME ')
- CBLEETRN('GETINFO ')
- CBLEETRN('RMCLSMK ')
- CBLEETRN('ROLLMARK')

DB キュー機能で提供している関数の一覧と使用できる機能について、説明を追加した。

追加・変更内容
他サーバ読み出しの DB キューの読み出し状態を変更できるようにした。これに伴い、次の API 関数を変更した。 • ee_dbq_acsctl • CBLEEDBQ('ACSCTL ')
DB キューでのユーザキューアクセス機能と通番の管理機能のサポートに伴い、次の API 関数を変更した。 • ee_dbq_conctl • ee_dbq_inquire • ee_dbq_msgrefer • ee_dbq_skip • CBLEEDBQ('CONCTL ') • CBLEEDBQ('INQUIRE ') • CBLEEDBQ('MSGREFER') • CBLEEDBQ('SKIP ')
UAP 実行前に、COBOL 実行空間を毎回再生成するかどうかを指定できるようにした。 これに伴い、サービスプログラムの処理での注意を追加した。
エラーコードを追加した。
各バージョンの変更内容を追加した。

付録 B.7 変更内容 (3000-3-F52-10)

次の製品の変更内容 (3000-3-F52-10) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-70
- uCosminexus TP1/Server Base Enterprise Option(64) 07-70
- uCosminexus TP1/EE/Extended Transaction Controller 01-02

追加・変更内容
リードオンリー最適化しない関数の説明を追加した。
排他制御モードに、PR モードを追加した。
プロセス管理テーブルユーザエリア、およびスレッド対応インタフェースユーザエリアのサイズを拡張した。 これに伴い、トランザクションインタフェース情報に次の情報を追加した。 • 拡張ユーザ IFA エリアサイズ • 拡張ユーザ SAT エリアサイズ
トランザクションの同期点を取得する処理でエラーが発生した場合の、注意事項を追加した。
CL 構成の場合に、メッセージを永続化できるようにした。 これに伴い、次の API 関数のリターン値を追加した。 • ee_rpc_reply_suspend • CBLEERPC('R-SPEND ')

追加・変更内容
RPC メッセージ、およびタイマメッセージを、次の API 関数で受信できるようにした。 • ee_scd_msg_receive • CBLEESCD('MSGRECV ')

付録 B.8 変更内容 (3000-3-F52)

次の製品の変更内容 (3000-3-F52) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-60
- uCosminexus TP1/EE/Extended Transaction Controller 01-01

追加・変更内容
1 相最適化がされない条件を変更した。
XTC を使用する際、大量な処理を一時的に扱うためのメモリ領域として、次の領域を使用できるようにした。 • 大量処理用システム領域 (MPSPPOOL) • 大量処理用ユーザ領域 (MPUPPOOL) これに伴い、次の API 関数を追加した。 • ee_mem_mdpstart • CBLEEMEM('MDPSTART')
XTC を使用する際、ロールバックリトライ回数を API 関数でクリアできるようにした。 これに伴い、次の API 関数を追加した。 • ee_scd_clear_rollback_cnt • CBLEESCD('CLRRBCNT')
XTC を使用する際、タイマトランザクションを即時起動する場合などに、タイマ登録を行ったトランザクションが終了してからトランザクションを起動できるようにした。 これに伴い、次の API 関数の説明を変更した。 • ee_tim_execap • CBLEETIM('EXECAP ')

次の製品の変更内容 (3000-3-F52) を表に示します。

- uCosminexus TP1/Server Base Enterprise Option 07-50
- uCosminexus TP1/EE/Extended Transaction Controller 01-00

追加・変更内容
TP1/EE のライブラリ関数を使ってできる機能として、次の二つを補足した。 • タイマトランザクション • DB キューによるメッセージ制御
XTC のサポートに伴い、TP1/EE 間でのシステム間通信を UDP プロトコルによって行う高速メッセージ送信制御をできるようにした。

追加・変更内容

これに伴い、次の API 関数を追加した。

- ee_mch_cmtsend
- ee_mch_cmtsend_get
- ee_mch_cmtsend_sync
- ee_mch_otqbkctl
- ee_mch_otqinquire
- ee_mch_otqskip
- CBLEEMCH('CMTSND ')
- CBLEEMCH('CMTSNDGT')
- CBLEEMCH('CMTSND SY')
- CBLEEMCH('OTQBKCTL')
- CBLEEMCH('OTQINQIR')
- CBLEEMCH('OTQSKIP ')

次の API 関数の説明を変更した。

- ee_rpc_call
- CBLEERPC('CALL ')

次のコマンドを追加した。

- eemchotqdct
- eemchotqact
- eemchotqskip
- eemchotqls
- eemchotqend

オンライン終了時にキューにメッセージが滞留する場合、キューダンプ機能によって、滞留メッセージのキューダンプファイルへの出力および編集が行えるようにした。

これに伴い、次の API 関数を追加した。

- ee_scd_msg_receive
- ee_scd_msg_rollback
- ee_scd_msg_skip
- CBLEESCD('MSGRECV ')
- CBLEESCD('MSGRLBK ')
- CBLEESCD('MSGSKIP ')

また、eepcerefer コマンドを追加した。

TP1/EE がクライアント UAP (SPP) の場合に、非同期応答型 RPC を使用できるようにした。

これに伴い、次の API 関数を追加した。

- ee_rpc_discard_further_replies
- ee_rpc_discard_specific_reply
- ee_rpc_get_error_descriptor
- ee_rpc_poll_any_replies
- CBLEERPC('DISCARD F')
- CBLEERPC('DISCARDS')
- CBLEERPC('GETERDES')
- CBLEERPC('POLLANYR')

追加・変更内容
TP1/EE のオプション製品として、次の製品をサポートした。 • XDB • XTC • MCP これに伴い、次の API 関数の説明を変更した。 • ee_rpc_call • ee_rpc_cmtsend • ee_rpc_reply_suspend • ee_thd_abdctl • ee_tim_execap • CBLEERPC('CALL ') • CBLEERPC('CMTSND ') • CBLEERPC('R-SPEND ') • CBLEETHD('ABDCTL ') • CBLEETIM('EXECAP ')
TP1/EE で対応する処理を決定できない障害が発生した場合に、ユーザに応答を促すメッセージが出力されるようにした。 これに伴い、eetrbwtor コマンドを追加した。
UAP で使用する資源の排他制御をできるようにした。 これに伴い、次の API 関数を追加した。 • ee_lck_get • ee_lck_release_all • ee_lck_release_byname • CBLEELCK('GET ') • CBLEELCK('RELALL ') • CBLEELCK('RELNAME ') また、eelckls コマンドを追加した。
XTC のサポートに伴い、すべての API のリターン値に EECOMER_CNDUOC を追加した（COBOL の場合、すべての API のステータスコードに 00004 を追加した）。
ERRTRN4 の起動要因が、サービス閉塞によるものかオンライン終了によるものかを UAP で判断できるようにした。
トランザクションインタフェース情報および拡張トランザクションインタフェース情報に変数名を補足した。

付録 B.9 変更内容（3000-3-983-40）

uCosminexus TP1/Server Base Enterprise Option 07-01 の製品の変更内容（3000-3-983-40）を表に示します。

追加・変更内容
Oracle をリソースマネージャとして使用できるようにした。また、リソースマネージャとして Oracle を使用する場合に SQL トレースを取得できるようにした。

追加・変更内容
これに伴い、eedbqtblo コマンドを追加した。
リソースマネージャとのコネクションを解放することで、Oracle の暗黙カーソルをクローズできるようにした。 これに伴い、次の API 関数を追加した。
• ee_trn_rm_close_mark • CBLEETRN('RMCLSMK ')

付録 B.10 変更内容 (3000-3-983-30)

uCosminexus TP1/Server Base Enterprise Option 07-00 の製品の変更内容 (3000-3-983-30) を表に示します。

追加・変更内容
トラブルシュート機能で次のことができるようにした。 • UAP トレース情報の編集出力、および強制出力 • TASKTM 情報の編集出力 (コアファイルまたはメモリダンプファイル入力) • 回線トレース情報の編集出力 (コアファイルまたはメモリダンプファイル入力) これに伴い、次のコマンドを追加した。 • eetrbtaskdump • eetrblinedump • eetrbuatfput • eetrbuated
適用 OS に Red Hat Enterprise Linux AS 4 (IPF)を追加した。
TP1/EE をリモート API 機能の rap サーバとして動作できるようにした。
RPC メッセージの最大長拡張機能を追加した。 これに伴い、API 関数の引数を拡張した。

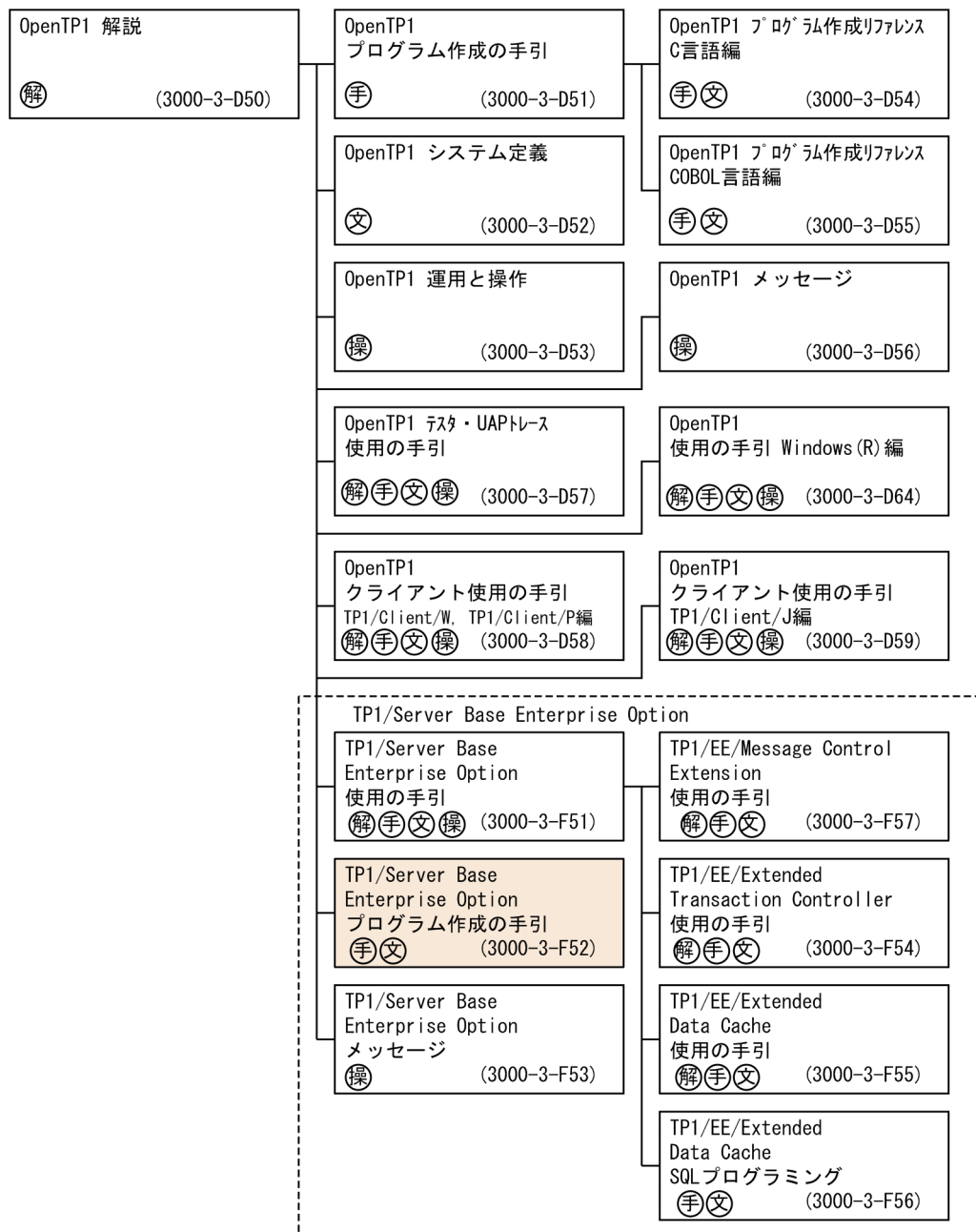
付録 C このマニュアルの参考情報

このマニュアルの関連マニュアルを次に示します。必要に応じてお読みください。

付録 C.1 関連マニュアル

このマニュアルの関連マニュアルを次に示します。必要に応じてお読みください。

- OpenTP1 Version 7



<記号>

- 解 : 解説書
- 手 : 手引書
- 文 : 文法書
- 操 : 操作書

・ 関連製品

- ・ COBOL85 使用の手引 (3000-3-354)
- ・ COBOL2002 使用の手引 手引編 (3000-3-D42)
- ・ COBOL2002 使用の手引 操作編 (3000-3-D43)
- ・ HiRDB Version 9 構造型データベース機能 (3000-6-361)
- ・ HiRDB Version 9 解説 (3020-6-450)

- HiRDB Version 9 システム導入・設計ガイド (UNIX(R)用) (3000-6-452)
- HiRDB Version 9 システム定義 (UNIX(R)用) (3000-6-453)
- HiRDB Version 9 システム運用ガイド (UNIX(R)用) (3000-6-454)
- HiRDB Version 9 コマンドリファレンス (UNIX(R)用) (3000-6-455)
- HiRDB Version 9 UAP 開発ガイド (3020-6-456)
- HiRDB Version 9 SQL リファレンス (3020-6-457)
- HiRDB Version 9 メッセージ (3020-6-458)
- HiRDB Version 9 XDM/RD E2 接続機能 (3020-6-465)
- HiRDB Version 10 解説 (3020-6-551)
- HiRDB Version 10 システム導入・設計ガイド (UNIX(R)用) (3020-6-552)
- HiRDB Version 10 システム定義 (UNIX(R)用) (3020-6-554)
- HiRDB Version 10 システム運用ガイド (UNIX(R)用) (3020-6-556)
- HiRDB Version 10 コマンドリファレンス (UNIX(R)用) (3020-6-558)
- HiRDB Version 10 UAP 開発ガイド (3020-6-560)
- HiRDB Version 10 SQL リファレンス (3020-6-561)
- HiRDB Version 10 メッセージ (3020-6-562)
- HiRDB Version 10 XDM/RD E2 接続機能 (3020-6-565)
- HiRDB Version 10 構造型データベース機能 (3020-6-578)
- 高信頼化システム監視機能 HA モニタ Linux(R)編 (3000-9-132)
- 高信頼化システム監視機能 HA モニタ Linux(R) (x86) 編 (3000-9-201)
- 高信頼化システム監視機能 HA モニタ メッセージ (3000-9-203)
- Cosminexus Reliable Messaging (3020-3-M17)
- Cosminexus アプリケーションサーバ V8 Cosminexus Reliable Messaging (3020-3-U21)

付録 C.2 このマニュアルでの表記

(1) 製品名

このマニュアルで使用する製品名称の略称を次に示します。

略称		名称
Linux	Linux (EM64T)	Red Hat Enterprise Linux Server 6 (64-bit x86_64)
		Red Hat Enterprise Linux Server 7 (64-bit x86_64)

略称		名称
Linux	Linux (EM64T)	Red Hat Enterprise Linux Server 8 (64-bit x86_64)
		Red Hat Enterprise Linux Server 9 (64-bit x86_64)
	Linux (IA32)	Red Hat Enterprise Linux Server 6 (64-bit x86_64)
MCP		uCosminexus TP1/EE/Message Control Extension
Oracle	Oracle 10g R2	Oracle Database 10g Release 2
	Oracle 11g R1	Oracle Database 11g Release 1
TP1/Client	TP1/Client/J	uCosminexus TP1/Client/J
	TP1/Client/P	uCosminexus TP1/Client/P
	TP1/Client/W	uCosminexus TP1/Client/W
TP1/EE	TP1/Server Base Enterprise Option	uCosminexus TP1/Server Base Enterprise Option
		uCosminexus TP1/Server Base Enterprise Option(64)
TP1/FSP		uCosminexus TP1/Financial Service Platform
TP1/Server Base		uCosminexus TP1/Server Base
UNIX		UNIX(R)
XDB		uCosminexus TP1/EE/Extended Data Cache
XTC		uCosminexus TP1/EE/Extended Transaction Controller

(2) アーキテクチャによる違い

TP1/EE マニュアルでは、32 ビットアーキテクチャ対応 OS と 64 ビットアーキテクチャ対応 OS で記述を書き分けている個所があります。ご使用の OS をご確認の上、アーキテクチャに応じた記載箇所をお読みください。

このマニュアルでのアーキテクチャの違いによる表記と対応 OS を次の表に示します。

マニュアルの表記	OS
32 ビットアーキテクチャの場合	Red Hat Enterprise Linux Server 6 (64-bit x86_64)
64 ビットアーキテクチャの場合	Red Hat Enterprise Linux Server 6 (64-bit x86_64), Red Hat Enterprise Linux Server 7 (64-bit x86_64), Red Hat Enterprise Linux Server 8 (64-bit x86_64), Red Hat Enterprise Linux Server 9 (64-bit x86_64)

(3) インストールディレクトリのパス

このマニュアルでは、TP1/EE のインストールディレクトリを次のように表記しています。

このマニュアルでの表記	読み替え
インストールディレクトリ	/opt/OpenTP1

付録 C.3 英略語

このマニュアルで使用する英略語の一覧を次に示します。

英略語	英字での表記
AP	Application Program
API	Application Programming Interface
C/S	Client/Server
CPU	Central Processing Unit
CR	Carriage Return
CSV	Comma Separated Value
CUP	Client User Program
DAM	Direct Access Method
DB	Database
DBMS	Database Management System
DML	Data Manipulation Language
DNS	Domain Name System
DTP	Distributed Transaction Processing
EOF	End of File
FIFO	First-In-First-Out
GUI	Graphical User Interface
HA	High Availability
I/O	Input/Output
ID	Identifier
IP	Internet Protocol
J2EE	Java 2 Platform Enterprise Edition
LAN	Local Area Network
LF	Line Feed
MCF	Message Control Facility

英略語	英字での表記
MTU	Maximum Transmission Unit
NFS	Network File System
OLTP	Online Transaction Processing
OS	Operating System
OSI	Open Systems Interconnection
OSI TP	Open Systems Interconnection Transaction Processing
PC	Personal Computer
RAC	Real Application Clusters
RPC	Remote Procedure Call
SDB	Structured Database
SPP	Service Providing Program
SUP	Service Using Program
TAM	Table Access Method
TCP/IP	Transmission Control Protocol/Internet Protocol
TP	Transaction Processing
UAP	User Application Program
UDP	User Datagram Protocol
UOC	User Own Coding
WAN	Wide Area Network
WS	Workstation
XA	Extended Architecture

TP1/EE マニュアルで使用する略称の一覧を次に示します。

略称	名称
AP	アプリケーションプログラム
API	アプリケーションプログラミングインタフェース
CUP	クライアントユーザプログラム
DB	データベース
IFA	スレッド対応エリア
IMA	入力メッセージ引き渡しエリア
OMA	出力メッセージ引き渡しエリア

略称	名称
RPC	リモートプロシジャコール
SPP	サービス提供プログラム
SUP	サービス利用プログラム
UAP	ユーザアプリケーションプログラム
WS	ワークステーション
XID	トランザクション識別子

付録 C.4 KB（キロバイト）などの単位表記について

1KB（キロバイト）、1MB（メガバイト）、1GB（ギガバイト）、1TB（テラバイト）はそれぞれ 1,024 バイト、 $1,024^2$ バイト、 $1,024^3$ バイト、 $1,024^4$ バイトです。

索引

数字

- 1 相最適化 67
- 2 相コミット 62
- 2 相コミットの流れ 63

A

- ANSI C [C++言語] 131
- ANSI C [C 言語] 131
- API 関数 131
- API の代理実行 117

C

- C++言語 131
- C++言語の仕様 131
- CBLEEDADM('COMMAND ') 415
- CBLEEDBQ('ACSCNTL ') 421
- CBLEEDBQ('CLEAR ') 425
- CBLEEDBQ('CONCTL ') 429
- CBLEEDBQ('INQSRV ') 432
- CBLEEDBQ('INQUIRE2') 439
- CBLEEDBQ('INQUIRE ') 435
- CBLEEDBQ('MSGGET ') 445
- CBLEEDBQ('MSGPUT ') 449
- CBLEEDBQ('MSGREFE2') 452
- CBLEEDBQ('MSGREFER') 455
- CBLEEDBQ('OBSCAN ') 458
- CBLEEDBQ('OBSCDT ') 460
- CBLEEDBQ('OBSEND ') 463
- CBLEEDBQ('OBSINQ ') 465
- CBLEEDBQ('OBSRBSP ') 469
- CBLEEDBQ('OBSRST ') 471
- CBLEEDBQ('OBSSKIP ') 474
- CBLEEDBQ('OBSSTART') 477
- CBLEEDBQ('OBSSTOP ') 480
- CBLEEDBQ('RBSKIP ') 482
- CBLEEDBQ('SKIP ') 484
- CBLEEDBQ('SRVCTL ') 487

- CBLEELCK('GET ') 490
- CBLEELCK('RELALL ') 493
- CBLEELCK('RELNAME ') 495
- CBLEELOG('PRINT ') 498
- CBLEEMCH('CMTSNDGT') 508
- CBLEEMCH('CMTSNDST') 512
- CBLEEMCH('CMTSND ') 504
- CBLEEMCH('OTQBKCTL') 516
- CBLEEMCH('OTQINQIR') 519
- CBLEEMCH('OTQSKIP ') 522
- CBLEEMEM('GETWK ') 526
- CBLEEMEM('MDPSTART') 528
- CBLEEMEM('RLSWK ') 530
- CBLEERPC('CALLTO ') 541
- CBLEERPC('CALL ') 533
- CBLEERPC('CMTSNDTO') 553
- CBLEERPC('CMTSND ') 549
- CBLEERPC('DISCARD') 557
- CBLEERPC('DISCARDS') 559
- CBLEERPC('GETERDES') 561
- CBLEERPC('GETWATCH') 563
- CBLEERPC('POLLANYR') 565
- CBLEERPC('R-SEND ') 570
- CBLEERPC('R-SPEND ') 573
- CBLEERPC('SETWATCH') 576
- CBLEESCD('CLRRBCNT') 579
- CBLEESCD('MSGRECV ') 581
- CBLEESCD('MSGRLBK ') 584
- CBLEESCD('MSGSKIP ') 586
- CBLEETHD('ABDCTL ') 589
- CBLEETHD('CNTGET ') 591
- CBLEETHD('GETTIME ') 593
- CBLEETHD('PRTEND ') 599
- CBLEETHD('PRTSTART') 601
- CBLEETHD('UIFAGET ') 603
- CBLEETIM('CANCELAL') 609
- CBLEETIM('CANCEL ') 606

- CBLEETIM('EXECAP ') 611
- CBLEETRB('UINFOTM ') 617
- CBLEETRB('UTPUT ') 619
- CBLEETRNB('C-COMMIT') 622
- CBLEETRNB('C-ROLL ') 625
- CBLEETRNB('EXPTIME ') 629
- CBLEETRNB('GETINFO ') 631
- CBLEETRNB('RMCLSMK ') 635
- CBLEETRNB('ROLLMARK') 637
- COBOL2002 132
- COBOL85 132
- COBOL 言語 131
- COBOL 言語でコーディングする場合 [サービス関数] 132
- COBOL 言語のコーディング例 652
- COBOL-UAP 作成用プログラム 132
- COBOL-UAP 作成用プログラムの説明形式 381
- C 言語 131
- C 言語, COBOL 言語の API 関数の対応 136
- C 言語のコーディング例 645
- C 言語または C++言語でコーディングする場合 [サービス関数] 131

D

- DB キュー書き込み抑止機能 87
- DB キューサービスの接続照会 [COBOL 言語] 432
- DB キューサービスの接続照会 [C 言語] 195
- DB キューサービスのトランザクションのコミット決着 82
- DB キューサービスの閉塞制御 [COBOL 言語] 487
- DB キューサービスの閉塞制御 [C 言語] 246
- DB キュー自動読み出し停止機能 88
- DB キュー情報の照会 [COBOL 言語]
[CBLEEDBQ('INQUIRE2')] 439
- DB キュー情報の照会 [COBOL 言語]
[CBLEEDBQ('INQUIRE ')] 435
- DB キュー情報の照会 [C 言語] [ee_dbq_inquire] 198
- DB キュー情報の照会 [C 言語] [ee_dbq_inquire2] 202

- DB キュースケジュール状態一括変更機能 89
- DB キュー制御 (CBLEEDBQ) 419
- DB キュー制御 (ee_dbq_~) 183
- DB キューとシステムとの関連 82
- DB キューによるメッセージ制御 79
- DB キューの接続制御 [COBOL 言語] 429
- DB キューの接続制御 [C 言語] 192
- DB キューの通番初期化 [COBOL 言語] 425
- DB キューの通番初期化 [C 言語] 189
- DB キューの通番の管理 88
- DB キューの読み出し制御 [COBOL 言語] 421
- DB キューの読み出し制御 [C 言語] 185
- DB キューメッセージスキップ機能 88
- DB キュー読み出しサーバ変更機能 87
- DB キューを使用した AP 間通信 79
- DB キューを使用したシステム間通信 82

E

- ee_adm_call_command 179
- ee_dbq_acsctl 185
- ee_dbq_clear 189
- ee_dbq_conctl 192
- ee_dbq_inqsrv 195
- ee_dbq_inquire 198
- ee_dbq_inquire2 202
- ee_dbq_msgget 208
- ee_dbq_msgput 212
- ee_dbq_msgrefer 215
- ee_dbq_msgrefer2 218
- ee_dbq_obscancel 221
- ee_dbq_obschdata 223
- ee_dbq_obsend 225
- ee_dbq_obsinquire 227
- ee_dbq_obsrbskip 230
- ee_dbq_obsrestart 232
- ee_dbq_obsrskip 234
- ee_dbq_obsstart 236
- ee_dbq_obsstop 239
- ee_dbq_rbskip 241

ee_dbq_skip 243
ee_dbq_srvctl 246
ee_lck_get 249
ee_lck_release_all 251
ee_lck_release_byname 252
ee_logprint 255
ee_mch_cmtsend 260
ee_mch_cmtsend_get 264
ee_mch_cmtsend_sync 268
ee_mch_otqbctl 271
ee_mch_otqinquire 273
ee_mch_otqskip 276
ee_mem_getwkseg 279
ee_mem_mdpstart 281
ee_mem_rlswkseg 283
ee_rpc_call 285
ee_rpc_call_to 293
ee_rpc_cmtsend 301
ee_rpc_cmtsend_to 305
ee_rpc_discard_further_replies 308
ee_rpc_discard_specific_reply 310
ee_rpc_get_error_descriptor 312
ee_rpc_get_watch_time 313
ee_rpc_poll_any_replies 314
ee_rpc_reply_send 319
ee_rpc_reply_suspend 322
ee_rpc_set_watch_time 324
ee_scd_clear_rollback_cnt 326
ee_scd_msg_receive 328
ee_scd_msg_rollback 331
ee_scd_msg_skip 333
ee_thd_abdctl 336
ee_thd_gettime 338
ee_thd_protend 344
ee_thd_protstart 345
ee_thd_thdcntget 346
ee_thd_uifaget 348
ee_tim_execap 351
ee_tim_execap_cancel 356

ee_tim_execap_cancel_all 358
ee_trb_uinfo_tm 361
ee_trb_ustrace_put 363
ee_trn_chained_commit 366
ee_trn_chained_rollback 369
ee_trn_getinfo 372
ee_trn_rm_close_mark 375
ee_trn_rollback_mark 376
ee_trn_set_exp_time 378
ERRTRN1 107
ERRTRN2 108
ERRTRN3 109
ERRTRN4 110
ERRTRNR 111
ERRTRNS 112
EX モード 124

P

PC 18
PR モード 124

R

RPC 19
RPC 応答メッセージ送信の抑止〔COBOL 言語〕 573
RPC 応答メッセージ送信の抑止〔C 言語〕 322
RPC の形態 29
RPC のデータの受け渡し 29
RPC のネスト例 49
RPC を使った通信のクライアント／サーバの関係 28

S

SPP (TP1/EE) の環境設定 134
SPP (TP1/EE) の作成 129
SPP (TP1/EE) の作成手順 129
SPP (TP1/EE) の作成の概要 24
SPP (TP1/EE) の作成の詳細 128
SPP (TP1/EE) の実行 135
SPP の概要 23
SPP の作成手順 130

SPP の名称 134

SQL 文 22

T

TCP/IP 19

TCP/IP 通信でイベントを受信したときの処理の流れ 85

TCP/IP 通信でイベントを送信したときの処理の流れ 86

TP1/EE および XTC の API 136

TP1/EE および XTC のアプリケーションプログラムと API 136

TP1/EE および XTC のライブラリ関数の文法 (COBOL 言語編) 380

TP1/EE および XTC のライブラリ関数の文法 (C 言語編) 145

TP1/EE と UAP のネットワーク内の位置 18

TP1/EE として定められているコーディング規約 133

TP1/EE の API 関数 131

TP1/EE のアプリケーションプログラム 17

TP1/EE の基本機能 26

TP1/EE への登録 134

TP1/Server Base と RPC 通信をする場合の設定 54

U

UAP 18

UAP 異常終了通知 (ERRTRN3) 109

UAP 共用ライブラリの作成 133

UAP 実行形式ファイルの作成 133

UAP または UOC の異常で発生する同期シグナル 122

UOC インタフェース 640

UOC で使用できる関数 641

W

WS 18

X

XA インタフェース 66

あ

アプリケーションプログラミングインタフェースの機能 25

い

一方送信メッセージ情報の取得 [COBOL 言語] 508

一方送信メッセージ情報の取得 [C 言語] 264

一方送信メッセージの送信スキップ [COBOL 言語] 522

一方送信メッセージの送信スキップ [C 言語] 276

イベント型オンラインバッチ処理 92

う

運用コマンド実行機能 95

運用コマンド実行機能使用時の定義 102

運用コマンド実行機能での TP1/EE コマンドの自 TP1/EE プロセスに対する実行可否 95

運用コマンド実行機能での XTC コマンドの自 TP1/EE プロセスに対する実行可否 98

運用コマンド実行機能の環境設定 99

運用コマンド実行機能の時間監視 100

運用コマンド実行機能の時間監視によるサービス要求の応答待ち時間の変更 101

運用コマンド実行機能の処理 99

運用コマンド実行機能の処理概要 100

運用コマンド実行サーバ 101

運用コマンド実行サーバの TP1/Server Base のユーザサービス定義 102

運用コマンドの実行 [COBOL 言語] 415

運用コマンドの実行 [C 言語] 179

え

エラーが発生した非同期応答型 RPC の識別子取得 [COBOL 言語] 561

エラーが発生した非同期応答型 RPC の識別子取得 [C 言語] 312

エラートランザクション 107

エラートランザクションの登録に失敗した場合の後処理 113

遠隔サービスの要求 [COBOL 言語] 533

お

- オートコネクトモード 114
- オンライン開始 UOC インタフェース 642
- オンライントランザクション処理 18
- オンラインバッチ機能 90
- オンラインバッチ機能で実行できる処理 93
- オンラインバッチ機能の概要 90
- オンラインバッチ機能の処理形態 91
- オンラインバッチ処理の開始 [COBOL 言語] 477
- オンラインバッチ処理の開始 [C 言語] 236
- オンラインバッチ処理の強制終了 [COBOL 言語] 458
- オンラインバッチ処理の強制終了 [C 言語] 221
- オンラインバッチ処理の再開 [COBOL 言語] 471
- オンラインバッチ処理の再開 [C 言語] 232
- オンラインバッチ処理の終了 [COBOL 言語] 463
- オンラインバッチ処理の終了 [C 言語] 225
- オンラインバッチ処理の中断 [COBOL 言語] 480
- オンラインバッチ処理の中断 [C 言語] 239

か

- 各送信形態の概要 39
- 格納するディレクトリ 134
- 関数の説明形式 146

く

- クライアント／サーバ形態の UAP 18
- クライアント／サーバ形態の UAP の概要 19
- クライアント／サーバ形態の通信で使用する UAP の役割と位置の概要 22
- クライアント／サーバ形態の通信のトランザクション 55
- クライアント／サーバ形態の通信プロトコル 19
- クライアント／サーバ形態のユーザアプリケーションプログラム 18
- クライアント UAP 19
- グローバルトランザクション 55

け

- 現在時刻の取得 [COBOL 言語] 593
- 現在時刻の取得 [C 言語] 338
- 現在のトランザクション識別情報の取得 [COBOL 言語] 631
- 現在のトランザクション識別情報の取得 [C 言語] 372

こ

- 更新目的の排他 124
- 高速メッセージ送信 (CBLEEMCH) 503
- 高速メッセージ送信 (ee_mch_~) 259
- コーディング規約 132
- コーディング例 644
- コミット 20
- コミット決着失敗通知 (ERRTRNR) 111
- コミット処理 62
- コミット送信 41
- コミット [同期点の取得] 56

さ

- サーバ 18
- サーバ UAP 19
- サーバ UAP (SPP) としての TP1/EE 21
- サーバ側のトランザクションブランチがある場合の 1 相最適化の概要 69
- サービス 18
- サービス関数に値が渡される引数 154
- サービス関数のコーディング 130
- サービス関数の作成 (SPP) 148
- サービスの応答待ち時間の更新 [COBOL 言語] 576
- サービスの応答待ち時間の更新 [C 言語] 324
- サービスの応答待ち時間の参照 [COBOL 言語] 563
- サービスの応答待ち時間の参照 [C 言語] 313
- サービスのネスト 49
- サービスプログラムに値が渡されるデータ領域 389
- サービスプログラムの作成 (SPP) 383
- サービス閉塞通知 (ERRTRN2) 108
- サービス閉塞の制御 [COBOL 言語] 589

サービス閉塞の制御 [C 言語] 336
サービス名 19
サービス要求の応答待ち時間の参照と更新 49
参照目的の排他 124

し

資源の排他 [COBOL 言語] 490
資源の排他 [C 言語] 249
資源の排他制御 124
資源の排他制御 (CBLEELCK) 489
資源の排他制御 (ee_lck_~) 248
資源名称を指定した排他の解除 [COBOL 言語] 495
資源名称を指定した排他の解除 [C 言語] 252
時刻または経過時間の指定と TP1/EE サービス定義との関係 75
時刻または経過時間の指定方法 73
システム運用の管理 (CBLEEADM) 414
システム運用の管理 (ee_adm_~) 178
システム終了時のタイマランザクションの扱い 75
自プロセス送信 41
受信メッセージの差し戻し [COBOL 言語] 584
受信メッセージの差し戻し [C 言語] 331
出力キュー (OTQ) の情報の照会 [COBOL 言語] 519
出力キュー (OTQ) の情報の照会 [C 言語] 273
出力キュー (OTQ) の閉塞制御 [COBOL 言語] 516
出力キュー (OTQ) の閉塞制御 [C 言語] 271
常設コネクション 114
常設コネクションの解放 119
常設コネクションの確立 116
処理スレッド数の取得 [COBOL 言語] 591
処理スレッド数の取得 [C 言語] 346

す

スケジュール方式 81
ステータスコード 132
すべての非同期応答型 RPC の応答受信拒否 [COBOL 言語] 557
すべての非同期応答型 RPC の応答受信拒否 [C 言語] 308

スレッド制御 122
スレッド制御 (CBLEETHD) 588
スレッド制御 (ee_thd_~) 335

せ

先行許容時間／後退許容時間と指定時刻の関係 75
全資源の排他の解除 [COBOL 言語] 493
全資源の排他の解除 [C 言語] 251
全タイマランザクションの起動取り消し [COBOL 言語] 609
全タイマランザクションの起動取り消し [C 言語] 358

そ

送信障害通知 (ERRTRNS) 112
即時送信の流れ (同期応答型 RPC) 32
即時送信の流れ (非応答型 RPC) 40
即時送信の流れ (非同期応答型 RPC) 36
即時登録方式 72

た

第 1 相 62
第 2 相 62
タイマランザクション 72
タイマランザクション (CBLEETIM) 605
タイマランザクション (ee_tim_~) 350
タイマランザクション起動破棄通知 (ERRTRN4) 110
タイマランザクションの起動 [COBOL 言語] 611
タイマランザクションの起動 [C 言語] 351
タイマランザクションの起動取り消し [COBOL 言語] 606
タイマランザクションの起動取り消し [C 言語] 356
タイマランザクションの登録 72
タイマランザクションの取り消し 74
滞留メッセージ受信 (CBLEESCD) 578
滞留メッセージ受信 (ee_scd_~) 325
滞留メッセージの受信 [COBOL 言語] 581
滞留メッセージの受信 [C 言語] 328

滞留メッセージのスキップ [COBOL 言語] 586
滞留メッセージのスキップ [C 言語] 333
大量処理用メモリ管理機能の使用開始 [COBOL 言語] 528
大量処理用メモリ管理機能の使用開始 [C 言語] 281
単接続と複接続の接続形態、サービス関数起動の概念 79

つ

通常のトランザクション処理 (2 相コミット) の概要 67
通信先を指定した遠隔サービスの要求 [COBOL 言語] 541
通信先を指定した遠隔サービスの要求 [C 言語] 293
通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求 [COBOL 言語] 553
通信先を指定したトランザクションと同期して送信する非応答型 RPC 要求 [C 言語] 305
通信先を指定したリモートプロシジャコール 42

て

データ型オンラインバッチ処理 91
データベース言語 (SQL) 131

と

同期応答型 RPC 30
同期応答型 RPC でメッセージを受信したときの処理の流れ 31
同期応答型 RPC と同期点の関係 59
同期点 20
同期点の取得 56
同期点を取得する関数 56
特定の非同期応答型 RPC の応答受信拒否 [COBOL 言語] 559
特定の非同期応答型 RPC の応答受信拒否 [C 言語] 310
トラブルシュート (CBLEETRB) 616
トラブルシュート (ee_trb_~) 360
トランザクショナル RPC 50
トランザクショナル RPC および非トランザクショナル RPC の流れ 51

トランザクション 20
トランザクションインタフェース情報 [COBOL 言語] 393
トランザクションインタフェース情報 [C 言語] 157
トランザクション決着結果 56
トランザクション種別 677
トランザクション処理の監視時間の変更 [COBOL 言語] 629
トランザクション処理の監視時間の変更 [C 言語] 378
トランザクション制御 55
トランザクション制御 (CBLEETRN) 621
トランザクション制御 (ee_trn_~) 365
トランザクションで利用できる API 関数 139
トランザクション同期登録方式 73
トランザクション同期の一方送信メッセージの送信 [COBOL 言語] 512
トランザクション同期の一方送信メッセージの送信 [C 言語] 268
トランザクションと同期して送信する非応答型 RPC 要求 [COBOL 言語] 549
トランザクションと同期して送信する非応答型 RPC 要求 [C 言語] 301
トランザクションの開始 55
トランザクションの開始および終了 55
トランザクションの開始と終了 55
トランザクションの回復 55
トランザクションの最適化 65
トランザクションの終了 56
トランザクションの縮退 55
トランザクションの処理から非トランザクショナル RPC の発行 51
トランザクションのロールバック指示 [COBOL 言語] 637
トランザクションのロールバック指示 [C 言語] 376
トランザクション非同期の一方送信メッセージの送信 [COBOL 言語] 504
トランザクション非同期の一方送信メッセージの送信 [C 言語] 260

ね

ネットワーク内の位置 18

の

ノード 19

は

パーソナルコンピュータ 18
排他制御モード 124
排他の解除方法 125
排他のテスト 127
排他待ち限界経過時間の指定 125
バッチ終了トランザクション 90
バッチ処理トランザクション 90

ひ

非応答型 RPC 36, 37
非応答型 RPC でメッセージを受信したときの処理の流れ 38
非応答型 RPC と同期点の関係 61
引き継ぎ情報の更新 [COBOL 言語] 460
引き継ぎ情報の更新 [C 言語] 223
非同期応答型 RPC 33
非同期応答型 RPC でメッセージを受信したときの処理の流れ 34
非同期応答型 RPC と同期点の関係 60
非同期応答型 RPC の応答受信 [COBOL 言語] 565
非同期応答型 RPC の応答受信 [C 言語] 314
非保護区を設定した場合の制御 123
ヒューリスティック決着 64
ヒューリスティック決着の流れ 65
ヒューリスティックコミット 64
ヒューリスティックハザード 64
ヒューリスティックミックス 64
ヒューリスティックロールバック 64

ふ

不正サービス名称検出通知 (ERRTRN1) 107
フラグメントメッセージの送信 29

プリペア処理 62

へ

別トランザクションによる RPC 同期応答 52
別トランザクションによる RPC 同期応答処理の流れ 53

ほ

保護区／非保護区に設定する処理と異常発生時の処理 123
保護区／非保護区の制御 122
保護区の開始 [COBOL 言語] 601
保護区の開始 [C 言語] 345
保護区の終了 [COBOL 言語] 599
保護区の終了 [C 言語] 344

め

メインフレーム 18
メッセージ受信処理を実行できない場合 121
メッセージの書き込み [COBOL 言語] 449
メッセージの書き込み [C 言語] 212
メッセージの書き込み処理の流れ 84
メッセージの参照読み出し [COBOL 言語] [CBLEEDBQ('MSGREFE2')] 452
メッセージの参照読み出し [COBOL 言語] [CBLEEDBQ('MSGREFER')] 455
メッセージの参照読み出し [C 言語] [ee_dbq_msgrefer] 215
メッセージの参照読み出し [C 言語] [ee_dbq_msgrefer2] 218
メッセージの読み出し [COBOL 言語] 445
メッセージの読み出し [C 言語] 208
メッセージの読み出し処理の流れ 83
メッセージの読み出しスキップ [COBOL 言語] [CBLEEDBQ('OBSSKIP ')] 474
メッセージの読み出しスキップ [COBOL 言語] [CBLEEDBQ('SKIP ')] 484
メッセージの読み出しスキップ [C 言語] [ee_dbq_obsskip] 234
メッセージの読み出しスキップ [C 言語] [ee_dbq_skip] 243
メッセージログの出力 (CBLEELOG) 497

メッセージログの出力 (ee_log～) 254
メッセージログの出力 [COBOL 言語] 498
メッセージログの出力 [C 言語] 255
メモリ管理 (CBLEEMEM) 525
メモリ管理 (ee_mem_～) 278

ゆ

ユーザ IFA 情報格納領域アドレスの取得 [COBOL 言語] 603
ユーザ IFA 情報格納領域アドレスの取得 [C 言語] 348
ユーザアプリケーションプログラム 18
ユーザアプリケーションプログラムと業務形態の関係 18
ユーザアプリケーションプログラムの種類 21
ユーザアプリケーションプログラムのトランザクション処理 20
ユーザオウンコーディング (UOC) 639
ユーザ固有のトレースの取得 [COBOL 言語] 619
ユーザ固有のトレースの取得 [C 言語] 363
ユーザサーバ 18
ユーザサーバ名 134
ユーザ情報の TASKTM 出力 361, 617

よ

抑止していた RPC 応答メッセージの送信 [COBOL 言語] 570
抑止していた RPC 応答メッセージの送信 [C 言語] 319

ら

ライブラリ関数の文法 [COBOL 言語] 380
ライブラリ関数の文法 [C 言語] 145

り

リードオンリー最適化 69
リードオンリー最適化の概要 70
リソースマネージャとのコネクションの解放 57
リソースマネージャとのコネクションの解放指示 [COBOL 言語] 635

リソースマネージャとのコネクションの解放指示 [C 言語] 375

リターン値 131

リモート API 機能 114

リモート API 機能を使用する場合の、コネクション確立からコネクション切断までの流れ 115

リモートプロシジャコール 19

リモートプロシジャコール (CBLEERPC) 532

リモートプロシジャコール (ee_rpc_～) 284

リモートプロシジャコール [TP1/EE の基本機能] 27

リモートプロシジャコールでのデータの受け渡し 28

リモートプロシジャコールでメッセージを送受信できない場合 47

リモートプロシジャコールと同期点の関係 58

リモートプロシジャコールのエラーリターン値と同期点 62

リモートプロシジャコールの形態 29

リモートプロシジャコールの形態と同期点の関係 58

リモートプロシジャコールの実現方法 27

れ

連鎖モードのコミット [COBOL 言語] 622

連鎖モードのコミット [C 言語] 366

連鎖モードのロールバック [COBOL 言語] 625

連鎖モードのロールバック [C 言語] 369

ろ

ロールバック 20

ロールバック処理 62

ロールバック [同期点の取得] 56

ロールバック要求時のメッセージの読み出しスキップ [COBOL 言語] [CBLEEDBQ('OBSRBSP ')] 469

ロールバック要求時のメッセージの読み出しスキップ [COBOL 言語] [CBLEEDBQ('RBSKIP ')] 482

ロールバック要求時のメッセージの読み出しスキップ [C 言語] [ee_dbq_obsrbskip] 230

ロールバック要求時のメッセージの読み出しスキップ [C 言語] [ee_dbq_rbskip] 241

ロールバックリトライ回数のクリア [COBOL 言語] 579

ロールバックリトライ回数のクリア [C 言語] 326

ロックマイグレーション 125

ロット 90

ロットの情報の照会〔COBOL 言語〕 465

ロットの情報の照会〔C 言語〕 227

わ

ワークステーション 18

ワークセグメントの返還〔COBOL 言語〕 530

ワークセグメントの返還〔C 言語〕 283

ワークセグメントの割り当て〔COBOL 言語〕 526

ワークセグメントの割り当て〔C 言語〕 279