HITACHI
Inspire the Next

OpenTP1 Version 7
分散トランザクション処理機能

OpenTP1 プログラム作成リファレンス COBOL 言語編

手引・文法書

3000-3-D55-60

前書き

■ 対象製品

マニュアル「OpenTP1 解説」を参照してください。

■ 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

HP-UX は，Hewlett-Packard Development Company, L.P.のオペレーティングシステムの名称です。

IBM，AIX は，世界の多くの国で登録された International Business Machines Corporation の商標です。

Motif は，Open Software Foundation, Inc.の商標です。

Oracle と Java は，Oracle Corporation 及びその子会社，関連会社の米国及びその他の国における登録商標です。

OSF は，Open Software Foundation, Inc.の商標です。

UNIX は，The Open Group の米国ならびに他の国における登録商標です。

Windows は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

X/Open は，The Open Group の英国ならびに他の国における登録商標です。

XATMI は，X/Open Company Limited が開発したアプリケーションインタフェースの名称です。

その他記載の会社名，製品名は，それぞれの会社の商標もしくは登録商標です。

プログラムプロダクト「P-9D64-3F31，P-9D64-8531，P-9D64-8931，R-19451-216，R-19452-216，R-19453-216，R-19454-216，R-19455-216，R-19456-216，R-19459-216，R-1945A-216，R-1945C-216，R-1945D-216，R-1945E-216，R-F19456-2156，R-F19456-21C6」には，Oracle Corporation またはその子会社，関連会社が著作権を有している部分が含まれています。

プログラムプロダクト「P-9D64-3F31，P-9D64-8531，P-9D64-8931，R-19451-216，R-19452-216，R-19453-216，R-19454-216，R-19455-216，R-19456-216，R-19459-216，R-1945A-216，R-1945C-216，R-1945D-216，R-1945E-216，R-F19456-2156，R-F19456-21C6」には，UNIX System Laboratories, Inc.が著作権を有している部分が含まれています。

本書には，X/Open の許諾に基づき X/Open CAE Specification System Interfaces and Headers, Issue4,（C202 ISBN 1-872630-47-2）Copyright (C) July 1992，X/Open Company Limited の内容が含まれています；

なお，その一部は IEEE Std 1003.1-1990，(C) 1990 Institute of Electrical and Electronics Engineers, Inc.及び IEEE std 1003.2/D12，(C) 1992 Institute of Electrical and Electronics Engineers, Inc.を基にしています。

事前に著作権所有者の許諾を得ずに，本書の該当部分を複製，複写及び転記することは禁じられています。

本書には，X/Open の許諾に基づき X/Open Preliminary Specification Distributed Transaction Processing：The TxRPC Specification（P305 ISBN 1-85912-000-8）Copyright (C) July 1993, X/Open Company Limited の内容が含まれています；

事前に著作権所有者の許諾を得ずに，本書の該当部分を複製，複写及び転記することは禁じられています。

本書には，Open Software Foundation, Inc.が著作権を有する内容が含まれています。

This document and the software described herein are furnished under a license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the above copyright notice. Title to and ownership of the document and software remain with OSF or its licensors.

■ 発行

2015 年 4 月 3000-3-D55-60

■ 著作権

All Rights Reserved. Copyright (C) 2006, 2015, Hitachi, Ltd.

変更内容

変更内容（3000-3-D55-60） uCosminexus TP1/Server Base 07-50，uCosminexus TP1/Server Base(64) 07-50，uCosminexus TP1/Message Control 07-50，uCosminexus TP1/Message Control(64) 07-50，uCosminexus TP1/NET/Library 07-50，uCosminexus TP1/NET/Library(64) 07-50

追加・変更内容	変更個所
初期状態での受け取り領域の説明を追加した。	2. メッセージ送受信（CBLDCMCF） CBLDCMCF('TEMPGET '), 3. サービス機能 RECEIVE － 一時記憶データの受け取り
アプリケーション属性定義との関連について説明を追加した。	2. メッセージ送受信（CBLDCMCF） CBLDCMCF('TEMPPUT '), 3. サービス機能 SEND － 一時記憶データの更新
次の項目の設定値に関する説明を追加した。 • CBLDCMCF('TIMERSET')のデータ名 G	2. メッセージ送受信（CBLDCMCF） CBLDCMCF('TIMERSET')
次の関数の要求コードの説明を変更した。 • CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR') • CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU') • CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')	2. TAM ファイルサービス（CBLDCTAM） CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR'), CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU'), CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')
参照目的の入力の場合に，排他しないで関数を呼び出した場合の説明を追加した。	2. TAM ファイルサービス（CBLDCTAM） CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU')

追加・変更内容	変更個所
CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')のステータスコードの説明を変更した。	2. TAM ファイルサービス（CBLDCTAM） CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')

単なる誤字・脱字などはお断りなく訂正しました。

変更内容（3000-3-D55-50） uCosminexus TP1/Server Base 07-06，uCosminexus TP1/Server Base(64) 07-06

追加・変更内容
DCUAPCONFPATH 環境変数にユーザサービスデフォルト定義ファイルが設定できる説明を追加した。
dcsvstart コマンドの-a オプションを使用して，SUP のメイン関数に第 1 引数を渡せる旨の説明を追加した。
次の関数のステータスコード，および注意事項の説明を追加した。 • CBLDCDAM('READ') • CBLDCDAM('REWT') • CBLDCDAM('WRIT')
CBLDCMCF('RECEIVE ') で受信できるメッセージに，ユーザタイマ監視を設定したときに指定したメッセージを追加した。

uCosminexus TP1/Server Base 07-05，uCosminexus TP1/Server Base(64) 07-05，uCosminexus TP1/Message Control 07-05，uCosminexus TP1/Message Control(64) 07-05

追加・変更内容
一つのリソースマネジャを複数の制御単位に分け，接続するユーザ名称などを変更してリソースマネジャに接続できるようにした（リソースマネジャ接続先選択機能）。 これに伴い，次の関数を追加した。 • CBLDCTRN('RMSELECT')
コーディング規約の注意事項を追加した。
データ名のけた数に関する注意事項，および 2 進形式のデータ項目に PICTURE 句で指定したけた数を超える値を指定する場合の説明を追加した。
CBLDCADT('PRINT ')のステータスコード，および注意事項の説明を追加した。
CBLDCRPC('CALL ')のステータスコードの説明を追加した。
CBLDCRPC('OPEN ')が OpenTP1 の各機能の環境設定（初期化）をする処理について注意事項を追加した。
XATMI インタフェース用の COPY ファイルの定義を変更した。

uCosminexus TP1/Message Control 07-00，uCosminexus TP1/Message Control(64) 07-00

追加・変更内容
リモート MCF サービスに関する記述を削除した。

変更内容（3000-3-D55-40） uCosminexus TP1/Server Base 07-04，uCosminexus TP1/Server Base(64) 07-04，uCosminexus TP1/Message Control 07-05，uCosminexus TP1/Message Control(64) 07-05，uCosminexus TP1/NET/Library 07-05，uCosminexus TP1/NET/Library(64) 07-05

追加・変更内容
次の項目の設定値に関する説明を追加した。 • CBLDCDMB('CRAT')のデータ名 D，データ名 G，およびデータ名 H • CBLDCDMB('OPEN')のデータ名 H
CBLDCJUP('RDGETRPT')の DATA DIVISION のデータ名 R の指定形式を変更した。
次の項目に関する説明を追加した。 • CBLDCLOG('PRINT ')のデータ名 G • CBLDCMCF('ROLLBACK')のデータ名 C • ROLLBACK － MHP のロールバックの WITH STOPPING 句
非応答型の MHP からの問い合わせ応答をできるようにした。
データ操作言語で，MHP をコミットできるようにした。 これに伴い，データ操作言語の次のプログラムを追加した。 • COMMIT － MHP のコミット

変更内容（3000-3-D55-30） uCosminexus TP1/Server Base 07-03，uCosminexus TP1/Server Base(64) 07-03，uCosminexus TP1/Message Control 07-03，uCosminexus TP1/Message Control(64) 07-03，uCosminexus TP1/NET/Library 07-04，uCosminexus TP1/NET/Library(64) 07-04

追加・変更内容
送受信できる一つのセグメントの最大長について記述を追加した。
データ操作言語の次のプログラムについて，DATA DIVISION（通信記述項）の指定に一意名の宣言方法を追加した。 • RECEIVE － メッセージの受信 • RECEIVE － 一時記憶データの受け取り • SEND － アプリケーションプログラムの起動 • SEND － 一時記憶データの更新 • SEND － 運用コマンドの実行 • SEND － ユーザジャーナルの取得

uCosminexus TP1/Message Control 07-02，uCosminexus TP1/NET/Library 07-03

追加・変更内容
アプリケーションに関するタイマ起動要求を，ライブラリ関数で削除できるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('ADLTAP ')
コネクションの状態表示，確立，および解放を，ライブラリ関数でできるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('TACTCN ') • CBLDCMCF('TDCTCN ') • CBLDCMCF('TLSCN ')
MCF 通信サービスまたはアプリケーション起動サービスの状態を，ライブラリ関数で表示できるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('TLSCOM ')
論理端末の状態表示，閉塞，閉塞解除，および出力キューの削除を，ライブラリ関数でできるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('TACTLE ') • CBLDCMCF('TDCTLE ') • CBLDCMCF('TDLQLE ') • CBLDCMCF('TLSLE ')
コネクションの確立要求の受付状態を，ライブラリ関数で表示できるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('TLSLN ')
サーバ型コネクションの確立要求の受付開始・終了を，ライブラリ関数でできるようにした。 これに伴い，次の関数を追加した。 • CBLDCMCF('TOFLN ') • CBLDCMCF('TONLN ')
MHP でサービス関数動的ローディング機能を使用できるようにした。

変更内容（3000-3-D55-20） uCosminexus TP1/Server Base 07-02，uCosminexus TP1/Message Control 07-01，uCosminexus TP1/NET/Library 07-01

追加・変更内容
監査ログを出力する機能を追加した。 これに伴い，CBLDCADT('PRINT ')関数を追加した。
サービス関数を動的にローディングできる機能を追加した。
システムジャーナルファイルを使用しないでシステムを運用する機能（ジャーナルファイルレス機能）を追加した。 これに伴い，関数のステータスコード，およびリターン値を変更した。

追加・変更内容
リモート API 機能に関する説明を変更した。 これに伴い，ステータスコードを追加，変更した。

はじめに

このマニュアルは，OpenTP1 のアプリケーションプログラムで使える，専用の COBOL-UAP 作成用プログラムの文法について説明しています。OpenTP1 のプログラムプロダクトを次に示します。

- 分散トランザクション処理機能　TP1/Server Base
- 分散アプリケーションサーバ　TP1/LiNK

このマニュアルでは，アプリケーションプログラムの英略称を「ユーザが作成するアプリケーションプログラム」の意味で，UAP（User Application Program）と表記します。

本文中に記載されている製品のうち，このマニュアルの対象製品ではない製品については，OpenTP1 Version 7 対応製品の発行時期をご確認ください。

■ 対象読者

TP1/Server Base，または TP1/LiNK で使うアプリケーションプログラム（UAP）を作成するプログラマの方々を対象としています。

オペレーティングシステム，オンラインシステム，使うマシンの操作，およびコーディングに使う COBOL 言語の文法の知識があることを前提としています。

このマニュアルは，マニュアル「OpenTP1 プログラム作成の手引」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。

■ 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。

(1)文法記述記号

設定する項目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
［　］ きっ甲	この記号で囲まれている項目は，省略できることを示します。 （例） stbmake ［-s［スタブソースファイル名］］ 　-s　スタブソースファイル名を省略できることを示します。
… 点線	記述が省略されていることを示します。この記号の直前に示された項目を繰り返し複数個指定できます。 （例） entry "プログラム ID" ［"プログラム ID"…］　； 　"プログラム ID"を続けて指定できることを示します。

文法記述記号	意味
△ 白三角	半角スペースを示します。

(2)属性表示記号

コマンドに指定する値の範囲の説明で使用する記号の一覧を示します。

属性表示記号	意味
～	この記号の前に示された項目が，記号 ～ に続く 〈 〉で示す規則に従って，値を指定することを示します。
〈 〉	項目を記述するときに従う構文要素を示します。

(3)構文要素

設定する項目の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	指 定 で き る 値
数字	0,1,2,3,4,5,6,7,8,9
英字	A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z, a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,アンダスコア(_)
英数字	数字，英字
記号	! # $ % & ’ ( ) * + - . / : ; < = > ? @ [＼ ¥] ^ _ ` { | } ~
パス名	/(スラント)，英数字，および .(ピリオド)の並び パス名は使用している OS に依存します。

(4)データ領域に設定する値の条件

データ領域に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 バイト以内のアスキー文字列で設定します。ただしヌル文字，空白，@（アットマーク），およびピリオドは除きます。 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 バイト以内のアスキー文字列で設定します。ただしヌル文字，空白は除きます。 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。

設定する値	指定できる値の条件
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。

■ X/Open 発行のドキュメントの内容から引用した記述について

X/Open 発行の「X/Open CAE Specification Distributed Transaction Processing : The XATMI Specification」の内容から引用した部分

このマニュアルの記述のうち，次に示す部分は，上記ドキュメントの「Chapter 7　COBOL Language Reference Manual Pages」の記述を，日本語訳したものです。

- 4 章　X/Open に準拠したアプリケーションプログラミングインタフェース

XATMI インタフェースのアプリケーションプログラミングインタフェース（TP～）

X/Open 発行の「X/Open CAE Specification Distributed Transaction Processing : The TX（Transaction Demarcation） Specification」の内容から引用した部分

このマニュアルの記述のうち，次に示す部分は，上記ドキュメントの「Chapter 6　COBOL Reference Manual Pages」の記述を，日本語訳したものです。

- 4 章　X/Open に準拠したアプリケーションプログラミングインタフェース

TX インタフェースのアプリケーションプログラミングインタフェース（TX～）

■ 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。OpenTP1 のアプリケーションプログラムのインタフェース仕様のうち，データ操作言語（DML　Data Manipulation Language）の仕様は，CODASYL　COBOL（1981）の通信節，RECEIVE 文，SEND 文，COMMIT 文，及び ROLLBACK 文を参考にし，それに日立製作所独自の解釈と仕様を追加して開発した。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。

■ その他の前提条件

このマニュアルをお読みになる際のその他の前提情報については，マニュアル「OpenTP1 解説」を参照してください。

目次

前書き　2
変更内容　4
はじめに　9

1　アプリケーションプログラムの作成　19
1.1　アプリケーションプログラムのコーディング　20
1.1.1　アプリケーションプログラムと COBOL-UAP 作成用プログラムの対応　20
1.1.2　コーディング規約　37
1.2　アプリケーションプログラムの作成（TCP/IP 通信）　40
1.2.1　アプリケーションプログラムの作成手順　40
1.2.2　スタブの作成方法　45
1.2.3　スタブのソースファイルの作成　47
1.2.4　stbmake（スタブのソースファイルの作成）　48
1.2.5　アプリケーションプログラムの翻訳と結合　49
1.3　XATMI インタフェースを使うアプリケーションプログラムの作成（TCP/IP 通信，OSI TP 通信）　52
1.3.1　アプリケーションプログラムの作成手順　52
1.3.2　XATMI インタフェース用スタブの作成方法　53
1.3.3　XATMI インタフェース用スタブのソースファイルの作成　60
1.3.4　stbmake（XATMI インタフェース用スタブの作成　TCP/IP 通信）　61
1.3.5　tpstbmk（XATMI インタフェース用スタブの作成　OSI TP 通信）　63
1.4　アプリケーションプログラムの実行　65
1.4.1　アプリケーションプログラムの開始と終了　65
1.4.2　OpenTP1 で開始したアプリケーションプログラムの動作環境　66
1.4.3　アプリケーションプログラムの環境変数　68
1.4.4　トラブルシュート関連　68

2　OpenTP1 の COBOL-UAP 作成用プログラムの文法　70
COBOL-UAP 作成用プログラムの説明形式　71
メインプログラムとサービスプログラムの作成　74
メインプログラムの作成（SUP，SPP，MHP）　75
サービスプログラムの作成（SPP）　77
サービスプログラムの作成（MHP）　81
システム運用の管理（CBLDCADM）　83
CBLDCADM('COMMAND ')　84
CBLDCADM('COMPLETE')　87
CBLDCADM('STATUS ')　89

監査ログの出力（CBLDCADT）　91
CBLDCADT('PRINT ')　92
DAM ファイルサービス（CBLDCDAM，CBLDCDMB）　97
CBLDCDAM('CLOS')　98
CBLDCDAM('END ')　101
CBLDCDAM('HOLD')　103
CBLDCDAM('OPEN')　105
CBLDCDAM('READ')　110
CBLDCDAM('REWT')　116
CBLDCDAM('RLES')　120
CBLDCDAM('STAT')　123
CBLDCDAM('STRT')　127
CBLDCDAM('WRIT')　129
CBLDCDMB('BSEK')　134
CBLDCDMB('CLOS')　136
CBLDCDMB('CRAT')　138
CBLDCDMB('DGET')　142
CBLDCDMB('DPUT')　145
CBLDCDMB('GET ')　148
CBLDCDMB('OPEN')　151
CBLDCDMB('PUT ')　154
IST サービス（CBLDCIST）　157
CBLDCIST('CLOS')　158
CBLDCIST('OPEN')　160
CBLDCIST('READ')　162
CBLDCIST('WRIT')　165
ユーザジャーナルの取得（CBLDCJNL）　168
CBLDCJNL('UJPUT ')　169
ジャーナルデータの編集（CBLDCJUP）　171
CBLDCJUP('CLOSERPT')　172
CBLDCJUP('OPENRPT ')　174
CBLDCJUP('RDGETRPT')　177
資源の排他制御（CBLDCLCK）　186
CBLDCLCK('GET ')　187
CBLDCLCK('RELALL ')　190
CBLDCLCK('RELNAME ')　192
メッセージログの出力（CBLDCLOG）　194
CBLDCLOG('PRINT ')　195
メッセージ送受信（CBLDCMCF）　198
CBLDCMCF('ADLTAP ')　200
CBLDCMCF('APINFO ')　203
CBLDCMCF('CLOSE ')　209
CBLDCMCF('COMMIT ')　211
CBLDCMCF('CONTEND ')　214
CBLDCMCF('EXECAP ')　216

CBLDCMCF('MAINLOOP') 224
CBLDCMCF('OPEN ') 226
CBLDCMCF('RECEIVE ') 228
CBLDCMCF('RECVSYNC') 234
CBLDCMCF('REPLY ') 235
CBLDCMCF('RESEND ') 236
CBLDCMCF('ROLLBACK') 237
CBLDCMCF('SEND ') 240
CBLDCMCF('SENDRECV') 241
CBLDCMCF('SENDSYNC') 242
CBLDCMCF('TACTCN ') 243
CBLDCMCF('TACTLE ') 247
CBLDCMCF('TDCTCN ') 251
CBLDCMCF('TDCTLE ') 255
CBLDCMCF('TDLQLE ') 259
CBLDCMCF('TEMPGET ') 262
CBLDCMCF('TEMPPUT ') 267
CBLDCMCF('TIMERCAN') 271
CBLDCMCF('TIMERSET') 273
CBLDCMCF('TLSCN ') 277
CBLDCMCF('TLSCOM ') 282
CBLDCMCF('TLSLE ') 286
CBLDCMCF('TLSLN ') 289
CBLDCMCF('TOFLN ') 292
CBLDCMCF('TONLN ') 294
性能検証用トレース（CBLDCPRF） 296
CBLDCPRF('PRFGETN ') 297
CBLDCPRF('PRFPUT ') 299
リモート API 機能(CBLDCRAP) 301
CBLDCRAP('CONNECT ') 302
CBLDCRAP('CONNECTX') 305
CBLDCRAP('DISCNCT ') 308
リモートプロシジャコール（CBLDCRPC，CBLDCRSV） 310
CBLDCRPC('CALL ') 311
CBLDCRPC('CLOSE ') 328
CBLDCRPC('CLTSEND ') 330
CBLDCRPC('DISCARDF') 333
CBLDCRPC('DISCARDS') 335
CBLDCRPC('GETCLADR') 337
CBLDCRPC('GETERDES') 339
CBLDCRPC('GETGWADR') 341
CBLDCRPC('GETSVPRI') 343
CBLDCRPC('GETWATCH') 345
CBLDCRPC('OPEN ') 347
CBLDCRPC('POLLANYR') 349

CBLDCRPC('SETSVPRI') 355
CBLDCRPC('SETWATCH') 357
CBLDCRPC('SVRETRY ') 359
CBLDCRSV('MAINLOOP') 361
リアルタイム統計情報サービス（CBLDCRTS） 363
CBLDCRTS('RTSPUT ') 364
TAM ファイルサービス（CBLDCTAM） 367
CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR') 368
CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU') 372
CBLDCTAM('GST ') 378
CBLDCTAM('INFO') 381
CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ') 386
トランザクション制御（CBLDCTRN） 390
CBLDCTRN('BEGIN ') 391
CBLDCTRN('C-COMMIT') 393
CBLDCTRN('C-ROLL ') 396
CBLDCTRN('INFO ') 399
CBLDCTRN('RMSELECT') 401
CBLDCTRN('U-COMMIT') 404
CBLDCTRN('U-ROLL ') 407
オンラインテスタの管理（CBLDCUTO） 409
CBLDCUTO('T-STATUS') 410

3 OpenTP1 の COBOL-UAP 作成用プログラムの文法（DML インタフェース） 413
データ操作言語を使ったコーディング 414
データコミュニケーション機能 419
RECEIVE － メッセージの受信 420
SEND － メッセージの送信 424
サービス機能 425
COMMIT － MHP のコミット 426
DISABLE － 継続問い合わせ応答の終了 427
RECEIVE － 一時記憶データの受け取り 429
ROLLBACK － MHP のロールバック 432
SEND － アプリケーションプログラムの起動 433
SEND － 一時記憶データの更新 439
SEND － 運用コマンドの実行 441
SEND － ユーザジャーナルの取得 444

4 X/Open に準拠したアプリケーションプログラミングインタフェース 446
X/Open に準拠した API 447
XATMI インタフェースのアプリケーションプログラミングインタフェース（TP～） 450
TPINTRO 451
TPACALL 454
TPADVERTISE 458

TPCALL　460
TPCANCEL　466
TPCONNECT　468
TPDISCON　473
TPGETRPLY　475
TPRECV　480
TPRETURN　485
TPSEND　489
TPSVCSTART　494
TPUNADVERTISE　497
TX インタフェースのアプリケーションプログラミングインタフェース（TX～）　499
TXINTRO　500
TXBEGIN　503
TXCLOSE　505
TXCOMMIT　507
TXINFORM　510
TXOPEN　512
TXROLLBACK　514
TXSETCOMMITRET　517
TXSETTIMEOUT　520
TXSETTRANCTL　522

5　OpenTP1 の COBOL-UAP 作成用プログラムの文法（アソシエーションの状態の通知）　524
アソシエーションの操作（CBLDCXAT）　525
CBLDCXAT('CONNECT ')　526
受信する通信イベントの形式　528

6　コーディング例　530
6.1　クライアント／サーバ形態の UAP のコーディング例（SUP，SPP　DAM アクセス）　531
6.1.1　SUP の例　531
6.1.2　SPP の例（メインプログラム）　533
6.1.3　SPP の例（サービスプログラム）　536
6.2　クライアント／サーバ形態の UAP のコーディング例（SPP　TAM アクセス）　539
6.2.1　SPP の例（メインプログラム）　539
6.2.2　SPP の例（サービスプログラム）　540
6.3　メッセージ送受信形態の UAP のコーディング例（MHP）　543
6.3.1　MHP の例（メインプログラム）　543
6.3.2　MHP の例（サービスプログラム）　545
6.3.3　MHP の例（サービスプログラム　DML の例）　549
6.4　X/Open に準拠した UAP のコーディング例　552
6.4.1　XATMI インタフェースの例　552
6.4.2　TX インタフェースの例　575

7 アプリケーション起動関連のリファレンス 579
タイマ起動引き継ぎ決定 UOC の関数形式 580
タイマ起動メッセージ廃棄通知イベント（ERREVT4）のデータ形式 584

付録 586
付録 A OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合 587
付録 A.1 併用する形態 587
付録 A.2 併用するアプリケーションプログラムのスタブの作成手順 588
付録 A.3 呼び出せる XATMI インタフェースの API 589

索引 590

1 アプリケーションプログラムの作成

この章では，OpenTP1 のアプリケーションプログラムを COBOL 言語でコーディングする場合の概要について説明します。

1.1.1　アプリケーションプログラムと COBOL-UAP 作成用プログラムの対応

OpenTP1 の UAP で使える OpenTP1 の機能と，COBOL-UAP 作成用プログラムの対応を次の表に示します。

表 1-1　OpenTP1 の機能と COBOL-UAP 作成用プログラムの対応

機能名		CALL 文で呼び出す COBOL-UAP 作成用プログラム
システム運用の管理	運用コマンドの実行	`CBLDCADM('COMMAND ')`
	ユーザサーバの開始処理完了の報告	`CBLDCADM('COMPLETE')`
	ユーザサーバの状態の報告	`CBLDCADM('STATUS  ')`
DAM ファイルサービス	論理ファイルのクローズ	`CBLDCDAM('DCDAMSVC','CLOS    ')`
	回復対象外 DAM ファイル使用の終了	`CBLDCDAM('DCDAMSVC','END     ')`
	論理ファイルの閉塞	`CBLDCDAM('DCDAMSVC','HOLD    ')`
	論理ファイルのオープン	`CBLDCDAM('DCDAMSVC','OPEN    ')`
	論理ファイルからブロックの入力	`CBLDCDAM('DCDAMSVC','READ    ')`
	論理ファイルのブロックの更新	`CBLDCDAM('DCDAMSVC','REWT    ')`
	論理ファイルの閉塞の解除	`CBLDCDAM('DCDAMSVC','RLES    ')`
	論理ファイルの状態の参照	`CBLDCDAM('DCDAMSVC','STAT    ')`
	回復対象外 DAM ファイル使用の開始	`CBLDCDAM('DCDAMSVC','STRT    ')`
	論理ファイルへブロックの出力	`CBLDCDAM('DCDAMSVC','WRIT    ')`
	物理ファイルのブロックの検索	`CBLDCDMB('DCDAMINT','BSEK    ')`
	物理ファイルのクローズ	`CBLDCDMB('DCDAMINT','CLOS    ')`
	物理ファイルの割り当て	`CBLDCDMB('DCDAMINT','CRAT    ')`
	物理ファイルからブロックの直接入力	`CBLDCDMB('DCDAMINT','DGET    ')`

機能名		CALL 文で呼び出す COBOL-UAP 作成用プログラム
DAM ファイルサービス	物理ファイルへブロックの直接出力	CBLDCDMB('DCDAMINT','DPUT ')
	物理ファイルからブロックの入力	CBLDCDMB('DCDAMINT','GET ')
	物理ファイルのオープン	CBLDCDMB('DCDAMINT','OPEN ')
	物理ファイルへブロックの出力	CBLDCDMB('DCDAMINT','PUT ')
IST サービス	IST テーブルのクローズ	CBLDCIST('DCISTSVC','CLOS ')
	IST テーブルのオープン	CBLDCIST('DCISTSVC','OPEN ')
	IST テーブルからレコードの入力	CBLDCIST('DCISTSVC','READ ')
	IST テーブルへレコードの出力	CBLDCIST('DCISTSVC','WRIT ')
ユーザジャーナルの取得	ユーザジャーナルの取得	CBLDCJNL('UJPUT ')
ジャーナルデータの編集	jnlrput 出力ファイルのクローズ	CBLDCJUP('CLOSERPT')
	jnlrput 出力ファイルのオープン	CBLDCJUP('OPENRPT ')
	jnlrput 出力ファイルからジャーナルデータの入力	CBLDCJUP('RDGETRPT')
資源の排他制御	資源の排他	CBLDCLCK('GET ')
	全資源の排他の解除	CBLDCLCK('RELALL ')
	資源名称を指定した排他の解除	CBLDCLCK('RELNAME ')
監査ログの出力	監査ログの出力	CBLDCADT('PRINT ')
メッセージログの出力	メッセージログの出力	CBLDCLOG('PRINT ')
メッセージ送受信	アプリケーションに関するタイマ起動要求の削除	CBLDCMCF('ADLTAP ')
	アプリケーション情報通知	CBLDCMCF('APINFO ')
	MCF 環境のクローズ	CBLDCMCF('CLOSE ')
	MHP のコミット	CBLDCMCF('COMMIT ')
	継続問い合わせ応答の終了	CBLDCMCF('CONTEND ')
	アプリケーションプログラムの起動	CBLDCMCF('EXECAP ')

機能名		CALL 文で呼び出す COBOL-UAP 作成用プログラム
メッセージ送受信	MHP のサービス開始	CBLDCMCF('MAINLOOP')
	MCF 環境のオープン	CBLDCMCF('OPEN ')
	メッセージの受信	CBLDCMCF('RECEIVE ')
	同期型のメッセージの受信	CBLDCMCF('RECVSYNC')
	応答メッセージの送信	CBLDCMCF('REPLY ')
	メッセージの再送	CBLDCMCF('RESEND ')
	MHP のロールバック	CBLDCMCF('ROLLBACK')
	メッセージの送信	CBLDCMCF('SEND ')
	同期型のメッセージの送受信	CBLDCMCF('SENDRECV')
	同期型のメッセージの送信	CBLDCMCF('SENDSYNC')
	コネクションの確立	CBLDCMCF('TACTCN ')
	論理端末の閉塞解除	CBLDCMCF('TACTLE ')
	コネクションの解放	CBLDCMCF('TDCTCN ')
	論理端末の閉塞	CBLDCMCF('TDCTLE ')
	論理端末の出力キュー削除	CBLDCMCF('TDLQLE ')
	一時記憶データの受け取り	CBLDCMCF('TEMPGET ')
	一時記憶データの更新	CBLDCMCF('TEMPPUT ')
	ユーザタイマ監視の取り消し	CBLDCMCF('TIMERCAN')
	ユーザタイマ監視の設定	CBLDCMCF('TIMERSET')
	コネクションの状態取得	CBLDCMCF('TLSCN ')
	MCF 通信サービスの状態取得	CBLDCMCF('TLSCOM ')
	論理端末の状態取得	CBLDCMCF('TLSLE ')
	サーバ型コネクションの確立要求の受付状態取得	CBLDCMCF('TLSLN ')

機能名		CALL 文で呼び出す COBOL-UAP 作成用プログラム
メッセージ送受信	サーバ型コネクションの確立要求の受付終了	CBLDCMCF('TOFLN ')
	サーバ型コネクションの確立要求の受付開始	CBLDCMCF('TONLN ')
性能検証用トレース	性能検証用トレース取得通番の通知	CBLDCPRF('PRFGETN ')
	ユーザ固有の性能検証用トレースの取得	CBLDCPRF('PRFPUT ')
リモート API 機能	rap リスナーとのコネクションの確立	CBLDCRAP('CONNECT ') CBLDCRAP('CONNECTX')
	rap リスナーとのコネクションの解放	CBLDCRAP('DISCNCT ')
リモートプロシジャコール	遠隔サービスの要求	CBLDCRPC('CALL ')
	アプリケーションプログラムの終了	CBLDCRPC('CLOSE ')
	CUP への一方通知	CBLDCRPC('CLTSEND ')
	処理結果の受信の拒否	CBLDCRPC('DISCARDF')
	特定の処理結果の受信の拒否	CBLDCRPC('DISCARDS')
	クライアント UAP のノードアドレスの取得	CBLDCRPC('GETCLADR')
	エラーが発生した非同期応答型 RPC 要求の記述子の取得	CBLDCRPC('GETERDES')
	ゲートウェイのノードアドレスの取得	CBLDCRPC('GETGWADR')
	サービス要求のスケジュールプライオリティの参照	CBLDCRPC('GETSVPRI')
	サービスの応答待ち時間の参照	CBLDCRPC('GETWATCH')
	アプリケーションプログラムの開始	CBLDCRPC('OPEN ')
	処理結果の非同期受信	CBLDCRPC('POLLANYR')
	サービス要求のスケジュールプライオリティの設定	CBLDCRPC('SETSVPRI')
	サービスの応答待ち時間の更新	CBLDCRPC('SETWATCH')
	サービスプログラムのリトライ	CBLDCRPC('SVRETRY ')

機能名		CALL 文で呼び出す COBOL-UAP 作成用プログラム
リモートプロシジャコール	SPP のサービス開始	CBLDCRSV('MAINLOOP')
リアルタイム統計情報サービス	任意区間でのリアルタイム統計情報の取得	CBLDCRTS('RTSPUT ')
TAM ファイルサービス※	TAM テーブルのレコードの削除	CBLDCTAM('ERS ')('ERSR')('ZRS ')('ZRSR')
	TAM テーブルからレコードの入力	CBLDCTAM('FxxR')('FxxU')('VxxR')('VxxU')
	TAM テーブルの状態の取得	CBLDCTAM('GST ')
	TAM テーブルの情報の取得	CBLDCTAM('INFO')
	TAM テーブルのレコードの更新/追加	CBLDCTAM('MFY ')('MFYS')('STR ')('WFY ') ('WFYS')('YTR ')
トランザクション制御	トランザクションの開始	CBLDCTRN('BEGIN ')
	連鎖モードのコミット	CBLDCTRN('C-COMMIT')
	連鎖モードのロールバック	CBLDCTRN('C-ROLL ')
	現在のトランザクションに関する情報の報告	CBLDCTRN('INFO ')
	非連鎖モードのコミット	CBLDCTRN('U-COMMIT')
	非連鎖モードのロールバック	CBLDCTRN('U-ROLL ')
	リソースマネジャ接続先選択	CBLDCTRN('RMSELECT')
オンラインテスタの管理	ユーザサーバのテスト状態の報告	CBLDCUTO('T-STATUS')

注※　次に示す TAM ファイルサービスの API は，COBOL 言語では使えません。

- TAM テーブルのオープン
- TAM テーブルのクローズ
- TAM テーブルのレコードの入力取り消し

(1)　SUP で使える機能とプログラム

SUP で使える機能と要求コードを次の表に示します。

表 1-2　SUP で使える機能と要求コード

SUP で 使える機能	SUP で使える機能名と， COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SUP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
システム運用の管理	運用コマンドの実行	'COMMAND '	○	○
	ユーザサーバの開始処理完了の報告	'COMPLETE'	○	−
	ユーザサーバの状態の報告	'STATUS '	○	○
DAM ファイルサービス	論理ファイルのクローズ	'DCDAMSVC','CLOS'	○	○
	回復対象外 DAM ファイル使用の終了	'DCDAMSVC','END '	○	○
	論理ファイルの閉塞	'DCDAMSVC','HOLD'	○	○
	論理ファイルのオープン	'DCDAMSVC','OPEN'	○	○
	論理ファイルからブロックの入力	'DCDAMSVC','READ'	○	○
	論理ファイルのブロックの更新	'DCDAMSVC','REWT'	(○)	○
	論理ファイルの閉塞の解除	'DCDAMSVC','RLES'	○	○
	論理ファイルの状態の参照	'DCDAMSVC','STAT'	○	○
	回復対象外 DAM ファイル使用の開始	'DCDAMSVC','STRT'	○	○
	論理ファイルへブロックの出力	'DCDAMSVC','WRIT'	(○)	○
IST サービス	IST テーブルのクローズ	'DCISTSVC','CLOS'	○	○
	IST テーブルのオープン	'DCISTSVC','OPEN'	○	○
	IST テーブルからレコードの入力	'DCISTSVC','READ'	○	○
	IST テーブルへレコードの出力	'DCISTSVC','WRIT'	○	○
ユーザジャーナルの取得	ユーザジャーナルの取得	'UJPUT '	○	○
資源の排他制御	資源の排他	'GET '	−	○
	全資源の排他の解除	'RELALL '	−	○
	資源名称を指定した排他の解除	'RELNAME '	−	○

SUP で使える機能	SUP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SUP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
監査ログの出力	監査ログの出力	'PRINT '	○	○
メッセージログの出力	メッセージログの出力	'PRINT '	○	○
性能検証用トレース	性能検証用トレース取得通番の通知	'PRFGETN '	○	○
	ユーザ固有の性能検証用トレースの取得	'PRFPUT '	○	○
リモート API 機能	rap リスナーとのコネクションの確立	'CONNECT ' 'CONNECTX'	○	−
	rap リスナーとのコネクションの解放	'DISCNCT '	○	−
リモートプロシジャコール	遠隔サービスの要求	'CALL '	○	○
	アプリケーションプログラムの終了	'CLOSE '	○	−
	処理結果の受信の拒否	'DISCARDF'	○	○
	特定の処理結果の受信の拒否	'DISCARDS'	○	○
	エラーが発生した非同期型応答型 RPC 要求の記述子の取得	'GETERDES'	○	○
	サービス要求のスケジュールプライオリティの参照	'GETSVPRI'	○	○
	サービスの応答待ち時間の参照	'GETWATCH'	○	○
	アプリケーションプログラムの開始	'OPEN '	○	−
	処理結果の非同期受信	'POLLANYR'	○	○
	サービス要求のスケジュールプライオリティの設定	'SETSVPRI'	○	○
	サービスの応答待ち時間の更新	'SETWATCH'	○	○
リアルタイム統計情報サービス	任意区間でのリアルタイム統計情報の取得	'RTSPUT '	○	○
TAM ファイルサービス	TAM テーブルのレコードの削除	'ERS '/'ERSR'/'ZRS '/'ZRSR'	−	○
	TAM テーブルからレコードの入力	'FxxR'/'FxxU'/'VxxR'/'VxxU'	−	○

SUP で使える機能	SUP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SUP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
TAM ファイルサービス	TAM テーブルの状態の取得	'GST '	○	○
	TAM テーブルの情報の取得	'INFO'	○	○
	TAM テーブルのレコードの更新/追加	'MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR '	−	○
トランザクション制御	トランザクションの開始	'BEGIN '	○	−
	連鎖モードのコミット	'C-COMMIT'	−	○
	連鎖モードのロールバック	'C-ROLL '	−	○
	現在のトランザクションに関する情報の報告	'INFO '	○	○
	非連鎖モードのコミット	'U-COMMIT'	−	○
	非連鎖モードのロールバック	'U-ROLL '	−	○
	リソースマネジャ接続先選択	'RMSELECT'	○	−
オンラインテスタの管理	ユーザサーバのテスト状態の報告	'T-STATUS'	○	○

(凡例)

○：該当する条件で使えます。

(○)：回復対象外の DAM ファイルへアクセスするときだけ，使えます。

−：該当する条件では使えません。

(2) SPP で使える機能とプログラム

SPP で使える機能と要求コードを次の表に示します。

表 1-3 SPP で使える機能と要求コード

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
システム運用の管理	運用コマンドの実行	'COMMAND '	○	○	○

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
システム運用の管理	ユーザサーバの状態の報告	'STATUS '	○	○	○
DAM ファイルサービス	論理ファイルのクローズ	'DCDAMSVC','CLOS'	○	○	○
	回復対象外 DAM ファイル使用の終了	'DCDAMSVC','END '	○	○	○
	論理ファイルの閉塞	'DCDAMSVC','HOLD'	○	○	○
	論理ファイルのオープン	'DCDAMSVC','OPEN'	○	○	○
	論理ファイルからブロックの入力	'DCDAMSVC','READ'	○	○	○
	論理ファイルのブロックの更新	'DCDAMSVC','REWT'	(○)	○	○
	論理ファイルの閉塞の解除	'DCDAMSVC','RLES'	○	○	○
	論理ファイルの状態の参照	'DCDAMSVC','STAT'	○	○	○
	回復対象外 DAM ファイル使用の開始	'DCDAMSVC','STRT'	○	○	○
	論理ファイルへブロックの出力	'DCDAMSVC','WRIT'	(○)	○	○
IST サービス	IST テーブルのクローズ	'DCISTSVC','CLOS'	○	○	○
	IST テーブルのオープン	'DCISTSVC','OPEN'	○	○	○
	IST テーブルからレコードの入力	'DCISTSVC','READ'	○	○	○
	IST テーブルへレコードの出力	'DCISTSVC','WRIT'	○	○	○
ユーザジャーナルの取得	ユーザジャーナルの取得	'UJPUT '	○	○	○
資源の排他制御	資源の排他	'GET '	−	○	○
	全資源の排他の解除	'RELALL '	−	○	○
	資源名称を指定した排他の解除	'RELNAME '	−	○	○

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
監査ログの出力	監査ログの出力	'PRINT '	○	○	○
メッセージログの出力	メッセージログの出力	'PRINT '	○	○	○
メッセージ送受信	アプリケーションに関するタイマ起動要求の削除	'ADLTAP '	○	○	○
	MCF 環境のクローズ	'CLOSE '	○M	−	−
	アプリケーションプログラムの起動	'EXECAP '	−	○	○
	MCF 環境のオープン	'OPEN '	○M	−	−
	同期型のメッセージの受信	'RECVSYNC'	○	○	○
	メッセージの再送	'RESEND '	−	○	○
	メッセージの送信	'SEND '	−	○	○
	同期型のメッセージの送受信	'SENDRECV'	○	○	○
	同期型のメッセージの送信	'SENDSYNC'	○	○	○
	コネクションの確立	'TACTCN '	○	○	○
	論理端末の閉塞解除	'TACTLE '	○	○	○
	コネクションの解放	'TDCTCN '	○	○	○
	論理端末の閉塞	'TDCTLE '	○	○	○
	論理端末の出力キュー削除	'TDLQLE '	○	○	○
	ユーザタイマ監視の取り消し	'TIMERCAN'	○	○	○
	ユーザタイマ監視の設定	'TIMERSET'	○	○	○
	コネクションの状態取得	'TLSCN '	○	○	○
	MCF 通信サービスの状態取得	'TLSCOM '	○	○	○

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
メッセージ送受信	論理端末の状態取得	'TLSLE　'	○	○	○
	サーバ型コネクションの確立要求の受付状態取得	'TLSLN　'	○	○	○
	サーバ型コネクションの確立要求の受付終了	'TOFLN　'	○	○	○
	サーバ型コネクションの確立要求の受付開始	'TONLN　'	○	○	○
性能検証用トレース	性能検証用トレース取得通番の通知	'PRFGETN '	○	○	○
	ユーザ固有の性能検証用トレースの取得	'PRFPUT　'	○	○	○
リモート API 機能	rap リスナーとのコネクションの確立	'CONNECT ' 'CONNECTX'	○	−	−
	rap リスナーとのコネクションの解放	'DISCNCT '	○	−	−
リモートプロシジャコール	遠隔サービスの要求	'CALL　'	○	○	○
	アプリケーションプログラムの終了	'CLOSE　'	○M	−	−
	CUP への一方通知	'CLTSEND '	○	○	○
	処理結果の受信の拒否	'DISCARDF'	○	○	○
	特定の処理結果の受信の拒否	'DISCARDS'	○	○	○
	クライアント UAP のノードアドレスの取得	'GETCLADR'	○	○	○
	エラーが発生した非同期応答型 RPC 要求の記述子の取得	'GETERDES'	○	○	○
	ゲートウェイのノードアドレスの取得	'GETGWADR'	○	○	○
	サービス要求のスケジュールプライオリティの参照	'GETSVPRI'	○	○	○

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
リモートプロシジャコール	サービスの応答待ち時間の参照	'GETWATCH'	○	○	○
	アプリケーションプログラムの開始	'OPEN　'	○M	−	−
	処理結果の非同期受信	'POLLANYR'	○	○	○
	サービス要求のスケジュールプライオリティの設定	'SETSVPRI'	○	○	○
	サービスの応答待ち時間の更新	'SETWATCH'	○	○	○
	サービスプログラムのリトライ	'SVRETRY '	○	−	−
	SPP のサービス開始	'MAINLOOP'	○M	−	−
リアルタイム統計情報サービス	任意区間でのリアルタイム統計情報の取得	'RTSPUT　'	○	○	○
TAM ファイルサービス	TAM テーブルのレコードの削除	'ERS '/'ERSR'/'ZRS '/'ZRSR'	−	○	○
	TAM テーブルからレコードの入力	'FxxR'/'FxxU'/'VxxR'/'VxxU'	−	○	○
	TAM テーブルの状態の取得	'GST '	○	○	○
	TAM テーブルの情報の取得	'INFO'	○	○	○
	TAM テーブルのレコードの更新/追加	'MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR '	−	○	○
トランザクション制御	トランザクションの開始	'BEGIN　'	○	−	−
	連鎖モードのコミット	'C-COMMIT'	−	○	−
	連鎖モードのロールバック	'C-ROLL　'	−	○	−
	現在のトランザクションに関する情報の報告	'INFO　'	○	○	○
	非連鎖モードのコミット	'U-COMMIT'	−	○	−

SPP で使える機能	SPP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		SPP が稼働している条件		
			トランザクションの処理の範囲でない	トランザクションの範囲	
				ルート	ルート以外
トランザクション制御	非連鎖モードのロールバック	'U-ROLL '	−	○	○
	リソースマネジャ接続先選択	'RMSELECT'	○	−	−
オンラインテスタの管理	ユーザサーバのテスト状態の報告	'T-STATUS'	○	○	○

(凡例)

○：該当する条件で使えます。

(○)：回復対象外の DAM ファイルへアクセスするときだけ，使えます。

$○_M$：メインプログラムからだけ，使えます。

−：該当する条件では使えません。

注

「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。

(3) MHP で使える機能とプログラム

MHP で使える機能と要求コードを次の表に示します。

表 1-4 MHP で使える機能と要求コード

MHP で使える機能	MHP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		MHP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
システム運用の管理	運用コマンドの実行	'COMMAND '	○	○
	ユーザサーバの状態の報告	'STATUS '	○	○
DAM ファイルサービス	論理ファイルのクローズ	'DCDAMSVC','CLOS'	○	○
	回復対象外 DAM ファイル使用の終了	'DCDAMSVC','END '	○	○
	論理ファイルの閉塞	'DCDAMSVC','HOLD'	○	○
	論理ファイルのオープン	'DCDAMSVC','OPEN'	○	○

MHP で使える機能	MHP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		MHP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
DAM ファイルサービス	論理ファイルからブロックの入力	'DCDAMSVC','READ'	○	○
	論理ファイルのブロックの更新	'DCDAMSVC','REWT'	(○)	○
	論理ファイルの閉塞の解除	'DCDAMSVC','RLES'	○	○
	論理ファイルの状態の参照	'DCDAMSVC','STAT'	○	○
	回復対象外 DAM ファイル使用の開始	'DCDAMSVC','STRT'	○	○
	論理ファイルへブロックの出力	'DCDAMSVC','WRIT'	(○)	○
IST サービス	IST テーブルのクローズ	'DCISTSVC','CLOS'	○	○
	IST テーブルのオープン	'DCISTSVC','OPEN'	○	○
	IST テーブルからレコードの入力	'DCISTSVC','READ'	○	○
	IST テーブルへレコードの出力	'DCISTSVC','WRIT'	○	○
ユーザジャーナルの取得	ユーザジャーナルの取得	'UJPUT '	○	○
資源の排他制御	資源の排他	'GET '	−	○
	全資源の排他の解除	'RELALL '	−	○
	資源名称を指定した排他の解除	'RELNAME '	−	○
監査ログの出力	監査ログの出力	'PRINT '	○	○
メッセージログの出力	メッセージログの出力	'PRINT '	○	○
メッセージ送受信	アプリケーションに関するタイマ起動要求の削除	'ADLTAP '	○	○
	アプリケーション情報通知	'APINFO '	$○_{NO}$	○
	MCF 環境のクローズ	'CLOSE '	$○_{M}$	$○_{M}$
	MHP のコミット	'COMMIT '	−	○

MHP で使える機能	MHP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		MHP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
メッセージ送受信	継続問い合わせ応答の終了	'CONTEND '	○NO	○
	アプリケーションプログラムの起動	'EXECAP '	○NO	○
	MHP のサービス開始	'MAINLOOP'	○M	−
	MCF 環境のオープン	'OPEN '	○M	○M
	メッセージの受信	'RECEIVE '	○NO	○
	同期型のメッセージの受信	'RECVSYNC'	○	○
	応答メッセージの送信	'REPLY '	○NO	○
	メッセージの再送	'RESEND '	−	○
	MHP のロールバック	'ROLLBACK'	−	○
	メッセージの送信	'SEND '	○NO	○
	同期型のメッセージの送受信	'SENDRECV'	○	○
	同期型のメッセージの送信	'SENDSYNC'	○	○
	コネクションの確立	'TACTCN '	○	○
	論理端末の閉塞解除	'TACTLE '	○	○
	コネクションの解放	'TDCTCN '	○	○
	論理端末の閉塞	'TDCTLE '	○	○
	論理端末の出力キュー削除	'TDLQLE '	○	○
	一時記憶データの受け取り	'TEMPGET '	○NO	○
	一時記憶データの更新	'TEMPPUT '	○NO	○
	ユーザタイマ監視の取り消し	'TIMERCAN'	○	○
	ユーザタイマ監視の設定	'TIMERSET'	○	○

MHP で使える機能	MHP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		MHP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
メッセージ送受信	コネクションの状態取得	'TLSCN '	○	○
	MCF 通信サービスの状態取得	'TLSCOM '	○	○
	論理端末の状態取得	'TLSLE '	○	○
	サーバ型コネクションの確立要求の受付状態取得	'TLSLN '	○	○
	サーバ型コネクションの確立要求の受付終了	'TOFLN '	○	○
	サーバ型コネクションの確立要求の受付開始	'TONLN '	○	○
性能検証用トレース	性能検証用トレース取得通番の通知	'PRFGETN '	○	○
	ユーザ固有の性能検証用トレースの取得	'PRFPUT '	○	○
リモート API 機能	rap リスナーとのコネクションの確立	'CONNECT ' 'CONNECTX'	○	−
	rap リスナーとのコネクションの解放	'DISCNCT '	○	−
リモートプロシジャコール	遠隔サービスの要求	'CALL '	○M	○
	アプリケーションプログラムの終了	'CLOSE '	○M	−
	CUP への一方通知	'CLTSEND '	○	○
	処理結果の受信の拒否	'DISCARDF'	○	○
	特定の処理結果の受信の拒否	'DISCARDS'	○	○
	エラーが発生した非同期応答型 RPC 要求の記述子の取得	'GETERDES'	○	○
	サービス要求のスケジュールプライオリティの参照	'GETSVPRI'	○	○
	サービスの応答待ち時間の参照	'GETWATCH'	○	○
	アプリケーションプログラムの開始	'OPEN '	○M	−

MHP で使える機能	MHP で使える機能名と，COBOL-UAP 作成用プログラムの先頭のデータ名に指定する要求コード		MHP が稼働している条件	
			トランザクションの処理の範囲でない	トランザクションの処理の範囲
リモートプロシジャコール	処理結果の非同期受信	'POLLANYR'	○M	○
	サービス要求のスケジュールプライオリティの設定	'SETSVPRI'	○	○
	サービスの応答待ち時間の更新	'SETWATCH'	○	○
リアルタイム統計情報サービス	任意区間でのリアルタイム統計情報の取得	'RTSPUT '	○	○
TAM ファイルサービス	TAM テーブルのレコードの削除	'ERS '/'ERSR'/'ZRS '/'ZRSR'	−	○
	TAM テーブルからレコードの入力	'FxxR'/'FxxU'/'VxxR'/'VxxU'	−	○
	TAM テーブルの状態の取得	'GST '	○	○
	TAM テーブルの情報の取得	'INFO'	○	○
	TAM テーブルのレコードの更新/追加	'MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR '	−	○
トランザクション制御	トランザクションの開始	'BEGIN '	○M	−
	現在のトランザクションに関する情報の出力	'INFO '	○	○
	非連鎖モードのコミット	'U-COMMIT'	−	○M
	非連鎖モードのロールバック	'U-ROLL '	−	○M
	リソースマネジャ接続先選択	'RMSELECT'	−	−
オンラインテスタの管理	ユーザサーバの状態の報告	'T-STATUS'	○	○

(凡例)

○：該当する条件で使えます。

(○)：回復対象外の DAM ファイルへアクセスするときだけ，使えます。

○M：メインプログラムからだけ，使えます。

○NO：非トランザクション属性の MHP のサービスプログラムの範囲でだけ，使えます。

−：該当する条件では使えません。

注

「トランザクションの処理の範囲でない」とは，非トランザクション属性の MHP，または MHP のメインプログラムの範囲を示します。

(4)　オフラインの業務をするUAPで使える機能とプログラム

オフラインの業務をするUAPで使える機能と要求コードを次の表に示します。

表1-5　オフラインの業務をするUAPで使える機能と要求コード

オフラインの業務をするUAPで使える機能	オフラインの業務をするUAPで使える機能名と，COBOL-UAP作成用プログラムの先頭のデータ名に指定する要求コード	
DAMファイルサービス	物理ファイルのブロックの検索	'DCDAMINT','BSEK'
	物理ファイルのクローズ	'DCDAMINT','CLOS'
	物理ファイルの割り当て	'DCDAMINT','CRAT'
	物理ファイルからブロックの直接入力	'DCDAMINT','DGET'
	物理ファイルへブロックの直接出力	'DCDAMINT','DPUT'
	物理ファイルからブロックの入力	'DCDAMINT','GET '
	物理ファイルのオープン	'DCDAMINT','OPEN'
	物理ファイルへブロックの出力	'DCDAMINT','PUT '
ジャーナルデータの編集	jnlrput出力ファイルのクローズ	'CLOSERPT'
	jnlrput出力ファイルのオープン	'OPENRPT '
	jnlrput出力ファイルからジャーナルデータの入力	'RDGETRPT'
性能検証用トレース	性能検証用トレース取得通番の通知	'PRFGETN '
	ユーザ固有の性能検証用トレースの取得	'PRFPUT '

1.1.2　コーディング規約

(1)　コーディング上の注意

OpenTP1で使うUAPをCOBOL言語でコーディングするときは，COBOL85またはCOBOL2002の様式に従ってください。OpenTP1の機能を使うときは，OpenTP1のライブラリにあるCOBOL-UAP作成用プログラムを使います。

さらに，システムコールや任意のプログラムのライブラリも使えますが，コーディング時には，UAPの移植性を高めるため，OSで標準的に提供している命令文やシステムコールを使うことをお勧めします。

システムコールや任意のプログラムのライブラリを使う UAP を COBOL 言語で作成するときは，次のことに注意してください。

1. UAP でシグナルを使うときは，SIGILL や SIGBUS などシグナルのデフォルト動作がコアファイルを作成するタイプのシグナルハンドラは登録しないでください。これらのシグナルハンドラを登録すると，プログラムが異常終了してもコアファイルが作成されないため，トラブルシュートができない場合があります。
2. UAP でシグナルを使うときは，シグナルハンドラからは OpenTP1 のライブラリにある関数に該当する COBOL-UAP 作成用プログラムは使わないでください。
3. 次のシステムコールは使わないでください。
   - chdir（カレントワーキングディレクトリを変更）
4. CBLDCRPC('OPEN ')を呼び出したあとには，次のシステムコールは使わないでください。
   - fork（新プロセスの生成）
   - exec（ファイルの実行）
   - system（シェルコマンドの発行）
5. サービスプログラム間にわたるジャンプ（GOTO 文など）は使わないでください。
6. ほかのプログラムのライブラリを使うときは，Xlib や OSF/Motif など，イベント駆動型のディスパッチング制御をするプログラムは使えません。
7. COBOL 言語で UAP を作成する場合，一意名 1 などのデータ領域は，必ず偶数番地から始まるように設定してください。一意名 1 などのデータ領域の先頭アドレスが奇数番地だとバスエラーが発生します。

   例えば，CBLDCMCF で使用する一意名 3 を配列で定義し，送信データが奇数バイトで，かつ構造体に"SYNC"の指定がない（境界調整されていない）場合，配列の 2 件目のデータを CBLDCMCF の引数に設定すると，関数処理中にバスエラーが発生します。
8. COBOL 言語で作成した C 系および V 系 MCF イベント用 MHP で受け取る MCF イベント情報領域には，C 言語用の MCF イベント情報の予備領域長に加えて 2 以上の予備領域を設定してください。
9. データ部（DATA DIVISION）の 2 進形式のデータ項目に PICTURE 句で指定したけた数を超える値を指定する場合（CBLDCRTS('RTSPUT ')の一意名 2 にデータ名 C PIC S9（9） COMP VALUE2147483647 を指定するなど），コンパイル時にエラーが発生します。この場合，次のどちらかで対処してください。
   - データ部（DATA DIVISION）では値を指定しないで，手続き部（PROCEDURE DIVISION）で MOVE 文などによって値を指定するようにコーディングする。
   - 2 進形式のデータ項目に指定できる初期値を拡張するコンパイルオプション（COBOL85 の場合：-Vx，COBOL2002 の場合：-BinExtend）を指定して，コンパイルする。
10. CBLDCRPC('OPEN ')を呼び出したあとには，UAP プロセスでオープンするファイルディスクリプタの最大数を変更しないでください。変更した場合の動作は保証できません。

OS が HP-UX の場合，リンケージ時のバインドモードには必ず "immediate"を指定してください。"immediate"以外のバインドモードで作成した実行形式ファイルを OpenTP1 の UAP として使った場合，

システムの動作は保証しません。作成した UAP のバインドモードが"immediate"かどうかは，OS の chatr コマンドで確認してください。

(2) 名称の付け方の注意

ユーザがコーディングする変数名や定義名などは，先頭に何文字かのプレフィックスを付加することをお勧めします。OS や OpenTP1 などと名称が重複した場合の動作は保証しません。

(a) サービスプログラム名，プログラム名，入り口名

サービスプログラムの名称は，20 文字以内の長さで，先頭が英字で始まる英数字を付けます。サービスプログラム名，プログラム名，入り口名には，次の名称は使わないでください。

- "dc" で始まる名称
- "CBLDC" で始まる名称
- "tx" ，または "TX" で始まる名称
- "tp" ，または "TP" で始まる名称

(b) 外部変数名

外部変数名には，次の名称は使わないでください。ただし，このマニュアルの記述に従って使う場合を除きます。

- "dc" で始まる名称
- "CBLDC" で始まる名称
- "tx" ，または "TX" で始まる名称
- "tp" ，または "TP" で始まる名称

(3) 終了のしかたについて

OpenTP1 から直接起動されるプログラム（**メインプログラム**）は，STOP　RUN で終了して，**サービスプログラム**は，EXIT　PROGRAM で終了してください。メインプログラムを STOP　RUN を出さないで終了すると，COBOL のカウント情報などが出力されません。

(4) Windows を使う場合

OpenTP1（TP1/LiNK）を Windows で使う場合に，UAP をコンパイル，リンケージするときは，Windows で使う COBOL コンパイラの仕様に従ってください。

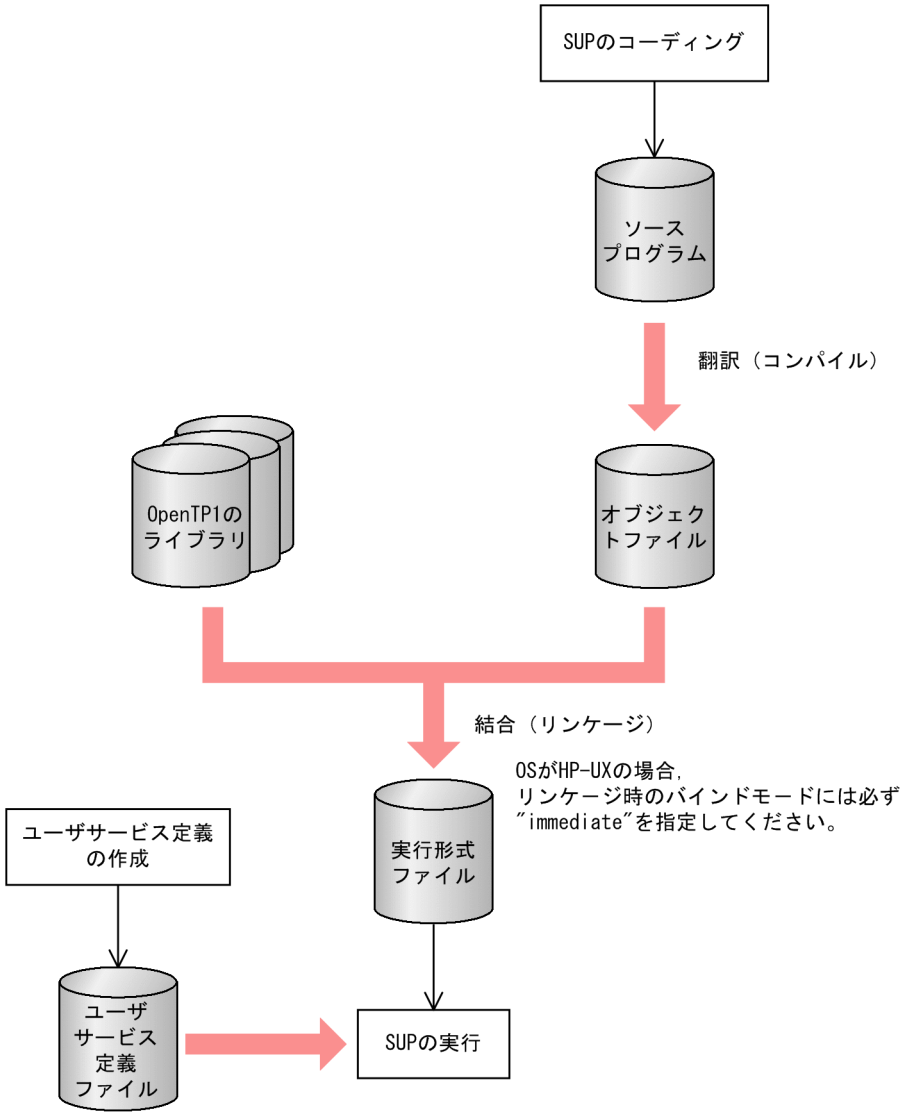
1.2　アプリケーションプログラムの作成（TCP/IP 通信）

1.2.1　アプリケーションプログラムの作成手順

(1)　SUP の作成手順

SUP の作成手順を次の図に示します。

図 1-1　SUP の作成手順



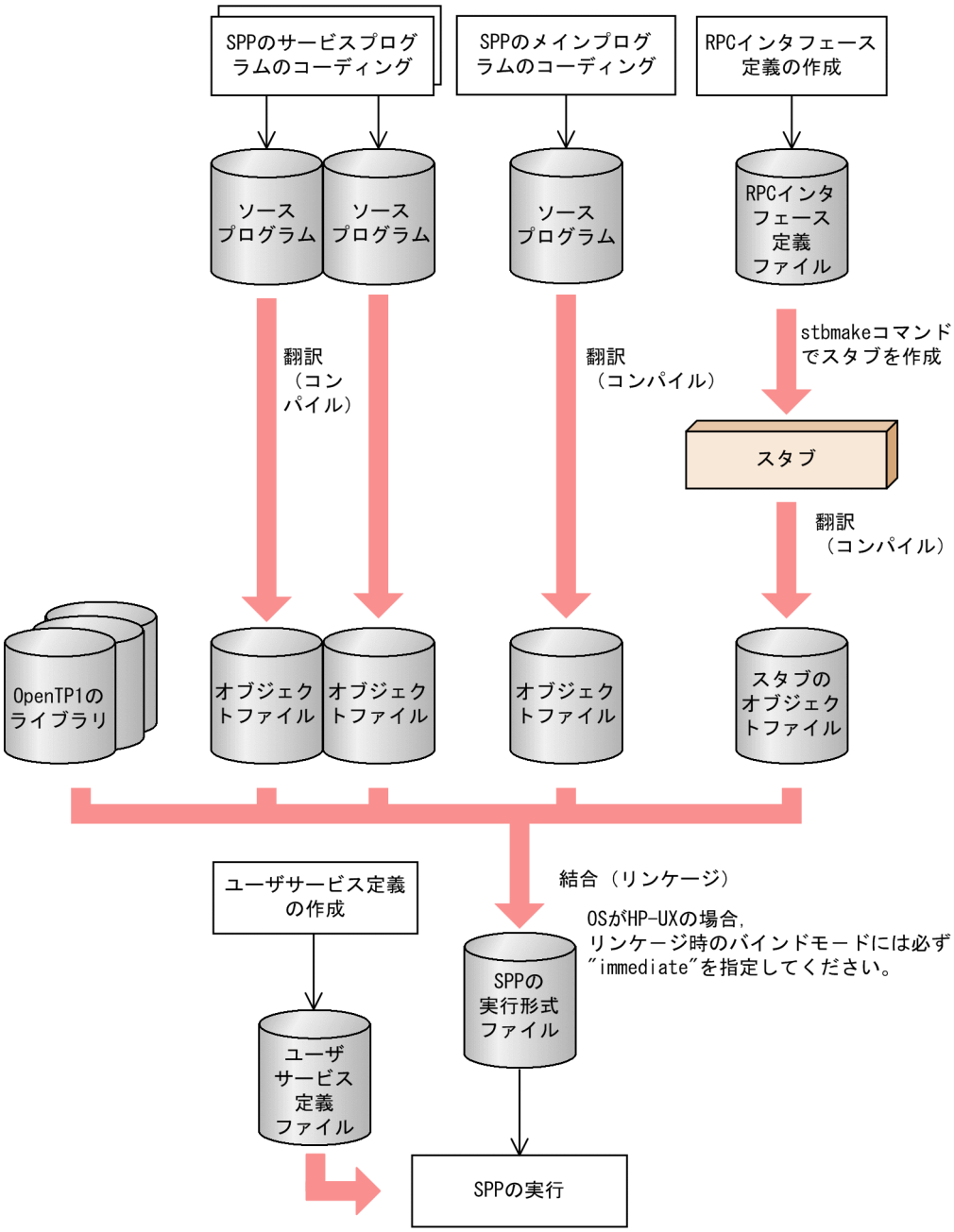
(2)　SPP の作成手順

SPP の作成手順は，「スタブを使用する SPP」か「サービス関数動的ローディング機能を使用する SPP」かで異なります。

(a)　SPP の作成手順（スタブ使用時）

スタブを使用する SPP の作成手順を，次の図に示します。

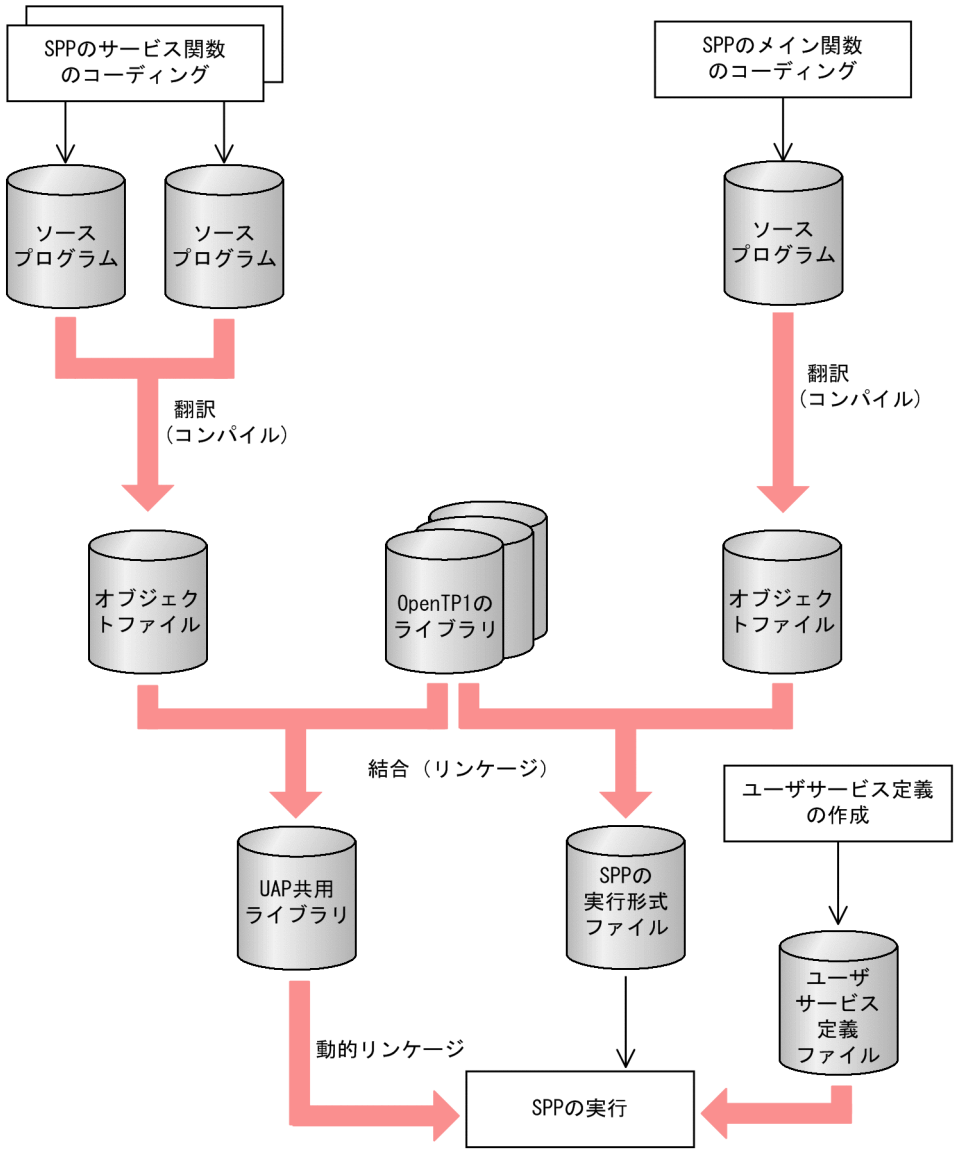
図 1-2　SPP の作成手順（スタブ使用時）



(b)　SPP の作成手順（サービス関数動的ローディング機能使用時）

サービス関数動的ローディング機能を使用する SPP の作成手順を，次の図に示します。

図 1-3　SPP の作成手順（サービス関数動的ローディング機能使用時）



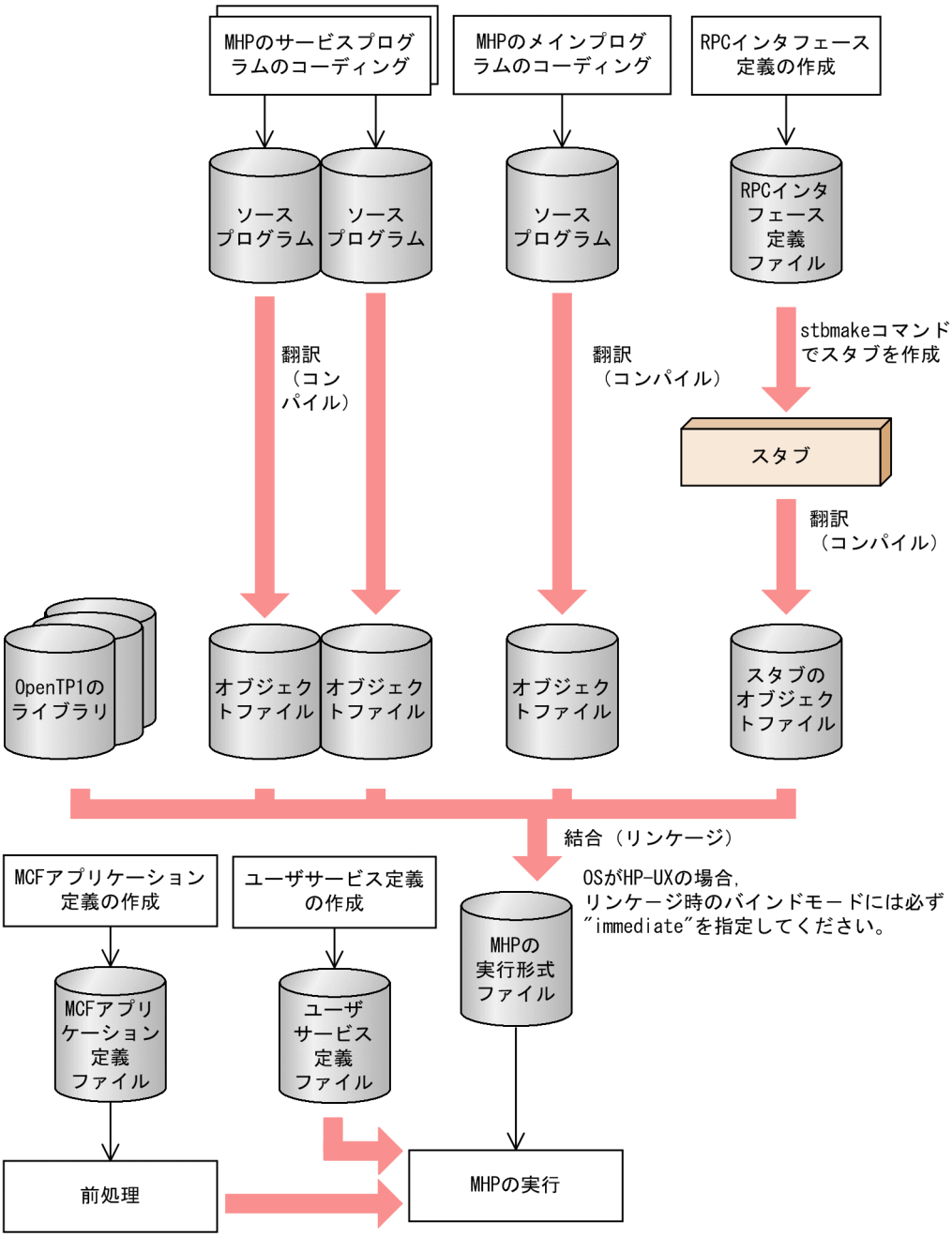
(3)　MHP の作成手順

MHP の作成手順は，「スタブを使用する MHP」か「サービス関数動的ローディング機能を使用する MHP」かで異なります。

(a)　MHP の作成手順（スタブ使用時）

スタブを使用する MHP の作成手順を，次の図に示します。

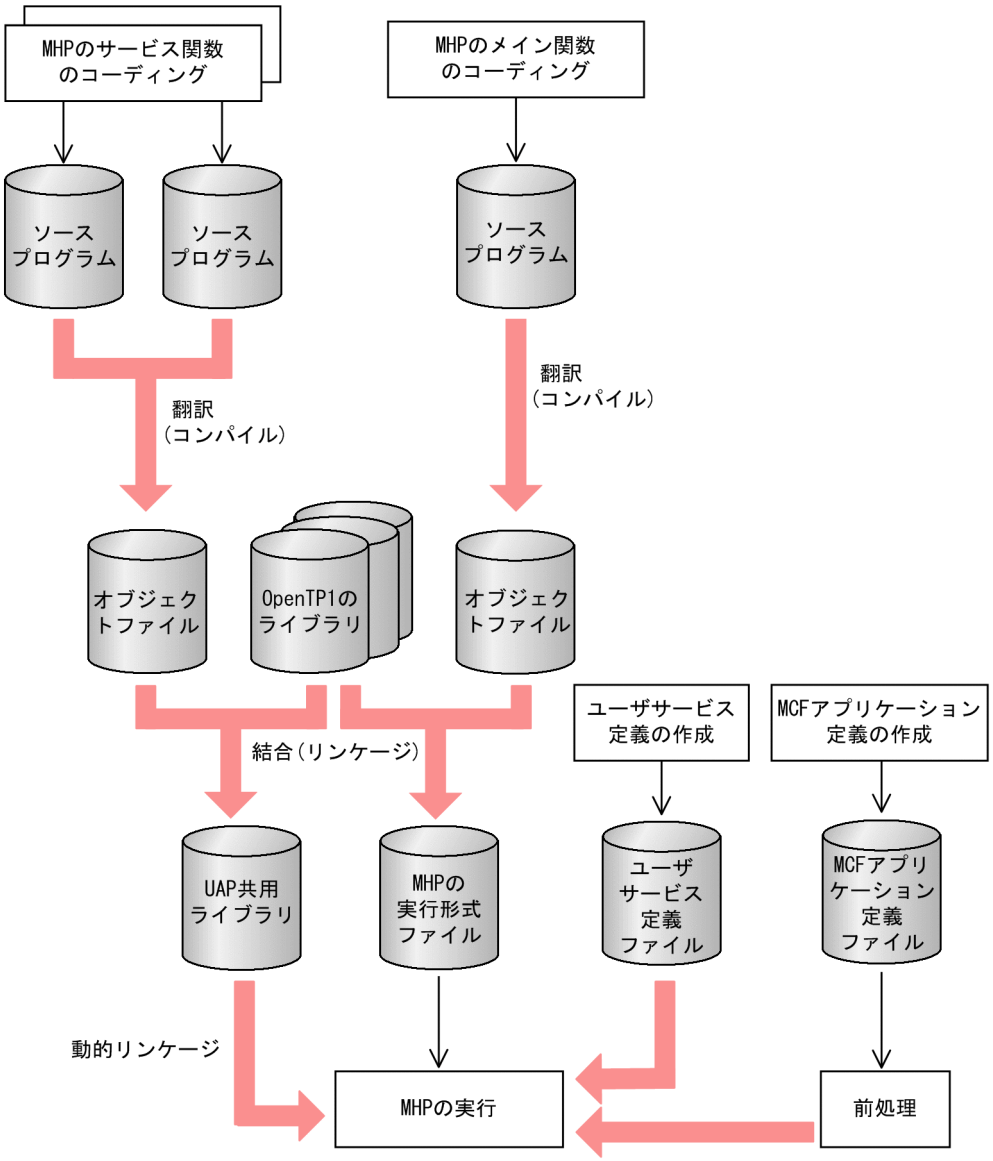
図 1-4　MHP の作成手順（スタブ使用時）



(b)　MHP の作成手順（サービス関数動的ローディング機能使用時）

サービス関数動的ローディング機能を使用する MHP の作成手順を，次の図に示します。

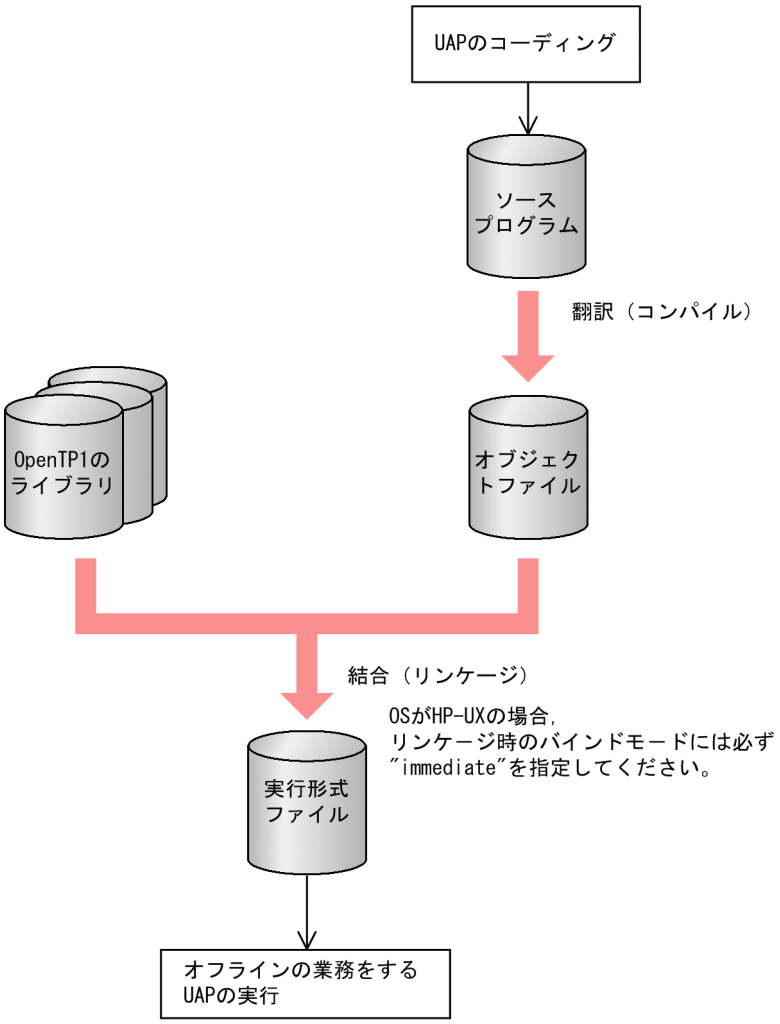
図 1-5　MHP の作成手順（サービス関数動的ローディング機能使用時）



(4)　オフラインの業務をする UAP の作成手順

オフラインの業務をする UAP の作成手順を次の図に示します。

図 1-6　オフラインの業務をする UAP の作成手順



1.2.2　スタブの作成方法

OpenTP1 で使う UAP には，UAP 間のサービス要求をするためのライブラリが必要となります。このライブラリを**スタブ**といいます。

(1)　スタブが必要になるアプリケーションプログラム

OpenTP1 で使う UAP のうち，**サービスプログラムを持つ UAP（SPP，MHP）**には，スタブが必要です。ただし，すべてのサービス関数を **UAP 共用ライブラリ**化してサービス関数動的ローディング機能を使う場合は，スタブは不要です。UAP 共用ライブラリ化とは，UAP のソースファイルを翻訳（コンパイル）して作成した UAP オブジェクトファイルを結合（リンケージ）して，共用ライブラリとしてまとめることです。

また，SUP とオフラインの業務をする UAP は，サービス関数がないので，作成する必要はありません。

(2)　スタブの作成手順

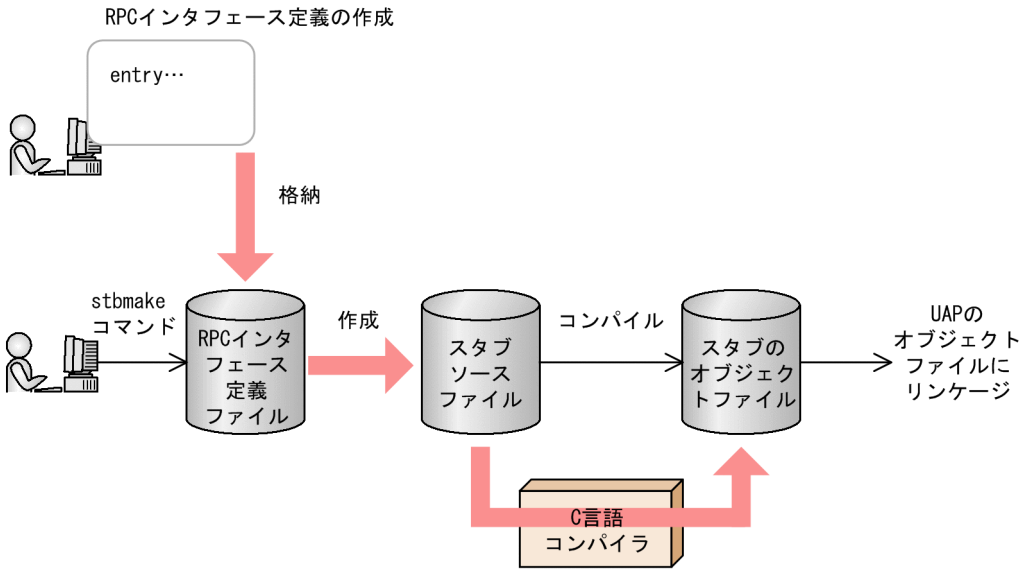
スタブを作成するときは，まず，UAP のサービスプログラムのプログラム ID を定義したファイル（**RPC インタフェース定義ファイル**）を作成します。そして，そのファイルを引数にして **stbmake コマンド**を実行します。

stbmake コマンドを実行すると，**スタブのソースファイル**（C 言語のソースファイル）が作成されます。このファイルを C 言語のコンパイラで翻訳して，UAP のオブジェクトファイルに結合させます。

スタブの内容を変更するときは，UAP を作成する一連の作業をやり直します。RPC インタフェース定義ファイルの内容を変更して，スタブを作り直してから，コンパイルし直した UAP のオブジェクトファイルに結合させてください。

スタブの作成手順を次の図に示します。

図 1-7　スタブの作成手順



(3)　RPC インタフェース定義ファイルの作成

スタブを作成するには，SPP，または MHP のサービスプログラムのプログラム ID（入り口点）を定義したファイルを作成します。定義内容を **RPC インタフェース定義**といい，定義を格納するファイルを **RPC インタフェース定義ファイル**といいます。

RPC インタフェース定義ファイルは，SPP，または MHP の一つの実行形式ファイルごとに作成します。

(a)　RPC インタフェース定義の形式

RPC インタフェース定義は，次のように記述します。

形式

```
entry "プログラムID"〔"プログラムID"…〕;
```

機能

SPP，またはMHPのサービスプログラムのプログラムID（入り口点）を指定します。プログラムIDとして，COBOL言語のプログラムIDを指定します。プログラムID（入り口点）は20文字以内としてください。

プログラムIDとサービス名の対応は，ユーザサービス定義で指定している名称と合わせてください。

RPCインタフェース定義に注釈文を記述するときは，"/*"で始めて，"*/"で終わらせてください。注釈文のネストはできません。また，注釈文はキーワードや識別子など，文字列の中には記述できません。

entryの文は，一つのファイルに複数行にわたって定義できます。RPCインタフェース定義の作成例を次に示します。

使用例

プログラムIDが「sv01」と「sv02」のサービスプログラムがあるUAPのRPCインタフェースの指定（次に示すどちらかの形式で指定します）

```
形式1
 entry   ”sv01” ;
 entry   ”sv02” ;

形式2
 entry   ”sv01”  ”sv02” ;
```

(4) RPCインタフェース定義ファイルの名称

ファイル名には，RPCインタフェース定義ファイルを示すサフィックス".def" を必ず付けてください。RPCインタフェース定義ファイルを格納するディレクトリは，stbmakeコマンドが探せるパスであれば，特に制限はありません。

RPCインタフェース定義ファイルのファイル名の長さは，最大255文字です。ただし，OSの制限で255文字まで指定できないことがあります。

stbmakeコマンドを実行したあと，スタブのソースファイルはRPCインタフェース定義ファイルとは別の名称で作成されます。そのため，OpenTP1の稼働中にはRPCインタフェース定義ファイルは使われません。

1.2.3 スタブのソースファイルの作成

スタブのソースファイルを作成するときは，RPCインタフェース定義ファイル名を引数に設定した，stbmakeコマンドを実行します。

(1) stbmakeコマンドで作成されるファイル

stbmakeコマンドを実行すると，次のファイルが作成されます（xxxxxは，RPCインタフェース定義ファイルのサフィックス".def"を除いた名称です）。

- スタブのソースファイル（ファイル名：xxxxx_sstb.c）

ソースファイルのディレクトリ名，およびファイル名は，コマンドのオプションで変更できます。

ソースファイルのファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

ソースファイルとして作成するファイル名は，コマンドのオプションで変更できます。スタブのソースファイルは，C 言語のコンパイラで翻訳して，UAP のオブジェクトファイルに結合させます。

1.2.4　stbmake（スタブのソースファイルの作成）

(1)　形式

```
stbmake 〔 -s〔スタブソースファイル名〕〕 定義ファイル名
```

(2)　機能

RPC インタフェース定義ファイルから，スタブのソースファイルを作成します。

OpenTP1 のリモートプロシジャコールと XATMI インタフェースの両方を使う UAP を作成する場合は，stbmake コマンドについて「付録 A　OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合」を参照してください。

(3)　オプション

●-s スタブソースファイル名　～〈パス名〉

作成するスタブのソースファイル名を，パス名で指定します。

フラグ引数を省略した場合は，スタブのソースファイル名は，RPC インタフェース定義ファイルのサフィックス ".def" が "_sstb.c" に置き変わった名称でカレントディレクトリに作成されます。

指定したソースファイル名がすでにある場合は，上書きされて元のファイルの内容はなくなります。

(4)　コマンド引数

●定義ファイル名　～〈パス名〉

RPC インタフェース定義ファイルの名称を，パス名で指定します。

(5) 注意事項

stbmake コマンドで入出力できるファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

(6) 使用例

stbmake コマンドの使用例を次に示します。

（例）カレントディレクトリの RPC インタフェース定義ファイル"test.def"から，スタブのソースファイルを作成する場合

形式 1

```
stbmake  test.def
```

カレントディレクトリの RPC インタフェース定義ファイル"test.def "から，スタブのソースファイル" test_sstb.c"が作成されます。

形式 2

```
stbmake  -s  stub/test.c  test.def
```

カレントディレクトリの下にディレクトリ"stub"が作成されて，その下にスタブのソースファイル"test.c" が作成されます。

1.2.5 アプリケーションプログラムの翻訳と結合

UAP の翻訳と結合方法については，使用する OS のリファレンスマニュアルを参照してください。

- UAP 作成時の注意

  UAP を作成するときは，OpenTP1 のバージョンに気を付けてください。システムサービスによっては，古いバージョンの UAP からの命令文を受け付けないことがあります。旧バージョンで作成した UAP を使用する場合，現在使用しているバージョンの OpenTP1 で，コンパイル/リンケージし直すことをお勧めします。

(1) プログラムの翻訳（コンパイル）

COBOL 言語の UAP のオブジェクトファイルを作成するには，ソースプログラムを COBOL コンパイラで翻訳します。

翻訳方法の詳細については，COBOL 言語のマニュアルを参照してください。

(2) スタブの翻訳（コンパイル）

スタブのオブジェクトファイルを作成するには，スタブのソースプログラムを C コンパイラで翻訳します。

(3) 結合（リンケージ）

ここでの説明中の※1～※3 の意味を次に示します。

注※1

リソースマネジャに XA インタフェースでアクセスするトランザクションを実行する場合に必要です（OpenTP1 で提供するリソースマネジャは，すべて XA インタフェースでアクセスします）。トランザクション制御用オブジェクトファイルは，OpenTP1 のコマンド（trnmkobj コマンド）で作成します。trnmkobj コマンド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。

注※2

リソースマネジャにアクセスする場合に必要です。OpenTP1 で提供するリソースマネジャのオブジェクトファイルを結合するときは，リンケージのコマンドに次の引数を指定します。

メッセージ送受信機能を使う場合：-lmcf，-lmnet

DAM アクセス機能を使う場合：-ldam

TAM アクセス機能を使う場合：-ltam

ISAM 機能を使う場合：-lismb，-lisam，-lrsort

メッセージキューイング機能を使う場合：-lmqacb

他社のリソースマネジャのオブジェクトファイルの結合方法については，使用する他社リソースマネジャのリファレンスマニュアルを参照してください。

注※3

CBLDCUTO('T-STATUS')（ユーザサーバのテスト状態の報告）を使う場合に必要です。オンラインテスタのオブジェクトファイルを結合するときは，リンケージのコマンドに次の引数を指定します。

ユーザサーバのテスト状態を報告する場合：-luto

(a) SPP，MHP に結合させるファイル

SPP または MHP の実行形式ファイルは，次に示すファイルを結合させて作成します。

- **UAP のオブジェクトファイル**（メインプログラムとサービスプログラム）
- **スタブのオブジェクトファイル**
- **トランザクション制御用オブジェクトファイル**※1
- **リソースマネジャで提供するオブジェクトファイル**※2
- **オンラインテスタで提供するオブジェクトファイル**※3
- **OpenTP1 のライブラリ**
- **COBOL のライブラリ**（COBOL85 言語で作成した UAP の場合は COBOL85 のライブラリ，COBOL2002 言語で作成した UAP の場合は COBOL2002 のライブラリ）

(b) SUP に結合させるファイル

SUP の実行形式ファイルは，次に示すファイルを結合させて作成します。

- UAP のオブジェクトファイル（メインプログラム）
- トランザクション制御用オブジェクトファイル※1
- リソースマネジャで提供するオブジェクトファイル※2
- オンラインテスタで提供するオブジェクトファイル※3
- OpenTP1 のライブラリ
- COBOL のライブラリ（COBOL85 言語で作成した UAP の場合は COBOL85 のライブラリ，COBOL2002 言語で作成した UAP の場合は COBOL2002 のライブラリ）

(c) オフラインの業務をする UAP に結合させるファイル

オフラインの業務をする UAP の実行形式ファイルは，次に示すファイルを結合させて作成します。

- UAP のオブジェクトファイル（メインプログラム）
- OpenTP1 のライブラリ
- COBOL のライブラリ（COBOL85 言語で作成した UAP の場合は COBOL85 のライブラリ，COBOL2002 言語で作成した UAP の場合は COBOL2002 のライブラリ）

(d) サービス関数動的ローディング機能を使用する SPP，MHP に結合させるファイル

サービス関数動的ローディング機能を使用する SPP，MHP の実行形式ファイルは，次に示すファイルを結合させて作成します。

- UAP のオブジェクトファイル（メイン関数）
- OpenTP1 のライブラリ
- トランザクション制御用オブジェクトファイル※1
- リソースマネジャで提供するオブジェクトファイル※2
- オンラインテスタで提供するオブジェクトファイル※3

また，サービス関数動的ローディング機能とスタブでのサービス検索を併用する場合は，次のファイルが必要です。

- UAP のオブジェクトファイル（サービス関数）
- スタブのオブジェクトファイル

(4) 注意事項

OS が HP-UX の場合，リンケージ時のバインドモードには必ず"immediate"を指定してください。"immediate"以外のバインドモードで作成した実行形式ファイルを OpenTP1 の UAP として使った場合，システムの動作は保証しません。作成した UAP のバインドモードが"immediate"かどうかは，OS の chatr コマンドで確認してください。

1.3 XATMI インタフェースを使うアプリケーションプログラムの作成（TCP/IP 通信，OSI TP 通信）

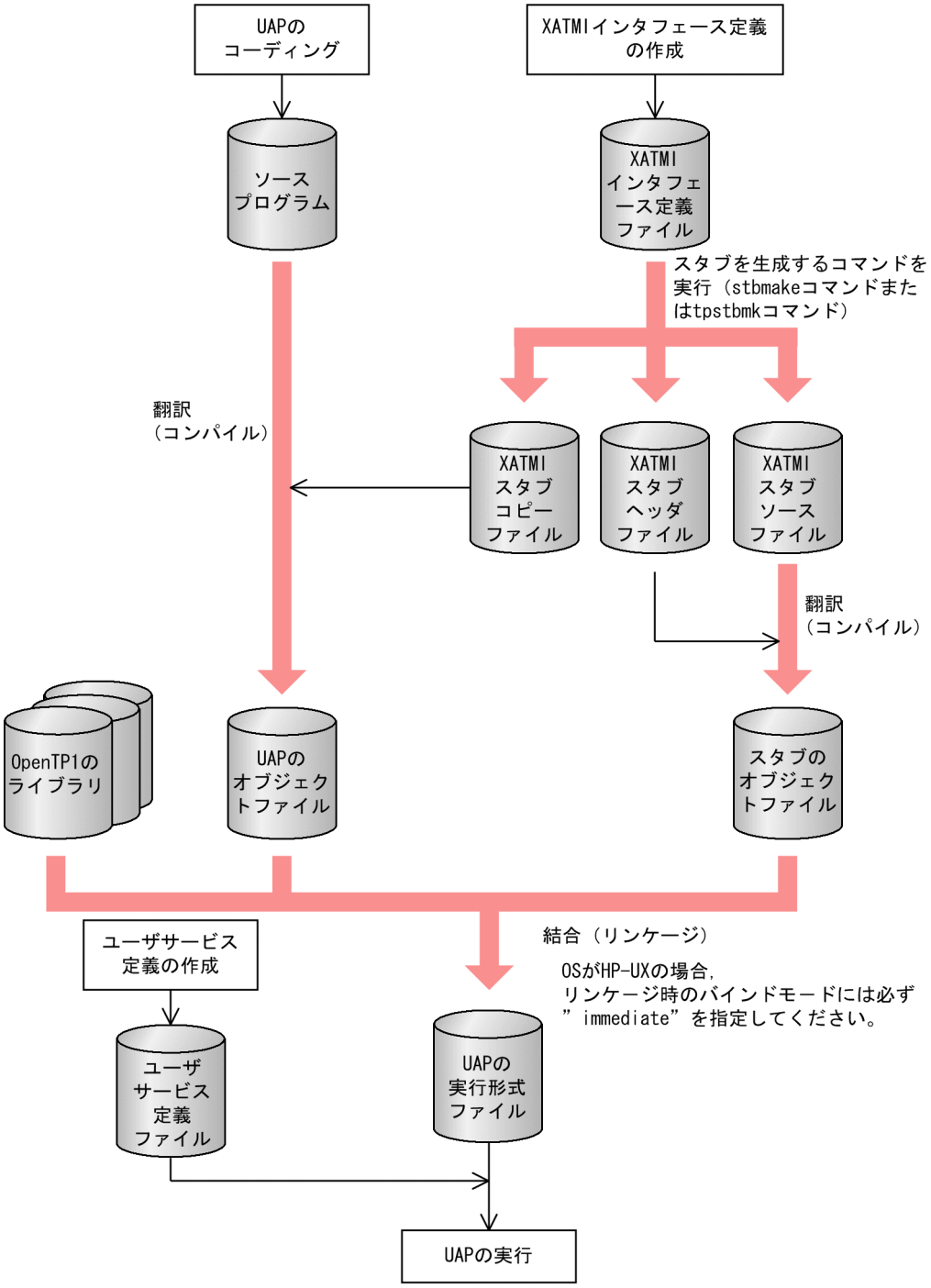
通信プロトコルに TCP/IP および OSI TP を使う場合の，XATMI インタフェースを使う UAP の作成方法について説明します。

OpenTP1 の RPC を使う UAP の作成方法とは，スタブの作成方法（stbmake コマンド，tpstbmk コマンドの実行形式）と UAP にリンケージするファイルが異なります。そのほかの作成手順は，OpenTP1 の UAP と同じです。UAP の作成手順については，「1.1　アプリケーションプログラムのコーディング」および「1.4　アプリケーションプログラムの実行」を参照してください。

1.3.1　アプリケーションプログラムの作成手順

XATMI インタフェースを使った UAP の作成手順を次の図に示します。

図 1-8　アプリケーションプログラムの作成手順（XATMI インタフェース TCP IP 通信，OSI TP 通信）



1.3.2　XATMI インタフェース用スタブの作成方法

XATMI インタフェース用のスタブの作成方法について説明します。XATMI インタフェースの通信をする UAP の場合は，**クライアント UAP とサーバ UAP の両方**に，スタブが必要です。

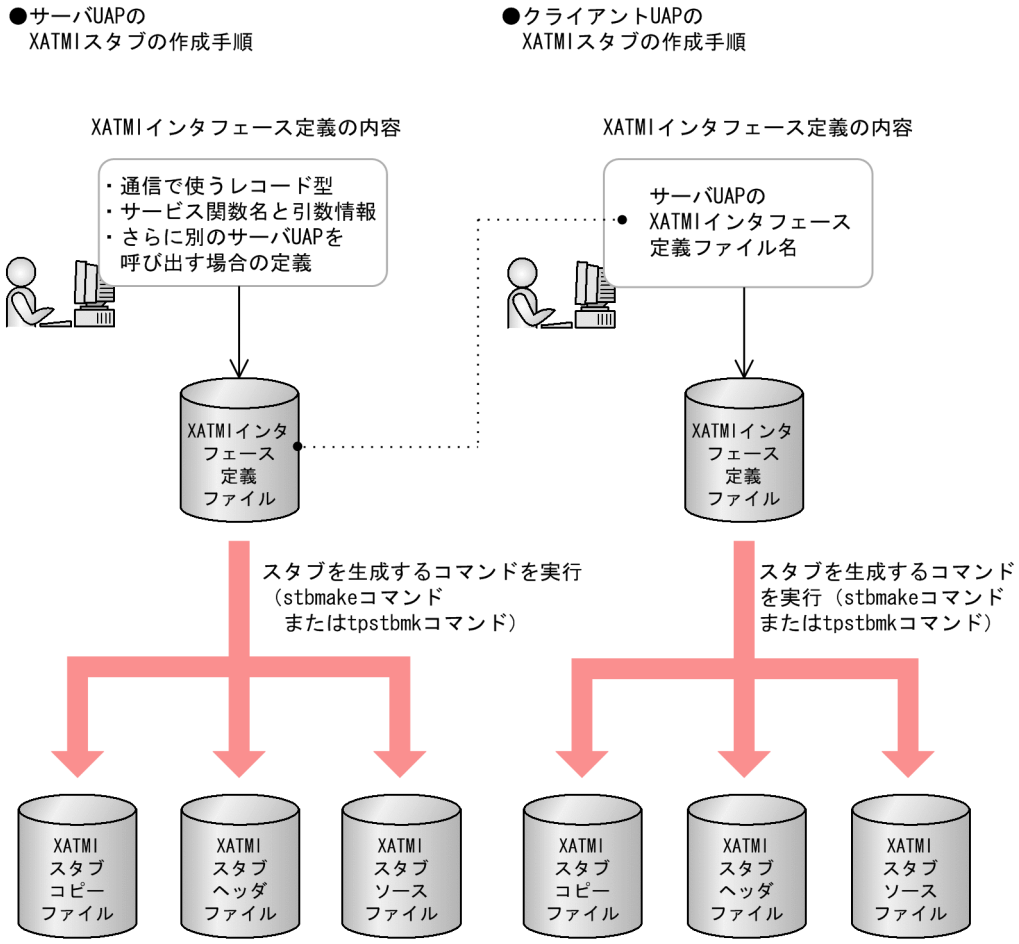
スタブを作成するときは，XATMI インタフェース定義を格納したファイル（**XATMI インタフェース定義ファイル**）を作成して，スタブを生成するコマンドを実行します。スタブを生成するコマンドを，次に示します。

- TCP/IP 通信をする UAP の場合：**stbmake コマンド**
- OSI TP 通信をする UAP の場合：**tpstbmk コマンド**

作成したスタブのソースファイルは，C 言語のコンパイラで翻訳して，UAP のオブジェクトファイルに結合させます。

XATMI インタフェース用スタブの作成手順を次の図に示します。

図 1-9　XATMI インタフェース用スタブの作成手順（TCP/IP 通信，OSI TP 通信）



(1)　XATMI インタフェース定義（クライアント UAP 用）

クライアント UAP（SUP，または SPP）用の XATMI インタフェース定義の形式について説明します。

形式

```
called_servers = { ”サーバの定義ファイル名”
                        〔,”サーバの定義ファイル名”〕…};
```

機能

サーバ UAP の XATMI インタフェース定義のファイル名を，すべて指定します。サーバ UAP の XATMI インタフェース定義ファイル名を指定することで，サーバ UAP で定義されているレコード型は，クライアント UAP でも使えるようになります。

パラメタ

- サーバの定義ファイル名

  サーバ UAP の XATMI インタフェース定義ファイルのファイル名を指定します。定義ファイル名は，サフィックスが ”.def” のファイルです。

  一つの called_servers 文の括弧 ｛ ｝ の中に，複数の定義ファイル名を指定できます。また，一つの XATMI インタフェース定義ファイルに複数の called_servers 文を記述することもできます。

指定例

サーバ UAP1 とサーバ UAP2 と XATMI インタフェースの通信をするクライアント UAP の定義（サーバ UAP1 の定義ファイル名を serv1.def，サーバ UAP2 の定義ファイル名を serv2.def とします）。

形式 1

```
called_servers = { ”serv1.def”,”serv2.def”};
```

形式 2

```
called_servers = { ”serv1.def”};
called_servers = { ”serv2.def”};
```

(2) XATMI インタフェース定義（サーバ UAP 用）

サーバ UAP の XATMI インタフェース定義に指定する項目を次に示します。指定は順不同です。

- 通信で使うレコード型の定義
- サービスプログラム名とレコード型の定義
- called_servers 文（サーバ UAP から，さらに別のサーバ UAP を呼び出す場合）

(a) 通信で使うレコード型の定義

形式

```
タイプ名 サブタイプ名{
          データ型 データ名;
        〔データ型 データ名;〕
                :
                :
      };
```

機能

サーバ UAP で使うレコード型のタイプ名，サブタイプ名，およびレコードを定義します。サーバ UAP から ほかのサーバ UAP プロセスのサービスを呼び出す場合，呼び出すプロセスで使えるレコード型は

すべて自プロセスでも使えます。そのため，ここでは自プロセス内のサービスプログラムが入出力として使うレコード型だけを定義します。

X_OCTET は定義しなくても常に認識されます。X_OCTET を定義した場合，スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）の実行時にエラーとなります。

X_C_TYPE は COBOL 言語の API では使えません。X_C_TYPE を定義した場合，スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）に -b オプションを付けて実行するとエラーとなります。

パラメタ

- タイプ名

  サーバ UAP で使うレコード型の，タイプ名を指定します。

- サブタイプ名

  サーバ UAP で使うレコード型の，サブタイプ名を指定します。

- データ型

  サーバ UAP で使うレコード型の，データ型を指定します。

- データ名

  サーバ UAP で使うレコード型の，データ名を指定します。

タイプで使えるデータ型の一覧

タイプで使えるデータ型の一覧を次の表に示します。識別子とは XATMI インタフェース定義に記述するデータ型を示し，COBOL 言語のデータとは実際にスタブに定義される型付きレコードを示します。OpenTP1 以外のシステムと通信するためにデータ型を変換する場合は，変換する識別子を XATMI インタフェース定義に指定します。

表 1-6　タイプで使えるデータ型の一覧

タイプ	識別子	COBOL 言語のデータ	通信プロトコル		備考
			TCP/IP	OSI TP	
X_OCTET	―※1	―※1	○	○	なし
X_COMMON	short a	PIC S9(9) COMP-5	○	○	なし
	short a[n]	PIC S9(9) COMP-5 OCCURS n TIMES	○	○	なし
	long a	PIC S9(9) COMP-5	○	○	なし
	long a[n]	PIC S9(9) COMP-5 OCCURS n TIMES	○	○	なし
	char a※2	PIC X	○	○	無変換配列
	octet a	PIC X	○	○	無変換配列
	tchar a	PIC X	－	○	変換配列
	char a[n]※2	PIC X(n)	○	○	無変換配列

タイプ	識別子	COBOL 言語のデータ	通信プロトコル		備考
			TCP/IP	OSI TP	
X_COMMON	octet a[n]	PIC X(n)	○	○	無変換配列
	tchar a[n]	PIC X(n)	−	○	変換配列
X_C_TYPE	＿※3	＿※3	×	×	なし

（凡例）

○：該当する通信プロトコルで使えます。

×：該当する通信プロトコルでは使えません。

−：変換の識別子でも，無変換としてそのまま処理されます。

注※1

X_OCTET は，定義しなくても自動的に認識されます。XATMI インタフェース定義に X_OCTET を指定した場合は，スタブを生成するコマンドを実行したときにエラーになります。

注※2

この識別子も使えますが，新規で作成する場合は次に示す識別子を使うことをお勧めします。

X_COMMON の場合：octet または tchar

X_C_TYPE の場合：str または tstr

注※3

X_C_TYPE は，COBOL 言語の API では使えません。X_C_TYPE を定義した場合，スタブを生成するコマンドに-b オプションを指定して実行するとエラーになります。

指定例

```
X_COMMON subtype1 {
                   char  name[8];
                   long  data[10];
                   long  flags;
                   };
```

(b) サービスプログラム名とレコード型の定義

形式

```
service サービスプログラム名{
                (タイプ名 〔サブタイプ名〕)|(ALL)|( 〔void〕 )};
```

機能

サーバ UAP にある，サービスプログラムのプログラム名と，渡されるレコード型のタイプ名とサブタイプ名を指定します。

X_OCTET の場合はサブタイプ名がないので，タイプ名だけを指定します。また，サービスプログラム内でレコード型を一度も使わない処理の場合は，サービスプログラムのあとに何も指定しないか，void を指定します。

TPCALL，TPACALL，TPCONNECT は，型付きレコードを送信しないで，サービスプログラムを呼び出せます。ただし，サービスプログラムでレコード型を明示的に参照したくない場合は，データ領域に何も設定しないか，空白を設定します。このように指定したサービスプログラムを呼び出すには，クライアント側の TPCALL，TPACALL，TPCONNECT が送信する型付きレコードに，空白を設定してください。ただし，X_OCTET の場合は，データ領域が空白でなくても，送信データの長さが 0 の場合でも，サービスを要求できます。

データ領域として受け取るレコード型を限定しない指定をする場合は，ALL を指定します。ALL を指定して定義したサービスプログラムは，自プロセスで認識できるレコード型であれば，どの型でも受信できます。

パラメタ

- サービスプログラム名

  サーバ UAP にある，プログラム名を指定します。

- タイプ名

  API のデータ領域に指定した，タイプ名を指定します。

- サブタイプ名

  API のデータ領域に指定した，サブタイプ名を指定します。

指定例

例 1

```
service svc_func1(X_COMMON subtype1);
```

例 2（タイプ名が X_OCTET の場合）

```
service svc_func2(X_OCTET);
```

例 3（型付きレコードを受信しないサービスプログラムの場合）

```
service svc_func3(void);  または  service svc_func3();
```

例 4（レコード型を限定しないサービスプログラムの場合）

```
service svc_func4(ALL);
```

(c) サーバ UAP から，さらに別のサーバ UAP を呼び出す場合

クライアント UAP の XATMI インタフェース定義（called_servers 文）を指定します。

(3) XATMI インタフェース定義ファイルの名称

ファイル名には，XATMI インタフェース定義ファイルを示すサフィックス ".def" を必ず付けてください。XATMI インタフェース定義ファイルを格納するディレクトリは，スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）が探せるパスであれば，特に制限はありません。

XATMI インタフェース定義ファイルのファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）を実行したあと，スタブのソースファイルは XATMI インタフェース定義ファイルとは別の名称で作成されます。そのため，OpenTP1 の稼働中には XATMI インタフェース定義ファイルは使われません。

(4) 定義ファイルのインクルード

異なるプロセスで同じレコード型を使うときは，共通のレコード型の定義ファイルを一つ作成して，それを各プロセスの定義ファイルにインクルードできます。

インクルードする文は，C 言語と同じ書式です。次のように記述します。

```
#include ＜ファイル名＞　または　#include ”ファイル名”
```

インクルードファイルは，スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）の-i オプションで指定したサーチパスから読み込まれます。サーチパス内に該当するファイルがない場合は，最後にカレントディレクトリを探します。

インクルードするファイルの名称は任意です（サフィックスが .h でなくてもかまいません）。ただし，そのファイルを XATMI インタフェース定義ファイルとして直接 スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）に指定する場合は，その定義の名称規則に従ってください。

インクルードするファイルの内容は，XATMI インタフェース定義ファイルと同じです。ただし，名称が重複することもありますので，自プロセス内のサービス関数の定義は含めないことをお勧めします。

(5) 名称の付け方の注意

1. サービスプログラム名，サブタイプ名は，OpenTP1 で規定する条件に従ってください。
   - "dc"，"DC"，"CBLDC"，"tx"，"TX"，"tp"，"TP" で始まる名称は使えません。
   - サービスプログラム名は，20 文字以内で指定してください。
   - サブタイプ名の最大長は 32 文字です。そのうち 16 文字までが有効になります。重複があるかどうかは，16 文字までの範囲でチェックしています。
   - 型付きレコードの中で使うデータのデータ名の最大長は，32 文字です。
2. 同じプロセス内では，サービスプログラム名が重複しないようにしてください。
3. 同じプロセス内でサブタイプ名が重複した場合は，そのタイプおよび構造が一致するときは許可されます。不一致のときは，スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）がエラーリターンします。
4. 異なるプロセス内では，サービスプログラム名やサブタイプ名は同じものを許可します。ただし，サーバとして異なるプロセスにあっても，一つのクライアントから呼ばれる場合，クライアント側からは同じプロセスと見なされます。

1.3.3　XATMI インタフェース用スタブのソースファイルの作成

作成した XATMI インタフェース定義ファイルから，XATMI 用のスタブを作成します。

スタブを作成するときは，XATMI インタフェース定義を格納したファイル（**XATMI インタフェース定義ファイル**）を作成して，スタブを生成するコマンドを実行します。スタブを生成するコマンドを，次に示します。

- TCP/IP 通信をする UAP の場合：**stbmake コマンド**
- OSI TP 通信をする UAP の場合：**tpstbmk コマンド**

クライアント UAP とサーバ UAP のそれぞれに，次に示す方法でスタブを作成してください。

(1)　stbmake コマンドまたは tpstbmk コマンドで作成されるファイル

コマンドを実行すると，次の三つのファイルが作成されます（XXXXX は，XATMI インタフェース定義ファイルのサフィックス ".def" を除いた名称です）。

ファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

ファイルを作成するディレクトリ，およびファイル名は，コマンドのオプションで変更できます。

- XATMI スタブソースファイル（デフォルトのファイル名：XXXXX_stbx.c）
- XATMI スタブヘッダファイル（デフォルトのファイル名：XXXXX_stbx.h）
- XATMI スタブコピーファイル（サブタイプ名に ".cbl" が付いた名称）

(a)　XATMI スタブソースファイル

XATMI 用スタブソースファイルは，C 言語のコンパイラで翻訳して，UAP のオブジェクトファイルに結合させます。

(b)　XATMI スタブヘッダファイル

作成された XATMI スタブヘッダファイルは，UAP のソースファイルおよび XATMI スタブソースファイルにインクルードされます。

(c)　XATMI スタブコピーファイル

作成された XATMI スタブコピーファイルは，COBOL 言語で作成した UAP から COPY 文で呼び出されます。XATMI 用スタブコピーファイルによって，型付きレコードが使えるようになります。

1.3.4　stbmake（XATMI インタフェース用スタブの作成　TCP/IP 通信）

(1)　形式

```
stbmake 〔-x〕〔-b〕〔-S　スタブソースファイル名〕
        〔-H　スタブヘッダファイル名〕
        〔-i　インクルードファイルのサーチパス名〕
        〔-m　サーバの定義ファイルのサーチパス名〕〔-p〕　定義ファイル名
```

(2)　機能

XATMI インタフェースの通信を TCP/IP 通信で使う場合に必要な，XATMI 用スタブのソースファイルを作成します。stbmake コマンドは，XATMI インタフェース定義ファイルを基に，次に示すファイルを出力します。

- XATMI スタブソースファイル
- XATMI スタブヘッダファイル（C 言語で作成した UAP で使います）
- XATMI スタブコピーファイル（COBOL 言語で作成した UAP で使います）

XATMI インタフェースと OpenTP1 のリモートプロシジャコールの両方を使う UAP を作成する場合は，stbmake コマンドについて「付録 A　OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合」を参照してください。

(3)　オプション

- -x

  XATMI インタフェースを使う UAP のスタブを作成することを示します。-x オプションは，省略できます。

- -b

  COBOL 言語で作成した UAP で使う，XATMI スタブコピーファイルを作成するときに指定します。このオプションを省略すると，XATMI スタブコピーファイルは出力されません。

  XATMI スタブコピーファイルは，サブタイプ名に ".cbl" が付いた名称で出力されます。COBOL 言語の XATMI インタフェースでは，レコード型に X_C_TYPE は使えません。XATMI インタフェース定義に X_C_TYPE を指定した場合，stbmake コマンドに-b オプションを指定して実行するとエラーリターンします。

- -S スタブソースファイル名　　～〈パス名〉

  作成する XATMI スタブソースファイルのファイル名を指定します。ファイル名には，相対パス名，絶対パス名を使えます。

  -S オプションを省略すると，カレントディレクトリに XXXXX_stbx.c という名称で XATMI スタブソースファイルが作成されます。

- -H スタブヘッダファイル名　　～〈パス名〉

作成する XATMI スタブヘッダファイルのファイル名を指定します。ファイル名には，相対パス名，絶対パス名を使えます。

-H オプションを省略すると，カレントディレクトリに XXXXX_stbx.h という名称で XATMI スタブヘッダファイルが作成されます。

- -i インクルードファイルのサーチパス名　　～〈パス名〉

  XATMI インタフェース定義ファイルの＃include 文に指定したインクルードファイル名を，サーチパスで指定します。-i オプションで指定したディレクトリから，インクルードファイルを探します。

  -i オプションを省略した場合は，コマンドを実行したカレントディレクトリから探します。

  -i オプションを指定できるのは，1 回だけです。複数のサーチパスを指定したい場合は，複数のパスをコロン（:）で区切って指定します。複数のサーチパスを指定した場合は，-i オプションの引数に記述した順番で，パスが検索されます。

  サーチパスを指定するときは，英数字，アンダスコア（_），スラント（/），およびピリオド（.）を使ってください。

- -m サーバの定義ファイルのサーチパス名　　～〈パス名〉

  XATMI インタフェース定義ファイルの called_servers 文に指定したサーバの定義ファイル名を，サーチパスで指定します。-m オプションで指定したディレクトリから，インクルードファイルを探します。

  -m オプションを省略した場合は，コマンドを実行したカレントディレクトリから探します。

  サーチパスを指定するときは，英数字，アンダスコア（_），スラント（/），およびピリオド（.）を使ってください。

  -m オプションを指定できるのは，1 回だけです。複数のサーチパスを指定したい場合は，複数のパスをコロン（:）で区切って指定します。複数のサーチパスを指定した場合は，-m オプションの引数に記述した順番で，パスが検索されます。

- -p

  型付きレコードのメモリ内での配置状態を，標準出力に出力する場合に指定します。オンラインテスタを使う場合で，XATMI で使う構造体の各メンバが，メモリ内でどのように配置されているかを知りたいときに，-p オプションを指定します。

  -p オプションを指定した場合は，stbmake コマンドはファイルを作成しません。そのため，-S オプションおよび-H オプションに出力ファイルを指定しても無視されます。-m オプションおよび-i オプションはファイルを検索するため，必要に応じて指定しておいてください。

(4)　コマンド引数

- 定義ファイル名　　～〈パス名〉

  XATMI インタフェース定義ファイル名を指定します。このファイル名は，サフィックスが ".def" であることが前提です。

(5)　注意事項

- stbmake コマンドのオプションを指定できるのは，すべて 1 回ずつだけです。複数回指定した場合は，最後に指定した値が有効になります。

- stbmake コマンドで入出力できるファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

1.3.5 tpstbmk（XATMI インタフェース用スタブの作成 OSI TP 通信）

(1) 形式

```
tpstbmk 〔-b〕〔-S スタブソースファイル名〕
        〔-H スタブヘッダファイル名〕
        〔-i インクルードファイルのサーチパス名〕
        〔-m サーバの定義ファイルのサーチパス名〕 定義ファイル名
```

(2) 機能

XATMI インタフェースの通信を OSI TP 通信で使う場合に必要な，XATMI 用スタブのソースファイルを作成します。tpstbmk コマンドは，XATMI インタフェース定義ファイルを基に，次に示すファイルを出力します。

- XATMI スタブソースファイル
- XATMI スタブヘッダファイル（C 言語で作成した UAP で使います）
- XATMI スタブコピーファイル（COBOL 言語で作成した UAP で使います）

XATMI インタフェースと OpenTP1 のリモートプロシジャコールの両方を使う UAP を作成する場合は，tpstbmk コマンドについて「付録 A OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合」を参照してください。

(3) オプション

- -b

  COBOL 言語で作成した UAP で使う，XATMI スタブコピーファイルを作成するときに指定します。このオプションを省略すると，XATMI スタブコピーファイルは出力されません。

  XATMI スタブコピーファイルは，サブタイプ名に ".cbl" が付いた名称で出力されます。COBOL 言語の XATMI インタフェースでは，レコード型に X_C_TYPE は使えません。XATMI インタフェース定義に X_C_TYPE を指定した場合，tpstbmk コマンドに-b オプションを指定して実行するとエラーリターンします。

- -S スタブソースファイル名 ～〈パス名〉

  作成する XATMI スタブソースファイルのファイル名を指定します。ファイル名には，相対パス名，絶対パス名を使えます。

  -S オプションを省略すると，カレントディレクトリに XXXXX_stbx.c という名称で XATMI スタブソースファイルが作成されます。

- -H スタブヘッダファイル名 ～〈パス名〉

作成する XATMI スタブヘッダファイルのファイル名を指定します。ファイル名には，相対パス名，絶対パス名を使えます。

-H オプションを省略すると，カレントディレクトリに XXXXX_stbx.h という名称で XATMI スタブヘッダファイルが作成されます。

- **-i インクルードファイルのサーチパス名　　～〈パス名〉**

  XATMI インタフェース定義ファイルの＃include 文に指定したインクルードファイル名を，サーチパスで指定します。-i オプションで指定したディレクトリから，インクルードファイルを探します。

  -i オプションを省略した場合は，コマンドを実行したカレントディレクトリから探します。

  -i オプションを指定できるのは，1 回だけです。複数のサーチパスを指定したい場合は，複数のパスをコロン（:）で区切って指定します。複数のサーチパスを指定した場合は，-i オプションの引数に記述した順番で，パスが検索されます。

  サーチパスを指定するときは，英数字，アンダスコア（_），スラント（/），およびピリオド（.）を使ってください。

- **-m サーバの定義ファイルのサーチパス名　　～〈パス名〉**

  XATMI インタフェース定義ファイルの called_servers 文に指定したサーバの定義ファイル名を，サーチパスで指定します。-m オプションで指定したディレクトリから，インクルードファイルを探します。

  -m オプションを省略した場合は，コマンドを実行したカレントディレクトリから探します。

  サーチパスを指定するときは，英数字，アンダスコア（_），スラント（/），およびピリオド（.）を使ってください。

  -m オプションを指定できるのは，1 回だけです。複数のサーチパスを指定したい場合は，複数のパスをコロン（:）で区切って指定します。複数のサーチパスを指定した場合は，-m オプションの引数に記述した順番で，パスが検索されます。

(4)　コマンド引数

- **定義ファイル名　　～〈パス名〉**

  XATMI インタフェース定義ファイル名を指定します。このファイル名は，サフィックスが ".def" であることが前提です。

(5)　注意事項

- tpstbmk コマンドのオプションを指定できるのは，すべて 1 回ずつだけです。複数回指定した場合は，最後に指定した値が有効になります。
- tpstbmk コマンドで入出力できるファイル名の長さは，最大 255 文字です。ただし，OS の制限で 255 文字まで指定できないことがあります。

1.4　アプリケーションプログラムの実行

UAP の開始方法と終了方法，および実行環境について説明します。

1.4.1　アプリケーションプログラムの開始と終了

(1)　SUP の開始と終了

(a)　開始

SUP は，次の場合に開始します。

- ユーザサービス構成定義に SUP のサーバ名を指定した場合は，OpenTP1 を開始したとき
- ユーザサービス構成定義に SUP のサーバ名を指定していない場合は，dcsvstart コマンドを実行したとき

SUP から SPP へサービスを要求する場合は，サービスがある SPP が開始していることが前提です。SUP からサービスを要求する先の SPP は，SUP よりも先に開始させておいてください。

(b)　終了

開始した SUP を OpenTP1 から正常に終了させることはできません。OpenTP1 を正常終了させるコマンドを実行しても，OpenTP1 内のすべての SUP が終了するまで，OpenTP1 は終了しません。

SUP をコーディングするときは，SUP 自身で処理を終了するように作成しておいてください。SUP の処理がうまくいかない場合に異常終了させるときは，プログラムの処理を抜ける COBOL 言語の命令文を使って，SUP 自身で終了するようにコーディングしてください。

SUP は，dcsvstop コマンドで正常終了させることはできません。ただし，SUP を強制停止させたい場合に限り，dcsvstop -f コマンドで終了できます。

SUP のプロセスを，kill コマンドで終了させないでください。

(2)　SPP，MHP の開始と終了

(a)　開始

SPP，MHP は，ユーザサーバ（サービスグループ）単位で開始します。SPP，MHP は，次に示す場合に開始します。

- ユーザサービス構成定義に SPP，MHP のサーバ名を指定した場合は，OpenTP1 を開始したとき
- ユーザサービス構成定義に SPP，MHP のサーバ名を指定していない場合は，dcsvstart コマンドを実行したとき

マルチサーバ機能を使っている場合，常駐プロセスとして指定した数だけ，ユーザサーバのプロセスが確保されます。サービス要求が増えると，非常駐プロセスも開始します。

(b) 終了

SPP または MHP が終了するのは次の場合です。

- 次に示す OpenTP1 の終了コマンドを実行したために，終了処理に入ったとき
  dcstop コマンド（正常終了）
  dcstop　-n（強制正常終了）
  dcstop　-a（計画停止 A）
  dcstop　-b（計画停止 B）
  dcstop　-f（強制停止）
- オンラインで稼働中のプロセスを，次に示すサーバの終了コマンドを実行したために，終了処理に入ったとき
  dcsvstop コマンド（正常終了）
  dcsvstop　-f（強制停止）
- オンライン稼働中のプロセスが，ユーザサービス定義の最大プロセス数の指定を超えたために，OpenTP1 からプロセスを終了させられるとき
- SPP または MHP を非常駐プロセスで実行しているときには，サービスの処理を終了したとき
- SPP をマルチサーバで負荷分散しているときは，該当するサービスグループへのサービス要求が減少したとき

SPP，または MHP のプロセスを，kill コマンドで終了させないでください。

(3) オフラインの業務をする UAP の開始と終了

開始方法は任意です。終了もシェルからプロセスを終了させます。オフラインの業務をする UAP はユーザの責任で開始/終了を管理してください。

1.4.2 OpenTP1 で開始したアプリケーションプログラムの動作環境

- SUP，SPP，MHP の標準入力（stdin），標準出力（stdout），および標準エラー出力（stderr）は，OpenTP1 によってリダイレクトされます。
- COBOL の STOP 命令を使ってもオペレータからの入力は受け付けられません。DISPLAY 命令も使えますが，ほかの UAP からの出力と混ざる場合があります。
- UAP を開始すると，「$DCDIR/tmp/home/**ユーザサーバ名**.XX」（XX は通し番号を示します）というディレクトリが作成されます。このディレクトリをカレントワーキングディレクトリとして，UAP が稼働します。

このディレクトリはシステム共通定義の prc_current_work_path オペランドを設定することによって変更できます。

- ユーザ ID（UID）とグループ ID（GID）は，ユーザサーバの環境設定時に指定した値になります。
- ルートディレクトリは「/」のままです。
- 次に示すファイル記述子が，UAP 実行時にはオープンされています。
  ファイル記述子 0：標準入力ファイル記述子
  ファイル記述子 1：標準出力ファイル記述子
  ファイル記述子 2：標準エラー出力ファイル記述子
- umask は 000 です。
- 制御端末はありません。
- OpenTP1 では，UAP のプロセスを生成するときに自動的に UAP のシグナルを設定します。OpenTP1 で設定する UAP のシグナルの一覧を次の表に示します。

表 1-7　OpenTP1 で設定する UAP のシグナル

シグナル名	UAP プロセスの生成時の設定	動作
SIGHUP	SIG_DFL（デフォルト）	exit
SIGINT	SIG_IGN（無視）	ignore
SIGQUIT	SIG_DFL（デフォルト）	core
SIGILL	SIG_DFL（デフォルト）	core
SIGTRAP	SIG_IGN（無視）	ignore
SIGIOT※	SIG_DFL（デフォルト）	core
SIGABRT※	SIG_DFL（デフォルト）	core
SIGEMT	SIG_DFL（デフォルト）	core
SIGFEP	SIG_DFL（デフォルト）	core
SIGKILL	−	exit
SIGBUS	SIG_DFL（デフォルト）	core
SIGSEGV	SIG_DFL（デフォルト）	core
SIGSYS	SIG_DFL（デフォルト）	core
SIGPIPE※	SIG_IGN（無視）	ignore
SIGALRM	SIG_IGN（無視）	ignore
SIGTERM	SIG_DFL（デフォルト）	exit
SIGUSR1	SIG_IGN（無視）	ignore
SIGUSR2	SIG_IGN（無視）	ignore

シグナル名	UAP プロセスの生成時の設定	動作
SIGCLD	SIG_DFL（デフォルト）	ignore

（凡例）

－：該当しません。

注

UAP でシグナルの動作を設定する場合，設定したシグナルハンドラ内で exit()や abort()などを呼び出してプロセスを停止させないでください。シグナルハンドラ内でプロセスを停止させた場合，シグナルに割り込まれたときに，OpenTP1 のクリティカルな処理を実行していると OpenTP1 システムがダウンします。また，シグナルハンドラ内で外部変数 errno の値を書き換えないでください。

注※

再設定できないシグナルを示します。UAP を作成する際には，プログラム内でこれらのシグナルの動作を設定し直さないでください。

1.4.3　アプリケーションプログラムの環境変数

UAP の環境変数は，ユーザサーバの環境を設定するときに，ユーザサーバごとに設定できます。ただし，次に示す環境変数は，OpenTP1 で設定されます。

- DCDIR：OpenTP1 ホームディレクトリ
- DCCONFPATH：OpenTP1 のシステム定義ファイルを格納するディレクトリ
- DCSVNAME：ユーザサーバ名
- DCSVGNAME：サービスグループ名（SPP，MHP の場合だけ参照できます）
- DCUAPCONFPATH：OpenTP1 のユーザサービス定義ファイルまたはユーザサービスデフォルト定義ファイルを格納するディレクトリ（DCCONFPATH とは別のディレクトリに格納したい場合に設定します）

上記のほかにも，"DC"で始まる環境変数は，OpenTP1 が使います。これらの環境変数は参照できますが，変更しないでください。変更した場合，システムの動作は保証しません。

OpenTP1 配下で動作する SUP，SPP，MHP には，telnet などで OpenTP1 管理者でログインした際に設定される環境変数は引き継がれません。これらの環境変数は，ユーザサービス定義で設定し直してください。

1.4.4　トラブルシュート関連

障害に備えて，COBOL 言語を使うときは，環境変数 CBLCORE に 1 を指定します（OpenTP1 がデフォルトで設定しています）。

この指定をしないと，UAP プロセスの異常終了時にコアファイルが作成されないので，トラブルシュートができなくなります。

2 OpenTP1 の COBOL-UAP 作成用プログラムの文法

この章では，OpenTP1 の UAP を COBOL 言語で作成する場合の，UAP 作成用プログラム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

OpenTP1 の UAP を COBOL 言語で作成するときは，OpenTP1 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。コーディングする COBOL 言語として，COBOL85 と COBOL2002 が使えます。

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。

ここで示す形式は，COBOL85 と COBOL2002 で共通です。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。

データ名のけた数は，断りがないかぎり変更しないでください。COBOL-UAP 作成用プログラムが正常に動作しないことがあります。

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。

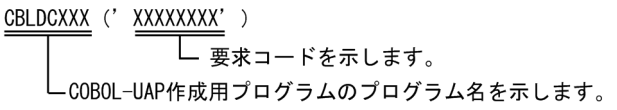
データ名として指定する文字の長さなどは，コーディングで使う COBOL 言語および COBOL コンパイラの仕様に従ってください。

COBOL 言語でコーディングするときは，OpenTP1 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，**/BeTRAN/examples/COBOL/ディレクトリ**の下に，各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。

DCXXX.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。



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。データ領域に値を設定する場合が限られているときは，そのデータ名の説明に値を設定する場合を【 】で示します。

2 進形式のデータ項目に PICTURE 句で指定したけた数を超える値を指定する場合，次のどちらかで対処してください。

- データ部（DATA DIVISION）では値を指定しないで，手続き部（PROCEDURE DIVISION）で MOVE 文などによって値を指定するようにコーディングする。
- 2 進形式のデータ項目に指定できる初期値を拡張するコンパイルオプション（COBOL85 の場合：-Vx，COBOL2002 の場合：-BinExtend）を指定して，コンパイルする。

OpenTP1 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，OpenTP1 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。データ領域に OpenTP1 から値が返される場合が限られているときは，そのデータ名の説明に値が返される場合を【】で示します。

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

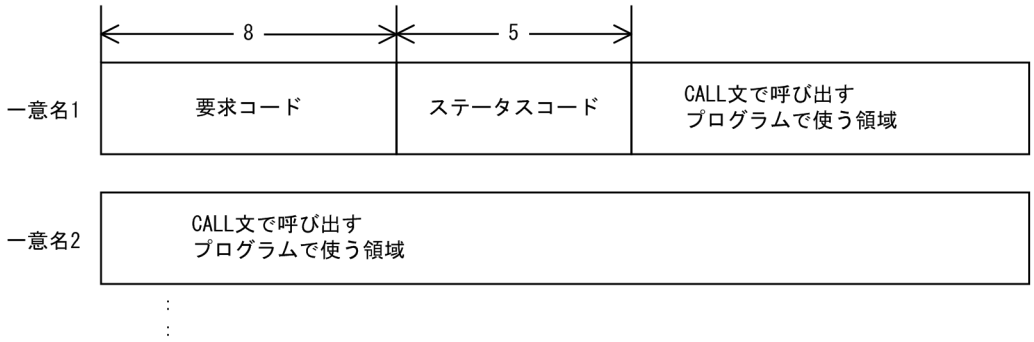
同期応答型 RPC，非同期応答型 RPC の場合に，サービスプログラムから値が返されるデータ名です。CBLDCRPC('CALL ')，CBLDCRPC('POLLANYR')を呼び出した UAP では，ここに示すデータ領域の値を参照できます。

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どうかがわかります。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。

メインプログラムとサービスプログラムの作成

UAP のメインプログラムとサービスプログラムの文法について説明します。SPP，MHP はメインプログラムとサービスプログラムを，SUP はメインプログラムだけを作成します。

- メインプログラムの作成（SUP，SPP，MHP）
- サービスプログラムの作成（SPP）
- サービスプログラムの作成（MHP）

OpenTP1 の UAP のうち，TP1/LiNK で使えるのは SUP と SPP です。MHP を使うためには，TP1/Messaging が必要です。

メインプログラムの作成（SUP，SPP，MHP）

形式

コーディングに使う COBOL 言語の仕様に従って作成します。ただし，必ず STOP RUN で終了してください。そのほかは，OpenTP1 では特に制限はありません。

機能

UAP のプロセスが開始して，OS から最初に呼ばれるプログラムです。

SUP のメインプログラムの場合

SUP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラムを次に示します。

1. CBLDCRPC('OPEN ')
   アプリケーションプログラムの開始
2. CBLDCADM('COMPLETE')
   ユーザサーバの開始処理完了の報告
3. CBLDCRPC('CLOSE ')
   アプリケーションプログラムの終了（業務の終了後に呼び出し）

SUP のメインプログラムでは，上記の COBOL-UAP 作成用プログラム以外にも，業務に必要な UAP プロセスの初期化や，終了処理，およびリモートプロシジャコール（CBLDCRPC('CALL ')）も使えます。

SPP のメインプログラムの場合

SPP から提供するサービスとして作成したサービスプログラムを，一つの実行形式ファイルにまとめます。一つのメインプログラムと複数のサービスプログラムから構成される実行形式ファイルが，一つのサービスグループに対応します。

SPP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラムを次に示します。

1. CBLDCRPC('OPEN ')
   アプリケーションプログラムの開始
2. CBLDCRSV('MAINLOOP')
   SPP のサービス開始
3. CBLDCRPC('CLOSE ')
   アプリケーションプログラムの終了（業務の終了後に呼び出し）

SPP のサービスで MCF の機能を使う（CBLDCMCF を呼び出す）場合は，次に示す COBOL-UAP 作成用プログラムもあわせて呼び出してください。

- CBLDCMCF('OPEN ') − MCF 環境のオープン（1.と 2.の間）

- CBLDCMCF('CLOSE ') − MCF 環境のクローズ（2.と 3.の間）

初期化の処理後，メインプログラムは CBLDCRSV('MAINLOOP')で止まっています。その間に該当するサービスプログラムへのサービス要求を処理します。SPP のメインプログラムでは，上記の COBOL-UAP 作成用プログラム以外にも，業務に必要な UAP プロセスの初期化や，終了処理，およびリモートプロシジャコール（CBLDCRPC('CALL ')）も使えます。

MHP のメインプログラムの場合

メッセージを処理するアプリケーションとして作成したサービスプログラムを，一つの実行形式ファイルにまとめます。一つのメインプログラムと複数のサービスプログラムから構成される実行形式ファイルが，一つのサービスグループに対応します。サービスグループ名はドメイン内（全ネットワーク内）で一意になる名称にしてください。

MHP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラムを次に示します。

1. CBLDCRPC('OPEN ')
   アプリケーションプログラムの開始
2. CBLDCMCF('OPEN ')
   MCF 環境のオープン
3. CBLDCMCF('MAINLOOP')
   MHP のサービス開始
4. CBLDCMCF('CLOSE ')
   MCF 環境のクローズ（業務の終了後に呼び出し）
5. CBLDCRPC('CLOSE ')
   アプリケーションプログラムの終了（業務の終了後に呼び出し）

アプリケーション名に該当するサービスプログラムがある MHP が開始されて，初期化の処理後，メインプログラムは CBLDCMCF('MAINLOOP')で止まっています。その間に該当するサービスプログラムで処理します。MHP のメインプログラムでは，上記の COBOL-UAP 作成用プログラム以外にも，業務に必要な UAP プロセスの初期化や，終了処理，およびリモートプロシジャコール（CBLDCRPC('CALL ')）も使えます。

引数

dcsvstart コマンドの-a オプションを使用して，SUP のメインプログラムに第 1 引数を渡すことができます。dcsvstart 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。

注意事項

メインプログラムを COBOL で作成するときは，正常終了時には必ず STOP RUN で終了してください。EXIT PROGRAM では終了しないでください。

サービスプログラムの作成（SPP）

形式

```
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.

PROCEDURE DIVISION USING 一意名1 一意名2 一意名3 一意名4
         :
         :
サービスの処理
         :
         :
EXIT PROGRAM.
```

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 で呼び出した CBLDCRPC('CALL ')から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/Server Base の場合
  ユーザサービス定義で指定します。
- TP1/LiNK の場合
  UAP の環境を設定するコマンドを実行して，対話形式で指定します。

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。これは，クライアント UAP が CBLDCRPC('CALL ')に設定した値です。

- 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）

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。

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トランザクションとして実行したかどうか，または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。

●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。

●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。

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

1. 非応答型 RPC の CBLDCRPC('CALL ')（データ名 C に 1 を設定）で呼び出したサービスプログラムでは，データ名 C とデータ名 D は参照できません。
2. 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関数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。

3. SPP のサービスプログラムから呼び出せない COBOL-UAP 作成用プログラムを次に示します。
   - CBLDCRPC('OPEN ') － アプリケーションプログラムの開始
   - CBLDCADM('COMPLETE') － ユーザサーバの開始処理完了の報告
   - CBLDCRSV('MAINLOOP') － SPP のサービス開始
   - CBLDCRPC('CLOSE ') － アプリケーションプログラムの終了

   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。このような命令文を使った場合の UAP の動作は保証しません。また，子プロセスを生成するシステムコールを使った場合，生成した子プロセスでは OpenTP1 の COBOL-UAP 作成用プログラムを呼び出せません。
4. SPP のサービスプログラムから MCF の機能（CBLDCMCF）を呼び出す場合は，メインプログラムで次に示す COBOL-UAP 作成用プログラムを呼び出しておいてください。
   - CBLDCMCF('OPEN ') － MCF 環境のオープン
   - CBLDCMCF('CLOSE ') － MCF 環境のクローズ
5. SPP のサービスプログラムでは，CBLDCMCF('RECEIVE ')は呼び出せません。
6. データ名 A に渡された入力パラメタに対して，データ名 B に渡された入力パラメタ長の領域を超える操作や参照などはしないでください。操作や参照などをした場合の動作は保証しません。プロセスが異常終了する場合があります。
7. SPP のサービスプログラムから MCF の関数（CBLDCMCF）を呼び出す場合は，一意名 1 などのデータ領域は，必ず偶数番地から始まるよう設定してください。一意名 1 などのデータ領域の先頭アドレスが奇数番地だとバスエラーが発生します。

   例えば，CBLDCMCF で使用する一意名 3 を配列で定義し，送信データが奇数バイトで，かつ構造体に SYNC の指定がない（境界調整されていない）場合，配列の 2 件目のデータを CBLDCMCF の引数に設定すると，関数の処理中にバスエラーが発生します。

トランザクションとサービスプログラムの関係

UAP の実行環境を設定するときに，トランザクション属性とする指定をしていて，クライアント UAP がトランザクションとして実行している場合は，サービスプログラムもトランザクションとして実行されます。この場合，サービスプログラムで CBLDCTRN('BEGIN ')を呼び出さないでください。

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。

UAP の実行環境を設定するときに，トランザクション属性とする指定をしていても，クライアント UAP がトランザクションとして実行していないと，サービスプログラムはトランザクションとして実行されま

せん。サービスプログラムの処理をトランザクションとして実行する場合は，任意の時点で CBLDCTRN('BEGIN ')と同期点を取得するプログラムを呼び出してください。

UAP の実行環境を設定するときに，トランザクション属性としない指定をした場合は，サービスプログラムから CBLDCTRN('BEGIN ')を呼び出しても，サービスプログラムはトランザクションとして実行できません。

ステータスコード

ステータスコードはありません。OpenTP1 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。

サービスプログラムの作成（MHP）

形式

```
PROGRAM-ID. プログラム名.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
          :
DATA DIVISION.
WORKING-STORAGE SECTION.
          :
          :
PROCEDURE DIVISION.
          :
          :
サービスの処理
          :
          :
EXIT PROGRAM.
```

機能

サービスを実行して結果を返す MHP のサービスプログラムです。MCF でメッセージを受信すると，アプリケーション名に該当するサービスプログラムがある MHP が起動されます。

MHP のサービスプログラムは，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 の名称に対応させます。この対応づけはサービスプログラムを実行するプロセスのユーザサービス定義で指定します。

サービス名とアプリケーション名の対応づけは，MCF アプリケーション定義で指定します。

データ領域の設定

なし。

サービスプログラムの処理での注意

1. 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プログラムが 1 回実行されるたびに初期化されます。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の処理で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。
2. MHP のサービスプログラムから呼び出せない COBOL-UAP 作成用プログラムを次に示します。
   CBLDCRPC('OPEN ') － アプリケーションプログラムの開始
   CBLDCRPC('CLOSE ') － アプリケーションプログラムの終了
   CBLDCADM('COMPLETE') － ユーザサーバの開始処理完了の報告
   CBLDCRSV('MAINLOOP') － SPP のサービス開始
   CBLDCMCF('MAINLOOP') － MHP のサービス開始

CBLDCMCF('OPEN ') − MCF 環境のオープン

CBLDCMCF('CLOSE ') − MCF 環境のクローズ

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。このような命令文を使った場合の UAP の動作は保証しません。また，子プロセスを生成するシステムコールを使った場合，生成した子プロセスでは OpenTP1 の COBOL-UAP 作成用プログラムを呼び出せません。

3. MHP のサービスプログラムへは，ほかの UAP から CBLDCRPC('CALL ')でサービスを要求できません。
4. CBLDCMCF の一意名 1 などのデータ領域は，必ず偶数番地から始まるよう設定してください。一意名 1 などのデータ領域の先頭アドレスが奇数番地だとバスエラーが発生します。

   例えば，CBLDCMCF で使用する一意名 3 を配列で定義し，送信データが奇数バイトで，かつ構造体に SYNC の指定がない（境界調整されていない）場合，配列の 2 件目のデータを CBLDCMCF の引数に設定すると，関数の処理中にバスエラーが発生します。

ステータスコード

ステータスコードはありません。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。

システム運用の管理（CBLDCADM）

UAP から OpenTP1 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。

- CBLDCADM('COMMAND ') − 運用コマンドの実行
- CBLDCADM('COMPLETE') − ユーザサーバの開始処理完了の報告
- CBLDCADM('STATUS ') − ユーザサーバの状態の報告

システム運用の管理の COBOL-UAP 作成用プログラム（CBLDCADM）は，TP1/Server Base と TP1/LiNK のどちらの UAP でも使えます。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。システム運用の管理（CBLDCADM）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCADM.cbl に格納してあります。

CBLDCADM('COMMAND ')

名称

運用コマンドの実行

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCADM’ 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 ’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.
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  PIC X(n).
```

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 からデータ名 G を sh(1)に渡します。この時のプロセスは，シェルが処理を完了するまで待ち，シェルの exit のステータスを戻します。コマンドの処理が終了すると，領域に標準出力情報と標準エラー出力情報が返ります。

コマンドを実行する UAP を使う OpenTP1 には，コマンドを格納しているディレクトリをサーチパスに追加してください。なお，サーチパスは次に示すどれかの方法で追加してください。

- プロセスサービス定義の prcsvpath オペランドにコマンドのパス名を指定
- prcpath コマンドでサーチパスを追加
- ユーザサービス定義に環境変数を putenv PATH と指定

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 E

データ名 G に設定したコマンドの長さを設定します。

●データ名 G

コマンドの文字列を設定します。

●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内容（データ名 J に返される値）の大きさをバイト数で設定します。事前にデータ名 J で指すアドレスから，データ名 H に設定するバイト数分の領域を確保しておいてください。データ名 H に指定するバイト数は，UAP で実行するコマンドに応じて決めてください。

処理終了後はデータ名 J に格納された文字列の長さが OpenTP1 から返されます。データ名 H 自身の長さは含みません。

●データ名 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内容（データ名 M に返される値）の大きさをバイト数で設定します。事前にデータ名 M で指すアドレスから，データ名 K に設定するバイト数分の領域を確保しておいてください。データ名 K に指定するバイト数は，UAP で実行するコマンドに応じて決めてください。

処理終了後はデータ名 M に格納した文字列の長さが OpenTP1 から返されます。データ名 K 自身の長さは含みません。

OpenTP1 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

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。

＊：waitpid(2)で指定したフォーマットで，sh(1)の終了ステータス

●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。データ名 H 自身の長さは含みません。

●データ名 J

コマンドが標準出力に出力した文字を格納する領域です。格納できる最大文字長はデータ名 H で設定した値までです。データ名 H に設定した値を超える長さの文字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 に格納した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。

●データ名 M

コマンドが標準エラー出力に出力した文字を格納する領域です。格納できる最大文字長はデータ名 K に設定した値までです。データ名 K に設定した値を超える長さの文字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。
01801	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。
01802	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。
01803	標準出力のデータが，領域に入り切りませんでした。
01804	標準エラー出力のデータが，領域に入り切りませんでした。
01805	標準出力のデータと標準エラー出力のデータの両方が，領域に入り切りませんでした。
01806	システムコール（close，pipe，dup，または read）に失敗しました。
01807	CBLDCRPC('OPEN ')を呼び出していません。
01808	メモリが不足しました。

注意事項

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，正しいコマンドが起動されないで別のコマンドが起動されます。また，コマンド名は，OpenTP1 が提供するコマンド群（$DCDIR/bin の下）のコマンド名とも重複しないようにしてください。

CBLDCADM('COMPLETE')

名称

ユーザサーバの開始処理完了の報告

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCADM’ 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 ’COMPLETE’.
   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

SUP の開始処理が終了したことを，OpenTP1 に報告します。呼び出した CBLDCADM('COMPLETE') が正常に終了したことで，SUP の起動が完了します。

SPP と MHP では，CBLDCRSV('MAINLOOP')および CBLDCMCF('MAINLOOP')が正常に終了したことで開始処理の終了と見なすため，CBLDCADM('COMPLETE')を呼び出す必要はありません。

オフラインの業務をする UAP からは，CBLDCADM('COMPLETE')を呼び出せません。

UAP で値を設定するデータ領域

●データ名 A

ユーザサーバの開始処理完了の報告を示す要求コードを「VALUE 'COMPLETE'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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。
01802	要求コード（データ名 A）が間違っています。
01830	プロセス間通信でエラーが起こりました。
01831	データ名に設定した値が間違っています。
01832	ステータス情報の入出力エラーが起こりました。
01833	ユーザサーバが正常開始中，または再開始中ではありません。

CBLDCADM('STATUS ')

名称

ユーザサーバの状態の報告

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCADM’ 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 ’STATUS  ’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(5).
   02 データ名Z  PIC S9(9)  COMP VALUE ZERO.
```

機能

このプログラムを呼び出したユーザサーバの状態を報告します。ユーザサーバの状態は，データ名 C に返されるステータスコードで報告されます。

UAP で値を設定するデータ領域

●データ名 A

ユーザサーバの状態の報告を示す要求コードを「VALUE 'STATUS△△'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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

データ名 B に「00000」が返されたとき（正常終了時）に，ユーザサーバの状態が返されます。

ステータスコード

データ名 B に 00000 が返されたとき（正常終了時）に，データ名 C に返されるステータスコード（ユーザサーバの状態を示します）

ステータスコード	意味
00001	ユーザサーバは正常開始中です。
00002	ユーザサーバは再開始中です。
00003	ユーザサーバはオンライン中です。
00004	ユーザサーバは終了中です。

データ名 B に返されるステータスコード（エラーが起こったかどうかを示します）

ステータスコード	意味
00000	正常に終了しました。
01802	要求コード（データ名 A）が間違っています。
01830	プロセス間通信でエラーが起こりました。
01831	データ名に設定した値が間違っています。
01832	ステータス情報の入出力エラーが起こりました。
01833	このプログラムをオフラインの業務をする UAP から呼び出しています。オフラインの業務をする UAP からは呼び出せません。
	CBLDCRPC('OPEN')を呼び出していません。

監査ログの出力（CBLDCADT）

UAP から監査ログを出力するプログラムについて説明します。監査ログの出力の COBOL-UAP 作成用プログラムを次に示します。

- CBLDCADT('PRINT ') − 監査ログの出力

CBLDCADT('PRINT ')

名称

監査ログの出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCADT’ 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 VALUE ZERO.
01 一意名2.
   02 データ名C  PIC X(12).
   02 データ名D  PIC X(3).
   02 FILLER    PIC X(1).
   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.
01 一意名3.
   02 データ名H  PIC S9(9) COMP.
   02 データ名I  PIC X(n).
```

機能

引数に設定した情報に，OpenTP1 でヘッダ情報，通番，日時，発生プログラム名，発生プロセス ID，発生場所，サブジェクト識別情報，オブジェクト情報，オブジェクトロケーション情報，リクエスト送信元ホスト，ロケーション識別情報を付けて，監査ログファイルに出力します。出力する監査ログの発生プログラム名は OpenTP1 です。監査ログの出力処理でエラーが発生した場合，エラーメッセージを標準エラー出力および syslog に出力します。

OpenTP1 では，CBLDCADT('PRINT ')で使うメッセージ ID 用に，34000 から 34999 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，34000 から 34999 までの範囲の値を付与してください。

監査ログの出力項目については，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。

UAP で値を設定するデータ領域

●データ名 A

監査ログの出力要求を示す要求コードを「VALUE 'PRINT△△△'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

監査ログごとに付けられる識別子（メッセージ ID）を設定します。

「KFCA$n_1n_2n_3n_4n_5$－x」の形式（11 文字）で，最後に空白文字を付けて設定します。UAP から出力する通番（$n_1n_2n_3n_4n_5$ の部分）には，34000 から 34999 までの間の数値を設定します。x の部分には，出力する監査ログの内容によってメッセージの種類（「E」，「W」，または「I」）を指定してください。

●データ名 D

監査ログを出力した 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，最後は空白文字で終わらせて設定します。使用できる文字は，数字，英字，または記号の 2 文字で，最後はヌル文字で終わらせて設定します。監査ログには，先頭にアスタリスク（*）が付いて，「*AA」の形式で出力されます（AA はデータ名 D で指定した文字列です）。

●データ名 E

監査ログに出力する「監査事象種別」の値を設定します。次の数値のどれかを設定します。

監査事象種別	数値	意味
StartStop	1000	「起動・停止」の監査事象を表します。
Authentication	1001	「識別・認証」の監査事象を表します。
AccessControl	1002	「アクセス制御」の監査事象を表します。
ConfigurationAccess	1003	「構成定義」の監査事象を表します。
Failure	1004	「障害」の監査事象を表します。
LinkStatus	1005	「リンク状態」の監査事象を表します。
ExternalService	1006	「外部サービス」の監査事象を表します。
ContentAccess	1007	「重要情報アクセス」の監査事象を表します。
Maintenance	1008	「保守」の監査事象を表します。
AnomalyEvent	1009	「異常事象」の監査事象を表します。
ManagementAction	1010	「管理動作」の監査事象を表します。

監査事象種別の詳細については，マニュアル「OpenTP1 運用と操作」を参照してください。

●データ名 F

監査ログに出力する「監査事象結果」の値を設定します。次の数値のどれかを設定します。

監査事象結果	数値	意味
Success	2000	事象の成功を表します。
Failure	2001	事象の失敗を表します。
Occurrence	2002	成功または失敗の分類がない事象の発生を表します。

●データ名 G

監査ログに出力する「動作情報」の値を設定します。データ名 E に指定した「監査事象種別」と対応した，次の数値のどれかを設定します。0 を指定した場合，この項目は出力されません。

表 2-1　監査事象種別と予約語の対応

監査事象種別	予約語	数値	意味
StartStop（起動・停止）	Start	3000	開始・起動
	Stop	3001	終了・停止
Authentication（識別・認証）	Login	3002	ログイン
	Logout	3003	ログアウト
	Logon	3004	ログオン
	Logoff	3005	ログオフ
	Disable	3006	アカウントの無効化
AccessControl（アクセス制御）	Enforce	3007	実施
ConfigurationAccess（構成定義）	Refer	3008	参照
	Add	3009	追加
	Update	3010	更新
	Delete	3011	削除
Failure（障害）	Occur	3012	発生
LinkStatus（リンク状態）	Up	3013	リンク活性
	Down	3014	リンク非活性
ExternalService（外部サービス）	Request	3015	要求
	Response	3016	応答
	Send	3017	発信
	Receive	3018	受信
ContentAccess（重要情報アクセス）	Refer	3008	参照
	Add	3009	追加
	Update	3010	更新

監査事象種別	予約語	数値	意味
ContentAccess （重要情報アクセス）	Delete	3011	削除
Maintenance（保守）	Install	3019	インストール
	Uninstall	3020	アンインストール
	Update	3010	更新（アップデート）
	Backup	3021	バックアップ
	Maintain	3022	保守作業
AnomalyEvent（異常事象）	Occur	3012	発生
ManagementAction（管理動作）	Invoke	3023	（管理者の）呼び出し
	Notify	3024	（管理者への）通知

●データ名 H

監査ログに出力する「自由記述」の文字列の長さを設定します。0 を設定した場合は，この項目は出力されません。

●データ名 I

監査ログに出力する「自由記述」の値を設定します。

使用できる文字は数字，英字，記号，スペース，引用符（"），およびコンマ（,）です。最大 1024 文字です。

データ名 I の表示内容には指定した情報の先頭と末尾に引用符（"）が付与されます。また，文字列中に引用符（"）がある場合は，その文字の前に引用符（"）が付与されます。

OpenTP1 で値を設定するデータ領域

●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
00001	監査ログの出力が無効になっています。次に示すことが考えられます。 • ログサービス定義の log_audit_out オペランドに N を指定しているか，または log_audit_out オペランドの指定がありません。 • ユーザサービス定義，またはユーザサービスデフォルト定義のどちらかで log_audit_out_suppress オペランドに Y が指定されています。
	次のどれかの定義ファイルで，データ名 C で指定したメッセージ ID が log_audit_message オペランドに指定されていません。 • ログサービス定義

ステータスコード	意味
00001	• ユーザサービス定義 • ユーザサービスデフォルト定義
	規定された範囲外のメッセージを指定しています。
00000	正常に終了しました。
01900	データ領域に設定した値が間違っています。
01904	定義解析に失敗しました。
01999	dc_rpc_open 関数が発行されていません。
01997	上記以外のエラーが発生しました。

注意事項

CBLDCRSV('MAINLOOP')が終了したあとで CBLDCADT('PRINT ')を呼び出した場合，リクエスト送信元ホストには，最後に要求した CBLDCRPC('CALL ')発行元ノードのアドレスが取得されます。

DAM ファイルサービス（CBLDCDAM，CBLDCDMB）

DAM ファイルサービスで使うプログラムについて説明します。DAM ファイルサービスの COBOL-UAP 作成用プログラムを次に示します。

<オンライン環境の場合にだけ使用できる関数>

- CBLDCDAM('CLOS') － 論理ファイルのクローズ
- CBLDCDAM('END ') － 回復対象外 DAM ファイル使用の終了
- CBLDCDAM('HOLD') － 論理ファイルの閉塞
- CBLDCDAM('OPEN') － 論理ファイルのオープン
- CBLDCDAM('READ') － 論理ファイルからブロックの入力
- CBLDCDAM('REWT') － 論理ファイルのブロックの更新
- CBLDCDAM('RLES') － 論理ファイルの閉塞の解除
- CBLDCDAM('STAT') － 論理ファイルの状態の参照
- CBLDCDAM('STRT') － 回復対象外 DAM ファイル使用の開始
- CBLDCDAM('WRIT') － 論理ファイルへブロックの出力

<オフライン環境の場合にだけ使用できる関数>

- CBLDCDMB('BSEK') － 物理ファイルのブロックの検索
- CBLDCDMB('CLOS') － 物理ファイルのクローズ
- CBLDCDMB('CRAT') － 物理ファイルの割り当て
- CBLDCDMB('DGET') － 物理ファイルからブロックの直接入力
- CBLDCDMB('DPUT') － 物理ファイルへブロックの直接出力
- CBLDCDMB('GET ') － 物理ファイルからブロックの入力
- CBLDCDMB('OPEN') － 物理ファイルのオープン
- CBLDCDMB('PUT ') － 物理ファイルへブロックの出力

DAM ファイルサービスの COBOL-UAP 作成用プログラム（オンライン機能：CBLDCDAM，オフライン機能：CBLDCDMB）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，DAM ファイルサービスの COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。DAM ファイルサービス（CBLDCDAM，CBLDCDMB）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCDAM.cbl と DCDMB.cbl に格納してあります。

CBLDCDAM('CLOS')

名称

論理ファイルのクローズ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名H  PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名E  PIC X(4) VALUE ’CLOS’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

論理ファイルをクローズします。

- 回復対象の DAM ファイルの場合

  トランザクション処理の範囲内でオープンした論理ファイルのうち，トランザクションの終了までにクローズしていないものは，DAM サービスが同期点処理時にクローズします。ただし，トランザクションの範囲外（CBLDCTRN('BEGIN ')を呼び出す前）にオープンした場合や，回復対象外の DAM ファイルは，クローズしません。

  トランザクションの開始前に論理ファイルをオープンしたときは，UAP 終了時に必ずクローズしてください。

- 回復対象外の DAM ファイルの場合

  トランザクション処理とは同期しないため，論理ファイルをクローズするときに任意に CBLDCDAM('CLOS')を呼び出せます。ただし，CBLDCDAM('END ')を呼び出す前に，オープンしている論理ファイルを CBLDCDAM('CLOS')でクローズしておいてください。

  論理ファイルをクローズするときは，CBLDCDAM('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字で設定します。8 文字に満たない名称を設定するときは，後ろを空白で埋めてください。

●データ名 E

論理ファイルのクローズを示す要求コードを「VALUE 'CLOS'」と設定します。

●データ名 H

論理ファイルをオープンしたときに返されたファイル記述子を設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	論理ファイルを正常にクローズしました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	トランザクションの範囲外でオープンしている DAM ファイルを，トランザクションの範囲内でクローズしています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	ユーザサービス定義の atomic_update の指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01603	設定したファイル記述子は，正常にオープンして得られた記述子ではありません。
	DAM ファイルをオープンしていません。

ステータスコード	意味
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 E）が間違っています。

CBLDCDAM('END ')

名称

回復対象外 DAM ファイル使用の終了

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 FILLER    PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名C  PIC X(4) VALUE ’END ’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
```

機能

回復対象外の DAM ファイルの使用を終了することを宣言します。

CBLDCDAM('STRT')を呼び出した場合は，必ず CBLDCDAM('END ')を呼び出してください。呼び出さないと，回復対象外の DAM ファイルへのアクセスに使った資源が，UAP の終了まで解放されないままになります。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

回復対象外 DAM ファイルの使用終了を示す要求コードを「VALUE 'END△'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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。回復対象外の DAM ファイルの使用を終了しました。
01600	CBLDCRPC('OPEN ') を呼び出していません。
01605	CBLDCDAM('STRT') を呼び出していません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 C）が間違っています。

CBLDCDAM('HOLD')

名称

論理ファイルの閉塞

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC X(32).
01 一意名2.
   02 データ名E  PIC X(4) VALUE ’HOLD’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

論理ファイルを論理閉塞します。論理ファイルを閉塞したあとには，ここに設定した論理ファイルに対する，ほかの UAP からのアクセス要求は，すべて論理閉塞エラーになります。

- 回復対象の DAM ファイルの場合

  CBLDCDAM('HOLD')を呼び出した場合に，設定した論理ファイルがほかのトランザクションでコミット処理中のときは，コミット処理が終了したあとに閉塞します。ただし，コミット処理が完了していなくても，制御は CBLDCDAM('HOLD ')を呼び出した UAP にリターンします。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

論理ファイルの閉塞を示す要求コードを「VALUE 'HOLD'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	設定したファイルは正常に閉塞しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01601	設定された論理ファイル名は定義されていません。
01607	メモリが不足しました。
01610	論理ファイル名として設定した値が間違っています。
01618	UAP が，現在稼働している DAM サービスでは動作できないバージョンの DAM ライブラリと結合されています。
01625	データ名 C に設定した論理ファイル名は，すでに論理閉塞しています。
01626	データ名 C に設定した論理ファイル名は，すでに障害閉塞しています。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。論理ファイルを閉塞しようとした UAP には，アクセス権限がありません。
01646	閉塞しようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 E）が間違っています。

CBLDCDAM('OPEN')

名称

論理ファイルのオープン

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名H  PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名E  PIC X(4) VALUE ’OPEN’.
   02 データ名F  PIC X(1).
   02 データ名G  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

論理ファイルをオープンします。

- 回復対象の DAM ファイルの場合

  論理ファイルに対して，ファイル排他をするかブロック排他をするかを設定します。ファイル排他を指定できるのは次の場合です。

  - トランザクションブランチ単位で排他制御する指定で，トランザクションの範囲内で論理ファイルをオープンした場合

  次に示す場合は，ファイル排他を指定できません。ブロック排他で排他制御をしてください。

  - トランザクションの範囲外で論理ファイルをオープンした場合
  - グローバルトランザクション単位で排他制御する指定をした場合

  同じトランザクションブランチ内では，いったんクローズした論理ファイルを再びオープンした場合，クローズ前の状態を引き継ぎます。

- 回復対象外の DAM ファイルの場合

トランザクション処理とは同期しないため，排他に関する制限はありません。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

論理ファイルのオープンを示す要求コードを「VALUE 'OPEN'」と設定します。

●データ名 F

ブロック排他をするかファイルの排他をするかを設定します。

VALUE 'B'…ブロック排他

VALUE 'F'…ファイル排他

ファイル排他を指定できるのは，次に示す場合です。

- 回復対象の DAM ファイルを，トランザクションの範囲内でオープンする場合
- 回復対象外の DAM ファイルをオープンする場合

●データ名 G

論理ファイルへのアクセスで排他エラーが起こったとき，排他解除待ちをするかどうかを設定します。データ名 F に「VALUE 'F'」を設定した場合は，使う DAM ファイルが回復対象かどうかで動作が異なります。

VALUE 'W'… 排他解除待ちをします。

VALUE 'N'… 待たないで，エラーリターンします。

どちらも設定しなかった場合は「VALUE 'N'」が仮定されます。

データ名 F，データ名 G に設定できる値は，使う DAM ファイルが回復対象かどうかで異なります。詳細は次のとおりです。

- **回復対象の DAM ファイルの場合**
  データ名 G の排他解除待ちは，論理ファイルのオープン以降のアクセス（CBLDCDAM('READ')と CBLDCDAM('WRIT')）で排他エラーが起こった場合の設定となります。CBLDCDAM('OPEN')で排他エラーが起こった場合の設定ではありません。CBLDCDAM('OPEN')で排他エラーが起こった場合には，無条件に「01602」でエラーリターンします。

回復対象の DAM ファイルにアクセスする場合の，データ名 F，データ名 G に設定した値と排他の設定内容を次に示します。

データ名 F	データ名 G※	排他の設定内容
'F'	△	ファイル排他
'B'	'W'	ブロック排他，排他エラー時は解除待ち
	'N'	ブロック排他，排他エラー時はエラーリターン

（凡例）

△：''（空白）を設定してください。

注※

データ名 F に'B'を設定し，データ名 G の設定を省略した場合は，'N'が仮定されます。

- **回復対象外の DAM ファイルの場合**

  データ名 G の排他解除待ちは，排他エラーが起こった場合の設定となります。

  CBLDCDAM('OPEN')，CBLDCDAM('READ')および CBLDCDAM('WRIT')で排他エラーが起こった場合に，データ名 G に設定した値に従って，排他の解除を待つかどうかが決まります。データ名 G に 'N' を設定，または省略した場合に排他エラーが起こったときは，「01602」でエラーリターンします。

  回復対象外の DAM ファイルにアクセスする場合の，データ名 F，データ名 G に設定した値と排他の設定内容を次に示します。

データ名 F	データ名 G※	排他の設定内容
'F'	'W'	ファイル排他，排他エラー時は解除待ち
	'N'	ファイル排他，排他エラー時はエラーリターン
'B'	'W'	ブロック排他，排他エラー時は解除待ち
	'N'	ブロック排他，排他エラー時はエラーリターン

注※

省略した場合は，'N'が仮定されます。

回復対象か回復対象外かに関係なく，データ名 F にファイル排他を設定するとファイル全体に排他を掛けるため，CBLDCDAM('READ')および CBLDCDAM('WRIT')で排他エラーは起こりません。そのため，排他解除待ちをするかどうかを設定できません。CBLDCDAM('READ')および CBLDCDAM('WRIT')のデータ名に設定した排他解除待ち種別は無視されます。

●データ名 Z

0 を設定します。

OpenTP1 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 H

ファイル記述子が返されます。

ステータスコード

ステータスコード	意味
00000	ファイル記述子がデータ名 H に正常に設定されました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
	ユーザサービス定義の atomic_update オペランドの指定が'N'の場合に，CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
	DAM ファイルの排他を，トランザクションの範囲外から指定しています (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
	グローバルトランザクション単位の排他制御で，DAM ファイルにファイル排他を設定しています (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01601	データ名 C に設定した論理ファイル名は定義されていません。
01602	排他エラーが起こりました。
01605	ユーザサービス定義の atomic_update オペランドの指定が'Y'の場合に，CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
01607	メモリが不足しました。
01608	データ名 C に設定した論理ファイルはオープン済みです。
01610	論理ファイル名として設定した値が間違っています。
01611	データ名 F，またはデータ名 G に設定した値が間違っています。
01621	データ名 C に設定した論理ファイルは，論理閉塞されています。
01622	データ名 C に設定した論理ファイルは，障害閉塞されています。
01627	キャラクタ型スペシャルファイルのオープン数が最大値を超えました。
01628	キャラクタ型スペシャルファイルのアクセス権がありません。
01629	トランザクションサービスでエラーが起こりました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01642	デッドロックが起こりました (回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。

ステータスコード	意味
01643	ロックサービス定義で指定した待ち時間のタイムアウトのため資源を確保できませんでした（回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
01645	最大同時排他要求数を超えて，排他を要求しています。
01646	オープンしようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 E）が間違っています。

CBLDCDAM('READ')

名称

論理ファイルからブロックの入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ USING 一意名1 一意名2 一意名n
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   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H  PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名F  PIC X(4) VALUE ’READ’.
   02 データ名G  PIC X(1).
   02 データ名L  PIC X(1).
   02 データ名M  PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
   02 一意名3.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
   02 一意名4.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
       :
       :
   02 一意名m.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
01 一意名n.
   02 データ名K  PIC X(n).
```

機能

指定した論理ファイルから，指定した範囲のブロックを，参照または更新目的で入力します。

- 回復対象の DAM ファイルの場合

  論理ファイルをオープンしたときの設定に従ってブロック排他をします。

  トランザクションの範囲でない処理からでも，論理ファイルからブロックを入力できます。ただし，参照目的の入力に限り，排他の指定はできません。

複数のブロック番号の設定で入力の要求をした場合は，それらのブロックのうちで一つでもエラーが起こると，入力バッファにはブロックを入力しないでエラーを返します。このとき要求があったすべてのブロック排他は解放されます。

参照目的で入力したあとに，同じブロックに対して更新目的での入力要求をした場合に，この更新目的での入力でエラーが起こると，参照目的での入力の排他も解除されます。

トランザクション途中でブロックを更新する指定（DAM サービス定義の dam_update_block_over=flush）があっても，次のような場合には，ステータスコード「01613」でエラーリターンする場合があります。

- 一つのトランザクションブランチの中で DAM ファイルのブロックを更新（REWRITE）しないで，更新目的のブロック入力（READ）を呼び出して，そのブロック数が一括更新最大ブロック数（DAM サービス定義の dam_update_block の値）を超えた場合。

回復対象の DAM ファイルのブロックを入力する場合は，トランザクションの範囲内から CBLDCDAM('READ')を呼び出してください。

- **回復対象外の DAM ファイルの場合**

  回復対象外の DAM ファイルのブロックを入力する場合は，CBLDCDAM('READ')を呼び出す条件に制限はありません。

  回復対象外の DAM ファイルに対して，DAM サービス定義の dam_update_block に指定した値を超えて，更新目的で論理ファイルのブロックを入力した場合は，ステータスコード「01648」でエラーリターンします。

論理ファイルのブロックを入力するときは，論理ファイル名と，CBLDCDAM('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 D

一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）を設定します。

●データ名 E

入力バッファ長を設定します。入力バッファ長は（入力ブロック数×入力ブロック長）以上にしてください。

入力ブロック数とは，一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）で設定するブロック数の合計です。

●データ名 H

論理ファイルをオープンしたときに返されたファイル記述子を設定します。

●データ名 F

論理ファイルのブロック入力を示す要求コードを「VALUE 'READ'」と設定します。

●データ名 G

入力要求の種別を，次のどちらかで設定します。必ずどちらか一方を設定してください。

VALUE 'R'…参照目的の入力

VALUE 'M'…更新目的の入力

●データ名 L

参照目的の入力の場合，排他を掛けるかどうかを指定します。'E'を設定した場合は，同期点の処理まで排他を掛けます。

排他を掛けないで論理ファイルのブロックを入力した場合，入力処理中にほかのトランザクションから該当するブロックが更新されることがあります。この場合，入力するブロックの内容は，ほかのトランザクションでの更新処理の状態に依存します。よって，最新のブロックの内容を参照する場合には，必ず'E'を指定してください。

更新目的の入力の場合は，空白を設定します。空白以外の値を設定しても無視されます。

VALUE 'E'…排他をします。

VALUE 'N'…排他をしません。

トランザクションの外から回復対象の DAM ファイルに参照目的でアクセスする場合は，排他する指定はできません。

データ名 G に「VALUE 'M'」を設定した場合は，データ名 L に' '（空白）を設定してください。

●データ名 M

排他エラーが起こったとき，排他解除待ちをするかどうかを設定します。データ名 L に「VALUE 'N'」を設定した場合は，データ名 M に' '（空白）を設定してください。

VALUE 'W' … 排他解除待ちをします。

VALUE 'N' … 待たないで，エラーリターンします。

VALUE SPACE… CBLDCDAM('OPEN')のデータ名 G に設定した値に従います。

「VALUE SPACE」を設定，または何も設定しなかった場合は次のようになります。

- CBLDCDAM('OPEN')のデータ名 G に「VALUE 'W'」を設定していれば，排他解除待ちをします。

- CBLDCDAM('OPEN')のデータ名 G に「VALUE 'N'」を設定，または省略している場合は待たないでエラーリターンします。

データ名 H に設定したファイル記述子の CBLDCDAM('OPEN')で，排他種別にファイル排他を設定していた場合は，ここに設定する値は意味を持ちません。

データ名 G，データ名 L，データ名 M に設定した値と，排他の設定内容の対応を次に示します。

データ名 G	データ名 L※1	データ名 M※2	排他の設定内容
'R'	'E'	'W'	参照目的，排他あり，排他エラー時は解除待ち
		'N'	参照目的，排他あり，排他エラー時はエラーリターン
	'N'	△	参照目的，排他なし※3
'M'	△	'W'	更新目的，排他エラー時は解除待ち
		'N'	更新目的，排他エラー時はエラーリターン

（凡例）

△ ：''（空白）を設定してください。

注※1

データ名 G に'R'を設定し，データ名 L の設定を省略した場合は，'N'が仮定されます。

注※2

次の場合，データ名 M に空白を設定すると，排他の設定内容は論理ファイルをオープンしたときの排他待ち種別の値に従います。

・データ名 G に'R'を設定し，かつデータ名 L に'E'を設定した場合

・データ名 G に'M'を設定した場合

注※3

回復対象の DAM ファイルの場合は，トランザクションの範囲でない処理から論理ファイルのブロックを入力する場合，データ名 G に'R'を設定して，データ名 L に'N'を設定したときだけ有効です。これ以外の値を設定して論理ファイルからブロックを入力した場合はステータスコード「01600」でエラーリターンします。

●データ名 I

アクセスするブロックの，先頭の相対ブロック番号を設定します。

●データ名 J

アクセスするブロックの，最後の相対ブロック番号を設定します。0 を設定した場合は，データ名 I で設定した相対ブロック番号のブロックだけを入力します。

●データ名 K

入力データ領域を設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	すべてのブロック入力は正常に終了しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	トランザクションの範囲外で，更新目的の入力，または排他を指定した参照目的の入力をしています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません （回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01602	排他エラーが起こりました。
01603	データ名 H に設定したファイル記述子は，正常にオープンして得られたファイル記述子ではありません。
	DAM ファイルをオープンしていません。
01604	入力バッファ長に設定した長さが，すべてのブロックを入力するためには不十分です。
01606	相対ブロック番号が間違っています。
01607	メモリが不足しました。
01609	データ名 D に 1 より小さい値を設定しています。
01611	データ名 G またはデータ名 L に設定した値が間違っています。
01613	ブロック更新した回数※が，DAM サービス定義の 1 トランザクションで更新できる最大ブロック数に指定した値を超えました。
01618	UAP が，現在稼働している DAM サービスでは動作できないバージョンの DAM ライブラリと結合されています。
01620	入力エラーが起こりました。
01621	指定したファイルが論理閉塞されています。
01622	指定したファイルが障害閉塞されています。

ステータスコード	意味
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。論理ファイルのブロックを入力しようとした UAP には，アクセス権限がありません。
01629	トランザクションサービスでエラーが起こりました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01642	デッドロックが起こりました。
01643	ロックサービス定義で指定した待ち時間のタイムアウトのため，資源を確保できませんでした。
01645	最大同時排他要求数を超えて，排他要求されました。
01648	回復対象外の DAM ファイルにアクセスできるブロック数※を超えました (回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 F）が間違っています。

注※

詳細については，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランドを参照してください。

注意事項

ステータスコード「01613」「01648」が返された場合，次のどれかの方法で処置してください。

- DAM サービス定義の dam_update_block オペランドの指定値を大きくする。
- DAM サービス定義の dam_update_block_over オペランドに error を指定して「01613」となった場合，dam_update_block_over オペランドの指定を flush に変更する。
- 更新目的で入力するブロック数を DAM サービス定義の dam_update_block オペランドの指定値以下にする。

定義を変更する場合は，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランド，および dam_update_block_over オペランドを参照し，注意事項を確認してください。

CBLDCDAM('REWT')

名称

論理ファイルのブロックの更新

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ USING 一意名1 一意名2 一意名n
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   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H  PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名F  PIC X(4) VALUE ’REWT’.
   02 データ名G  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
   02 一意名3.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
   02 一意名4.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
        :
        :
   02 一意名m.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
01 一意名n.
   02 データ名K  PIC X(n).
```

機能

論理ファイルから入力したブロックを，更新目的で出力します。または更新要求を取り消します。

ブロックを更新するタイミングを次に示します。

- 回復対象の DAM ファイルの場合

更新したデータはDAMサービス専用共用メモリ上に蓄えておき，トランザクションがコミットしたときに，実際にファイルを更新します。ただし，ディファード更新を指定したDAMファイルの場合は，トランザクションのコミットとは非同期に更新されます。

- **回復対象外のDAMファイルの場合**
  CBLDCDAM('REWT')が正常に終了した時点で，DAMファイルが更新されます。

複数のブロックを一括して設定したときに，指定したブロックのうちで一つでもエラーになると，処理を中断してエラーリターンします。このとき更新はしません。

論理ファイルのブロックを更新するときは，論理ファイル名と，CBLDCDAM('OPEN')で返されたファイル記述子を設定します。

UAPで値を設定するデータ領域

●データ名A

DAMファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名C

論理ファイル名を8文字以内で設定します。8文字に満たない場合は，後ろを空白で埋めてください。

●データ名D

一意名3から一意名mの数（データ名Iとデータ名Jの組の数）を設定します。

●データ名E

更新データ長を設定します。更新データ長は（更新ブロック長×更新ブロック数）にしてください。

更新ブロック数とは，一意名3から一意名mの数（データ名Iとデータ名Jの組の数）で設定するブロック数の合計です。

●データ名H

論理ファイルをオープンしたときに返されたファイル記述子を設定します。

●データ名F

論理ファイルのブロック更新を示す要求コードを「VALUE 'REWT'」と設定します。

●データ名G

更新種別を，次のどちらかで設定します。

VALUE 'U'…更新要求

VALUE 'C'…更新の取り消し

●データ名 I

アクセスするブロックの，先頭の相対ブロック番号を設定します。

●データ名 J

アクセスするブロックの，最後の相対ブロック番号を設定します。0 を設定した場合は，データ名 I に設定した相対ブロック番号のブロックだけを更新します。

●データ名 K

更新データ領域を設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	すべてのブロック更新は正常に終了しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	トランザクションの範囲外で，CBLDCDAM('REWT')を呼び出しています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01603	データ名 H に設定したファイル記述子は正常にオープンして得られたファイル記述子ではありません。
	DAM ファイルをオープンしていません。
01604	更新データ長（更新ブロック長×更新ブロック数）が短過ぎます。
01605	更新目的で論理ファイルのブロックを入力していません。
01606	相対ブロック番号が間違っています。

ステータスコード	意味
01607	メモリが不足しました。
01609	データ名 D に 1 より小さい値を設定しています。
01611	データ名 G に設定した値が間違っています。
01613	ブロック更新した回数※が，DAM サービス定義の 1 トランザクションで更新できる最大ブロック数に指定した値を超えました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01620	出力エラーが起こりました（回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
01621	指定したファイルが論理閉塞されています。
01622	指定したファイルが障害閉塞されています。
01629	トランザクションサービスでエラーが起こりました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01641	更新データ長（更新ブロック長×更新ブロック数）が長過ぎます。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 F）が間違っています。

注※

詳細については，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランドを参照してください。

注意事項

ステータスコード「01613」が返された場合，次のどれかの方法で処置してください。

- DAM サービス定義の dam_update_block オペランドの指定値を大きくする。
- DAM サービス定義の dam_update_block_over オペランドに error を指定して「01613」となった場合，dam_update_block_over オペランドの指定を flush に変更する。
- 更新するブロック数を DAM サービス定義の dam_update_block オペランドの指定値以下にする。

定義を変更する場合は，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランド，および dam_update_block_over オペランドを参照し，注意事項を確認してください。

CBLDCDAM('RLES')

名称

論理ファイルの閉塞の解除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC X(32).
01 一意名2.
   02 データ名E  PIC X(4) VALUE ’RLES’.
   02 データ名F  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

CBLDCDAM('HOLD')で閉塞された論理ファイル，および障害閉塞されている論理ファイルの閉塞を解除します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

論理ファイルの閉塞解除を示す要求コードを「VALUE 'RLES'」と設定します。

●データ名 F

閉塞の解除種別を，次のどちらかで設定します。

VALUE 'L'…論理閉塞を解除します。

VALUE 'O'…障害閉塞を解除します。

●データ名 Z

0 を設定します。

OpenTP1 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	設定した論理ファイルの閉塞を正常に解除しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
	CBLDCRPC('STRT')を呼び出していません (回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01601	データ名 C に設定した論理ファイルは定義されていません。
01607	メモリが不足しました。
01610	論理ファイル名として設定した値が間違っています。
01611	データ名 F に設定した値が間違っています。
01618	OpenTP1 ファイルシステムのバージョンが，OpenTP1 システムのバージョンと一致していません。
01619	データ名 C で指定した論理ファイルに対応する物理ファイルが存在しません。
01620	入力エラーが起こりました。
01623	指定した論理ファイルは，論理閉塞されていません。
01624	指定した論理ファイルは，障害閉塞されていません。
01627	キャラクタ型スペシャルファイルのオープン数が，最大値を超えました。
01628	キャラクタ型スペシャルファイルにはアクセス権がありません。

ステータスコード	意味
01628	アクセスしようとしたDAMファイルは，セキュリティ機能で保護されています。論理ファイルの閉塞を解除しようとしたUAPには，アクセス権限がありません。
01632	物理ファイルがキャラクタ型スペシャルファイルではありません。または，指定したスペシャルファイルに対応する装置がありません。
01633	データ名Cで指定した論理ファイルに対応する物理ファイルが，OpenTP1ファイルシステムとして初期化されていません。
01638	データ名Cで指定した論理ファイルに対応する物理ファイルに対するアクセス権がありません。
01646	閉塞を解除しようとしたDAMファイルは，セキュリティ機能で保護されています。該当するファイルに対するACLがありません。
01690	インタフェースコード（データ名A）が間違っています。
01691	要求コード（データ名E）が間違っています。

CBLDCDAM('STAT')

名称

論理ファイルの状態の参照

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 データ名E  PIC S9(9) COMP  VALUE  ZERO.
   02 データ名F  PIC S9(9) COMP.
   02 データ名G  PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名H  PIC X(1).
   02 データ名I  PIC X(1).
   02 データ名J  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(20).
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’STAT’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
01 一意名3.
   02 データ名K  PIC X(64).
```

機能

論理ファイルの現在の状態が返されます。返される内容を次に示します。

- 論理ファイルのブロック数
- 論理ファイルのブロック長
- 論理ファイルに対応した物理ファイル名
- 論理ファイルの現在の状態（閉塞されているかどうか）
- DAM サービス定義で指定した論理ファイルの属性
- DAM サービス定義で指定した論理ファイルのセキュリティ属性

論理ファイルの状態は，論理ファイルをオープンする前でも，オープンしたあとでも参照できます。

論理ファイルの状態を参照するときは，論理ファイル名を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を，8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

0 を設定します。

●データ名 D

論理ファイルの状態の参照を示す要求コードを「VALUE 'STAT'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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

論理ファイルのブロック長が返されます。

●データ名 G

論理ファイルのブロック数が返されます。

●データ名 H

論理ファイルの現在の状態が，次に示すどれかの値で設定されます。

VALUE 'N' … 論理ファイルへアクセスできます。

VALUE 'L' … 論理ファイルは，論理閉塞されています。

VALUE 'O' … 論理ファイルは，障害閉塞されています。

VALUE 'H' … 論理ファイルへ閉塞要求中です。

●データ名 I

DAM サービス定義で指定した論理ファイルの属性が，次に示すどれかの値で設定されます。

VALUE 'Q' … ディファード更新処理の対象でない DAM ファイルです。

VALUE 'D' … ディファード更新処理対象の DAM ファイルです。

VALUE 'N' … 回復対象外の DAM ファイルです。

VALUE 'C' … キャッシュレスアクセス指定の回復対象外の DAM ファイルです。

●データ名 J

DAM サービス定義で指定した論理ファイルのセキュリティ属性が，次に示すどれかの値で設定されます。

VALUE 'N' … セキュリティの指定はありません。

VALUE 'S' … セキュリティの指定があります。

●データ名 K

論理ファイルに対応した物理ファイル名が設定されます。

ステータスコード

ステータスコード	意味
00000	データ名 H に，論理ファイルの状態を正常に設定しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません（回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
01601	データ名 C に設定した論理ファイル名は定義されていません。
01607	メモリが不足しました。
01610	データ名 C に設定した論理ファイル名が間違っています。
01611	データ名 E に設定した値が間違っています。
01612	OpenTP1 から値が返されるデータ名に設定した値が間違っているか，空白が設定されていません。
01618	UAP が，現在稼働している DAM サービスでは動作できないバージョンの DAM ライブラリと結合されています。
01628	状態を参照しようとした DAM ファイルは，セキュリティ機能で保護されています。 CBLDCDAM('STAT')を呼び出した UAP には，アクセス権限がありません。

ステータスコード	意味
01646	状態を参照しようとしたDAMファイルは，セキュリティ機能で保護されています。該当するファイルに対するACLがありません。
01690	インタフェースコード（データ名A）が間違っています。
01691	要求コード（データ名D）が間違っています。

注意事項

DAMファイルの状態を参照すると，DAMサービスは情報を取得するための排他制御をします。そのため，CBLDCDAM('STAT')を頻繁に呼び出すと，排他解除待ち時間が起こってスループットが低下することがあります。オンライン中にDAMファイルの状態を参照するのは，必要最小限にしてください。

CBLDCDAM('STRT')

名称

回復対象外 DAM ファイル使用の開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ 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 ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 FILLER    PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名C  PIC X(4) VALUE ’STRT’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
```

機能

回復対象外の DAM ファイルを使うことを宣言します。回復対象外の DAM ファイルを使うときは，論理ファイルをオープンする前に，必ず CBLDCDAM('STRT')を呼び出します。CBLDCDAM('STRT')は，UAP プロセスごとに呼び出します。

CBLDCDAM('STRT')が正常に終了すると，回復対象外の DAM ファイルにアクセスする環境が設定されます。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

回復対象外 DAM ファイルの使用開始を示す要求コードを「VALUE 'STRT'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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。回復対象外の DAM ファイルを使える状態になりました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
01607	メモリが不足しました。
01618	UAP が，現在稼働している DAM サービスでは動作できないバージョンの DAM ライブラリと結合されています。
01647	CBLDCDAM('STRT')は，すでに呼び出しています。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 C）が間違っています。

CBLDCDAM('WRIT')

名称

論理ファイルへブロックの出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDAM’ USING 一意名1 一意名2 一意名n
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’DCDAMSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   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H  PIC S9(9) COMP.
   02 FILLER    PIC X(28).
01 一意名2.
   02 データ名F  PIC X(4) VALUE ’WRIT’.
   02 データ名G  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
   02 一意名3.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
   02 一意名4.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
        :
        :
   02 一意名m.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
01 一意名n.
   02 データ名K  PIC X(n).
```

機能

指定したブロックを出力します。ブロックを出力するタイミングを次に示します。

- 回復対象の DAM ファイルの場合

  更新したデータは DAM サービス専用共用メモリ上に蓄えておき，トランザクションがコミットしたときに，実際にファイルを更新します。ただし，ディファード出力を指定した DAM ファイルの場合は，トランザクションのコミットとは非同期に出力されます。

- 回復対象外の DAM ファイルの場合
  CBLDCDAM('WRIT') がリターンした時点で，DAM ファイルが出力されます。

複数のブロックを一括して出力要求したときに，指定したブロックのうちで一つでもエラーになると，処理を中断してエラーリターンします。このときはブロックを出力しません。

論理ファイルへブロックを出力するときは，論理ファイル名と，CBLDCDAM('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMSVC'」と設定します。

●データ名 C

論理ファイル名を 8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 D

一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）を設定します。

●データ名 E

出力データ長を設定します。出力データ長は（出力ブロック長×出力ブロック数）にしてください。

出力ブロック数とは，一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）で設定するブロック数の合計です。

●データ名 H

論理ファイルをオープンしたときに返されたファイル記述子を設定します。

●データ名 F

論理ファイルのブロック出力を示す要求コードを「VALUE 'WRIT'」と設定します。

●データ名 G

排他エラーが起こったとき，排他解除待ちをするかどうかを設定します。

VALUE 'W' … 排他解除待ちをします。

VALUE 'N' … 待たないで，エラーリターンします。

VALUE SPACE… CBLDCDAM('OPEN')のデータ名 G に設定した値に従います。

「VALUE SPACE」を設定，または何も設定しなかった場合は次のようになります。

- CBLDCDAM('OPEN')のデータ名 G に「VALUE 'W'」を設定していれば，排他解除待ちをします。

- CBLDCDAM('OPEN')のデータ名 G に「VALUE 'N'」を設定，または省略している場合は待たないでエラーリターンします。

データ名 H に設定したファイル記述子の CBLDCDAM('OPEN')で，排他種別にファイル排他を設定していた場合は，ここに設定する値は意味を持ちません。

●データ名 Z

0 を設定します。

●データ名 I

アクセスするブロックの，先頭の相対ブロック番号を設定します。

●データ名 J

アクセスするブロックの，最後の相対ブロック番号を設定します。0 を設定した場合は，データ名 I に設定した相対ブロック番号のブロックだけを出力します。

●データ名 K

出力データ領域（バッファ）を設定します。

OpenTP1 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	すべてのブロック出力は正常に終了しました。
01600	CBLDCRPC('OPEN ')を呼び出していません。
	トランザクションの範囲外で，CBLDCDAM('WRIT')を呼び出しています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	ユーザサービス定義の atomic_update オペランドの指定が'N'になっています（回復対象の DAM ファイルにアクセスした場合だけリターンされます）。
	CBLDCDAM('STRT')を呼び出していません （回復対象外の DAM ファイルにアクセスした場合だけリターンされます）。
	次に示すように，UAP を正しくリンケージしていません。 • DAM サービスの API で TAM ファイルにアクセスする場合に使うライブラリ（-ltdam）を，不当にリンケージしています。 • トランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。
01602	排他エラーが起こりました。

ステータスコード	意味
01603	データ名 H に設定したファイル記述子は，正常にオープンして得られたファイル記述子ではありません。
	DAM ファイルをオープンしていません。
01604	出力データ長（出力ブロック長×出力ブロック数）が短過ぎます。
01605	DAM ファイルにアクセスする順序が間違っています。
01606	相対ブロック番号が間違っています。
01607	メモリが不足しました。
01609	データ名 D に 1 より小さい値を設定しています。
01611	データ名 G に設定した値が間違っています。
01613	ブロック更新した回数※が，DAM サービス定義の 1 トランザクションで更新できる最大ブロック数に指定した値を超えました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01620	出力エラーが起こりました (回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
01621	指定したファイルが論理閉塞されています。
01622	指定したファイルが障害閉塞されています。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。論理ファイルへブロックを出力しようとした UAP には，アクセス権限がありません。
01629	トランザクションサービスでエラーが起こりました (回復対象の DAM ファイルにアクセスした場合だけリターンされます)。
01641	出力データ長（出力ブロック長×出力ブロック数）が長過ぎます。
01642	デッドロックが起こりました。
01643	ロックサービス定義で指定した待ち時間のタイムアウトのため，資源を確保できませんでした。
01645	最大同時排他要求数を超えて，排他要求されました。
01648	アクセスできる最大ブロック数※を超えました (回復対象外の DAM ファイルにアクセスした場合だけリターンされます)。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 F）が間違っています。

注※

詳細については，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランドを参照してください。

注意事項

ステータスコード「01613」「01648」が返された場合，次のどれかの方法で処置してください。

- DAM サービス定義の dam_update_block オペランドの指定値を大きくする。
- DAM サービス定義の dam_update_block_over オペランドに error を指定して「01613」となった場合，dam_update_block_over オペランドの指定を flush に変更する。
- 出力するブロック数は，DAM サービス定義の dam_update_block オペランド指定値以下にする。
- CBLDCDAM('REWT')で更新されていないブロックがある場合は，
  CBLDCDAM('WRIT')を呼び出す前に更新する。

定義を変更する場合は，マニュアル「OpenTP1 システム定義」の DAM サービス定義の dam_update_block オペランド，および dam_update_block_over オペランドを参照し，注意事項を確認してください。

CBLDCDMB('BSEK')

名称

物理ファイルのブロックの検索

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 データ名H  PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名I  PIC S9(9) COMP.
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’BSEK’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
```

機能

物理ファイルの相対ブロック番号を指定して，該当するブロックに位置づけます。CBLDCDMB('BSEK')は，物理ファイルを再作成出力要求でオープンしたあとで，呼び出してください。

該当する相対ブロック番号がファイルにある場合，相対ブロック番号がそのまま返されます。

物理ファイルのブロックを検索するときは，CBLDCDMB('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を，（スペシャルファイル名＋ 14 文字）以内で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 I

検索するブロックの相対ブロック番号を設定します。

●データ名 D

物理ファイルの検索を示す要求コードを「VALUE 'BSEK'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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 H

検索したブロックの相対ブロック番号が返されます。

ステータスコード

ステータスコード	意味
00000	相対ブロック番号が，データ名 H に正常に設定されました。
01603	データ名 E に設定したファイル記述子は，DAM ファイルを正常にオープンして得られた記述子ではありません。
	DAM ファイルをオープンしていません。
01605	DAM ファイルにアクセスする順序が間違っています。
01606	相対ブロック番号が間違っています。
01620	出力エラーが起こりました。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 D）が間違っています。

CBLDCDMB('CLOS')

名称

物理ファイルのクローズ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 FILLER    PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC X(4).
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’CLOS’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

OpenTP1 ファイルシステム上に作成した物理ファイルをクローズします。

データがファイルの終わりまで達していない場合は，ファイルの終わりまで空白で埋めます。ただし，CBLDCDMB('OPEN')のデータ名 I に作成出力要求を設定しているか，CBLDCDMB('CRAT')を呼び出している場合にだけ，ブロックを空白で埋めます。

物理ファイルをクローズするときは，物理ファイル名と，CBLDCDMB('OPEN')，または CBLDCDMB('CRAT')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を（スペシャルファイル名＋ 14 文字）以内のパス名で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 D

物理ファイルのクローズを示す要求コードを「VALUE 'CLOS'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	正常にクローズしました。
01603	データ名 E に設定したファイル記述子は，DAM ファイルを正常にオープンして得られたファイル記述子ではありません。
	DAM ファイルをオープンしていません。
01620	出力エラーが起こりました。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 D）が間違っています。

CBLDCDMB('CRAT')

名称

物理ファイルの割り当て

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   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G  PIC S9(9) COMP.
   02 データ名H  PIC S9(9) COMP.
01 一意名2.
   02 データ名F  PIC X(4) VALUE ’CRAT’.
   02 データ名I  PIC X(1).
   02 データ名J  PIC X(1).
   02 データ名K  PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

OpenTP1 ファイルシステムに物理ファイルを割り当てます。

物理ファイルの大きさは，（ブロック長＋ 8）×（ブロック数＋ 1）になります。

物理ファイルを割り当てたあとには，物理ファイルをオープンする必要はありません。

CBLDCDMB('CRAT')で物理ファイルを割り当てたあとには，次に示す COBOL-UAP 作成用プログラムは呼び出せません。

- CBLDCDMB('GET ')
- CBLDCDMB('BSEK')
- CBLDCDMB('DGET')
- CBLDCDMB('DPUT')

出力バッファの大きさは（ブロック長＋ 8）×一括処理ブロック数になります。

アクセス権にすべて'N'（アクセス権なし）を設定した場合は，省略時の値を仮定します。

物理ファイルを割り当てるときは，物理ファイル名を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

作成する物理ファイル名を（スペシャルファイル名＋ 14 文字）以内のパス名で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 D

OpenTP1 ファイルシステム上に割り当てるブロックのブロック長を設定します。

32760 を超えないセクタ長×n－8（n は正の整数）を満たす値を設定してください。

●データ名 G

OpenTP1 ファイルシステム上に割り当てるブロック数を設定します。

設定できる範囲は 1～2147483647 です。

●データ名 H

入出力の単位となる，一括処理するブロック数を設定します。

設定できる範囲は 0～2147483647 です。0 を指定した場合は，10 になります。

●データ名 F

物理ファイルの割り当てを示す要求コードを「VALUE 'CRAT'」と設定します。

●データ名 I

ファイル所有者のアクセス権を設定します。

VALUE 'N'…アクセス権なし

VALUE 'R'…読み出し権だけあり

VALUE 'W'…書き込み権だけあり

VALUE 'B'…読み出し権と書き込み権の両方あり

●データ名 J

ファイル所有者グループのアクセス権を設定します。

VALUE 'N'…アクセス権なし

VALUE 'R'…読み出し権だけあり

VALUE 'W'…書き込み権だけあり

VALUE 'B'…読み出し権と書き込み権の両方あり

●データ名 K

ファイル所有者以外のアクセス権を設定します。

VALUE 'N'…アクセス権なし

VALUE 'R'…読み出し権だけあり

VALUE 'W'…書き込み権だけあり

VALUE 'B'…読み出し権と書き込み権の両方あり

●データ名 Z

0 を設定します。

OpenTP1 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 E

ファイル記述子が返されます。

ステータスコード

ステータスコード	意味
00000	ファイル記述子がデータ名 E に正常に設定されました。
01607	メモリが不足しました。
01608	データ名 C に設定した物理ファイルはオープン済みです。
01611	データ名 I，データ名 J，データ名 K に設定した値が間違っています。
01614	物理ファイル名が間違っています。
01615	一括処理ブロック数の値が間違っています。
01617	すでに同じ物理ファイル名が存在しています。
01618	OpenTP1 ファイルシステムのバージョンが作成時と割り当て時で一致していません。
01620	入出力エラーが起こりました。

ステータスコード	意味
01628	スペシャルファイルに対するアクセス権はありません。
	割り当てようとした DAM ファイルは，セキュリティ機能で保護されています。 CBLDCDMB('CRAT')を呼び出した UAP には，アクセス権限がありません。
01630	ブロック長が適当な値ではありません。
01631	ブロック数が適当な値ではありません。
01632	物理ファイル名がキャラクタ型スペシャルファイルではありません。またはこのスペシャルファイルに対応する装置がありません。
01633	指定した OpenTP1 ファイルが，OpenTP1 ファイルシステムとして初期化されていません。
01634	OpenTP1 ファイルシステムとして初期化したときに指定したファイル数以上の OpenTP1 ファイル（物理ファイル）を割り当てようとしています。
01635	実行しているプロセスでオープンできるファイル数の最大値を超えた値を設定しています。
01636	データ名 C に設定した物理ファイルは，現在オンラインで使っています。またはほかのプロセスで使っています。
01640	OpenTP1 ファイルシステムに，物理ファイルを割り当てる分の空き領域がありません。
01646	割り当てようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 F）が間違っています。

CBLDCDMB('DGET')

名称

物理ファイルからブロックの直接入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 データ名H  PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 データ名G  PIC S9(9) COMP.
   02 データ名I  PIC S9(9) COMP.
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’DGET’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
01 一意名3.
   02 データ名F  PIC X(n).
```

機能

物理ファイルから，設定した相対ブロック番号に該当するブロックを入力します。CBLDCDMB('DGET')は，物理ファイルを再作成出力要求でオープンしてから呼び出してください。

ブロック長がバッファ長よりも小さい場合は，ブロックを入力してそのブロック長をデータ名 H に返します。ブロック長がバッファ長よりも大きい場合は，CBLDCDMB('DGET')はエラーリターンします。

物理ファイルからブロックを直接入力するときは，CBLDCDMB('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を，（スペシャルファイル名＋ 14 文字）以内で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 G

入力バッファ長を設定します。

●データ名 I

入力するブロックの相対ブロック番号を設定します。

●データ名 D

物理ファイルからブロックの直接入力を示す要求コードを「VALUE 'DGET'」と設定します。

●データ名 F

入力バッファを設定します。

●データ名 Z

0 を設定します。

OpenTP1 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 H

検索したブロックのブロック長が返されます。

ステータスコード

ステータスコード	意味
00000	入力ブロック長がデータ名 H に正常に設定されました。
01603	データ名 E に設定したファイル記述子は，DAM ファイルを正常にオープンして得られた記述子ではありません。
	DAM ファイルをオープンしていません。
01604	データ名 G に，ブロック長よりも小さいバッファ長を設定しています。
01605	DAM ファイルにアクセスする順序が間違っています。
01606	相対ブロック番号が間違っています。

ステータスコード	意味
01620	入力エラーが起こりました。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。物理ファイルのブロックを入力しようとした UAP には，アクセス権限がありません。
01646	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 D）が間違っています。

CBLDCDMB('DPUT')

名称

物理ファイルへブロックの直接出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 データ名H  PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 データ名G  PIC S9(9) COMP.
   02 データ名I  PIC S9(9) COMP.
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’DPUT’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP  VALUE  ZERO.
01 一意名3.
   02 データ名F  PIC X(n).
```

機能

物理ファイルへ，設定した相対ブロック番号に該当するブロックを出力します。

CBLDCDMB('DPUT')は，物理ファイルを再作成出力要求でオープンしてから呼び出してください。

出力データ長がブロック長よりも小さい場合は，ブロックへ出力して残りの領域をヌル文字で埋めます。そして，そのブロック長をデータ名 H に返します。出力データ長がブロック長よりも大きい場合は，CBLDCDMB('DPUT')はエラーリターンします。

物理ファイルへブロックを直接出力するときは，CBLDCDMB('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を，（スペシャルファイル名＋ 14 文字）以内で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 G

出力データ長を設定します。

●データ名 I

出力するブロックの相対ブロック番号を設定します。

●データ名 D

物理ファイルからブロックの直接出力を示す要求コードを「VALUE 'DPUT'」と設定します。

●データ名 F

出力するデータを設定します。

●データ名 Z

0 を設定します。

OpenTP1 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 H

出力したブロックのブロック長が返されます。

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。
01603	データ名 E に設定したファイル記述子は，DAM ファイルを正常にオープンして得られた記述子ではありません。

ステータスコード	意味
01603	DAM ファイルをオープンしていません。
01604	出力データ長に，ブロック長よりも大きい値を設定しています。
01605	DAM ファイルにアクセスする順序が間違っています。
01606	相対ブロック番号が間違っています。
01620	出力エラーが起こりました。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。物理ファイルのブロックを出力しようとした UAP には，アクセス権限がありません。
01646	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 D）が間違っています。

CBLDCDMB('GET ')

名称

物理ファイルからブロックの入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 データ名H  PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 データ名G  PIC S9(9) COMP.
   02 FILLER    PIC X(4).
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’GET ’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
01 一意名3.
   02 データ名F  PIC X(n).
```

機能

OpenTP1 ファイルシステムの物理ファイルから，ブロック単位でデータを順に入力します。物理ファイルをオープンしてから，物理ファイルのブロックを入力してください。

ブロック長がバッファ長よりも小さい場合は，ブロックを入力してそのブロック長をデータ名 H に返します。ブロック長がバッファ長よりも大きい場合はエラーリターンします。

物理ファイルからブロックを入力するときは，物理ファイル名と，CBLDCDMB('OPEN')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT '」と設定します。

●データ名 C

物理ファイル名を（スペシャルファイル名＋ 14 文字）以内のパス名で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 G

入力バッファ長を設定します。

●データ名 D

物理ファイルからのブロック入力を示す要求コードを「VALUE 'GET△'」と設定します。

●データ名 F

入力バッファを設定します。

●データ名 Z

0 を設定します。

OpenTP1 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 H

入力したブロックのブロック長が返されます。

ステータスコード

ステータスコード	意味
00000	入力ブロック長がデータ名 H に正常に設定されました。
01603	データ名 E に設定したファイル記述子は，正常にオープンして得られた記述子ではありません。
	DAM ファイルをオープンしていません。
01604	ブロック長にバッファ長よりも大きい値を設定しています。
01605	DAM ファイルにアクセスする順序が間違っています。
01620	入力エラーが起こりました。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。物理ファイルのブロックを入力しようとした UAP には，アクセス権限がありません。
01637	ファイルの終わりに達しました。

ステータスコード	意味
01646	アクセスしようとしたDAMファイルは，セキュリティ機能で保護されています。該当するファイルに対するACLがありません。
01690	インタフェースコード（データ名A）が間違っています。
01691	要求コード（データ名D）が間違っています。

CBLDCDMB('OPEN')

名称

物理ファイルのオープン

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 FILLER    PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名H  PIC S9(9) COMP.
01 一意名2.
   02 データ名F  PIC X(4) VALUE ’OPEN’.
   02 データ名I  PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
```

機能

OpenTP1 ファイルシステム上に作成した物理ファイルをオープンします。ただし，オンラインで使っている物理ファイルはオープンできません。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を（スペシャルファイル名＋ 14 文字）以内のパス名で設定します。63 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 H

入出力の単位となる，一括処理するブロック数を設定します。

設定できる範囲は 0～2147483647 です。0 を指定した場合は，10 になります。

●データ名 F

物理ファイルのオープンを示す要求コードを「VALUE 'OPEN'」と設定します。

●データ名 I

作成出力要求か再作成（オーバライト）出力要求かを設定します。ここに設定した値で，ファイルのクローズ時に，残りの領域を空白ブロックで埋めるかどうかが決まります。ここに設定した値は，CBLDCDMB('PUT ')を呼び出したあとで，CBLDCDMB('CLOS')を呼び出して正常に終了したときに有効になります。

CBLDCDMB('PUT ')を呼び出していても，CBLDCDMB('CLOS')を呼び出さないで UAP の処理を終了した場合は，ヌル文字のブロックで埋めません。

VALUE 'I'…作成出力要求を設定

VALUE 'O'…再作成出力要求を設定

●データ名 Z

0 を設定します。

OpenTP1 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 E

オープン処理で返されるファイル記述子が返されます。

ステータスコード

ステータスコード	意味
00000	ファイル記述子がデータ名 E に正常に設定されました。
01607	メモリが不足しました。
01608	データ名 C に設定した物理ファイルはオープン済みです。
01611	データ名 I に設定した値が間違っています。
01614	物理ファイル名が間違っています。
01615	一括処理ブロック数の値が間違っています。
01616	データ名 C に設定した物理ファイルは，DAM ファイルではありません。
01618	OpenTP1 ファイルシステムのバージョンが，作成時と割り当て時とで一致していません。

ステータスコード	意味
01619	データ名 C に設定した物理ファイルが存在しません。
01620	入出力エラーが起こりました。
01628	スペシャルファイルに対するアクセス権がありません。
01632	物理ファイル名がキャラクタ型スペシャルファイルではありません。または設定したスペシャルファイルに対応する装置がありません。
01633	データ名 C に設定したファイルが OpenTP1 ファイルシステムとして初期化されていません。
01635	プロセスでオープンできる物理ファイル数の最大値を超えた値を設定しています。
01636	データ名 C に設定した物理ファイルは現在オンラインで使っています。またはほかのプロセスで使っています。
01638	物理ファイルに対するアクセス権がありません。
01639	物理ファイルの破壊を検出しました。
01690	インタフェースコード（データ名 A）が間違っています。
01691	要求コード（データ名 F）が間違っています。

CBLDCDMB('PUT ')

名称

物理ファイルへブロックの出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCDMB’ 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 ’DCDAMINT’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(1).
   02 データ名C  PIC X(63).
   02 FILLER    PIC X(3).
   02 FILLER    PIC S9(9) COMP.
   02 データ名E  PIC S9(9) COMP.
   02 データ名G  PIC S9(9) COMP.
   02 FILLER    PIC X(4).
01 一意名2.
   02 データ名D  PIC X(4) VALUE ’PUT ’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名Z  PIC S9(9) COMP VALUE ZERO.
01 一意名3.
   02 データ名F  PIC X(n).
```

機能

OpenTP1 ファイルシステム上に作成した物理ファイルにブロック単位でデータを順に出力します。データ長がブロック長よりも小さい場合は，データの後ろを空白で埋めます。大きい場合は，エラーリターンします。

物理ファイルにブロックを出力するときは，物理ファイル名と，CBLDCDMB('OPEN')，またはCBLDCDMB('CRAT')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

DAM ファイルとのインタフェースコードを「VALUE 'DCDAMINT'」と設定します。

●データ名 C

物理ファイル名を（スペシャルファイル名＋ 14 文字）以内のパス名で設定します。63 文字に満たない名称を設定するときは，後ろを空白で埋めてください。

●データ名 E

ファイル記述子を設定します。

●データ名 G

出力するデータの長さを設定します。

●データ名 D

物理ファイルにブロック出力を示す要求コードを「VALUE 'PUT△'」と設定します。

●データ名 F

出力するデータを設定します。

●データ名 Z

0 を設定します。

OpenTP1 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	データ名 F に設定したデータを正常に出力しました。
01603	設定したファイル記述子は，DAM ファイルを正常にオープンして得られた記述子ではありません。
	DAM ファイルをオープンしていません。
01604	データ長にブロック長よりも大きい値を設定しています。
01605	DAM ファイルにアクセスする順序が間違っています。
01620	出力エラーが起こりました。
01628	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。物理ファイルへブロックを出力しようとした UAP には，アクセス権限がありません。
01637	ファイルの終わりに達しました。
01646	アクセスしようとした DAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01690	インタフェースコード（データ名 A）が間違っています。

ステータスコード	意味
01691	要求コード（データ名 D）が間違っています。

IST サービス（CBLDCIST）

IST テーブルへアクセスするプログラムについて説明します。IST サービスの COBOL-UAP 作成用プログラムを次に示します。

- CBLDCIST('CLOS') − IST テーブルのクローズ
- CBLDCIST('OPEN') − IST テーブルのオープン
- CBLDCIST('READ') − IST テーブルからレコードの入力
- CBLDCIST('WRIT') − IST テーブルへレコードの出力

IST サービスの COBOL-UAP 作成用プログラム（CBLDCIST）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，IST サービスの COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。IST サービス（CBLDCIST）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCIST.cbl に格納してあります。

CBLDCIST('CLOS')

名称

IST テーブルのクローズ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCIST’ 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 ’DCISTSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名F  PIC S9(9) COMP.
   02 FILLER    PIC X(12).
01 一意名2.
   02 データ名G  PIC X(4) VALUE ’CLOS’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名H  PIC S9(9) COMP VALUE ZERO.
```

機能

指定した IST テーブルをクローズします。

UAP で値を設定するデータ領域

●データ名 A

IST テーブルとのインタフェースコードを「VALUE 'DCISTSVC'」と設定します。

●データ名 C

クローズする IST テーブルの IST テーブル名を，8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 F

IST テーブルをオープンして返された，テーブル記述子を設定します。

●データ名 G

IST テーブルのクローズを示す要求コードを「VALUE 'CLOS'」と設定します。

●データ名 H

0 を設定します。

OpenTP1 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	IST テーブルを正常にクローズしました。
03800	IST テーブルへアクセスする順序が間違っています。
03803	データ名 F に設定したテーブル記述子は，正常にオープンして得られたテーブル記述子ではありません。
	IST テーブルをオープンしていません。
03811	データ名 H に設定した値が間違っています。
03890	インタフェースコード（データ名 A）が間違っています。
03891	要求コード（データ名 G）が間違っています。

CBLDCIST('OPEN')

名称

IST テーブルのオープン

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCIST’ 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 ’DCISTSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   02 FILLER    PIC S9(9) COMP.
   02 FILLER    PIC S9(9) COMP.
   02 データ名F  PIC S9(9) COMP.
   02 FILLER    PIC X(12).
01 一意名2.
   02 データ名G  PIC X(4) VALUE ’OPEN’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名H  PIC S9(9) COMP VALUE ZERO.
```

機能

指定した IST テーブルをオープンします。IST テーブルを正常にオープンできると，テーブル記述子がリターンされます。

UAP で値を設定するデータ領域

●データ名 A

IST テーブルとのインタフェースコードを「VALUE 'DCISTSVC'」と設定します。

●データ名 C

オープンする IST テーブルの IST テーブル名を，8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 G

IST テーブルのオープンを示す要求コードを「VALUE 'OPEN'」と設定します。

●データ名 H

0 を設定します。

OpenTP1 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

IST テーブルのテーブル記述子が返されます。

ステータスコード

ステータスコード	意味
00000	テーブル記述子が，データ名 F に正常に返されました。
03800	IST テーブルへアクセスする順序が間違っています。
03801	データ名 C に設定した IST テーブル名は定義されていません。
03807	メモリが不足しました。
03808	データ名 C に設定した IST テーブル名はオープン済みです。
03810	IST テーブル名に設定した値の長さが間違っています。
03811	データ名 H に設定した値が間違っています。
03890	インタフェースコード（データ名 A）が間違っています。
03891	要求コード（データ名 G）が間違っています。

CBLDCIST('READ')

名称

IST テーブルからレコードの入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCIST’ USING 一意名1 一意名2 一意名n
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’DCISTSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   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.
   02 FILLER    PIC X(12).
01 一意名2.
   02 データ名G  PIC X(4) VALUE ’READ’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名H  PIC S9(9) COMP VALUE ZERO.
   02 一意名3.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
   02 一意名4.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
        :
        :
   02 一意名m.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
01 一意名n.
   02 データ名K  PIC X(n).
```

機能

指定した IST テーブルから指定した範囲のレコードを，参照の目的で入力します。複数のレコードを一括して指定した場合に，指定したレコードのうち一つでもエラーが起こると，入力バッファにはレコードを入力しないで CBLDCIST('READ')はエラーリターンします。

IST テーブルのレコードを入力するときは，IST テーブル名と，CBLDCIST('OPEN')で返されたテーブル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

IST テーブルとのインタフェースコードを「VALUE 'DCISTSVC'」と設定します。

●データ名 C

アクセスする IST テーブルの IST テーブル名を，8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 D

一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）を設定します。

●データ名 E

入力バッファ長を設定します。入力バッファ長には，（入力レコード長 × 入力レコード数）以上の値を設定してください。

●データ名 F

IST テーブルをオープンして返された，テーブル記述子を設定します。

●データ名 G

IST テーブルからレコードを入力することを示す要求コードを「VALUE 'READ'」と設定します。

●データ名 H

0 を設定します。

●データ名 I

アクセスするレコードの，先頭の相対レコード番号を設定します。

●データ名 J

アクセスするレコードの，最後の相対レコード番号を設定します。0 を設定した場合は，データ名 I で指定した相対レコード番号のレコードだけを入力します。

●データ名 K

入力データ領域（バッファ）を設定します。

OpenTP1 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	指定したレコードは，すべて正常に入力しました。
03800	IST テーブルへアクセスする順序が間違っています。
03803	データ名 F に設定したテーブル記述子は正常にオープンして得られたテーブル記述子ではありません。
	IST テーブルをオープンしていません。
03804	すべてのレコードを入力するには，データ名 E に設定した入力バッファ長が足りません。
03806	相対レコード番号が間違っています。
03807	メモリが不足しました。
03809	データ名 D に設定した値が，1 未満です。
03811	データ名 H に設定した値が間違っています。
03890	インタフェースコード（データ名 A）が間違っています。
03891	要求コード（データ名 G）が間違っています。

CBLDCIST('WRIT')

名称

IST テーブルへレコードの出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCIST’ USING 一意名1 一意名2 一意名n
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’DCISTSVC’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(8).
   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.
   02 FILLER    PIC X(12).
01 一意名2.
   02 データ名G  PIC X(4) VALUE ’WRIT’.
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(1).
   02 データ名H  PIC S9(9) COMP VALUE ZERO.
   02 一意名3.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
   02 一意名4.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
        :
        :
   02 一意名m.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC S9(9) COMP.
01 一意名n.
   02 データ名K  PIC X(n).
```

機能

指定した範囲のレコードを，IST テーブルに出力します。複数のレコードを一括して指定した場合に，指定したレコードのうち一つでもエラーが起こると，出力バッファにはレコードを出力しないで CBLDCIST('WRIT')はエラーリターンします。

CBLDCIST('WRIT')が正常に終了すると，自ノードのレコードの内容が更新されます。他ノードの IST テーブルへは，CBLDCIST('WRIT')が正常に終了してから，時間をおいて更新されます。

IST テーブルのレコードを出力するときは，IST テーブル名と，CBLDCIST('OPEN')で返されたテーブル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

IST テーブルとのインタフェースコードを「VALUE 'DCISTSVC'」と設定します。

●データ名 C

アクセスする IST テーブルの IST テーブル名を，8 文字以内で設定します。8 文字に満たない場合は，後ろを空白で埋めてください。

●データ名 D

一意名 3 から一意名 m の数（データ名 I とデータ名 J の組の数）を設定します。

●データ名 E

出力バッファ長を設定します。出力バッファ長には，（出力レコード長 × 出力レコード数）の値を設定してください。

●データ名 F

IST テーブルをオープンして返された，テーブル記述子を設定します。

●データ名 G

IST テーブルからレコードを出力することを示す要求コードを「VALUE 'WRIT'」と設定します。

●データ名 H

0 を設定します。

●データ名 I

アクセスするレコードの，先頭の相対レコード番号を設定します。

●データ名 J

アクセスするレコードの，最後の相対レコード番号を設定します。0 を設定した場合は，データ名 I で指定した相対レコード番号のレコードだけを出力します。

●データ名 K

出力データ領域（バッファ）を設定します。

OpenTP1 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	指定したレコードは，すべて正常に出力しました。
03800	IST テーブルへアクセスする順序が間違っています。
03803	データ名 F に設定したテーブル記述子は正常にオープンして得られたテーブル記述子ではありません。
	IST テーブルをオープンしていません。
03804	すべてのレコードを出力するには，データ名 E に設定した出力バッファ長が足りません。
03806	相対レコード番号が間違っています。
03807	メモリが不足しました。
03809	データ名 D に設定した値が，1 未満です。
03811	データ名 H に設定した値が間違っています。
03841	出力バッファ長が，出力するすべてのレコード長の合計よりも長過ぎます。
03890	インタフェースコード（データ名 A）が間違っています。
03891	要求コード（データ名 G）が間違っています。

ユーザジャーナルの取得（CBLDCJNL）

ユーザジャーナルを取得するプログラムについて説明します。ユーザジャーナルの取得の COBOL-UAP 作成用プログラムを次に示します。

- CBLDCJNL('UJPUT ') − ユーザジャーナルの取得

ユーザジャーナルの出力の COBOL-UAP 作成用プログラム（CBLDCJNL）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，ユーザジャーナルの COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。ユーザジャーナルの出力（CBLDCJNL）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCJNL.cbl に格納してあります。

CBLDCJNL('UJPUT ')

名称

ユーザジャーナルの取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCJNL’ 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 ’UJPUT   ’.
   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(n).
```

機能

UAP の履歴情報であるユーザジャーナル（UJ）をシステムジャーナルファイル（system_jnl_file）に取得します。CBLDCJNL('UJPUT ')を 1 回呼び出して取得する UJ の単位を，UJ レコードといいます。

CBLDCJNL('UJPUT ')を呼び出しても，すぐにはシステムジャーナルファイルへ出力されません。ジャーナルバッファに空きがなくなったときか，トランザクション処理がコミットした時点で，UJ レコードがシステムジャーナルファイルへ出力されます。

CBLDCJNL('UJPUT ')は，CBLDCRPC('OPEN ')の呼び出し後から CBLDCRPC('CLOSE ')の呼び出しまでの間で呼び出せます。出力済みの UJ レコードは，CBLDCJNL('UJPUT ')を呼び出したトランザクションで障害が発生しても，ロールバック（部分回復）で無効にできません。CBLDCJNL('UJPUT ')を呼び出したトランザクションをロールバックしても，UJ レコードはシステムジャーナルファイルへ出力されます。

UAP で値を設定するデータ領域

●データ名 A

ユーザジャーナルの取得を示す要求コードを「VALUE 'UJPUT△△△'」と設定します。

●データ名 Z

UJ レコードを取得する時点で，システムジャーナルファイルに UJ レコードを出力するかどうかを，次に示す形式で設定します。

1

UJ レコードを取得する時点で，システムジャーナルファイルに UJ レコードを出力します。トランザクション内で UJ レコードが取得されている場合，この設定は無視されます。

0

UJ レコードを取得する時点では，システムジャーナルファイルに UJ レコードを出力しません。

●データ名 C

取得する UJ の長さを設定します。設定できる長さは 1 から（取得先システムジャーナルサービス定義の jnl_max_datasize オペランドの値−8）までです。

●データ名 D

UJ コードを，0 から 255 までの値で設定します。

●データ名 E

取得する UJ のデータを設定します。UJ として有効なデータは，データ名 C で設定した長さです。

OpenTP1 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。
01101	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。
01102	ユーザジャーナルの長さ（データ名 C）に，0 以下のデータ長を設定しています。
01103	ユーザジャーナルの長さ（データ名 C）に，設定できる範囲以上の値を設定しています。
01105	CBLDCRPC('OPEN ')が呼び出されていません。または，該当するシステムの実行環境がジャーナルファイルレスモードのため，CBLDCJNL('UJPUT ')が動作できません。

注意事項

トランザクション外 UJ レコードは，ジャーナルバッファに空きがなくなったとき，またはほかのアプリケーションのトランザクションが正常終了した同期点（コミットした時点）で，システムジャーナルファイルに出力されます。トランザクションが発生しないアプリケーションで UJ レコードを取得する場合は，データ名 Z に 1 を設定した CBLDCJNL('UJPUT ')を，適切なタイミングで呼び出してください。

ジャーナルデータの編集（CBLDCJUP）

jnlrput コマンドで出力したジャーナルデータを扱うプログラムについて説明します。jnlrput コマンドのオプションと出力するデータの関係および出力形式については，マニュアル「OpenTP1 運用と操作」の jnlrput コマンドの文法の説明を参照してください。

ジャーナルデータの編集の COBOL-UAP 作成用プログラムを次に示します。

- CBLDCJUP('CLOSERPT') － jnlrput 出力ファイルのクローズ
- CBLDCJUP('OPENRPT') － jnlrput 出力ファイルのオープン
- CBLDCJUP('RDGETRPT') － jnlrput 出力ファイルからジャーナルデータの入力

ジャーナルデータの編集の COBOL-UAP 作成用プログラム（CBLDCJUP）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，ジャーナルデータの編集の COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。ジャーナルデータの編集（CBLDCJUP）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCJUP.cbl に格納してあります。

CBLDCJUP('CLOSERPT')

名称

jnlrput 出力ファイルのクローズ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCJUP’ 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 ’CLOSERPT’.
   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 FILLER    PIC X(256).
   02 データ名C  PIC S9(9) COMP.
   02 FILLER    PIC 9(9) COMP.
   02 FILLER    PIC 9(9) COMP.
```

機能

jnlrput コマンド出力ファイルをクローズします。

CBLDCJUP('CLOSERPT')には，CBLDCJUP('OPENRPT ')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

jnlrput コマンド出力ファイルのクローズを示す要求コードを「VALUE 'CLOSERPT'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

ファイル記述子を設定します。

OpenTP1 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	jnlrput コマンド出力ファイルを正常にクローズしました。
01101	要求コード（データ名 A）が間違っています。
01271	設定したファイル記述子（データ名 C）は，正常にオープンして得られた記述子ではありません。
	jnlrput コマンド出力ファイルをオープンしていません。

CBLDCJUP('OPENRPT ')

名称

jnlrput 出力ファイルのオープン

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCJUP’ 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 ’OPENRPT ’.
   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(256).
   02 データ名D  PIC S9(9) COMP.
   02 FILLER    PIC 9(9) COMP.
   02 FILLER    PIC 9(9) COMP.
01 一意名3.
   02 データ名E  PIC 9(9) COMP.
   02 データ名F  PIC X(4).
   02 データ名G  PIC X(8).
   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(4).
   02 FILLER    PIC X(96).
```

機能

jnlrput コマンド出力ファイルをオープンします。

jnlrput コマンド出力ファイルをオープンするときのファイル名には，jnlrput コマンドを実行したときに作成した名称を設定します。

入力エラー，およびメモリが不足した場合は，CBLDCJUP('OPENRPT ')は jnlrput コマンド出力ファイルをクローズしてリターンします。

UAP で値を設定するデータ領域

●データ名 A

jnlrput コマンド出力ファイルのオープンを示す要求コードを「VALUE 'OPENRPT△'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

jnlrput コマンド出力ファイル名を，256 文字以内のパス名で設定します。256 文字に満たない場合は，後ろを空白で埋めてください。

OpenTP1 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

ファイル記述子が返されます。

●データ名 E

一意名 3 の長さが返されます。

●データ名 F

jnlrput コマンド出力ファイルを示す識別子「JUP」が返されます。

●データ名 G

ノード識別子が返されます。

●データ名 H

jnlrput コマンド出力ファイルの，レコード管理情報の長さが返されます。

●データ名 I

jnlrput コマンド出力ファイルの，レコードデータヘッダの長さが返されます。

●データ名 J

jnlrput コマンド出力ファイルの，最大レコード長が返されます。

●データ名 K

jnlrput コマンド出力ファイルの，フォーマットバージョンが返されます。

ステータスコード

ステータスコード	意味
00000	ファイル記述子が，データ名 D に正常に設定されました。
01101	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
01272	データ名 C に設定した jnlrput コマンド出力ファイルは，存在しません。
	入力エラーが起こりました（オープンエラー）。
01273	データ名 C に設定したファイルは，jnlrput 出力ファイルではありません。
01274	入力エラーが起こりました（読み込みエラー）。
01278	データ名 C に設定した jnlrput コマンド出力ファイルは，すでにオープンしています。
01270	メモリが不足しました。

CBLDCJUP('RDGETRPT')

名称

jnlrput 出力ファイルからジャーナルデータの入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCJUP’ USING 一意名1 一意名2 一意名3 一意名4 一意名5
```

DATA DIVISION の指定

```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’RDGETRPT’.
   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 FILLER    PIC X(256).
   02 データ名C  PIC S9(9) COMP.
   02 データ名D  PIC 9(9) COMP.
   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.
   02 データ名G  PIC X(1).
   02 データ名Y  PIC X(1).
   02 FILLER    PIC X(2).
   02 データ名H  PIC X(4).
   02 データ名I  PIC X(8).
   02 データ名J  PIC X(9).
   02 FILLER    PIC X(3).
   02 データ名K  PIC X(4).
   02 データ名L  PIC X(8).
   02 FILLER    PIC X(12).
   02 データ名M  PIC X(12).
   02 データ名N  PIC X(12).
   02 FILLER    PIC X(1).
   02 FILLER    PIC X(3).
   02 FILLER    PIC 9(9) COMP.
   02 データ名W  PIC X(4).
   02 FILLER    PIC X(36).
01 一意名4.
   02 データ名O.
      03 データ名O1 PIC 9(9) COMP.
      03 データ名O2 PIC 9(9) COMP.
      03 FILLER    PIC X(120).
   02 データ名P REDEFINES データ名O.
      03 データ名P1 PIC X(2).
      03 データ名P2 PIC X(1).
      03 データ名P3 PIC X(1).
      03 データ名P4 PIC X(4).
      03 データ名P5 PIC X(4).
```

```
      03 データ名P6 PIC X(4).
      03 データ名P7 PIC X(4).
      03 データ名P8 PIC X(9).
      03 FILLER    PIC X(3).
      03 データ名P9 PIC X(32).
      03 データ名P10 PIC X(4).
      03 データ名P11 PIC X(4).
      03 データ名P12 PIC X(4).
      03 データ名P13 PIC X(4).
      03 データ名P14 PIC X(4).
      03 データ名P15 PIC X(4).
      03 FILLER    PIC X(40).
   02 データ名Q REDEFINES データ名O.
      03 データ名Q1 PIC X(16).
      03 データ名Q2 PIC X(10).
      03 FILLER    PIC X(2).
      03 データ名Q3 PIC X(12).
      03 データ名Q4 PIC X(9).
      03 FILLER    PIC X(3).
      03 データ名Q5 PIC X(1).
      03 データ名Q6 PIC X(1).
      03 FILLER    PIC X(2).
      03 データ名Q7 PIC 9(9) COMP.
      03 FILLER    PIC X(68).
   02 データ名R REDEFINES データ名O.
      03 データ名R1 PIC X(16).
      03 データ名R2 PIC X(16).
      03 データ名R3 PIC X(1).
      03 データ名R4 PIC X(1).
      03 FILLER    PIC X(2).
      03 データ名R5 PIC 9(9) COMP.
      03 FILLER    PIC X(88).
   02 データ名S REDEFINES データ名O.
      03 データ名S1 PIC X(16).
      03 データ名S2 PIC X(10).
      03 FILLER    PIC X(2).
      03 データ名S3 PIC X(1).
      03 データ名S4 PIC X(1).
      03 FILLER    PIC X(2).
      03 データ名S5 PIC 9(9) COMP.
      03 データ名S6 PIC 9(9) COMP.
      03 FILLER    PIC X(88).
   02 データ名T REDEFINES データ名O.
      03 データ名T1 PIC X(16).
      03 データ名T2 PIC X(10).
      03 FILLER    PIC X(2).
      03 データ名T3 PIC X(1).
      03 データ名T4 PIC X(1).
      03 FILLER    PIC X(2).
      03 データ名T5 PIC 9(9) COMP.
      03 FILLER    PIC X(92).
   02 データ名U REDEFINES データ名O.
      03 データ名U1 PIC X(16).
      03 データ名U2 PIC X(10).
      03 FILLER    PIC X(2).
      03 データ名U3 PIC X(12).
      03 データ名U4 PIC 9(9) COMP.
      03 FILLER    PIC X(84).
```

```
01 一意名5.
   02 データ名V  PIC X(n).
```

機能

jnlrput コマンド出力ファイルから，（レコード管理情報（一意名 3） ＋ レコードデータヘッダ（一意名 4） ＋ レコードデータ（一意名 5））単位で，ジャーナルデータを順番に入力します。

jnlrput コマンド出力ファイルを CBLDCJUP('OPENRPT ') でオープンしてから，ジャーナルデータを CBLDCJUP('RDGETRPT') で入力してください。

データ名 V には，ジャーナルデータ入力用のバッファの長さを設定します。この長さには，設定したジャーナルデータの長さと同じ値を設定してください。データ名 V の領域の長さがジャーナルデータの長さよりも小さい場合は，CBLDCJUP('RDGETRPT')の動作は保証しません。

データ名 D に設定したジャーナルデータ長が，実際のジャーナルデータ長よりも大きい場合は，ジャーナルデータを入力して，そのジャーナルデータ長がデータ名 E に返されます。

データ名 D に設定したジャーナルデータ長が，実際のジャーナルデータ長よりも小さい場合は，CBLDCJUP('RDGETRPT')はエラーリターンします。

入力エラーが起こった場合は，CBLDCJUP('RDGETRPT')は jnlrput コマンド出力結果ファイルをクローズしてリターンします。

ファイルの終わりに達した場合は，ジャーナルデータは返されません。

CBLDCJUP('RDGETRPT')には，CBLDCJUP('OPENRPT ')で返されたファイル記述子を設定します。

UAP で値を設定するデータ領域

●データ名 A

jnlrput コマンド出力ファイルからデータを入力することを示す要求コードを「VALUE 'RDGETRPT'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

ファイル記述子を設定します。

●データ名 D

入力するジャーナルデータの長さを設定します。設定できる長さは，1 から（取得先システムジャーナルサービス定義の jnl_max_datasize オペランドの値）までです。

OpenTP1 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 E

入力したジャーナルデータのデータ長が返されます。

●データ名 F

一意名 3，一意名 4，一意名 5 を合計した長さが返されます。

●データ名 G

入力したジャーナルデータの，レコード種別が返されます。

'U' … UJ レコードが返されたことを示します。

'S' … SJ レコードが返されたことを示します。

'I' … IJ レコードが返されたことを示します。

'M' … MJ レコードが返されたことを示します。

'O' … OJ レコードが返されたことを示します。

'A' … AJ レコードが返されたことを示します。

'G' … GJ レコードが返されたことを示します。

●データ名 Y

入力したジャーナルデータの，トランザクション ID 設定情報の有無が返されます。

0x00…トランザクション ID 設定情報はありません。

0x80…トランザクション ID 設定情報があります。この場合，データ名 M にレコードトランザクショングローバル識別子，データ名 N にレコードトランザクションブランチ識別子が返されます。

●データ名 H

入力したジャーナルデータの，レコード取得時刻が返されます。

●データ名 I

入力したジャーナルデータの，レコード取得要求元ノード識別子が返されます。

●データ名 J

入力したジャーナルデータの，レコード取得要求元サーバ名が返されます。

●データ名K

入力したジャーナルデータの，レコード取得要求元サーバタイムスタンプが返されます。

●データ名L

入力したジャーナルデータの，レコードユーザ情報が返されます。

●データ名M

入力したジャーナルデータの，レコードトランザクショングローバル識別子が返されます。トランザクション外 UJ の場合，0 が返されます。

●データ名N

入力したジャーナルデータの，レコードトランザクションブランチ識別子が返されます。トランザクション外 UJ の場合，0 が返されます。

●データ名W

入力したジャーナルデータの，レコード取得時刻（マイクロ秒）が返されます。

ただし，TP1/Server Base 06-01 より前のバージョンのジャーナルデータを，jnlrput コマンドで編集出力したファイルを入力した場合は 0 が返されます。

●データ名O

入力した UJ データのデータヘッダ情報が返されます。データ名 O に返される値は，データ名 G に「U」が返されている場合にだけ有効です。

- データ名O1
  入力した UJ データの，データ長が返されます。
- データ名O2
  入力した UJ データの，UJ コードが返されます。

●データ名P

入力した SJ データのデータヘッダ情報が返されます。データ名 P に返される値は，データ名 G に「S」が返されている場合にだけ有効です。

- データ名P1
  入力した SJ データの，子トランザクションブランチを含むトランザクション決着方式が設定されます。
  'C ' … コミット決着を示します。
  'R ' … ロールバック決着を示します。
  'HC' … コマンドでコミット決着したことを示します。
  'HR' … コマンドでロールバック決着したことを示します。
  'HM' … コマンドでミックス決着したことを示します。
  'HH' … コマンドでハザード決着したことを示します。

- データ名 P2
  入力した SJ データの，プロセス種別が返されます。
  'U' … ユーザサーバプロセスで決着したことを示します。
  'R' … 回復プロセスで決着したことを示します。
- データ名 P3
  入力した SJ データの，トランザクションブランチ本体の決着方法が返されます。
  'C' … コミット決着を示します。
  'R' … ロールバック決着を示します。
- データ名 P4
  入力した SJ データの，ブランチ実行時間秒データが返されます。
- データ名 P5
  入力した SJ データの，ブランチ実行時間秒以下データが返されます。
- データ名 P6
  入力した SJ データの，ブランチ同期点処理実行時間秒データが返されます。
- データ名 P7
  入力した SJ データの，ブランチ同期点処理実行時間秒以下データが返されます。
- データ名 P8
  入力した SJ データの，ユーザサーバ名が返されます。
- データ名 P9
  入力した SJ データの，サービス名が返されます。
- データ名 P10
  入力した SJ データの，トランザクション全 CPU 時間（マイクロ秒単位）が返されます。
- データ名 P11
  入力した SJ データの，OpenTP1 の CPU 時間（マイクロ秒単位）が返されます。
- データ名 P12
  入力した SJ データの，UAP の CPU 時間（マイクロ秒単位）が返されます。
- データ名 P13
  入力した SJ データの，TP1/FS/Direct Access の CPU 時間（マイクロ秒単位）が返されます。
- データ名 P14
  入力した SJ データの，TP1/FS/Table Access の CPU 時間（マイクロ秒単位）が返されます。
- データ名 P15
  入力した SJ データの，ISAM/B の CPU 時間（マイクロ秒単位）が返されます。

●データ名 Q

入力した IJ データのデータヘッダ情報が返されます。データ名 Q に返される値は，データ名 G に「I」が返されている場合にだけ有効です。

- データ名 Q1
  入力した IJ データの，入力元論理端末名称が返されます。
- データ名 Q2
  入力した IJ データの，アプリケーション名が返されます。
- データ名 Q3
  入力した IJ データの，メッセージ入力通番が返されます。
- データ名 Q4
  入力した IJ データの，マップ名が返されます。
- データ名 Q5
  入力した IJ データの，入力メッセージ種別が返されます。
- データ名 Q6
  入力した IJ データの，順序識別子が返されます。
- データ名 Q7
  入力した IJ データの，入力メッセージの長さが返されます。

●データ名 R

入力した MJ データのデータヘッダ情報が返されます。データ名 R に返される値は，データ名 G に「M」が返されている場合にだけ有効です。

- データ名 R1
  入力した MJ データの，論理端末名称が返されます。
- データ名 R2
  入力した MJ データの，コネクション名が返されます。
- データ名 R3
  入力した MJ データの，MJ タイプが返されます。
- データ名 R4
  入力した MJ データの，順序識別子が返されます。
- データ名 R5
  入力した MJ データの，メッセージの長さが返されます。

●データ名 S

入力した OJ データのデータヘッダ情報が返されます。データ名 S に返される値は，データ名 G に「O」が返されている場合にだけ有効です。

- データ名 S1
  入力した OJ データの，出力先論理端末名称が返されます。
- データ名 S2
  入力した OJ データの，アプリケーション名が返されます。
- データ名 S3
  入力した OJ データの，出力メッセージ種別が返されます。
- データ名 S4
  入力した OJ データの，出力通番の有無が返されます。
- データ名 S5
  入力した OJ データの，メッセージ出力通番が返されます。
- データ名 S6
  入力した OJ データの，出力メッセージの長さが返されます。

●データ名 T

入力した AJ データのデータヘッダ情報が返されます。データ名 T に返される値は，データ名 G に「A」が返されている場合にだけ有効です。

- データ名 T1
  入力した AJ データの，出力先論理端末名称が返されます。
- データ名 T2
  入力した AJ データの，アプリケーション名が返されます。
- データ名 T3
  入力した AJ データの，出力メッセージ種別が返されます。
- データ名 T4
  入力した AJ データの，出力通番の有無が返されます。
- データ名 T5
  入力した AJ データの，メッセージ出力通番が返されます。

●データ名 U

入力した GJ データのデータヘッダ情報が返されます。データ名 U に返される値は，データ名 G に「G」が返されている場合にだけ有効です。

- データ名 U1
  入力した GJ データの，入力元論理端末名称が返されます。
- データ名 U2
  入力した GJ データの，アプリケーション名が返されます。
- データ名 U3

入力した GJ データの，メッセージ入力通番が返されます。

- **データ名 U4**
  入力した GJ データの，入力メッセージの長さが返されます。

●データ名 V

入力したジャーナルデータが返されます。ジャーナルデータとして有効なデータは，データ名 E に返された長さです。

データ名 G に「U」，「I」，「M」，「O」，「G」が返された場合にだけ，データ名 V の値は有効となります。

ステータスコード

ステータスコード	意味
00000	ジャーナルデータが，正常に設定されました。
01275	ファイルの終わりに達しました。
01101	要求コード（データ名 A）が間違っています。
01271	データ名 C に設定したファイル記述子は，正常にオープンして得られた記述子ではありません。
	jnlrput コマンド出力ファイルをオープンしていません。
01276	データ名 D に，データ長よりも小さい長さを設定しています。
01274	入力エラーが起こりました（読み込みエラー）。

資源の排他制御（CBLDCLCK）

任意のユーザファイル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。

- CBLDCLCK('GET ') − 資源の排他
- CBLDCLCK('RELALL ') − 全資源の排他の解除
- CBLDCLCK('RELNAME ') − 資源名称を指定した排他の解除

資源の排他制御の COBOL-UAP 作成用プログラム（CBLDCLCK）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，資源の排他制御の COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。資源の排他制御（CBLDCLCK）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCLCK.cbl に格納してあります。

CBLDCLCK('GET ')

名称

資源の排他

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCLCK’ 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。データ名 C で示す資源を排他します。排他の管理は，OpenTP1 のトランザクションマネ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（全資源の排他を解除，または資源名称を指定して排他を解除）を呼び出すか，CBLDCLCK('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 バイトの英数字文字列で設定します。OpenTP1 のロックサービスはここで設定した資源名称を基に排他制御します。

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，デッドロック情報，タイムアウト情報，および lckls コマンドの表示が乱れることがあります。

●データ名 D

排他制御モードを，次に示すどちらかで設定します。

VALUE 'PR'…参照します。ほかの UAP からの参照だけを許可し，更新は禁止します。

VALUE 'EX'…更新します。ほかの UAP からの参照・更新を禁止します。

●データ名 E

ほかの UAP と資源を競合した場合に，エラーリターンするか，資源の解放待ちとするかを設定します。次に示すどちらかで設定します。

VALUE 'WAIT△△'…資源が競合した場合，資源の解放待ちにします。

VALUE 'NOWAIT'…資源が競合した場合，エラーリターンします。

●データ名 F

ここで指定する排他をテストとするかどうかを設定します。次に示すどちらかを設定します。

VALUE 'TEST△△'…この排他はテストです。

VALUE 'NOTEST'…この排他はテストではありません。

テストを指定した場合は，CBLDCLCK('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ていないので注意してください。

●データ名 G

「VALUE 'MIGRATE'」を設定します。

OpenTP1 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。
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含みます。
00450	ほかの UAP が，指定した資源を使っています。
00452	デッドロックが起こりました。
00453	OpenTP1 のロックサービス定義で指定した待ち時間のタイムアウトで，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ない UAP から呼び出しています。
00457	OpenTP1 のライブラリとロックサービスのバージョンが一致していません。

CBLDCLCK('RELALL ')

名称

全資源の排他の解除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCLCK’ 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

CBLDCLCK('GET ')で指定した資源をすべて解放します。同期点取得前に排他を解除するときに使います。

排他を掛けたグローバルトランザクションが終了したときには，OpenTP1 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

全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。

OpenTP1 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。
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含みます。
00455	トランザクション処理でない 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では，資源を確保していません。
00457	OpenTP1 のライブラリとロックサービスのバージョンが一致していません。

CBLDCLCK('RELNAME ')

名称

資源名称を指定した排他の解除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCLCK’ 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

CBLDCLCK('GET ')で指定した資源を，資源名称を指定して解放します。同期点取得前に排他を解除するときに使います。

排他を掛けたグローバルトランザクションが終了したときには，OpenTP1 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

資源名称を指定して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

排他を解除する資源の名称を設定します。資源名称は資源を排他するときに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。

OpenTP1 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。
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含みます。
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。
00455	トランザクション処理でない UAP から呼び出しています。
00457	OpenTP1 のライブラリとロックサービスのバージョンが一致していません。

メッセージログの出力（CBLDCLOG）

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。

- CBLDCLOG('PRINT ') − メッセージログの出力

メッセージログの出力の COBOL-UAP 作成用プログラム（CBLDCLOG）は，TP1/Server Base と TP1/LiNK のどちらの UAP でも使えます。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。メッセージログの出力（CBLDCLOG）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCLOG.cbl に格納してあります。

CBLDCLOG('PRINT ')

名称

メッセージログの出力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCLOG’ 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 VALUE ZERO.
   02 データ名C  PIC X(12).
   02 データ名D  PIC X(3).
   02 FILLER    PIC X(1).
   02 データ名E  PIC S9(9) COMP.
01 一意名2.
   02 データ名F  PIC S9(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 に指定した文字列に，OpenTP1 で行ヘッダ，OpenTP1 識別子，日時，要求元ノード名，要求元プログラム ID，メッセージ ID を付けて，メッセージログファイルに出力します。

OpenTP1 では，CBLDCLOG('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。

障害が起こって UAP からメッセージログが出力できない場合でも，CBLDCLOG('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が，メッセージログの抜けはメッセージログファイル中のメッセージログ通番で確認できます。

一つのプロセスから複数回メッセージログを出力した場合は，メッセージログファイルへの出力順序は保証されます。ただし，複数のプロセスから別々にメッセージログを出力した場合は，出力順にメッセージログファイルに出力されない場合があります。

通信障害「01901」，およびログサービス未起動「01905」のエラーが起こった場合は，UAP から出力したメッセージをその UAP プロセス上で編集して，標準エラー出力に出力します。このとき，メッセージの終わりには，次に示すエラー要因を示すコードを付けます。

- E1…ログサービスが未起動であるため，メッセージログファイルに出力できなかったメッセージを示します。
- E2…通信障害のためメッセージログファイルに出力できなかったメッセージを示します。

（例）

KFCA05201-I　SPP1：サービス要求を受け取りました。(E1)
KFCA05410-I　SPP1：更新処理を開始します。(E2)

E1，E2 以外のエラーを検出した場合，OpenTP1 はエラーの原因を示すメッセージログに CBLDCLOG('PRINT ')に指定したメッセージ ID の番号を付けて，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要求を示す要求コードを「VALUE 'PRINT△△△'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

メッセージログごとに付けられる識別子（メッセージ ID）を設定します。「KFCA$n_1n_2n_3n_4n_5$－x」の形式（11 文字）で，最後に空白文字を付けて設定します。UAP から出力する通番（$n_1n_2n_3n_4n_5$ の部分）には，05000 から 06999 までの間の数値を設定します。

●データ名 D

メッセージログを出力した UAP を識別する値（要求元プログラム ID）を，ユーザ任意で設定します。英数字 2 文字で，最後は空白文字で終わらせて設定します。

●データ名 E

UAP から出力したメッセージログを NETM の操作支援端末に出力する場合の，表示色を設定します。次のどれかを設定します。

VALUE '1' … 緑

VALUE '2' … 赤

VALUE '3' … 白

VALUE '4' … 青

VALUE '5' … 紫

VALUE '6' … 空色

VALUE '7' … 黄色

上記以外の数値や空白を設定した場合は，緑が仮定されます。

●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。最大 222 バイト設定できます。

●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する文字列を設定します。

なお，データ名 F で設定した文字列の長さより前にヌル文字のデータがあると，そこまでの情報を文字列として扱います。

●データ名 H

0 を設定します。

OpenTP1 で値を設定するデータ領域

●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。
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。
01901	通信障害が発生したか，または CBLDCRPC('OPEN ')が発行されていません。
01902	メモリが不足しました。
01904	システム定義が間違っています。
01905	メッセージログサービスが稼働していません。
01906	メッセージログサービスがメッセージログに付ける情報を取得したときに，障害が起こりました。

注意事項

ログ出力量が多い場合は，CBLDCLOG('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激しく多くなると，トランザクション処理時間が延びてしまいます。これは，スローダウンの要因になりますので，注意してください。

メッセージ送受信（CBLDCMCF）

メッセージ送受信形態の通信をするときに使うプログラムについて説明します。メッセージ送受信のCOBOL-UAP 作成用プログラムを次に示します。

- CBLDCMCF('ADLTAP ') − アプリケーションに関するタイマ起動要求の削除
- CBLDCMCF('APINFO ') − アプリケーション情報通知
- CBLDCMCF('CLOSE ') − MCF 環境のクローズ
- CBLDCMCF('COMMIT ') − MHP のコミット
- CBLDCMCF('CONTEND ') − 継続問い合わせ応答の終了
- CBLDCMCF('EXECAP ') − アプリケーションプログラムの起動
- CBLDCMCF('MAINLOOP') − MHP のサービス開始
- CBLDCMCF('OPEN ') − MCF 環境のオープン
- CBLDCMCF('RECEIVE ') − メッセージの受信
- CBLDCMCF('RECVSYNC') − 同期型のメッセージの受信※
- CBLDCMCF('REPLY ') − 応答メッセージの送信※
- CBLDCMCF('RESEND ') − メッセージの再送※
- CBLDCMCF('ROLLBACK') − MHP のロールバック
- CBLDCMCF('SEND ') − メッセージの送信※
- CBLDCMCF('SENDRECV') − 同期型のメッセージの送受信※
- CBLDCMCF('SENDSYNC') − 同期型のメッセージの送信※
- CBLDCMCF('TACTCN ') − コネクションの確立※
- CBLDCMCF('TACTLE ') − 論理端末の閉塞解除※
- CBLDCMCF('TDCTCN ') − コネクションの解放※
- CBLDCMCF('TDCTLE ') − 論理端末の閉塞※
- CBLDCMCF('TDLQLE ') − 論理端末の出力キュー削除
- CBLDCMCF('TEMPGET ') − 一時記憶データの受け取り
- CBLDCMCF('TEMPPUT ') − 一時記憶データの更新
- CBLDCMCF('TIMERCAN') − ユーザタイマ監視の取り消し
- CBLDCMCF('TIMERSET') − ユーザタイマ監視の設定
- CBLDCMCF('TLSCN ') − コネクションの状態取得※
- CBLDCMCF('TLSCOM ') − MCF 通信サービスの状態取得

- CBLDCMCF('TLSLE ') − 論理端末の状態取得※
- CBLDCMCF('TLSLN ') − サーバ型コネクションの確立要求の受付状態取得※
- CBLDCMCF('TOFLN ') − サーバ型コネクションの確立要求の受付終了※
- CBLDCMCF('TONLN ') − サーバ型コネクションの確立要求の受付開始※

注※

COBOL-UAP 作成用プログラムの文法については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

メッセージ送受信の COBOL-UAP 作成用プログラム（CBLDCMCF）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，メッセージ送受信の COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。メッセージ送受信（CBLDCMCF）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCMCF.cbl に格納してあります。DCMCF.cbl に格納してある API の使い方は，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編の文法説明を参照してください。

注意事項

データ部（DATA DIVISION）の指定で，一意名の先頭は語長の境界に置かれるようにしてください。

CBLDCMCF('ADLTAP ')

名称

アプリケーションに関するタイマ起動要求の削除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’ADLTAP  ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

指定されたアプリケーションに関するタイマ要求を削除し，アプリケーションの起動を停止します。ただし，ans 型，および cont 型のアプリケーションに関するタイマ要求は削除できません。

UAP で値を設定するデータ領域

●データ名 A

アプリケーションに関するタイマ要求削除を示す要求コードを「VALUE 'ADLTAP△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象のアプリケーションを持つアプリケーション起動サービスのアプリケーション起動プロセス識別子を設定します。設定できる範囲は 1～239 です。

●データ名 F1

起動を停止するアプリケーションのアプリケーション名を設定します。アプリケーション名は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('ADLTAP ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('ADLTAP ')が受け付けられません。
71004	CBLDCMCF('ADLTAP ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71007	指定されたアプリケーション名は登録されていません。
	タイマ起動要求されていないアプリケーション名を指定しています。
	問合せ応答型または継続問い合わせ応答型のアプリケーション名を指定しています。
71009	CBLDCMCF('ADLTAP ')が，該当するアプリケーション起動プロセスではサポートされていません。
71010	指定されたアプリケーションに関するタイマ起動要求の削除を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 以下または 240 以上の値が設定されています。

ステータスコード	意味
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。

CBLDCMCF('APINFO ')

名称

アプリケーション情報通知

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’APINFO  ’.
    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 X(4) VALUE SPACE.
    02  データ名E  PIC X(2).
    02  データ名F  PIC X(2) VALUE SPACE.
    02  データ名G  PIC X(8).
    02  データ名H  PIC X(28) VALUE LOW-VALUE.
01  一意名2.
    02  データ名I  PIC X(4) VALUE SPACE.
    02  データ名J  PIC X(8).
    02  データ名K  PIC X(2).
    02  データ名L  PIC X(2) VALUE SPACE.
    02  データ名M  PIC X(4).
    02  データ名M1 PIC X(1).
    02  データ名M2 PIC X(3) VALUE SPACE.
    02  データ名N  PIC X(31).
    02  データ名N1 PIC X(4).
    02  データ名N2 PIC X(4).
    02  データ名N3 PIC X(1) VALUE SPACE.
    02  データ名O  PIC X(31).
    02  データ名P  PIC X(4).
    02  データ名Q  PIC X(5) VALUE SPACE.
    02  データ名R  PIC 9(9)  COMP.
    02  データ名S  PIC 9(9)  COMP.
    02  データ名T  PIC 9(9)  COMP.
    02  データ名U  PIC X(4).
    02  データ名V  PIC X(1).
    02  データ名W  PIC X(79) VALUE LOW-VALUE.
```

機能

MHP からアプリケーションに関するさまざまな情報を取得できます。

通知の対象となるアプリケーションは，CBLDCMCF('APINFO ')を呼び出した MHP 自身のアプリケーションか，ほかの MHP のアプリケーションです。

CBLDCMCF('APINFO ')が正常終了した場合だけ，アプリケーション情報は有効となります。

UAP で値を設定するデータ領域

●データ名 A

アプリケーション情報通知を示す要求コードを「VALUE 'APINFO△△'」と設定します。

●データ名 C

参照するアプリケーションによって，次のどちらかを設定します。

'SELF'

CBLDCMCF('APINFO ')を呼び出した MHP に関するアプリケーション情報を取得する場合に設定します。

'OTHE'

アプリケーション定義が登録されている MCF 通信サービスのプロセス識別子とアプリケーション名によって，特定のアプリケーションに関するアプリケーション情報を取得する場合に設定します。

●データ名 D

空白を設定します。

●データ名 E

プロセス識別子を設定します。

- データ名 C に'SELF'を設定した場合
  空白を設定します。
- データ名 C に'OTHE'を設定した場合
  参照するアプリケーションの定義が登録されている MCF 通信プロセス識別子，またはアプリケーション起動プロセス識別子を設定します。

●データ名 F

空白を設定します。

●データ名 G

アプリケーション名を設定します。

- データ名 C に'SELF'を設定した場合
  空白を設定します。
- データ名 C に'OTHE'を設定した場合
  参照したいアプリケーション名を設定します。アプリケーション名は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

ただし，エラーイベント名（ERREVT1，ERREVT2，ERREVT3，ERREVT4）を設定した場合，データ名 M1 には，アプリケーション定義の省略時解釈値である非応答型（'N'）が設定されます。

●データ名 H

「LOW-VALUE」を設定します。

●データ名 I，データ名 L，データ名 M2，データ名 N3，データ名 Q

空白を設定します。

●データ名 W

「LOW-VALUE」を設定します。

OpenTP1 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 J

通知対象のアプリケーション名が設定されます。

●データ名 K

通知対象のアプリケーションが登録されている MCF 通信サービスのプロセス識別子が設定されます。

●データ名 M

アプリケーションの閉塞，閉塞解除の状態が設定されます。

'INDA' … 入力閉塞状態

'SCDA' … スケジュール閉塞状態

'DACT' … 入力・スケジュール閉塞状態

'ACT△' … 閉塞解除状態

●データ名 M1

アプリケーションの型が設定されます（MCF アプリケーション定義 mcfaalcap の-n オプションの type オペランドの型が設定されます）。

'A' … 応答型

'N' … 非応答型

'C' … 継続問い合わせ型

ただし，データ名 C に'OTHE'を設定し，データ名 G にエラーイベント名（ERREVT1，ERREVT2，ERREVT3，ERREVT4）を設定した場合は，実際の型が通知されません。アプリケーション定義の省略時解釈値である非応答型（'N'）が設定されます。

●データ名 N

アプリケーションに対応したサービスグループ名が設定されます。

●データ名 N1

サービスグループの閉塞，閉塞解除の状態が設定されます。

'INDA' … 入力閉塞状態

'SCDA' … スケジュール閉塞状態

'DACT' … 入力・スケジュール閉塞状態

'ACT△' … 閉塞解除状態

●データ名 N2

サービスグループの保留，保留解除の状態が設定されます。

'INHO' … 入力保留状態

'SCHO' … スケジュール保留状態

'HOLD' … 入力・スケジュール保留状態

'RLSS' … 保留解除状態

●データ名 O

アプリケーションに対応したサービス名が設定されます。

●データ名 P

サービスの閉塞，閉塞解除の状態がで設定されます。

'INDA' … 入力閉塞状態

'SCDA' … スケジュール閉塞状態

'DACT' … 入力・スケジュール閉塞状態

'ACT△' … 閉塞解除状態

●データ名 R

非トランザクション MHP の限界経過時間が秒単位で設定されます。

ただし，データ名 U が'TRN△'の場合は，0 が設定されます（MCF アプリケーション定義 mcfaalcap の-v オプションの ntmetim オペランドの値が設定されます。ただし，MCF アプリケーションを省略した場合は，MCF マネジャ定義 mcfmuap の-u オプションの ntmetim オペランドの値が設定されます）。

●データ名 S

継続問い合わせ応答用一時記憶データ格納用領域サイズが設定されます。

ただし，データ名 M1 が'C'以外の場合は，0 が設定されます（MCF アプリケーション定義 mcfaalcap の-n オプションの tempsize オペランドの値が設定されます）。

●データ名 T

入力メッセージ最大格納数が設定されます（MCF アプリケーション定義 mcfaalcap の-n オプションの msgcnt オペランドの値が設定されます）。

●データ名 U

アプリケーションのトランザクション属性が設定されます（MCF アプリケーション定義 mcfaalcap の-n オプションの trnmode オペランドの値が設定されます）。

'TRN△' … トランザクションとして管理します。

'NTRN' … トランザクションとして管理しません。

●データ名 V

受信メッセージの割り当て先が設定されます（MCF アプリケーション定義 mcfaalcap の-g オプションの quekind オペランドの値が設定されます）。

'D' … ディスクキューに割り当てる場合。

'M' … メモリキューに割り当てる場合。

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。
72000	MHP サービス以外から呼び出されました。
72001	データ名 G に設定した値が間違っています。 データ名 G とデータ名 E の組み合わせが間違っています。
72016	データ名 C に設定した値が間違っています。 データ名 E に設定した値が間違っています。 データ名 H に設定した値が間違っています。 データ名 W に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。

ステータスコード	意味
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

注意事項

ERREVT1，ERREVT2，ERREVT3，ERREVT4 は，同時に複数起動した場合，同名のエラーイベントであっても異なるアプリケーションの型を持つことがあります。したがって，CBLDCMCF('APINFO ') を呼び出した MHP 以外のエラーイベント（ERREVT1，ERREVT2，ERREVT3，ERREVT4）に対する型の通知は行いません。この場合，アプリケーション定義の省略時解釈値である「非応答型」を通知します。

CBLDCMCF('CLOSE ')

名称

MCF 環境のクローズ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’CLOSE   ’.
   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(12) VALUE LOW-VALUE.
```

機能

MCF の機能を使う環境をクローズします。CBLDCMCF('OPEN ')を呼び出した UAP のメインプログラムで，CBLDCRPC('CLOSE ')を呼び出す前に，プロセスで 1 回だけ呼び出します。

UAP で値を設定するデータ領域

●データ名 A

MHP の終了を示す要求コードを「VALUE 'CLOSE△△△'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

「LOW-VALUE」を設定します。

OpenTP1 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。
72016	データ名 D に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('COMMIT ')

名称

MHP のコミット

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’COMMIT  ’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(16) VALUE LOW-VALUE.
```

機能

MHP から開始するグローバルトランザクションで，ルートトランザクションブランチとして，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ジャに，処理が正常に終了したこと（コミット）を知らせます。

CBLDCMCF('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。

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（CBLDCMCF('COMMIT ')を呼び出した MHP だけでないとき）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かぎりコミットされません。また，複数のリソースマネジャで構成されるときも同様に，それぞれのリソースマネジャの処理結果がコミットとならないかぎりコミットされません。コミットされない場合は，すべてのトランザクションブランチがロールバックされて，ステータスコード 「70906」でエラーリターンします。

CBLDCMCF('COMMIT ')は，MCF アプリケーション定義で非応答型（type=noans）と指定した MHP からだけ呼び出せます。それ以外の型の MHP から呼び出した場合は，ステータスコード 「72000」でエラーリターンします。また，MHP 以外の UAP から呼び出した場合も同様に，ステータスコード 「72000」でエラーリターンします。

UAP で値を設定するデータ領域

●データ名 A

MHP のコミットを示す要求コードを「VALUE 'COMMIT△△'」と設定します。

●データ名 C

「LOW-VALUE」を設定します。

OpenTP1 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。このステータスコードが返った場合，CBLDCMCF('COMMIT ')を呼び出したプロセスでは，新しいトランザクションが開始しています。
70906	トランザクションは，コミットできないでロールバックしました。このステータスコードが返った場合，CBLDCMCF('COMMIT ')を呼び出したプロセスでは，新しいトランザクションが開始しています。
70907	CBLDCMCF('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定のため，あるトランザクションブランチはコミットとなり，また，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が返った場合，CBLDCMCF('COMMIT ')を呼び出したプロセスでは，新しいトランザクションが開始しています。
70908	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完了しました。しかし，障害のため，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同期点の結果が判明しません。このステータスコードが返った場合，CBLDCMCF('COMMIT ')を呼び出したプロセスでは，新しいトランザクションが開始しています。 トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，1 相コミット時にリソースマネジャからのリターン値が XAER_NOTA の場合も，ステータスコード 70908 を返します。
72000	＜ MHP の実行でリターンした場合＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，CBLDCMCF('COMMIT ')を呼び出しています。 非応答型（type=noans）でない MHP から，CBLDCMCF('COMMIT ') を呼び出しています。 非トランザクション属性の MHP から，CBLDCMCF('COMMIT ') を呼び出しています。
	＜ SPP の実行でリターンした場合＞ SPP から，CBLDCMCF('COMMIT ')を呼び出せません。
72028	データ名 A に設定した値が間違っています。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

注意事項

CBLDCMCF('COMMIT ')が正常に終了しても，入力メッセージは入力キューから削除されません。そのため，MHP がスケジュールし直されたあとに該当するメッセージを処理し続ける場合，どの時点までコミットしていたかは不定です。MHP が再びスケジュールされるのは次の場合です。

- MCF イベントが通知されて，MCF イベント処理用 MHP をスケジュールした
- システムが異常終了して，OpenTP1 が該当する処理の MHP を再スケジュールした

スケジュールし直された MHP で該当するメッセージを処理し続ける場合，コミットした範囲はユーザで管理してください。

CBLDCMCF('CONTEND ')

名称

継続問い合わせ応答の終了

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’CONTEND ’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(16) VALUE LOW-VALUE.
```

機能

継続問い合わせ応答の処理を終了します。継続問い合わせ応答を終了するときは，その MHP から呼び出した CBLDCMCF('REPLY ')のデータ名 R が空白であること，および継続問い合わせ応答型のアプリケーションを起動する CBLDCMCF('EXECAP ')を呼び出していないことが前提です。次起動アプリケーションを CBLDCMCF('REPLY ')のデータ名 R に設定したり，継続問い合わせ応答型のアプリケーションを起動する CBLDCMCF('EXECAP ')を呼び出したりした場合は，CBLDCMCF('CONTEND ')はエラーリターンします。

CBLDCMCF('CONTEND ')を呼び出したあとは，一時記憶データにアクセスするプログラム（CBLDCMCF('TEMPGET '），CBLDCMCF('TEMPPUT ')）は使えません。

UAP で値を設定するデータ領域

●データ名 A

継続問い合わせ応答の終了を示す要求コードを「VALUE 'CONTEND△'」と設定します。

●データ名 C

「LOW-VALUE」を設定します。

OpenTP1 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。
72000	＜ MHP の実行でリターンした場合＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，CBLDCMCF('CONTEND ')を呼び出しています。
	＜ SPP の実行でリターンした場合＞ SPP では CBLDCMCF('CONTEND ')を呼び出せません。
72016	データ名 C に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72101	継続問い合わせ応答型でないアプリケーションで，CBLDCMCF('CONTEND ') を呼び出しています。
72107	CBLDCMCF('CONTEND ')を 2 回以上呼び出しています。
72111	次起動アプリケーションを設定して CBLDCMCF('REPLY ')を呼び出したあと，CBLDCMCF('CONTEND ')を呼び出しています。
	継続問い合わせ応答型のアプリケーション名を設定して CBLDCMCF('EXECAP ')を呼び出したあと，CBLDCMCF('CONTEND ')を呼び出しています。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('EXECAP ')

名称

アプリケーションプログラムの起動

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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C  PIC  X(4) VALUE SPACE.
   02 データ名D  PIC  X(4) VALUE SPACE.
   02 データ名E  PIC  9(8).
   02 データ名F  PIC  9(8).
   02 データ名G  PIC  9(9) COMP VALUE ZERO.
   02 データ名H  PIC  X(4).
   02 データ名I  PIC  X(4) VALUE SPACE.
   02 データ名J  PIC  X(4) VALUE SPACE.
   02 データ名K  PIC  X(4) VALUE SPACE.
   02 データ名L  PIC  X(8).
   02 データ名M  PIC  X(4) VALUE SPACE.
   02 データ名N  PIC  X(8).
   02 データ名01 PIC  X(4).
   02 データ名02 PIC  9(9) COMP VALUE ZERO.
   02 データ名03 PIC  9(9) COMP VALUE ZERO.
   02 データ名04 PIC  X(1) VALUE SPACE.
   02 データ名05 PIC  X(1).
   02 データ名P  PIC  X(14) VALUE LOW-VALUE.
01 一意名2.
   02 データ名Q  PIC  X(4) VALUE SPACE.
   02 データ名R  PIC  X(8) VALUE SPACE.
   02 データ名S  PIC  X(8) VALUE SPACE.
   02 データ名T  PIC  X(6) VALUE SPACE.
   02 データ名U  PIC  X(2) VALUE SPACE.
   02 データ名V  PIC  X(28) VALUE LOW-VALUE.
01 一意名3.
   02 データ名W  PIC  9(x) COMP.
   02 データ名X  PIC  X(x).
   02 データ名Y  PIC  X(n).
```

機能

UAP（SPP，または MHP）から，データ名 N に設定したアプリケーション名の MHP を起動させます。トランザクション終了後またはサービス関数終了後，すぐに起動させることも，設定した時間経過後に起

動させることもできます。詳細については，マニュアル「OpenTP1 プログラム作成の手引」の「アプリケーションプログラムの起動」の説明を参照してください。

SPP からアプリケーションプログラムを起動する場合は，SPP がトランザクションとして処理していることと，その SPP のメインプログラムで CBLDCMCF('OPEN ')を呼び出していることが前提です。

MHP から起動させる場合，起動された MHP で受け取るメッセージ入力元の論理端末名称は，最初に受信したメッセージ中の名称になります。さらに，その MHP からアプリケーションプログラムを起動した場合も，受け取るメッセージ入力元の論理端末名称は，最初にメッセージを受信したときの名称が引き渡されます。

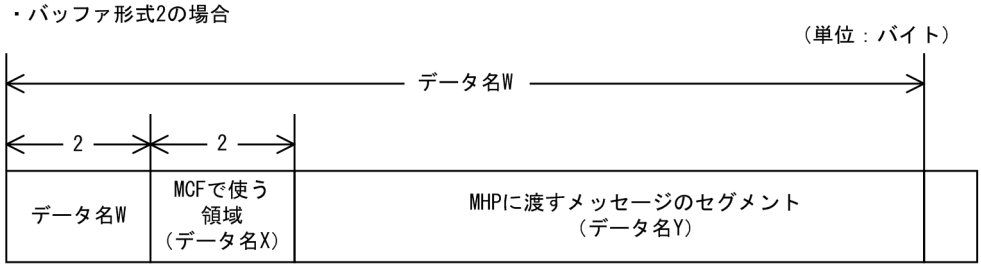
SPP から MHP を起動させる場合，起動された MHP で受け取るメッセージ入力元の論理端末名称は「*」となります。さらに，その MHP からアプリケーションプログラムを起動した場合も，受け取るメッセージ入力元の論理端末名称は「*」となります。

送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

起動させる MHP に渡すセグメントの領域（一意名 3 で示す領域）の形式を次に示します。

・バッファ形式1の場合

・バッファ形式2の場合



UAP で値を設定するデータ領域

●データ名 A

アプリケーションプログラムの起動を示す要求コードを「VALUE 'EXECAP△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E，データ名 F

MCF で使う領域です。

●データ名 G

0 を設定します。

●データ名 H

起動させる MHP に渡すセグメントが，論理メッセージの最終セグメントかどうかを設定します。次のどちらかを設定します。

'ESI△'

先頭セグメント，または中間セグメントを渡すときに設定します。この値を設定した CBLDCMCF('EXECAP ')を呼び出した場合は，そのあとに必ずデータ名 H に「'EMI△'」を設定した CBLDCMCF('EXECAP ')を呼び出してください。

'EMI△'

最終セグメントを渡すときに設定します。

論理メッセージが単一セグメントの場合も「'EMI△'」を設定します。

先頭セグメント，または中間セグメントの送信後，メッセージの送信の終了を連絡する場合にも「'EMI△'」を設定します。

●データ名 I，データ名 J，データ名 K

空白を設定します。

●データ名 L

- 経過時間指定のタイマ起動（データ名 O1 に「'INTV'」を設定）の場合

  CBLDCMCF('EXECAP ')を呼び出してから，何時間何分何秒後に MHP を起動させるかを設定します。時間は「HHMMSS00」（HH：時間，MM：分，SS：秒 00 は固定）の形式で設定します。設定できる時間の範囲は，'00000100'（1 秒後に開始）から '99595900'（99 時間 59 分 59 秒後に開始）までです。

- 時刻指定のタイマ起動（データ名 O1 に「'TIME'」を設定）の場合

  MHP を起動させる時刻を設定します。時刻は「HHMMSS00」（HH：時，MM：分，SS：秒 00 は固定）の形式で設定します。設定できる時刻の範囲は，'00000000'（0 時 0 分 0 秒に開始）から '23595900'（23 時 59 分 59 秒に開始）です。時刻はローカルタイムで指定します。

データ名 L に設定した値は，タイマ起動の場合にだけ有効となります。即時起動の場合は，データ名 L に設定した値は無視されます。

一定の時間間隔で起動時間に達したかどうかを監視するので，データ名 L に設定した時間と実際の起動時間には誤差が生じます。時間監視の精度は，MCF 通信構成定義 mcfttim の-t オプションの btim オペランドに指定した時間監視間隔の値に依存します。

●データ名 M

空白を設定します。

●データ名 N

CBLDCMCF('EXECAP ')を呼び出した MHP が終了したあとに起動する MHP のアプリケーション名を設定します。アプリケーション名は最大 8 バイトです。アプリケーション名の最後には空白を付けます。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 O1

MHP をいつ起動させるかを設定します。

'JUST'
　即時起動の場合に設定します。このとき，データ名 L に設定した値は無視されます。

'INTV'
　経過時間指定のタイマ起動の場合に設定します。CBLDCMCF('EXECAP ')を呼び出してからデータ名 L に設定した時間だけ経過したあとに，MHP を起動させます。

'TIME'
　時刻指定のタイマ起動の場合に設定します。データ名 L に設定した時刻に，MHP を起動させます。

空白を設定
　省略されたものとして，'JUST'（即時起動）が仮定されます。

●データ名 O2，データ名 O3

0 を設定します。

●データ名 O4

空白を設定します。

●データ名 O5

使用するバッファ形式を設定します。

'1'…バッファ形式 1 を使うときに設定します。

'2'…バッファ形式 2 を使うときに設定します。

空白を設定…省略されたものとして，'1'（バッファ形式 1）が仮定されます。

●データ名 P

「LOW-VALUE」を設定します。

●データ名 Q，データ名 R，データ名 S，データ名 T，データ名 U

空白を設定します。

●データ名 V

「LOW-VALUE」を設定します。

●データ名 W

【バッファ形式 1 の場合】 PIC 9（9）

送信セグメント長を設定します。

【バッファ形式 2 の場合】 PIC 9（4）

送信セグメント長＋ 4 を設定します。

先頭セグメント，または中間セグメントの送信後，メッセージの送信の終了を連絡する場合は，送信セグメント長には 0 を設定します。

●データ名 X

【バッファ形式 1 の場合】 PIC X（8）

【バッファ形式 2 の場合】 PIC X（2）

MCF で使用する領域です。

●データ名 Y

起動する MHP に渡す，メッセージのセグメントの内容を設定します。先頭セグメント，または中間セグメントの送信後，メッセージの送信の終了を連絡する場合にも，必ず設定してください。

OpenTP1 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。
71002	メッセージキューへの入出力処理時に障害が起こりました。
	メッセージキューが閉塞されています。
	メッセージキューが割り当てられていません。
	セグメント長に 32000 バイトを超える値を設定しています。
	MCF の終了処理中のため，データ名 N に設定した MHP を起動できません。
71003	メッセージキューが満杯です。
71004	メッセージを格納するバッファをメモリ上に確保できませんでした。
71108	データ名 N に設定したアプリケーション名の MHP を起動しようとしましたが，起動させようとした MHP の管理テーブルが確保できませんでした。
	プロセスのローカルメモリが不足しています。

ステータスコード	意味
72000	＜MHP の実行でリターンした場合＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，CBLDCMCF('EXECAP ')を呼び出しています。
	＜SPP の実行でリターンした場合＞ トランザクションでない SPP の処理から，CBLDCMCF('EXECAP ') を呼び出しています。
72001	設定したアプリケーション名は，MCF で定義していません。
	アプリケーション名が間違っています。
	MCF マネジャ定義の通信サービス定義（mcfmcname 定義コマンド）に，アプリケーション起動プロセス名を指定していません。
	アプリケーション起動プロセスに対応する MCF アプリケーション定義の環境定義（mcfaenv 定義コマンドの -p オプション）に，アプリケーション起動プロセス識別子を指定していません。
	アプリケーション環境定義（mcfaenv 定義コマンドの -p オプション）で指定したアプリケーション起動プロセス識別子と，アプリケーション起動プロセスの通信構成定義（mcftenv 定義コマンド）で指定する識別子が一致していません。
	＜非応答型の MHP を起動する場合＞ • 起動先アプリケーションのアプリケーション属性定義の，論理端末（mcfaalcap 定義コマンドの -n オプションの lname オペランド）に値を指定していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した論理端末を，アプリケーション起動プロセスの通信構成定義（mcftalcle 定義コマンド）に定義していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した論理端末が，一方送信型（mcftalcle -t=send）ではありません。 • 起動先アプリケーションのアプリケーション属性定義で指定した論理端末は，アプリケーション起動を使えません。
	＜応答型および継続問い合わせ応答型の MHP を起動する場合＞ • 起動先アプリケーションのアプリケーション属性定義に，内部通信路（mcfaalcap 定義コマンドの -n オプションの cname オペランド）を指定していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した内部通信路を，アプリケーション起動プロセスの通信構成定義（mcftpsvr 定義コマンドの -c オプション）に定義していません。 • アプリケーション起動プロセスの通信構成定義（mcftalcle 定義コマンド）に，問い合わせ型論理端末（mcftalcle -t=request）を指定していません。
	＜SPP からアプリケーションを起動する場合＞ • アプリケーション起動プロセス識別子を，起動元の UAP のユーザサービス定義 またはユーザサービスデフォルト定義の mcf_psv_id オペランドに指定していません。 • 起動元の UAP のユーザサービス定義，またはユーザサービスデフォルト定義の mcf_psv_id オペランドに指定しているアプリケーション起動プロセス識別子が，アプリケーション起動プロセスの通信構成定義（mcftenv 定義コマンドの -s オプション），およびアプリケーション環境定義（mcfaenv 定義コマンドの -p オプション）で指定しているアプリケーション起動プロセス識別子と一致していません。 • 起動元の UAP のユーザサービス定義，またはユーザサービスデフォルト定義の mcf_mgrid オペランドに指定している MCF マネジャ識別子が，アプリケーション起動プロセスが属している MCF マネジャの識別子と一致していません。

ステータスコード	意味
72005	データ名 H に「'ESI△'」を設定した CBLDCMCF('EXECAP ')で，メッセージのセグメント長（データ名 W）に 1 バイト未満の値（バッファ形式 1 の場合），または 5 バイト未満の値（バッファ形式 2 の場合）を設定しています。
72007	CBLDCMCF('REPLY ')をすでに呼び出した 応答型（type=ans）の MHP から，応答型の MHP を CBLDCMCF('EXECAP ')で起動させています。
	CBLDCMCF('REPLY ')をすでに呼び出した 継続問い合わせ応答型（type=cont）の MHP から，継続問い合わせ応答型の MHP を CBLDCMCF('EXECAP ')で起動させています。
72009	応答型（type=ans）の MHP から，CBLDCMCF('EXECAP ')で応答型の MHP を 2 回以上起動させています。
	継続問い合わせ応答型（type=cont）の MHP から，CBLDCMCF('EXECAP ')で継続問い合わせ応答型の MHP を 2 回以上起動させています。
72011	応答型（type=ans）でない MHP から，CBLDCMCF('EXECAP ')で応答型の MHP を起動させています。
	継続問い合わせ応答型（type=cont）でない MHP から，CBLDCMCF('EXECAP ')で継続問い合わせ応答型の MHP を起動させています。
72016	データ名 O1，データ名 O2，データ名 O3，データ名 P，データ名 V に設定した値が間違っています。
72024	データ名 Q に設定した値が間違っています。
72026	データ名 H のセグメント種別（最終セグメント 'EMI△'，または最終セグメント以外 'ESI△'）に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72041	＜メッセージが単一セグメントの場合＞ バッファ形式 1 の場合，データ名 W に 0 バイト，またはマイナス値を設定しています。 バッファ形式 2 の場合，データ名 W に 0 から 4 バイト，またはマイナス値を設定しています。
	＜メッセージが複数セグメントの場合＞ データ名 H に「'ESI△'」を設定した CBLDCMCF('EXECAP ')を呼び出さないで，メッセージの送信の終了を連絡しています。
72044	CBLDCMCF('CONTEND ')をすでに呼び出した 継続問い合わせ応答型（type=cont）の MHP から，継続問い合わせ応答型の MHP を CBLDCMCF('EXECAP ')で起動させています。
72108	データ名 L に設定できる範囲の値を超えています。
72109	継続問い合わせ応答型（type=cont）の MHP を，タイマ起動で起動させようとしています。
77001	起動しようとしたアプリケーションに対応する論理端末（LE）は，処理中で使えません。または使える論理端末がありません。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

注意事項

1. MCF マネジャ定義の UAP 共通定義（mcfmuap -c order）の指定によって，アプリケーションプログラムの起動順序が異なりますので注意してください。
2. 一つのサービス関数で，TAM または DAM ファイルの更新と CBLDCMCF('EXECAP ')の呼び出しをして，起動先のアプリケーションで更新後の TAM または DAM ファイルを参照させたい場合，TAM または DAM ファイルを排他ありで参照してください。排他なしで参照した場合，更新前のデータが入力されることがあります。

CBLDCMCF('MAINLOOP')

名称

MHP のサービス開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’MAINLOOP’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(16) VALUE LOW-VALUE.
```

機能

アプリケーション名に該当するサービスプログラムへのスケジュールを受け付けます。
CBLDCMCF('MAINLOOP')は，MHP のメインプログラムから呼び出します。プロセスで 1 回だけ呼び出してください。

CBLDCMCF('MAINLOOP')を呼び出してから，OpenTP1 から MHP の終了が要求されるまで，CBLDCMCF('MAINLOOP')はリターンしません。

UAP で値を設定するデータ領域

●データ名 A

MHP のサービス開始を示す要求コードを「VALUE 'MAINLOOP'」と設定します。

●データ名 C

「LOW-VALUE」を設定します。

OpenTP1 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	OpenTP1 から MHP の終了が要求されました。MHP はすぐにプロセスの終了処理をして，CBLDCMCF('CLOSE ')と CBLDCRPC('CLOSE ')を呼び出してから，メインプログラムを STOP RUN で終了してください。
70900	データ名に設定した値が間違っています。
70901	CBLDCMCF('MAINLOOP')を呼び出す前に，CBLDCRPC('OPEN ')を呼び出していません。
70902	サービスを開始できませんでした。
70903	ローカルメモリが不足しました。
72016	データ名 C に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。

CBLDCMCF('OPEN ')

名称

MCF 環境のオープン

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’OPEN    ’.
   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(12) VALUE LOW-VALUE.
```

機能

MCF の機能を使う環境をオープンします。MCF のプログラム（CBLDCMCF）を使う UAP のメインプログラムでは，必ず CBLDCMCF('OPEN ')を呼び出してください。

CBLDCMCF('OPEN ')は，CBLDCRPC('OPEN ')を呼び出したあとに，サービスを開始するプログラム（MHP の場合は CBLDCMCF('MAINLOOP')，SPP の場合は CBLDCRPC('MAINLOOP')）を呼び出す前に，プロセスで 1 回だけ呼び出します。

CBLDCMCF('OPEN ')を呼び出す位置を次に示します。

```
CALL  ’CBLDCRPC’(OPEN)
CALL  ’CBLDCMCF’(OPEN)
CALL  ’CBLDCMCF’(MAINLOOP) (SPPの場合は，CALL ’CBLDCRPC’(MAINLOOP))
      :
      :
CALL  ’CBLDCMCF’(CLOSE)
CALL  ’CBLDCRPC’(CLOSE)
```

UAP で値を設定するデータ領域

●データ名 A

MCF 環境のオープンを示す要求コードを「VALUE 'OPEN△△△△'」と設定します。

OpenTP1 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

0 を設定します。

●データ名 D

「LOW-VALUE」を設定します。

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。
70900	データ名 C に設定した値が間違っています。
70901	CBLDCRPC('OPEN ')を呼び出していません。
	CBLDCMCF('OPEN ')はすでに呼び出しています。
70902	初期化処理に失敗しました。
70903	メモリが不足しました。
72016	データ名 D に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。

CBLDCMCF('RECEIVE ')

名称

メッセージの受信

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’RECEIVE ’.
   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 X(4) VALUE SPACE.
   02 データ名E  PIC 9(8).
   02 データ名F  PIC 9(8).
   02 データ名G  PIC 9(9) COMP.
   02 データ名H  PIC X(4) VALUE SPACE.
   02 データ名I  PIC X(4) VALUE SPACE.
   02 データ名J  PIC X(4) VALUE SPACE.
   02 データ名K  PIC X(4) VALUE SPACE.
   02 データ名L  PIC X(8) VALUE SPACE.
   02 データ名M1 PIC X(4) VALUE SPACE.
   02 データ名M2 PIC X(8) VALUE SPACE.
   02 データ名M3 PIC X(4) VALUE SPACE.
   02 データ名M4 PIC 9(9) COMP VALUE ZERO.
   02 データ名M5 PIC 9(9) COMP VALUE ZERO.
   02 データ名M6 PIC X(1) VALUE SPACE.
   02 データ名M7 PIC X(1).
   02 データ名N  PIC X(14) VALUE LOW-VALUE.
01 一意名2.
   02 データ名O  PIC X(4) VALUE SPACE.
   02 データ名P  PIC X(8).
   02 データ名Q  PIC X(8).
   02 データ名R  PIC X(8) VALUE SPACE.
   02 データ名T  PIC X(28) VALUE LOW-VALUE.
01 一意名3.
   02 データ名U  PIC 9(x) COMP.
   02 データ名V  PIC X(x).
   02 データ名W  PIC X(n).
```

機能

メッセージのうち，一つのセグメントを受け取ります。一つの論理メッセージを受信する場合は，セグメントの数だけ CBLDCMCF('RECEIVE ')を呼び出します。

CBLDCMCF('RECEIVE ')では，次に示すメッセージを受信できます。

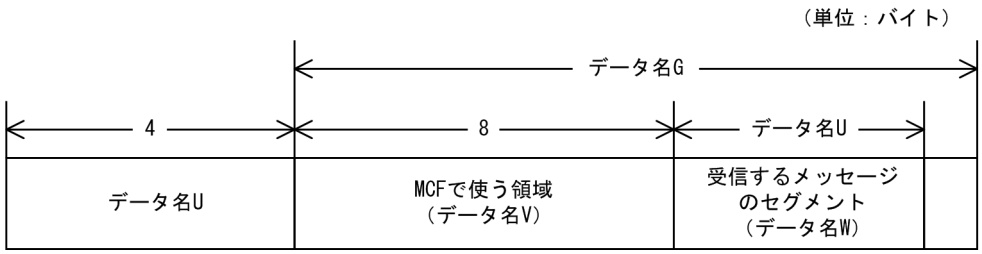
- 相手システムから通信プロトコルを介して送られたメッセージ
- 自システムから通知された MCF イベント
- 自システムの UAP からアプリケーション起動（CBLDCMCF('EXECAP ')）で送られたメッセージ
- 自システムで mcfuevt コマンドを実行して送られたメッセージ
- ユーザタイマ監視を設定したときに指定したメッセージ

相手システムから通信プロトコルを介して送られたメッセージを受信する場合，通信プロトコルによってCBLDCMCF('RECEIVE ')の文法が異なります。相手システムからのメッセージを受信するCBLDCMCF('RECEIVE ')の文法については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

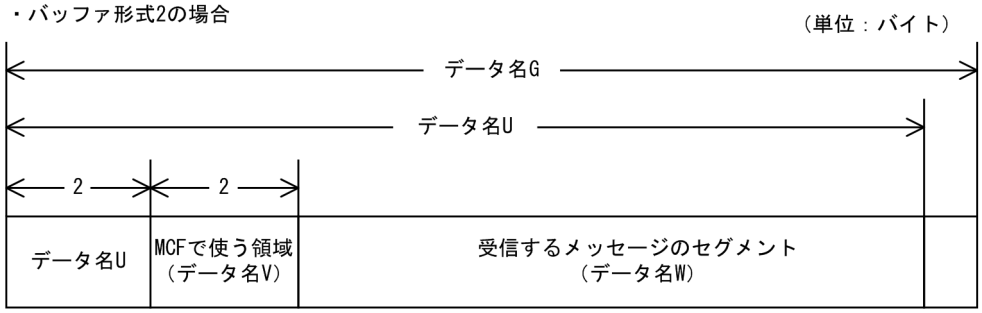
受信できる一つのセグメントの最大長は，1 メガバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

メッセージを受信する領域（一意名 3 で示す領域）の形式を次に示します。

・バッファ形式1の場合



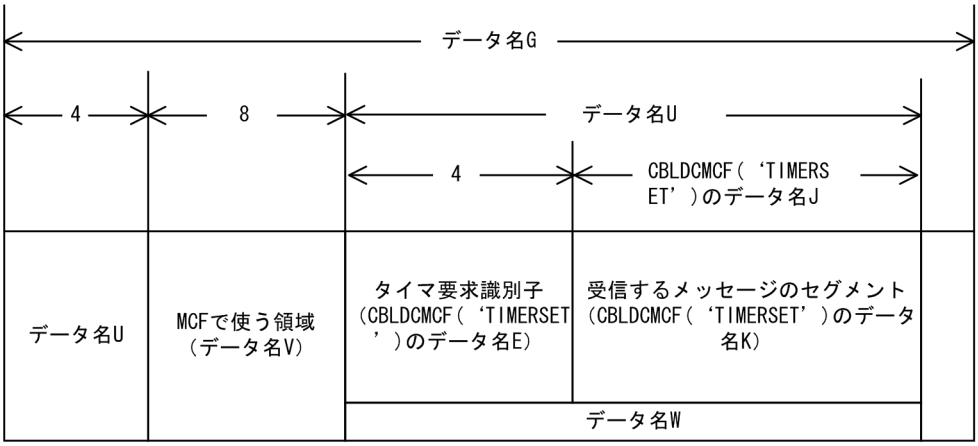
・バッファ形式2の場合



また，ユーザタイマ監視を設定したときに指定したメッセージを受信する領域（一意名 3 で示す領域）の形式を次に示します。

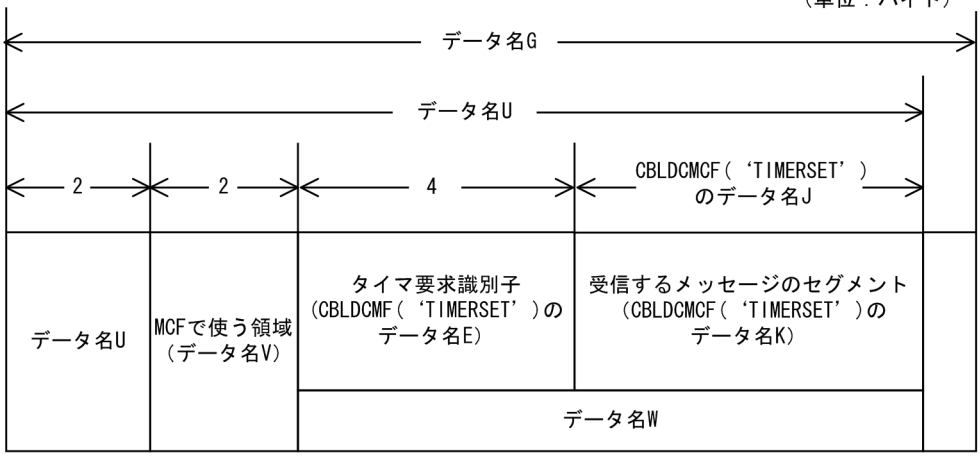
・バッファ形式1の場合

（単位：バイト）



・バッファ形式2の場合

（単位：バイト）



UAP で値を設定するデータ領域

●データ名 A

メッセージの受信を示す要求コードを「VALUE 'RECEIVE△'」と設定します。

●データ名 C

メッセージの先頭セグメントを受信するかどうかを設定します。次のどちらかを設定します。

'FRST'

先頭セグメントを受信するときに設定します。

メッセージが単一セグメントの場合も，「'FRST'」を設定します。

'SEG△'

中間セグメントおよび最終セグメントを受信するときに設定します。

●データ名 D

空白を設定します。

●データ名 G

セグメントを受信する領域の長さを設定します。

●データ名 H，データ名 I，データ名 J，データ名 K，データ名 L，データ名 M1，データ名 M2，データ名 M3

空白を設定します。

●データ名 M4，データ名 M5

0 を設定します。

●データ名 M6

空白を設定します。

●データ名 M7

使うバッファ形式を設定します。

'1'…バッファ形式 1 を使うときに設定します。通常は，バッファ形式 1 を使います。

'2'… バッファ形式 2 を使うときに設定します。

空白を設定…省略されたものとして，'1'（バッファ形式 1）が仮定されます。

●データ名 N

「LOW-VALUE」を設定します。

●データ名 O

空白を設定します。

●データ名 P【中間セグメント，最終セグメントを受信するとき】

入力元の論理端末名称を設定します。先頭セグメントを受信したときに返された論理端末名称を設定します。論理端末名称は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 Q

MCF で使う領域です。

●データ名 R

空白を設定します。

●データ名 T

「LOW-VALUE」を設定します。

●データ名 V

【バッファ形式 1 の場合】 PIC X（8）

【バッファ形式 2 の場合】 PIC X（2）

MCF で使う領域です。

OpenTP1 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 E

メッセージを受信した日付が「YYYYMMDD」（YYYY：西暦の年 MM：月 DD：日）の形式で返されます。

●データ名 F

メッセージを受信した時刻が「HHMMSS00」（HH：時 MM：分 SS：秒 00 は固定）の形式で返されます。

●データ名 P【先頭セグメントを受信するとき】

入力元の論理端末名称が返されます。

中間セグメントおよび最終セグメントを受信するときは，ここで返された論理端末名称をデータ名 P に設定します。

●データ名 U

【バッファ形式 1 の場合】 PIC 9（9）

受信したセグメントの長さが返されます。

【バッファ形式 2 の場合】 PIC 9（4）

受信したセグメントの長さ＋ 4 が返されます。

●データ名 W

受信したセグメントの内容が返されます。

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。
71000	先頭セグメントを受信する CBLDCMCF('RECEIVE ')を 2 回以上呼び出しています。中間セグメントおよび最終セグメントを受信する場合は，データ名 C に「'SEG '」を設定して CBLDCMCF('RECEIVE ')を呼び出してください。

ステータスコード	意味
71001	メッセージの最終セグメントを受信したあとで，次のセグメントを受信する CBLDCMCF('RECEIVE ')を呼び出しています。直前に呼び出した CBLDCMCF('RECEIVE ')でメッセージはすべて受信しました。 このステータスコードが返されたあとに，再び CBLDCMCF('RECEIVE ')を呼び出した場合は，ステータスコード「72000」が返されます。
71002	メッセージキューへの入力処理時に障害が起こりました。
	メッセージキューが閉塞されています。
71108	プロセスのローカルメモリが不足しています。
72000	＜ MHP の実行でリターンした場合＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，中間セグメントおよび最終セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出しています。先頭セグメントを受信する場合は，データ名 C に「'FRST'」を設定して CBLDCMCF('RECEIVE ')を呼び出してください。 ステータスコード「71001」が返されたあとに，再び CBLDCMCF('RECEIVE ')を呼び出しています。
	＜ SPP の実行でリターンした場合＞ SPP では CBLDCMCF('RECEIVE ')を呼び出せません。
72001	データ名 P に設定した論理端末名称が間違っています。
72013	受信領域の長さを超えるセグメントを受信しました。受信領域の長さを超えた部分は切り捨てられました。
	バッファ形式 2 の場合で，32767 バイトを超えるセグメントを受信しました。32767 バイトを超えた部分は切り捨てられました。
72016	データ名 D に設定した値が間違っています。
	データ名 N またはデータ名 T に設定した値が間違っています。
	データ名 M7 に設定した値が間違っています。
72024	データ名 O に設定した値が間違っています。
72025	データ名 C に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72036	セグメントを受信する領域の長さが不足しています。バッファ形式 1 の場合は 9 バイト以上，バッファ形式 2 の場合は 5 バイト以上の領域を確保してください。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('RECVSYNC')

名称

同期型のメッセージの受信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムから送信された論理メッセージを，稼働中の UAP で受信します。

受信できる一つのセグメントの最大長は，1 メガバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

CBLDCMCF('REPLY ')

名称

応答メッセージの送信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムへ，応答のために論理メッセージを送信します。

応答メッセージは，アプリケーションの型が ans 型，cont 型，または noans 型の MHP から呼び出せます。noans 型の MHP から呼び出す場合は，非応答型の MHP からの問い合わせ応答をするための設定（UAP 共通定義（mcfmuap）の-c オプションで noansreply オペランドに yes を指定）が必要です。

送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

CBLDCMCF('RESEND ')

名称

メッセージの再送

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムへ，送信済みの論理メッセージを再び送信します。再送するメッセージは，送信済みのメッセージとは別の，新しいメッセージとして扱います。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

注意事項

MCF マネジャ定義の UAP 共通定義（mcfmuap -c order）の指定によって，メッセージの再送順序が異なりますので注意してください。

CBLDCMCF('ROLLBACK')

名称

MHP のロールバック

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’ROLLBACK’.
   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 X(12) VALUE LOW-VALUE.
```

機能

トランザクション属性を定義した MHP のサービスプログラムの開始から，ここまでの処理を取り消します。データ名 C に「'RTRY'」を設定した場合は，MHP の開始からロールバックまでの処理を取り消して，取り消した処理の MHP をスケジュールし直します。

UAP で値を設定するデータ領域

●データ名 A

部分回復を示す要求コードを「VALUE 'ROLLBACK'」と設定します。

●データ名 C

部分回復の種別を設定します。次のどれか一つを設定します。

'RTRY'

MHP を開始した時点から，ロールバックまでの処理を取り消して，MHP を異常終了します。取り消した処理はスケジュールし直します（受信メッセージを，該当する入力キューの最後に格納し，MHP をスケジュールし直します）。

MHP は異常終了しますが，UAP 異常終了通知イベント（ERREVT3）は起動しません。また，アプリケーション異常終了限界回数にはカウントしません。アプリケーション属性定義（mcfaalcap）で異常終了時に閉塞する指定をしていても，アプリケーション，サービスグループ，およびサービスは閉塞しません。

'RTN△'

MHP を開始した時点から，ロールバックまでの処理を取り消して，リターンします。「'RTN△'」を設定した CBLDCMCF('ROLLBACK')が正常リターンしたあとの処理は，別のトランザクションとして処理します。

'NRTN'

MHP を開始した時点から，ロールバックまでの処理を取り消します。CBLDCMCF('ROLLBACK')からリターンしないで，MHP を異常終了します。

このとき，UAP 異常終了通知イベント（ERREVT3）を起動します。また，アプリケーション異常終了限界回数にカウントします。アプリケーション属性定義（mcfaalcap）の指定によっては，アプリケーション，サービスグループ，およびサービスを閉塞します。

設定する値とシステムの動作の関係を，次の表に示します。

表 2-2　設定する値とシステムの動作の関係（CBLDCMCF('ROLLBACK')）

設定値	MHP の動作	ERREVT3 の起動	各種閉塞処理
'RTRY'	異常終了します。	起動しません。	閉塞しません。
'RTN△'	CALL 文がリターンします。	起動しません。	閉塞しません。
'NRTN'	異常終了します。	起動します。	アプリケーション属性定義（mcfaalcap）の指定によっては，アプリケーション，サービスグループ，およびサービスが閉塞することがあります。

●データ名 D

「LOW-VALUE」を設定します。

OpenTP1 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。
72000	＜MHP の実行でリターンした場合＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，データ名 C に「'RTN△'」を設定した CBLDCMCF('ROLLBACK')を呼び出しています。 非トランザクション属性の MHP から，CBLDCMCF('ROLLBACK') を呼び出しています。
	＜SPP の実行でリターンした場合＞ SPP では，CBLDCMCF('ROLLBACK')を呼び出せません。

ステータスコード	意味
72016	データ名 D に設定した値が間違っています。
72027	部分回復種別（データ名 C）に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('SEND ')

名称

メッセージの送信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムへ，論理メッセージを送信します。

送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

注意事項

MCF マネジャ定義の UAP 共通定義（mcfmuap -c order）の指定によって，メッセージの送信順序が異なりますので注意してください。

CBLDCMCF('SENDRECV')

名称

同期型のメッセージの送受信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムへ，UAP から論理メッセージを送信して，応答を受信します。UAP では，論理メッセージの送信から応答を受信するまでの間，処理を待ちます。応答を受信してから，続きの処理をします。

受信できる一つのセグメントの最大長は，1 メガバイトまでです。また，送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

CBLDCMCF('SENDSYNC')

名称

同期型のメッセージの送信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

相手システムへ，稼働中の UAP から論理メッセージを送信します。UAP では，MCF がメッセージの送信を完了するまで待ちます。送信が完了してから続きの処理をします。

送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

データ領域に設定する値とステータスコードは，使う通信プロトコルによって異なります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

CBLDCMCF('TACTCN ')

名称

コネクションの確立

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TACTCN ’.
   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D1  PIC X(1) VALUE SPACE.
   02 データ名D2  PIC X(1).
   02 データ名D3  PIC X(26) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K1  PIC 9(9)  COMP.
   02 データ名K2  PIC X(n).

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

コネクションを確立します。

なお，CBLDCMCF('TACTCN ')の正常終了は，コネクション確立要求をプロトコル製品が正常に受け付けたことを意味します。このため，相手システムとのコネクションの確立が正常に完了したことを示すものではありません。

CBLDCMCF('TACTCN ')の呼び出し後にコネクションに関する何らかの処理をする場合は，CBLDCMCF('TLSCN ')を用いてコネクションの状態を確認してください。

UAP で値を設定するデータ領域

●データ名 A

コネクション確立を示す要求コードを「VALUE 'TACTCN△△'」と設定します。

●データ名 C

確立するコネクションの指定方法を設定します。

'LE△△'

確立するコネクションを論理端末名称で指定するときに設定します。TP1/NET/NCSB，および TP1/NET/X25-Extended では指定できません。

'CN△△'

確立するコネクションをコネクション ID で指定するときに設定します。

空白

省略されたものとして，'LE△△'（論理端末名称指定）が仮定されます。

●データ名 D1

空白を設定します。

●データ名 D2

通信プロトコルに依存する機能を使用するかどうかを設定します。

'1'

通信プロトコルに依存する機能を使用するときに設定します。

'0'

通信プロトコルに依存する機能を使用しないときに設定します。

空白

省略されたものとして，'0'（使用しない）が仮定されます。

●データ名 D3

空白を設定します。

●データ名 E

処理対象のコネクションを持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

論理端末名称を使用してコネクションの確立を要求する場合は，無効となります。

0 を指定すると，該当するコネクション ID が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

確立するコネクションの論理端末名称，またはコネクション ID を設定します。論理端末名称，またはコネクション ID は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K1

プロトコル固有領域の長さを設定します。プロトコル固有領域は 1024 バイトまで設定できます。

通信プロトコルに依存する機能を使用しないときは，0 を設定します。

●データ名 K2

プロトコル固有領域の内容を設定します。

設定する値は使用する通信プロトコルによって異なります。詳細は，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

●データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TACTCN ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TACTCN ')が受け付けられません。
71004	CBLDCMCF('TACTCN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71007	指定されたコネクション名は登録されていません。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TACTCN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにコネクションの確立を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	コネクションが削除されているため，CBLDCMCF('TACTCN ')が受け付けられません。

ステータスコード	意味
71014	TP1/NET/NCSB，もしくは TP1/NET/X25-Extended の論理端末名称を指定しています。または，TP1/NET/OSI-TP，もしくは TP1/NET/TCP/IP のコネクショングループ名を指定しています。
72028	データ名 A に設定した値が間違っています。
72052	＜データ名 D2 に'0'，または空白を設定＞ データ名 K1 に 0 でない値が設定されています。
	＜データ名 D2 に'1'を設定＞ データ名 K1 に 0 未満，または 1025 以上の値を設定しています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に'LE△△'，'CN△△'，または空白以外が設定されています。
72059	データ名 D1 またはデータ名 D3 に空白でない値が設定されています。
	データ名 D2 に 1，0，または空白以外の値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 に空白が設定されています。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。

CBLDCMCF('TACTLE ')

名称

論理端末の閉塞解除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TACTLE  ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D1  PIC X(1) VALUE SPACE.
   02 データ名D2  PIC X(1).
   02 データ名D3  PIC X(26) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K1  PIC 9(9)  COMP.
   02 データ名K2  PIC X(n).

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

論理端末の閉塞を解除します。

なお，CBLDCMCF('TACTLE ')の正常終了は，論理端末の閉塞解除要求をプロトコル製品が正常に受け付けたことを意味します。このため，論理端末の閉塞解除が正常に完了したことを示すものではありません。

CBLDCMCF('TACTLE ')の呼び出し後に論理端末に関する何らかの処理をする場合は，CBLDCMCF('TLSLE ')を用いて論理端末の状態を確認してください。

UAP で値を設定するデータ領域

●データ名 A

論理端末の閉塞の解除を示す要求コードを「VALUE 'TACTLE△△'」と設定します。

●データ名 C，データ名 D1

空白を設定します。

●データ名 D2

通信プロトコルに依存する機能を使用するかどうかを設定します。

'1'
　通信プロトコルに依存する機能を使用するときに設定します。

'0'
　通信プロトコルに依存する機能を使用しないときに設定します。

空白
　省略されたものとして，'0'（使用しない）が仮定されます。

●データ名 D3

空白を設定します。

●データ名 E

処理対象の論理端末を持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

0 を指定すると，該当する論理端末名称が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

閉塞解除する論理端末の論理端末名称を設定します。論理端末名称は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K1

プロトコル固有領域の長さを設定します。プロトコル固有領域は 1024 バイトまで設定できます。

通信プロトコルに依存する機能を使用しないときは，0 を設定します。

●データ名 K2

プロトコル固有領域の内容を設定します。

設定する値は使用する通信プロトコルによって異なります。詳細は，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

●データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TACTLE ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TACTLE ')が受け付けられません。
71004	CBLDCMCF('TACTLE ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TACTLE ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスに論理端末の閉塞の解除を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	論理端末が削除されているため，CBLDCMCF('TACTLE ')が受け付けられません。
72028	データ名 A に設定した値が間違っています。
72052	＜データ名 D2 に'0'，または空白を設定＞ データ名 K1 に 0 でない値が設定されています。
	＜データ名 D2 に'1'を設定＞ データ名 K1 に 0 未満，または 1025 以上の値を設定しています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D1 またはデータ名 D3 に空白でない値が設定されています。
	データ名 D2 に 1，0，または空白以外の値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。

ステータスコード	意味
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。

CBLDCMCF('TDCTCN ')

名称

コネクションの解放

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TDCTCN  ’.
   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D1  PIC X(1).
   02 データ名D2  PIC X(1).
   02 データ名D3  PIC X(26) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K1  PIC 9(9)  COMP.
   02 データ名K2  PIC X(n).

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

コネクションを解放します。

なお，CBLDCMCF('TDCTCN ')の正常終了は，コネクション解放要求をプロトコル製品が正常に受け付けたことを意味します。このため，相手システムとのコネクションの解放が正常に完了したことを示すものではありません。

CBLDCMCF('TDCTCN ')の呼び出し後にコネクションに関する何らかの処理をする場合は，CBLDCMCF('TLSCN ')を用いてコネクションの状態を確認してください。

UAP で値を設定するデータ領域

●データ名 A

コネクション解放を示す要求コードを「VALUE 'TDCTCN△△'」と設定します。

●データ名 C

解放するコネクションの指定方法を設定します。

'LE△△'

解放するコネクションを論理端末名称で指定するときに設定します。TP1/NET/NCSB，および TP1/NET/X25-Extended では指定できません。

'CN△△'

解放するコネクションをコネクション ID で指定するときに設定します。

空白

省略されたものとして，'LE△△'（論理端末名称指定）が仮定されます。

●データ名 D1

コネクションを強制的に解放するかどうかを設定します。

'1'

コネクションを強制的に解放します。

'0'

コネクションを正常に解放します。

空白

省略されたものとして，'0'（正常解放）が仮定されます。

●データ名 D2

通信プロトコルに依存する機能を使用するかどうかを設定します。

'1'

通信プロトコルに依存する機能を使用するときに設定します。

'0'

通信プロトコルに依存する機能を使用しないときに設定します。

空白

省略されたものとして，'0'（使用しない）が仮定されます。

●データ名 D3

空白を設定します。

●データ名 E

処理対象のコネクションを持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

論理端末名称を使用してコネクションの解放を要求する場合は，無効となります。

0 を指定すると，該当するコネクション ID が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

解放するコネクションの論理端末名称，またはコネクション ID を設定します。論理端末名称，またはコネクション ID は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K1

プロトコル固有領域の長さを設定します。プロトコル固有領域は 1024 バイトまで設定できます。

通信プロトコルに依存する機能を使用しないときは，0 を設定します。

●データ名 K2

プロトコル固有領域の内容を設定します。

設定する値は使用する通信プロトコルによって異なります。詳細は，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

●データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TDCTCN ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TDCTCN ')が受け付けられません。
71004	CBLDCMCF('TDCTCN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71007	指定されたコネクション名は登録されていません。

ステータスコード	意味
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TDCTCN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにコネクションの解放を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	コネクションが削除されているため，CBLDCMCF('TDCTCN ')が受け付けられません。
71014	TP1/NET/NCSB，もしくは TP1/NET/X25-Extended の論理端末名称を指定しています。または，TP1/NET/OSI-TP，もしくは TP1/NET/TCP/IP のコネクショングループ名を指定しています。
72028	データ名 A に設定した値が間違っています。
72052	＜データ名 D2 に'0'，または空白を設定＞ データ名 K1 に 0 でない値が設定されています。
	＜データ名 D2 に'1'を設定＞ データ名 K1 に 0 未満，または 1025 以上の値を設定しています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に'LE△△'，'CN△△'，または空白以外が設定されています。
72059	データ名 D2 に 1，0，または空白以外の値が設定されています。
	データ名 D3 に空白でない値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。
72075	データ名 D1 に 1，0，または空白以外の値が設定されています。

CBLDCMCF('TDCTLE ')

名称

論理端末の閉塞

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TDCTLE  ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D1  PIC X(1) VALUE SPACE.
   02 データ名D2  PIC X(1).
   02 データ名D3  PIC X(26) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K1  PIC 9(9)  COMP.
   02 データ名K2  PIC X(n).

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

論理端末を閉塞します。

なお，CBLDCMCF('TDCTLE ')の正常終了は，論理端末の閉塞要求をプロトコル製品が正常に受け付けたことを意味します。このため，論理端末の閉塞が正常に完了したことを示すものではありません。

CBLDCMCF('TDCTLE ')の呼び出し後に論理端末に関する何らかの処理をする場合は，CBLDCMCF('TLSLE ')を用いて論理端末の状態を確認してください。

UAP で値を設定するデータ領域

●データ名 A

論理端末の閉塞を示す要求コードを「VALUE 'TDCTLE△△'」と設定します。

●データ名 C，データ名 D1

空白を設定します。

●データ名 D2

通信プロトコルに依存する機能を使用するかどうかを設定します。

'1'
　通信プロトコルに依存する機能を使用するときに設定します。

'0'
　通信プロトコルに依存する機能を使用しないときに設定します。

空白
　省略されたものとして，'0'（使用しない）が仮定されます。

●データ名 D3

空白を設定します。

●データ名 E

処理対象の論理端末を持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

0 を指定すると，該当する論理端末名称が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

閉塞する論理端末の論理端末名称を設定します。論理端末名称は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K1

プロトコル固有領域の長さを設定します。プロトコル固有領域は 1024 バイトまで設定できます。

通信プロトコルに依存する機能を使用しないときは，0 を設定します。

●データ名 K2

プロトコル固有領域の内容を設定します。

設定する値は使用する通信プロトコルによって異なります。詳細は，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

●データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TDCTLE ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TDCTLE ')が受け付けられません。
71004	CBLDCMCF('TDCTLE ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TDCTLE ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスに論理端末の閉塞を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	論理端末が削除されているため，CBLDCMCF('TDCTLE ')が受け付けられません。
72028	データ名 A に設定した値が間違っています。
72052	＜データ名 D2 に'0'，または空白を設定＞ データ名 K1 に 0 でない値が設定されています。
	＜データ名 D2 に'1'を設定＞ データ名 K1 に 0 未満，または 1025 以上の値を設定しています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D2 に 1，0，または空白以外の値が設定されています。
	データ名 D3 に空白でない値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。

ステータスコード	意味
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。
72075	データ名 D1 に空白以外の値が設定されています。

CBLDCMCF('TDLQLE ')

名称

論理端末の出力キュー削除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TDLQLE  ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

論理端末の出力キューを削除します。

出力キューを正常に削除すると，未処理送信メッセージ廃棄通知イベント（ERREVTA）を通知します。

UAP で値を設定するデータ領域

●データ名 A

論理端末の出力キュー削除を示す要求コードを「VALUE 'TDLQLE△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象の論理端末を持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

0 を指定すると，該当する論理端末名称が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP から CBLDCMCF('TDLQLE ')を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

出力キューを削除する論理端末の論理端末名称を設定します。論理端末名称は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TDLQLE ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TDLQLE ')が受け付けられません。
71004	CBLDCMCF('TDLQLE ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TDLQLE ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスに論理端末の出力キューの削除を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	論理端末が削除されているため，CBLDCMCF('TDLQLE ')が受け付けられません。
71017	論理端末が閉塞されていないため，CBLDCMCF('TDLQLE ')が受け付けられません。
71018	セッションが終了されていないため，CBLDCMCF('TDLQLE ')が受け付けられません。
71019	代行送信中のため，CBLDCMCF('TDLQLE ')が受け付けられません。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。

ステータスコード	意味
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。

CBLDCMCF('TEMPGET ')

名称

一時記憶データの受け取り

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TEMPGET ’.
   02 データ名B  PIC X(5).
   02 FILLER     PIC X(3).
   02 データ名C  PIC X(4) VALUE SPACE.
   02 データ名D  PIC X(4) VALUE SPACE.
   02 データ名E  PIC 9(8).
   02 データ名F  PIC 9(8).
   02 データ名G  PIC 9(9) COMP.
   02 データ名H  PIC X(4) VALUE SPACE.
   02 データ名I  PIC X(4) VALUE SPACE.
   02 データ名J  PIC X(4) VALUE SPACE.
   02 データ名K  PIC X(4) VALUE SPACE.
   02 データ名L  PIC X(8) VALUE SPACE.
   02 データ名M1 PIC X(4) VALUE SPACE.
   02 データ名M2 PIC X(8) VALUE SPACE.
   02 データ名M3 PIC X(4) VALUE SPACE.
   02 データ名M4 PIC 9(9) COMP VALUE ZERO.
   02 データ名M5 PIC 9(9) COMP VALUE ZERO.
   02 データ名M6 PIC X(1) VALUE SPACE.
   02 データ名M7 PIC X(1).
   02 データ名N  PIC X(14) VALUE LOW-VALUE.
01 一意名2.
   02 データ名O  PIC X(4) VALUE SPACE.
   02 データ名P  PIC X(8) VALUE SPACE.
   02 データ名Q  PIC X(8) VALUE SPACE.
   02 データ名R  PIC X(8) VALUE SPACE.
   02 データ名S  PIC X(28) VALUE LOW-VALUE.
01 一意名3.
   02 データ名T  PIC 9(x) COMP.
   02 データ名U  PIC X(x).
   02 データ名V  PIC X(n).
```

機能

継続問い合わせ応答用一時記憶領域に格納されている一時記憶データを受け取ります。

データ名 G の長さ（9～32008 バイト（バッファ形式 1 の場合），または 7～32006 バイト（バッファ形式 2 の場合））を超える一時記憶データがある場合，超えた分については切り捨てます。

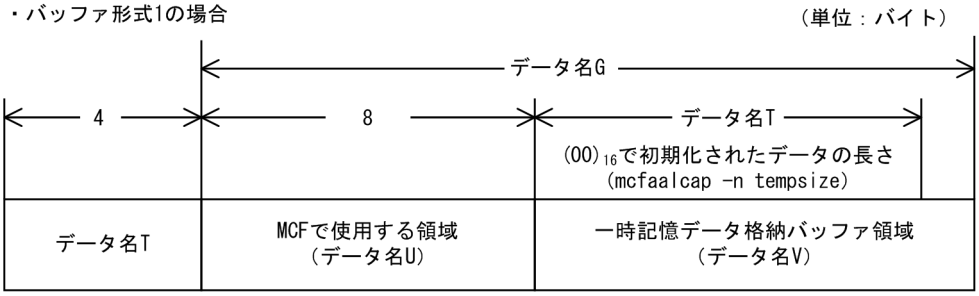
データ名Ｇから８（バッファ形式１の場合），または６（バッファ形式２の場合）を減算した値と比べて一時記憶データ長の方が短い場合，データ名Ｖに一時記憶データを設定します。データ名Ｖの残りの領域については何も設定しません。

受け取り要求実行時，初期状態（継続問い合わせ応答開始後，CBLDCMCF('TEMPPUT ')を１回も実行していない状態）の場合，アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドで指定した長さの$(00)_{16}$の一時記憶データがあるものとして CBLDCMCF('TEMPGET ')を実行します。

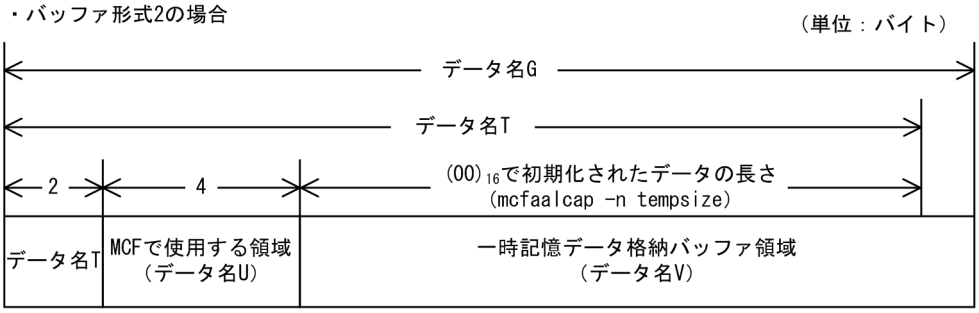
受け取り領域（一意名３で示す領域）の形式を次に示します。

●CBLDCMCF(' TEMPPUT ')未実行（継続問い合わせ応答開始後，CBLDCMCF(' TEMPPUT ')を実行していない（初期状態））

・バッファ形式1の場合

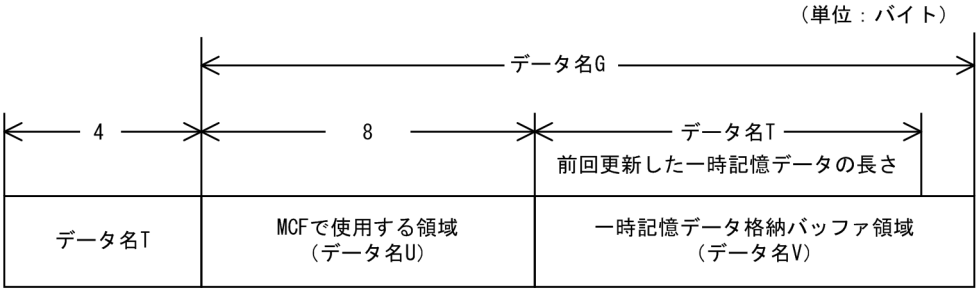


・バッファ形式2の場合

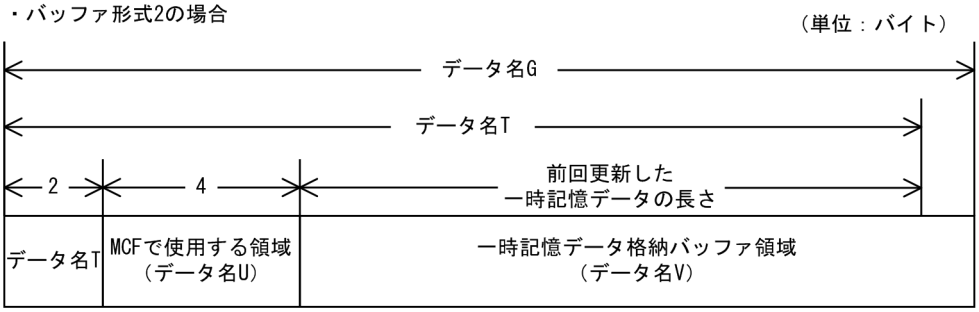


●CBLDCMCF(' TEMPPUT ')実行済み（継続問い合わせ応答開始後，CBLDCMCF(' TEMPPUT ')を1回以上実行）

・バッファ形式1の場合



・バッファ形式2の場合



UAP で値を設定するデータ領域

●データ名 A

一時記憶データの受け取りを示す要求コードを「VALUE 'TEMPGET△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E，データ名 F

MCF で使用する領域です。

●データ名 G

一時記憶データを受け取る領域の長さを 9～32008 バイト（バッファ形式 1 の場合），または 7～32006 バイト（バッファ形式 2 の場合）で設定します。

●データ名 H，データ名 I，データ名 J，データ名 K，データ名 L，データ名 M1，データ名 M2，データ名 M3

空白を設定します。

●データ名 M4，データ名 M5

0 を設定します。

●データ名 M6

空白を設定します。

●データ名 M7

使用するバッファ形式を設定します。

'1' … バッファ形式 1 を使用する場合に設定します。

'2' … バッファ形式 2 を使用する場合に設定します。

●データ名 N

「LOW-VALUE」を設定します。

●データ名 O，データ名 P，データ名 Q，データ名 R

空白を設定します。

●データ名 S

「LOW-VALUE」を設定します。

●データ名 U

【バッファ形式 1 の場合】 PIC X（8）

【バッファ形式 2 の場合】 PIC X（4）

MCF で使用する領域です。

OpenTP1 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 T

【バッファ形式 1 の場合】 PIC 9（9）

前回更新した一時記憶データの長さが返されます。初期状態の場合，継続問い合わせ応答用一時記憶領域の長さ（アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドの指定値）が返されます。

【バッファ形式 2 の場合】 PIC 9（4）

前回更新した一時記憶データの長さ＋ 6 が返されます。初期状態の場合，継続問い合わせ応答用一時記憶領域の長さ（アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドの指定値）＋ 6 が返されます。

●データ名 V

受け取った一時記憶データが返されます。初期状態の場合，継続問い合わせ応答用一時記憶領域の長さ（アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドの指定値）分だけ$(00)_{16}$が埋められます。

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。
72000	SPP では CBLDCMCF('TEMPGET ')を呼び出せません。
72013	一時記憶データ格納バッファ領域の長さ（データ名 G から 8（バッファ形式 1 の場合），または 6（バッファ形式 2 の場合）を減算した値）を超える一時記憶データを受け取りました。一時記憶データ格納バッファ領域の長さを超える一時記憶データを切り捨てました。
72016	データ名 N またはデータ名 S に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72036	データ名 G の設定値が不足しています。9 バイト以上（バッファ形式 1 の場合），または 7 バイト以上（バッファ形式 2 の場合）の領域を確保してください。
72101	継続問い合わせ応答型でないアプリケーションで，CBLDCMCF('TEMPGET ')を呼び出しました。

ステータスコード	意味
72106	先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，CBLDCMCF('TEMPGET ')を呼び出しました。
72107	CBLDCMCF('CONTEND ')を呼び出したあとで，CBLDCMCF('TEMPGET ')を呼び出しました。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('TEMPPUT ')

名称

一時記憶データの更新

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ USING 一意名1 一意名2 一意名3
```

DATA DIVISION の指定

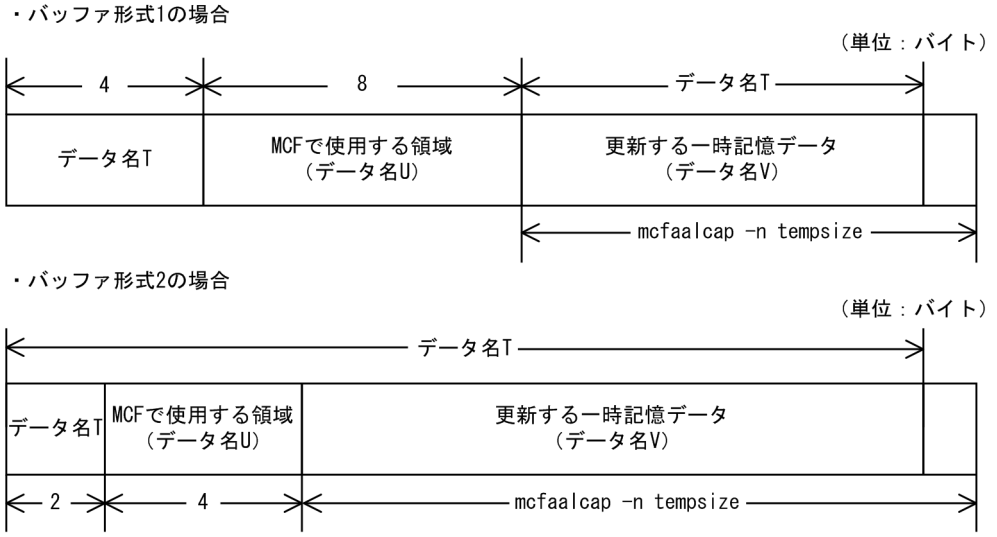
```
01 一意名1.
   02 データ名A  PIC X(8) VALUE ’TEMPPUT ’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(4) VALUE SPACE.
   02 データ名D  PIC X(4) VALUE SPACE.
   02 データ名E  PIC 9(8).
   02 データ名F  PIC 9(8).
   02 データ名G  PIC 9(9) COMP VALUE ZERO.
   02 データ名H  PIC X(4) VALUE SPACE.
   02 データ名I  PIC X(4) VALUE SPACE.
   02 データ名J  PIC X(4) VALUE SPACE.
   02 データ名K  PIC X(4) VALUE SPACE.
   02 データ名L  PIC X(8) VALUE SPACE.
   02 データ名M1 PIC X(4) VALUE SPACE.
   02 データ名M2 PIC X(8) VALUE SPACE.
   02 データ名M3 PIC X(4) VALUE SPACE.
   02 データ名M4 PIC 9(9) COMP VALUE ZERO.
   02 データ名M5 PIC 9(9) COMP VALUE ZERO.
   02 データ名M6 PIC X(1) VALUE SPACE.
   02 データ名M7 PIC X(1).
   02 データ名N  PIC X(14) VALUE LOW-VALUE.
01 一意名2.
   02 データ名O  PIC X(4) VALUE SPACE.
   02 データ名P  PIC X(8) VALUE SPACE.
   02 データ名Q  PIC X(8) VALUE SPACE.
   02 データ名R  PIC X(8) VALUE SPACE.
   02 データ名S  PIC X(28) VALUE LOW-VALUE.
01 一意名3.
   02 データ名T  PIC 9(x) COMP.
   02 データ名U  PIC X(x).
   02 データ名V  PIC X(n).
```

機能

継続問い合わせ応答用一時記憶領域に格納されている一時記憶データを更新します。

アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドには，更新する一時記憶データ長（データ名 T（バッファ形式 1 の場合），またはデータ名 T から 6 を減算した値（バッファ形式 2 の場合））以上の値を指定してください。

更新する領域（一意名 3 で示す領域）の形式を次に示します。



UAP で値を設定するデータ領域

●データ名 A

一時記憶データの更新を示す要求コードを「VALUE 'TEMPPUT△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E，データ名 F

MCF で使用する領域です。

●データ名 G

0 を設定します。

●データ名 H，データ名 I，データ名 J，データ名 K，データ名 L，データ名 M1，データ名 M2，データ名 M3

空白を設定します。

●データ名 M4，データ名 M5

0 を設定します。

●データ名 M6

空白を設定します。

●データ名 M7

使用するバッファ形式を設定します。

'1' … バッファ形式 1 を使用する場合に設定します。

'2' … バッファ形式 2 を使用する場合に設定します。

●データ名 N

「LOW-VALUE」を設定します。

●データ名 O，データ名 P，データ名 Q，データ名 R

空白を設定します。

●データ名 S

「LOW-VALUE」を設定します。

●データ名 T

【バッファ形式 1 の場合】 PIC 9（9）

【バッファ形式 2 の場合】 PIC 9（4）

一時記憶データの更新データ長を設定します。

●データ名 U

【バッファ形式 1 の場合】 PIC X（8）

【バッファ形式 2 の場合】 PIC X（4）

MCF で使用する領域です。

●データ名 V

一時記憶データが格納されている領域を設定します。

OpenTP1 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。
71103	一時記憶データを更新するための領域をメモリ上に確保できませんでした。

ステータスコード	意味
72000	SPP では CBLDCMCF('TEMPPUT ')を呼び出せません。
72016	データ名 N またはデータ名 S に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72035	データ名 T に設定した更新データの長さが，アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドで定義した長さを超えています。
	バッファ形式 1 の場合はデータ名 T に 0 バイト，またはマイナス値を設定しています。バッファ形式 2 の場合はデータ名 T に 0 から 6 バイト，またはマイナス値を設定しています。
72101	継続問い合わせ応答型でないアプリケーションで，CBLDCMCF('TEMPPUT ')を呼び出しました。
72105	CBLDCMCF('TEMPGET ')を呼び出す前に，CBLDCMCF('TEMPPUT ')を呼び出しました。
72106	先頭セグメントを受信する CBLDCMCF('RECEIVE ')を呼び出す前に，CBLDCMCF('TEMPPUT ')を呼び出しました。
72107	CBLDCMCF('CONTEND ')を呼び出したあとで，CBLDCMCF('TEMPPUT ')を呼び出しました。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('TIMERCAN')

名称

ユーザタイマ監視の取り消し

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TIMERCAN’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(16) VALUE LOW-VALUE.
01 一意名2.
   02 データ名D  PIC 9(9) COMP VALUE ZERO.
   02 データ名E  PIC 9(9) COMP.
   02 データ名F  PIC X(16) VALUE LOW-VALUE.
   02 データ名G  PIC X(8).
   02 データ名H  PIC X(8) VALUE LOW-VALUE.
   02 データ名I  PIC X(16) VALUE LOW-VALUE.
```

機能

CBLDCMCF('TIMERSET')で設定したユーザタイマ監視を取り消します。

設定したユーザタイマ監視は，CBLDCMCF('TIMERCAN')が正常にリターンした時点で取り消されています。

また，CBLDCMCF('TIMERCAN')を呼び出した時点で，設定したユーザタイマ監視がタイムアウトしていて，MHP がすでに起動されている場合には，CBLDCMCF('TIMERCAN')が 70910 でエラーリターンします。

CBLDCMCF('TIMERCAN')は，ユーザサーバからだけ呼び出せます。

UAP で値を設定するデータ領域

●データ名 A

ユーザタイマ監視の取り消しを示す要求コードを「VALUE 'TIMERCAN'」と設定します。

●データ名 C

「LOW-VALUE」を設定します。

●データ名 D

0 を設定します。

●データ名 E

CBLDCMCF('TIMERSET')でユーザタイマ監視を設定したときと同じタイマ要求識別子を指定します。

●データ名 F

「LOW-VALUE」を設定します。

●データ名 G

CBLDCMCF('TIMERSET')でユーザタイマ監視を設定したときと同じ論理端末名称を指定します。8 文字に満たない名称を指定する場合は後ろを空白で埋めます。

●データ名 H

「LOW-VALUE」を設定します。

●データ名 I

「LOW-VALUE」を設定します。

OpenTP1 から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。
72016	データ名 C に設定した値が間違っています。
	データ名 I に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
70910	データ名 E に設定した値が間違っています。
	データ名 E に指定したタイマ要求識別子は登録されていません。
	すでにタイムアウトが発生してアプリケーションが起動されているか，またはすでにユーザタイマ監視が取り消されています。
70911	データ名 F に設定した値が間違っています。
70912	データ名 G に設定した値が間違っています。
70916	要求された機能は MCF で定義されていません。
上記以外	プログラム破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('TIMERSET')

名称

ユーザタイマ監視の設定

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TIMERSET’.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(16) VALUE LOW-VALUE.
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(16) VALUE LOW-VALUE.
   02 データ名G  PIC X(8).
   02 データ名H  PIC X(8).
   02 データ名I  PIC X(16) VALUE LOW-VALUE.
01 一意名3.
   02 データ名J  PIC 9(9) COMP.
   02 データ名K  PIC X(n).
```

機能

ユーザで任意の時間監視をするには，UAP から CBLDCMCF('TIMERSET')でユーザタイマ監視を設定します。CBLDCMCF('TIMERSET')を呼び出すには，MCF 通信構成定義 mcfttim の-p オプションに usertime=yes を指定する必要があります。

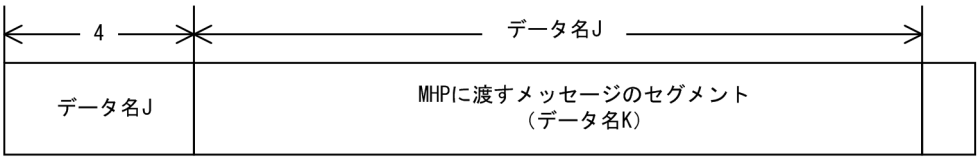
CBLDCMCF('TIMERSET')は，ユーザサーバからだけ呼び出せます。

データ名 D に指定した時間（単位：秒）を経過した（タイムアウトが発生した）場合，データ名 G で指定した論理端末からイベントを生成し，データ名 H に設定したアプリケーション名の MHP を起動させます。MHP から CBLDCMCF('TIMERSET')を呼び出す場合，データ名 G を省略できます。省略した場合，入力元論理端末を仮定します。

タイムアウト発生時に起動させる MHP は，非応答型（noans 型）の MHP でなくてはなりません。この MHP にメッセージを渡す場合，一意名 3 で示す領域は次のようになります。

・一意名3で示す領域

（単位：バイト）



この MHP にメッセージが渡されたときの形式は「CBLDCMCF('RECEIVE ')」を参照してください。

CBLDCMCF('TIMERSET')で設定したユーザタイマ監視は，同一のデータ名 E およびデータ名 G を指定した CBLDCMCF('TIMERCAN')関数を呼び出すことで取り消せます。

時間監視は，CBLDCMCF('TIMERSET')を呼び出した直後から行います。

同時に時間監視できる数は，MCF 通信構成定義 mcfttim の-p オプションの timereqno オペランドに指定した最大タイマ監視要求数までです。

UAP で値を設定するデータ領域

●データ名 A

ユーザタイマ監視の設定を示す要求コードを「VALUE 'TIMERSET'」と設定します。

●データ名 C

「LOW-VALUE」を設定します。

●データ名 D

CBLDCMCF('TIMERSET')を呼び出してから，何秒後に MHP を起動させるかを設定します。設定できる秒数は，1 から 360000 まで（1 秒から 100 時間まで）です。

一定の時間間隔でタイムアウトが発生したかどうかを監視しているので，データ名 D に設定した時間と実際のタイムアウト検出までの時間には誤差が生じます。時間監視の精度は，MCF 通信構成定義 mcfttim の-t オプションの btim オペランドに指定した時間監視間隔の値に依存します。

●データ名 E

タイマ要求識別子を設定します。

データ名 E はこのタイマを一意に識別するための情報です。データ名 E は，データ名 G で指定した論理端末内で必ずユニークになるようにしてください。

●データ名 F

「LOW-VALUE」を設定します。

●データ名 G

タイムアウトが発生したときにイベントを生成する論理端末名称を指定します。8 文字に満たない名称を指定する場合は後ろを空白で埋めます。論理端末名称は，アプリケーション起動サービスの論理端末を指定してもかまいません。

MHP からこの関数を呼び出す場合は，論理端末名称を省略できます。省略する場合には，空白を指定します。省略した場合，入力元論理端末を仮定します。

●データ名 H

起動させる MHP のアプリケーション名を指定します。このアプリケーションの属性は，データ名 G で指定した論理端末を持つ MCF 通信サーバの MCF 通信構成定義 mcftenv -a オプションで指定した MCF アプリケーション定義内のアプリケーション属性定義（mcfaalcap）に定義してください。8 文字に満たない名称を指定する場合は後ろを空白で埋めます。MHP は非応答型（noans 型）の MHP でなければなりません。指定するアプリケーション名はユーザイベントでなければなりません。

●データ名 I

「LOW-VALUE」を設定します。

●データ名 J

起動させる MHP に渡すメッセージセグメントの長さを設定します。起動させる MHP に渡すセグメントがない場合は，0 を設定します。指定できる値の範囲は 0 から 256 までです。設定できる値の最大値は MCF 通信構成定義 mcfttim の-p オプションの msgsize オペランドに指定した最大メッセージ長に依存します。

●データ名 K

起動させる MHP に渡す，メッセージのセグメントの内容を設定します。複数のセグメントを設定できません。起動させる MHP に渡すセグメントがない場合は，ヌル文字を設定します。

OpenTP1 から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。
72016	データ名 C に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
70900	データ名 I に設定した値が間違っています。
70909	データ名 D に設定した値が間違っています。

ステータスコード	意味
70910	データ名 E に設定した値が間違っています。
	指定したタイマ要求識別子はすでに登録されています。
70911	データ名 F に設定した値が間違っています。
70912	データ名 G に設定した値が間違っています。
70913	データ名 H に設定した値が間違っています。
70914	データ名 J に設定した値が間違っています。
70915	データ名 K に設定した値が間違っています。
70916	要求された機能は MCF で定義されていません。
70917	タイマ登録領域に空きがないためユーザタイマ監視が設定できません。タイマ登録領域を確保するため，MCF 通信構成定義 mcfttim の-p オプションの timereqno オペランドの値を見直してください。必要に応じて，MCF マネジャ定義 mcfmcomn の-p オプション，およびシステム環境定義 static_shmpool_size オペランドの値を確認してください。
上記以外	プログラム破壊などによる，予期しないエラーが発生しました。

CBLDCMCF('TLSCN ')

名称

コネクションの状態取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TLSCN   ’.
   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 X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.

01 一意名3.
   02 データ名M   PIC 9(9)  COMP.
   02 一意名4.
      03 データ名N   PIC X(8).
      03 データ名O   PIC X(4).
      03 データ名P   PIC X(4).
      03 データ名Q   PIC X(40) VALUE LOW-VALUE.
```

機能

コネクションの状態を取得します。

UAP で値を設定するデータ領域

●データ名 A

コネクション状態取得を示す要求コードを「VALUE 'TLSCN△△△'」と設定します。

●データ名 C

状態を取得するコネクションの指定方法を設定します。

'LE△△'

状態を取得するコネクションを論理端末名称で指定するときに設定します。TP1/NET/NCSB，および TP1/NET/X25-Extended では指定できません。

'CN△△'

状態を取得するコネクションをコネクション ID で指定するときに設定します。

空白

省略されたものとして，'LE△△'（論理端末名称指定）が仮定されます。

●データ名 D

空白を設定します。

●データ名 E

処理対象のコネクションを持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

論理端末名称を使用してコネクションの状態取得を要求する場合は，無効となります。

0 を指定すると，該当するコネクション ID が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

状態を取得するコネクションの論理端末名称，またはコネクション ID を設定します。論理端末名称，またはコネクション ID は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

●データ名 M

一意名 4 から一意名 n の数（データ名 N・データ名 O・データ名 P とデータ名 Q の組の数）として，1 を設定します。

処理終了後は，該当するコネクションの個数が返されます。

●データ名 Q

MCF で使用する領域です。

OpenTP1 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 M

この関数の対象となったコネクションの個数が返されます。

●データ名 N

要求したコネクションのコネクション ID が設定されます。

●データ名 O

要求したコネクションのプロトコル種別が設定されます。

'UA△△'
　TP1/NET/User Agent（OSAS/UA プロトコル）

'hds△'
　TP1/NET/HDLC（HDLC プロトコル）

'X25△'
　TP1/NET/X25（X.25 プロトコル）

'TP△△'
　TP1/NET/OSI-TP（OSI TP プロトコル）

'XP△△'
　TP1/NET/XMAP3

'HS1△'
　TP1/NET/HSC（HSC1 手順）

'HS2△'
　TP1/NET/HSC（HSC2 手順）

'CSB△'
　TP1/NET/NCSB（NCSB プロトコル）

'NIF△'
　TP1/NET/OSAS-NIF（NIF/OSI プロトコル）

'SL2△'
　TP1/NET/Secondary Logical Unit - TypeP2（SLUTYPE-P プロトコル（2 次局））

'TCP△'

TP1/NET/TCP/IP（TCP/IP プロトコル）

'X2E△'

TP1/NET/X25-Extended（X.25 プロトコル）

●データ名 P

要求したコネクション状態として，次の値が設定されます。

'ACT△'

コネクションが確立されていることを示します。

'ACTB'

コネクションが確立処理中であることを示します。

'DCT△'

コネクションが解放されていることを示します。

'DCTB'

コネクションが解放処理中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。
71001	MCF が開始処理中のため，CBLDCMCF('TLSCN ')が受け付けられません。
71004	CBLDCMCF('TLSCN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71007	指定されたコネクション名は登録されていません。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TLSCN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにコネクションの状態取得を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	コネクションが削除されているため，CBLDCMCF('TLSCN ')が受け付けられません。
71014	TP1/NET/NCSB，もしくは TP1/NET/X25-Extended の論理端末名称を指定しています。または，TP1/NET/OSI-TP，もしくは TP1/NET/TCP/IP のコネクショングループ名を指定しています。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。

ステータスコード	意味
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に'LE△△'，'CN△△'，または空白以外が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。
72076	データ名 M に 1 でない値が設定されています。

CBLDCMCF('TLSCOM ')

名称

MCF 通信サービスの状態取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TLSCOM  ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP VALUE ZERO.
   02 データ名F   PIC X(64) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.

01 一意名3.
   02 データ名M   PIC 9(9)  COMP.
   02 一意名4.
      03 データ名N   PIC 9(9) COMP.
      03 データ名O   PIC X(8).
      03 データ名P   PIC X(20).
      03 データ名Q   PIC X(12).
      03 データ名R   PIC X(20) VALUE LOW-VALUE.
    :
   02 一意名n.
      03 データ名N   PIC 9(9) COMP.
      03 データ名O   PIC X(8).
      03 データ名P   PIC X(20).
      03 データ名Q   PIC X(12).
      03 データ名R   PIC X(20) VALUE LOW-VALUE.
```

機能

MCF 通信サービスまたはアプリケーション起動サービスの状態を取得します。

UAP で値を設定するデータ領域

●データ名 A

MCF 通信サービスまたはアプリケーション起動サービスの状態取得を示す要求コードを「VALUE 'TLSCOM△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

0 を設定します。

●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

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

●データ名 M

一意名 4 から一意名 n の数（データ名 N・データ名 O・データ名 P・データ名 Q とデータ名 R の組の数）を設定します。

処理終了後は，MCF サービスに登録されている MCF 通信サービスまたはアプリケーション起動サービスの個数が返されます。

●データ名 R

MCF で使用する領域です。

OpenTP1 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 M

MCF サービスに登録されている MCF 通信サービスまたはアプリケーション起動サービスの個数が返されます。

●データ名 N

MCF 通信プロセス識別子またはアプリケーション起動プロセス識別子が設定されます。

●データ名 O

MCF 通信サービス名が設定されます。

●データ名 P

プロトコル種別が設定されます。

'MCF△△△△△△△△△△△△△△△△△△'
　TP1/Message Control のアプリケーション起動サービス

'User△Agent△△△△△△△△△△'
　TP1/NET/User Agent（OSAS/UA プロトコル）

'HDLC△△△△△△△△△△△△△△△△△'
　TP1/NET/HDLC（HDLC プロトコル）

'X25△△△△△△△△△△△△△△△△△△'
　TP1/NET/X25（X.25 プロトコル）

'TP△△△△△△△△△△△△△△△△△△'
　TP1/NET/OSI-TP（OSI TP プロトコル）

'XMAP3△△△△△△△△△△△△△△△'
　TP1/NET/XMAP3

'HSC△△△△△△△△△△△△△△△△△'
　TP1/NET/HSC（HSC プロトコル）

'NCSB△△△△△△△△△△△△△△△△'
　TP1/NET/NCSB（NCSB プロトコル）

'OSAS-NIF△△△△△△△△△△△△△'
　TP1/NET/OSAS-NIF（NIF/OSI プロトコル）

'NET/SLUP2△△△△△△△△△△△△'
　TP1/NET/Secondary Logical Unit - TypeP2（SLUTYPE-P プロトコル（2 次局））

'TCP/IP△△△△△△△△△△△△△△△'
　TP1/NET/TCP/IP（TCP/IP プロトコル）

'X25-EX△△△△△△△△△△△△△△△'
　TP1/NET/X25-Extended（X.25 プロトコル）

'UDP/IP△△△△△△△△△△△△△△△'
　TP1/NET/User Datagram Protocol（UDP プロトコル）

●データ名 Q

MCF 通信サービスまたはアプリケーション起動サービスの状態として，次の値が設定されます。

'OFFLINE△△△△△'
　サービスが停止されていることを示します。

'STARTING△△△△'

サービスが準備中であることを示します。

'ONLINE△△△△△△'

サービスが開始されている，または終了準備中であることを示します。

'PREENDING△△△'

サービスが部分停止の終了準備中であることを示します。

'ENDING△△△△△△'

サービスが終了中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。
71001	MCF が開始処理中のため，CBLDCMCF('TLSCOM ')が受け付けられません。
71004	CBLDCMCF('TLSCOM ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
72013	MCF 通信サービスまたはアプリケーション起動サービスの数がデータ名 M の指定値を超えました。データ名 M の指定値を超えた MCF 通信サービスまたはアプリケーション起動サービスの情報は切り捨てられました。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 でない値が設定されています。
72065	データ名 F に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72076	データ名 M に 0 以下の値が設定されています。

CBLDCMCF('TLSLE ')

名称

論理端末の状態取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TLSLE   ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F1  PIC X(8).
   02 データ名F2  PIC X(56) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.

01 一意名3.
   02 データ名M   PIC 9(9)  COMP.
   02 一意名4.
      03 データ名N   PIC X(8).
      03 データ名O   PIC X(4) VALUE LOW-VALUE.
      03 データ名P   PIC X(4).
      03 データ名Q   PIC X(40) VALUE LOW-VALUE.
```

機能

論理端末の状態を取得します。

UAP で値を設定するデータ領域

●データ名 A

論理端末状態取得を示す要求コードを「VALUE 'TLSLE△△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象の論理端末を持つ MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 0～239 です。

0 を指定すると，該当する論理端末名称が属する MCF 通信サービスを検索します。MCF 通信サービスが多い構成や UAP からこの関数を多数発行する場合は，MCF 通信プロセス識別子の指定をお勧めします。

●データ名 F1

状態を取得する論理端末の論理端末名称を設定します。論理端末名称は 8 バイト以内で設定してください。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 F2，データ名 G，データ名 H，データ名 I，データ名 J

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

●データ名 M

一意名 4 から一意名 n の数（データ名 N・データ名 O・データ名 P とデータ名 Q の組の数）として，1 を設定します。

処理終了後は，該当する論理端末の個数が返されます。

●データ名 O，データ名 Q

MCF で使用する領域です。

OpenTP1 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 M

この関数の対象となった論理端末の個数が返されます。

●データ名 N

要求した論理端末の名称が設定されます。

●データ名 P

要求した論理端末の状態として，次の値が設定されます。

'ACT△'

論理端末が閉塞解除されていることを示します。

'DCT△'

論理端末が閉塞されていることを示します。

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。
71001	MCF が開始処理中のため，CBLDCMCF('TLSLE ')が受け付けられません。
71004	CBLDCMCF('TLSLE ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71008	指定された論理端末名称は登録されていません。
71009	CBLDCMCF('TLSLE ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスに論理端末の状態取得を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
71011	論理端末が削除されているため，CBLDCMCF('TLSLE ')が受け付けられません。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 未満または 240 以上の値が設定されています。
72063	データ名 F1 の先頭が空白です。
72065	データ名 F2 に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72074	データ名 F1 に設定された文字列中に不正な文字があります。
72076	データ名 M に 1 でない値が設定されています。

CBLDCMCF('TLSLN ')

名称

サーバ型コネクションの確立要求の受付状態取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TLSLN   ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F   PIC X(64) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.

01 一意名3.
   02 データ名M   PIC 9(9)  COMP VALUE 1.
   02 一意名4.
      03 データ名N   PIC X(4).
      03 データ名O   PIC X(60) VALUE LOW-VALUE.
```

機能

サーバ型コネクションの確立要求の受付状態を取得します。

UAP で値を設定するデータ領域

●データ名 A

コネクションの確立要求の受付状態取得を示す要求コードを「VALUE 'TLSLN△△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象の MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 1～239 です。

●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

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

●データ名 M

1 を設定します。

●データ名 O

MCF で使用する領域です。

OpenTP1 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 N

サーバ型コネクションの確立要求の受付状態として，次の値が設定されます。

'LSTN'
　サーバ型コネクションの確立要求の受付が開始されていることを示します。

'RTRY'
　サーバ型コネクションの確立要求の受付が開始処理中であることを示します。

'ON_W'
　サーバ型コネクションの確立要求の受付が開始要求待ち状態であることを示します。

'INIT'
　サーバ型コネクションの確立要求の受付終了したことを示します。

それぞれの状態のときに使用できる関数の関係を次の表に示します。

データ名 N の設定値	使用できる COBOL-UAP 作成用プログラム	
	CBLDCMCF('TONLN ')	CBLDCMCF('TOFLN ')
LSTN	×	○
RTRY	×	○
ON_W	○	○

データ名 N の設定値	使用できる COBOL-UAP 作成用プログラム	
	CBLDCMCF('TONLN ')	CBLDCMCF('TOFLN ')
INIT	○	×

（凡例）

○：使用できます。

×：使用できません。

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。
71001	MCF が開始処理中のため，CBLDCMCF('TLSLN ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TLSLN ')が受け付けられません。
71004	CBLDCMCF('TLSLN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71009	CBLDCMCF('TLSLN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにサーバ型コネクションの確立要求の受付状態取得を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 以下または 240 以上の値が設定されています。
72065	データ名 F に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。
72076	データ名 M に 1 でない値が設定されています。

CBLDCMCF('TOFLN ')

名称

サーバ型コネクションの確立要求の受付終了

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TOFLN   ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F   PIC X(64) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

サーバ型コネクションの確立要求の受付を終了します。

UAP で値を設定するデータ領域

●データ名 A

コネクションの確立要求の受付終了を示す要求コードを「VALUE 'TOFLN△△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象 MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 1～239 です。

●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

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TOFLN ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TOFLN ')が受け付けられません。
71004	CBLDCMCF('TOFLN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71009	CBLDCMCF('TOFLN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにサーバ型コネクションの確立要求の受付終了を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 以下または 240 以上の値が設定されています。
72065	データ名 F に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。

CBLDCMCF('TONLN ')

名称

サーバ型コネクションの確立要求の受付開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCMCF’ 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 ’TONLN   ’.
   02 データ名B   PIC X(5).
   02 FILLER      PIC X(3).
   02 データ名C   PIC X(4) VALUE SPACE.
   02 データ名D   PIC X(28) VALUE SPACE.
   02 データ名E   PIC 9(9)  COMP.
   02 データ名F   PIC X(64) VALUE SPACE.
   02 データ名G   PIC X(8)  VALUE SPACE.
   02 データ名H   PIC X(8)  VALUE SPACE.
   02 データ名I   PIC X(144) VALUE SPACE.
   02 データ名J   PIC X(184) VALUE SPACE.
   02 データ名K   PIC 9(9)  COMP VALUE ZERO.

01 一意名2.
   02 データ名L   PIC 9(9)  COMP VALUE ZERO.
```

機能

サーバ型コネクションの確立要求の受付を開始します。

UAP で値を設定するデータ領域

●データ名 A

コネクションの確立要求の受付開始を示す要求コードを「VALUE 'TONLN△△△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E

処理対象の MCF 通信サービスの MCF 通信プロセス識別子を設定します。設定できる範囲は 1～239 です。

●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

空白を設定します。

●データ名 K，データ名 L

0 を設定します。

OpenTP1 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。
71001	MCF が開始処理中のため，CBLDCMCF('TONLN ')が受け付けられません。
71002	MCF が終了処理中のため，CBLDCMCF('TONLN ')が受け付けられません。
71004	CBLDCMCF('TONLN ')の処理中にメモリ不足が発生しました。
71005	通信障害が発生しました。原因については，メッセージログファイルを参照してください。
71006	内部障害が発生しました。原因については，メッセージログファイルを参照してください。
71009	CBLDCMCF('TONLN ')が，該当する通信プロセスではサポートされていません。
71010	MCF 通信プロセスにサーバ型コネクションの確立要求の受付開始を要求しましたが，受け付けられませんでした。原因については，メッセージログファイルを参照してください。
72028	データ名 A に設定した値が間違っています。
72052	データ名 K に 0 でない値が設定されています。
72053	データ名 L に 0 でない値が設定されています。
72058	データ名 C に空白でない値が設定されています。
72059	データ名 D に空白でない値が設定されています。
72061	データ名 E に 0 以下または 240 以上の値が設定されています。
72065	データ名 F に空白でない値が設定されています。
72066	データ名 G に空白でない値が設定されています。
72068	データ名 H に空白でない値が設定されています。
72070	データ名 I に空白でない値が設定されています。
72072	データ名 J に空白でない値が設定されています。

性能検証用トレース（CBLDCPRF）

性能検証用トレースで使えるプログラムについて説明します。性能検証用トレースの COBOL-UAP 作成用プログラムを次に示します。

- CBLDCPRF('PRFGETN ')－性能検証用トレース取得通番の通知
- CBLDCPRF('PRFPUT ')－ユーザ固有の性能検証用トレースの取得

性能検証用トレースの COBOL-UAP 作成用プログラム（CBLDCPRF）は，TP1/Server Base の UAP，および TP1/LiNK の UAP で使えます。ただし，この機能は，TP1/Extension 1 をインストールしていることが前提です。TP1/Extension 1 をインストールしていない場合の動作は保証できませんので，ご了承ください。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。性能検証用トレース（CBLDCPRF）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCPRF.cbl に格納してあります。

CBLDCPRF('PRFGETN ')

名称

性能検証用トレース取得通番の通知

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCPRF’ 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 ’PRFGETN ’.
   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     PIC 9(4) COMP.
```

機能

CBLDCPRF ('PRFGETN ')を呼び出す前に取得した，最新の性能検証用トレース（prf トレース）のプロセス内取得通番を，CBLDCPRF ('PRFGETN ')呼び出し元に通知します。

CBLDCPRF ('PRFGETN ')を呼び出したプロセスで一度も性能検証用トレースを取得していない場合，プロセス内取得番号は 0 となります。

UAP で値を設定するデータ領域

●データ名 A

性能検証用トレース取得通番の通知を示す要求コードを「VALUE 'PRFGETN△'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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。

●一意名 2

性能検証用トレース取得通番が返されます。

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。
04601	データ名に設定した値に誤りがあります。 要求コード（データ名 A）が間違っている場合も含みます。

CBLDCPRF('PRFPUT ')

名称

ユーザ固有の性能検証用トレースの取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCPRF’ 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 ’PRFPUT  ’.
   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(4) COMP.
   02 データ名D  PIC 9(4) COMP.
   02 データ名E  PIC X(n).
```

機能

ユーザ固有の性能検証用トレース（prf トレース）を取得します。

UAP で値を設定するデータ領域

●データ名 A

ユーザ固有の性能検証用トレースの取得を示す要求コードを「VALUE 'PRFPUT△△'」と設定します。

●データ名 Z

0 を設定します。

●データ名 C

取得するイベントのイベント ID を設定します。使用できるイベント ID の範囲は 0x0001～0x0040 です。

●データ名 D

取得するトレースデータのデータ長を設定します。設定できるデータ長は 4 バイト以上 256 バイト以下です。また，このデータ長は 4 バイトの倍数でなければなりません。

●データ名 E

取得するトレースデータを設定します。トレースデータとして有効なデータは，データ名 D で設定した長さです。

OpenTP1 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。
04601	データ名に設定した値に誤りがあります。 要求コード（データ名 A）が間違っている場合も含みます。

注意事項

CBLDCPRF('PRFPUT ')がステータスコード「00000」を返してもトレースが正しく取得されているとは限りません。これは，トレースの取得処理で，排他を使用しないため複数のプロセスから同時に取得要求が出された場合，データが消失してしまうことがあるためです。

リモート API 機能(CBLDCRAP)

リモート API 機能で，コネクションの確立・解放をユーザが管理する場合に使用するプログラムについて説明します。リモート API 機能の COBOL-UAP 作成用プログラムを次に示します。

- CBLDCRAP('CONNECT ') − rap リスナーとのコネクションの確立
- CBLDCRAP('CONNECTX') − rap リスナーとのコネクションの確立
- CBLDCRAP('DISCNCT ') − rap リスナーとのコネクションの解放

リモート API 機能の COBOL-UAP 作成用プログラム（CBLDCRAP）は，TP1/Server Base の UAP，および TP1/LiNK の UAP で使えます。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。リモート API 機能（CBLDCRAP）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCRAP.cbl に格納してあります。

CBLDCRAP('CONNECT ')

名称

rap リスナーとのコネクションの確立

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRAP’ 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 ’CONNECT ’.
   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.
01 一意名2.
   02 データ名E  PIC X(64).
   02 データ名F  PIC S9(9) COMP.
```

機能

rap リスナーと rap クライアントとの間にコネクションを確立します。

UAP で値を設定するデータ領域

●データ名 A

rap リスナーとのコネクションの確立要求を示す要求コードを「VALUE 'CONNECT△'」と設定します。

●データ名 C

0 を設定します。

●データ名 E

rap リスナーが起動されている OpenTP1 ノードのホスト名を設定します。

指定するホスト名は 1～64 文字です。65 文字以上のホスト名を指定する場合は,
CBLDCRAP('CONNECTX')を使用してください。

●データ名 F

rap リスナーの使用しているウェルノウンポートのポート番号を設定します。

OpenTP1 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

サービス ID を受け取る領域です。

rap リスナーとのコネクションが正常に確立された場合にサービス ID が設定されます。
CBLDCRAP('DISCNCT ')でコネクションを解放するときにも同じサービス ID を使用します。

ステータスコード

ステータスコード	意味
00000	正常終了しました。rap リスナーとのコネクションが確立されました。
05501	データ領域に設定した値に誤りがあります。または，データ領域が設定されていません。次の値を見直してください。 • データ名 A，データ名 C の値('CONNECT ',0) • データ名 E の文字数（1～64） • データ名 F の範囲（1～65535） • データ名 D の領域
05502	プロトコル不正です。要因としては次のことが考えられます。 • CBLDCRPC('OPEN ')が呼び出されていません。 • ユーザサービス定義の rpc_rap_auto_connect オペランドに Y が指定されていますが，CBLDCRAP('CONNECT ')が呼び出されました。 • ユーザサービスネットワーク定義の dcsvgdef 定義コマンドで，-w オプションが指定されていません。
05503	メモリ不足が発生しました。
05517	一つのプロセスから CBLDCRAP('CONNECT ')が呼び出せる上限値を超えました。
05505	rap リスナーとの通信でネットワーク障害が発生しました。
05506	rap リスナーとの通信でタイムアウトが発生しました。
05507	ソケット不足が発生しました。
05508	ホスト名称が解決できません。
05521	rap リスナーは停止中です。
05522	続行できない障害が発生しました。障害の要因として次のことが考えられます。 • 予期しないメッセージを受信しました。 • 予期しない相手からのメッセージを受信しました。
05523	システムコールで予期しないエラーが発生しました。
05531	接続されていないネットワーク上の rap リスナーに対してコネクションを確立しようとしています。

ステータスコード	意味
05520	rap リスナーまたは rap サーバでメモリ不足が発生しました。
05532	rap リスナーサービス定義の rap_watch_time オペランドに指定したメッセージ送受信監視時間内にコネクションが確立できませんでした。
05533	rap リスナーでシステム障害が発生しました。
05528	rap リスナーは開始処理中，または停止処理中です。
05529	すでに rap リスナーとコネクションは確立しています。
05534	rap リスナーの管理する rap クライアントのコネクション要求受付可能最大数を超えました。

注意事項

「05529」以外のステータスコードで，CBLDCRAP('CONNECT ')がエラーリターンした場合，rap リスナーとのコネクションは確立されていません。

UAP トレースに取得されるエラー要因コードは次のとおりです。

0：エラーなし。

1：一意名 1 が設定されていません。

2：要求コード（データ名 A）が間違っています。

3：一意名 2 が設定されていません。

6：ユーザサービス定義の rpc_rap_auto_connect オペランドの指定値が Y の場合に，CBLDCRAP ('CONNECT'）が呼び出されました。または，ユーザサービスネットワーク定義が定義されていません。

CBLDCRAP('CONNECTX')

名称

rap リスナーとのコネクションの確立

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRAP’ 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 ’CONNECTX’.
   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 1.
   02 データ名D  PIC S9(9) COMP.
01 一意名2.
   02 データ名E  PIC S9(9) COMP.
   02 データ名F  PIC X(n).
   02 FILLER    PIC X(1) VALUE LOW-VALUE.
```

機能

rap リスナーと rap クライアントとの間にコネクションを確立します。ホスト名の領域は可変長です。65 文字以上のホスト名にも対応しています。

UAP で値を設定するデータ領域

●データ名 A

rap リスナーとのコネクションの確立要求を示す要求コードを「VALUE 'CONNECTX'」と設定します。

●データ名 C

1 を設定します。

●データ名 E

rap リスナーが使用しているウェルノウンポートのポート番号を設定します。

●データ名 F

rap リスナーが起動されている OpenTP1 ノードのホスト名を設定します。指定するホスト名は 1～255 文字です。

OpenTP1 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

サービス ID を受け取る領域です。

rap リスナーとのコネクションが正常に確立された場合にサービス ID が設定されます。
CBLDCRAP('DISCNCT ')でコネクションを解放するときにも同じサービス ID を使用します。

ステータスコード

ステータスコード	意味
00000	正常終了しました。rap リスナーとのコネクションが確立されました。
05501	データ領域に設定した値に誤りがあります。または，データ領域が設定されていません。次の値を見直してください。 • データ名 A，データ名 C の値('CONNECTX',1) • データ名 E の範囲（1～65535） • データ名 F の文字数（1～255） • データ名 D の領域
05502	プロトコル不正です。要因としては次のことが考えられます。 • CBLDCRPC('OPEN ')が呼び出されていません。 • ユーザサービス定義の rpc_rap_auto_connect オペランドに Y が指定されていますが，CBLDCRAP('CONNECTX')が呼び出されました。 • ユーザサービスネットワーク定義の dcsvgdef 定義コマンドで，-w オプションが指定されていません。
05503	メモリ不足が発生しました。
05505	rap リスナーとの通信でネットワーク障害が発生しました。
05506	rap リスナーとの通信でタイムアウトが発生しました。
05507	ソケット不足が発生しました。
05508	ホスト名称が解決できません。データ名 F または DNS サーバを見直してください。
05517	一つのプロセスから CBLDCRAP('CONNECTX')が呼び出せる上限値を超えました。
05520	rap リスナーまたは rap サーバでメモリ不足が発生しました。
05521	rap リスナーは停止中です。または，データ名 E を見直してください。
05522	続行できない障害が発生しました。障害の要因として次のことが考えられます。 • 予期しないメッセージを受信しました。 • 予期しない相手からのメッセージを受信しました。
05523	システムコールで予期しないエラーが発生しました。

ステータスコード	意味
05528	rap リスナーは開始処理中，または停止処理中です。
05529	すでに rap リスナーとコネクションは確立しています。
05531	接続されていないネットワーク上の rap リスナーに対してコネクションを確立しようとしています。
05532	rap リスナーサービス定義の rap_watch_time オペランドに指定したメッセージ送受信監視時間内にコネクションが確立できませんでした。
05533	rap リスナーでシステム障害が発生しました。
05534	rap リスナーの管理する rap クライアントのコネクション要求受付可能最大数を超えました。

注意事項

「05529」以外のステータスコードで，CBLDCRAP('CONNECTX ')がエラーリターンした場合，rap リスナーとのコネクションは確立されていません。

UAP トレースに取得されるエラー要因コードは次のとおりです。

0：エラーなし。

1：一意名 1 が設定されていません。

2：要求コード（データ名 A）が間違っています。

3：一意名 2 が設定されていません。

6：ユーザサービス定義の rpc_rap_auto_connect オペランドの指定値が Y の場合に，CBLDCRAP ('CONNECT'）が呼び出されました。または，ユーザサービスネットワーク定義が定義されていません。

CBLDCRAP('DISCNCT ')

名称

rap リスナーとのコネクションの解放

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRAP’ 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 ’DISCNCT ’.
   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

rap リスナーと rap クライアントとの間に確立されているコネクションを解放します。

UAP で値を設定するデータ領域

●データ名 A

rap リスナーとのコネクションの解放要求を示す要求コードを「VALUE 'DISCNCT△'」と設定します。

●データ名 C

0 を設定します。

●データ名 D

CBLDCRAP('CONNECT ')で受け取ったサービス ID を設定します。

OpenTP1 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	正常終了しました。rap リスナーとのコネクションが解放されました。

ステータスコード	意味
05501	データ領域に設定した値に誤りがあります。または，データ領域が設定されていません。次の値を見直してください。 • データ名 A，データ名 C の値('DISCNCT ',0) • データ名 D の領域
05502	プロトコル不正です。要因としては次のことが考えられます。 • CBLDCRPC('OPEN ')が呼び出されていません。 • ユーザサービス定義の rpc_rap_auto_connect オペランドに Y が指定されていますが，CBLDCRAP('DISCNCT ')が呼び出されました。 • ユーザサービスネットワーク定義の dcsvgdef 定義コマンドで，-w オプションが指定されていません。
05503	メモリ不足が発生しました。
05505	rap リスナーとの通信でネットワーク障害が発生しました。
05506	rap リスナーとの通信でタイムアウトが発生しました。
05521	rap リスナーは停止中です。
05522	続行できない障害が発生しました。障害の要因として以下のことが考えられます。 • 予期しないメッセージを受信しました。 • 予期しない相手からのメッセージを受信しました。
05523	システムコールで予期しないエラーが発生しました。

注意事項

「05501」および「05502」以外のステータスコードで，CBLDCRAP('DISCNCT ')関数がエラーリターンした場合は，rap リスナーとのコネクションは解放されています。

UAP トレースに取得されるエラー要因コードは次のとおりです。

0：エラーなし。

1：一意名 1 が設定されていません。

2：要求コード（データ名 A）が間違っています。

リモートプロシジャコール（CBLDCRPC，CBLDCRSV）

クライアント/サーバ形態の通信をするときに使う，OpenTP1 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に関する COBOL-UAP 作成用プログラムを次に示します。

- CBLDCRPC('CALL ') － 遠隔サービスの要求
- CBLDCRPC('CLOSE ') － アプリケーションプログラムの終了
- CBLDCRPC('CLTSEND ') － CUP への一方通知
- CBLDCRPC('DISCARDF') － 処理結果の受信の拒否
- CBLDCRPC('DISCARDS') － 特定の処理結果の受信の拒否
- CBLDCRPC('GETCLADR') － クライアント UAP のノードアドレスの取得
- CBLDCRPC('GETERDES') － エラーが発生した非同期 RPC 応答型要求の記述子の取得
- CBLDCRPC('GETGWADR') － ゲートウェイのノードアドレスの取得
- CBLDCRPC('GETSVPRI') － サービス要求のスケジュールプライオリティの参照
- CBLDCRPC('GETWATCH') － サービスの応答待ち時間の参照
- CBLDCRPC('OPEN ') － アプリケーションプログラムの開始
- CBLDCRPC('POLLANYR') － 処理結果の非同期受信
- CBLDCRPC('SETSVPRI') － サービス要求のスケジュールプライオリティの設定
- CBLDCRPC('SETWATCH') － サービスの応答待ち時間の更新
- CBLDCRPC('SVRETRY ') － サービスプログラムのリトライ
- CBLDCRSV('MAINLOOP') － SPP のサービス開始

リモートプロシジャコールの COBOL-UAP 作成用プログラム（CBLDCRPC，CBLDCRSV）は，TP1/Server Base と TP1/LiNK のどちらの UAP でも使えます。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。リモートプロシジャコール（CBLDCRPC，CBLDCRSV）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCRPC.cbl と DCRSV.cbl に格納してあります。

CBLDCRPC('CALL ')

名称

遠隔サービスの要求

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 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).
```

機能

SPP のサービスを要求します。CBLDCRPC('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 CBLDCRPC('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。

CBLDCRPC('CALL ')を呼び出す UAP は，トランザクションとして実行していてもしていなくてもかまいません。トランザクションとして実行している処理から CBLDCRPC('CALL ')でサービスを要求するときは，要求するサービスの処理はトランザクションブランチとして稼働します。

CBLDCRPC('CALL ')を使う場合，サーバ UAP があるノードの OpenTP1 が稼働していることが前提です。

CBLDCRPC('CALL ')を実行して応答を待っている間にシグナルを受信しても，リターンされません。

ステータスコードの一覧のあとに，次の説明を掲載しています。CBLDCRPC('CALL ')の詳しい説明を知りたいときに参照してください。

(1)CBLDCRPC('CALL ')のデータ領域について

(2)CBLDCRPC('CALL ')がエラーになる場合

(3)CBLDCRPC('CALL ')がエラーになるタイミング

(4)CBLDCRPC('CALL ')がエラーになったときに再実行する指定

(5)サービス要求に優先度を付ける場合

(6)ステータスコード「00310」と「00306」の違い

(7)ステータスコード「00378」をリターンさせる指定

(8)ステータスコードと同期点処理の関係

(9)サービスを要求するときの注意

(10)ドメイン修飾をしてサービスを要求する場合

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

2 … 非同期応答型 RPC

1 … 非応答型 RPC

4 … 連鎖 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。

サービスグループ名にドメイン修飾をする場合は，RPC の形態を示す数値に「256」を加算します。ドメイン修飾をした RPC はトランザクションブランチにできません。そのため，トランザクションの処理から CBLDCRPC('CALL ')を使う場合は，必ず「256」に「32」を加算してください。

（例 1）

同期応答型 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 0 ＋ 32 ＝ 32 を設定します。

（例 2）

トランザクションの処理から，非同期応答型 RPC でドメイン修飾をしたサービス要求をする場合

データ名 C には 2 ＋（256 ＋ 32）＝ 290 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。

ドメイン修飾をしてサービスを要求するときは，サービスグループ名のあとに @（アットマーク）と DNS のドメイン名を付けて，文字列の最後には空白を設定してください。

●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 から DCRPC_MAX_MESSAGE_SIZE※までの範囲の長さが設定できます。DCRPC_MAX_MESSAGE_SIZE は，dcrpc.h で定義してあります。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

●データ名 H

サービスの入力パラメタを設定します。

●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。1 から DCRPC_MAX_MESSAGE_SIZE※までの範囲の長さが設定できます。DCRPC_MAX_MESSAGE_SIZE は，dcrpc.h で定義してあります。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

サーバ UAP 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。

●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，OpenTP1 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。
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。
00302	CBLDCRPC('OPEN ')を呼び出していません。
00304	メモリが不足しました。または，サービス要求先 SPP のメッセージ格納バッファプール（message_store_buflen オペランド）が不足したため，サービス要求を受け付けられませんでした。 サービス要求先 SPP のユーザサービス定義，またはユーザサービスデフォルト定義の message_store_buflen オペランドの指定値を見直してください。
00306	通信障害が起こりました。 ネットワークに障害が発生していないか確認してください。
	リモート API 機能を使用している場合は，CBLDCRPC('CALL ')の応答待ち時間を満了した可能性があります。CBLDCRPC('CALL ')の応答待ち時間の設定（watch_time オペランド，CBLDCRPC('SETWATCH')）を見直してください。 リモート API 機能使用時に CBLDCRPC('CALL ')の応答待ち時間を満了した場合のステータスコードを 00307 に変更するには，ユーザサービス定義に rap_extend_function オペランドを指定してください。 rap_extend_function オペランドについてはマニュアル「OpenTP1 システム定義」を参照してください。
00307	CBLDCRPC('CALL ')の応答待ち時間を満了しました。 CBLDCRPC('CALL ')の応答待ち時間の設定（watch_time オペランド，CBLDCRPC('SETWATCH')）を見直してください。
	サービス要求先 SPP が，サービスプログラムの実行中に異常終了しました。 サービス要求先 SPP が異常終了した要因を調査してください。
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。 データ名 G の設定値を見直してください。
00309	サービス要求先 SPP のサービスプログラムで設定した応答の長さ（サービスプログラムのデータ名 D）が，CBLDCRPC('CALL ')の応答の長さ（サービスプログラムのデータ名 I）を超えました。 サービス要求先 SPP のサービスプログラムでの応答の長さ（サービスプログラムのデータ名 D）の設定を見直してください。

ステータスコード	意味
00310	データ名 F に設定したサービスグループ名が不正であるか，データ名 F に設定したサービスグループのサービス要求先 SPP が起動されていません。 データ名 F の設定を見直すか，データ名 F に設定したサービスグループのサービス要求先 SPP を起動してください。
00311	データ名 E に設定したサービス名が不正であるか，サービス要求先 SPP でデータ名 E に設定したサービス名がユーザサービス定義ファイルの service オペランドに設定されていません。 データ名 E の設定を見直すか，データ名 E に設定したサービス名をサービス要求先 SPP の service オペランドに設定してください。
00312	データ名 F で設定したサービスグループのサービス要求先 SPP は，サーバ閉塞またはサービス閉塞しています。閉塞要因を調査し，閉塞を解除してください。
00313	サービス要求先 SPP は，終了処理中です。
00314	データ名 F に設定したサービスグループ名のサービス要求先 SPP は起動していません。または，サービス要求送信処理で通信障害が起きたおそれがあります。 データ名 F に設定したサービスグループ名のサービス要求先 SPP を起動してください。すでに起動している場合は，ネットワーク障害が発生していないかを確認してください。
	サービス要求の応答待ち時間（watch_time オペランド，CBLDCRPC('SETWATCH')）に 0 を指定している場合に，サービスプログラムを実行中のサービス要求先 SPP が異常終了しました。サービス要求先 SPP が異常終了した要因を調査してください。
00315	サービス要求先 SPP の OpenTP1 が起動していません。OpenTP1 が停止処理中であるか，サービス要求送信処理で通信障害が起きたおそれがあります。 サービス要求先 SPP の OpenTP1 を起動するか，ネットワーク障害が発生していないかどうかを確認してください。
00316	サービス要求先 SPP で，システムエラー（内部矛盾）が起こりました。
00317	サービス要求先 SPP で，メモリが不足しました。
00318	サービス要求元 UAP で，システムエラー（内部矛盾）が起こりました。
00319	サービス要求先 SPP のサービスプログラムで設定する応答の長さ（サービスプログラムのデータ名 D）が，1 から DCRPC_MAX_MESSAGE_SIZE※で定義されている値の範囲にありません。 サービス要求先 SPP のサービスプログラム内の応答の長さ（サービスプログラムのデータ名 D）の設定を見直してください。
00320	サービス要求先 SPP の OpenTP1 は，開始処理中です。
00323	サービス要求元 UAP，またはサービス要求先 SPP で，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，トランザクションブランチはロールバックします。 サービス要求元 UAP，またはサービス要求先 SPP で，不要なメモリを確保していないか見直してください。
00324	サービス要求元 UAP で，システムエラー（内部矛盾）が起こりました。このステータスコードが返った場合，トランザクションブランチはロールバックします。
00325	サービス要求先 SPP で，システムエラー（内部矛盾）が起こりました。このステータスコードが返った場合，トランザクションブランチはロールバックします。

ステータスコード	意味
00326	サービス要求先 SPP のサービスプログラムで設定した応答の長さ（サービスプログラムのデータ名 D）が，CBLDCRPC('CALL ')の応答の長さ（データ名 I）を超えました。このステータスコードが返った場合，トランザクションブランチはロールバックします。 サービス要求先 SPP のサービスプログラムで，応答の長さ（サービスプログラムのデータ名 D）の設定を見直してください。
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を使用している場合に，同一サービスグループ名のサービス要求先 SPP のトランザクション属性（atomic_update オペランド）が不一致になっています。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，要求元 UAP の OpenTP1 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が返るのは，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用している SPP にサービスを要求した場合だけです。 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用する SPP のトランザクション属性（atomic_update オペランド），または OpenTP1 のバージョンを見直してください。
	dcsvgdef 定義コマンドを使用して，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）のユーザサーバに対し，CBLDCRPC('CALL ')のデータ名 C に「32」を加算しないでサービスを要求しました。
00328	データ名 F のドメイン修飾をしたサービスグループ名のドメイン名が間違っています。 ドメイン名を見直してください。
00329	データ名 F にドメイン修飾をしてサービスを要求しましたが，ドメイン代表スケジュールサービスのポート番号が見つかりません。 システム共通定義の domain_masters_por オペランドの設定，および/etc/services のドメイン代表スケジューラサービスのポート番号の設定を見直してください。
00356	サービス要求先 SPP（ソケット受信型サーバ）が，サービス要求電文を受け取れません。 サービス要求先 SPP のユーザサービス定義，またはユーザサービスデフォルト定義の max_socket_msg オペランド，max_socket_msglen オペランドの設定を見直してください。
00366	オンラインテスタを使用している環境で，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。 UAP のテストモードの設定を見直してください。
00367	トランザクション属性の連鎖 RPC を実行したあと，データ名 C をトランザクションにしない設定にした状態で CBLDCRPC('CALL ')によってサービスを要求しています。
00370	サービス要求先 SPP は，セキュリティ機能で保護されています。 CBLDCRPC('CALL ')を実行したサービス要求元 UAP には，サービス要求先 SPP へのアクセス権限がありません。サービス要求先 SPP のアクセス権限を見直してください。
00372	サービス要求先 SPP の OpenTP1 で，同時に起動できるトランザクションブランチの数を超えました。そのため，トランザクションブランチを開始できません。 トランザクションサービス定義の trn_tran_process_count オペランドの設定を見直してください。
	サービス要求元 UAP が，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ました。そのため，トランザクションブランチを開始できません。

ステータスコード	意味
00372	トランザクションサービス定義の trn_max_subordinate_count オペランドの設定を見直してください。
	トランザクション内でドメイン修飾をしたサービス要求で，データ名 C に「32」を加算していません。
	サービス要求先 SPP で，リソースマネジャ（RM）にエラーが発生しました。そのため，トランザクションブランチを開始できません。 リソースマネジャ（RM）のエラー要因を解決したあと，再度実行してください。
	TP1/LiNK の「システム環境設定」ウィンドウの「トランザクション機能」が［あり］に設定されていません。 TP1/LiNK の「システム環境設定」の「トランザクション機能」の設定を見直してください。
00378	サービス要求先 SPP が，サービスプログラムの実行中に異常終了しました。 サービス要求先 SPP のサービスプログラム処理を見直してください。このステータスコードが返るのは，ユーザサービス定義の rpc_extend_function オペランドに"00000001"を指定したサービス要求元 UAP の場合だけです。rpc_extend_function オペランドに"00000000"を指定，またはオペランドを省略した場合は，このステータスコードではなく，「00307」または「00314」が返ります。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

（1）CBLDCRPC('CALL ')のデータ領域について

CBLDCRPC('CALL ')のデータ領域について説明します。

サーバ UAP に渡す値

サービスを要求するときには，サービスプログラムから返ってくる応答の領域（データ名 J）を確保しておきます。クライアント UAP では CBLDCRPC('CALL ')に次の値を設定します。

- 入力パラメタ（データ名 H）
- 入力パラメタ長（データ名 G）
- 応答の長さ（データ名 I）

入力パラメタ，入力パラメタ長，応答の長さは，クライアント UAP の CBLDCRPC('CALL ')で設定した値が，そのままサービスプログラムに渡されます。文字コードや数字の表記形式を変えたい場合は，クライアント UAP，または要求されたサービスプログラムの処理で変換してください。応答を返さないサービスプログラムのサービスを要求するときは，応答の長さを設定しても無視されます。

入力パラメタ長と応答の長さの最大値は，それぞれヘッダファイル dcrpc.h の DCRPC_MAX_MESSAGE_SIZE※で宣言しています。最大値を確認する場合は，dcrpc.h の内容を参照してください。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

サーバ UAP から戻ってくる値

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 I はサービスプログラムから実際に返ってきた応答の長さです。データ名 J とデータ名 I を参照できる場合を次に示します。

- 同期応答型 RPC，連鎖 RPC の場合
  CBLDCRPC('CALL ')がリターンしたあと。
- 非同期応答型 RPC の場合
  CBLDCRPC('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 I は参照できません。
- 非応答型 RPC の場合
  データ名 J とデータ名 I は参照できません。

CBLDCRPC('CALL ')，または CBLDCRPC('POLLANYR')がエラーリターンした場合には，データ名 J とデータ名 I は参照できません。

返ってきた応答が，クライアント UAP で確保した応答の領域（データ名 I）よりも大きい場合は，ステータスコード「00309」でエラーリターンします。

データ名 C に設定する値

データ名 C に設定した値と CBLDCRPC('CALL ')の実行結果について説明します。

- 同期応答型 RPC（データ名 C に 0 を設定）
  CBLDCRPC('CALL ')は，応答が返るか，通信先でエラーが起こるまでリターンしません。
- 非同期応答型 RPC（データ名 C に 2 を設定）
  CBLDCRPC('CALL ')は，すぐにリターンします。ただし，応答と応答の長さを参照できるのは，CBLDCRPC('POLLANYR')で非同期に応答を受信したあとです。応答（データ名 J）の領域は，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。
  - CBLDCRPC('POLLANYR')で応答を受け取る
  - CBLDCRPC('DISCARDF')で応答の受信を拒否
  - トランザクションの処理からサービスを要求した場合は，コミット，またはロールバックした

トランザクションの処理で非同期応答型 RPC を使う場合，同期点処理（コミット，またはロールバック）前に CBLDCRPC('POLLANYR')で応答を受け取ってください。**同期点処理後には，CBLDCRPC('POLLANYR')で応答は受け取れません。**

CBLDCRPC('POLLANYR')が受信する応答を特定する場合は，CBLDCRPC('CALL ')がリターンしたときに返された正の整数（記述子）を，CBLDCRPC('POLLANYR')の引数に設定します。受信する応答を特定する場合は，CBLDCRPC('CALL ')のデータ名 D の値を保持しておいてください。

なお，非トランザクションの処理で，同期点処理後に応答を受け取りたい場合は，システムサービス定義の rpc_extend_function オペランドで該当するオプションを指定する必要があります。
rpc_extend_function については，マニュアル「OpenTP1 システム定義」を参照してください。

- **非応答型 RPC（データ名 C に 1 を設定）**

  CBLDCRPC('CALL ')は，サービスプログラムの処理が終わるのを待たないで，すぐにリターンします。サービスプログラムは応答を返さないサービスと見なされます。そのため，サービスプログラムが実行されたかどうかは，サービスを要求した UAP からはわかりません。非応答型 RPC を設定した場合は，応答（データ名 J）と応答の長さ（データ名 I）の値は参照できません。

- **連鎖 RPC（データ名 C に 4 を設定）**

  CBLDCRPC('CALL ')は，応答が返るか，通信先でエラーが起こるまでリターンしません。連鎖 RPC で同じサービスグループに属するサービスを複数回要求する場合は，最初の要求時と同じプロセスで実行できます。

  連鎖 RPC を使う場合には，次に示す制限があります。

  1. 2 回目以降の CBLDCRPC('CALL ')では，ユーザサーバ，およびサービスの閉塞を検出できません。
  2. 2 回目以降の CBLDCRPC('CALL ')でサービスプログラムの処理で異常が起こった場合には，ユーザサーバ全体が閉塞します。サービス単位では閉塞しません。

(2) CBLDCRPC('CALL ')がエラーになる場合

CBLDCRPC('CALL ')のエラーリターンする理由について説明します。

サーバ UAP があるノードの OpenTP1 が稼働していない場合

サービスを要求される OpenTP1 が稼働していない場合は，CBLDCRPC('CALL ')は次に示すどれかのステータスコードでエラーリターンします。

「00306」

「00314」

「00315」

「00320」

サーバ UAP が稼働していない場合

サーバ UAP がマルチサーバの場合，異常終了中，または部分回復処理中であっても，サービス要求は OpenTP1 で新しく起動されたプロセスで実行されます。ただし，次に示す場合は，CBLDCRPC('CALL ')はエラーリターンします。

1. 閉塞している SPP へはサービスを要求できません。サービスグループが閉塞されている場合は，「00312」でエラーリターンします。
2. ユーザサーバの停止コマンド（dcsvstop コマンド）または OpenTP1 の停止コマンド（dcstop コマンド）で，SPP が終了処理中または終了している場合は，次に示すどれかのステータスコードでエラーリターンします。

   「00310」

   「00312」

   「00313」

   これらのうち，どのステータスコードが返るかは，CBLDCRPC('CALL ')を呼び出したタイミングで決まります。
3. OpenTP1 が開始処理中の場合は，「00320」でエラーリターンします。この場合は，サーバ UAP，または OpenTP1 が起動完了したあとに正常にサービスを要求できることがあります。OpenTP1 は，メッセージ ID KFCA01809-I のメッセージログが出力されると起動を完了するので，このメッセージが表示されてから再びサービスを要求してください。

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サービスを要求した場合

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の環境では，該当するサービスのスケジュールが閉塞していると，OpenTP1 が自動的にほかのノードにサービス要求を転送します。ただし，次に示す条件の場合 CBLDCRPC('CALL ')は「00327」でエラーリターンします。

1. トランザクション処理の場合，転送しようとした先のノードにあるサービスのトランザクション属性が，閉塞していたサービスと一致していないとき。
2. 転送しようとした先のノードにある OpenTP1 のバージョンが，サービスを要求した OpenTP1 のノードのバージョンよりも低いとき。

これらのエラーリターンが起こった場合は，次に示す処置をしてください。

1. 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構成する複数の SPP のトランザクション属性を一致させてください。
2. 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構成する複数の OpenTP1 のバージョンを一致させてください。

ソケット受信型サーバへサービスを要求する場合

ソケット受信型サーバでは，ユーザサービス定義の max_socket_msg オペランドと max_socket_msglen オペランドの指定で，データの輻輳制御をしています。そのため，定義した値を超えた場合，サービス要求を受信できない場合があります。このとき CBLDCRPC('CALL ')は，「00356」でエラーリターンしま

す。このステータスコードが返った場合，クライアント UAP はしばらく時間をおいてから再実行すれば，サービスを要求できることがあります。

連鎖 RPC を使った場合

トランザクションとして処理している連鎖 RPC を使っている UAP から，同じサーバ UAP へトランザクションでない CBLDCRPC('CALL ')を呼び出すと，「00367」でエラーリターンします。

オンラインテスタを使っている場合

オンラインテスタを使っている場合に，テストモードの UAP とテストモードでない UAP との間で CBLDCRPC('CALL ')を呼び出すと，「00366」でエラーリターンします。

セキュリティ機能を使っている場合

目的のサービスがセキュリティ機能で保護されていて，CBLDCRPC('CALL ')を呼び出したクライアント UAP に SPP へのアクセス権がない場合は，「00370」でエラーリターンします。

(3) CBLDCRPC('CALL ')がエラーになるタイミング

サービスを要求された SPP が異常終了した場合，クライアント UAP でエラーが返るタイミングについて説明します。

- 同期応答型 RPC または連鎖 RPC（データ名 C に 0 または 4 を設定）
  サービスを実行する SPP の処理が終わる前に異常終了すると，「00307」でエラーリターンします。クライアント UAP のユーザサービス定義 watch_time オペランドに無限を指定している場合は，「00314」でエラーリターンします。
- 非同期応答型 RPC（データ名 C に 2 を設定）
  サービスを実行する SPP の処理が終わる前に異常終了すると，上記のステータスコードが CBLDCRPC('POLLANYR')にエラーリターンします。
- 非応答型 RPC（データ名 C に 1 を設定）
  サーバ UAP の異常終了をクライアント UAP では検知できません。

クライアント UAP の時間監視でエラーになる場合

次に示す場合には，クライアント UAP のユーザサービス定義の watch_time オペランドに指定した時間が経過したあとで，「00307」でエラーリターンします。

- SPP があるノードの OpenTP1 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，またはサーバ UAP の処理が完了してからクライアント UAP に結果が届く前に障害が起こった場合

(4) CBLDCRPC('CALL ')がエラーになったときに再実行する指定

開始処理中，または系切り替え中などでサービス要求先の OpenTP1 が起動していない場合でも，CBLDCRPC('CALL ')をエラーとしないで，OpenTP1 で要求先検索，およびサービス要求送信を再実行させることができます。

要求先検索，およびサービス要求送信を再実行させる場合は，システム共通定義に rpc_retry オペランドに Y を指定してください。要求先検索とサービス要求送信の再実行回数，および再実行間隔は，システム共通定義の rpc_retry_count オペランドと rpc_retry_interval オペランドで指定します。システム共通定義で指定した再実行回数を超えた場合は，CBLDCRPC('CALL ')は次に示すどれかのステータスコードでエラーリターンします。

「00301」

「00306」

「00310」

「00314」

「00315」

「00320」

(5) サービス要求に優先度を付ける場合

サービス要求のスケジュールプライオリティは，CBLDCRPC('CALL ')を呼び出す直前に，CBLDCRPC('SETSVPRI')を呼び出して設定します。プライオリティを設定しない場合は，スケジュールサービスの省略時の解釈で，サービス要求の優先度が決まります。

(6) ステータスコード「00310」と「00306」の違い

上記のステータスコードは，該当するサービスグループ名のユーザサーバが見つからなかった場合にリターンされます。

- 「00310」
  システム共通定義の all_node オペランドに指定した，すべてのノードを探して見つからなかったことを示します。
- 「00306」
  探している途中で，all_node オペランドに指定した一つ以上のノードとの通信で障害が起こったことを示します。これは，該当する OpenTP1 システムがなかった場合も含みます。

(7) ステータスコード「00378」をリターンさせる指定

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たことを「00307」または「00314」以外のステータスコードで判別したい場合には，ユーザサービス定義の rpc_extend_function オペランドに"00000001"を指定します。この指定をすると，上記のエラー時に「00378」がリターンされるようになります。rpc_extend_function オペランドに"00000000"を指定，またはオペランドを省略した場合は，「00378」はリターンされないで，「00307」または「00314」がリターンされます。

(8) ステータスコードと同期点処理の関係

CBLDCRPC('CALL ')のステータスコードと同期点処理（コミット，ロールバック）の関係について説明します。ここで説明する内容は，サービス要求がトランザクション処理になる場合に該当します。トランザクションでないサービス要求（データ名 C の RPC の形態を示す数値に「32」を加算した場合も含む）には該当しません。

CBLDCRPC('CALL ')がエラーリターンしてもコミットとなる場合

サービスを要求されたサービスプログラムの異常終了や，ノードの障害，ネットワーク障害の場合でも，「00307」がリターンすることがあります。クライアント UAP がトランザクション処理でない場合は，「00307」が返っても，要求したサービスがあるサービスプログラムは正常に終了していて，データベースへの更新などが実行されているときもあります。

ロールバック処理が必要なステータスコード

トランザクション処理から呼び出した CBLDCRPC('CALL ')がエラーリターンした場合，ステータスコードによっては，必ずトランザクションがロールバック（サーバ UAP が rollback_only 状態）になります。この場合，コミット，またはロールバックの COBOL-UAP 作成用プログラムのどちらを使っても，必ずロールバックになります。必ずロールバックになる CBLDCRPC('CALL ')のステータスコードを次に示します。

「00311」

「00317」

「00319」

「00326」

(9) サービスを要求するときの注意

1. CBLDCRPC('CALL ')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 の環境設定で指定しておいてください。間違ったサービスグループ名またはサービス名を CBLDCRPC('CALL ')に設定してサービスを要求すると，「00310」または「00311」でエラーリターンします。

   応答を返さないサービスプログラムの場合，「00310」はリターンしません。

2. サーバ UAP は，クライアント UAP とは別のプロセスで実行します。そのため，通常の手続き呼び出しとは，次の点が異なります。

   - OS がクライアント UAP のプロセスに与えている属性は，サーバ UAP へは引き継がれません（例えば，環境変数，スケジュールの優先順位（nice 値）など）。
   - クライアント UAP のノードで設定した OpenTP1 の環境設定は，サーバ UAP の OpenTP1 へは引き継がれません（例えば，トランザクション属性の指定有無，トランザクションブランチの限界経過時間，スケジュールの優先順位など）。

3. 入力パラメタ，およびサービスプログラムの応答に，同じバッファ領域を設定しないでください。

4. データ名 C に「1」（非応答型 RPC）を設定した場合には，次に示すステータスコードは返りません。

- 起こらないエラー
  「00309」
  「00319」
- 起こっても検出できないエラー
  「00311」
  「00313」
  「00316」
  「00370」

また，エラー発生時に OpenTP1 にメッセージを出力することもありません。エラーを検出しなければならない場合，データ名 C に「0」を設定する（同期応答型 RPC）ことを検討してください。

5. トランザクションの処理から CBLDCRPC('CALL ')でサービスグループを呼び出すと，トランザクションが決着するまで一つの SPP を占有します。同じトランザクションの処理から，CBLDCRPC('CALL ')で同じサービスを複数回使う場合は，次に示すようにしてください。
   - 使う回数に応じて，ユーザサービス定義の balance_count オペランドおよび parallel_count オペランドの指定値を見積もり直す。
   - プロセスが増えないように，連鎖 RPC でサービスを要求する。

   balance_count オペランドおよび parallel_count オペランドの指定値に誤りがあると，トランザクションが異常終了したり，デッドロックが起こったりする場合があります。
6. 非同期応答型 RPC を使う場合には，CBLDCRPC('POLLANYR')ですべての非同期の応答を受信するか，CBLDCRPC('DISCARDF')で非同期の応答を拒否するまで，サーバ UAP（SPP）を占有する場合があります。これは，トランザクションの処理の場合にもトランザクションの処理でない場合にも起こります。非同期応答型 RPC を使う場合には，使う回数に応じて，常駐プロセス数の指定を増やしておいてください。

   非同期応答型 RPC は，SPP を占有すること以外にも多くの資源を必要とします。UAP の処理効率が下がったり，必要ない SPP が開始されたりすることを防ぐため，非同期応答型 RPC の CBLDCRPC('CALL ')を使ったあとには，確実に応答を受信するか，受信を拒否するかしてください。
7. 非同期応答型 RPC でサービス要求を連続して複数回使ったあとで応答を受信する場合には，非同期応答型 RPC のサービスを要求するときに，それぞれのサービス要求に対して，次のように異なる応答格納領域を設定してください。同じ領域を設定すると，あとから受信した応答が上書きされて，応答を正しく受信できなくなります。

```
01 一意名3-1.
   02 データ名I1 PIC S9(9) COMP.
   02 データ名J1 PIC X(n).
01 一意名3-2.
   02 データ名I2 PIC S9(9) COMP.
   02 データ名J2 PIC X(n).
01 一意名3-3.
   02 データ名I3 PIC S9(9) COMP.
   02 データ名J3 PIC X(n).
```

8. 非同期応答型 RPC でサービス要求したサーバ UAP（SPP）は，非同期応答型 RPC を実行したプロセスが CBLDCRPC('POLLANYR')を発行したかどうかにかかわらず，サービス関数実行後にすぐに応答を送信します。CBLDCRPC('POLLANYR')を発行しないで，複数回の非同期応答型 RPC を大量に実行すると，SPP から送信される応答が TCP/IP のバッファにたまり，SPP の応答送信処理が失敗する場合があります。SPP で応答送信処理が失敗すると，非同期応答型 RPC 発行元では，CBLDCRPC('POLLANYR')を発行しても，SPP から応答を受信することはできません。
9. トランザクション属性の非同期応答型 RPC，または非応答型 RPC を大量に実行すると，SPP から送信されるトランザクションについてのメッセージを受信できなくなり，トランザクションがロールバックすることがあります。

（10）ドメイン修飾をしてサービスを要求する場合

ドメイン修飾したサービスグループ名を設定すると，DNS のドメイン内にある OpenTP1 のサービスを要求できます。ドメイン修飾をしたサービスグループ名は，サービスグループ名のあとに @ と DNS のドメイン名を続けて設定します。

ドメイン修飾をしてサービスを要求するときの注意

1. ドメイン修飾をしてサービスを要求する場合は，CBLDCRPC('CALL ')のデータ名 C の RPC の形態を示す数値に（256 + 32）を加算します。（256 + 32）を加算しないでドメイン修飾をしたサービスグループ名を設定すると，CBLDCRPC('CALL ')は「00310」でエラーリターンします。
2. ドメイン修飾した RPC は，CBLDCRPC('CALL ')を呼び出したプロセスがトランザクションの処理でも，トランザクションを拡張できません。そのため，トランザクションの処理からドメイン修飾をしたサービス要求をする場合は，データ名 C の RPC の形態を示す数値に（256 + 32）を加算して，トランザクションを拡張しないようにしてください。なお，ドメイン名に自ドメインを設定した場合でも，トランザクションを拡張できません。
3. ドメイン修飾をした RPC では，キュー受信型サーバにだけサービスを要求できます。ソケット受信型サーバへは，ドメイン修飾をしたサービス要求はできません。
4. ドメイン修飾をしたサービス要求では，namdomainsetup コマンドで登録したホストで起動されるドメイン代表スケジュールサービスへサービス要求を送信します。ドメイン代表スケジュールサービスのポート番号は，/etc/services から取得します。サービス要求の送信で障害が起こった場合は，namdomainsetup コマンドで複数のホスト名を登録していれば，順次送信を試みます。ドメイン修飾をした RPC が正常に終了しても，ドメイン代表スケジュールサービスへの送信で障害が起こっている場合もあります。

ドメイン修飾をしたサービス要求をする前の準備

ドメイン修飾をした RPC では，次に示す環境設定をしてください。

1. DNS のドメインデータファイルへ，ドメイン代表スケジュールサービスを起動するホスト名を登録します。登録するときは，namdomainsetup コマンドを使います。
2. ドメイン修飾をしたサービス要求をする OpenTP1 を起動するホストの /etc/services へ，ドメイン代表スケジュールサービスのポート番号を設定します。ポート番号は，次の形式で記述してください。

```
    OpenTP1scd  ポート番号/tcp
```

3. ドメイン代表スケジュールサービスを起動する OpenTP1 のスケジュールサービス定義の scd_port オペランドに，ドメイン代表スケジュールサービスのウェルノウンポートを指定します。

指定例

サービス「ADD」を実行するクライアント UAP

```
01 一意名1.
   02 データ名A     PIC X(8) VALUE 'CALL    '.
   02 STATUS-CODE  PIC X(5) VALUE SPACES.
   02 FILLER       PIC X(3).
   02 FLAGS        PIC S9(9) COMP VALUE ZERO.
   02 DESCRIPTOR   PIC S9(9) COMP VALUE ZERO.
   02 SERVICE-NAME PIC X(32) VALUE SPACES.
   02 GROUP-NAME   PIC X(32) VALUE SPACES.
01 一意名2.
   02 NAME-LENG    PIC S9(9) COMP.
   02 NAME         PIC X(32) VALUE SPACES.
01 一意名3.
   02 RESULT-LENG  PIC S9(9) COMP.
   02 RESULT       PIC X(20).
*
 MOVE 'ADDRESS-BOOK ' TO GROUP-NAME.
 MOVE 'ADD' TO SERVICE-NAME.
 MOVE 'SATO' TO NAME.
 MOVE 4 TO NAME-LENG.
 MOVE 20 TO RESULT-LENG.
 CALL 'CBLDCRPC' USING 一意名1 一意名2 一意名3.
 IF STATUS-CODE NOT = '00000' THEN
    DISPLAY 'FAILED'
 END-IF.
 STOP RUN.
```

サービス「ADD」を実行するサーバ UAP

```
 PROGRAM-ID. ADD.
 DATA DIVISION.
 LINKAGE SECTION.
 01 一意名1.
    02 ADD-NAME     PIC X(20).
 01 一意名2.
    02 NAME-LENG    PIC S9(9) COMP.
 01 一意名3.
    02 RESULT       PIC X(20).
 01 一意名4.
    02 RESULT-LENG  PIC S9(9) COMP.
      :
      :
 PROCEDURE DIVISION USING 一意名1 一意名2 一意名3 一意名4.
      :
* 受け取った一意名1の内容を処理する
     :
 MOVE 'ADD-COMPLETE' TO RESULT.
```

```
MOVE 12 TO RESULT-LENG.
EXIT PROGRAM.
```

注意事項

システム共通定義の all_node 句で指定したドメイン以外の OpenTP1 システムにトランザクショナル RPC を行う場合，自ドメインおよび他ドメイン内のすべての OpenTP1 システムのノード識別子(システム共通定義の node_id オペランド)は一意にする必要があります。また，すべての OpenTP1 システムは，バージョン 03-02 以降にする必要があります。これらの条件を満たしていないと，トランザクションが正しく回復できなくなる場合があります。

CBLDCRPC('CLOSE ')

名称

アプリケーションプログラムの終了

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’CLOSE   ’.
   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

OpenTP1 の COBOL-UAP 作成用プログラムを使うための環境をクローズします。CBLDCRPC('CLOSE ')を呼び出したあとには，OpenTP1 の COBOL-UAP 作成用プログラムは使えません。

CBLDCRPC('CLOSE ')は，メインプログラムで呼び出します。プロセスで 1 回だけ呼び出してください。

CBLDCRPC('CLOSE ')は，UAP の正常終了を OpenTP1 に知らせます。

CBLDCRPC('CLOSE ')を呼び出さないで UAP を終了すると，UAP の異常終了と見なされて，サービスグループの閉塞やプロセスの再起動の対象になることがあります。また，OpenTP1 で管理する各種資源が解放されないままになって，そのあとの処理に影響を与えることがあります。OpenTP1 の UAP はすべて，CBLDCRPC('OPEN ')を呼び出したあとで，STOP RUN で終了する前に，必ず CBLDCRPC('CLOSE ')を呼び出してください。

CBLDCRPC('OPEN ')がエラーリターンした場合でも，必ず CBLDCRPC('CLOSE ')を呼び出してください。

CBLDCRPC('CLOSE ')を呼び出したあとは，CBLDCRPC('OPEN ')は呼び出せません。

UAP で値を設定するデータ領域

●データ名 A

アプリケーションプログラムの終了を示す要求コードを「VALUE 'CLOSE△△△'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

CBLDCRPC('CLTSEND ')

名称

CUP への一方通知

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’CLTSEND ’.
   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(n).
01 一意名2.
   02 データ名F PIC S9(9) COMP.
   02 データ名G PIC X(n).
```

機能

CUP へ一方的にデータを送ります。データ名 D とデータ名 E で示すホストのポート番号に対応するプロセス（CUP）に，データ名 G で示すデータをデータ名 F で示す長さだけ送ります。送信できるデータの長さは，0 から DCRPC_MAX_MESSAGE_SIZE※で示すバイト長までです。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

CBLDCRPC('CLTSEND ')で送信したデータは，TP1/Client/W または TP1/Client/P のライブラリ関数 dc_clt_chained_accept_notification 関数または dc_clt_accept_notification 関数で受け取ります。これらの関数については，マニュアル「OpenTP1 クライアント使用の手引 TP1/Client/W，TP1/Client/P 編」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

CUP への一方通知を示す要求コードを「VALUE 'CLTSEND '」と設定します。

●データ名 C

0 を設定します。

●データ名 D

送信先プロセスのポート番号を設定します。

●データ名 E

送信先のホスト名を設定します。指定できるホスト名は，1 から 255 文字までの文字列です。文字列の後ろには，空白文字を付けてください。

●データ名 F

送信するデータの長さを設定します。

●データ名 G

送信するデータを設定します。

OpenTP1 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。
00301	引数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。
00306	ネットワークに障害が起こりました。
00302	CBLDCRPC('OPEN ')を呼び出していません。
00304	メモリが不足しました。
00308	送信するデータの長さが，DCRPC_MAX_MESSAGE_SIZE※を超えています。
00314	送信先プロセスがありません。
	ネットワークに障害が起こりました。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

注意事項

1. CBLDCRPC('CLTSEND ')は，送信先のプロセスが TP1/Client/W または TP1/Client/P の dc_clt_chained_accept_notification 関数または dc_clt_accept_notification 関数を呼び出している

ことが明らかな場合にだけ，使ってください。送信先プロセスが稼働していない場合は，CBLDCRPC('CLTSEND ')は「00314」でエラーリターンします。

2. CBLDCRPC('CLTSEND ')の正常リターンは，RPC の通信プロトコル（TCP/IP）レベルでの送信が完了したことを意味します。そのため，CBLDCRPC('CLTSEND ')が正常に終了しても，CUP が dc_clt_chained_accept_notification 関数または dc_clt_accept_notification 関数でデータを正常に受信したことを確認できません。
3. CBLDCRPC('CLTSEND ')でデータを通知できるのは，CUP の dc_clt_chained_accept_notification 関数または dc_clt_accept_notification 関数だけです。SPP のプロセスや自プロセスへは，データを送信できません。
4. AIX 版 TP1/Server Base で，80 文字～255 文字のホスト名を指定する場合，libbetran2.a を使用して UAP を再コンパイルおよび再リンケージしてください。

CBLDCRPC('DISCARDF')

名称

処理結果の受信の拒否

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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。CBLDCRPC('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 CBLDCRPC('DISCARDF')を呼び出してください。呼び出さないと，CBLDCRPC('POLLANYR')が不要な応答を受信してしまうことがあります。

CBLDCRPC('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 を複数回使ったあとで，最初に返ってきた応答だけを必要とする場合

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。

OpenTP1 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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

CBLDCRPC('DISCARDS')

名称

特定の処理結果の受信の拒否

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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。CBLDCRPC('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

ゲートウェイのノードアドレスの取得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 CBLDCRPC('CALL ')が正常に終了したときに返された記述子を設定します。

OpenTP1 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。
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。
00302	CBLDCRPC('OPEN ')を呼び出していません。
00322	データ名 D に指定した記述子は存在しません。要因としては次のことが考えられます。 • 指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

CBLDCRPC('GETCLADR')

名称

クライアント UAP のノードアドレスの取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’GETCLADR’.
   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

クライアント UAP のプロセスが稼働するノードアドレスを，サーバ UAP で取得します。リターンしたアドレス値で，クライアント UAP のセキュリティチェックができます。

返されたアドレスを使って，サービスの応答やエラーの応答などは送信できません。

CBLDCRPC('GETCLADR')は，サービスプログラムから呼び出してください。サービスプログラム以外から呼び出した場合の処理は保証しません。

UAP で値を設定するデータ領域

●データ名 A

クライアント UAP のノードアドレスの取得を示す要求コードを「VALUE 'GETCLADR'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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

クライアント UAP のノード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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

注意事項

次の条件が重なった場合，CBLDCRPC('GETCLADR')が返すクライアント UAP のノードアドレスは，実際にクライアント UAP が通信で使用したノードアドレスと異なる場合があります。

- リモート API 機能を使用して，サービス要求を受け付けた。
- クライアント UAP が存在するホストがマルチホームドホスト形態である。

CBLDCRPC('GETERDES')

名称

エラーが発生した非同期応答型 RPC 要求の記述子の取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 CBLDCRPC('POLLANYR')がエラーリターンした直後に呼び出すことで，エラーが発生したサービス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，SPP 側でエラーが発生した場合だけです。

CBLDCRPC('POLLANYR')の呼び出し側でエラーが発生した場合には，CBLDCRPC('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。

OpenTP1 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

ステータスコード	意味
00000	正常に終了しました。
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。

CBLDCRPC('GETGWADR')

名称

ゲートウェイのノードアドレスの取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’GETGWADR’.
   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

アプリケーションゲートウェイ型ファイアウォールなどの，ゲートウェイを介してクライアント UAP からのサービス要求を受信した時，サーバ UAP でゲートウェイのノードアドレスを取得します。

リモート API 機能を使用してサービスを要求した場合，サーバ UAP で，ゲートウェイのノードアドレスを取得できます。

返されたアドレスを使って，サービスの応答やエラーの応答などの送信はできません。

CBLDCRPC('GETGWADR')は，サービスプログラムから呼び出してください。サービスプログラム以外から呼び出した場合の処理は保証しません。

UAP で値を設定するデータ領域

●データ名 A

ゲートウェイのノードアドレスの取得を示す要求コードを「VALUE 'GETGWADR'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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

ノードアドレスが返されます。リモート API 機能を使用していない場合は，0 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。
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。
00302	サービスプログラムから呼び出されていません。

CBLDCRPC('GETSVPRI')

名称

サービス要求のスケジュールプライオリティの参照

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’  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 ’GETSVPRI’.
   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

CBLDCRPC('SETSVPRI')で設定したサービス要求のスケジュールプライオリティを参照します。CBLDCRPC('GETSVPRI')がリターンする値は，UAP から CBLDCRPC('SETSVPRI')を再び呼び出すまで変わりません。

次に示す場合は，CBLDCRPC('GETSVPRI')は省略時仮定値「4」をデータ名 C にリターンします。

- UAP から CBLDCRPC('SETSVPRI')を呼び出していない場合
- CBLDCRPC('SETSVPRI')のデータ名 C に 0 を設定して呼び出している場合

UAP で値を設定するデータ領域

●データ名 A

サービス要求のスケジュールプライオリティの参照を示す要求コードを「VALUE 'GETSVPRI'」と設定します。

OpenTP1 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

スケジュールプライオリティが，1 から 8 の範囲の値で返されます。

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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

CBLDCRPC('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 ’CBLDCRPC’ 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 VALUE ZERO.
```

機能

現在のサービス要求の応答待ち時間を参照します。CBLDCRPC('GETWATCH')は，CBLDCRPC('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。

CBLDCRPC('GETWATCH')は，CBLDCRPC('SETWATCH')で変更したサービス応答待ち時間をリターンします。変更していない場合は，次に示す値をリターンします。

- TP1/Server Base の場合：システム共通定義の watch_time オペランドの値
- TP1/LiNK の場合：180 秒

CBLDCRPC('GETWATCH')で得られる値は，OpenTP1 のリモートプロシジャコール（CBLDCRPC('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。

●データ名 C

0 を設定します。

OpenTP1 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

現在のサービス応答待ち時間が返されます。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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

CBLDCRPC('OPEN ')

名称

アプリケーションプログラムの開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’OPEN    ’.
   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

OpenTP1 の COBOL-UAP 作成用プログラムを使う準備をします。

CBLDCRPC('OPEN ')は，メインプログラムで呼び出します。すべての OpenTP1 の COBOL-UAP 作成用プログラムの呼び出しに先立って，プロセスで 1 回だけ呼び出してください。CBLDCRPC('OPEN ')が実行する初期化手順を次に示します。

1. プロセス間通信をするためのエントリポイントを開きます。
2. OpenTP1 で使う共用メモリを取得します。
3. OpenTP1 に UAP を開始することを通知して，プロセスの監視を要求します。
4. そのほか，設定した UAP の実行環境に従って，OpenTP1 の各機能を初期化します。

UAP をトランザクション属性であると指定している場合は，そのノードでは OpenTP1 のトランザクションサービスとプロセスサービスが稼働していることが前提となります。

CBLDCRPC('OPEN ')は，OpenTP1 が正常に開始したあとで呼び出されないと有効になりません。OpenTP1 が正常開始する前に CBLDCRPC('OPEN ')を呼び出すと，ステータスコード「00315」でエラーリターンします。この場合，リモートプロシジャコールは使えません。

UAP トレースは，CBLDCRPC('OPEN ')が正常に終了したあとで呼び出した，すべての OpenTP1 の COBOL-UAP 作成用プログラムに対して取得されます。そのため，CBLDCRPC('OPEN ')がエラーリターンした場合，UAP トレースは取得されているときもあれば取得されていないときもあります。

UAP で値を設定するデータ領域

●データ名 A

UAP のサービス開始を示す要求コードを「VALUE 'OPEN△△△△'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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。
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。
00302	CBLDCRPC('OPEN ') は，すでに呼び出しています。
00303	初期化処理に失敗しました。以降，OpenTP1 の COBOL-UAP 作成用プログラムは呼び出せません。
00315	UAP があるノードの OpenTP1 が稼働していません。
00369	待機系のユーザサーバが，待機の終了を要求されました。
00371	セキュリティ機能を使った OpenTP1 の場合で，セキュリティ環境の初期化処理でエラーが起こりました。

注意事項

CBLDCRPC('OPEN ')は，定義ファイルで設定した内容に従って，UAP が使う OpenTP1 の各機能の環境設定（初期化）をします。

初期化処理では，UAP プロセスでオープンするファイルディスクリプタの最大数を OS に対して設定します。したがって，CBLDCRPC('OPEN ')を呼び出したあとには，UAP プロセスでオープンするファイルディスクリプタの最大数を OS に対して再設定（変更）しないでください。変更した場合の動作は保証できません。

CBLDCRPC('POLLANYR')

名称

処理結果の非同期受信

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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非同期に受信します。

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 CBLDCRPC('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。

CBLDCRPC('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切れになった場合

CBLDCRPC('POLLANYR')が正常に終了すると，非同期応答型 RPC を使った CBLDCRPC('CALL ')の応答を格納する領域に，値が設定されます。

ステータスコードの一覧のあとに，次の説明を掲載しています。CBLDCRPC('POLLANYR')の詳しい説明を知りたいときに参照してください。

(1)CBLDCRPC('POLLANYR')のデータ名 F について

(2)CBLDCRPC('POLLANYR')がエラーになるタイミング

(3)ステータスコード「00378」をリターンさせる指定

(4)ステータスコードと同期点処理の関係

(5)CBLDCRPC('POLLANYR')で応答が受け取れない場合

(6)CBLDCRPC('POLLANYR')を使うときの注意

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…待ち時間の単位は秒で指定し，CBLDCRPC('POLLANYR')で受信する非同期の応答を特定しません。

1…待ち時間の単位は秒で指定し，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…待ち時間の単位はミリ秒で指定し，CBLDCRPC('POLLANYR')で受信する非同期の応答を特定しません。

17…待ち時間の単位はミリ秒で指定し，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 CBLDCRPC('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

CBLDCRPC('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秒単位で設定します。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。

CBLDCRPC('POLLANYR')で非同期応答を受信する場合は，UAP に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，応答が返っていないと「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。

●データ名 D

0 を設定します。

OpenTP1 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 CBLDCRPC('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 CBLDCRPC('POLLANYR')が正常に終了した場合には，0 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。
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 を設定した場合に返されます。
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。
00302	CBLDCRPC('OPEN ')を呼び出していません。
00304	メモリが不足しました。
00306	ネットワークに障害が起こりました。
00307	CBLDCRPC('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。
00308	CBLDCRPC('CALL ')のデータ名 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	CBLDCRPC('CALL ')のデータ名 F に設定したサービスグループは，定義されていません。
00311	CBLDCRPC('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。
00312	CBLDCRPC('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	CBLDCRPC('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	CBLDCRPC('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。
	データ名 F に-1 を設定した場合に，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。

ステータスコード	意味
00315	CBLDCRPC('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	CBLDCRPC('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	CBLDCRPC('CALL ')に設定したサービスで，メモリが不足しました。
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 から DCRPC_MAX_MESSAGE_SIZE※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。
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。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。
00328	ドメイン修飾をしたサービスグループ名の，ドメイン名が間違っています。
00329	ドメイン修飾をしてサービスを要求しましたが，ドメイン代表スケジュールサービスのポート番号が見つかりません。
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	オンラインテスタ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	サービスを要求された SPP は，セキュリティ機能で保護されています。CBLDCRPC('CALL ')でサービスを要求した UAP には，SPP へのアクセス権限がありません。
00372	同時に起動できるトランザクションブランチの数を超えたため，トランザクションブランチを開始できません。
	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。
00378	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。このステータスコードは，ユーザサービス定義の rpc_extend_function オペランドに"00000001"を指定したクライアント UAP の場合にだけリターンされます。rpc_extend_function オペランドに"00000000"を指定，またはオペランドを省略した場合は，「00378」はリターンされないで，「00307」または「00314」がリターンされます。

注※

rpc_max_message_size オペランドを使用した場合，DCRPC_MAX_MESSAGE_SIZE の値（1 メガバイト）ではなく，rpc_max_message_size オペランドに指定した値になります。

(1) CBLDCRPC('POLLANYR')のデータ名 C について

非同期受信の監視時間は，応答が返るたびにリセットされます。そのため，受信する非同期応答を特定する（データ名 C に 1，または 17 を設定）場合は，データ名 F に設定した時間を過ぎても，応答を受信できる場合があります。また，データ名 F に設定した時間を過ぎても，「00307」でエラーリターンしない場合もあります。

(2) CBLDCRPC('POLLANYR')がエラーになるタイミング

サービスを要求された SPP が異常終了した場合，クライアント UAP でエラーが返るタイミングについて説明します。

サービスを実行する SPP が，処理が終わる前に異常終了すると，CBLDCRPC('POLLANYR')は「00307」でエラーリターンします。CBLDCRPC('POLLANYR')のデータ名 F に-1 を設定している場合は，「00314」でエラーリターンします。

CBLDCRPC('POLLANYR')の時間監視でエラーになる場合

次に示す場合には，CBLDCRPC('POLLANYR')のデータ名 F に設定した時間が経過したあとで，「00307」でエラーリターンします。

- SPP があるノードの OpenTP1 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，またはサーバ UAP の処理が完了してからクライアント UAP に結果が届く前に障害が起こった場合

(3) ステータスコード「00378」をリターンさせる指定

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たことを「00307」または「00314」以外のステータスコードで判別したい場合には，ユーザサービス定義の rpc_extend_function オペランドに"00000001"を指定します。この指定をすると，上記のエラー時に「00378」がリターンされるようになります。rpc_extend_function オペランドに"00000000"を指定，またはオペランドを省略した場合は，「00378」はリターンされないで，「00307」または「00314」がリターンされます。

(4) ステータスコードと同期点処理の関係

CBLDCRPC('POLLANYR')のステータスコードと同期点処理（コミット，ロールバック）の関係について説明します。ここで説明する内容は，サービス要求がトランザクション処理になる場合に該当します。トランザクションでないサービス要求（CBLDCRPC('CALL ')のデータ名 C に「32」を加算した場合も含む）には該当しません。

CBLDCRPC('POLLANYR')がエラーリターンしてもコミットとなる場合

サービスを要求されたサービスプログラムの異常終了や，ノードの障害，ネットワーク障害の場合でも，「00307」がリターンすることがあります。クライアント UAP がトランザクション処理でない場合は，

「00307」が返っても，要求したサービスがある SPP は正常終了していて，データベースへの更新などが実行されているときもあります。

ロールバック処理が必要なステータスコード

トランザクション処理から呼び出した CBLDCRPC('POLLANYR')がエラーリターンした場合，ステータスコードによっては，必ずトランザクションがロールバック（サーバ UAP が rollback_only 状態）になります。この場合，コミット，またはロールバックのどちらを実行しても，必ずロールバックになります。必ずロールバックになる CBLDCRPC('POLLANYR')のステータスコードを次に示します。

「00309」

「00311」

「00317」

「00319」

(5) CBLDCRPC('POLLANYR')で応答が受け取れない場合

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，CBLDCRPC('POLLANYR')で応答を受け取れません。

1. CBLDCRPC('DISCARDF')で，非同期応答の受信を拒否した場合
2. トランザクションの処理の場合，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 CBLDCRPC('POLLANYR')ですべて受け取ってください。

(6) CBLDCRPC('POLLANYR')を使うときの注意

1. CBLDCRPC('POLLANYR')を待ち時間 0 を設定（データ名 F に 0 を設定）して呼び出すと，マルチスレッド環境のスケジューリングの関係で，応答が到着していても受信できない場合があります。そのため，待ち時間 0 ですべての応答を受信する CBLDCRPC('POLLANYR')を呼び出す UAP は，無限ループになってしまうことがありますので，注意してください。
2. 記述子を特定しない CBLDCRPC('POLLANYR')がエラーリターンした場合，エラーになった応答の記述子を特定できません。CBLDCRPC('POLLANYR')がエラーリターンしたときの該当する記述子がわ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。

CBLDCRPC('SETSVPRI')

名称

サービス要求のスケジュールプライオリティの設定

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’SETSVPRI’.
   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

サービス要求のプライオリティを設定します。サービス要求単位でスケジュールプライオリティを制御する場合に呼び出します。ここで設定したプライオリティは，CBLDCRPC('SETSVPRI')を再び呼び出すまで更新されません。したがって，同じプライオリティでまとめてサービス要求をする場合は，CBLDCRPC('SETSVPRI')を 1 回だけ呼び出します。

CBLDCRPC('SETSVPRI')で指定したプライオリティは，直後に呼び出す CBLDCRPC('CALL ')で，スケジュールキューを経由してサーバに通知されます。

CBLDCRPC('SETSVPRI')を 1 回も呼び出さない処理の場合は，スケジュールサービスの省略時解釈値である 4 が，サービス要求のプライオリティとして指定されます。

UAP で値を設定するデータ領域

●データ名 A

サービス要求のスケジュールプライオリティの設定を示す要求コードを「VALUE 'SETSVPRI'」と設定します。

●データ名 C

サービス要求のスケジュールプライオリティを，0，または 1 から 8 の範囲で設定します。データ名 C の設定は省略できません。

最も高いプライオリティの値は 1 で，最も低いプライオリティの値は 8 です。

0 を設定した場合は，スケジュールサービスの省略時解釈となります。

上記以外の値を設定した場合は，CBLDCRPC('SETSVPRI')は無視されます。

OpenTP1 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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

注意事項

1. キュー受信型サーバでは，設定したサービス要求のプライオリティは，サーバ UAP のユーザサービス定義に，service_priority_control ＝ Y（プライオリティを制御する）を指定している場合だけ有効です。サービス要求する相手のサーバ UAP でプライオリティを制御していない場合は，CBLDCRPC('SETSVPRI')を呼び出しても無効になります。
2. 2 回目以降の連鎖 RPC をする CBLDCRPC('CALL ')と，連鎖 RPC を終了するために呼び出す CBLDCRPC('CALL ')に対して，サービス要求のプライオリティを設定しても無効になります。
3. CBLDCRPC('CALL ')は，CBLDCRPC('SETSVPRI')で変更した値をリセットしません。プライオリティをリセットする場合は，データ名 C に 0 を設定した CBLDCRPC('SETSVPRI')を呼び出してください。

CBLDCRPC('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 ’CBLDCRPC’ 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。CBLDCRPC('SETWATCH')で変更した値は，CBLDCRPC('CLOSE ')を呼び出すまで有効です。

サービス要求の応答待ち時間を CBLDCRPC('SETWATCH')を呼び出す前の値に戻すときは，事前に呼び出した CBLDCRPC('GETWATCH')でリターンされた値を，CBLDCRPC('SETWATCH')に設定して呼び出してください。

CBLDCRPC('SETWATCH')を呼び出しても，システム共通定義の watch_time オペランドの値に指定した値は変更されません。CBLDCRPC('SETWATCH')に設定した値は，あとから呼び出す CBLDCRPC('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

サービス要求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。1 から 65535 の範囲で設定します。無制限に待ち続ける場合は，0 を設定します。

OpenTP1 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。
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。
上記以外	プログラムの破壊などによる，予期しないエラーが起こりました。

CBLDCRPC('SVRETRY ')

名称

サービスプログラムのリトライ

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRPC’ 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 ’SVRETRY ’.
   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

実行中のサービスプログラムの処理をリトライします。リトライする場合は，サービスプログラムでCBLDCRPC('SVRETRY ')を呼び出したあとで，リトライするサービスプログラムをリターンしてください。リターンしたあと，同じプロセスで同じサービスプログラムが再起動されます。

応答型 RPC で呼ばれたサービスプログラムがリトライされた場合は，リトライ前のサービスプログラムが設定した値（応答を格納する領域と応答の長さ）は無効になります。

ユーザサービス定義の rpc_service_retry_count オペランドに指定した回数を超えた（rpc_service_retry_count オペランドに 0 を指定した場合も含む）あとで CBLDCRPC('SVRETRY ')を呼び出した場合，CBLDCRPC('SVRETRY ')は「00377」でエラーリターンします。このとき，サービスプログラムはリトライされません。応答型 RPC で呼ばれたサービスプログラムの場合は，応答を格納する領域の内容をクライアント UAP に返します。

UAP で値を設定するデータ領域

●データ名 A

サービスプログラムのリトライを示す要求コードを「VALUE 'SVRETRY△'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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。
00301	要求コード（データ名 A）が間違っています。
00377	ユーザサービス定義の rpc_service_retry_count オペランドに指定したサービスリトライ回数の最大値を超えて，CBLDCRPC('SVRETRY ')を呼び出しています。これ以上サービスプログラムをリトライできません。
00302	CBLDCRPC('SVRETRY ')を呼び出す条件が間違っています。要因としては次のことが考えられます。 • サービスプログラムの中で呼び出していません。 • グローバルトランザクションの範囲の中で呼び出しています。

注意事項

1. CBLDCRPC('SVRETRY ')を呼び出す場合は，次に示す条件を満たしてください。この条件を満たしていない場合，CBLDCRPC('SVRETRY ')はエラーリターンします。
   - サービスプログラムの中で CBLDCRPC('SVRETRY ')を呼び出していること。
   - 実行中のサービスプログラムが，グローバルトランザクションの範囲でないこと。
2. CBLDCRPC('SVRETRY ')を呼び出すサービスプログラムでは，クライアント UAP から渡されたデータは参照できますが，変更できません。入力データ領域の内容を変更した場合，システムの動作は保証しません。
3. CBLDCRPC('SVRETRY ')は，OpenTP1 独自のリモートプロシジャコール（CBLDCRPC('CALL ')）でサービスを要求されたサービスプログラムでだけ呼び出せます。それ以外のサービスプログラムの処理は，CBLDCRPC('SVRETRY ')でリトライできません。

CBLDCRSV('MAINLOOP')

名称

SPP のサービス開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRSV’ 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 ’MAINLOOP’.
   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

SPP にあるサービスプログラムへのスケジュールを受け付けます。

CBLDCRSV('MAINLOOP')は，SPP のメインプログラムから呼び出します。プロセスで 1 回だけ呼び出してください。

CBLDCRSV('MAINLOOP')を呼び出してから，OpenTP1 から SPP の終了が要求されるまで，CBLDCRSV('MAINLOOP')はリターンしません。OpenTP1 から SPP の終了が要求されるのは，次に示す場合です。

- 次に示す OpenTP1 の終了コマンドを実行したため，終了処理に入ったとき
  dcstop コマンド（正常終了）
  dcstop -n（強制正常終了）
  dcstop -a（計画停止 A）
  dcstop -b（計画停止 B）
- SPP のプロセスを，次に示すサーバの終了コマンドを実行したために，終了処理に入ったとき
  dcsvstop コマンド（正常終了）
- SPP のプロセスが，ユーザサービス定義の最大プロセス数の指定を超えたために，OpenTP1 からプロセスを終了させられるとき
- SPP を非常駐プロセスで実行しているときには，サービスの処理を終了したとき
- SPP をマルチサーバで負荷分散しているときは，該当するサービスグループへのサービス要求が減少したとき

UAP で値を設定するデータ領域

●データ名 A

SPP のサービス開始を示す要求コードを「VALUE 'MAINLOOP'」と設定します。

●データ名 C

0 を設定します。

OpenTP1 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	OpenTP1 から SPP の終了が要求されました。SPP はすぐにプロセスの終了処理をして，CBLDCRPC('CLOSE ')を呼び出してから，メインプログラムを STOP RUN で終了してください。
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。
00302	CBLDCRSV('MAINLOOP')を呼び出す前に，CBLDCRPC('OPEN ')を呼び出していません。
	CBLDCRSV('MAINLOOP')か CBLDCMCF('MAINLOOP')が，すでに呼び出されています。
00303	SPP のサービスを開始できませんでした。

注意事項

CBLDCRSV('MAINLOOP')は，OpenTP1 から終了要求があるとリターンします。ただし，次に示す場合には，CBLDCRSV('MAINLOOP')がリターンしないまま，プロセスが終了します。

- OpenTP1 を強制停止するコマンド（dcstop -f コマンド）や，ユーザサーバを強制停止するコマンド（dcsvstop -f コマンド）を実行したために，SPP が終了するとき
- UAP または OpenTP1 の不良が原因で，プロセスが異常終了したとき
- サービスプログラムで，プログラムの処理を抜ける命令文を使ったとき
- ハードウェアやオペレーティングシステム，または OpenTP1 そのものが障害になったとき

上記のような場合には，CBLDCRSV('MAINLOOP')が正常に終了したあとで終了処理をするようにコーディングしてあっても，その処理は実行されませんので注意してください。

リアルタイム統計情報サービス（CBLDCRTS）

リアルタイム統計情報サービスのプログラムについて説明します。リアルタイム統計情報サービスのCOBOL-UAP 作成用プログラムを次に示します。

- CBLDCRTS('RTSPUT ') － 任意区間でのリアルタイム統計情報の取得

CBLDCRTS('RTSPUT ')

名称

任意区間でのリアルタイム統計情報の取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCRTS’ 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 ’RTSPUT  ’.
   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 S9(9) COMP.
   02 データ名D  PIC X(1).
   02 FILLER     PIC X(3).
```

機能

UAP 内の任意の区間で，データ名 C に設定した項目の実行時間および実行回数を，リアルタイム統計情報として取得します。

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 'RTSPUT△△'」と設定します。

●データ名 C

取得するリアルタイム統計情報の項目 ID を設定します。

ID に使用できる値は，1000000～2147483647 です。

●データ名 D

実行する処理を設定します。

VALUE 'S'

データ名 C に設定した項目 ID の実行時間の計測を開始します。

VALUE'S'を設定して CBLDCRTS('RTSPUT ')を呼び出した時点では，リアルタイム統計情報を取得しません。

VALUE 'E'

データ名 C に設定した項目 ID の実行時間を取得して，計測を終了します。

VALUE '△'

データ名 C に設定した項目 ID の実行回数だけを取得します。実行時間は 0 秒となります。

●データ名 Z

0 を設定します。

OpenTP1 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。
07801	要求コード（データ名 A）が間違っています。
07802	データ名に設定した値が間違っています。
07803	CBLDCRPC('OPEN ')を呼び出していません。
	すでに実行時間の計測を開始している項目 ID をデータ名 C に設定して，データ名 D に'S'を設定した CBLDCRTS('RTSPUT ')を呼び出しました。
	実行時間の計測を開始していない項目 ID をデータ名 C に設定して，データ名 D に'E'を設定した CBLDCRTS('RTSPUT ')を呼び出しました。
07804	取得項目の数がリアルタイム統計情報サービス定義の rts_item_max オペランドの指定値を超えるため，情報を取得できません。
07805	サーバ単位の取得項目の数が，リアルタイム統計情報サービス定義の rts_item_max オペランドの指定値を超えるため，情報を取得できません。このステータスコードでリターンした場合，サービス単位またはサービス以外の処理の統計情報は取得しています。
07806	サービス単位またはサービス以外の処理での取得項目の数が，リアルタイム統計情報サービス定義の rts_item_max オペランドの指定値を超えるため，情報を取得できません。このステータスコードでリターンした場合，サーバ単位の統計情報は取得しています。
07807	プロセスメモリが不足したため，処理を実行できません。
07808	リアルタイム統計情報サービスが開始していません。
07809	CBLDCRTS('RTSPUT ')の呼び出し元が，サーバ単位およびサービス単位のどちらもリアルタイム統計情報の取得対象に登録されていません。
07810	UAP が，現在稼働しているリアルタイム統計情報サービスでは稼働できないバージョンのライブラリと結合しています。

注意事項

1. CBLDCRTS('RTSPUT ')では，システム全体のリアルタイム統計情報は取得できません。
2. マルチサーバを使用しているUAPでは，同じ呼び出し元サービスおよび同じデータ名を設定したCBLDCRTS('RTSPUT ')を複数プロセスから同時に呼び出した場合，プロセスによっては統計情報が取得されないことがあります。これは，統計情報の取得処理では排他制御がされないため，書き込み処理が同時に行われることが要因です。
3. XATMIインタフェースを使用しているUAPでは，サービス単位のリアルタイム統計情報は取得できません。すべてサービス以外の処理の統計情報として取得されます。
4. CBLDCRTS('RTSPUT ')は，UAPトレースを取得しません。
5. データ名Dに'S'を指定したCBLDCRTS('RTSPUT ')が，ステータスコード「07808」または「07809」でリターンしたあとで，同じデータ名Cでデータ名Dに'E'を指定したCBLDCRTS('RTSPUT ')を呼び出すまでに，リアルタイム統計情報サービスを開始して呼び出し元のUAPを取得対象に追加した場合，CBLDCRTS('RTSPUT ')はステータスコード「07803」でリターンします。

TAM ファイルサービス（CBLDCTAM）

TAM ファイルサービスのプログラムについて説明します。TAM ファイルサービスの COBOL-UAP 作成用プログラムを次に示します。

- CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR') - TAM テーブルのレコードの削除
- CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU') - TAM テーブルからレコードの入力
- CBLDCTAM('GST ') - TAM テーブルの状態の取得
- CBLDCTAM('INFO') - TAM テーブルの情報の取得
- CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')- TAM テーブルのレコードの更新/追加

TAM ファイルサービスの COBOL-UAP 作成用プログラム（CBLDCTAM）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，TAM ファイルサービスの COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。TAM ファイルサービス（CBLDCTAM）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCTAM.cbl に格納してあります。

CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR')

名称

TAM テーブルのレコードの削除

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTAM’ 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(4).
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(32).
   02 FILLER    PIC X(68).
   02 データ名D  PIC S9(4)  COMP.
   02 FILLER    PIC X(2).
   02 データ名J  PIC S9(9)  COMP.
   02 FILLER    PIC X(392).
01 一意名2.
   02 データ名E  PIC X(4).
   02 FILLER    PIC X(3).
   02 データ名I  PIC X(1).
01 一意名3.
   02 データ名F  PIC X(m).
01 一意名4.
   02 データ名G  PIC X(n).
```

機能

キー値に示すレコードを，TAM テーブルから削除します。削除するレコードをバッファに退避することもできます。ただし，このプログラムがエラーリターンした場合には，バッファの内容は保証できません。

レコードを削除するときは，更新排他でテーブル排他を確保します。

TAM テーブルのレコード削除がエラーリターンした場合は，設定した資源はすべて解放して，プログラムを呼び出す前の状態に戻ります。ただし，プログラムを呼び出す前に参照排他で確保されていた TAM テーブルを削除しようとした場合は，更新排他となり，参照排他には戻りません。

UAP で値を設定するデータ領域

●データ名 A

TAM の識別子を設定します。ただし，TAM サービスではこの値を参照しません。

●データ名C

削除するレコードの TAM テーブル名を 32 文字以内で設定します。32 文字未満の場合は，後ろを空白で埋めます。

●データ名D

削除するレコードを退避する場合に，1～32767 バイト以下のバッファ長を設定します。バッファ長は，レコード長以上にします。データ名 E に「VALUE 'ERSR'」を設定した場合だけ，この設定は有効になります。

●データ名J

削除するレコードを退避する場合に，32768 バイト以上のバッファ長を設定します。バッファ長は，レコード長以上にします。データ名 E に「VALUE 'ZRSR'」を設定した場合だけ，この設定は有効になります。

●データ名E

要求コードを設定します。

VALUE 'ERS△'，VALUE 'ZRS△'…削除するレコードを退避しない削除。

VALUE 'ERSR'，VALUE 'ZRSR'…削除するレコードを退避する削除。

レコード長が 32767 バイト以下の TAM テーブルの場合は「VALUE 'ERS△'」または「VALUE 'ERSR'」を使用します。

レコード長が 32768 バイト以上の TAM テーブルの場合は「VALUE 'ZRS△'」または「VALUE 'ZRSR'」を使用します。

●データ名F

キー値を設定します。キー値は，削除するレコードのキー領域の長さで設定します。

●データ名G

削除するレコードを退避する場合に，退避するバッファを設定します。データ名 E に「VALUE 'ERS△'」または「VALUE 'ZRS△'」（レコードを退避しない）を設定した場合，この設定は無効です。

●データ名I

排他解除待ちの種別を，次に示す値で設定します。データ名 I に設定した値は，TAM サービス定義の tam_cbl_level に '2' を指定した場合に有効になります。

VALUE 'W' … 排他の解除を待ちます。

VALUE 'N' … 排他の解除を待たないで，エラーリターンします。

TAM サービス定義の tam_cbl_level に '0' または '1' を指定した場合は，データ名 I を指定する必要はありません。

OpenTP1 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

データ名 E に「VALUE 'ERSR'」または「VALUE 'ZRSR'」を設定したレコードの削除が正常に終了した場合，削除したレコードが退避されます。

ステータスコード

ステータスコード	意味
00000	レコードを正常に削除しました。
01701	データ名 C に設定した TAM テーブル名が間違っています。
01702	データ名 F に設定したキー値が間違っています。
01704	データ名 G に設定した値が間違っています。
01705	データ名 D，またはデータ名 J に設定したバッファ長が短過ぎます。
01708	データ名 E，またはデータ名 I に設定した値が間違っています。
01709	データ名 C に設定したテーブルは TAM テーブルではありません。
01710	TAM テーブルが定義されていません。
01720	TAM サービスが終了中です。
01721	TAM テーブルへアクセスする順序が間違っています。
	UAP にリンケージしているトランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。または，UAP にトランザクション制御用オブジェクトファイルをリンケージしていません。
	CBLDCTAM を呼び出した UAP のユーザサービス定義に，トランザクション属性なし（atomic_update=N）を指定しています。
01723	TAM テーブルが削除されています。
01724	TAM テーブルがロードされていません。
01727	TAM テーブルが論理閉塞状態です。
01728	TAM テーブルが障害閉塞状態です。
01730	TAM サービス定義で指定した TAM テーブルのアクセス形態では実行できません。
01731	指定されたレコードは存在しません。
01736	排他エラーが起こりました。TAM サービス定義の tam_cbl_level オペランドに 0 を指定した場合，または tam_cbl_level オペランドに 2 を指定してデータ名 I に W を設定した場合は，ロックサービス定義で指定した待ち時間のタイムアウトのため，資源を確保できませんでした。
01737	デッドロックが起こりました。

ステータスコード	意味
01760	UAP が，現在稼働している TAM テーブルでは動作できないバージョンの TAM ライブラリと結合されています。
01761	UAP が，現在稼働している OpenTP1 ファイルサービスでは動作できないバージョンの TAM ライブラリと結合されています。
01762	UAP が，現在稼働している TAM サービスでは動作できないバージョンの TAM ライブラリと結合されています。
01764	レコードが破壊されています。
01765	TAM サービスで管理できるトランザクション数を超えました。
01766	キャラクタ型スペシャルファイルのオープン数の制限値を超えました。
01767	スペシャルファイルに対するアクセス権がありません。
01768	TAM ファイルに対するアクセス権がありません。
01769	メモリが不足しました。
01770	入出力エラーが起こりました。
01771	トランザクションサービスでエラーが起こりました。
01772	削除しようとした TAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01773	アクセスしようとした TAM ファイルは，セキュリティ機能で保護されています。TAM テーブルからレコードを削除しようとした UAP には，アクセス権限がありません。

注意事項

ハッシュ形式の TAM テーブルに格納されているレコードを全件削除する場合，レコードの先頭検索と，見つかったレコードの削除を繰り返すと CPU が高負荷になることがあります。したがって，レコードを全件削除する場合，次の手順で行ってください。

1. 先頭検索して見つかったレコードのキー値を変数 1 に退避する
2. 変数 1 のキー値を使い，NEXT 検索する
3. 手順 2.で見つかったレコードのキー値を変数 2 に退避する
4. 退避しておいた変数 1 のキー値のレコードを削除する
5. 変数 2 のキー値を変数 1 に退避する
6. 手順 2.がエラーになるまで手順 2.から手順 5.を繰り返す（NEXT 検索する）
7. 手順 2.がエラーになったあとで，最後に変数 1 に退避したキー値のレコードを削除する

CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU')

名称

TAM テーブルからレコードの入力

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTAM’ 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(4).
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(32).
   02 FILLER    PIC X(68).
   02 データ名D  PIC S9(4) COMP.
   02 FILLER    PIC X(2).
   02 データ名J  PIC S9(9)  COMP.
   02 FILLER    PIC X(392).
01 一意名2.
   02 データ名E  PIC X(4).
   02 データ名F  PIC X(1).
   02 FILLER    PIC X(2).
   02 データ名I  PIC X(1).
01 一意名3.
   02 データ名G  PIC X(m).
01 一意名4.
   02 データ名H  PIC X(n).
```

機能

データ名 E に設定した検索種別に従って，TAM テーブル上のレコードを，参照または更新の目的で入力します。検索種別とインデクス種別の関係を次の表に示します。

表 2-3　検索種別とインデクス種別の関係

検索種別	検索処理の概要	
	インデクス種別：ハッシュ	インデクス種別：ツリー
'キー値＝'検索	設定したキー値を持つレコードを検索します。 設定したキー値を持つレコードがない場合はエラーリターンします。	設定したキー値を持つレコードを検索します。 設定したキー値を持つレコードがない場合はエラーリターンします。
'キー値＜＝'検索	エラーリターンします。	設定したキー値以上のキー値を持つレコードを検索します。

検索種別	検索処理の概要	
	インデクス種別：ハッシュ	インデクス種別：ツリー
'キー値＜'検索	エラーリターンします。	設定したキー値より大きいキー値を持つレコードを検索します。
'キー値＞＝'検索	エラーリターンします。	設定したキー値以下のキー値を持つレコードを検索します。
'キー値＞'検索	エラーリターンします。	設定したキー値より小さいキー値を持つレコードを検索します。
先頭検索※	キー値に対応してハッシングをした先頭レコードを検索します。データ名 G に設定したキー値は無視します。	エラーリターンします。
NEXT 検索※	キー値に対応してハッシングをした，次のレコードを検索します。	エラーリターンします。

注※

インデクス種別がハッシュで，TAM テーブルファイルの初期作成時に，データ部にキー値を付けている（tamcre コマンドに-s オプションを設定していない）場合，先頭検索と NEXT 検索を使って，TAM テーブル上の全レコードを検索できます。

参照目的の入力で排他を掛ける場合，参照排他でテーブル排他とレコード排他を確保します。レコード排他でオープンした TAM テーブルを更新目的で入力する場合は，参照排他でテーブル排他を確保し，更新排他でレコード排他を確保します。

ただし，TAM サービス定義の「アクセス時のテーブル排他モード」に，「テーブル排他なしモード」を指定している場合は，アクセス形態が「参照型」「追加・削除できない更新型」のテーブルには，テーブル排他を確保しません。

TAM テーブルのレコード入力がエラーリターンした場合は，このプログラムで設定した資源はすべて解放して，プログラムを呼び出す前の状態に戻ります。ただし，プログラムを呼び出す前に参照排他で確保されていたレコードを更新目的で入力した場合は，更新排他となり参照排他には戻りません。また，エラーリターンした場合には，バッファの内容は保証できません。

UAP で値を設定するデータ領域

●データ名 A

TAM の識別子を設定します。ただし，TAM サービスではこの値を参照しません。

●データ名 C

入力するレコードの TAM テーブル名を 32 文字以内で設定します。32 文字未満の場合は，後ろを空白で埋めます。

●データ名 D

入力するレコードが，1～32767 バイト以下のときのバッファ長を設定します。バッファ長は，レコード長以上にします。データ名 E に「VALUE 'Fxxx'」を設定した場合だけ，この設定は有効になります。

●データ名 J

入力するレコードが 32768 バイト以上のときのバッファ長を設定します。バッファ長は，レコード長以上にします。データ名 E に「VALUE 'Vxxx'」を設定した場合だけ，この設定は有効になります。

●データ名 E

要求コードを設定します。

VALUE 'FCHR',VALUE 'VCHR'…参照目的で'キー値＝'を検索する（ハッシュ，ツリー)。

VALUE 'FGER',VALUE 'VGER'…参照目的で'キー値＜＝'を検索する（ツリー)。

VALUE 'FGTR',VALUE 'VGTR'…参照目的で'キー値＜'を検索する（ツリー)。

VALUE 'FLER',VALUE 'VLER'…参照目的で'キー値＞＝'を検索する（ツリー)。

VALUE 'FLTR',VALUE 'VLTR'…参照目的で'キー値＞'を検索する（ツリー)。

VALUE 'FTPR',VALUE 'VTPR'…参照目的で設定したキー値の，先頭から検索する（ハッシュ)。

VALUE 'FNXR',VALUE 'VNXR'…参照目的で設定したキー値の，次のレコードから検索する（ハッシュ，ツリー)。

VALUE 'FCHU',VALUE 'VCHU'…更新目的で'キー値＝'を検索する（ハッシュ，ツリー)。

VALUE 'FGEU',VALUE 'VGEU'…更新目的で'キー値＜＝'を検索する（ツリー)。

VALUE 'FGTU',VALUE 'VGTU'…更新目的で'キー値＜'を検索する（ツリー)。

VALUE 'FLEU',VALUE 'VLEU'…更新目的で'キー値＞＝'を検索する（ツリー)。

VALUE 'FLTU',VALUE 'VLTU'…更新目的で'キー値＞'を検索する（ツリー)。

VALUE 'FTPU',VALUE 'VTPU'…更新目的で設定したキー値の，先頭から検索する（ハッシュ)。

VALUE 'FNXU',VALUE 'VNXU'…更新目的で設定したキー値の，次のレコードから 検索する（ハッシュ，ツリー)。

レコード長が 32767 バイト以下の TAM テーブルの場合は「VALUE 'Fxxx'」を使用します。

レコード長が 32768 バイト以上の TAM テーブルの場合は「VALUE 'Vxxx'」を使用します。

●データ名 F

参照目的の入力の場合の，排他の要否種別を設定します。

VALUE '△' … 排他する（同期点で排他を解除します)。

VALUE 'D' … 排他する（この文の終了時に排他を解除します)。

VALUE 'N' … 排他しない。

排他しないで CBLDCTAM('FxxR'/'VxxR')を呼び出した場合，CBLDCTAM('FxxR'/'VxxR')処理中にほかの UAP から該当する TAM レコードが更新されることがあります。

この場合，CBLDCTAM('FxxR'/'VxxR')で入力するレコードの内容は，ほかの UAP での更新処理の状態（コミット処理が完了するタイミング）に依存します。したがって，UAP 間でデータの整合性をとるためには，必ず VALUE 'D'または VALUE '△'を指定してください。

VALUE 'N'は，参照型の TAM テーブルの場合や，同一の TAM レコードに対して業務処理全体として参照と更新が競合する可能性がない場合に使用します。

●データ名 G

キー値を設定します。キー値は，検索するレコードのキー領域の長さで設定します。

●データ名 H

レコードを入力するデータ領域（バッファ）を設定します。

●データ名 I

排他解除待ちの種別を，次に示す値で設定します。データ名 I に設定した値は，TAM サービス定義の tam_cbl_level に '2' を指定した場合に有効になります。

VALUE 'W' … 排他の解除を待ちます。

VALUE 'N' … 排他の解除を待たないで，エラーリターンします。

TAM サービス定義の tam_cbl_level に '0' または '1' を指定した場合は，データ名 I を指定する必要はありません。

OpenTP1 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 H

入力したレコードが返されます。

ステータスコード

ステータスコード	意味
00000	TAM テーブルは正常に入力しました。
01701	データ名 C に設定した TAM テーブル名が間違っています。
01702	データ名 G に設定したキー値が間違っています。

ステータスコード	意味
01704	データ名 H に設定した値が間違っています。
01705	データ名 D，またはデータ名 J に設定したバッファ長が短過ぎます。
01708	データ名 E，データ名 F，またはデータ名 I に設定した値が間違っています。
01709	データ名 C に設定したテーブルは TAM テーブルではありません。
01710	TAM テーブルが定義されていません。
01720	TAM サービスが終了中です。
01721	TAM テーブルへアクセスする順序が間違っています。
	UAP にリンケージしているトランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。または，UAP にトランザクション制御用オブジェクトファイルをリンケージしていません。
	CBLDCTAM を呼び出した UAP のユーザサービス定義に，トランザクション属性なし（atomic_update=N）を指定しています。
01723	TAM テーブルが削除されています。
01724	TAM テーブルがロードされていません。
01727	TAM テーブルが論理閉塞状態です。
01728	TAM テーブルが障害閉塞状態です。
01729	TAM テーブルファイルの初期作成で設定した TAM テーブルのインデクス種別では実行できません。
01730	TAM サービス定義で指定した TAM テーブルのアクセス形態では実行できません。
01731	データ名 E に設定した検索条件を満たすレコードがありません。
01736	排他エラーが起こりました。TAM サービス定義の tam_cbl_level オペランドに 0 を指定した場合，または tam_cbl_level オペランドに 2 を指定してデータ名 I に W を設定した場合は，ロックサービス定義で指定した待ち時間のタイムアウトのため，資源を確保できませんでした。
01737	デッドロックが起こりました。
01760	UAP が，現在稼働している TAM テーブルでは動作できないバージョンの TAM ライブラリと結合されています。
01761	UAP が，現在稼働している OpenTP1 ファイルサービスでは動作できないバージョンの TAM ライブラリと結合されています。
01762	UAP が，現在稼働している TAM サービスでは動作できないバージョンの TAM ライブラリと結合されています。
01764	レコードが破壊されています。
01765	TAM サービスで管理できるトランザクション数を超えました。
01766	キャラクタ型スペシャルファイルのオープン数の制限値を超えました。
01767	スペシャルファイルに対するアクセス権がありません。

ステータスコード	意味
01768	TAM ファイルに対するアクセス権がありません。
01769	メモリが不足しました。
01770	入出力エラーが起こりました。
01771	トランザクションサービスでエラーが起こりました。
01772	オープンしようとした TAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01773	アクセスしようとした TAM ファイルは，セキュリティ機能で保護されています。TAM テーブルからレコードを入力しようとした UAP には，アクセス権限がありません。

CBLDCTAM('GST ')

名称

TAM テーブルの状態の取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTAM’ 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(4).
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(35).
   02 データ名C  PIC X(4).
   02 FILLER    PIC X(64).
   02 データ名D  PIC S9(4) COMP.
   02 FILLER    PIC X(398).
01 一意名2.
   02 データ名E  PIC X(4).
   02 FILLER    PIC X(4).
01 一意名3.
   02 一意名4.
      03 データ名F  PIC X(32).
      03 データ名G  PIC X(2).
      03 FILLER    PIC X(6).
        :
        :
   02 一意名n.
      03 データ名F  PIC X(32).
      03 データ名G  PIC X(2).
      03 FILLER    PIC X(6).
```

機能

TAM テーブルの状態を取得します。取得する TAM テーブルの状態を次に示します。

- オープン状態
- クローズ状態
- 論理閉塞状態
- 障害閉塞状態

TAM テーブルの状態は，トランザクション内でもトランザクション外でも取得できます。

ほかのプロセスで TAM テーブルにアクセスしている場合でも，TAM テーブルはオープン状態としてリターンします。

UAP で値を設定するデータ領域

●データ名 A

TAM の識別子を設定します。ただし，TAM サービスではこの値を参照しません。

●データ名 D

TAM テーブルの情報領域長（一意名 3 の長さ）を設定します。

●データ名 E

TAM テーブルの情報取得を示す要求コードを「VALUE 'GST△'」と設定します。

●データ名 F

状態を取得するテーブルの TAM テーブル名を，32 文字以内で設定します。32 文字未満の場合は，後ろを空白で埋めます。

OpenTP1 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

設定された TAM テーブルの全体の状態が，4 バイトの領域に返されます。1 バイトごとの状態の意味を次の表に示します。

表 2-4　TAM テーブルの全体の状態

バイト位置	状態	状態の意味
0	L	論理閉塞状態の TAM テーブルがあります。
	空白	論理閉塞状態の TAM テーブルがありません。
1	B	障害閉塞状態の TAM テーブルがあります。
	空白	障害閉塞状態の TAM テーブルがありません。
2	O	オープン状態の TAM テーブルがあります。
	空白	オープン状態の TAM テーブルがありません。
3	C	クローズ状態の TAM テーブルがあります。
	空白	クローズ状態の TAM テーブルがありません。

●データ名 G

設定された TAM テーブルの状態が返されます。

VALUE 'RO' …オープン状態

VALUE 'RC' …クローズ状態

VALUE 'HL' …論理閉塞状態

VALUE 'HB' …障害閉塞状態

ステータスコード

ステータスコード	意味
00000	TAM テーブルの情報を正常に取得しました。
01701	データ名 F に設定した TAM テーブル名が間違っています。
01705	データ名 D に設定した値が間違っています。
01708	データ名 E に設定した値が間違っています。
01710	TAM テーブルが定義されていません。
01720	TAM サービスが終了中です。
01721	TAM テーブルへアクセスする順序が間違っています。
	UAP にリンケージしているトランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。または，UAP にトランザクション制御用オブジェクトファイルをリンケージしていません。
	CBLDCTAM を呼び出した UAP のユーザサービス定義に，トランザクション属性なし（atomic_update=N）を指定しています。
01762	UAP が，現在稼働している TAM サービスでは動作できないバージョンの TAM ライブラリと結合されています。
01772	アクセスしようとした TAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01773	アクセスしようとした TAM ファイルは，セキュリティ機能で保護されています。TAM テーブルの状態を取得しようとした UAP には，アクセス権限がありません。

CBLDCTAM('INFO')

名称

TAM テーブルの情報の取得

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTAM’ 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(4).
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(103).
   02 データ名C  PIC S9(4) COMP.
   02 FILLER    PIC X(398).
01 一意名2.
   02 データ名D  PIC X(4).
   02 FILLER    PIC X(4).
01 一意名3.
   02 一意名4.
      03 データ名E  PIC X(32).
      03 データ名F  PIC X(2).
      03 データ名G  PIC X(64).
      03 FILLER    PIC X(2).
      03 データ名H  PIC S9(9) COMP.
      03 データ名I  PIC S9(9) COMP.
      03 データ名J  PIC X(1).
      03 データ名K  PIC X(1).
      03 データ名L  PIC X(1).
      03 FILLER    PIC X(1).
      03 データ名M  PIC S9(9) COMP.
      03 データ名N  PIC S9(9) COMP.
      03 データ名O  PIC S9(9) COMP.
      03 データ名P  PIC X(1).
      03 FILLER    PIC X(3).
            :
            :
            :
   02 一意名n.
      03 データ名E  PIC X(32).
      03 データ名F  PIC X(2).
            :
      03 データ名P  PIC X(1).
      03 FILLER    PIC X(3).
```

機能

TAM テーブルの情報をリターンします。リターンする値を次に示します。

- TAM ファイル名
- TAM テーブルの状態
- 使用中のレコード数
- 最大レコード数
- インデクス種別
- アクセス形態
- ローディング契機
- TAM レコード長
- キー長
- キー開始位置
- セキュリティ属性

TAM テーブルの情報は，トランザクション内でもトランザクション外でも取得できます。

UAP で値を設定するデータ領域

●データ名 A

TAM の識別子を設定します。ただし，TAM サービスではこの値を参照しません。

●データ名 C

TAM テーブルの情報領域長（一意名 3 の長さ）を設定します。

●データ名 D

TAM テーブルの情報を取得することを示す要求コードを「VALUE 'INFO'」と設定します。

●データ名 E

情報を取得するテーブルの TAM テーブル名を，32 文字以内で設定します。32 文字未満の場合は，後ろを空白で埋めます。

OpenTP1 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

TAM テーブルの状態が，次に示す値のどれかで設定されます。

VALUE 'RO' … オープン状態であることを示します。

VALUE 'RC' … クローズ状態であることを示します。

VALUE 'HL' … 論理閉塞状態であることを示します。

VALUE 'HB' … 障害閉塞状態であることを示します。

●データ名 G

TAM テーブルの TAM ファイル名が返されます。

●データ名 H

TAM テーブルで現在使っているレコードの数が設定されます。ただし，CBLDCTAM('INFO')を呼び出したあとでレコードの追加や削除があった場合は，値を保証しません。

●データ名 I

TAM テーブルの最大レコード数が設定されます。

●データ名 J

TAM テーブルのインデクス種別が，次に示す値のどれかで設定されます。

VALUE 'H' … ハッシュ形式であることを示します。

VALUE 'T' … ツリー形式であることを示します。

●データ名 K

TAM テーブルのアクセス形態が，次に示す値のどれかで設定されます。

VALUE 'R' … 参照型であることを示します。

VALUE 'W' … 追加・削除できない更新型であることを示します。

VALUE 'A' … 追加・削除できる更新型であることを示します。

VALUE 'L' … テーブル排他を確保しない，追加・削除できる更新型であることを示します。

●データ名 L

TAM テーブルのローディング契機が，次に示す値のどれかで設定されます。

VALUE 'S' … TAM サービスの開始時であることを示します。

VALUE 'L' … dc_tam_open 関数で TAM テーブルをオープンしたときであることを示します。

VALUE 'C' … tamload コマンドを実行したときであることを示します。

●データ名 M

TAM テーブルのレコード長が設定されます。

●データ名 N

TAM テーブルのキー長が設定されます。

●データ名 O

TAM テーブルのデータ内のキー開始位置が設定されます。

●データ名 P

TAM サービス定義で指定した TAM テーブルのセキュリティ属性が，次に示す値のどれかで設定されます。

VALUE 'N' … セキュリティの指定がないことを示します。

VALUE 'S' … セキュリティの指定が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	TAM テーブルの情報を正常に取得しました。
01701	データ名 E に設定した値が間違っています。
01705	データ名 C に設定した値が間違っています。
01708	データ名 D（要求コード）に設定した値が間違っています。
01709	データ名 E に設定した名称は，TAM テーブルではありません。
01710	TAM テーブルが定義されていません。
01720	TAM サービスが終了中です。
01721	TAM テーブルへアクセスする順序が間違っています。
	UAP にリンケージしているトランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。または，UAP にトランザクション制御用オブジェクトファイルをリンケージしていません。
	CBLDCTAM を呼び出した UAP のユーザサービス定義に，トランザクション属性なし（atomic_update=N）を指定しています。
01760	UAP が，現在稼働している TAM テーブルでは動作できないバージョンの TAM ライブラリと結合されています。
01762	UAP が，現在稼働している TAM サービスでは動作できないバージョンの TAM ライブラリと結合されています。
01766	キャラクタ型スペシャルファイルのオープン数の制限値を超えました。
01767	スペシャルファイルに対するアクセス権がありません。
01769	メモリが不足しました。
01770	入出力エラーが起こりました。

ステータスコード	意味
01772	情報を取得しようとした TAM テーブルは，セキュリティ機能で保護されています。該当する TAM テーブルに対する ACL がありません。
01773	情報を取得しようとした TAM テーブルは，セキュリティ機能で保護されています。CBLDCTAM('INFO')を呼び出した UAP には，アクセス権限がありません。

CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ')

名称

TAM テーブルのレコードの更新/追加

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTAM’ 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(4).
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(32).
   02 FILLER    PIC X(68).
   02 データ名D  PIC S9(4) COMP.
   02 FILLER    PIC X(2).
   02 データ名J  PIC S9(9)  COMP.
   02 FILLER    PIC X(392).
01 一意名2.
   02 データ名E  PIC X(4).
   02 FILLER    PIC X(3).
   02 データ名I  PIC X(1).
01 一意名3.
   02 データ名F  PIC X(m).
01 一意名4.   02 データ名G  PIC X(n).
```

機能

キー値に示すレコードを，TAM テーブル上に更新または追加します。

レコード排他で TAM テーブルがオープンしている場合，次のように更新排他でレコード排他を確保します。

- アクセス種別が「更新」の場合（データ名 E に VALUE 'MFY△'，または VALUE 'WFY△'を設定）
  参照排他でテーブル排他を確保して，更新排他でレコード排他を確保します。
  ただし，TAM サービス定義の「アクセス時のテーブル排他モード」に，「テーブル排他なしモード」を指定している場合は，アクセス形態が「参照型」「追加・削除できない更新型」のテーブルには，テーブル排他を確保しません。
- アクセス種別が「更新または追加」，「追加」の場合（データ名 E に VALUE 'MFYS'， VALUE 'WFYS'， VALUE 'STR△'，または VALUE 'YTR△'を設定）
  更新排他でテーブル排他を確保します。

TAM テーブルのレコード更新/追加がエラーリターンした場合，このプログラムで確保した資源はすべて解放して，プログラムを呼び出す前の状態に戻ります。ただし，このプログラムを呼び出す前に，参照排他で確保されていた TAM テーブルを更新または追加した場合は，更新排他となり参照排他には戻りません。

更新または追加するデータ内のキー値の格納位置，およびキー領域長は，TAM テーブルファイルの初期作成時の tamcre コマンドに設定した値です。

TAM テーブルファイルの初期作成時，データ部にキー値を付けている（tamcre コマンドに-s オプションを指定していない）場合は，データ部にキー値があります。そのため，TAM テーブルのレコード更新/追加に設定したキー値が，更新または追加するデータ内になければ，エラーリターンします。また，データ部にキー値を付けていない（tamcre コマンドに-s オプションを指定）場合は，データ部にキー値はありません。この場合は，更新または追加するデータの内容をチェックしません。

UAP で値を設定するデータ領域

●データ名 A

TAM の識別子を設定します。ただし，TAM サービスではこの値を参照しません。

●データ名 C

更新または追加するレコードの TAM テーブル名を 32 文字以内で設定します。32 文字未満の場合は，後ろを空白で埋めます。

●データ名 D

更新または追加するデータが，1～32767 バイト以下のデータ長を設定します。データ長は，レコード長以上にします。データ名 E に「VALUE 'MFY△'」，「VALUE 'MFYS'」または「VALUE 'STR△'」を設定した場合だけ，この設定は有効になります。

●データ名 J

更新または追加するデータが 32768 バイト以上のときのデータ長を設定します。データ長は，レコード長以上にします。データ名 E に「VALUE 'WFY△'」，「VALUE 'WFYS'」または「VALUE 'YTR△'」を設定した場合だけ，この設定は有効になります。

●データ名 E

要求コードを設定します。

VALUE 'MFY△'，VALUE 'WFY△'…レコードの更新だけ。

VALUE 'MFYS'，VALUE 'WFYS'…レコードの更新または追加。

VALUE 'STR△'，VALUE 'YTR△'…レコードの追加だけ。

レコード長が 32767 バイト以下の TAM テーブルの場合は「VALUE 'MFY△'」，「VALUE 'MFYS'」または「VALUE 'STR△'」を使用します。

レコード長が 32768 バイト以上の TAM テーブルの場合は「VALUE 'WFY△'」,「VALUE 'WFYS'」または「VALUE 'YTR△'」を使用します。

●データ名 F

キー値を設定します。キー値は，更新または追加するレコードのキー領域の長さで設定します。

●データ名 G

更新または追加するデータ領域（バッファ）を設定します。

●データ名 I

排他解除待ちの種別を，次に示す値で設定します。データ名 I に設定した値は，TAM サービス定義の tam_cbl_level に '2' を指定した場合に有効になります。

VALUE 'W' … 排他の解除を待ちます。

VALUE 'N' … 排他の解除を待たないで，エラーリターンします。

TAM サービス定義の tam_cbl_level に '0' または '1' を指定した場合は，データ名 I を指定する必要はありません。

OpenTP1 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	TAM テーブルのレコードを正常に更新または追加しました。
01701	データ名 C に設定した TAM テーブル名が間違っています。
01702	データ名 F に設定したキー値が間違っています。
01706	データ名 G に設定したデータ領域内のキー値が，データ名 F で設定したキー値と一致しません。
01707	データ名 D，またはデータ名 J に設定したデータ長が短過ぎます。
01708	データ名 E，またはデータ名 I に設定した値が間違っています。
01709	データ名 C に設定したテーブルは TAM テーブルではありません。
01710	TAM テーブルが定義されていません。
01720	TAM サービスが終了中です。
01721	TAM テーブルへアクセスする順序が間違っています。

ステータスコード	意味
01721	UAP にリンケージしているトランザクション制御用オブジェクトファイルのリソースマネジャ登録が間違っています。または，UAP にトランザクション制御用オブジェクトファイルをリンケージしていません。
	CBLDCTAM を呼び出した UAP のユーザサービス定義に，トランザクション属性なし（atomic_update=N）を指定しています。
01723	TAM テーブルが削除されています。
01724	TAM テーブルがロードされていません。
01727	TAM テーブルが論理閉塞状態です。
01728	TAM テーブルが障害閉塞状態です。
01730	TAM サービス定義で設定した TAM テーブルのアクセス形態では実行できません。
01731	指定されたレコードは存在しません。
01735	データ名 F に設定したキー値が TAM テーブルに存在するので，レコードの追加はできません。
01736	排他エラーが起こりました。TAM サービス定義の tam_cbl_level オペランドに 0 を指定した場合，または tam_cbl_level オペランドに 2 を指定してデータ名 I に W を設定した場合は，ロックサービス定義で指定した待ち時間のタイムアウトのため，資源を確保できませんでした。
01737	デッドロックが起こりました。
01760	UAP が，現在稼働している TAM テーブルでは動作できないバージョンの TAM ライブラリと結合されています。
01761	UAP が，現在稼働している OpenTP1 ファイルサービスでは動作できないバージョンの TAM ライブラリと結合されています。
01762	UAP が，現在稼働している TAM サービスでは動作できないバージョンの TAM ライブラリと結合されています。
01763	TAM テーブルに空きレコードがありません。
01764	レコードが破壊されています。
01765	TAM サービスで管理できるトランザクション数を超えました。
01766	キャラクタ型スペシャルファイルのオープン数の制限値を超えました。
01767	スペシャルファイルに対するアクセス権がありません。
01768	TAM ファイルに対するアクセス権がありません。
01769	メモリが不足しました。
01770	入出力エラーが起こりました。
01771	トランザクションサービスでエラーが起こりました。
01772	オープンしようとした TAM ファイルは，セキュリティ機能で保護されています。該当するファイルに対する ACL がありません。
01773	アクセスしようとした TAM ファイルは，セキュリティ機能で保護されています。TAM テーブルのレコードを更新/追加しようとした UAP には，アクセス権限がありません。

トランザクション制御（CBLDCTRN）

OpenTP1 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。

- CBLDCTRN('BEGIN ') － トランザクションの開始
- CBLDCTRN('C-COMMIT') － 連鎖モードのコミット
- CBLDCTRN('C-ROLL ') － 連鎖モードのロールバック
- CBLDCTRN('INFO ') － 現在のトランザクションに関する情報の報告
- CBLDCTRN('RMSELECT')－リソースマネジャ接続先選択
- CBLDCTRN('U-COMMIT') － 非連鎖モードのコミット
- CBLDCTRN('U-ROLL ') － 非連鎖モードのロールバック

トランザクション制御の COBOL-UAP 作成用プログラム（CBLDCTRN）は，TP1/Server Base と TP1/LiNK のどちらの UAP でも使えます。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。トランザクション制御（CBLDCTRN）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCTRN.cbl に格納してあります。

CBLDCTRN('BEGIN ')

名称

トランザクションの開始

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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 ’BEGIN   ’.
   02 データ名B PIC X(5).
```

機能

グローバルトランザクションを，CBLDCTRN('BEGIN ')を呼び出したプロセスから開始します。トランザクションを開始したプロセスは，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になります。

トランザクションを開始する UAP は，実行環境の設定でトランザクション属性を指定しておいてください。また，すでにトランザクションを開始しているグローバルトランザクションの中では，どのトランザクションブランチからも CBLDCTRN('BEGIN ')は呼び出せません。一つのグローバルトランザクション中で重複して呼び出した場合はエラーリターンします。

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開始を示す要求コードを「VALUE 'BEGIN△△△'」と設定します。トランザクションを開始したあとの処理でも，この内容は変化しません。

OpenTP1 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。グローバルトランザクションが開始して，CBLDCTRN('BEGIN ')を呼び出したプロセスはその範囲に含まれています。

ステータスコード	意味
00905	サービスプログラムは正しくないコンテクスト（例えば，すでにトランザクション内にいる）からサービス要求されました。または，実行環境がジャーナルファイルレスモードのため，トランザクションは開始できませんでした。
00906	リソースマネジャ（RM）でエラーが起こりました。トランザクションは開始できませんでした。
00907	トランザクションサービスでエラーが起こったので，トランザクションは開始できませんでした。トランザクションサービス定義の trn_tran_process_count オペランドの指定値が不足している可能性があります。 このステータスコードが戻った場合は，再び実行すれば成功する可能性が高いので，再実行してください。
00908	要求コード（データ名 A）が間違っています。

指定例

```
01 MISC.
    02 CMD-CODE     PIC X(8).
    02 STATUS-CODE  PIC X(5).
      :
      :
MOVE 'BEGIN' TO CMD-CODE OF MISC.
CALL 'CBLDCTRN' USING MISC.
IF STATUS-CODE OF MISC NOT EQUAL TO '00000' THEN
    MOVE 'CANNOT BEGIN TRANSACTION' TO ERRMSG OF OUT-ERROR-REC
    WRITE OUT-ERROR-REC
END IF.
```

CBLDCTRN('C-COMMIT')

名称

連鎖モードのコミット

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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，リソースマネジャに知らせます。

CBLDCTRN('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て，CBLDCTRN('C-COMMIT')を呼び出したプロセスは，トランザクションの範囲内です。ただし，CBLDCTRN('C-COMMIT')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。

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（CBLDCTRN('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。

CBLDCTRN('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（トランザクションを開始した UAP）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「00905」でエラーリターンします。

CBLDCTRN('C-COMMIT')を呼び出すプロセスは，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。

CBLDCTRN('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。CBLDCTRN('C-COMMIT')を呼び出すサービスが正常終了するためには，UAP の実行環境を設定するときに，トランザクション属性を指定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

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，この内容は変化しません。

OpenTP1 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。CBLDCTRN('C-COMMIT')が終了したあとも，このプロセス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。
00902	現在のトランザクションは，コミットできないでロールバックしました。このステータスコードが返ったあとも，このプロセスはトランザクション下にあって，新しく開始したグローバルトランザクションの範囲内です。
00903	CBLDCTRN('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定のため，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定の結果が，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，このプロセス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。
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完了しました。しかし，障害のため，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同期点の結果が判明しません。 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，このプロセス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。 トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，1 相コミット時にリソースマネジャからのリターン値が XAER_NOTA の場合も，ステータスコード 00904 を返します。
00905	CBLDCTRN('C-COMMIT')を，正しくないコンテクスト（例えば，すでにトランザクション中にいない）から呼び出しています。トランザクションモードに対する影響はありません。
00908	要求コード（データ名 A）が間違っています。
00924	コミット処理は正常に終了しましたが，新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。

ステータスコード	意味
00925	コミットしようとしたトランザクションは，コミットできないでロールバックしました。新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。
00926	CBLDCTRN('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定のため，あるトランザクションブランチはコミットとなって，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定の結果が，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されます。 このステータスコードが返される原因となった UAP，リソースマネジャ，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の内容を参照してください。新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。
00927	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完了しました。しかし，障害のため，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される原因となった UAP，リソースマネジャ，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の内容を参照してください。 新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。 トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，リソースマネジャからのリターン値が XAER_NOTA の場合も，ステータスコード 00927 を返します。

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 'CBLDCTRN' 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.
```

CBLDCTRN('C-ROLL ')

名称

連鎖モードのロールバック

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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).
```

機能

トランザクションをロールバックします。連鎖モードでロールバックしたあとには続けてトランザクションが開始します。

CBLDCTRN('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ジャにロールバックを知らせます。

CBLDCTRN('C-ROLL ')が正常に終了すると，CBLDCTRN('C-ROLL ')を呼び出したプロセス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て，CBLDCTRN('C-ROLL ')を呼び出したプロセスはトランザクションの範囲内です。ただし，CBLDCTRN('C-ROLL ')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。

CBLDCTRN('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（トランザクションを開始した UAP）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「00905」でエラーリターンします。

CBLDCTRN('C-ROLL ')を呼び出すプロセスは，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。

CBLDCTRN('C-ROLL ')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。CBLDCTRN('C-ROLL ')を呼び出すサービスが正常終了するためには，UAP の実行環境を設定するときに，トランザクション属性を指定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

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。

OpenTP1 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。CBLDCTRN('C-ROLL ')が終了したあとも，このプロセス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。
00903	CBLDCTRN('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定のため，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定の結果が，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，このプロセス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。
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完了しました。しかし，障害のため，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同期点の結果が判明しません。 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，このプロセス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。
00905	CBLDCTRN('C-ROLL ')を正しくないコンテクスト（例えば，すでにトランザクション中にいない）から呼び出しています。トランザクションモードに対する影響はありません。
00908	要求コード（データ名 A）が間違っています。
00924	ロールバック処理は正常に終了しましたが，新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。
00926	CBLDCTRN('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定のため，あるトランザクションブランチはコミットとなって，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定の結果が，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されます。

ステータスコード	意味
00926	このステータスコードが返される原因となったUAP，リソースマネジャ，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の内容を参照してください。新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。
00927	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完了しました。しかし，障害のため，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される原因となったUAP，リソースマネジャ，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の内容を参照してください。 新しいトランザクションは開始できませんでした。このステータスコードが返ったあと，このプロセスはトランザクション下にはありません。

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  'CBLDCTRN' 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.
```

CBLDCTRN('INFO ')

名称

現在のトランザクションに関する情報の報告

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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 ’INFO    ’.
   02 データ名B    PIC X(5).
   02 FILLER     PIC X.
   02 データ名C.
      03 データ名D PIC S9(4) COMP VALUE ZERO.
```

機能

CBLDCTRN('INFO ')を呼び出した UAP が，現在トランザクションとして処理しているかどうかをリターンします。

CBLDCTRN('INFO ')を呼び出すプロセスは，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。CBLDCTRN('INFO ')を呼び出すサービスが正常終了するためには，UAP の実行環境を設定するときに，トランザクション属性を指定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

現在のトランザクションに関する情報の報告を示す要求コードを「VALUE 'INFO△△△△'」と設定します。以降の処理でも，この内容は変わりません。

●データ名 C

現在のトランザクションに関する情報を格納する領域です。

●データ名 D

0 を設定します。

OpenTP1 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
00001	CBLDCTRN('INFO ')を呼び出したプロセスは，トランザクションとして稼働しています。
00000	CBLDCTRN('INFO ')を呼び出したプロセスは，トランザクションとして稼働していません。
00908	要求コード（データ名 A）が間違っています。

指定例

```
01 MISC.
    02 CMD-CODE     PIC X(8).
    02 STATUS-CODE  PIC X(5).
    02 FILLER       PIC X.
    02 TRAN-INFO.
       03 LEN       PIC S9(4) COMP.
         :
         :
MOVE ZERO TO LEN OF TRAN-INFO OF MISC.
MOVE 'INFO' TO CMD-CODE OF MISC.
CALL 'CBLDCTRN' USING MISC.
IF STATUS-CODE OF MISC NOT EQUAL TO '00001' THEN
    MOVE 'NOW IN TRANSACTION' TO ERRMSG OF OUT-ERROR-REC
    WRITE OUT-ERROR-REC
END IF.
```

注意事項

この API は UAP トレースを取得していません。

CBLDCTRN('RMSELECT')

名称

リソースマネジャ接続先選択

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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 ’RMSELECT’.
   02 データ名B PIC X(5).
01 一意名2.
   02 データ名C PIC S9(9) COMP VALUE ZERO.
   02 データ名D PIC X(32).
   02 データ名E PIC X(3).
```

機能

ユーザサービス定義の trnrmid 定義コマンドに-k オプションを指定したリソースマネジャについて，該当するプロセスまたはトランザクションで接続先となるリソースマネジャ（リソースマネジャ名＋リソースマネジャ拡張子）を指定します。

CBLDCTRN('RMSELECT')は，グローバルトランザクション外でだけ呼び出せます。トランザクションブランチ内で CBLDCTRN('RMSELECT')を呼び出した場合は，エラーリターンします。

CBLDCTRN('RMSELECT')が正常に終了したあとは，指定されたリソースマネジャにだけ接続するようになり，trnrmid 定義コマンドに-k オプションの指定があるほかの同一名称のリソースマネジャには接続しません。同一 UAP プロセス内で，先に指定したリソースマネジャから別のリソースマネジャに変更する場合は，実行中のトランザクションが完了し新たなトランザクションを開始する前に変更対象となるリソースマネジャ（リソースマネジャ名とリソースマネジャ拡張子）を指定した CBLDCTRN('RMSELECT')を呼び出してください。

UAP で値を設定するデータ領域

●データ名 A

接続先となるリソースマネジャ名の設定を示す要求コードを「VALUE 'RMSELECT'」と設定します。以降の処理でも，この内容は変わりません。

●データ名 C

0 を設定します。

●データ名 D

接続先となるリソースマネジャ名（trnrmid 定義コマンドの-n オプション指定値）を 31 文字以内の識別子で指定します。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

データ名 D に指定したリソースマネジャに指定した拡張子（trnrmid 定義コマンドの-n オプション指定値）のうち，接続先となる拡張子を 2 文字以内の識別子で指定します。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。

OpenTP1 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。 指定されたリソースマネジャは，CBLDCTRN('RMSELECT')を呼び出したプロセスで実行されるトランザクションで接続先として設定されました。
00905	CBLDCTRN('RMSELECT')を正しくないコンテクスト（例えば，すでにトランザクション内にいる）で呼び出しています。または，実行環境がジャーナルファイルレスモードの場合に関数が呼び出されました。
00906	リソースマネジャでエラーが起こりました。指定されたリソースマネジャの XA 関数の呼び出し処理でエラーが発生しました。
00908	引数の指定値に次の誤りがあります。 • トランザクションサービス定義の trnstring 定義コマンドに指定されたリソースマネジャが存在しない。 • trnrmid 定義コマンドの-k オプションが指定されていない。 • データ名 C に無効な値が指定されている。

指定例

```
01 MISC1.
   02 CMD-CODE PIC X(8) VALUE 'RMSELECT'.
   02 STATUS-CODE PIC X(5).
01 MISC2.
   02 FLAGS PIC S9(9) COMP VALUE ZERO.
   02 RMNAME PIC X(32) VALUE SPACES.
   02 INFO PIC X(3) VALUE SPACES.
     :
     :
MOVE 'RM_A ' TO RMNAME.
MOVE 'Z1 ' TO INFO.
CALL 'CBLDCTRN' USING MISC1 MISC2.
IF STATUS-CODE OF MISC1 NOT EQUAL TO '00000' THEN
```

```
   MOVE 'CANNOT RM SELECT' TO ERRMSG OF OUT-ERROR-REC
   WRITE OUT-ERROR-REC
END IF.
```

CBLDCTRN('U-COMMIT')

名称

非連鎖モードのコミット

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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 ’U-COMMIT’.
   02 データ名B PIC X(5).
```

機能

グローバルトランザクション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，リソースマネジャに知らせます。

CBLDCTRN('U-COMMIT')が正常に終了したあとには，新しいグローバルトランザクションは開始しません。

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（CBLDCTRN('U-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。

CBLDCTRN('U-COMMIT')を呼び出せるのは，ルートトランザクションブランチ（トランザクションを開始した UAP）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「00905」でエラーリターンします。

CBLDCTRN('U-COMMIT')を呼び出すプロセスは，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。

CBLDCTRN('U-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。CBLDCTRN('U-COMMIT')を呼び出すサービスが正常終了するためには，UAP の実行環境を設定するときに，トランザクション属性を指定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

非連鎖モードのコミットを示す要求コードを「VALUE 'U-COMMIT'」と設定します。非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。

OpenTP1 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。このプロセスはトランザクション下にありません。プロセスはグローバルトランザクションの範囲の外です。
00902	現在のトランザクションは，コミットできないでロールバックしました。プロセスはトランザクションの範囲の外です。
00903	CBLDCTRN('U-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定のため，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定の結果が，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は，このプロセスはトランザクション下にはありません。プロセスはグローバルトランザクションの範囲の外です。
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完了しました。しかし，障害のため，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同期点の結果が判明しません。 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，このプロセスはトランザクション下にはありません。プロセスはグローバルトランザクションの範囲の外です。 トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，1 相コミット時にリソースマネジャからのリターン値が XAER_NOTA の場合も，ステータスコード 00904 を返します。
00905	CBLDCTRN('U-COMMIT')を，正しくないコンテクスト（例えば，すでにトランザクション中にいない）から呼び出しています。トランザクションモードに対する影響はありません。
00908	要求コード（データ名 A）が間違っています。

指定例

```
01 MISC.
   02 CMD-CODE     PIC X(8).
   02 STATUS-CODE  PIC X(5).
     :
     :
MOVE 'U-COMMIT' TO CMD-CODE OF MISC.
CALL 'CBLDCTRN' 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.
```

CBLDCTRN('U-ROLL ')

名称

非連鎖モードのロールバック

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCTRN’ 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 ’U-ROLL  ’.
   02 データ名B PIC  X(5).
```

機能

トランザクションをロールバックします。非連鎖モードでロールバックしたあとには，トランザクションは続けて開始しません。

CBLDCTRN('U-ROLL ')を呼び出すことで，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ジャにロールバックを知らせます。

CBLDCTRN('U-ROLL ')は，グローバルトランザクションのどのトランザクションブランチからでも呼び出せます。ルートトランザクションブランチから呼び出した場合，CBLDCTRN('U-ROLL ')が正常に終了したあとには，新しいトランザクションは開始しません。ルートトランザクションブランチ以外から呼び出した場合は，そのトランザクションブランチを rollback_only 状態にします。この場合，ルートトランザクションブランチの同期点処理が完了するまで，CBLDCTRN('U-ROLL ')を呼び出したトランザクションブランチはトランザクションの範囲内です。

CBLDCTRN('U-ROLL ')を呼び出すプロセスは，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。

CBLDCTRN('U-ROLL ')を呼び出すサービスが正常に終了するためには，UAP の実行環境を設定するときに，トランザクション属性を指定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

非連鎖モードのロールバックを示す要求コードを「VALUE 'U-ROLL△△'」と設定します。非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。

OpenTP1 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。CBLDCTRN('U-ROLL ')をルートトランザクションブランチから呼び出している場合は，このプロセスはトランザクション下にありません。プロセスはグローバルトランザクションの範囲の外です。ルートトランザクションブランチ以外から呼び出している場合は，このプロセスを rollback_only 状態とします。
00903	CBLDCTRN('U-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定のため，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定の結果が，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，このプロセスはトランザクション下にはありません。プロセスはグローバルトランザクションの範囲の外です。
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完了しました。しかし，障害のため，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同期点の結果が判明しません。 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，このプロセスはトランザクション下にはありません。プロセスはグローバルトランザクションの範囲の外です。
00905	CBLDCTRN('U-ROLL ')を，正しくないコンテクスト（例えば，すでにトランザクション中にいない）から呼び出しています。トランザクションモードに対する影響はありません。
00908	要求コード（データ名 A）が間違っています。

指定例

```
01 MISC.
    02 CMD-CODE     PIC X(8).
    02 STATUS-CODE  PIC X(5).
      :
      :
MOVE  'U-ROLL  ' TO CMD-CODE OF MISC.
CALL  'CBLDCTRN' 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.
```

オンラインテスタの管理（CBLDCUTO）

OpenTP1 でオンラインテスタ（TP1/Online Tester）を使っている場合に，ユーザサーバから状態を管理するプログラムについて説明します。オンラインテスタの管理の COBOL-UAP 作成用プログラムを次に示します。

- CBLDCUTO('T-STATUS') − ユーザサーバのテスト状態の報告

オンラインテスタの管理の COBOL-UAP 作成用プログラム（CBLDCUTO）は，TP1/Server Base の UAP でだけ使えます。TP1/LiNK の UAP では，オンラインテスタの COBOL-UAP 作成用プログラムは使えません。

COBOL-UAP 作成用プログラムのデータ部（DATA DIVISION）を定義するときには，サンプルとして COBOL 言語用テンプレートが使えます。オンラインテスタの管理（CBLDCUTO）の COBOL 言語用テンプレートは，/BeTRAN/examples/COBOL/ディレクトリの下の DCUTO.cbl に格納してあります。

CBLDCUTO('T-STATUS')

名称

ユーザサーバのテスト状態の報告

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCUTO’ 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 ’T-STATUS’.
   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(1).
   02 データ名E  PIC X(1).
   02 データ名F  PIC X(1).
   02 データ名G  PIC X(1).
   02 データ名H  PIC X(1).
   02 FILLER    PIC X(22).
```

機能

CBLDCUTO('T-STATUS')を呼び出したユーザサーバのテスト状態を報告します。テスト状態は，CBLDCUTO('T-STATUS')が正常に終了したあと，データ名 D，データ名 E，データ名 F，データ名 G，データ名 H に設定されます。

CBLDCUTO('T-STATUS')がエラーリターンした場合は，テスト状態を示す情報は保証しません。

UAP で値を設定するデータ領域

●データ名 A

ユーザサーバのテスト状態の報告を示す要求コードを「VALUE 'T-STATUS'」と設定します。

●データ名 Z

0 を設定します。

OpenTP1 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

テストユーザ ID（環境変数 DCUTOKEY に設定した値）が設定されます。

●データ名 D

ユーザサーバがテストモードで稼働しているかどうかが設定されます。

VALUE 'T' … テストモードで稼働しています。

VALUE 'N' … テストモードで稼働していません。

●データ名 E

グローバルトランザクションの処理状態が設定されます。

VALUE 'C' … 同期点処理でコミットします。

VALUE 'R' … 同期点処理でロールバックします。

VALUE 'N' … 非トランザクションの状態です。

空白 … 非テストモードです。または，MCF のライブラリを結合した MHP です。

●データ名 F

ユーザサービス定義の test_mode オペランドに指定した，テスト種別が設定されます。

VALUE 'T' … テスト専用 UAP（target）としてテストしています。

VALUE 'U' … 使用可能 UAP（usable）としてテストしています。

VALUE 'S' … シミュレート MHP（simmhp）としてテストしています。

VALUE 'N' … テスト対象外 UAP（no）です。

●データ名 G

ユーザサービス定義の test_transaction_commit オペランドで指定した，トランザクションの同期点の扱いが設定されます。

VALUE 'C' … 同期点でコミット（Y）します。

VALUE 'R' … 同期点でロールバック（N）します。

空白 … 非テストモードです。または，MCF のライブラリを結合した MHP です。

●データ名 H

ユーザサービス定義の test_adm_call_command オペランドで指定した，コマンドの実行結果の扱いが設定されます。

VALUE 'D' … コマンドを実行（do）します。

VALUE 'S' … 実行結果に仮定値を設定（skip）します。

VALUE 'F' … 運用コマンド結果データファイルのデータを使用（file）します。

空白 … 非テストモードです。または，MCF のライブラリを結合した MHP 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。各データ領域にテストの状態が設定されました。
02701	CBLDCRPC('OPEN ')を呼び出していません。
02734	UAP が，現在稼働しているトランザクションサービスでは動作できないバージョンの OpenTP1 ライブラリと結合されています。
02757	データ名 Z に設定した値が間違っています。
02759	要求コード（データ名 A）が間違っています。

注意事項

MCF のライブラリと結合した MHP から，CBLDCUTO('T-STATUS')を呼び出した場合は，データ領域には次に示す値が設定されます。

- データ名 C：テストユーザ ID
- データ名 D：現在動作しているサービスのモード
- データ名 E：空白
- データ名 F：'N'
- データ名 G：空白
- データ名 H：空白

3 OpenTP1 の COBOL-UAP 作成用プログラムの文法（DML インタフェース）

この章では，OpenTP1 の UAP を COBOL 言語で作成する場合の UAP 作成用プログラム（DML インタフェース）の文法について説明します。

データ操作言語を使ったコーディング

COBOL 言語で UAP を作成するときに，OpenTP1 のメッセージ送受信機能を，**データ操作言語**（DML Data Manipulation Language）で作成できます。一つのサービスをコーディングするときには，CALL 文と DML は共存させることができます。

DML は，TP1/Server Base の場合にだけ使えます。TP1/LiNK では，DML は使えません。

DML は，作業領域を定義する**通信記述項**と，命令文として使う**通信文**で構成されます。

DML で使えるメッセージ送受信の機能を次の表に示します。

表 3-1　DML で使えるメッセージ送受信の機能

通信文		機能	対応する CALL インタフェース
データコミュニケーション機能	RECEIVE※1	メッセージの受信	CBLDCMCF('RECEIVE ')
		同期型のメッセージの受信	CBLDCMCF('RECVSYNC')
	SEND※1	メッセージの送信	CBLDCMCF('SEND ')
		応答メッセージの送信	CBLDCMCF('REPLY ')
		同期型のメッセージの送信	CBLDCMCF('SENDSYNC')
		同期型のメッセージの送受信	CBLDCMCF('SENDRECV')
	ENABLE※2	同期型のメッセージの送信	CBLDCMCF('SENDSYNC')
	DISABLE※2	同期型のメッセージの送信	CBLDCMCF('SENDSYNC')
サービス機能	COMMIT	MHP のコミット	CBLDCMCF('COMMIT ')
	DISABLE	継続問い合わせ応答の終了	CBLDCMCF('CONTEND ')
	RECEIVE	一時記憶データの受け取り	CBLDCMCF('TEMPGET ')
	ROLLBACK	MHP のロールバック	CBLDCMCF('ROLLBACK')
	SEND	アプリケーションプログラムの起動	CBLDCMCF('EXECAP ')
	SEND	一時記憶データの更新	CBLDCMCF('TEMPPUT ')
	SEND	運用コマンドの実行	CBLDCADM('COMMAND ')
	SEND	ユーザジャーナルの取得	CBLDCJNL('UJPUT ')

注

メッセージの再送 － CBLDCMCF('RESEND ')の DML インタフェースはありません。

注※1

文法については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

注※2

通信プロトコル対応製品に TP1/NET/OSI-TP を使った場合にだけ使えます。文法については，マニュアル「OpenTP1 プロトコル TP1/NET/OSI-TP 編」を参照してください。

文法上の一般規則

OpenTP1 の UAP で使う DML は，COBOL 言語本来の文法で記述された部分と，通信節，通信文から構成されます。ここでは通信節，通信文の記述規則，および COBOL 言語本来の記述上で従わなければならない文法規則について説明します。ここで説明していない項目は COBOL 言語の文法規則に従います。COBOL 言語の一般的な文法規則については，該当するマニュアルを参照してください。

記述記号

ここで説明する，記述形式に使う記号を次に示します。

記述記号	意味
〔　〕	この記号で囲まれている内容は省略できることを示します。 （例）〔BEFORE　ERASING〕
{　}	この記号で囲まれている項目のうち，一つだけ選択することを示します。
____	この下線記号で示す予約語は，必要語なので省略できないことを示します。下線がない予約語は，補助語なので書いても書かなくてもかまいません。

一意名およびデータ名

このマニュアルでは，一意名およびデータ名という用語を COBOL 言語で使う場合と同じ意味で使います。

（例）

データ名には添字や指標は付けられませんが，一意名には付けられます。ただし，通信文中の一意名には付けられません。

形式の詳細

通信節（COMMUNICATION SECTION）の記述規則

通信節は，データ部（DATA DIVISION）に記述します。通信節を含めた DATA DIVISION の一般形式を次に示します。データ部の各節は，ここに示す形式で記述してください。

```
DATA DIVISION.
  〔 FILE SECTION.
    :  〕
  〔 WORKING-STORAGE SECTION.
    :  〕
  〔 LINKAGE SECTION.
```

```
   :  ]
[ COMMUNICATION SECTION.
通信記述項
   :  ]
[ REPORT SECTION.
   :  ]
```

通信節は，節の見出し（COMMUNICATION SECTION という予約語列をピリオドと空白で止めたもの）で書き始めます。見出しのあとに一つ以上の通信記述項（CD）を記述します。

通信記述項（CD）の記述規則

通信記述項は，通信種別および UAP と OpenTP1 のインタフェース領域を定義します。通信記述項の一般形式を次に示します。

```
CD 通信記述名
FOR  {INPUT|OUTPUT|I-O}  [STORAGE|JOURNAL|PROGRAM|COMMAND]
  [ STATUS KEY IS データ名1]
  [ SYMBOLIC TERMINAL IS データ名2]
  [ MESSAGE DATE IS データ名3]
  [ MESSAGE TIME IS データ名4]
  [ MAP NAME IS データ名5]
  [ SYNCHRONOUS MODE IS {SYNC|ASYNC|データ名6}]
  [ SWITCHING MODE IS {NORMAL|PRIOR|データ名7}]
  [ NEXT TRANSACTION IS データ名8]
  [ ACTIVE INTERVAL IS データ名9]
  [ DETAIL MODE IS データ名10]
  [ WAITING TIME IS データ名11] .
```

- 通信記述項は，レベル指示語である CD で書き始めて，必ずピリオドと空白で止めます。
- レベル指示語 CD に続けて，FOR 句を記述します。FOR 句以降の各句の順序は，任意です。
- 通信記述名，およびデータ名 1 からデータ名 11 には，使用する COBOL 言語で使える名称を指定します。また，OpenTP1 の UAP では使えない名称は指定できません。
- 一つの通信記述項は，20 行以内で記述してください。COBOL の継続行の機能は使えません。さらに，一つの通信記述項の途中に，注釈行，空白行は記述できません。

通信文の記述規則

通信文は，手続き部（PROCEDURE DIVISION）で使います。通信文の一般形式を次に示します。

```
RECEIVE 通信記述名 { [FIRST] SEGMENT|MESSAGE}
        [INTO 一意名1]  [ BEFORE ERASING ] .


SEND 通信記述名   [FROM 一意名1]
     [WITH {ESI|EMI|一意名2}]
     [ BEFORE RECEIVING MESSAGE INTO 一意名3] .


DISABLE 通信記述名   [ WITH 一意名1] .
```

```
COMMIT.

ROLLBACK  〔WITH STOPPING〕.
```

- 通信文の各句は，一般形式で示した順序で記述します。
- 通信文中の通信記述名は，必ず通信節で定義しておいてください。
- 通信文中に一意名を指定する場合，それは作業場所節で定義したものでなければなりません。また，OpenTP1 で使えない名称は指定できません。
- 通信文は複数行にわたって記述できます。ただし，標識領域（indicator area）にハイフン（-）を書く継続行の記述方法は使えません。また，一つの通信文の途中に，注釈行，または空白行は使えません。
- 通信文は COBOL 言語本来の文，または段落名と，同じ行に記述しないでください。
- 通信文中に記述する一意名は，01 レベル，または 77 レベルを使ってください。

通信記述項について

通信文で使う通信記述項は，複数の通信文で共用できます。これらの通信記述項には，UAP で値を設定するデータ名と，OpenTP1 から値が返されるデータ名があります。OpenTP1 から値が返されるデータ名以外は，通信文の前後で内容が変わりません。したがって，複数の文で同じ通信記述項を使う場合，前回使った通信文と同じ内容でよいデータ名は，値を再設定する必要がありません。

通信記述項のうちで，データ名を指定する句と，編集形式の一覧を次の表に示します。

表 3-2　通信記述項でデータ名を指定する句と，編集形式の一覧

データ名を指定する句	データ領域の形式	データ領域の値の設定元※									
		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
STATUS KEY	データ名 1 PIC X(5).	B	B	B	B	B	B	B	B	B	B
SYMBOLIC TERMINAL	データ名 2 PIC X(8).	B	−	U	U	U	U	U	U	−	−
MESSAGE DATE	データ名 3 PIC 9(6).	B	−	−	−	−	−	−	−	−	−
MESSAGE TIME	データ名 4 PIC 9(8).	B	−	−	−	−	−	−	−	−	−
MAP NAME	データ名 5 PIC X(8).	b	−	U	U	−	−	−	−	−	−
SYNCHRONOUS MODE	データ名 6 PIC X(1).	−	−	u	u	−	−	−	−	−	−
SWITCHING MODE	データ名 7	−	−	u	u	−	−	−	−	−	−

データ名を指定する句	データ領域の形式	データ領域の値の設定元※									
		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
SWITCHING MODE	PIC X(1).	－	－	u	u	－	－	－	－	－	－
NEXT TRANSACTION	データ名 8 PIC X(8).	－	－	u	－	－	－	－	－	－	－
ACTIVE INTERVAL	データ名 9 PIC X(8).	－	－	－	－	－	－	u	－	－	－
DETAIL MODE	データ名 10 PIC X(1).	－	－	u	u	－	－	－	－	－	－
WAITING TIME	データ名 11 PIC 1(32) BIT.	－	－	u	u	－	－	－	－	－	－

（凡例）

1.：先頭セグメントの受信（RECEIVE）

2.：中間，最終セグメントの受信（RECEIVE）

3.：先頭セグメントの送信（SEND）

4.：中間，最終セグメントの送信（SEND）

5.：継続問い合わせ応答の終了（DISABLE）

6.：一時記憶データの受け取り（RECEIVE）

7.：アプリケーションプログラムの起動（SEND）

8.：一時記憶データの更新（SEND）

9.：運用コマンドの実行（SEND）

10.：ユーザジャーナルの取得（SEND）

B：OpenTP1 から値が返されます。

b：ある条件のときには，OpenTP1 から値が返されます。

U：UAP で値を設定します。

u：ある条件のときには，UAP で値を設定します。

－：該当しません。

注※

COMMIT 文および ROLLBACK 文では通信記述項を使いません。

データコミュニケーション機能

DML のデータコミュニケーション機能で使うプログラムを示します。データコミュニケーション機能の COBOL-UAP 作成用プログラムを次に示します。

- RECEIVE − メッセージの受信
- SEND − メッセージの送信

これらの DML の形式，データ領域に設定する値，およびステータスコードは，使う通信プロトコルによって異なります。詳細はマニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

RECEIVE － メッセージの受信

名称

メッセージの受信

形式

DATA DIVISION（通信記述項）の指定

```
CD 通信記述名
   FOR {INPUT|I-O}
  〔STATUS KEY IS データ名1〕
  〔SYMBOLIC TERMINAL IS データ名2〕
  〔MESSAGE DATE IS データ名3〕
  〔MESSAGE TIME IS データ名4〕.

01 一意名1.
   02 データ名5  PIC  9(4) COMP.
   02 データ名6  PIC  X(4).
   02 データ名7  PIC  X(n).
```

PROCEDURE DIVISION（通信文）の指定

```
RECEIVE 通信記述名
    〔FIRST〕 SEGMENT
       INTO 一意名1.
```

機能

次に示す CALL インタフェースの機能を実現します。

- メッセージの受信　CBLDCMCF('RECEIVE ')
- 同期型のメッセージの受信　CBLDCMCF('RECVSYNC')

受信できる一つのセグメントの最大長は，32763 バイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

また，相手システムから通信プロトコルを介して送られたメッセージを受信する場合，通信プロトコルによって RECEIVE 文の文法が異なります。相手システムからのメッセージを受信する RECEIVE 文の文法についても，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

通信記述項に設定する項目

●FOR 句

次のうちどちらかを指定します。

INPUT：問い合わせ以外のメッセージの受信

I-O：問い合わせメッセージの受信

●STATUS KEY 句

ステータスコードを受け取りたいときに指定します。省略した場合は，ステータスコードを受け取りません。

●SYMBOLIC TERMINAL 句

入力元の論理端末名称を参照するデータ項目を指定します。

●MESSAGE DATE 句

メッセージを受信した日付を参照するデータ項目を指定します。「YYMMDD」（YY：西暦の下 2 けた　MM：月　DD：日）の形式で参照できます。

●MESSAGE TIME 句

メッセージを受信した時刻を参照するデータ項目を指定します。「HHMMSS00」（HH：時　MM：分　SS：秒　00 は固定）の形式で参照できます。

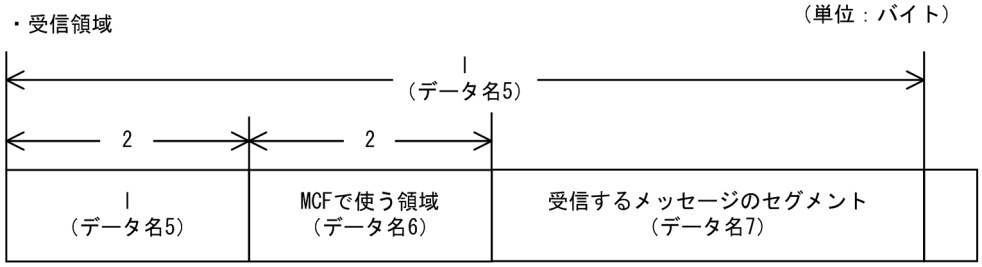
通信文に指定する項目

●FIRST

先頭セグメントを受信するときに指定します。

●一意名 1

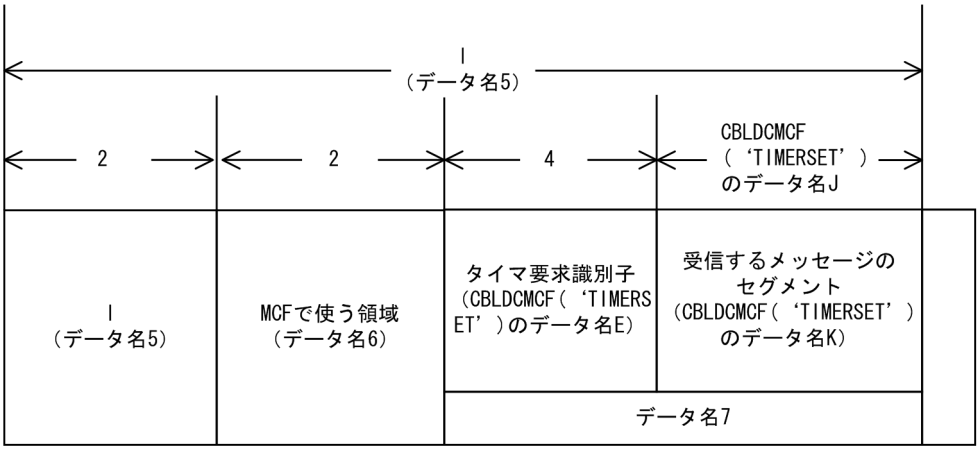
セグメントを受信するデータ項目を指定します。自システムから送信したメッセージの場合，受け取れるセグメントの最大長は 32000 バイトです。相手システムから送信されたメッセージの場合，受け取れるセグメントの最大長は通信プロトコル対応製品別で異なります。



また，ユーザタイマ監視を設定したときに指定したメッセージを受信する領域（一意名 1 で示す領域）の形式を次に示します。

・受信領域

（単位：バイト）



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。
	受信領域の長さより短いセグメントを受信しました。余りの受信領域には空白が設定されます。
71000	先頭セグメントを受信する RECEIVE 文を 2 回以上実行しています。中間セグメントおよび最終セグメントを受信する場合は，FIRST を指定しないで RECEIVE 文を実行してください。
71001	メッセージの最終セグメントを受信したあとで，次のセグメントを受信する RECEIVE 文を実行しています。直前に実行した RECEIVE 文でメッセージはすべて受信しました。 このステータスコードが返されたあとに，再び RECEIVE 文を実行した場合は，ステータスコード「72000」が返されます。
71002	メッセージキューへの入出力処理時に障害が起こりました。
	メッセージキューが閉塞されています。
72000	＜ MHP の実行でリターンした場合＞ 先頭セグメントを受信する RECEIVE 文を実行する前に，中間セグメントおよび最終セグメントを受信する RECEIVE 文を実行しています。先頭セグメントを受信する場合は，FIRST を指定して RECEIVE 文を実行してください。 ステータスコード「71000」が返されたあとで，RECEIVE 文を実行しています。
	＜ SPP の実行でリターンした場合＞ SPP では RECEIVE 文を実行できません。
72001	SYMBOLIC TERMINAL 句に設定した論理端末名称が間違っています。
	RECEIVE 文を実行できない論理端末を設定しています。
72013	受信領域の長さを超えるセグメントを受信しました。受信領域の長さを超えた部分は切り捨てられました。
	バッファ形式 2 の場合で，32767 バイトを超えるセグメントを受信しました。32767 バイトを超えた部分は切り捨てられました。
72020	SYNCHRONOUS MODE 句に設定した値が間違っています。

ステータスコード	意味
72024	FOR 句に設定した値が間違っています。
72036	セグメントを受信する領域（一意名 1）の長さが不足しています。5 バイト以上の領域を確保してください。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

SEND － メッセージの送信

名称

メッセージの送信

形式

形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

機能

次に示す CALL インタフェースの機能を実現します。

- 応答メッセージの送信　CBLDCMCF('REPLY ')
- メッセージの送信　CBLDCMCF('SEND ')
- 同期型のメッセージの送受信　CBLDCMCF('SENDRECV')
- 同期型のメッセージの送信　CBLDCMCF('SENDSYNC')

受信できる一つのセグメントの最大長は，32763 バイトまでです。また，送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

また，メッセージを送信する SEND 文は，通信プロトコルによって文法が異なります。相手システムへメッセージを送信する SEND 文の文法についても，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

注意事項

CBLDCMCF('SEND ')に該当する機能を使用する場合，MCF マネジャ定義の UAP 共通定義（mcfmuap -c order）の指定によって，メッセージの送信順序が異なりますので注意してください。

サービス機能

DML のサービス機能で使うプログラムについて説明します。DML のサービス機能の COBOL-UAP 作成用プログラムを次に示します。

- COMMIT − MHP のコミット
- DISABLE − 継続問い合わせ応答の終了
- RECEIVE − 一時記憶データの受け取り
- ROLLBACK − MHP のロールバック
- SEND − アプリケーションプログラムの起動
- SEND − 一時記憶データの更新
- SEND − 運用コマンドの実行
- SEND − ユーザジャーナルの取得

COMMIT － MHP のコミット

名称

MHP のコミット

形式

PROCEDURE DIVISION（通信文）の指定

```
COMMIT.
```

機能

次に示す CALL インタフェースの機能を実現します。

- MHP のコミット　CBLDCMCF('COMMIT ')

COMMIT 文を使用する場合は，UAP 共通定義（mcfmuap）の-c オプションの commitdml オペランドに yes を指定してください。

COMMIT 文は，非応答型に指定（アプリケーション属性定義（mcfaalcap）の-n オプションで type オペランドに noans を指定）した MHP からだけ呼び出せます。

コミットに失敗した場合は，KFCA11199-E メッセージを出力し，MHP を異常終了させます。コミットが失敗する要因（メッセージ中の詳細コード）については，CBLDCMCF('COMMIT ')のステータスコードを参照してください。

UAP 共通定義（mcfmuap）の-c オプションの commitdml オペランドを省略または no を指定した場合，エラーを検出できません。

SPP から COMMIT 文を使用した場合，無効となります。

ステータスコード

ステータスコードはありません。

DISABLE － 継続問い合わせ応答の終了

名称

継続問い合わせ応答の終了

形式

DATA DIVISION（通信記述項）の指定

```
CD 通信記述名
   FOR  I-O  STORAGE
  〔STATUS KEY IS データ名1〕
```

PROCEDURE DIVISION（通信文）の指定

```
DISABLE 通信記述名.
```

機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCMCF('CONTEND ')　継続問い合わせ応答の終了

通信記述項に設定する項目

●FOR 句

継続問い合わせ応答の終了を示す I-O STORAGE を設定します。

●STATUS KEY 句

ステータスコードを受け取りたい場合に設定します。省略した場合は，ステータスコードを受け取れません。データ名 1 にステータスコードが返されます。

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。
72000	＜ MHP の実行でリターンした場合＞ 先頭セグメントを受信する RECEIVE 文を呼び出す前に，DISABLE（継続問い合わせ応答の終了）文を呼び出しました。
	＜ SPP の実行でリターンした場合＞ SPP では DISABLE（継続問い合わせ応答の終了）文を呼び出せません。
72042	通信記述項に SYMBOLIC TERMINAL 句の設定があり，空白以外が設定されています。

ステータスコード	意味
72101	継続問い合わせ応答型でないアプリケーションで，DISABLE（継続問い合わせ応答の終了）文を呼び出しました。
72107	DISABLE（継続問い合わせ応答の終了）文を 2 回以上呼び出しました。
72111	次起動アプリケーションを設定して SEND（メッセージの送信）文を呼び出したあと，DISABLE（継続問い合わせ応答の終了）文を呼び出しました。
	継続問い合わせ応答型のアプリケーション名を設定して SEND（アプリケーションプログラムの起動）文を呼び出したあと，DISABLE（継続問い合わせ応答の終了）文を呼び出しました。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

RECEIVE － 一時記憶データの受け取り

名称

一時記憶データの受け取り

形式

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR {INPUT|I-O} STORAGE
   〔STATUS  KEY  IS データ名1〕.

01 一意名1.
   02 データ名2  PIC  9(4) COMP.
   02 データ名3  PIC  X(4).
   02 データ名4  PIC  X(n).
```

PROCEDURE DIVISION（通信文）の指定

```
RECEIVE 通信記述名 {MESSAGE|SEGMENT}
   INTO 一意名1.
```

機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCMCF('TEMPGET ')　一時記憶データの受け取り

一時記憶データ格納バッファ領域（データ名 4）は，1～32000 バイトの長さを確保してください。

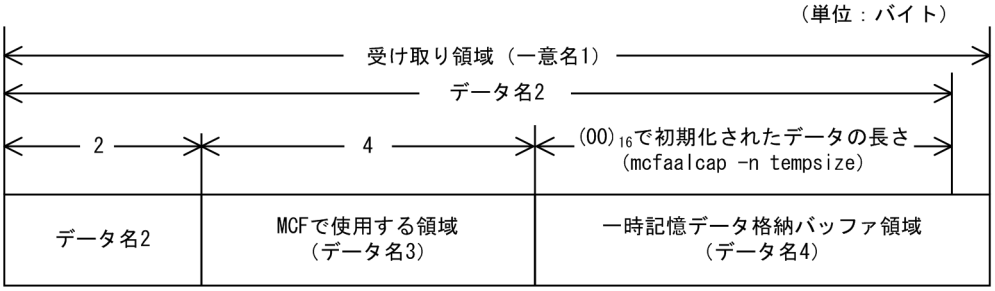
データ名 4 の長さを超える一時記憶データがある場合，超えた分については切り捨てます。

データ名 4 の長さと比べて一時記憶データの長さの方が短い場合，データ名 4 に一時記憶データを設定します。残りの領域については何も設定しません。

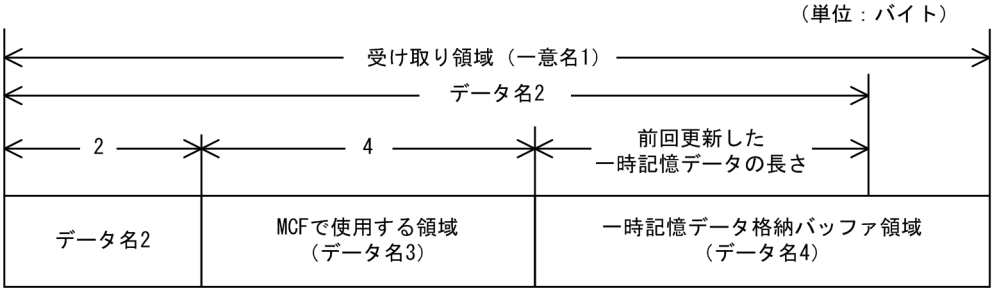
受け取り要求実行時，初期状態（継続問い合わせ応答開始後，SEND（一時記憶データの更新）文を 1 回も実行していない状態）の場合，アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドで指定した長さの$(00)_{16}$の一時記憶データがあるものとして実行します。

受け取り後の一意名 1 の形式を次に示します。

●SEND（一時記憶データの更新）文未実行（継続問い合わせ応答開始後，SEND（一時記憶データの更新）文を実行していない（初期状態））



●SEND（一時記憶データの更新）文実行済み（継続問い合わせ応答開始後，SEND（一時記憶データの更新）文を1回以上実行）



通信記述項に設定する項目

●FOR 句

一時記憶データの受け取りであることを示す，INPUT STORAGE または I-O STORAGE を設定します。

●STATUS KEY 句

ステータスコードを受け取りたい場合に設定します。省略した場合は，ステータスコードを受け取れません。データ名 1 にステータスコードが返されます。

通信文に指定する項目

●MESSAGE，SEGMENT

どちらかを設定します。

●データ名 2

前回更新した一時記憶データの長さ＋ 6 が返されます。初期状態の場合，継続問い合わせ応答用一時記憶領域の長さ（アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドの指定値）＋ 6 が返されます。

●データ名 3

MCF で使用する領域です。

●データ名 4

受け取った一時記憶データが返されます。初期状態の場合，継続問い合わせ応答用一時記憶領域の長さ（アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドの指定値）分だけ$(00)_{16}$が埋められます。

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。
72000	SPP では RECEIVE（一時記憶データの受け取り）文を呼び出せません。
72013	一時記憶データ格納バッファ領域の長さ（データ名 4 の長さ）を超える一時記憶データを受け取りました。一時記憶データ格納バッファ領域の長さを超える一時記憶データを切り捨てました。
72016	通信文に BEFORE ERASING が設定されています。
72036	受け取り領域の長さ（一意名 1 の長さ）が不足しています。7 バイト以上の領域を確保してください。
72042	通信記述項に SYMBOLIC TERMINAL 句の設定があり，空白以外が設定されています。
72101	継続問い合わせ応答型でないアプリケーションで，RECEIVE（一時記憶データの受け取り）文を呼び出しました。
72106	先頭セグメントを受信する RECEIVE 文を呼び出す前に，RECEIVE（一時記憶データの受け取り）文を呼び出しました。
72107	DISABLE（継続問い合わせ応答の終了）文を呼び出したあとで，RECEIVE（一時記憶データの受け取り）文を呼び出しました。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

ROLLBACK － MHP のロールバック

名称

MHP のロールバック

形式

PROCEDURE DIVISION（通信文）の指定

```
ROLLBACK 〔WITH STOPPING〕.
```

機能

次に示す CALL インタフェースの機能を実現します。

- MHP のロールバック　CBLDCMCF('ROLLBACK')

SPP および非トランザクション属性の MHP から ROLLBACK 文を使用した場合，無効となります。

通信文に指定する項目

●WITH STOPPING 句

MHP を異常終了させたあと，MHP に制御を戻さない場合に指定します。MHP に制御を戻す場合は，この句を省略します。ただし，メッセージを受信する前にこの句を省略した ROLLBACK 文を使っても，その ROLLBACK 文は無視されます。

このとき，UAP 異常終了通知イベント（ERREVT3）を起動します。また，アプリケーション異常終了限界回数にカウントします。アプリケーション属性定義（mcfaalcap）の指定によっては，アプリケーション，サービスグループ，およびサービスを閉塞します。

指定値の有無とシステムの動作の関係を，次の表に示します。

表 3-3　指定値の有無とシステムの動作の関係（ROLLBACK 文）

指定値	MHP の動作	ERREVT3 の起動	各種閉塞処理
WITH STOPPING 句なし	DML がリターンします。	起動しません。	閉塞しません。
WITH STOPPING 句あり	異常終了します。	起動します。	アプリケーション属性定義（mcfaalcap）の指定によっては，アプリケーション，サービスグループ，およびサービスが閉塞することがあります。

ステータスコード

ステータスコードはありません。

SEND － アプリケーションプログラムの起動

名称

アプリケーションプログラムの起動

形式 1（起動するアプリケーションへ渡すメッセージを設定して，アプリケーション起動する場合）

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR  OUTPUT  PROGRAM
   〔STATUS  KEY  IS データ名1〕
    SYMBOLIC  TERMINAL  IS データ名2
   〔ACTIVE {INTERVAL|TIME} IS データ名9〕.

01 一意名1.
   02 データ名10  PIC  9(4) COMP.
   02 データ名11  PIC  X(2).
   02 データ名12  PIC  X(n).
01 一意名2.
   02 データ名13  PIC  X(1).
```

PROCEDURE DIVISION（通信文）の指定

```
SEND 通信記述名 FROM 一意名1
    〔WITH {ESI|EMI|一意名2}〕.
```

形式 2（最終セグメント以外のセグメントの引き渡し要求後，メッセージ引き渡しの終了を連絡する場合）

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR  OUTPUT  PROGRAM
   〔STATUS  KEY  IS データ名1〕
    SYMBOLIC  TERMINAL  IS データ名2.
```

PROCEDURE DIVISION の指定

```
SEND 通信記述名 WITH EMI.
```

機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCMCF('EXECAP ')　アプリケーションプログラムの起動

送信できるメッセージの一つのセグメント長は，32 キロバイトまでです。ただし，プロトコルによって，実際の最大長が小さいことがあります。詳細については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

通信記述項に設定する項目

●FOR 句

一方送信メッセージを示す OUTPUT を指定します。

●STATUS KEY 句

ステータスコードを受け取りたいときに指定します。この指定を省略した場合は，ステータスコードを受け取れません。

●SYMBOLIC TERMINAL 句

起動するアプリケーション名を設定したデータ項目を指定します。

●ACTIVE 句

アプリケーションプログラムをタイマ起動で起動する場合に指定します。

INTERVAL：経過時間指定のタイマ起動。

TIME：時刻指定のタイマ起動。

データ名 9：

経過時間指定のタイマ起動の場合

アプリケーションプログラムを起動する SEND 文を使ってから，何時間何分何秒後に MHP を開始させるかを設定します。時間は「HHMMSS00」（HH：時間，MM：分，SS：秒　00 は固定）の形式で設定します。設定できる時間の範囲は，'00000100'（1 秒後に開始）から '99595900'（99 時間 59 分 59 秒後に開始）までです。

時刻指定のタイマ起動の場合

MHP を開始させる時刻を設定します。時刻は「HHMMSS00」（HH：時，MM：分，SS：秒　00 は固定）の形式で設定します。設定できる時刻の範囲は，'00000000'（0 時 0 分 0 秒に開始）から '23595900'（23 時 59 分 59 秒に開始）までです。時刻はローカルタイムで指定します。

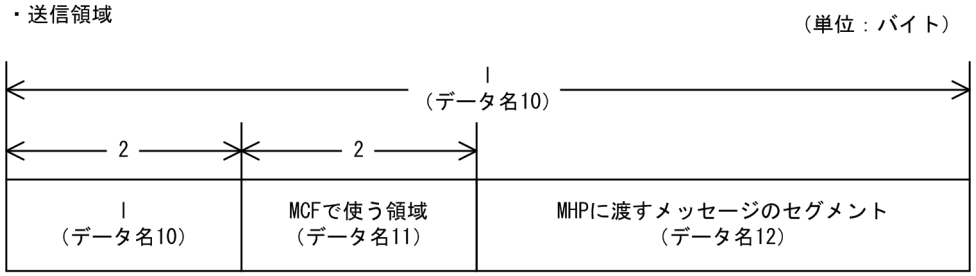
指定を省略した場合は，即時起動となります。

一定の時間間隔で起動時間に達したかどうかを監視するので，ACTIVE 句に設定した時間と実際の起動時間には誤差が生じます。時間監視の精度は，MCF 通信構成定義 mcfttim の-t オプションの btim オペランドに指定した時間監視間隔の値に依存します。

通信文に指定する項目

●一意名 1

起動するアプリケーションプログラムに渡す，メッセージセグメント送信領域を示すデータ項目を指定します。開始させる MHP に渡すセグメントの形式を次に示します。



●WITH 句

起動するアプリケーションプログラムに渡すセグメントが，論理メッセージの最終セグメントかどうかを指定します。

ESI：先頭セグメント，中間セグメントの場合。

EMI：最終セグメントの場合。渡すメッセージが単一セグメントの場合も EMI を設定します。

一意名 2：次の値を設定したデータ項目。

'1' …ESI（先頭セグメント，中間セグメント）

'2' …EMI（最終セグメント ，または単一セグメント）

指定を省略した場合は，EMI（最終セグメント，または単一セグメント）が仮定されます。

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。
71002	メッセージキューへの入出力処理時に障害が起こりました。
	メッセージキューが閉塞されています。
	メッセージキューが割り当てられていません。
	セグメント長に 32000 バイトを超える値を設定しています。
	MCF が終了処理中のため，アプリケーションプログラムを起動する SEND 文を受け付けられません。
71003	メッセージキューが満杯です。
71004	メッセージを格納するバッファをメモリ上に確保できませんでした。

ステータスコード	意味
71008	アプリケーションプログラムを起動しようとしましたが，送信先の管理テーブルが確保できませんでした。
	プロセスのローカルメモリが不足しています。
72000	＜ MHP の実行でリターンした場合＞ アプリケーションプログラムを起動する SEND 文を使った位置が間違っています。MHP で先頭セグメントを受信する前に，アプリケーションプログラムを起動する SEND 文を使っています。
	＜ SPP の実行でリターンした場合＞ トランザクションでない SPP の処理から，アプリケーションプログラムを起動する SEND 文を使っています。
72001	設定したアプリケーション名は，MCF で定義していません。
	アプリケーション名が間違っています。
	MCF マネジャ定義の通信サービス定義（mcfmcname 定義コマンド）に，アプリケーション起動プロセス名を指定していません。
	アプリケーション起動プロセスに対応する MCF アプリケーション定義の環境定義（mcfaenv 定義コマンドの-p オプション）に，アプリケーション起動プロセス識別子を指定していません。
	アプリケーション環境定義（mcfaenv 定義コマンドの-p オプション）で指定したアプリケーション起動プロセス識別子と，アプリケーション起動プロセスの通信構成定義（mcftenv 定義コマンド）で指定する識別子が一致していません。
	＜非応答型の MHP を起動する場合＞ • 起動先アプリケーションのアプリケーション属性定義の，論理端末（mcfaalcap 定義コマンドの-n オプションの lname オペランド）に値を指定していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した論理端末を，アプリケーション起動プロセスの通信構成定義（mcftalcle 定義コマンド）に定義していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した論理端末が，一方送信型（=send）ではありません。 • 起動先アプリケーションのアプリケーション属性定義で指定した論理端末は，アプリケーション起動を使えません。
	＜応答型 および継続問い合わせ応答型の MHP を起動する場合＞ • 起動先アプリケーションのアプリケーション属性定義に，内部通信路（mcfaalcap 定義コマンドの-n オプションの cname オペランド）を指定していません。 • 起動先アプリケーションのアプリケーション属性定義に指定した内部通信路を，アプリケーション起動プロセスの通信構成定義（mcftpsvr 定義コマンドの-c オプション）に定義していません。 • アプリケーション起動プロセスの通信構成定義（mcftalcle 定義コマンド）に，問い合わせ型論理端末（=request）を指定していません。
	＜ SPP からアプリケーションを起動する場合＞ • アプリケーション起動プロセス識別子を，起動元の UAP のユーザサービス定義 またはユーザサービスデフォルト定義の mcf_psv_id オペランドに指定していません。 • 起動元の UAP のユーザサービス定義 またはユーザサービスデフォルト定義の mcf_psv_id オペランドに指定しているアプリケーション起動プロセス識別子が，アプリケーション起動

ステータスコード	意味
72001	プロセスの通信構成定義（mcftenv 定義コマンドの-s オプション），およびアプリケーション環境定義（mcfaenv 定義コマンドの-p オプション）で指定しているアプリケーション起動プロセス識別子と一致していません。 • 起動元の UAP のユーザサービス定義 またはユーザサービスデフォルト定義の mcf_mgrid オペランドに指定している MCF マネジャ識別子が，アプリケーション起動プロセスが属している MCF マネジャの識別子と一致していません。
72005	WITH 句に'ESI'（先頭セグメント，または中間セグメント）を設定したアプリケーションプログラムの起動で，送信セグメント長に 5 バイト未満の値を設定しています。
72007	応答型（type = ans）の MHP で，応答メッセージを送信したあとで，応答型の MHP を起動しています。
	継続問い合わせ応答型（type = cont）の MHP で，応答メッセージを送信したあとで，継続問い合わせ応答型の MHP を起動しています。
72009	応答型（type = ans）の MHP を，2 回以上起動しています。
	継続問い合わせ応答型（type = cont）の MHP を，2 回以上起動しています。
72011	応答型（type = ans）でない MHP から，応答型の MHP を起動しています。
	継続問い合わせ応答型（type = cont）でない MHP から，継続問い合わせ応答型の MHP を起動しています。
72023	ACTIVE 句の TIME に設定したデータ名 3 の内容が空白です。
72024	FOR 句に設定した値が間違っています。
72026	WITH 句に設定した値が間違っています。
72041	＜メッセージが単一セグメントの場合＞ データ名 10 に 0 から 4 バイト，またはマイナス値を設定しています。
	＜メッセージが複数セグメントの場合＞ WITH 句に'ESI'を設定したアプリケーションプログラムを起動する SEND 文を呼び出さないで，メッセージ引き渡しの終了を連絡しています。
72044	継続問い合わせ応答を終了したあとで，次に起動するアプリケーション名を指定して，アプリケーションプログラムを起動する SEND 文を使っています。
72108	データ名 9 で指定した値は，設定できる値の範囲を超えています。
72109	継続問い合わせ応答型（type = cont）の MHP を，タイマ起動で起動しました。
77001	起動しようとするアプリケーションに対応する論理端末（LE）は，処理中で使える状態ではありません。または，そのほかの要因で使える状態ではありません。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

注意事項

1. MCF マネジャ定義の UAP 共通定義（mcfmuap -c order）の指定によって，アプリケーションプログラムの起動順序が異なりますので注意してください。

2. 一つのサービス関数で，TAM または DAM ファイルの更新とアプリケーションプログラムを起動する SEND 文を使い，起動先のアプリケーションで更新後の TAM または DAM ファイルを参照させたい場合，TAM または DAM ファイルを排他ありで参照してください。排他なしで参照した場合，更新前のデータが入力されることがあります。

SEND － 一時記憶データの更新

名称

一時記憶データの更新

形式

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR I-O STORAGE
   〔STATUS  KEY  IS データ名1〕.

01 一意名1.
   02 データ名2  PIC  9(4) COMP.
   02 データ名3  PIC  X(4).
   02 データ名4  PIC  X(n).
```

PROCEDURE DIVISION（通信文）の指定

```
SEND 通信記述名 FROM 一意名1.
```

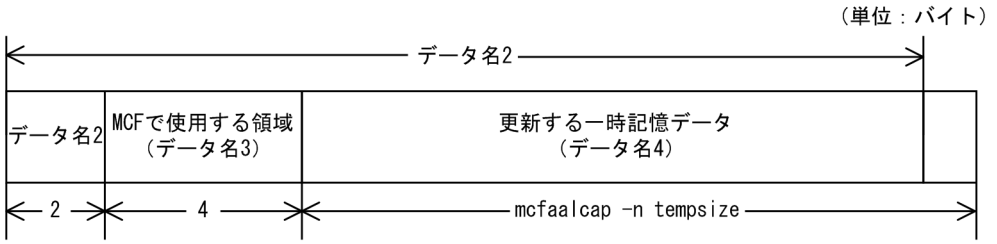
機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCMCF('TEMPPUT　') 一時記憶データの更新

アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドには，更新する一時記憶データ長（データ名 2 から 6 を減算した値）以上の値を指定してください。

更新する領域（一意名 1 で示す領域）の形式を次に示します。



通信記述項に設定する項目

●FOR 句

一時記憶データの更新であることを示す，I-O STORAGE を設定します。

●STATUS KEY 句

ステータスコードを受け取りたい場合に設定します。省略した場合は，ステータスコードを受け取れません。データ名 1 にステータスコードが返されます。

通信文に指定する項目

●データ名 2

一意名 1 の長さ（データ名 2 の長さ（2）＋データ名 3 の長さ（4）＋データ名 4 の長さ）を設定します。

●データ名 3

MCF で使用する領域です。

●データ名 4

一時記憶データが格納されている領域を設定します。

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。
71103	一時記憶データを更新するための領域をメモリ上に確保できませんでした。
72000	SPP では SEND（一時記憶データの更新）文を呼び出せません。
72024	FOR 句に設定した値が間違っています。
72035	データ名 2 に設定した更新データの長さが，アプリケーション属性定義（mcfaalcap -n）の tempsize オペランドで定義した長さを超えています。
	データ名 2 に 0 から 6 バイト，またはマイナス値を設定しています。
72042	通信記述項に SYMBOLIC TERMINAL 句の設定があり，空白以外が設定されています。
72101	継続問い合わせ応答型でないアプリケーションで，SEND（一時記憶データの更新）文を呼び出しました。
72105	RECEIVE（一時記憶データの受け取り）文を呼び出す前に，SEND（一時記憶データの更新）文を呼び出しました。
72106	先頭セグメントを受信する RECEIVE 文を呼び出す前に，SEND（一時記憶データの更新）文を呼び出しました。
72107	DISABLE（継続問い合わせ応答の終了）文を呼び出したあとで，SEND（一時記憶データの更新）文を呼び出しました。
上記以外	プログラムの破壊などによる，予期しないエラーが発生しました。

SEND － 運用コマンドの実行

名称

運用コマンドの実行

形式

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR {OUTPUT|I-O} COMMAND
   〔STATUS  KEY  IS データ名1〕.

01 一意名1.
   02 データ名2  PIC  9(4) COMP.
   02 データ名3  PIC  X(2) VALUE LOW-VALUE.
   02 データ名4  PIC  X(n).
01 一意名2.
   02 データ名5  PIC  9(4) COMP.
   02 データ名6  PIC  X(2) VALUE LOW-VALUE.
   02 データ名7  PIC  X(n).
```

PROCEDURE DIVISION（通信文）の指定

```
SEND 通信記述名 FROM 一意名1
    〔BEFORE  RECEIVING  MESSAGE  INTO 一意名2〕.
```

機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCADM('COMMAND ')　運用コマンドの実行

DML でコマンドを実行する場合，CALL インタフェースでコマンドを実行するときと，次の点が異なります。

- コマンドの処理結果のうち，標準エラー情報だけをリターンします。標準出力へ出力する文字列（例えば，表示コマンド（～ls など）の実行結果）は受け取れません。標準出力へ出力する文字列を受け取りたい場合は，CALL 文のコマンドの実行を使ってください。
- 標準出力へ出力するコマンドを実行した場合は，シェルの終了コードが 0 でも，ステータスコードは 00000 にはなりません。
- FOR 句に OUTPUT COMMAND を指定した文を使った場合に標準エラー出力へ出力データがあった場合は，標準エラー出力のデータが，領域に入り切らなかった旨のステータスコード（「01804」，または「01805」）をリターンします。

通信記述項に設定する項目

●FOR 句

次のどちらかを指定します。

OUTPUT COMMAND

コマンドの標準エラー出力を受け取らない場合に指定します。この値を指定した場合は，標準エラー出力の受け取り領域の長さには 0 が仮定されます。

I-O COMMAND

コマンドの標準エラー出力を受け取る場合に指定します。

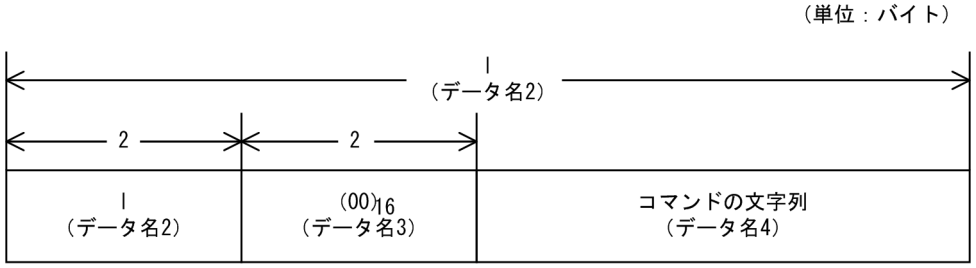
●STATUS KEY 句

ステータスコードを受け取りたいときに指定します。この指定を省略した場合は，ステータスコードを受け取れません。

通信文に指定する項目

●一意名 1

この文で実行するコマンドを設定する領域です。コマンドの文字列を設定する形式を次に示します。



●BEFORE 句

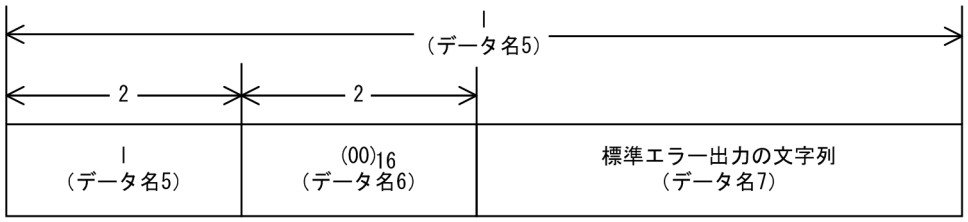
コマンドを実行した結果，標準エラー出力に出力した文字列を受け取る場合に指定します。この BEFORE 句は，FOR 句で I-O COMMAND を指定している場合に指定します。

一意名 2

標準エラー出力の文字列を格納するデータ項目。格納できる文字列の最大値は（一意名 2 の領域長）-4 です。この値を超える部分は切り捨てます。

受け取る文字列（一意名 2）の形式を次に示します。

（単位：バイト）



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。FOR 句に I-O COMMAND を指定している場合は，標準エラー出力の文字列を格納しました。
01801	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。FOR 句に I-O COMMAND を指定している場合は，標準エラー出力の文字列を格納しました。
01802	データ名に設定した値が間違っています。一意名 1，一意名 2 に設定した領域長の指定や，0 と設定する必要がある個所などに誤りがないか確認してください。
01803	標準出力への出力がありました（コマンドを実行する SEND 文では，標準出力の文字列は受け取れません）。
01804	標準エラー出力のデータが，領域に入り切りませんでした。
01805	標準出力への出力があり（コマンドを実行する SEND 文では，標準出力の文字列は受け取れません），かつ標準エラー出力のデータが，領域に入り切りませんでした。
01806	システムコール（close，pipe，dup，または read）に失敗しました。
01807	CBLDCRPC('OPEN ')を呼び出していません。
01808	メモリが不足しました。

SEND － ユーザジャーナルの取得

名称

ユーザジャーナルの取得

形式

DATA DIVISION（通信記述項）の指定

```
CD  通信記述名
    FOR OUTPUT JOURNAL
   〔STATUS  KEY  IS データ名1〕.

01 一意名1.
   02 データ名2  PIC  9(4) COMP.
   02 データ名3  PIC  X(2).
   02 データ名4  PIC  X(1).
   02 データ名5  PIC  X(n).
```

PROCEDURE DIVISION（通信文）の指定

```
SEND 通信記述名 FROM 一意名1.
```

機能

次に示す CALL インタフェースの機能を実現します。

- CBLDCJNL('UJPUT ') ユーザジャーナルの取得

通信記述項に設定する項目

●FOR 句

OUTPUT JOURNAL を指定します。

●STATUS KEY 句

ステータスコードを受け取りたいときに指定します。この指定を省略した場合は，ステータスコードを受け取れません。

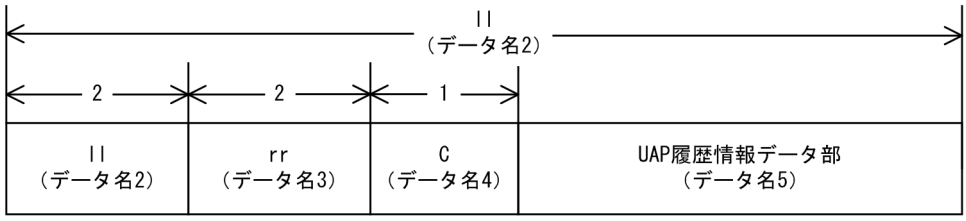
通信文に指定する項目

●一意名 1

指定した領域に UAP 履歴情報を格納します。この一意名 1 は，作業場所節，または連結節で定義した項目の名称を指定してください。

格納する UAP 履歴情報（一意名 1）の形式を次に示します。この領域は，UAP で作成・設定します。

（単位：バイト）



ll（データ名2）：UAP履歴情報長（UAP履歴情報データ部の長さ ＋ 5）
　5≦ll≦（取得先システムジャーナルサービス定義のjnl_max_datasizeオペランドの値 － 8）
rr（データ名3）：値は設定しません。
C（データ名4）：UJコードを，$(00)_{16}$から$(FF)_{16}$までの値で設定します。

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。
01101	パラメタの形式が間違っています。
01102	ユーザジャーナルの長さに，5 バイト未満の値を設定しています。
01103	ユーザジャーナルの長さに，設定できる範囲以上の値を設定しています。
01104	ユーザジャーナルを取得する SEND 文が，トランザクションでない UAP の処理で使っています。
01105	UAP 開始の準備がされていません。

4 X/Open に準拠したアプリケーションプログラミングインタフェース

この章では，X/Open で規定するアプリケーションプログラミングインタフェースの COBOL-UAP 作成用プログラミングの文法について説明します。

X/Open に準拠した API

X/Open に準拠した API（XATMI インタフェース，TX インタフェース）と機能の対応を表 4-1 に，OpenTP1 の UAP との関係を表 4-2 に示します。

表 4-1　X/Open に準拠した API と機能の対応

機能	X/Open に準拠した関数名と機能	
XATMI インタフェース	TPACALL	リクエスト／レスポンス型サービスの呼び出し
	TPADVERTISE	サービス名の広告
	TPCALL	リクエスト／レスポンス型サービスの呼び出しと応答の受信
	TPCANCEL	リクエスト／レスポンス型サービスのキャンセル
	TPCONNECT	会話型サービスとのコネクションの確立
	TPDISCON	会話型サービスとのコネクションの切断
	TPGETRPLY	リクエスト／レスポンス型サービスからの非同期応答の受信
	TPRECV	会話型サービスからのメッセージの受信
	TPRETURN	サービスルーチンからのリターン
	TPSEND	会話型サービスへのメッセージの送信
	TPSVCSTART	サービスルーチンの開始
	TPUNADVERTISE	サービス名の広告の取り消し
TX インタフェース	TXBEGIN	トランザクションの開始
	TXCLOSE	リソースマネジャ集合のクローズ
	TXCOMMIT	トランザクションのコミット
	TXINFORM	現在のトランザクションに関する情報の返却
	TXOPEN	リソースマネジャ集合のオープン
	TXROLLBACK	トランザクションのロールバック
	TXSETCOMMITRET	commit_return 特性の設定
	TXSETTIMEOUT	transaction_timeout 特性の設定
	TXSETTRANCTL	transaction_control 特性の設定

表 4–2　X/Open に準拠した API と OpenTP1 の UAP との関係

X/Open に準拠した API	SUP		SPP			MHP		オフラインの業務をする UAP
	Trn の処理の範囲でない	Trn の処理範囲	Trn の処理の範囲でない	Trn の範囲		Trn の処理の範囲でない	Trn の処理範囲	
		root		root	root 以外		root	
TPACALL	○	○	○	○	○	−	−	−
TPADVERTISE	−	−	○※1	○※1	○※1	−	−	−
TPCALL	○	○	○	○	○	−	−	−
TPCANCEL	○	○	○	○	○	−	−	−
TPCONNECT	○	○	○	○	○	−	−	−
TPDISCON	○	○	○	○	○	−	−	−
TPGETRPLY	○	○	○	○	○	−	−	−
TPRECV	○	○	○	○	○	−	−	−
TPRETURN	−	−	○※2	○※2	○※2	−	−	−
TPSEND	○	○	○	○	○	−	−	−
TPSVCSTART※3	−	−	−	−	−	−	−	−
TPUNADVERTISE	−	−	○※1	○※1	○※1	−	−	−
TXBEGIN※4	○	−	○	−	−	○	−	−
TXCLOSE	○	−	○	−	−	−	−	−
TXCOMMIT TX-CHAINED 指定※4	−	○	○	−	−	−	−	−
TXCOMMIT TX-UNCHAINED 指定※4	−	○	○	−	−	−	−	−
TXINFORM	○	○	○	○	○	−	−	−
TXOPEN	○	−	○	−	−	−	−	−
TXROLLBACK TX-CHAINED 指定※4	−	○	−	○	−	−	−	−
TXROLLBACK TX-UNCHAINED 指定※4	−	○	−	○	−	−	−	−
TXSETCOMMITRET※4	○	○	○	○	○	−	−	−
TXSETTIMEOUT※4	○	○	○	○	○	−	−	−
TXSETTRANCTL※4	○	○	○	○	○	−	−	−

（凡例）

Trn：トランザクション

root：ルート

○：該当する条件で呼び出せます。

－：該当する条件では呼び出せません。

注

MHP の「トランザクションの処理の範囲でない」とは，非トランザクション属性の MHP，または MHP のメインプログラムの範囲を示します。

注※1

サービスプログラムの中でだけ，呼び出せます。

注※2

XATMI インタフェースのサービスプログラムをリターンするためだけに使用します。

注※3

TPSVCSTART は，サービスプログラムで必ず呼び出す API です。

注※4

この API を呼び出す UAP は，ユーザサービス定義で atomic_update=Y を指定してください。

XATMI インタフェースのアプリケーションプログラミングインタフェース (TP～)

XATMI インタフェースの API の文法について説明します。この節の記述は，X/Open 発行の「X/Open CAE Specification Distributed TP：The XATMI Specification」の文法部である「Chapter 7 COBOL Language Reference Manual Pages」の記述を，日本語訳したものです。

なお，OpenTP1 の UAP で XATMI インタフェースの API を使うときに注意する項目として追加した文章は，『 』で示します。

XATMI インタフェースの API を次に示します（TPINTRO は，API から COPY 文で呼び出すデータ領域の定義を示します)。

- TPINTRO － XATMI インタフェース用の COPY ファイル
- TPACALL － リクエスト／レスポンス型サービスの呼び出し
- TPADVERTISE － サービス名の広告
- TPCALL － リクエスト／レスポンス型サービスの呼び出しと応答の受信
- TPCANCEL － リクエスト／レスポンス型サービスのキャンセル
- TPCONNECT － 会話型サービスとのコネクションの確立
- TPDISCON － 会話型サービスとのコネクションの切断
- TPGETRPLY － リクエスト／レスポンス型サービスからの非同期応答の受信
- TPRECV － 会話型サービスからのメッセージの受信
- TPRETURN － サービスルーチンからのリターン
- TPSEND － 会話型サービスへのメッセージの送信
- TPSVCSTART － サービスルーチンの開始
- TPUNADVERTISE － サービス名の広告の取り消し

XATMI インタフェースの API（TP～）は，TP1/Server Base の場合にだけ使えます。TP1/LiNK では，XATMI インタフェースの API は使えません。

TPINTRO

名称

XATMI インタフェース用の COPY ファイル

説明

次に示すリターンコードと領域の定義は，COBOL XATMI ルーチンで使います。XATMI インタフェースを使う場合には，次に示す 4 件の COPY ファイルに領域の定義をする必要があります。それぞれの COPY ファイルは，必要最小限のレコードを記述しています。

『以降の COBOL レコードは，$DCDIR/include/COBOL/ディレクトリに格納されています。UAP をコンパイルするときには，COPY ファイルの存在場所としてこのディレクトリを指定する必要があります。コンパイル時の指定については，使用する COBOL 言語のマニュアルを参照してください。』

```
*
* TPSTATUS.cbl
*
 05 TP-STATUS PIC S9(9) COMP-5.
          88 TPOK           VALUE 0.
          88 TPEBADDESC     VALUE 2.
          88 TPEBLOCK       VALUE 3.
          88 TPEINVAL       VALUE 4.
          88 TPELIMIT       VALUE 5.
          88 TPENOENT       VALUE 6.
          88 TPEOS          VALUE 7.
          88 TPEPROTO       VALUE 9.
          88 TPESVCERR      VALUE 10.
          88 TPESVCFAIL     VALUE 11.
          88 TPESYSTEM      VALUE 12.
          88 TPETIME        VALUE 13.
          88 TPETRAN        VALUE 14.
          88 TPEGOTSIG      VALUE 15.
          88 TPEITYPE       VALUE 17.
          88 TPEOTYPE       VALUE 18.
          88 TPEEVENT       VALUE 22.
          88 TPEMATCH       VALUE 23.
 05 TPEVENT PIC S9(9) COMP-5.
          88 TPEV-NOEVENT   VALUE 0.
          88 TPEV-DISCONIMM VALUE 1.
          88 TPEV-SENDONLY  VALUE 2.
          88 TPEV-SVCERR    VALUE 3.
          88 TPEV-SVCFAIL   VALUE 4.
          88 TPEV-SVCSUCC   VALUE 5.
 05 APPL-RETURN-CODE PIC S9(9) COMP-5.
```

次に示す COBOL レコードは，アプリケーションがデータを送受信するときには常に使われます。REC-TYPE には，送信されるデータレコードの型が含まれます。SUB-TYPE には，特定の型の subtype の名称が含まれます。LEN には，送信するデータの長さ，および受信するデータの長さが含まれます。

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

次に示す COBOL レコードは，コミュニケーションリソースマネジャから設定を渡されるため，またはコミュニケーションリソースマネジャへ設定を渡すために使われます。

```
*
* TPSVCDEF.cbl
*
 05 COMM-HANDLE         PIC S9(9) COMP-5.
 05 TPBLOCK-FLAG        PIC S9(9) COMP-5.
          88 TPBLOCK        VALUE 0.
          88 TPNOBLOCK      VALUE 1.
 05 TPTRAN-FLAG         PIC S9(9) COMP-5.
          88 TPTRAN         VALUE 0.
          88 TPNOTRAN       VALUE 1.
 05 TPREPLY-FLAG        PIC S9(9) COMP-5.
          88 TPREPLY        VALUE 0.
          88 TPNOREPLY      VALUE 1.
 05 TPTIME-FLAG         PIC S9(9) COMP-5.
          88 TPTIME         VALUE 0.
          88 TPNOTIME       VALUE 1.
 05 TPSIGRSTRT-FLAG     PIC S9(9) COMP-5.
          88 TPNOSIGRSTRT   VALUE 0.
          88 TPSIGRSTRT     VALUE 1.
 05 TPGETANY-FLAG       PIC S9(9) COMP-5.
          88 TPGETHANDLE    VALUE 0.
          88 TPGETANY       VALUE 1.
 05 TPSENDRECV-FLAG     PIC S9(9) COMP-5.
          88 TPSENDONLY     VALUE 0.
          88 TPRECVONLY     VALUE 1.
 05 TPNOCHANGE-FLAG     PIC S9(9) COMP-5.
          88 TPCHANGE       VALUE 0.
          88 TPNOCHANGE     VALUE 1.
 05 TPSERVICETYPE-FLAG PIC S9(9) COMP-5.
          88 TPREQRSP       VALUE 0.
          88 TPCONV         VALUE 1.
 05 SERVICE-NAME        PIC X(15).
```

次に示す COBOL レコードは，TPRETURN がトランザクションの状態を知らせるために使われます。

```
*
* TPSVCRET.cbl
*
 05 TP-RETURN-VAL PIC S9(9) COMP-5.
```

```
          88 TPSUCCESS   VALUE 0.
          88 TPFAIL      VALUE 1.
05 APPL-CODE     PIC S9(9) COMP-5.
```

TPACALL

名称

リクエスト／レスポンス型サービスの呼び出し

形式

```
01 TPSVCDEF-REC.
   COPY TPSVCDEF.
01 TPTYPE-REC.
   COPY TPTYPE.
01 DATA-REC.
   COPY Data record definition.
01 TPSTATUS-REC.
   COPY TPSTATUS.

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

機能

TPACALL は，SERVICE-NAME で示すサービスにサービス要求のメッセージを送信します。DATA-REC は，送信するレコードです。LEN には，送信するデータである DATA-REC の長さを設定します。

Note：DATA-REC が長さを指定する必要がないレコード型の場合は，LEN は無視されます（0 にしてください)。DATA-REC が長さを指定する必要があるレコード型の場合は，LEN には 0 を設定しないでください。REC-TYPE に subtype がないレコード型の場合は，SUB-TYPE は無視されます（SPACES（空白）にしてください)。REC-TYPE が SPACES（空白）の場合は，DATA-REC と LEN は無視されて，サービス要求はデータ部がないデータで送信されます。REC-TYPE と SUB-TYPE は，SERVICE-NAME で示すサービスで認識している type と subtype のうちのどれか一つに必ず一致させてください。

Note：トランザクションモードから送信した，それぞれのサービス要求に対する応答は，最終的には必ず受信するようにしてください。

『データ領域』

『●TPSVCDEF-REC

TPACALL の動作を示す値を設定します。ここに設定した値によって，リターンする値が決まります。設定する値と意味は，以降説明します。』

『●TPTYPE-REC

送信データのレコード型と，subtype のレコード名を示します。』

『●DATA-REC

送信データを示します。』

『●TPSTATUS-REC

TPACALL の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPNOTRAN

呼び出し側がトランザクションモードで，かつ　この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部にはなりません。呼び出し側がトランザクションモードで，かつ SERVICE-NAME がトランザクション処理ができないサーバに属している場合は，この値を必ず設定してください。呼び出し側がトランザクションモードであれば，この値が設定されていても，トランザクションタイムアウトは起こります（トランザクションモードでなければ起こりません)。この値で起動されたサービスが失敗しても，呼び出し側のトランザクションには影響しません。
TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPTRAN

呼び出し側がトランザクションモードで，かつ，この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部になります。呼び出し側がトランザクションモードでない場合は，この値を設定しても無視されます。TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPNOREPLY

サービス要求に対する応答が不要であることを，TPACALL に設定します。TPNOREPLY を設定すると，TPACALL の成功時には TPOK をリターンして COMM-HANDLE に 0 を設定します。この 0 は，通信ハンドルとしては意味を持ちません。呼び出し側がトランザクションモードの場合は，TPTRAN と一緒に設定してもこの値は使えません。TPNOREPLY または TPREPLY のどちらかを，必ず設定してください。

TPREPLY

サービス要求に対する応答が必要であることを，TPACALL に設定します。TPREPLY を設定すると，TPACALL の成功時には TPOK をリターンして COMM-HANDLE に通信ハンドルとして有効な値を設定します。呼び出し側がトランザクションモードの場合は，TPTRAN と一緒に設定してください。
TPNOREPLY または TPREPLY のどちらかを，必ず設定してください。

TPNOBLOCK

ブロッキング状態の場合（例えば，送信しようとするメッセージで内部バッファが満杯)，サービス要求は送信されません。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定していて，ブロッキング状態のときは，呼び出し側は状況が収まるか，タイムアウト（トランザクション，またはブロッキングタイムアウト）が起こるまで，ブロッキングしています。
TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

ブロッキング状態になった場合，ブロッキングタイムアウトの時間に達したときにブロッキングタイムアウトの通知を受け取ることを意味します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

リターン値

成功した場合，TPACALL は TP-STATUS に TPOK を設定します。さらに，TPSVCDEF-REC に TPREPLY を設定している場合は，TPACALL は COMM-HANDLE に有効な通信ハンドルをリターンします。この通信ハンドルは，サービス要求の応答を受信するために使います。

エラー

次のような場合，TPACALL はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。特に示さないかぎり，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPENOENT

SERVICE-NAME で示すサービスが存在しないので，送信できません。

TPEITYPE

REC-TYPE と SUB-TYPE の組み合わせが，SERVICE-NAME で使える形式ではありません。

TPELIMIT

完了していない非同期の送信要求が最大数に達したので，呼び出し側の要求は送信されません。

TPETRAN

SERVICE-NAME がトランザクション処理ができないサーバに属しているのに，TPTRAN を設定しています。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合は，これはトランザクションタイムアウトで，トランザクションは rollback_only 状態となります。そうでない場合は，TPNOBLOCK も TPNOTIME も設定されていない状態でブロッキングタイムアウトが起こったことを意味します。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，新しく送信しようとしていた送信や，完了していない応答は，TPETIME でエラーリターンします。

TPEBLOCK

TPNOBLOCK を設定した TPACALL で送信したときに，ブロッキング状態になりました。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPACALL が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPCALL，TPCANCEL，TPGETRPLY

『OpenTP1 で使う場合の注意事項』

1. 『該当バージョンの OpenTP1 では，TPNOBLOCK の設定は無効となります。そのため，TPEBLOCK が TP-STATUS に返ることはありません。OpenTP1 では，ブロッキング状態のため通信ができない場合は，ネットワークダウンが原因で通信ができない場合と同様に，TPESYSTEM をリターンする仕様となっています。』
2. 『該当バージョンの OpenTP1 では，TPNOTIME の設定は無効となります。』
3. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
4. 『該当バージョンの OpenTP1 では，TPEITYPE はリターンされません。SERVICE-NAME で使えない型のレコードを渡した場合，TPACALL は正常リターンしますが，TPGETRPLY を呼び出した時点で TPESYSTEM または TPESVCERR がリターンされて，エラーであることがわかります。もし呼び出し側がトランザクションモードであれば，rollback_only 状態となります。』
5. 『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』
6. 『該当バージョンの OpenTP1 では，ロールバックする必要があるレコードは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』
7. 『該当バージョンの OpenTP1 では，TPELIMIT はリターンされません。』
8. 『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合，送信するデータの長さは NET/Library 共通定義の　NET バッファグループ定義　nettbuf にある length オペランドに指定した値を超えないようにしてください。』

TPADVERTISE

名称

サービス名の広告

形式

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

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

機能

TPADVERTISE は，サーバで提供するサービスを広告します。TPADVERTISE を実行しない場合，サーバのサービスは，ブート時に広告されて，シャットダウンされたときに広告が取り消されます。

TPADVERTISE は，サーバの SERVICE-NAME を広告します。SERVICE-NAME は，15 文字以下にしてください。ただし，SPACES（空白）は使えません。長過ぎる名称でも有効となりますが，15 文字に切り詰められます。切り詰められた名称が ほかのサービス名と一致しないようにしてください。

PROGRAM-NAME は，サービスプログラムの名称です。このプログラムは，SERVICE-NAME へのサービス要求がサーバに受信されたら，いつでも起動されます。PROGRAM-NAME には，SPACES（空白）を設定しないでください。

SERVICE-NAME がすでにそのサーバで広告されていて，PROGRAM-NAME が広告されているプログラムと一致した場合は，TPADVERTISE は正常にリターンします（切り詰められた名称が，すでに広告されていた名称と一致していた場合も含みます）。しかし，その SERVICE-NAME がすでにそのサーバで広告されていた場合でも，PROGRAM-NAME が広告されているプログラムと一致しないときは，エラーリターンします（切り詰められた名称が，すでに広告されている名称と一致していた場合にも起こります）。

『データ領域』

『●SERVICE-NAME

応答するサービスのサービス名を設定します。』

『●PROGRAM-NAME

サービスプログラム名を設定します。』

『●TPSTATUS-REC

TPADVERTISE の実行結果を示すリターン値が設定されます。』

リターン値

成功した場合，TPADVERTISE は TP-STATUS に TPOK を設定します。

エラー

次のような場合，TPADVERTISE はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

TPEINVAL

SERVICE-NAME か PROGRAM-NAME のどちらかが SPACES（空白）になっています。または PROGRAM-NAME に設定した値が有効な名称ではありません。

TPELIMIT

領域の制限で，SERVICE-NAME を広告できません。

TPEMATCH

SERVICE-NAME はすでにそのサーバで広告されていますが，PROGRAM-NAME のプログラムではありません。TPADVERTISE が失敗しても，SERVICE-NAME は，現在広告しているプログラムを広告したままです（つまり，PROGRAM-NAME は現在広告されているプログラムと置き換わりません）。

TPEPROTO

TPADVERTISE が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPSVCSTART，TPUNADVERTISE

『OpenTP1 で使う場合の注意事項』

1. 『TPADVERTISE は，SPP でだけ呼び出せます。サーバの起動時には，ユーザサービス定義で指定した，すべてのサービスが自動的に広告されます。このプログラムのユーザサービス定義で指定してあるサービス名とプログラムの組み合わせだけ広告できます。』
2. 『OpenTP1 では，TPADVERTISE を呼び出す UAP のサービスグループと，広告されているサービスを広告した UAP のサービスグループが同じである場合は，広告済みと見なして，正常リターンします。サービスグループが一致していない場合は，エラーリターンします。』
3. 『OpenTP1 では，PROGRAM-NAME は 20 文字まで有効です。』

TPCALL

名称

リクエスト／レスポンス型サービスの呼び出しと応答の受信

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 ITPTYPE-REC.
   COPY  TPTYPE.
01 IDATA-REC.
   COPY  Data record definition.
01 OTPTYPE-REC.
   COPY  TPTYPE.
01 ODATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPCALL は，サービス要求を送信して，同期的に応答を待ちます。TPCALL の実行は，TPACALL を実行して，すぐに TPGETRPLY を実行するのと同じことです。TPCALL は SERVICE-NAME が示すサービスにサービス要求を送信します。サービス要求のデータ部は，IDATA-REC で設定されています。ITPTYPE-REC の LEN には，IDATA-REC をどれだけ送信するかを設定します。

Note：IDATA-REC が，長さを指定する必要がないレコード型を指す場合は，ITPTYPE-REC の LEN は無視されます（0 にしてください）。IDATA-REC が，長さを指定する必要があるレコード型を指す場合は，ITPTYPE-REC の LEN には 0 は設定しないでください。ITPTYPE-REC の REC-TYPE に subtype がない場合は，ITPTYPE-REC の SUB-TYPE は無視されます（SPACES（空白）を設定してください）。ITPTYPE-REC の REC-TYPE が SPACES（空白）の場合は，IDATA-REC（ITPTYPE-REC の）と LEN は無視されて，サービス要求はデータ部がないデータで送信されます。ITPTYPE-REC の REC-TYPE と ITPTYPE-REC の SUB-TYPE は，SERVICE-NAME で示すサービスで認識している type と subtype のうちのどれか一つに必ず一致させてください。

ODATA-REC は応答が読み込まれる応答がどこに格納するかを設定しています。データを入力するときには，OTPTYPE-REC の LEN は，ODATA-REC の中に格納する必要があるバイトの最大数を示します。送信と受信に同じレコードを使う場合は，ODATA-REC を IDATA-REC に REDEFINES（再定義）する必要があります。TPCALL が正常に終了したあとで，OTPTYPE-REC の LEN は，ODATA-REC に実際に格納したバイト数が設定されます。OTPTYPE-REC の REC-TYPE と OTPTYPE-REC の SUB-TYPE には，応答の type と subtype をそれぞれ格納しています。ODATA-REC の値よりも応答が大きい場合は，ODATA-REC にはレコードに入るだけのバイトが格納されます。応答の残りの部分は捨てられて，TPCALL は TPTRUNCATE を返します。

TPCALL が正常に終了した場合に OTPTYPE-REC の LEN が 0 であるときは，応答にはデータ部がなく，ODATA-REC は変更されません。入力時に OTPTYPE-REC が 0 の場合は，エラーになります。

『データ領域』

『●TPSVCDEF-REC

TPCALL の動作を示す値を設定します。設定する値と意味は，以降説明します。』

『●ITPTYPE-REC

送信データのレコード型と，subtype のレコード名を示します。』

『●IDATA-REC

送信データを示します。』

『●OTPTYPE-REC

受信データのレコード型，subtype のレコード名を示します。』

『●ODATA-REC

受信データを示します。』

『●TPSTATUS-REC

TPCALL の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPNOTRAN

呼び出し側がトランザクションモードで，かつ この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部にはなりません。呼び出し側がトランザクションモードで，かつ SERVICE-NAME がトランザクション処理ができないサーバに属している場合は，この値を必ず設定してください。呼び出し側がトランザクションモードであれば，この値が設定されていても，トランザクションタイムアウトは起こります（トランザクションモードでなければ起こりません)。この値を設定して起動されたサービスが失敗しても，呼び出し側のトランザクションには影響しません。TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPTRAN

呼び出し側がトランザクションモードで，かつ この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部になります。呼び出し側がトランザクションモードでない場合は，この値を設定しても無視されます。TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPNOCHANGE

この設定をする場合は，ODATA-REC の型は変換されません。応答レコードの type と subtype は，OTPTYPE-REC の REC-TYPE と OTPTYPE-REC の SUB-TYPE にそれぞれ一致させてください。TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPCHANGE

受信側が受け取ったレコードのタイプを認識できる場合に，応答レコードの type と subtype が OTPTYPE-REC の REC-TYPE と OTPTYPE-REC の SUB-TYPE と異なっていても受け付けることを示します。TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPNOBLOCK

ブロッキング状態の場合（例えば，送信しようとするメッセージで内部バッファが満杯），サービス要求は送信されません。

Note：これは TPCALL でデータを送信する場合にだけ起こります。ルーチンは応答を待つためにブロッキングします。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定していて，ブロッキング状態のときは，呼び出し側は状況が収まるか，タイムアウト（トランザクションタイムアウト，またはブロッキングタイムアウト）が起こるまで，ブロッキングします。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

ブロッキング状態になった場合に，ブロッキングタイムアウトの時間に達したときにブロッキングタイムアウトの通知を受け取ることを意味します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。
TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

リターン値

成功した場合，TPCALL は TP-STATUS に TPOK を設定します。TP-STATUS に TPOK か TPESVCFAIL が返された場合は，APPL-RETURN-CODE には TPRETURN の処理でアプリケーションが定義した値が返されます。TPCALL が受信したデータの長さが OTPTYPE-REC の LEN に設定した長さを超えた場合は，OTPTYPE-REC に TPTRUNCATE が設定され，OTPTYPE-REC の LEN で示す長さ分だけ，ODATA-REC に格納されます。受信したデータの残りの部分は捨てられます。

エラー

次のような場合，TPCALL はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。特に示さないかぎり，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPENOENT

SERVICE-NAME で示すサービスが存在しないので，送信できません。

TPEITYPE

ITPTYPE-REC の REC-TYPE と ITPTYPE-REC の SUB-TYPE の組み合わせが，SERVICE-NAME の type と subtype で使える形式ではありません。

TPEOTYPE

応答の type と subtype を，呼び出し側で認識できません。または，TPNOCHANGE を設定しているのに，OTPTYPE-REC の REC-TYPE と OTPTYPE-REC の SUB-TYPE が，サービスから送信されてきた応答の type と subtype に一致していません。このエラーが起こった場合，ODATA-REC も OTPTYPE-REC も変更されません。呼び出し側のトランザクションとしてサービス要求が実行されている場合は，応答を捨てるまで，トランザクションは rollback_only 状態となります。

TPETRAN

SERVICE-NAME がトランザクション処理ができないサーバに属しているのに，TPTRAN を設定しています。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合は，これはトランザクションタイムアウトで，トランザクションは rollback_only 状態となります。そうでない場合は，TPBLOCK と TPTIME が設定された状態でブロッキングタイムアウトが起こったことを意味します。どちらの場合も，ODATA-REC も OTPTYPE-REC も変更されません。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，新しく送信しようとしていた送信や，完了していない応答は，TPETIME でエラーリターンします。

TPESVCFAIL

呼び出し側の応答を送信するサービスルーチンが，TPFAIL を設定した TPRETURN を呼び出しました。これは，アプリケーションレベルの失敗です。送信されたサービスの応答の内容は，ODATA-REC として有効になります。サービス要求が呼び出し側の現在のトランザクションとして実行されている場合は，トランザクションは rollback_only 状態となります。

注

トランザクションタイムアウトにならないかぎり，これ以降の通信は，ロールバックするまでの処理が実行されます。この処理は正常になるか，または無効になります（エラーリターンまたはイベントが通知されます）。呼び出し側のトランザクションとして実行された処理は，トランザクションが完了したときにロールバックされます。継続する処理を有効にする通信には，TPNOTRAN を設定してください。トランザクション機能によって呼び出し側のトランザクションをロールバックする何らかの処理が実行されます。

TPESVCERR

このエラーは，サービスルーチンを呼び出しているときか，TPRETURN で完了するときに起こります（例えば，適切でない引数を設定した）。このエラーが起こったときは，応答のデータはリターンさ

れません（つまり，ODATA-REC も OTPTYPE-REC も変わっていません）。サービスに応答する処理が呼び出し側のトランザクションとして実行された場合は，トランザクションは rollback_only 状態になります。

注

トランザクションタイムアウトにならないかぎり，これ以降の通信は，ロールバックするまでの処理が実行されます。この処理は正常になるか，または無効になります（エラーリターンまたはイベントが通知されます）。継続する処理を有効にする通信には，TPNOTRAN を設定してください。トランザクション機能によって，呼び出し側のトランザクションをロールバックする何らかの処理が実行されます。

TPEBLOCK

TPNOBLOCK を設定した TPCALL で送信したときに，ブロッキング状態になりました。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPCALL が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPACALL，TPGETRPLY，TPRETURN

『OpenTP1 で使う場合の注意事項』

1. 『該当バージョンの OpenTP1 では，TPNOBLOCK の設定は無効となります。そのため，TPEBLOCK が TP-STATUS にリターンすることはありません。OpenTP1 では，ブロッキング状態のため通信ができない場合は，ネットワークダウンが原因で通信ができない場合と同様に，TPESYSTEM をリターンする仕様となっています。』
2. 『該当バージョンの OpenTP1 では，TPNOTIME の設定は応答受信時にだけ有効です。サービス要求の送信時にブロッキング状態が起こった場合は，TPNOTIME の設定は無効です。』
3. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
4. 『該当バージョンの OpenTP1 では，TPEITYPE はリターンされません。SERVICE-NAME で使えない型のレコードを渡した場合は，TPESYSTEM がリターンされます。呼び出し側がトランザクションモードの場合は，rollback_only 状態となります。』

5.『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』

6.『該当バージョンの OpenTP1 では，ロールバックする必要があるエラーは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』

7.『サービス要求先の SPP が異常終了した場合，定義の watch_time に指定した時間よりも早く TPETIME でリターンするときがあります。また，watch_time に 0（応答を無限に待つ）を指定している場合は，TPEPROTO でエラーリターンするときがあります。』

8.『OpenTP1 のセキュリティ機能を使っている場合で，サービス要求が認証されなかったときは，TPEPROTO でエラーリターンします。サービス要求が認証されなかったことが原因かどうかは，UAP トレースの詳細情報で確認してください。』

9.『TP1/NET/OSI-TP-Extended を使った OSI TP 通信では，回線障害は TPESVCERR でエラーリターンします。』

10.『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合，送受信するデータの長さは NET/Library 共通定義の NET バッファグループ定義 nettbuf にある length オペランドに指定した値を超えないようにしてください。』

TPCANCEL

名称

リクエスト／レスポンス型サービスのキャンセル

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPCANCEL は，TPACALL で返された通信ハンドル COMM-HANDLE をキャンセルします。グローバルトランザクションに属している通信ハンドルをキャンセルしようとすると，エラーになります。

成功した場合，COMM-HANDLE は無効となって，COMM-HANDLE として受信した応答は，暗黙的に捨てられます（コミュニケーションリソースマネジャによって)。

『データ領域』

『●COMM-HANDLE

キャンセルする通信ハンドルを設定します。』

『●TPSTATUS-REC

TPCANCEL の実行結果を示すリターン値が設定されます。』

リターン値

成功した場合，TPCANCEL は TP-STATUS に TPOK を設定します。

エラー

次のような場合，TPCANCEL はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

TPEBADDESC
　COMM-HANDLE は，間違った通信ハンドルです。

TPETRAN
　COMM-HANDLE が，呼び出し側のグローバルトランザクションに参加しています。エラーのあとでも，引き続き COMM-HANDLE は有効で，呼び出し側の現在のトランザクションには影響しません。

TPEPROTO

TPCANCEL が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPACALL

TPCONNECT

名称

会話型サービスとのコネクションの確立

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPCONNECT は，会話型サービスである SERVICE-NAME に，半二重型の通信路を設定します。

コネクションを確立する処理の一部として，呼び出し側は受信側のサービスプログラムへ，アプリケーションが定義したデータを渡せます。呼び出し側がデータを渡す場合は，DATA-REC はデータを含んでいて，LEN にはどれぐらいのデータを送信するかを設定します。

注

DATA-REC が，長さを指定する必要がないレコードを指す場合は，LEN は無視されます（0 にしてください)。DATA-REC が，長さを指定する必要があるレコードを指す場合は，LEN には 0 を設定しないでください。REC-TYPE に subtype がない場合は，SUB-TYPE は無視されます（SPACES（空白）にしてください)。REC-TYPE が SPACES（空白）の場合は，DATA-REC と LEN は無視されます（会話型サービスにはアプリケーションデータは送信されません)。DATA-REC の REC-TYPE と SUB-TYPE は，SERVICE-NAME で認識する type と subtype のうちのどれか一つに必ず一致させてください。

TPSVCSTART が成功してリターンした会話型サービスは，DATA-REC と LEN を TPSVCSTART で受信します。そのため会話型サービスは，TPCONNECT から送られたデータを受信するために TPRECV を呼ぶ必要はありません。

『データ領域』

『●TPSVCDEF-REC

TPCONNECT の動作を示す値を設定します。ここに設定した値によって，リターンする値が決まります。設定する値と意味は，以降説明します。』

『●TPTYPE-REC

送信データのデータ型と，subtype のレコード名を示します。』

『●DATA-REC

送信データを示します。』

『● TPSTATUS-REC

TPCONNECT の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPNOTRAN

呼び出し側がトランザクションモードで，かつ この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部にはなりません。呼び出し側がトランザクションモードで，かつ SERVICE-NAME がトランザクション処理ができないサーバに属している場合は，この値を必ず設定してください。呼び出し側がトランザクションモードであれば，この値が設定されていても，トランザクションタイムアウトは起こります（トランザクションモードでなければ起こりません）。この値を設定して起動されたサービスが失敗しても，呼び出し側のトランザクションには影響しません。TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPTRAN

呼び出し側がトランザクションモードで，かつ この値を設定している場合，起動された SERVICE-NAME は，呼び出し側のトランザクションの一部になります。呼び出し側がトランザクションモードでない場合は，この値を設定しても無視されます。TPNOTRAN または TPTRAN のどちらかを，必ず設定してください。

TPSENDONLY

呼び出し側がデータを送信して，呼ばれたサービスがデータの受信だけできるように，コネクションを最初に確立します（つまり，呼び出し側が最初にコネクションの制御権を得ます）。TPSENDONLY または TPRECVONLY のどちらかを，必ず設定してください。

TPRECVONLY

呼び出し側はデータを受信するだけで，呼ばれたサービスがデータの送信だけできるように，コネクションを最初に確立します（つまり，呼び出されたサービス側が最初にコネクションの制御権を得ます）。TPSENDONLY または TPRECVONLY のどちらかを，必ず設定してください。

TPNOBLOCK

ブロッキング状態の場合（例えば，送信されたメッセージで内部バッファが満杯），コネクションは確立されないで，データは送信されません。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定していて，ブロッキング状態のときは，呼び出し側は状況が収まるか，タイムアウト（トランザクション，またはブロッキングタイムアウト）が起こるまで，ブロッキングしています。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

ブロッキング状態になった場合に，ブロッキングタイムアウトの時間に達したときにブロッキングタイムアウトの通知を受け取ることを意味します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。
TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

リターン値

成功した場合，TPCONNECT は TP-STATUS に TPOK を設定します。さらに，COMM-HANDLE に有効な通信ハンドルをリターンします。この通信ハンドルは，次の呼び出しがコネクションを参照するために使います。

エラー

次のような場合，TPCONNECT はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。特に示さないかぎり，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPENOENT

SERVICE-NAME で示すサービスが存在しないので，コネクションを確立できません。

TPEITYPE

DATA-REC の type と subtype が，SERVICE-NAME で使える形式ではありません。

TPELIMIT

未解決のコネクションが最大数に達したので，呼び出し側の要求は送信されません。

TPETRAN

SERVICE-NAME がトランザクション処理ができないサーバに属しているのに，TPTRAN を設定しています。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合，これはトランザクションタイムアウトで，トランザクションは rollback_only 状態になります。そうでない場合は，

TPNOBLOCK も TPNOTIME も設定されていない状態で，ブロッキングタイムアウトが起こったことを意味します。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，どのコネクションでのどのような送信，受信の試みも，TPETIME でエラーリターンします。

TPEBLOCK

TPNOBLOCK を設定した TPCONNECT を実行したときに，ブロッキング状態になりました。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPCONNECT が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPDISCON，TPRECV，TPSEND，TPSVCSTART

『OpenTP1 で使う場合の注意事項』

1. 『該当バージョンの OpenTP1 では，TPNOBLOCK の設定は無効となります。そのため，TPEBLOCK が TP-STATUS にリターンすることはありません。OpenTP1 では，ブロッキング状態のため通信ができない場合は，ネットワークダウンが原因で通信ができない場合と同様に，TPESYSTEM をリターンする仕様となっています。』
2. 『該当バージョンの OpenTP1 では，TPNOTIME の設定は無効となります。』
3. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
4. 『該当バージョンの OpenTP1 では，TPEITYPE はリターンされません。SERVICE-NAME で使えない型のレコードを渡した場合，TPESYSTEM または TPESVCERR がリターンされます。呼び出し側がトランザクションモードの場合は，rollback_only 状態となります。』
5. 『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』
6. 『該当バージョンの OpenTP1 では，ロールバックする必要があるエラーは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』

7.『OpenTP1 のセキュリティ機能を使っている場合で，サービス要求が認証されなかったときは，TPEPROTO でエラーリターンします。サービス要求が認証されなかったことが原因かどうかは，UAP トレースの詳細情報で確認してください。』

8.『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合は，会話型サービスの通信はできません。OSI TP 通信で会話型サービスの通信を使った場合，システムの動作は保証しません。』

9.『サーバ AP が閉塞状態の場合は，ローカルノードの要求先 SPP が閉塞しているか，リモートノードの要求先 SPP が閉塞しているかによって，次のように動作します。

ローカルノードの要求先 SPP が閉塞しているとき

tpconnect()は-1 を返し，tperrno に TPEPROTO が設定されます。

リモートノードの要求先 SPP が閉塞しているとき

トランザクションモードであれば，トランザクションタイムアウトでサーバ AP が異常終了します。

非トランザクションモードであれば，tpconnect()は-1 を返し，tperrno に TPETIME が設定されます。』

TPDISCON

名称

会話型サービスとのコネクションの切断

形式

```
01 TPSVCDEF-REC.
   COPY TPSVCDEF.
01 TPSTATUS-REC.
   COPY TPSTATUS.

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

機能

TPDISCON は，COMM-HANDLE で示すコネクションをすぐに切断して，他端点のコネクションに TPEV-DISCONIMM のイベントを通知します。

TPDISCON は，会話を始めた側（オリジネータ）からだけしか呼び出せません。会話型サービス内で，このサービスを呼び出した通信ハンドルを使って，TPDISCON を呼び出せません。むしろ，会話型サービスは，会話の一方が完了したことを意味するために TPRETURN を使わなければなりません。同様に，会話型サービスと会話しているプログラムは TPDISCON を使えますが，結果を確実にするために望ましい方法は，サービスに TPRETURN でコネクションを終わらせてもらうことです。

TPDISCON は，コネクションをすぐに切断します（正常終了ではなく異常終了として）。送信先に届いていないデータは捨てられます。コネクションの他端点のプログラムが，呼び出し側のトランザクションに参加していても，TPDISCON を呼び出せます。この場合，トランザクションはロールバックします。また，TPDISCON が呼ばれたとき，呼び出し側は，コネクションの制御権を持っている必要はありません。

『データ領域』

『●TPSVCDEF-REC

COMM-HANDLE に，強制終了するコネクションの通信ハンドルを設定します。』

『●TPSTATUS-REC

TPDISCON の実行結果を示すリターン値が設定されます。』

リターン値

成功した場合，TPDISCON は TP-STATUS に TPOK を設定します。

エラー

次のような場合，TPDISCON はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

TPEBADDESC

COMM-HANDLE が間違っているか，または，呼び出された会話型サービスの通信ハンドルです。

TPETIME

タイムアウトが起こりました。この通信ハンドルは無効になります。

TPEPROTO

TPDISCON が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPCONNECT，TPRECV，TPRETURN，TPSEND

『OpenTP1 で使う場合の注意事項』

1. 『該当バージョンの OpenTP1 では，TPETIME が TP-STATUS にリターンすることはありません。』
2. 『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合は，会話型サービスの通信はできません。OSI TP 通信で会話型サービスの通信を使った場合，システムの動作は保証しません。』

TPGETRPLY

名称

リクエスト／レスポンス型サービスからの非同期応答の受信

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPGETRPLY は，先に送信したサービス要求の応答をリターンします。TPGETRPLY は，特定のサービス要求の応答か，または有効な不特定な応答を返します。どちらを返すかを示すオプションについて，次に説明します。

DATA-REC には，読む応答がどこにあるかを，そして LEN には DATA-REC に入れるバイトの最大値を設定します。TPGETRPLY が正常リターンすると，LEN には DATA-REC に入った実際の長さが設定されます。REC-TYPE と SUB-TYPE にはそれぞれ，データの type と subtype が設定されます。DATA-REC の値よりも応答が大きい場合は，DATA-REC にはレコードに入るだけのバイトが格納されます。応答の残りの部分は捨てられて，TPGETRPLY は TPTRUNCATE を返します。

TPGETRPLY が正常に終了した場合に，LEN が 0 であるときは，応答にはデータ部がなく，DATA-REC は変更されません。LEN に 0 を設定した場合はエラーになります。

『データ領域』

『●TPSVCDEF-REC

TPGETRPLY の動作を示す値と通信ハンドルを設定します。設定する値と意味は，以降説明します。』

『●TPTYPE-REC

受信するデータのデータ型と，subtype のレコード名を示します。』

『●DATA-REC

受信するデータを示します。』

『●TPSTATUS-REC

TPGETRPLY の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPGETANY

この値は，TPGETRPLY は入力時に COMM-HANDLE に設定した通信ハンドルを無視して，受信できる何らかの応答がある場合はそれをリターンして，返ってきた応答の通信ハンドルを出力の COMM-HANDLE に設定することを意味します。応答がなくて，この値を設定していない場合は，TPGETRPLY は応答の到着を待ちます。TPGETANY または TPGETHANDLE のどちらかを，必ず設定してください。

TPGETHANDLE

この値は，TPGETRPLY は入力時に COMM-HANDLE で示す通信ハンドルを使い，出力時にはこの通信ハンドルに有効な応答だけを返すことを意味します。応答がない場合は，TPGETRPLY は応答の到着を待ちます。TPGETANY または TPGETHANDLE のどちらかを，必ず設定してください。

TPNOCHANGE

この値を設定した場合，DATA-REC の型は変更されません。この値を設定した場合は，応答レコードの type と subtype が，REC-TYPE と SUB-TYPE にそれぞれ必ず一致させてください。
TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPCHANGE

受信レコードの type と subtype が REC-TYPE または SUB-TYPE と異なっていても，受け取り側が認識する限り，受信を許可することを示します。TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPNOBLOCK

TPGETRPLY は応答の到着を待ちません。応答が有効である場合，TPGETRPLY は応答を受信してリターンします。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定していて，利用できる応答がないときは，呼び出し側は応答が到着するか，タイムアウト（トランザクション，またはブロッキングタイムアウト）が起こるまで，ブロッキングします。
TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

呼び出し側にブロッキング状態が起こればブロッキングタイムアウトを受信して，ブロッキング時間に到着することを示します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

以下，特に記述しないかぎり，応答を受信したあとは，COMM-HANDLE は無効になります。

リターン値

成功した場合，TPGETRPLY は TP-STATUS に TPOK を設定します。TP-STATUS に TPOK か TPESVCFAIL が返された場合は，APPL-RETURN-CODE には TPRETURN の処理でアプリケーションが定義した値が返されます。TPGETRPLY が受信したデータの長さが入力の LEN に設定した長さを超えた場合は，TPTRUNCATE が設定され，LEN で示す長さ分だけ，DATA-REC に格納されます。受信したデータの残りの部分は捨てられます。

エラー

次のような場合，TPGETRPLY はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

注

TPGETHANDLE を設定した場合，特別な場合を除いて COMM-HANDLE は無効になります。応答が戻ってくる前にエラーが起こった場合は，COMM-HANDLE には 0 が設定されます。特に示さないかぎり，呼び出し側がトランザクション下にあった場合でも，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPEBADDESC

COMM-HANDLE に間違った通信ハンドルを設定しています。

TPEOTYPE

応答の type と subtype を，呼び出し側で認識できません。または，TPNOCHANGE を設定しているのに，REC-TYPE と SUB-TYPE が，サービスから送信されてきた応答の type と subtype に一致していません。REC-TYPE も SUB-TYPE も変更されていません。呼び出し側のトランザクションとしてサービス要求が実行されている場合，応答は捨てられて，トランザクションは rollback_only 状態となります。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合，これはトランザクションタイムアウトで，トランザクションは rollback_only 状態となります。そうでない場合は，TPNOBLOCK も TPNOTIME も設定されていない状態で，ブロッキングタイムアウトが起こったことを意味します。呼び出し側がトランザクションモードでないかぎり（そして，TPGETANY が設定

されていないかぎり），COMM-HANDLE は有効です。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，新しく送信しようとしていた送信や，解決していない応答は，TPETIME でエラーリターンします。

TPESVCFAIL

呼び出し側の応答を送信するサービスルーチンが，TPFAIL を設定した TPRETURN を呼び出しました。これは，アプリケーションレベルの失敗です。送信が完了している場合は，送信されたサービスの応答の内容は DATA-REC として有効になります。応答が呼び出し側の現在のトランザクションとして実行されている場合は，トランザクションは rollback_only 状態となります。

注

トランザクションがタイムアウトしないかぎり，それ以降の通信は，ロールバックする前までは実行されます。この処理は正常になるか，または無効になります（エラーリターンまたはイベントが通知されます）。継続する処理を有効にする通信には，TPNOTRAN を設定してください。トランザクション機能によって，呼び出し側のトランザクションをロールバックする何らかの処理が実行されます。

TPESVCERR

このエラーは，サービスルーチンを呼び出しているときか，TPRETURN で完了したときに起こります（例えば，適切でない引数を設定した）。このエラーが起こったときは，応答のデータはリターンされません（つまり，DATA-REC も TPTYPE-REC も変更されません）。サービスの応答が呼び出し側のトランザクションとして実行された場合は，トランザクションは rollback_only 状態になります。

注

トランザクションがタイムアウトしないかぎり，それ以降の通信は，ロールバックする前までは実行されます。この処理は正常になるか，または無効になります（エラーリターンまたはイベントが通知されます）。呼び出し側のトランザクションとして実行された処理は，トランザクションが完了したときにロールバックされます。継続する処理を有効にする通信には，TPNOTRAN を設定してください。トランザクション機能によって呼び出し側のトランザクションをロールバックする何らかの処理が実行されます。

TPEBLOCK

TPNOBLOCK を設定していたときに，ブロッキング状態になりました。COMM-HANDLE は有効なままです。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPGETRPLY が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが発生しました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPACALL，TPCANCEL，TPRETURN

『OpenTP1 で使う場合の注意事項』

1. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
2. 『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』
3. 『該当バージョンの OpenTP1 では，ロールバックする必要があるデータは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』
4. 『TPACALL で，呼び出したサービスが使えない型のレコードをサービスに渡していた場合，TPACALL は正常リターンして，TPGETRPLY でエラーとなります。TPGETRPLY で TPESYSTEM または TPESVCERR のエラーが起こった場合は，TPACALL の実行結果も確認してください。』
5. 『サービス要求先の SPP が異常終了した場合，定義の watch_time オペランドに指定した時間よりも早く TPETIME でリターンするときがあります。また，watch_time オペランドに 0（応答を無限に待つ）を指定している場合，TPEPROTO でエラーリターンするときがあります。』
6. 『OpenTP1 のセキュリティ機能を使っている場合で，サービス要求が認証されなかったときは，TPEPROTO でエラーリターンします。サービス要求が認証されなかったことが原因かどうかは，UAP トレースの詳細情報で確認してください。』
7. 『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合，受信するデータの長さは NET/Library 共通定義の NET バッファグループ定義 nettbuf にある length オペランドに指定した値を超えないようにしてください。』

TPRECV

名称

会話型サービスからのメッセージの受信

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPRECV は，ほかのプログラムからオープンコネクションを渡って送信されてきたデータを受信するために使います。COMM-HANDLE には，データを受信するためのオープンコネクションを設定します。COMM-HANDLE は，TPCONNECT または TPSVCSTART から返された通信ハンドルです。DATA-REC には，メッセージが読み込まれる位置を設定して，入力の場合は，LEN には DATA-REC に設定するバイトの最大値を設定します。

成功した場合，幾つかのイベントタイプでは，LEN には実際に DATA-REC に設定されたバイト長が設定されます。REC-TYPE と SUB-TYPE にはそれぞれ，データの type と subtype が設定されます。DATA-REC の値よりも応答が大きい場合は，DATA-REC にはレコードに入るだけのバイトが格納されます。応答の残りの部分は捨てられて，TPRECV は TPTRUNCATE を返します。

TPRECV が正常に終了した場合に TPTYPE-REC の LEN が 0 であるときは，応答にはデータ部がなく，DATA-REC は変更されません。入力時に LEN に 0 を設定した場合は，エラーになります。

TPRECV は，コネクションの制御権を持たないプログラムでだけ呼び出せます。

『データ領域』

『●TPSVCDEF-REC

TPRECV の動作を示す値と通信ハンドルを設定します。設定する値と意味は，以降説明します。』

『●TPTYPE-REC

受信するデータのレコード型と，subtype のレコード名を設定します。』

『●DATA-REC

受信するレコードを示します。』

『●TPSTATUS-REC

TPRECV の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPNOCHANGE

この値を設定した場合，DATA-REC の型は変更されません。この値を設定した場合は，応答レコードの type と subtype が，REC-TYPE と SUB-TYPE にそれぞれ必ず一致させてください。
TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPCHANGE

受信レコードの type と subtype が REC-TYPE または SUB-TYPE と異なっていても，受け取り側が認識する限り，それぞれ受信を許可することを示します。TPNOCHANGE または TPCHANGE のどちらかを，必ず設定してください。

TPNOBLOCK

TPRECV はデータの到着を待ちません。データがすでに受信できる状態の場合は，TPRECV はデータを受信してリターンします。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定して受信できるデータがない場合は，呼び出し側はデータを受信するまでブロック状態になります。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

ブロッキング監視時間が経過したあとにブロッキング状態がある場合に，ブロッキングタイムアウトが起こることを意味します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。
TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

イベント

COMM-HANDLE にイベントがあり，TPRECV がエラーにならない場合には，TPRECV は TP-STATUS に TPEEVENT を設定してリターンします。イベントのタイプは TP-EVENT に返されます。データは TPEV-SVCSUCC，TPEV-SVCFAIL と TPEV-SENDONLY のイベントとともに受信できます。TPRECV で有効なイベントを次に示します。

TPEV-DISCONIMM

会話のサブオーディネータに受信されるこのイベントは，会話のオリジネータが TPDISCON を実行してコネクションをすぐに切断したか，コネクションをオープンしたままで TPRETURN，TXCOMMIT，または TXROLLBACK を実行したときに受け取られます。また，このイベントは，通信エラーのためにコネクションが切断された場合（例えば，サーバ，マシン，ネットワークがダウンしたとき），オリジネータ，またはサブオーディネータにリターンされます。コネクションの切断はすぐに通知されるので（正常終了ではなく異常終了として），転送中のデータは捨てられます。二つのプログラムが同じトランザクションに参加している場合は，トランザクションは rollback_only 状態になります。このコネクションに使われている COMM-HANDLE は，無効になります。

TPEV-SENDONLY

コネクションの他端点のプログラムは，コネクションの制御権を放棄しています。このイベントの受け取り側はデータの送信が許可されていますが，制御権を放棄するまではデータを受信できません。

TPEV-SVCERR

会話のオリジネータに受信されるこのイベントは，会話のサブオーディネータが TPRETURN を実行していることを示します。TPRETURN は，サービスが成功してリターンすることを妨げるエラーに遭遇しました。例えば，TPRETURN に間違った引数が渡されていたり，サービスがほかのサブオーディネータのためにコネクションをオープンしている間に，TPRETURN が呼ばれたりした場合です。このイベントの性質のために，アプリケーションで定義されたデータやリターン値は役に立ちません。このコネクションは切断されて，COMM-HANDLE は無効になります。このイベントが受信側のトランザクションの一部で起こった場合は，トランザクションは rollback_only 状態となります。

TPEV-SVCFAIL

会話のオリジネータで受信されるこのイベントは，会話の他端点上のサブオーディネータのサービスがアプリケーションで指定されて，不成功に終了した（つまり，TPRETURN が TPFAIL を設定して呼ばれた）ことを示します。TPRETURN が呼ばれたとき，サブオーディネータのサービスがこのコネクションを制御する場合は，コネクションのオリジネータへレコードを返します。サービスルーチンの終了の部分で，サーバはコネクションを切断します。よって，COMM-HANDLE は無効になります。このイベントが受け取り側のトランザクションの一部で起こった場合は，トランザクションは rollback_only 状態となります。

TPEV-SVCSUCC

会話のオリジネータで受信されるこのイベントは，会話の他端点上のサブオーディネータのサービスがアプリケーションで指定されて，成功して終了した（つまり，TPSUCCESS を設定した TPRETURN を呼んだ）ことを示します。サービスルーチンの終了の部分で，サーバはコネクションを切断します。よって，COMM-HANDLE は無効になります。このイベントが受け取り側のトランザクションの一部で起こった場合は，コミットまたはロールバックするために，サーバ（もまた，トランザクションモードの場合）によってなされた処理を引き起こしたトランザクションのコミット（受け取り側がトランザクションを開始した場合）や，ロールバックのどちらかを実行できます。

リターン値

成功した場合，TPRECV は TP-STATUS に TPOK を設定します。TP-STATUS に TPOK か TPESVCFAIL が返された場合は，APPL-RETURN-CODE には TPRETURN の処理でアプリケーショ

ンが定義した値が返されます。受信したデータの長さが入力の LEN に設定した長さを超えた場合は，TPTRUNCATE が設定され，LEN で示す長さ分だけ，DATA-REC に格納されます。受信したデータの残りの部分は捨てられます。

エラー

次のような場合，TPRECV はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。特に示さないかぎり，呼び出し側がトランザクション下にあった場合でも，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPEBADDESC

COMM-HANDLE に間違った通信イベントを設定しています。

TPEOTYPE

到着したメッセージの type と subtype を，呼び出し側で認識していません。または，TPNOCHANGE を設定しているのに，REC-TYPE と SUB-TYPE は送信されてきたメッセージの type と subtype に一致していません。呼び出し側のトランザクションとしてサービス要求が実行されている場合は，到着したバッファを捨てるまで，トランザクションは rollback_only 状態となります。このエラーが起こった場合，COMM-HANDLE に該当するイベントは捨てられて，会話型通信は，未決着な状態になります。呼び出し側は，すぐに会話型通信を終了させてください。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合，これはトランザクションタイムアウトで，トランザクションは rollback_only 状態となります。そうでない場合は，TPNOBLOCK も TPNOTIME も設定されていない状態で，ブロッキングタイムアウトが起こったことを意味します。どちらの場合も，DATA-REC も TPTYPE-REC も変更されません。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，新しいメッセージを送受信しようとしても，新しいコネクションを開始しようとしても，TPETIME でエラーリターンします。

TPEEVENT

あるイベントが起こって，そのタイプは TP-EVENT に有効な値としてリターンされます。

TPEBLOCK

TPNOBLOCK を設定した TPRECV を実行したときに，ブロッキング状態になりました。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPRECV が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPCONNECT，TPDISCON，TPSEND

『OpenTP1 で使う場合の注意事項』

1. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
2. 『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』
3. 『該当バージョンの OpenTP1 では，ロールバックする必要があるエラーは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』
4. 『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合は，会話型サービスの通信はできません。OSI TP 通信で会話型サービスの通信を使った場合，システムの動作は保証しません。』

TPRETURN

名称

サービスルーチンからのリターン

形式

```
01 TPSVCRET-REC.
   COPY  TPSVCRET.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.

COPY TPRETURN  〔REPLACING TPSVCRET-REC BY TPSVCRET-REC〕
               〔REPLACING TPTYPE-REC BY TPTYPE-REC〕
               〔REPLACING DATA-REC BY DATA-REC〕 .
```

機能

TPRETURN は，サービスルーチンが完了したことを示します。TPRETURN はファイルで，サービスで実行される COBOL のコードの最後を含みます。このファイルには，次に示す三つの参照情報が含まれています。TPSVCRET-REC，TPTYPE-REC，および DATA-REC は，サービスで有効に使われるレコード名に置き換わります。TPRETURN は EXIT PROGRAM 文を含んでいるため，制御がコミュニケーションリソースマネジャへ返されるようにコミュニケーションリソースマネジャで呼び出された同じルーチンで実行される必要があります（つまり，制御がコミュニケーションリソースマネジャへ返らないので，TPRETURN はサービスルーチンのサブプログラムで呼び出さないようにしてください)。

TPRETURN は，サービスの応答メッセージを送信するために使います。TPRETURN が成功すると，TPCALL，TPGETRPLY，TPRECV で受信待ちしているプログラムのレコードに応答が返ります。

会話型サービスでは，TPRETURN はコネクションも終了させます。つまり，サービスルーチンは，TPDISCON を直接呼べません。正しい結果を保証するために，会話型サービスと通信しているプログラムは，TPDISCON を呼ばないでください。むしろ，会話型サービスが完了する通知を待ってください（つまり，TPRETURN から送信された TPEV-SVCSUCC か TPEV-SVCFAIL のようなイベントを待つ必要があります)。

サービスルーチンがトランザクションモードの場合は，TPRETURN は，トランザクションが完了したときに，そのサービスをコミットするかロールバックするかを決定します。サービスは，同じトランザクションの一部として複数回呼び出されるので，トランザクションのオリジネータから TXCOMMIT，TXROLLBACK が呼ばれるまで，必ずしも完全にコミット，またはロールバックされるとは限りません。

TPRETURN は，サービスルーチンから起動されたサービス要求からのすべての応答を受信したあとで呼んでください。そうでない場合は，サービスの性質に従って，サービスルーチンとの会話を開始したプログラムに TPESVCERR またはイベント TPEV-SVCERR がリターンされます。どのような未処理の応答

も，受信されていなければ，コミュニケーションリソースマネジャで自動的に捨てられます。さらに，これらの通信ハンドルは，無効になります。

TPRETURN は，サービスで開始されたすべてのコネクションがクローズされてから呼んでください。そうでない場合は，サービスの性質に従って，サービスルーチンとの会話を開始したプログラムに TPESVCERR またはイベント TPEV-SVCERR がリターンされます。さらに，緊急のコネクション切断イベント（TPEV-DISCONIMM）が，オープンコネクションに該当するすべてのオープンコネクションに送信されます。

コネクション制御に関して，TPRETURN を呼び出したサービスルーチンが，このサービスルーチンを呼び出したコネクションの制御権を持たない場合は，二つの結果が起こることが考えられます。一つ目は，もしサービスルーチンが，TP-RETURN-VAL（TPSVCRET-REC の）に TPFAIL を設定，REC-TYPE（TPTYPE-REC の）に SPACES（空白）を設定した TPRETURN を呼ぶ（つまり，データ部は送信されない）と，TPEV-SVCFAIL イベントが会話のオリジネータへ送信されます。二つ目は，ほかの値を設定した TPRETURN を呼ぶと，TPEV-SVCERR イベントがオリジネータへ送信されます。

会話型サービスは，開始していないオープンコネクションを一つだけしか持っていないので，コミュニケーションリソースマネジャは，送信された通信ハンドルのデータ（または，イベント）を認識しています。これらの理由で，通信ハンドルは TPRETURN に渡されません。

『データ領域』

『●TPSVCRET-REC

TP-RETURN-VAL と APPL-CODE を設定します。』

『●TPTYPE-REC

送信データのレコードの型と，subtype のレコード名を示します。』

『●DATA-REC

送信データを示します。』

TPRETURN の引数を次に示します。TP-RETURN-VAL には，次に示す値のうち一つが設定されます。

TPSUCCESS

サービスが成功して終了しました。データがある場合は，そのデータも送信されます（リターン処理中で失敗がなければ)。呼び出し側がトランザクションモードである場合は，TPRETURN は，そのサービスを最終的にコミットすることを決定します。

Note：TPRETURN は，必ずしもトランザクション全体を決着する訳ではありません。また，呼び出し側が成功を示しているときでも，未解決の応答やオープンコネクションがあるときや，サービスの中でトランザクションが rollback_only 状態となるような何らかの処理をしていた場合，エラーメッセージが送信されます（つまり，応答の受け取り側は，TPESVCERR またはイベント TPEV-SVCERR を受信します)。

Note：何らかの理由で，トランザクションがサービスルーチンで rollback_only 状態になる場合は，TP-RETURN-VAL には TPFAIL を設定してください。会話型サービスに TPSUCCESS が設定されていた場合は，イベント TPEV-SVCSUCC が通知されます。

TPFAIL

サービスがアプリケーションの立場から見て不成功に終了しました。エラーは応答を受信するプログラムに送信されます。つまり，応答を受信する呼び出しは失敗して，受け取り側は TPSVCFAIL またはイベント TPEV-SVCFAIL を受信します。呼び出し側がトランザクションモードである場合は，TPRETURN はトランザクションを rollback_only 状態にします（Note：トランザクションはすでに rollback_only 状態になっているかもしれません）。リターンの処理中で失敗がなければ，呼び出し側のデータがある場合は，送信されます。呼び出し側のデータが送信されない理由の一つは，トランザクションタイムアウトが起こったときです。この場合，応答を待っているプログラムは，TPETIME のエラーを受信します。

TP-RETURN-VAL に二つのうちのどちらでもない値が設定されている場合は，TPFAIL と見なされます。

アプリケーションで定義したリターンコードである APPL-CODE（TPSVCRET-REC の）は，サービスの応答を受信するプログラムへ送信されます。このコードは，応答が正常に送信されている限り，TP-RETURN-VAL に設定した内容に関係なく送信されます（つまり，呼び出しの受信で成功が返るか，TPESVCFAIL をリターンするか，TPEV-SVCSUCC または TPEV-SVCFAIL のイベントのうちの一つを受信する限り）。会話型サービスでは，APPL-CODE の値は，受け取り側の TP-STATUS-REC の APPL-RETURN-CODE として有効になります。

DATA-REC は，送信するレコードを示します。LEN（TPTYPE-REC の）には，DATA-REC のバイトの最大値を設定します。

Note：DATA-REC が長さを設定する必要がないタイプのレコードである場合は，LEN は無視されます（0 にしてください）。DATA-REC が長さを設定する必要があるタイプのレコードである場合は，LEN に 0 を設定しないでください。REC-TYPE に subtype が設定されていない場合は，SUB-TYPE は無視されます（SPACES（空白）にしてください）。REC-TYPE が SPACES（空白）の場合は，DATA-REC と LEN は無視されます。この場合，応答がサービスを呼び出したプログラムから要求されている場合は，データ部がない応答が送信されます。応答が要求されていない場合は，TPRETURN は渡されたデータを無視して，応答を送信しないでリターンします。

サービスが会話型の場合，データ部が送信されない理由には，次の二つの場合があります。

- 呼び出しが完了したときに，すでにコネクションが切断されている場合（つまり，呼び出し側がコネクションに関する TPEV-DISCONIMM を受信した），この呼び出し側は，単にサービスルーチンを終わらせて，トランザクションがある場合は，ロールバックします。この場合，呼び出し側のデータは送信されません。
- 上記で説明したように，呼び出し側がコネクションを制御していない場合は，TPEV-SVCFAIL または TPEV-SVCERR が，オリジネータ（コネクションの確立元）に送信されます。オリジネータがどのイベントを受け取るかどうかに関係なく，データは送信されません。しかし，オリジネータが TPEV-SVCFAIL を受信した場合は，オリジネータのリターンコードである TPSTATUS-REC の APPL-RETURN-CODE で有効になります。

リターン値

TPRETURN には EXIT PROGRAM 文が含まれているので，値は呼び出し側へ返らないで，また，制御をサービスルーチンに戻しません。サービスルーチンが TPRETURN を使わない場合（それは，直接 EXIT PROGRAM 文を使うか，またはサービスルーチンが終わったら），サーバはサービス要求者にサービスエラーをリターンします。すべてのサブオーディネータへのオープンコネクションはすぐに切断されて，未解決の非同期応答は捨てられます。サーバがトランザクションモードで失敗した場合は，トランザクションは rollback_only 状態になります。

Note：TPRETURN がサービスルーチンの外で（例えば，サービスでないルーチンから）使った場合，何の効果もありません。

エラー

TPRETURN はサービスルーチンを終了させるので，引数を処理しているときや，処理中のときに起こったエラーは，ルーチンの呼び出し側に通知できません。このようなエラーは，TPCALL か TPGETRPLY を経由してサービスの結果を受信するプログラムでは，TP-STATUS に TPESVCERR が返る要因となります。TPSEND や TPRECV を使っているプログラムでは，イベント TPEV-SVCERR が通知されます。

関連項目

TPCALL，TPCONNECT，TPDISCON，TPSEND，TPSVCSTART

『OpenTP1 で使う場合の注意事項』

1. 『COPY 文と REPLACING 句は，COBOL コンパイラによって動作が異なります。COBOL85 または COBOL2002 を使う場合には，次に示すようにコーディングしてください。

```
COPY TPRETURN 〔{REPLACING 〔TPSVCRET-REC BY TPSVCRET-REC〕
                           〔TPTYPE-REC BY TPTYPE-REC〕
                           〔DATA-REC BY DATA-REC〕 }〕 . 』
```

TPSEND

名称

会話型サービスへのメッセージの送信

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPSEND は，オープンコネクションを渡って，ほかのプログラムへデータを送信するために使います。呼び出し側は，コネクションの制御権がなくてはなりません。COMM-HANDLE には，データを送信するオープンコネクションを設定します。COMM-HANDLE には TPCONNECT または TPSVCSTART のどちらかから返された通信ハンドルです。

DATA-REC には送信されるデータを設定します。LEN にはデータをどれだけ送信したいかを設定します。

Note：DATA-REC が大きさを指定する必要がないレコードの場合は，LEN は無視されます（0 にしてください）。DATA-REC が大きさを指定する必要があるレコードの場合は，LEN は 0 にできません。REC-TYPE が，subtype がないデータ型の場合は，SUB-TYPE は無視されます（SPACES（空白）にしてください）。REC-TYPE が SPACES（空白）の場合は，DATA-REC と LEN は無視されて，データ部がないメッセージが送信されます（このような使い方をするのは，例えば，データを送信しないで，コネクションの制御を渡したい場合などです）。

『データ領域』

『●TPSVCDEF-REC

TPSEND の動作を示す値と通信ハンドルを設定します。設定する値と意味は，以降説明します。』

『●TPTYPE-REC

送信データのデータ型と，subtype のレコード名を設定します。』

『●DATA-REC

送信データを示します。』

『●TPSTATUS-REC

TPSEND の実行結果を示すリターン値が設定されます。』

TPSVCDEF-REC には，次に示す値を設定します。

TPRECVONLY

この値は，呼び出し側のデータが送信されたあと，呼び出し側は，コネクションの制御を放棄することを意味します（つまり，呼び出し側は，これ以上 TPSEND の呼び出しを実行できません）。コネクションの他端点である受信側が，TPSEND から送信されてきたこのデータを受信したとき，コネクションの制御を示すイベント（TPEV-SENDONLY）も受信します（これ以上，TPRECV の呼び出しを実行できません）。TPRECVONLY または TPSENDONLY のどちらかを，必ず設定してください。

TPSENDONLY

この値は，呼び出し側にコネクションの制御を残すことを意味します。TPRECVONLY または TPSENDONLY のどちらかを，必ず設定してください。

TPNOBLOCK

ブロッキング状態の場合（例えば，送信されたメッセージで内部バッファが満杯），データやイベントは送信されません。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPBLOCK

TPBLOCK を設定していて，ブロッキング状態のときは，呼び出し側は状況が収まるか，タイムアウト（トランザクション，またはブロッキングタイムアウト）が起こるまで，ブロッキングしています。TPNOBLOCK または TPBLOCK のどちらかを，必ず設定してください。

TPNOTIME

呼び出し側を無期限にブロックして，ブロッキングタイムアウトが起こらないことを意味します。トランザクションタイムアウトは起こります。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPTIME

ブロッキング状態になった場合に，ブロッキングタイムアウトの時間に達したときにブロッキングタイムアウトの通知を受け取ることを意味します。TPNOTIME または TPTIME のどちらかを，必ず設定してください。

TPSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼びます。
TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

TPNOSIGRSTRT

シグナルが実行中のシステムコールを中断したら，中断したシステムコールを再び呼ばないでルーチンは失敗します。TPSIGRSTRT または TPNOSIGRSTRT のどちらかを，必ず設定してください。

イベント

COMM-HANDLE にイベントが存在した場合は，TPSEND は，呼び出し側のデータを送信しないで失敗します。TP-EVENT にイベントタイプがリターンされます。TPSEND で有効なイベントを次に示します。

TPEV-DISCONIMM

会話のサブオーディネータに受信されるこのイベントは，会話のオリジネータが TPDISCON を実行してコネクションをすぐに切断したか，コネクションをオープンしたままで TPRETURN，TXCOMMIT，または TXROLLBACK を実行したことを示します。また，このイベントは，通信エラーのためにコネクションが切断された場合（例えば，サーバ，マシン，ネットワークがダウンしたとき），オリジネータ，またはサブオーディネータにリターンされます。

TPEV-SVCERR

会話のオリジネータに受信されるこのイベントは，会話のサブオーディネータが通信の制御権がない状態で TPRETURN を実行していることを示します。TPRETURN は，次に説明する TPEV-SVCFAIL とは異なった方法で実行されます。

TPEV-SVCFAIL

会話のオリジネータで受信されるこのイベントは，会話の他端点上のサブオーディネータが通信の制御権がない状態で TPRETURN を実行したことを示します。さらに，TPRETURN は TPFAIL コマンドとレコードなしで実行されました（つまり，TPRETURN に渡された REC-TYPE は，SPACES（空白）だったことを示します）。

これらのイベントは，コネクションがすぐに切断されたことを示します（正常終了ではなく異常終了として）。そのため，送信中のデータは失われます。このコネクションに使われている通信ハンドルは，無効になります。二つのプログラムが，同じトランザクションにある場合は，そのトランザクションは rollback_only 状態になります。

リターン値

成功した場合，TPSEND は TP-STATUS に TPOK を設定します。さらに，TP-STATUS に TPEEVENT が返って TP-EVENT が TPEV-SVCFAIL の場合は，APPL-RETURN-CODE には TPRETURN の一部として，アプリケーションが定義した値が返されます。

エラー

次のような場合，TPSEND はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。特に示さないかぎり，失敗は呼び出し側のトランザクションに影響を与えません。

TPEINVAL

間違った引数が与えられました（例えば，TPSVCDEF-REC に設定した値が間違っている）。

TPEBADDESC

COMM-HANDLE に無効な通信ハンドルが設定されています。

TPETIME

タイムアウトが起こりました。呼び出し側がトランザクションモードの場合は，トランザクションタイムアウトになり，トランザクションは rollback_only 状態となります。そうでない場合は，TPBLOCK か TPTIME が設定している状態で，ブロッキングタイムアウトが起こったことを意味します。どちらの場合も，DATA-REC も TPTYPE-REC も変更されていません。トランザクションタイムアウトが起こった場合は，トランザクションをロールバックするまでは，どのコネクションでのどのような送信，

受信の試みも，また 新しくコネクションを開始しようとする試みも，TPETIME でエラーリターンします。

TPEEVENT

イベントが起こりました。そのイベントは TP-EVENT に有効な値として設定されました。このエラーが起こったときは，DATA-REC は送信されません。

TPEBLOCK

TPNOBLOCK を設定した TPSEND を実行したときに，ブロッキング状態になりました。

TPEGOTSIG

シグナルは受信されましたが，TPNOSIGRSTRT を設定しています。

TPEPROTO

TPSEND が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPCONNECT，TPDISCON，TPRECV，TPRETURN

『OpenTP1 で使う場合の注意事項』

1. 『該当バージョンの OpenTP1 では，TPNOBLOCK の設定は無効となります。そのため，TPEBLOCK がリターンすることはありません。OpenTP1 では，ブロッキング状態のため通信ができない場合は，ネットワークダウンが原因で通信ができない場合と同様に，TPESYSTEM をリターンする仕様となっています。』
2. 『該当バージョンの OpenTP1 では，TPNOTIME の設定は無効となります。』
3. 『TPSIGRSTRT の設定は無効となります。この設定の有無に関係なく，シグナル受信時には，中断したシステムコールを再び呼びます。TPEGOTSIG がリターンすることはありません。』
4. 『OpenTP1 では，トランザクションタイムアウトが起こったときには，そのプロセスは異常終了します。そのため，TPETIME がリターンされるのは，ブロッキングタイムアウトの場合だけとなります。』
5. 『該当バージョンの OpenTP1 では，ロールバックする必要があるエラーは，X/Open で特に指定がないかぎり，TPESYSTEM としています。ただし，TPESYSTEM がリターンしても，rollback_only 状態とならない場合もあります。』
6. 『OpenTP1 では，会話の相手が TPDISCON，TPRETURN を実行している場合でも，TPSEND を実行するプロセスでイベントを受信していない場合は，TPSEND でイベントを通知できません。』

7.『TP1/NET/OSI-TP-Extended を使った OSI TP 通信をする場合は，会話型サービスの通信はできません。OSI TP 通信で会話型サービスの通信を使った場合，システムの動作は保証しません。』

TPSVCSTART

名称

サービスルーチンの開始

形式

```
01 TPSVCDEF-REC.
   COPY  TPSVCDEF.
01 TPTYPE-REC.
   COPY  TPTYPE.
01 DATA-REC.
   COPY  Data record definition.
01 TPSTATUS-REC.
   COPY  TPSTATUS.

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

機能

TPSVCSTART は，サービスルーチンをコーディングするときに最初に呼ばれます。TPSVCSTART を呼ぶ前に，サービスルーチン内でほかの XATMI の呼び出しをするとエラーになります。TPSVCSTART は，サービスへのパラメタとデータを検索するために使います。このルーチンは，TPCALL または TPACALL ルーチンからのサービス要求を受け取るために使われます。同様に TPCONNECT，TPSEND，TPRECV の要求を受け取るためにも使われます。

TPCALL または TPACALL を経由して実行されたサービス要求を処理するサービスルーチンは，多くても一つの入力されたメッセージ（TPSVCSTART が正常リターンしたこと）を受信して，一つの応答（TPRETURN でサービスルーチンを出たこと）を送信します。

他方，会話型サービスは，オープンコネクションを参照する手段のほかに，多くても一つの入力メッセージを伴った，コネクション要求で呼び出されます。TPSVCSTART が正常に終了すると，コネクション要求を出したプログラムも，会話型サービスも，アプリケーションで定義したデータを送信したり受信したりできるようになります。コネクションは半二重となっています。つまり，プログラムは，コネクションの制御権を通信相手が明示的に放棄するまで，会話を制御（つまり，データを送信する）できます。

トランザクションに関しては，トランザクションモードで起動した場合，サービスルーチンは，少なくとも一つのトランザクションに参加できます。サービスルーチンの作成者は，トランザクションは，サービスルーチンから戻ることで終了すると考えてください。サービスルーチンをトランザクションモードで起動しない場合，サービスルーチンは，必要なだけのトランザクションを TXBEGIN，TXCOMMIT，TXROLLBACK を使うことで開始します。

Note：TPRETURN は，トランザクションの完了には使われません。したがって，サービスルーチン内で起こったトランザクションを解決しないままにして TPRETURN を呼ぶと，エラーになります。

DATA-REC には，読む応答がどこにあるかを，そして LEN には DATA-REC に入れるバイトの最大値を設定します。TPSVCSTART が正常リターンすると，LEN には DATA-REC に入った実際の長さが設定されます。REC-TYPE と SUB-TYPE にはそれぞれ，データの type と subtype が設定されます。DATA-REC の値よりも応答が大きい場合は，DATA-REC にはレコードに入るだけのバイトが格納されます。応答の残りの部分は捨てられて，TPSVCSTART は TPTRUNCATE を返します。

TPSVCSTART が正常に終了した場合に，LEN が 0 であるときは，応答にはデータ部がなく，DATA-REC は変更されません。LEN に 0 を設定した場合はエラーになります。

TPSVCSTART が正常に終了した場合，SERVICE-NAME には要求するプログラムを起動するために使ったサービス名が設定されます。

『データ領域』

『●TPSVCDEF-REC

サービスルーチンの呼び出され方を示す値が設定されます。設定される値と意味は，以降説明します。』

『●TPTYPE-REC

受信データのデータ型と，subtype のレコード名が設定されます。』

『●DATA-REC

受信データが設定されます。』

『●TPSTATUS-REC

TPSVCSTART の実行結果を示すリターン値が設定されます。』

TPSVCSTART がリターンしたときに有効になる TPSVCDEF-REC の値を，次に示します。

TPREQRSP
: サービスは，TPCALL または TPACALL で呼び出されました。この設定は，TPCONV とは互いに排他的です。

TPCONV
: サービスは，TPCONNECT で呼び出されました。会話に使う通信ハンドルは，COMM-HANDLE として有効です。この設定は，TPREQRSP とは互いに排他的です。

TPNOTRAN
: サービスルーチンは，トランザクションモードではありません。この設定は，TPTRAN とは互いに排他的です。

TPTRAN
: サービスルーチンは，トランザクションモードです。この設定は，TPNOTRAN とは互いに排他的です。

TPNOREPLY

呼び出し側は応答を期待しません。このオプションは，TPREQRSP が設定されている場合にだけ意味を持ちます。この設定は，TPREPLY とは互いに排他的です。

TPREPLY

呼び出し側は応答を期待します。このオプションは，TPREQRSP が設定されている場合にだけ意味を持ちます。この設定は，TPNOREPLY とは互いに排他的です。

TPSENDONLY

このサービスは，コネクションを渡ってデータを送信だけできるように起動されて，コネクションの他端点のプログラムは，データを受信だけできるようになります。この値は，TPCONV が設定されているときだけ有効です。この設定は，TPRECVONLY とは互いに排他的です。

TPRECVONLY

このサービスは，コネクションを渡ってデータを受信だけできるように起動されて，コネクションの他端点のプログラムは，データを送信だけできるようになります。この値は，TPCONV が設定されているときだけ有効です。この設定は，TPSENDONLY とは互いに排他的です。

リターン値

成功した場合，TPSVCSTART は TP-STATUS に TPOK を設定します。受信したデータの長さが入力の LEN に設定した長さを超えた場合は，TPTRUNCATE が設定され，LEN で示す長さ分だけ，DATA-REC に格納されます。受信したデータの残りの部分は捨てられます。

エラー

次のような場合，TPSVCSTART はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

TPEINVAL

間違った引数が与えられました（例えば，LEN に 0 を設定した）。

TPEPROTO

TPSVCSTART が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPACALL，TPCALL，TPCONNECT，TPRETURN

TPUNADVERTISE

名称

サービス名の広告の取り消し

形式

```
01 SERVICE-NAME PIC X(15).
01 TPSTATUS-REC.
   COPY TPSTATUS.

CALL ”TPUNADVERTISE” USING SERVICE-NAME TPSTATUS-REC.
```

機能

TPUNADVERTISE は，あるサーバにそのサーバで提供する，あるサービスの広告を取り消すことを許可します。省略時には，サーバのサービスは，ブート時に広告されて，シャットダウンされたときに広告を取り消されます。

TPUNADVERTISE は，サーバで宣伝していた SERVICE-NAME を削除します。SERVICE-NAME には，SPACES（空白）は使えません。また，SERVICE-NAME は 15 文字以下になるようにしてください。15 文字よりも長い場合は，15 文字に切り詰められます。切り詰められた名称が ほかのサービス名と一致しないようにしてください。

『データ領域』

『●SERVICE-NAME

サービスのサービス名を設定します。』

『●TPSTATUS-REC

TPUNADVERTISE の実行結果を示すリターン値が設定されます。』

リターン値

成功した場合，TPUNADVERTISE は TP-STATUS に TPOK を設定します。

エラー

次のような場合，TPUNADVERTISE はエラーリターンして，次のうちどれか一つの値を TP-STATUS に設定します。

TPEINVAL
　SERVICE-NAME に SPACES（空白）を設定しています。

TPENOENT
　SERVICE-NAME は，現在このサーバで広告されていません。

TPEPROTO

TPUNADVERTISE が間違った状況で呼ばれました。

TPESYSTEM

コミュニケーションリソースマネジャシステムでエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

TPEOS

オペレーティングシステムにエラーが起こりました。厳密なエラーの性質は，product-specific な方法で定義されます。

関連項目

TPADVERTISE

『OpenTP1 で使う場合の注意事項』

1. 『一つのノードで負荷分散している場合（マルチサーバ），TPUNADVERTISE をどれか一つのプロセスが実行すると，負荷分散しているプロセスすべてでサービスを受け付けられなくなります。その後，TPADVERTISE で再びサービスを広告すれば，サービス要求を受け付けられるようになります。』
2. 『複数のノードで負荷分散している場合（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），TPUNADVERTISE を実行したプロセスのノードではサービスを実行しなくなりますが，ほかのノードのサーバでサービスを受け付けることができます。その後，TPADVERTISE で再びサービスを広告すれば，サービス要求を受け付けられるようになります。』

TX インタフェースのアプリケーションプログラミングインタフェース（TX～）

TX インタフェースの API の文法について説明します。この節の記述は，X/Open 発行の「X/Open CAE Specification Distributed TP：The TX （Transaction Demarcation） Specification」の文法部である「Chapter 5　COBOL Reference Manual Pages」の記述を，日本語訳したものです。

TX インタフェースの API を次に示します。

- TXINTRO － COBOL データ構造
- TXBEGIN － トランザクションの開始
- TXCLOSE － リソースマネジャ集合のクローズ
- TXCOMMIT － トランザクションのコミット
- TXINFORM － 現在のトランザクションに関する情報の返却
- TXOPEN － リソースマネジャ集合のオープン
- TXROLLBACK － トランザクションのロールバック
- TXSETCOMMITRET － commit_return 特性の設定
- TXSETTIMEOUT － transaction_timeout 特性の設定
- TXSETTRANCTL － transaction_control 特性の設定

TX インタフェースの API（TX～）は，TP1/Server Base と TP1/LiNK のどちらの UAP でも使えます。

TXINTRO

名称

COBOL データ構造

機能

概要

COBOL での TX インタフェースの各呼び出しの統合的な機能を示します。

各呼び出し処理は，次に示すような項目によって記述されています。

- 各呼び出し処理に必要な WORKING STORAGE に定義された COBOL のレコード。これは COPY ステートメントによって記述されています。
- PROCEDURE DIVISION で記述された呼び出しの形式
- 呼び出しの機能説明
- リターンコードの一覧

COBOL の TX インタフェースで使用されるデータ構造

二つの COBOL のレコード：TX-RETURN-STATUS と TX-INFO-AREA は，一般的に TX の呼び出し処理で使われます。これらは，COPY ステートメントの指定によって，WORKING-STORAGE SECTION で定義されているものとします。

『以降に示す二つの COBOL レコードは，$DCDIR/include/COBOL/ディレクトリに格納されています。UAP をコンパイルするときには，COPY ファイルの存在場所としてこのディレクトリを指定する必要があります。コンパイル時の指定については，使用する COBOL 言語のマニュアルを参照してください。』

TX-RETURN-STATUS

この章で記述する各機能は，要求によって，パラメタとしてこのレコードを参照します。このレコードは呼び出しに対して値を返すために使われます。このレコードは次の形式で使われるものとします。『TX-RETURN-STATUS の値 TXSTATUS は，データ名を FILLER としてあるので，処理から正しく参照できません。TX-RETURN-STATUS をほかの名称で再定義して，処理からは再定義した名称で参照するようにしてください。詳細は，「6.4　X/Open に準拠した UAP のコーディング例」を参照してください。』

```
01 TX-RETURN-STATUS.
   COPY TXSTATUS.
```

TXSTATUS は，符号付き整数で定義されて，COBOL の TEXT LIBRARY として存在していて，ステータスの値は次に示す値をとります。

```
 05  FILLER  PIC  S9(9)  COMP-5.
     88  TX-NOT-SUPPORTED       VALUE  1.
*        Normal execution
```

```
     88  TX-OK                  VALUE  0.
*        Normal execution
     88  TX-OUTSIDE             VALUE  -1.
*        Application is in an RM local  transaction
     88  TX-ROLLBACK            VALUE  -2.
*        Transaction  was  rolled  back
     88  TX-MIXED               VALUE  -3.
*        Transaction was partially committed and partially rolled back
     88  TX-HAZARD              VALUE  -4.
*        Transaction may have been partially committed and partially
*        rolled back
     88  TX-PROTOCOL-ERROR      VALUE  -5.
*        Routine invoked in an inproper context
     88  TX-ERROR               VALUE  -6.
*        Transient error
     88  TX-FAIL                VALUE  -7.
*        Fatal error
     88  TX-EINVAL              VALUE  -8.
*        Invalid arguments were given
     88  TX-COMMITTED           VALUE  -9.
*        The transaction was heuristically committed
     88  TX-NO-BEGIN            VALUE  -100.
*        Transaction committed plus new transaction could not be
*        started
     88  TX-ROLLBACK-NO-BEGIN   VALUE  -102.
*        Transaction rollback plus new transaction could not be
*        started
     88  TX-MIXED-NO-BEGIN      VALUE  -103.
*        Mixed plus new transaction could not be started
     88  TX-HAZARD-NO-BEGIN     VALUE  -104.
*        Hazard plus new transaction could not be started
     88  TX-COMMITTED-NO-BEGIN  VALUE  -109.
*        Heuristically committed plus new transaction could not be started
```

TX-INFO-AREA

このレコードは，TXINFORM の呼び出しの結果が格納されるデータ構造を定義します。このレコードは，次の形式で使われるものとします。

```
01 TX-INFO-AREA.
   COPY TXINFDEF.
```

TXINFDEF は，次に示すような COBOL の TEXT LIBRARY です。

```
*  XID  record
        05  XID-REC.
            10  FORMAT-ID       PIC  S9(9)  COMP-5.
*  A  value  of  -1  in  FORMAT-ID  means  that  the  XID  is  null
            10  GTRID-LENGTH    PIC  S9(9)  COMP-5.
            10  BRANCH-LENGTH   PIC  S9(9)  COMP-5.
            10  XID-DATA        PIC  X(128).
*  Transaction  mode  settings
        05  TRANSACTION-MODE    PIC  S9(9)  COMP-5.
            88  TX-NOT-IN-TRAN  VALUE  0.
            88  TX-IN-TRAN      VALUE  1.
```

```
*  Commit-return settings
        05  COMMIT-RETURN       PIC  S9(9)  COMP-5.
            88  TX-COMMIT-COMPLETED        VALUE  0.
            88  TX-COMMIT-DECISION-LOGGED  VALUE  1.
*  Transaction_control  settings
        05  TRANSACTION-CONTROL PIC  S9(9)  COMP-5.
            88  TX-UNCHAINED  VALUE  0.
            88  TX-CHAINED    VALUE  1.
*  Transaction_timeout  value
        05  TRANSACTION-TIMEOUT PIC  S9(9)  COMP-5.
            88  NO-TIMEOUT  VALUE  0.
*  Transaction_state  information
        05  TRANSACTION-STATE   PIC  S9(9)  COMP-5.
            88  TX-ACTIVE  VALUE  0.
            88  TX-TIMEOUT-ROLLBACK-ONLY VALUE  1.
            88  TX-ROLLBACK-ONLY VALUE  2.
```

TXBEGIN

名称

トランザクションの開始

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.

 PROCEDURE DIVISION.
 CALL "TXBEGIN" USING TX-RETURN-STATUS.
```

機能

TXBEGIN は，呼び出し元スレッドをトランザクションモードにします。呼び出しスレッドは，まずリンクされたリソースマネジャがオープンされていることを（TXOPEN で）保証しなければなりません。そのあとで，トランザクションの開始が可能になります。

TXBEGIN の呼び出し元が，すでにトランザクションモードに入っているか，または TXOPEN をまだ発行していない場合，TXBEGIN はエラーリターンします（TX-RETURN-STATUS に TX-PROTOCOL-ERROR が返ります）。

TXBEGIN でトランザクションモードに入ったスレッドは，TXCOMMIT，TXROLLBACK を呼び出して現在のトランザクションを必ず完了させてください。

トランザクションの開始では，呼び出し元が明示的に TXBEGIN の呼び出しを必要としない，トランザクション連鎖にかかわる場合があります。

トランザクション連鎖については，TXCOMMIT，TXROLLBACK を参照してください。

『TXBEGIN は MHP から呼び出せません。』

『次に示す API で設定した値は，TXBEGIN の処理に影響を与えます。』

- TXSETTIMEOUT

『データ領域』

「●TX-RETURN-STATUS

TXBEGIN の実行結果が返されます。』

リターン値

成功した場合，TXBEGIN は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』

エラー

次のような場合，TXBEGIN はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-OUTSIDE

呼び出し元のスレッドが現在，グローバルトランザクションの外部で，一つ以上のリソースマネジャに関連する作業をしています。この場合，トランザクションマネジャはグローバルトランザクションを開始できません。このような作業は，グローバルトランザクションが開始される前に完了していなければなりません。

呼び出し元の状態が，ローカルトランザクション中にあることに変わりはありません。

TX-PROTOCOL-ERROR

TXBEGIN は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がすでにトランザクションモードである場合）。

呼び出し元の状態がトランザクションモードになっているかどうかは変わりません。

TX-ERROR

トランザクションマネジャ，または一つ以上のリソースマネジャが，トランザクションを開始しようとした際に一時的なエラーが起こりました。

このエラーが返った場合は，呼び出し元の状態はトランザクションモードではありません。

TX-FAIL

トランザクションマネジャ，または一つ以上のリソースマネジャで，回復できないエラーが発生しました。トランザクションマネジャ，一つ以上のリソースマネジャのどちらか，またはその両方が，すでにアプリケーションに代わって作業できないことがエラーの原因です。

このエラーが返った場合は，呼び出し元はトランザクションモードではありません。

参照

TXCOMMIT，TXOPEN，TXROLLBACK，TXSETTIMEOUT

アプリケーションの使い方

グローバルトランザクションに参加するには，XA に準拠したリソースマネジャが正常にオープンされていることが前提です。詳細は TXOPEN を参照してください。

『OpenTP1 で使う場合の注意事項』

1. 『SPP からトランザクション処理を開始する場合は，TXBEGIN を必ず呼び出します。SPP の場合は，呼び出し元で TXBEGIN を呼び出していれば，トランザクション処理になります。』
2. 『TXBEGIN でトランザクションを開始するプロセスは，このマニュアルの記述に従って正しく結合された UAP の実行形式ファイルを起動したものでなければなりません。』
3. 『TXBEGIN と，OpenTP1 の CBLDCTRN は併用できません。』

TXCLOSE

名称

リソースマネジャ集合のクローズ

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.

 PROCEDURE DIVISION.
 CALL "TXCLOSE" USING TX-RETURN-STATUS.
```

機能

TXCLOSE は，互換性がある方法で，リソースマネジャの集合をクローズします。TXCLOSE でリソースをクローズすると，トランザクションマネジャがリソースマネジャ固有の情報をトランザクションマネジャ独自の方法で読み，この情報を，呼び出し元にリンクされているリソースマネジャに渡します。

TXCLOSE は，呼び出し元にリンクされているすべてのリソースマネジャをクローズします。TXCLOSE は，リソースマネジャ固有のクローズ呼び出しの代わりに発行され，アプリケーションプログラムからの互換性を阻害する呼び出しを不要にします。

リソースマネジャは，終了セマンティクスがそれぞれ異なっています。そのため，特定のリソースマネジャをクローズするのに必要な固有な情報は，それぞれのリソースマネジャから公開されてなければなりません。

アプリケーションスレッドで，もはやグローバルトランザクションに入るつもりがないときには，TXCLOSE を呼び出します。

呼び出し元がトランザクションモードにいる場合は，TXCLOSE は失敗します（TX-PROTOCOL-ERROR を返します)。すなわち，幾つかのリソースマネジャが現在のトランザクションに参加していない場合も，リソースマネジャはクローズされません。呼び出し元にリンクされているすべてのリソースマネジャがクローズできれば，TXCLOSE は正常にリターンします（TX-OK が返ります)。

『データ領域』

『●TX-RETURN-STATUS

TXCLOSE の実行結果が返されます。』

リターン値

成功した場合，TXCLOSE は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』
『TXCLOSE の呼び出し元に結合していたリソースマネジャの集合はクローズされます。』

エラー

次のような場合，TXCLOSE はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-PROTOCOL-ERROR

TXCLOSE は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がトランザクションモードにある場合）。リソースマネジャは一つもクローズされません。

TX-ERROR

トランザクションマネジャ，または一つ以上のリソースマネジャで一時的なエラーが起こりました。クローズできるリソースマネジャはすべてクローズされます。

TX-FAIL

トランザクションマネジャ，または一つ以上のリソースマネジャで，回復できないエラーが起こりました。トランザクションマネジャ，または一つ以上のリソースマネジャのどちらか，またはその両方が，すでにアプリケーションに代わって作業できないことがエラーの原因です。

参照

TXOPEN

『OpenTP1 で使う場合の注意事項』

1. 『TXCLOSE でクローズできるのは，X/Open の XA インタフェースに準拠しているリソースマネジャだけです。』

TXCOMMIT

名称

トランザクションのコミット

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.

 PROCEDURE DIVISION.
 CALL "TXCOMMIT" USING TX-RETURN-STATUS.
```

機能

TXCOMMIT は，呼び出し元のスレッドのトランザクションの作業をコミットします。

transaction_control 特性が，TX-UNCHAINED に設定されていた場合に，TXCOMMIT がリターンすると，呼び出し元スレッドはトランザクションモードにはいません。しかし，transaction_control 特性が TX-CHAINED に設定されていた場合は，TXCOMMIT がリターンすると，呼び出し元スレッドは新しいトランザクションのためにトランザクションモードにとどまります。transaction_control 特性については，TXSETTRANCTL を参照してください（リターン値を参照）。

『次に示す API で設定した値は，TXCOMMIT の処理に影響を与えます。』

- TXSETCOMMITRET
- TXSETTRANCTL
- TXSETTIMEOUT

『データ領域』

『●TX-RETURN-STATUS

TXCLOSE の実行結果が返されます。』

リターン値

成功した場合，TXCOMMIT は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』『transaction_control 特性が TX-CHAINED に設定されていた場合，新しいグローバルトランザクションが開始します。』

エラー

次のような場合，TXCOMMIT はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-NO-BEGIN

トランザクションはコミット終了しましたが，新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードにいません。

このリターン値は transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-ROLLBACK

トランザクションはコミットできないで，ロールバックしました。transaction_control 特性が TX-CHAINED に設定されている場合は，新しいトランザクションが開始されます。

TX-ROLLBACK-NO-BEGIN

トランザクションはコミットできないで，ロールバックしました。新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードではありません。

このリターン値は transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-MIXED

トランザクションは部分的にコミット，部分的にロールバックしました。transaction_control 特性が TX-CHAINED が設定されている場合は，新しいトランザクションが開始されます。

TX-MIXED-NO-BEGIN

トランザクションは部分的にコミット，部分的にロールバックしました。新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードにはいません。

このリターン値は transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-HAZARD

障害が原因で，トランザクションは部分的にコミット，部分的にロールバックされた可能性があります。transaction_control 特性が TX-CHAINED に設定されている場合は，新しいトランザクションが開始されます。

トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，1 相コミット時にリソースマネジャからのリターン値が XAER_NOTA の場合も，TX-HAZARD を返します。

TX-HAZARD-NO-BEGIN

障害が原因で，トランザクションは部分的にコミット，部分的にロールバックしました。新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードにはいません。

このリターン値は，トランザクション特性が TX-CHAINED のときだけ返ります。

トランザクションサービス定義で，trn_extend_function オペランドに 00000001 を指定し，1 相コミット時にリソースマネジャからのリターン値が XAER_NOTA の場合も，TX-HAZARD-NO-BEGIN を返します。

TX-PROTOCOL-ERROR

TXCOMMIT は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がトランザクションモードにいない場合)。呼び出し元のトランザクションに対する状態は変わりません。

TX-FAIL

トランザクションマネジャ，または一つ以上のリソースマネジャで，回復できないエラーが起こりました。トランザクションマネジャ，または一つ以上のリソースマネジャのどちらか，またはその両方が，

すでにアプリケーションに代わって作業できないことがエラーの原因です。呼び出し元のトランザクションに対する状態は不明です。

参照

TXBEGIN，TXSETCOMMITRET，TXSETTRANCTL，TXSETTIMEOUT

『OpenTP1 で使う場合の注意事項』

1. 『TXCOMMIT は，グローバルトランザクションを開始した UAP（TXBEGIN を呼び出した UAP）のプロセスからだけ呼び出せます。』
2. 『TXCOMMIT を呼び出すプロセスは，このマニュアルの記述に従って正しく結合された UAP の実行形式ファイルを起動したものでなければなりません。』
3. 『TXCOMMIT と，OpenTP1 の CBLDCTRN は併用できません。』

TXINFORM

名称

現在のトランザクションに関する情報の返却

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.
*
 01 TX-INFO-AREA.
    COPY TXINFDEF.

 PROCEDURE DIVISION.
 CALL "TXINFORM" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能

TXINFORM は，グローバルトランザクションの情報を，TX-INFO-AREA に返します。さらに，TXINFORM は呼び出し元が現在トランザクションモードにあるかどうかを示す値を返します。

『データ領域』

『●TX-INFO-AREA

TXINFORM は TX-INFO-AREA レコードに，グローバルトランザクションの情報を設定します。TX-INFO-AREA レコードの内容については，TXINTRO を参照してください。

トランザクションモードで TXINFORM を発行した場合は，TX-IN-TRAN が設定され，XID-REC に現在のトランザクションブランチの識別子を，TRANSACTION-STATE には現在のトランザクションの状態が設定されます。

トランザクションモードでない呼び出し元から TXINFORM を呼び出した場合には，TRANSACTION-MODE には TX-NOT-IN-TRAN が設定され，XID-REC には null XID が設定されます（詳細は TXINTRO を参照)。さらに，呼び出し元がトランザクションモードであるかないかにかかわらず，COMMIT-RETURN，TRANSACTION-CONTROL，TRANSACTION-TIMEOUT は，現在設定されている commit_return，transaction_control と秒単位のトランザクションタイムアウト値と同じ値になります。

返されたトランザクションタイムアウト値は，次のトランザクションモードが開始されたときに設定されて反映されます。しかし，呼び出し元の現在のトランザクションでは，タイムアウト値に反映されていない場合もあります。その理由は，現在のトランザクションの開始後 TXSETTIMEOUT が呼ばれた場合，timeout 値を変更しているかもしれないからです。』

『●TX-RETURN-STATUS

TXINFORM の実行結果が返されます。』

リターン値

TXINFORM の呼び出し元がトランザクションモードの場合は，1 が返ります。TXINFORM の呼び出し元がトランザクションモードでない場合は，0 が返ります。

エラー

次のような場合，TXINFORM はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-PROTOCOL-ERROR

TXINFORM は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がまだ TXOPEN を呼び出していない)。

TX-FAIL

トランザクションマネジャで，回復できないエラーが起こりました。トランザクションマネジャが，すでにアプリケーションに代わって作業できないことがエラーの原因であることを示します。

アプリケーションの使い方

同じグローバルトランザクション内で，複数回の TXINFORM を呼び出すと，TXINFORM への呼び出しでは，gtrid（グローバルトランザクション識別子）に同じ XID が供給されることを保証されています。しかし，bqual（ブランチ限定子）が同じかどうかは一概にはいえません。

参照

TXOPEN，TXSETCOMMITRET，TXSETTRANCTL，TXSETTIMEOUT

TXOPEN

名称

リソースマネジャ集合のオープン

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.

 PROCEDURE DIVISION.
 CALL "TXOPEN" USING TX-RETURN-STATUS.
```

機能

TXOPEN は，互換性がある方法で，リソースマネジャの集合をオープンします。TXOPEN でリソースをオープンすると，トランザクションマネジャがリソースマネジャ固有の情報をトランザクションマネジャ独自の方法で読み，この情報を，呼び出し元にリンクされているリソースマネジャに渡します。

TXOPEN は，呼び出し元にリンクされているすべてのリソースマネジャをオープンしようとします。TXOPEN は，リソースマネジャ固有のオープン呼び出しの代わりに発行され，アプリケーションプログラムからの互換性を阻害する呼び出しを不要にします。

リソースマネジャは，初期化セマンティクスがそれぞれ異なっています。そのため，特有のリソースマネジャをオープンするのに必要な情報は，それぞれのリソースマネジャから公開されてなければなりません。

TXOPEN が TX-ERROR を返したら，リソースマネジャは一つもオープンされていません。TXOPEN が TX-OK を返したら，一つ以上のリソースマネジャがオープンされています。アプリケーションからアクセスしたときにリソースマネジャがオープンしていないと，リソースマネジャ固有のエラーを返します。あるスレッドがグローバルトランザクションに参加する前には，TXOPEN が成功してリターンしていなければなりません。

TXOPEN が成功したあとでも，繰り返し TXOPEN を（TXCLOSE を呼び出す前なら）呼び出せます。繰り返し TXOPEN を呼び出しても正常にリターンしますが，トランザクションマネジャでは，リソースマネジャの再オープン処理は一切しません。

『データ領域』

『●TX-RETURN-STATUS

TXOPEN の実行結果が返されます。』

リターン値

成功した場合，TXOPEN は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』『TXOPEN の呼び出し元に結合している一つ以上のリソースマネジャの集合はオープンされます。』

エラー

次のような場合，TXOPEN はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-ERROR

トランザクションマネジャ，または一つ以上のリソースマネジャで一時的なエラーが起こりました。リソースマネジャは一つもオープンできませんでした。

TX-FAIL

トランザクションマネジャ，または一つ以上のリソースマネジャで，回復できないエラーが起こりました。トランザクションマネジャ，一つ以上のリソースマネジャのどちらか，またはその両方が，アプリケーションに代わって作業できないことがエラーの原因です。

または，実行環境がジャーナルファイルレスモードのため，トランザクションマネジャでエラーが起こりました。

参照

TXCLOSE

『OpenTP1 で使う場合の注意事項』

1. 『TXOPEN でオープンできるのは，X/Open の XA インタフェースに準拠しているリソースマネジャだけです。』
2. 『TXOPEN と，OpenTP1 の CBLDCTRN は併用できません。』

TXROLLBACK

名称

トランザクションのロールバック

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.

 PROCEDURE DIVISION.
 CALL "TXROLLBACK" USING TX-RETURN-STATUS.
```

機能

TXROLLBACK は，呼び出しスレッドのトランザクションの作業をロールバックします。

transaction_control 特性が，TX-UNCHAINED に設定されていた場合で，TXROLLBACK がリターンすると，呼び出し元スレッドはトランザクションモードにはいません。しかし，transaction_control 特性が TX-CHAINED に設定されていた場合で，TXROLLBACK がリターンすると，呼び出し元スレッドは新しいトランザクション下のトランザクションモードにとどまります。transaction_control 特性については，TXSETTRANCTL を参照してください。

『次に示す API で設定した値は，TXROLLBACK の処理に影響を与えます。』

- TXSETTRANCTL
- TXSETTIMEOUT

『TXROLLBACK は MHP からは呼び出せません。』

『データ領域』

『●TX-RETURN-STATUS

TXROLLBACK の実行結果が返されます。』

リターン値

成功した場合，TXROLLBACK は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』『transaction_control 特性が TX-CHAINED に設定されていた場合は，新しいトランザクションが開始されます。』『TXROLLBACK を呼び出した SPP がルートトランザクションブランチでない場合は，実際のロールバック処理はしないで，そのトランザクションブランチを rollback_only 状態にするだけです。ルートトランザクションブランチの同期点処理でロールバックの指示があるまで，トランザクションモードのままとどまります。』

エラー

次のような場合，TXROLLBACK はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。

TX-NO-BEGIN

トランザクションはロールバックしましたが，新しいトランザクションを開始できないで，呼び出し元はすでにトランザクションモードにいません。

このリターン値は，transaction_control 特性が TX-CHAINED のときだけ発生します。

TX-MIXED

トランザクションは部分的にコミット，部分的にロールバックしました。transaction_control 特性が TX-CHAINED と設定されていれば，新しいトランザクションが開始されます。

TX-MIXED-NO-BEGIN

トランザクションは部分的にコミット，部分的にロールバックしました。新しいトランザクションは開始できないで，呼び出し元は，すでにトランザクションモードにはいません。

このリターン値は，transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-HAZARD

障害が原因で，トランザクションは部分的にコミット，部分的にロールバックした可能性があります。transaction_control 特性が TX-CHAINED に設定されていれば，新しいトランザクションが開始されます。

TX-HAZARD-NO-BEGIN

障害が原因で，トランザクションは部分的にコミット，部分的にロールバックした可能性があります。新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードではありません。

このリターン値は，transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-COMMITTED

トランザクションは，ヒューリスティックにコミットしました。transaction_control 特性が TX-CHAINED に設定されていれば，新しいトランザクションが開始されます。

TX-COMMITTED-NO-BEGIN

トランザクションは，ヒューリスティックにコミットしました。新しいトランザクションは開始できないで，呼び出し元はすでにトランザクションモードにはいません。このリターン値は，transaction_control 特性が TX-CHAINED のときだけ返ります。

TX-PROTOCOL-ERROR

TXROLLBACK は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がトランザクションモードでない）。

TX-FAIL

トランザクションマネジャ，または一つ以上のリソースマネジャで，回復できないエラーが起こりました。トランザクションマネジャ，または一つ以上のリソースマネジャのどちらか，またはその両方が，

すでにアプリケーションに代わって作業できないことがエラーの原因です。呼び出し元のトランザクションに対する状態は不明です。

参照

TXBEGIN，TXSETTRANCTL，TXSETTIMEOUT

『OpenTP1 で使う場合の注意事項』

1. 『transaction_control 特性が TX-CHAINED に設定されていた場合，TXROLLBACK を呼び出せるのはルートトランザクションブランチ（TXBEGIN を呼び出した UAP）だけです。』
2. 『transaction_control 特性が TX-UNCHAINED に設定されていた場合，TXROLLBACK はルートトランザクションブランチでなくても呼び出せますが，呼び出したトランザクションブランチによって処理が異なります。TXROLLBACK の呼び出し元がルートブランチの場合，非ルートブランチに対して，RPC を介してロールバック要求をしますが，非ルートブランチで TXROLLBACK を呼び出した場合，呼び出し元の非ルートブランチは rollback_only を記録するだけで，ルートブランチに対して RPC を介してロールバック要求をしません。呼び出し元の非ルートブランチは，ルートブランチの指示を待ったあと，ロールバック処理をします。』
3. 『TXROLLBACK と，OpenTP1 の CBLDCTRN は併用できません。』

TXSETCOMMITRET

名称

commit_return 特性の設定

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.
*
 01 TX-INFO-AREA.
    COPY TXINFDEF.

 PROCEDURE DIVISION.
 CALL "TXSETCOMMITRET" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能

TXSETCOMMITRET は，commit_return 特性を when_return に指定した値に設定します。commit_return 特性は，TXCOMMIT が呼び出し元に制御を戻す方法に対して影響します。

TXSETCOMMITRET は，呼び出し元がトランザクションモードかどうかに関係なく呼び出せます。設定した commit_return 特性は，あとから呼び出される TXSETCOMMITRET で変更されるまで引き継がれて，効力を発揮します。commit_return 特性の初期設定値は，製品の仕様に依存します。『OpenTP1 の場合は，TX-COMMIT-COMPLETED です。』

『データ領域』

『●TX-INFO-AREA

次の二つのどちらかの値が，COMMIT-RETURN の妥当な設定です。

{TX-COMMIT-DECISION-LOGGED|TX-COMMIT-COMPLETED}』

- TX-COMMIT-DECISION-LOGGED『この値は，OpenTP1 の該当バージョンでは使えません。TX-INFO-AREA に TX-COMMIT-DECISION-LOGGED を設定した場合は，リターン値 TX-NOT-SUPPORTED でエラーリターンします。』

  このフラグは，コミット決定を 2 相コミットプロトコルの 1 相目にジャーナルに書いたが，2 相目がまだ完了していないときに，TXCOMMIT がリターンすることを示します。このフラグを設定すれば，TXCOMMIT の呼び出し元に早くリターンできますが，トランザクションの参加者がヒューリスティック決定をしてしまう危険性があります。この場合，TXCOMMIT はすでにリターンしているので『トランザクションマネジャは，それ独自の方法でリソースマネジャがヒューリスティック決定をしたことを示すことに注意』，ヒューリスティック決定が発生すると，呼び出し元にリターンコードで示すことはできません。

通常の条件では，1 相目にコミットを約束した参加者は，2 相目の処理の間にコミットをします。一定の特殊な条件下では，長期間のネットワーク，サイト障害などによって，2 相目の完了ができないで，ヒューリスティック決定が発生するようなこともあります。トランザクションマネジャは，このフラグは選択しないことにした場合，この値が設定されていないことを示すため，TX-NOT-SUPPORTED を返します。

- TX-COMMIT-COMPLETED

  このフラグは，2 相コミットプロトコルが完全に終了したあとで TXCOMMIT がリターンすることを示します。このフラグを設定すると，コミットの 2 相目でヒューリスティックな決着が起こっても，TXCOMMIT の呼び出し元にリターンコードで知らせることができます。トランザクションマネジャは，この機能を使えないようにできます。そして TX-NOT-SUPPORTED を返す理由にこのフラグを使えない意味を含めることができます。

リターン値

『リターン値が 0 で』成功した場合，TXSETCOMMITRET は TX-OK を返します。これは，負ではないリターン値です。『この場合，commit_return 特性の設定は，TX-INFO-AREA に設定した値に変更されます。』

『リターン値が正の値で』成功した場合，TXSETCOMMITRET は TX-NOT-SUPPORTED を返します。これは，負ではないリターン値です。

『 TX-INFO-AREA に設定した値は，システムのトランザクションマネジャでは使えません。』この場合の commit_return 特性は，変更されません。トランザクションマネジャは TX-COMMIT-COMPLETED または X-COMMIT-DECISION-LOGGED のどちらかは，COMMIT-RETURN の値として使えるようにしておく必要があります。『OpenTP1 の場合は，TX-COMMIT-RETURN です。』

エラー

次のような場合，TXSETCOMMITRET はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。負のリターン値が返った場合は，TXSETCOMMITRET は commit_return 特性を変更しません。

TX-EINVAL

TX-INFO-AREA に設定した値は，TX-COMMIT-DECISION-LOGGED でも，TX-COMMIT-COMPLETE でもありません。

TX-PROTOCOL-ERROR

TXSETCOMMITRET は，適切でないコンテクストで呼ばれました（例えば，呼び出し元が TXOPEN を呼び出していない）。

TX-FAIL

トランザクションマネジャで，回復できないエラーが起こりました。トランザクションマネジャが，すでにアプリケーションに代わって作業できないことが，エラーの原因です。

参照

TXCOMMIT，TXOPEN，TXINFORM

『OpenTP1 で使う場合の注意事項』

1.『TXSETCOMMITRET と，OpenTP1 の CBLDCTRN は併用できません。』

TXSETTIMEOUT

名称

transaction_timeout 特性の設定

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.
*
 01 TX-INFO-AREA.
    COPY TXINFDEF.

 PROCEDURE DIVISION.
 CALL "TXSETTIMEOUT" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能

TXSETTIMEOUT は，timeout に指定した値に transaction_timeout 特性を設定します。
transaction_timeout 特性の値には，トランザクションを完了させなければならない時間，つまりトランザクションをタイムアウトする時間を設定します。ここで示す時間とは，UAP が TXBEGIN を呼び出してから TXCOMMIT を呼び出すまで，または TXBEGIN を呼び出してから TXROLLBACK を呼び出すまでの時間を示します。

TXSETTIMEOUT は，呼び出し元がトランザクションモードにいるかどうかに関係なく呼び出せます。TXSETTIMEOUT がトランザクションモードから呼び出されても，新しいタイムアウトの値は，次のトランザクションまで効果を発揮しません。

transaction_timeout 特性の初期設定は，0（タイムアウトなし）です。

『システム定義の trn_expiration_time に値を指定している場合は，その値が初期設定になります。』

『データ領域』

『●TX-INFO-AREA

TRANSACTION-TIMEOUT で指定した秒数は，トランザクションがタイムアウトする時間を設定します。秒数は，そのシステムで定義された S9(9) COMP-5 の最大値の値が設定できます。』

『●TX-RETURN-STATUS

TXSETTIMEOUT の実行結果が返されます。』

リターン値

成功した場合，TXSETTIMEOUT は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』『transaction_timeout 特性の設定は，TX-INFO-AREA の値に変更されました。』

エラー

次のような場合，TXSETTIMEOUT はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。負のリターン値が返った場合は，TXSETTIMEOUT は transaction_timeout 特性を変更しません。

TX-EINVAL

TRANSACTION-TIMEOUT に設定した値が間違っています。

TX-PROTOCOL-ERROR

TXSETTIMEOUT は，適切でないコンテクストで呼ばれました（例えば，呼び出し元が TXOPEN をまだ呼び出していない）。

TX-FAIL

トランザクションマネジャで，回復できないエラーが起こりました。トランザクションマネジャが，すでにアプリケーションに代わって作業できないことがエラーの原因です。

参照

TXBEGIN，TXCOMMIT，TXOPEN，TXROLLBACK，TXINFORM

『OpenTP1 で使う場合の注意事項』

1.『TXSETTIMEOUT と，OpenTP1 の CBLDCTRN は併用できません。』

TXSETTRANCTL

名称

transaction_control 特性の設定

形式

```
 DATA DIVISION.
* Include TX definitions.
 01 TX-RETURN-STATUS.
    COPY TXSTATUS.
*
 01 TX-INFO-AREA.
    COPY TXINFDEF.

 PROCEDURE DIVISION.
 CALL "TXSETTRANCTL" USING TX-INFO-AREA TX-RETURN-STATUS.
```

機能

TXSETTRANCTL は，transaction_control 特性を，TRANSACTION-CONTROL に指定した値に設定します。transaction_control 特性は，TXCOMMIT，TXROLLBACK が呼び出し元にリターンする前に，新しいトランザクションが開始するかどうかを決定します。

TXSETTRANCTL は，呼び出し元がトランザクションモードかどうかに関係なく呼び出せます。設定した transaction_control 特性は，あとから呼び出される TXSETTRANCTL で変更されるまで引き継がれて，効力を発揮します。

transaction_control 特性の初期設定は，TX-UNCHAINED です。

『データ領域』

『●TRANSACTION-CONTROL

次の二つのどちらかの値が，TRANSACTION-CONTROL の妥当な設定です。

{TX-UNCHAINED|TX-CHAINED}』

- TX-UNCHAINED

  このフラグは，TXCOMMIT と TXROLLBACK が呼び出し元にリターンする前に，新たなトランザクションを開始しないことを示します。呼び出し元は，新しいトランザクションを開始する際には，TXBEGIN を発行しなければなりません。

- TX-CHAINED

  このフラグは，TXCOMMIT と TXROLLBACK が呼び出し元にリターンする前に，新しいトランザクションを開始することを示します。

リターン値

成功した場合，TXSETTRANCTL は TX-OK を返します。これは，負ではないリターン値です。『0 が返ります。』『transaction_control 特性の設定は，TRANSACTION-CONTROL の値に変更されました。』

エラー

次のような場合，TXSETTRANCTL はエラーリターンして，次のうちどれか一つの値を返します。これは，負のリターン値です。負のリターン値が返った場合は，TXSETTRANCTL は transaction_control 特性を変更しません。

TX-EINVAL

TRANSACTION-CONTROL に設定した値は，TX-UNCHAINED でも，TX-CHAINED でもありません。

TX-PROTOCOL-ERROR

TXSETTRANCTL は，適切でないコンテクストで呼ばれました（例えば，呼び出し元がまだ TXOPEN を呼び出していない）。

TX-FAIL

トランザクションマネジャで，回復できないエラーが起こりました。トランザクションマネジャが，すでにアプリケーションに代わって作業できないことがエラーの原因です。

参照

TXBEGIN，TXCOMMIT，TXOPEN，TXROLLBACK，TXINFORM

『OpenTP1 で使う場合の注意事項』

1.『TXSETTRANCTL と，OpenTP1 の CBLDCTRN の関数は併用できません。』

5 OpenTP1 の COBOL-UAP 作成用プログラムの文法（アソシエーションの状態の通知）

通信プロトコルに OSI TP を使ったクライアント/サーバ形態の通信では，通信イベント処理用 SPP が必要です。この章では，通信イベント処理用 SPP で使う COBOL-UAP 作成用プログラムと，受信する通信イベントの形式について説明します。

アソシエーションの操作（CBLDCXAT）

通信イベント処理用 SPP で使う，アソシエーションを操作する COBOL-UAP 作成用プログラムについて説明します。アソシエーションを操作する COBOL-UAP 作成用プログラムを次に示します。

- CBLDCXAT('CONNECT ') − アソシエーションの確立

アソシエーションを操作する COBOL-UAP 作成用プログラム（CBLDCXAT）は，TP1/Server Base の場合にだけ使えます。TP1/LiNK では，アソシエーションを操作する COBOL-UAP 作成用プログラムは使えません。

アソシエーションを操作する関数を呼び出せるのは，**通信イベント処理用 SPP だけ**です。ほかの OpenTP1 の UAP（SUP，SPP，MHP）では，アソシエーションを操作する関数は呼び出せません。

通信イベント処理用 SPP のユーザサービス定義の server_type オペランドには，必ず "betran" を指定してください。

COBOL-UAP 作成用プログラムの DATA DIVISION を設定すると，COBOL テンプレートでサンプルを使用できます。アソシエーションを操作するための COBOL テンプレートは，/BeTRAN/examples/COBOL/ディレクトリにある DCXAT.cbl に入っています。

CBLDCXAT('CONNECT ')

名称

アソシエーションの確立

形式

PROCEDURE DIVISION の指定

```
CALL ’CBLDCXAT’ 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 ’CONNECT ’.
   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 X(9).
   02 FILLER    PIC X(3).
   02 データ名D  PIC X(9).
```

機能

データ名 C に設定した XATMI 通信サービスに，データ名 D に設定したアソシエーションの確立を要求します。

CBLDCXAT('CONNECT ')は，相手システムにアソシエーションの確立要求を送信してからリターンします。アソシエーション確立の通知は，CBLDCXAT('CONNECT ')では受け取れません。

CBLDCXAT('CONNECT ')は，TP1/NET/OSI-TP-Extended を使った OSI TP 通信をするときにだけ使えます。

CBLDCXAT('CONNECT ')は，トランザクションの処理の範囲からでも範囲外からでも呼び出せます。

UAP で値を設定するデータ領域

●データ名 A

アソシエーションの確立を示す要求コードを「VALUE 'CONNECT△'」と設定します。

●データ名 C

アソシエーションの確立を要求する先の XATMI 通信サービス名を設定します。XATMI 通信サービス名として，XATMI 通信サービス定義の xat_invoke_server オペランドに指定する XATMI 通信サービス定義ファイル名を設定します。文字列の最後には空白を設定してください。

●データ名 D

確立するアソシエーション名を設定します。アソシエーション名として，TP1/NET/OSI-TP-Extended の定義のプロトコル固有定義 nettalccn オペランドの-c オプションに指定したコネクション名を設定します。文字列の最後には空白を設定してください。

●データ名 Z

0 を設定します。

OpenTP1 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。
04570	データ名に設定した値が間違っています。
04571	メモリが不足しました。
04572	CBLDCRPC('OPEN ')を呼び出していません。
04575	XATMI 通信サービスのアドレス情報を取得できません。
04576	XATMI 通信サービスは終了処理中です。
04577	XATMI 通信サービスへの送信時に，サービス要求が失敗しました。
04578	XATMI 通信サービスからの受信時に，サービス要求が失敗しました。コネクション確立要求が XATMI 通信サービスで実行している可能性があります。
04580	設定したアソシエーション名を定義していません。
04581	アソシエーションはすでに確立しています。
04582	アソシエーションは確立処理中です。
04583	アソシエーションは解放処理中です。
04584	アソシエーションは着呼モードのため，確立できません。

受信する通信イベントの形式

アソシエーションの状態を示す通信イベントの形式について説明します。通信イベントを受信するときは，**通信イベント処理用 SPP のサービスグループ名とサービス名を，XATMI 通信サービス定義**に指定しておきます。このとき，どのオペランドにサービスグループ名とサービス名を指定するかで，受け取れる通信イベントが異なります。

xat_aso_con_event_svcname オペランド

：アソシエーションの確立通知の通信イベント

xat_aso_discon_event_svcname オペランド

：アソシエーションの正常解放の通信イベント

xat_aso_failure_event_svcname オペランド

：アソシエーションの異常解放の通信イベント

複数のオペランドに同じサービスグループ名とサービス名を指定すると，一つの通信イベント処理用 SPP が複数の通信イベントを受信できるようにできます。

アソシエーションの状態を示す通信イベントの内容

アソシエーションの状態を示す通信イベントの内容を次に示します。

```
01 一意名1.
   02 データ名A  PIC S9(9) COMP.
   02 データ名B  PIC X(9).
   02 FILLER    PIC X(3).
   02 データ名C  PIC S9(9) COMP.
   02 データ名D  PIC 9(9) COMP.
   02 データ名E  PIC X(9).
   02 FILLER    PIC X(63).
```

●データ名 A

通信イベントを識別するコードが設定されます。( ) 内は該当するコードの数値表示（10 進数）を示します。

DCXAT_ASO_CONNECT（00000001）：アソシエーションの確立

DCXAT_ASO_DISCONNECT（00000002）：アソシエーションの正常解放

DCXAT_ASO_FAILURE（00000003）：アソシエーションの異常解放

●データ名 B

通信イベントで状態を通知するアソシエーションの，アソシエーション名が設定されます。

●データ名 C

確立したアソシエーションで，自システムが発呼となるか着呼となるかを示す値が設定されます。

( ) 内は該当するコードの数値表示（10 進数）を示します。

DCXAT_ASO_INIT（00000001）：自システムが発呼側

DCXAT_ASO_RESP（00000002）：自システムが着呼側

●データ名 D

アソシエーションが解放されたときの，理由コードが設定されます。( ) 内は該当するコードの数値表示（10 進数）を示します。

アソシエーションの正常解放の場合は次の値のどれかが設定されます。

00000001：コマンド実行によるアソシエーションの解放

00000005：XATMI によるアソシエーションの解放

00000007：相手システムからのアソシエーションの正常解放

00000008：TP 層によるアソシエーションの正常解放

アソシエーションの異常解放の場合は次の値のどれかが設定されます。

00000001：コマンド実行によるアソシエーションの強制解放

00000003：下位層の障害（回線障害，通信管理の障害など）

00000004：TP1/NET/OSI-TP-Extended の障害（プロトコルエラーなど）

00000005：XATMI 通信サービスによるアソシエーションの強制解放

00000006：アソシエーション確立の失敗

00000007：相手システムからのアソシエーションの強制解放

●データ名 E

XATMI 通信サービス名が設定されます。

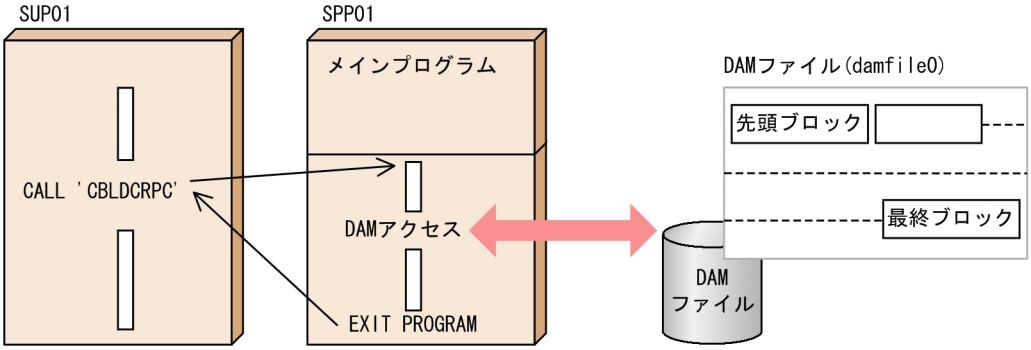
6 コーディング例

この章では，アプリケーションプログラム（UAP）のコーディング例について説明します。

6.1 クライアント／サーバ形態の UAP のコーディング例（SUP，SPP　DAM アクセス）

クライアント／サーバ形態の UAP 構成例を次の図に示します。

図 6-1　クライアント／サーバ形態の UAP 構成例（DAM アクセス）



（説明）

DAM ファイル damfile0 には，先頭ブロックに管理部があり，2 番目以降のブロックをデータレコードとしています。サービスの処理は，先頭ブロックを入力して更新したあと（CBLDCDAM('READ'), CBLDCDAM('REWT')），2 番目以降のブロックは CBLDCDAM('WRIT')で，直接 更新します。

ここでは，図に示した構成例のコーディング例を示します。

6.1.1 SUP の例

SUP のコーディング例を次に示します。

```
 10       *
 20       ************************************************************
 30       * SUP01                                                    *
 40       ************************************************************
 50       *
 60        IDENTIFICATION DIVISION.
 70       *
 80        PROGRAM-ID. MAIN.
 90       *
100       ************************************************************
110       *  データ領域の設定                                        *
120       ************************************************************
130       *
140        DATA DIVISION.
150        WORKING-STORAGE SECTION.
160        01  RPC-ARG1.
170            02 REQUEST      PIC X(8) VALUE SPACE.
180            02 STATUS-CODE  PIC X(5) VALUE SPACE.
190            02 FILLER       PIC X(3).
200            02 FLAGS        PIC S9(9) COMP VALUE ZERO.
```

```
210       *
220        01  RPC-ARG2.
230            02 REQUEST      PIC X(8) VALUE SPACE.
240            02 STATUS-CODE  PIC X(5) VALUE SPACE.
250            02 FILLER       PIC X(3).
260            02 FLAGS        PIC S9(9) COMP VALUE ZERO.
270            02 DESCRIPTOR   PIC S9(9) COMP VALUE ZERO.
280            02 S-NAME       PIC X(32) VALUE SPACE.
290            02 G-NAME       PIC X(32) VALUE SPACE.
300       *
310        01  RPC-ARG3.
320            02  SEND-DATA-LENG  PIC S9(9) COMP VALUE ZERO.
330            02  SEND-DATA       PIC X(32) VALUE SPACE.
340       *
350        01  RPC-ARG4.
360            02  RECEIVE-DATA-LENG PIC S9(9) COMP VALUE ZERO.
370            02  RECEIVE-DATA      PIC X(32) VALUE SPACE.
380       *
390        01  ADM-ARG1.
400            02  REQUEST      PIC X(8) VALUE SPACE.
410            02  STATUS-CODE  PIC X(5) VALUE SPACE.
420            02  FILLER       PIC X(3).
430            02  FLAGS        PIC S9(9) COMP VALUE ZERO.
440            02  FILLER       PIC X(3).
450       *
460        01  TRN-ARG1.
470            02  REQUEST      PIC X(8) VALUE SPACE.
480            02  STATUS-CODE  PIC X(5) VALUE SPACE.
490       *
500        PROCEDURE DIVISION.
510       *
520       ************************************************************
530       * RPC-OPEN(UAPの開始)                                      *
540       ************************************************************
550       *
560        MOVE 'OPEN' TO REQUEST OF RPC-ARG1.
570         MOVE ZERO   TO FLAGS   OF RPC-ARG1.
580         CALL 'CBLDCRPC' USING RPC-ARG1.
590             IF STATUS-CODE OF RPC-ARG1 NOT = '00000' THEN
600               DISPLAY 'SUP01:RPC-OPEN FAILED. CODE = '
610               STATUS-CODE OF RPC-ARG1
620               GO TO PROG-END
630             END-IF.
640        *
650        ************************************************************
660        * ADM-COMPLETE(ユーザサーバの開始処理完了の報告)       *
670        ************************************************************
680        *
690         MOVE 'COMPLETE' TO REQUEST OF ADM-ARG1.
700         CALL 'CBLDCADM' USING ADM-ARG1.
710             IF STATUS-CODE OF ADM-ARG1 NOT = '00000' THEN
720               DISPLAY 'SUP01:RPC-COMPLETE FAILED. CODE = '
730               STATUS-CODE OF ADM-ARG1
740               GO TO PROG-END
750             END-IF.
760        *
770        ************************************************************
780        * TRN-BEGIN(トランザクションの開始)                    *
```

```
 790        ********************************************************
 800        *
 810         MOVE 'BEGIN' TO REQUEST OF TRN-ARG1.
 820         CALL 'CBLDCTRN' USING TRN-ARG1.
 830             IF STATUS-CODE OF TRN-ARG1 NOT = '00000' THEN
 840               DISPLAY 'SUP01:TRN-BEGIN FAILED. CODE = '
 850               STATUS-CODE OF TRN-ARG1
 860               GO TO TRAN-END
 870             END-IF.
 880        *
 890        ********************************************************
 900        * RPC-CALL(遠隔サービスの要求)                         *
 910        ********************************************************
 920        *
 930         MOVE 'CALL'  TO REQUEST OF RPC-ARG2.
 940         MOVE 'SPP01' TO G-NAME  OF RPC-ARG2.
 950         MOVE 'SVR01' TO S-NAME  OF RPC-ARG2.
 960         MOVE 'SUP01:DATA OpenTP1' TO SEND-DATA OF RPC-ARG3.
 970         MOVE 32 TO SEND-DATA-LENG OF RPC-ARG3.
 980         MOVE 32 TO RECEIVE-DATA-LENG  OF RPC-ARG4.
 990         CALL 'CBLDCRPC' USING RPC-ARG2 RPC-ARG3 RPC-ARG4.
1000             IF STATUS-CODE OF RPC-ARG2 NOT = '00000' THEN
1010               DISPLAY 'SUP01:RPC-CALL RETURN CODE = '
1020               STATUS-CODE OF RPC-ARG2
1030               GO TO TRAN-END
1040             END-IF.
1050         DISPLAY 'SERVICE FUNCTION RETURN = ' RECEIVE-DATA.
1060         TRAN-END.
1070        *
1080        ********************************************************
1090        * TRN-UNCHAINED-COMMIT(非連鎖モードのコミット)         *
1100        ********************************************************
1110        *
1120         MOVE 'U-COMMIT' TO REQUEST OF TRN-ARG1.
1130         CALL 'CBLDCTRN' USING TRN-ARG1.
1140             IF STATUS-CODE OF TRN-ARG1 NOT = '00000' THEN
1150               DISPLAY 'SUP01:TRN-UNCHAINED-COMMIT FAILED. CODE = '
1160               STATUS-CODE OF TRN-ARG1
1170             END-IF.
1180         PROG-END.
1190        *
1200        ********************************************************
1210        * RPC-CLOSE(UAPの終了)                                 *
1220        ********************************************************
1230        *
1240         MOVE 'CLOSE' TO REQUEST OF RPC-ARG1.
1250         MOVE ZERO    TO FLAGS   OF RPC-ARG1.
1260         CALL 'CBLDCRPC' USING RPC-ARG1.
1270         DISPLAY 'SUP01:SUP PROCESS ENDED'.
1280         STOP RUN.
```

6.1.2 SPP の例（メインプログラム）

SPP のメインプログラムのコーディング例を次に示します。

```
      *
      ************************************************************
      * SPP01 メインプログラム                                   *
      ************************************************************
      *
       IDENTIFICATION DIVISION.
      *
       PROGRAM-ID. MAIN.
      *
      ************************************************************
      * データ領域の設定                                         *
      ************************************************************
      *
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01  FD-ID EXTERNAL.
           10  FD-SAVE      PIC S9(9) COMP.
       01  RPC-ARG1.
           02  REQ-CODE     PIC X(8) VALUE SPACE.
           02  STATUS-CODE  PIC X(5) VALUE SPACE.
           02  FILLER       PIC X(3).
           02  FLAGS        PIC S9(9) COMP.
       01  DAM-ARG1.
           02  REQUEST      PIC X(8) VALUE SPACE.
           02  STATUS-CODE  PIC X(5) VALUE SPACE.
           02  FILLER       PIC X(3).
           02  FILE-NAME    PIC X(8).
           02  FILLER       PIC S9(9) COMP.
           02  FILLER       PIC S9(9) COMP.
           02  FILDES       PIC S9(9) COMP VALUE ZERO.
           02  FILLER       PIC X(28).
       01  DAM-ARG2.
           02  ACCESS-CODE  PIC X(4).
           02  FLAG1        PIC X(1).
           02  FILLER       PIC X(1).
           02  FILLER       PIC X(1).
           02  FILLER       PIC X(1).
           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
      *
       PROCEDURE DIVISION.
      *
      ************************************************************
      * RPC-OPEN (UAPの開始)                                     *
      ************************************************************
      *
       MOVE 'OPEN' TO REQ-CODE OF RPC-ARG1.
       MOVE ZERO   TO FLAGS    OF RPC-ARG1.
       CALL 'CBLDCRPC' USING RPC-ARG1.
           IF STATUS-CODE OF RPC-ARG1 NOT = '00000' THEN
             DISPLAY 'SPP01:RPC-OPEN FAILED. CODE = '
             STATUS-CODE OF RPC-ARG1
             GO TO PROG-END
           END-IF.
      *
      ************************************************************
      * DAM-OPEN(論理ファイルのオープン)                         *
      ************************************************************
```

```
      *
       MOVE 'DCDAMSVC' TO REQUEST     OF DAM-ARG1.
       MOVE 'damfile0' TO FILE-NAME   OF DAM-ARG1.
       MOVE 'OPEN'     TO ACCESS-CODE OF DAM-ARG2.
       MOVE 'B'        TO FLAG1       OF DAM-ARG2.
       CALL 'CBLDCDAM' USING DAM-ARG1 DAM-ARG2.
           IF STATUS-CODE OF DAM-ARG1 NOT = '00000' THEN
             DISPLAY 'SPP01:DAM-OPEN FAILED. CODE = '
             STATUS-CODE OF DAM-ARG1
             GO TO DAM-END
           END-IF.
       MOVE FILDES TO FD-SAVE.
      *
      ************************************************************
      * RPC-MAINLOOP(SPPのサービス開始)                          *
      ************************************************************
      *
       DISPLAY 'SPP01: MAINLOOP START.'
       MOVE 'MAINLOOP' TO REQ-CODE OF RPC-ARG1.
       MOVE ZERO       TO FLAGS    OF RPC-ARG1.
       CALL 'CBLDCRSV' USING RPC-ARG1.
           IF STATUS-CODE OF RPC-ARG1 NOT = '00000' THEN
             DISPLAY 'SPP01:RPC-MAINLOOP FAILED. CODE ='
             STATUS-CODE OF RPC-ARG1
           END-IF.
       DAM-END.
      *
      ************************************************************
      * DAM-CLOSE(論理ファイルのクローズ)                        *
      ************************************************************
      *
       MOVE 'damfile0' TO FILE-NAME   OF DAM-ARG1.
       MOVE FD-SAVE    TO FILDES      OF DAM-ARG1.
       MOVE 'CLOS'     TO ACCESS-CODE OF DAM-ARG2.
       CALL 'CBLDCDAM' USING DAM-ARG1 DAM-ARG2.
           IF STATUS-CODE OF DAM-ARG1 NOT = '00000' THEN
             DISPLAY 'SPP01:DAM-CLOSE FAILED. CODE = '
             STATUS-CODE OF DAM-ARG1
           END-IF.
       PROG-END.
      *
      ************************************************************
      * RPC-CLOSE(UAPの終了)                                     *
      ************************************************************
      *
       MOVE 'CLOSE' TO REQ-CODE OF RPC-ARG1.
       MOVE ZERO    TO FLAGS    OF RPC-ARG1.
       CALL 'CBLDCRPC' USING RPC-ARG1.
      *
      ************************************************************
      * 終了処理                                                 *
      ************************************************************
      *
       DISPLAY 'SPP01:Good-by!'
       STOP RUN.
```

6.1.3　SPP の例（サービスプログラム）

SPP のサービスプログラムのコーディング例を次に示します。

```
 10       *
 20       ************************************************************
 30       * SPPサービスプログラム  SVR01                             *
 40       ************************************************************
 50       *
 60        IDENTIFICATION DIVISION.
 70       *
 80        PROGRAM-ID. SVR01.
 90       *
100       ************************************************************
110       * データ領域の設定                                         *
120       ************************************************************
130       *
140        DATA DIVISION.
150        WORKING-STORAGE SECTION.
160        01  FD-ID EXTERNAL.
170            10  FD-SAVE      PIC S9(9) COMP.
180        01  DAM-ARG1.
190            02  REQUEST      PIC X(8) VALUE SPACE.
200            02  STATUS-CODE  PIC X(5) VALUE SPACE.
210            02  FILLER       PIC X(3).
220            02  FILE-NAME    PIC X(8).
230            02  KEY-NO       PIC S9(9) COMP VALUE ZERO.
240            02  BUFFER-LEN   PIC S9(9) COMP VALUE ZERO.
250            02  FILDES       PIC S9(9) COMP VALUE ZERO.
260            02  FILLER       PIC X(28).
270        01  DAM-ARG2.
280            02  ACCESS-CODE  PIC X(4).
290            02  FLAG1        PIC X(1).
300            02  FLAG2        PIC X(1).
310            02  FLAG3        PIC X(1).
320            02  FLAG4        PIC X(1).
330            02  FLAGS        PIC S9(9) COMP VALUE ZERO.
340            02  DAMKEY.
350                03  FIRST-BLOCK-NO PIC S9(9) COMP.
360                03  LAST-BLOCK-NO  PIC S9(9) COMP.
370       *
380        01  CNTL-BUFFER.
390            02  W-COUNT      PIC S9(9) COMP.
400            02  RWT-DATA     PIC X(18) VALUE SPACE.
410            02  FILLER       PIC X(483) VALUE SPACE.
420       *
430        01  W-BUFFER.
440            02  FILLER       PIC X(504).
450       *
460          LINKAGE SECTION.
470          77  IN-DATA      PIC X(32).
480          77  IN-LENG      PIC S9(9) COMP.
490          77  OUT-DATA     PIC X(32).
500          77  OUT-LENG     PIC S9(9) COMP.
510       *
520          PROCEDURE DIVISION USING IN-DATA IN-LENG OUT-DATA OUT-LENG.
530          SVR01 SECTION.
```

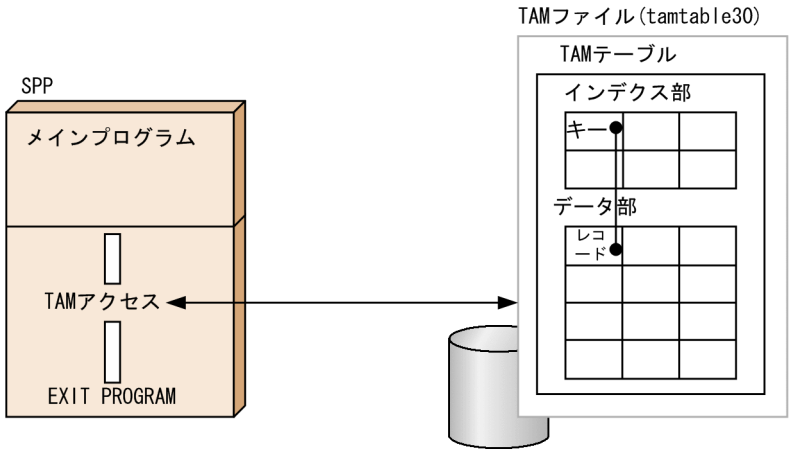
```
 540         DISPLAY  'SVR01:PROCEDURE START'.
 550       *
 560        ********************************************************
 570        * DAM-READ(論理ファイルからブロックの入力)             *
 580        ********************************************************
 590        *
 600         MOVE 'DCDAMSVC' TO REQUEST     OF DAM-ARG1.
 610         MOVE 'damfile0' TO FILE-NAME   OF DAM-ARG1.
 620         MOVE 1          TO KEY-NO      OF DAM-ARG1.
 630         MOVE 504        TO BUFFER-LEN  OF DAM-ARG1.
 640         MOVE FD-SAVE    TO FILDES      OF DAM-ARG1.
 650         MOVE 'READ'     TO ACCESS-CODE OF DAM-ARG2.
 660         MOVE 'M'        TO FLAG1       OF DAM-ARG2.
 670         MOVE SPACE      TO FLAG2       OF DAM-ARG2.
 680         MOVE 0          TO FIRST-BLOCK-NO OF DAMKEY.
 690         MOVE 0          TO LAST-BLOCK-NO  OF DAMKEY.
 700         CALL 'CBLDCDAM' USING DAM-ARG1 DAM-ARG2 CNTL-BUFFER.
 710             IF STATUS-CODE OF DAM-ARG1 NOT = '00000' THEN
 720                DISPLAY 'SVR01:DAM-READ FAILED. CODE ='
 730                STATUS-CODE OF DAM-ARG1
 740                MOVE 'SVR01: DAM READ FAILED' TO OUT-DATA
 750                MOVE 25 TO OUT-LENG
 760                GO TO PROG-END
 770             END-IF.
 780        *
 790        ********************************************************
 800        * DAM-WRITE(論理ファイルへブロックの出力)              *
 810        ********************************************************
 820        *
 830         DAM-WRITE.
 840         ADD 1 TO W-COUNT OF CNTL-BUFFER.
 850         MOVE 'DCDAMSVC' TO REQUEST     OF DAM-ARG1.
 860         MOVE 'damfile0' TO FILE-NAME   OF DAM-ARG1.
 870         MOVE 1          TO KEY-NO      OF DAM-ARG1.
 880         MOVE 504        TO BUFFER-LEN  OF DAM-ARG1.
 890         MOVE FD-SAVE    TO FILDES      OF DAM-ARG1.
 900         MOVE 'WRIT'     TO ACCESS-CODE OF DAM-ARG2.
 910         MOVE W-COUNT OF CNTL-BUFFER TO FIRST-BLOCK-NO OF DAMKEY.
 920         MOVE 0          TO LAST-BLOCK-NO OF DAMKEY.
 930         MOVE IN-DATA TO W-BUFFER.
 940         CALL 'CBLDCDAM' USING DAM-ARG1 DAM-ARG2 W-BUFFER.
 950             IF STATUS-CODE OF DAM-ARG1 NOT = '00000' THEN
 960               IF STATUS-CODE OF DAM-ARG1 = '01606' THEN
 970                 MOVE 0 TO W-COUNT OF CNTL-BUFFER
 980                 GO TO DAM-WRITE
 990               END-IF
1000               DISPLAY 'SVR01:DAM-WRITE FAILED. CODE = '
1010               STATUS-CODE OF DAM-ARG1
1020               MOVE 'SVR01:DAM WRITE FAILED' TO OUT-DATA
1030               MOVE 26 TO OUT-LENG
1040               GO TO PROG-END
1050             END-IF.
1060        *
1070        ********************************************************
1080        * DAM-REWRITE(論理ファイルのブロックの更新)            *
1090        ********************************************************
1100        *
1110         MOVE 'DCDAMSVC' TO REQUEST     OF DAM-ARG1.
```

```
1120         MOVE 'damfile0' TO FILE-NAME   OF DAM-ARG1.
1130         MOVE 1          TO KEY-NO      OF DAM-ARG1.
1140         MOVE 504        TO BUFFER-LEN  OF DAM-ARG1.
1150         MOVE FD-SAVE    TO FILDES      OF DAM-ARG1.
1160         MOVE 'REWT'     TO ACCESS-CODE OF DAM-ARG2.
1170         MOVE 'U'        TO FLAG1       OF DAM-ARG2.
1180         MOVE 0         TO FIRST-BLOCK-NO OF DAMKEY.
1190         MOVE 0         TO LAST-BLOCK-NO  OF DAMKEY.
1200         MOVE 'REWRITE COMPLETE' TO RWT-DATA OF CNTL-BUFFER.
1210         CALL 'CBLDCDAM' USING DAM-ARG1 DAM-ARG2 CNTL-BUFFER.
1220             IF STATUS-CODE OF DAM-ARG1 NOT = '00000' THEN
1230               DISPLAY 'SVR01:DAM-REWRITE FAILED. CODE = '
1240               STATUS-CODE OF DAM-ARG1
1250               MOVE 'SVR01:DAM REWRITE FAILED' TO OUT-DATA
1260               MOVE 28 TO OUT-LENG
1270               GO TO PROG-END
1280             END-IF.
1290         MOVE 'SVR01:PROCESS COMPLETE' TO OUT-DATA.
1300         MOVE 26 TO OUT-LENG.
1310         PROG-END.
1320         DISPLAY 'SVR01:Good-By!!'.
1330         END PROGRAM SVR01.
```

6.2　クライアント／サーバ形態の UAP のコーディング例（SPP　TAM アクセス）

クライアント／サーバ形態の UAP 構成例を次の図に示します。ここでは，SPP のコーディング例だけを掲載します。この SPP には，「6.1　クライアント／サーバ形態の UAP のコーディング例（SUP，SPP DAM アクセス）」と同じ SUP からサービス要求されるものとします。

図 6-2　クライアント／サーバ形態の UAP 構成例（TAM アクセス）



ここでは，図に示した構成例のコーディング例を示します。

6.2.1　SPP の例（メインプログラム）

SPP のメインプログラムのコーディング例を次に示します。

```
 10        *
 20        ************************************************************
 30        *  SPP01 メインプログラム                                  *
 40        ************************************************************
 50        *
 60         IDENTIFICATION DIVISION.
 70         PROGRAM-ID. MAIN.
 80        *
 90        ************************************************************
100        *  データ領域の設定                                        *
110        ************************************************************
120        *
130         DATA DIVISION.
140         WORKING-STORAGE SECTION.
150         01 RPC-ARG.
160            02 REQ-CODE       PIC  X(8)    VALUE SPACE.
170            02 STATUS-CODE    PIC  X(5)    VALUE SPACE.
180            02 FILLER          PIC  X(3).
190            02 FLAGS          PIC S9(9)   COMP.
200         PROCEDURE DIVISION.
210        *
```

```
220        ********************************************************
230        *  RPC-OPEN(UAPの開始)                                 *
240        ********************************************************
250        *
260             MOVE 'OPEN' TO REQ-CODE OF RPC-ARG.
270             MOVE ZERO   TO FLAGS  OF RPC-ARG.
280             CALL 'CBLDCRPC' USING RPC-ARG.
290             IF STATUS-CODE OF RPC-ARG NOT = '00000' THEN
300                 DISPLAY 'SPP01 : RPC-OPEN FAILED. CODE = '
310                         STATUS-CODE OF RPC-ARG
320                 GO TO PROG-END
330             END-IF.
340        *
350        ********************************************************
360        *  RPC-MAINLOOP(SPPのサービス開始)                     *
370        ********************************************************
380        *
390             MOVE 'MAINLOOP' TO REQ-CODE OF RPC-ARG.
400             MOVE ZERO     TO FLAGS    OF RPC-ARG.
410             CALL 'CBLDCRSV' USING RPC-ARG.
420             IF STATUS-CODE OF RPC-ARG NOT = '00000' THEN
430                 DISPLAY ' SPP01 : RPC-MAINLOOP FAILED. CODE = '
440                         STATUS-CODE OF RPC-ARG
450             END-IF.
460        *
470        ********************************************************
480        *  RPC-CLOSE(UAPの終了)                                *
490        ********************************************************
500        *
510             MOVE 'CLOSE   ' TO REQ-CODE OF RPC-ARG.
520             MOVE ZERO       TO FLAGS    OF RPC-ARG.
530             CALL 'CBLDCRPC' USING RPC-ARG.
540         PROG-END.
550        *
560        ********************************************************
570        *  終了処理                                            *
580        ********************************************************
590        *
600             DISPLAY ' SPP01 : GooD-by!' .
610             STOP RUN.
```

6.2.2 SPP の例（サービスプログラム）

SPP のサービスプログラムのコーディング例を次に示します。

```
 10   *
 20   ********************************************************
 30   *  SPPサービスプログラム  SVR01                        *
 40   ********************************************************
 50   *
 60    IDENTIFICATION DIVISION.
 70    PROGRAM-ID. SVR01.
 80   *
 90   ********************************************************
```

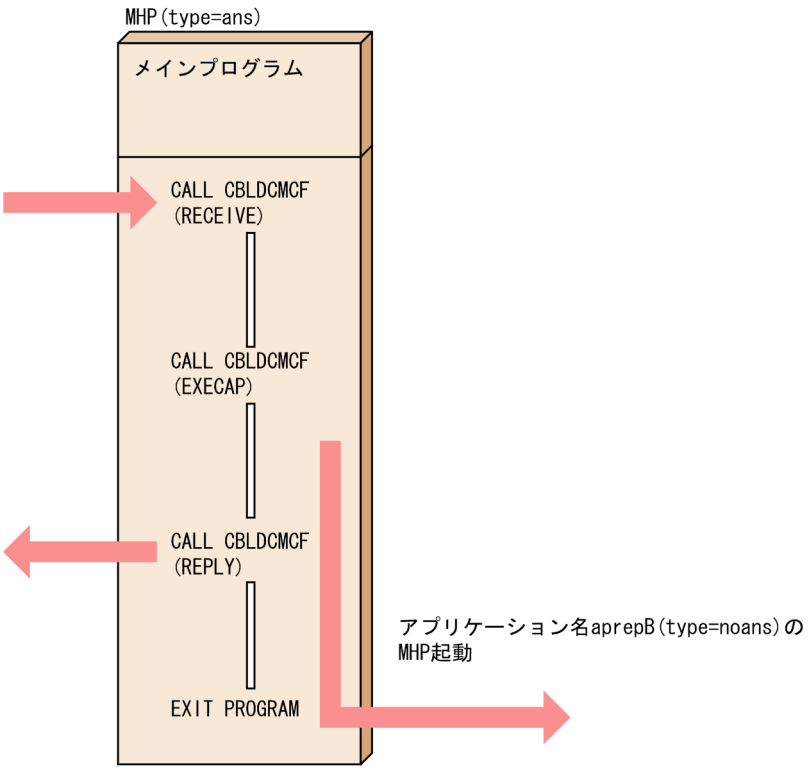
```
100   *  データ領域の設定                                        *
110   ************************************************************
120   *
130    DATA DIVISION.
140    WORKING-STORAGE SECTION.
150    01 TAM-ARG1.
160       02 REQ-CODE       PIC  X(4)    VALUE SPACE.
170       02 STATUS-CODE    PIC  X(5)    VALUE SPACE.
180       02 FILLER          PIC  X(3).
190       02 TABLE-NAME     PIC  X(32)   VALUE SPACE.
200       02 FILLER         PIC  X(68).
210       02 BUF-SIZE       PIC S9(4)   COMP VALUE ZERO.
220       02 FILLER         PIC  X(398).
230    01 READ-ARG1.
240       02 DML-KIND       PIC  X(4)    VALUE SPACE.
250       02 LOK-KIND       PIC  X(1)    VALUE SPACE.
260       02 FILLER         PIC  X(3).
270    01 WRITE-ARG1.
280       02 DML-KIND       PIC  X(4)    VALUE SPACE.
290       02 FILLER         PIC  X(4).
300    01 KEY-DATA1         PIC X(10)   VALUE X'0000000000000000001'.
310    01 KEY-DATA2         PIC X(10)   VALUE X'0000000000000000002'.
320    01 KEY-DATA4         PIC X(10)   VALUE X'0000000000000000004'.
330    01 KEY-ARG.
340       02 KEYNAME        PIC X(10)   VALUE SPACE.
350    01 W-BUFFER.
360       02 KEYNAME        PIC X(10)   VALUE SPACE.
370       02 DATAREA        PIC  X(118)  VALUE SPACE.
380    LINKAGE SECTION.
390       77 IN-DATA        PIC X(118).
400       77 IN-LENG        PIC S9(9)   COMP.
410       77 OUT-DATA       PIC X(32).
420       77 OUT-LENG       PIC S9(9)   COMP.
430    PROCEDURE DIVISION USING IN-DATA IN-LENG OUT-DATA OUT-LENG.
440        DISPLAY ' SVR01:PROCEDURE START' .
450   *
460   ************************************************************
470   *  TAM-READ(TAMテーブルの第1レコードを入力)                *
480   ************************************************************
490   *
500        MOVE 'tamtable30' TO TABLE-NAME  OF TAM-ARG1.
510        MOVE 128          TO BUF-SIZE    OF TAM-ARG1.
520        MOVE 'FCHU'       TO DML-KIND    OF READ-ARG1.
530        MOVE KEY-DATA1    TO KEY-ARG.
540         CALL 'CBLDCTAM' USING TAM-ARG1 READ-ARG1 KEY-ARG W-BUFFER.
550         IF STATUS-CODE OF TAM-ARG1 NOT = '00000' THEN
560             DISPLAY 'SVR01:TAM-READ FAILED. CODE = '
570                     STATUS-CODE OF TAM-ARG1
580             MOVE 'SVR01: TAM READ FAILED' TO OUT-DATA
590             MOVE 22 TO OUT-LENG
600             GO TO PROG-END
610         END-IF.
620    *
630    ************************************************************
640    * TAM-REWRITE(TAMテーブル第1レコードの入力が前提の更新)  *
650    ************************************************************
660    *
670         MOVE 'MFY '       TO DML-KIND    OF WRITE-ARG1.
```

```
 680         MOVE IN-DATA      TO DATAREA     OF W-BUFFER.
 690         CALL 'CBLDCTAM' USING TAM-ARG1 WRITE-ARG1 KEY-ARG W-BUFFER.
 700         IF STATUS-CODE OF TAM-ARG1 NOT = '00000' THEN
 710             DISPLAY 'SVR01:TAM-REWRITE FAILED. CODE = '
 720                     STATUS-CODE OF TAM-ARG1
 730             MOVE 'SVR01: TAM REWRITE FAILED' TO OUT-DATA
 740             MOVE 25 TO OUT-LENG
 750             GO TO PROG-END
 760         END-IF.
 770     *
 780     ******************************************************
 790     *  TAM-WRITE(TAMテーブルの第2レコードを更新)         *
 800     ******************************************************
 810     *
 820          MOVE 'MFY '       TO DML-KIND OF WRITE-ARG1.
 830          MOVE KEY-DATA2   TO KEY-ARG.
 840          MOVE KEY-DATA2   TO KEYNAME OF W-BUFFER.
 850          MOVE IN-DATA     TO DATAREA OF W-BUFFER.
 860          CALL 'CBLDCTAM' USING TAM-ARG1 WRITE-ARG1 KEY-ARG W-BUFFER.
 870          IF STATUS-CODE OF TAM-ARG1 NOT = '00000' THEN
 880              DISPLAY 'SVR01:TAM-WRITE FAILED. CODE = '
 890                      STATUS-CODE OF TAM-ARG1
 900              MOVE 'SVR01: TAM WRITE FAILED' TO OUT-DATA
 910              MOVE 23 TO OUT-LENG
 920              GO TO PROG-END
 930          END-IF.
 940     *
 950     ******************************************************
 960     *  TAM-DELETE(TAMテーブルの第4レコードを削除)        *
 970     ******************************************************
 980     *
 990          MOVE 'ERS '     TO DML-KIND OF WRITE-ARG1.
1000          MOVE KEY-DATA4 TO KEY-ARG.
1010          CALL 'CBLDCTAM' USING TAM-ARG1 WRITE-ARG1 KEY-ARG W-BUFFER.
1020          IF STATUS-CODE OF TAM-ARG1 NOT = '00000' THEN
1030              DISPLAY 'SVR01:TAM-DELETE FAILED. CODE = '
1040                      STATUS-CODE OF TAM-ARG1
1050              MOVE 'SVR01: TAM DELETE FAILED' TO OUT-DATA
1060              MOVE 24 TO OUT-LENG
1070          END-IF.
1080      PROG-END.
1090     *
1100     ******************************************************
1110     *  終了処理                                          *
1120     ******************************************************
1130     *
1140          DISPLAY 'SVR01:GooD-by!'.
1150          EXIT PROGRAM.
```

6.3　メッセージ送受信形態の UAP のコーディング例（MHP）

メッセージ送受信形態の UAP 構成例を次の図に示します。

図 6-3　メッセージ送受信形態の UAP 構成例（MHP）



ここでは，図に示した構成例のコーディング例を示します。

6.3.1　MHP の例（メインプログラム）

MHP のメインプログラムのコーディング例を次に示します。

```
 10       *
 20       ************************************************************
 30       * MHP メインプログラム                                      *
 40       ************************************************************
 50       *
 60        IDENTIFICATION DIVISION.
 70
 80        PROGRAM-ID. CBMAIN.
 90
100        ENVIRONMENT DIVISION.
110        CONFIGURATION SECTION.
120       *
130       ************************************************************
140       *     ワーク変数                                            *
150       ************************************************************
160       *
```

```
       DATA DIVISION.
       WORKING-STORAGE SECTION.
      *
      ********************************************************
      *  RPC-OPEN データ領域                                 *
      ********************************************************
      *
       01  ROPEN-PARM1.
         02  ROPEN-NAME         PIC X(8) VALUE 'OPEN    '.
         02  ROPEN-STATUS       PIC X(5).
         02  FILLER             PIC X(3).
         02  RO-FLG             PIC S9(9) COMP VALUE ZERO.
      *
      ********************************************************
      *  MCF-OPEN データ領域                                 *
      ********************************************************
      *
       01  MOPEN-PARM1.
         02  MOPEN-NAME         PIC X(8) VALUE 'OPEN    '.
         02  MOPEN-STATUS       PIC X(5).
         02  FILLER             PIC X(3).
         02  MO-FLG1            PIC S9(9) COMP VALUE ZERO.
         02  MO-RSV             PIC X(12) VALUE LOW-VALUE.
      *
      ********************************************************
      *  MCF-MAINLOOP データ領域                             *
      ********************************************************
      *
       01  MAIN-PARM1.
         02  MAIN-NAME         PIC X(8) VALUE 'MAINLOOP'.
         02  MAIN-STATUS       PIC X(5).
         02  FILLER            PIC X(3).
         02  M-RSV             PIC X(16) VALUE LOW-VALUE.
      *
      ********************************************************
      * MCF-CLOSE データ領域                                 *
      ********************************************************
       *
        01  MCLSE-PARM1.
          02  MCLSE-NAME         PIC X(8) VALUE 'CLOSE   '.
          02  MCLSE-STATUS       PIC X(5).
          02  MFILLER            PIC X(3).
          02  MC-FLG1            PIC S9(9) COMP VALUE ZERO.
          02  MC-RSV             PIC X(12) VALUE LOW-VALUE.
       *
       ********************************************************
       * RPC-CLOSE データ領域                                 *
       ********************************************************
       *
        01  RCLSE-PARM1.
          02  RCLSE-NAME         PIC X(8) VALUE 'CLOSE   '.
          02  RCLSE-STATUS       PIC X(5).
          02  FILLER             PIC X(3).
          02  RC-FLG             PIC S9(9) COMP VALUE ZERO.
       *
        PROCEDURE DIVISION.
       *
       ********************************************************
```

```
 750        *   RPC-OPEN(UAPの開始)                                  *
 760        ********************************************************
 770        *
 780           CALL 'CBLDCRPC' USING ROPEN-PARM1.
 790           IF ROPEN-STATUS IS NOT EQUAL TO '00000'
 800              GO TO RCLOS.
 810        *
 820        ********************************************************
 830        *   MCF-OPEN(MCF環境のオープン)                          *
 840        ********************************************************
 850        *
 860           CALL 'CBLDCMCF' USING MOPEN-PARM1.
 870           IF MOPEN-STATUS IS NOT EQUAL TO '00000'
 880              GO TO RCLOS.
 890        *
 900        ********************************************************
 910        *   MCF-MAINLOOP(MHPのサービス開始)                      *
 920        ********************************************************
 930        *
 940           CALL 'CBLDCMCF' USING MAIN-PARM1.
 950        *
 960        ********************************************************
 970        *  MCF-CLOSE(MCF環境のクローズ)                          *
 980        ********************************************************
 990        *
1000           CALL 'CBLDCMCF' USING MCLSE-PARM1.
1010        *
1020        ********************************************************
1030        *  RPC-CLOSE(UAPの終了)                                  *
1040        ********************************************************
1050        *
1060         RCLOS.
1070           CALL 'CBLDCRPC' USING RCLSE-PARM1.
1080        *
1090        ********************************************************
1100        *      終了処理                                         *
1110        ********************************************************
1120        *
1130           STOP RUN.
```

6.3.2 MHP の例（サービスプログラム）

MHP のサービスプログラムのコーディング例を次に示します。

```
  10        *
  20        ********************************************************
  30        *    MHPサービスプログラム                              *
  40        ********************************************************
  50        *
  60         IDENTIFICATION DIVISION.
  70
  80         PROGRAM-ID. SVRA.
  90
 100         ENVIRONMENT DIVISION.
```

```
110        CONFIGURATION SECTION.
120       *
130       ********************************************************
140       *    ワーク変数                                        *
150       ********************************************************
160       *
170        DATA DIVISION.
180        WORKING-STORAGE SECTION.
190       *
200       ********************************************************
210       *    MCF-RECEIVEデータ領域                             *
220       ********************************************************
230       *
240        01  RECV-PARM1.
250          02   RECV-NAME         PIC X(8) VALUE 'RECEIVE '.
260          02   RECV-STATUS       PIC X(5).
270          02   FILLER            PIC X(3).
280          02   FRST-ID           PIC X(4) VALUE 'FRST'
290          02   RE-RSV1           PIC X(4) VALUE SPACE.
300          02   DATE-ID           PIC 9(8).
310          02   TIME-ID           PIC 9(8).
320          02   RE-LENG           PIC 9(9) COMP VALUE 1024.
330          02   RE-RSV2           PIC X(4) VALUE SPACE.
340          02   RE-RSV3           PIC X(4) VALUE SPACE.
350          02   RE-RSV4           PIC X(4) VALUE SPACE.
360          02   RE-RSV5           PIC X(4) VALUE SPACE.
370          02   RE-RSV6           PIC X(8) VALUE SPACE.
380          02   RE-RSV7           PIC X(4) VALUE SPACE.
390          02   RE-RSV8           PIC X(8) VALUE SPACE.
400          02   RE-RSV9           PIC X(4) VALUE SPACE.
410          02   RE-RSV10          PIC 9(9) COMP VALUE ZERO.
420          02   RE-RSV11          PIC 9(9) COMP VALUE ZERO.
430          02   RE-RSV12          PIC X(1) VALUE SPACE.
440          02   RE-RSV13          PIC X(1) VALUE '1'.
450          02   RE-RSV14          PIC X(14) VALUE LOW-VALUE.
460        01  RECV-PARM2.
470          02   RE-RSV15          PIC X(4) VALUE SPACE.
480          02   TERM-ID           PIC X(8).
490          02   RE-RSV16          PIC X(8) VALUE SPACE.
500          02   RE-RSV17          PIC X(8) VALUE SPACE.
510          02   RE-RSV18          PIC X(28) VALUE LOW-VALUE.
520        01  RECV-PARM3.
530          02   RE-DATALENG       PIC 9(9) COMP.
540          02   RE-RSV19          PIC X(8).
550          02   RE-DATA           PIC X(1024).
560       *
570       ********************************************************
580       *    MCF-EXECAP データ領域                             *
590       ********************************************************
600       *
610        01   EXEC-PARM1.
620          02   EXEC-NAME         PIC X(8) VALUE 'EXECAP  '.
630          02   EXEC-STATUS       PIC X(5).
640          02   FILLER            PIC X(3).
650          02   EX-RSV1           PIC X(4) VALUE SPACE.
660          02   EX-RSV2           PIC X(4) VALUE SPACE.
670          02   EX-RSV3           PIC 9(8).
680          02   EX-RSV4           PIC 9(8).
```

```
 690          02   EX-RSV5           PIC 9(9) COMP VALUE ZERO.
 700          02   EX-EMI            PIC X(4) VALUE 'EMI '.
 710          02   EX-RSV6           PIC X(4) VALUE SPACE.
 720          02   EX-RSV7           PIC X(4) VALUE SPACE.
 730          02   EX-RSV8           PIC X(4) VALUE SPACE.
 740          02   EX-TIME           PIC X(8) VALUE '00000000'.
 750          02   EX-RSV9           PIC X(4) VALUE SPACE.
 760          02   EX-RSV10          PIC X(8) VALUE 'aprepB  '.
 770          02   EX-EXEC           PIC X(4) VALUE 'JUST'.
 780          02   EX-RSV11          PIC 9(9) COMP VALUE ZERO.
 790          02   EX-RSV12          PIC 9(9) COMP VALUE ZERO.
 800          02   EX-RSV13          PIC X(1) VALUE SPACE.
 810          02   EX-RSV14          PIC X(1) VALUE '1'.
 820          02   EX-RSV15          PIC X(14) VALUE LOW-VALUE.
 830        01   EXEC-PARM2.
 840          02   EX-RSV16          PIC X(4) VALUE SPACE.
 850          02   EX-RSV17          PIC X(8) VALUE SPACE.
 860          02   EX-RSV18          PIC X(8) VALUE SPACE.
 870          02   EX-RSV19          PIC X(8) VALUE SPACE.
 880          02   EX-RSV20          PIC X(28) VALUE LOW-VALUE.
 890        01   EXEC-PARM3.
 900          02   EX-DATALENG       PIC 9(9) COMP VALUE 16.
 910          02   EX-RSV21          PIC X(8).
 920          02   EX-DATA           PIC X(16) VALUE 'SVRA EXECAP DATA'.
 930       *
 940       ************************************************************
 950       *    MCF-REPLY データ領域                                  *
 960       ************************************************************
 970       *
 980        01  RPLY-PARM1.
 990          02   RPLY-NAME         PIC X(8) VALUE 'REPLY  '.
1000          02   RPLY-STATUS       PIC X(5).
1010          02   FILLER            PIC X(3).
1020          02   RP-RSV1           PIC X(4) VALUE SPACE.
1030          02   RP-RSV2           PIC X(4) VALUE SPACE.
1040          02   RP-RSV3           PIC 9(8).
1050          02   RP-RSV4           PIC 9(8).
1060          02   RP-RSV5           PIC 9(9) COMP VALUE ZERO.
1070          02   RP-EMI            PIC X(4) VALUE'EMI '.
1080          02   RP-RSV6           PIC X(4) VALUE SPACE.
1090          02   RP-RSV7           PIC X(4) VALUE SPACE.
1100          02   RP-RSV8           PIC X(4) VALUE SPACE.
1110          02   RP-RSV9           PIC X(8) VALUE SPACE.
1120          02   RP-RSV10          PIC X(4) VALUE SPACE.
1130          02   RP-RSV11          PIC X(8) VALUE SPACE.
1140          02   RP-RSV12          PIC X(4) VALUE SPACE.
1150          02   RP-RSV13          PIC 9(9) COMP VALUE ZERO.
1160          02   RP-RSV14          PIC 9(9) COMP VALUE ZERO.
1170          02   RP-RSV15          PIC X(1) VALUE SPACE.
1180          02   RP-RSV16          PIC X(1) VALUE '1'.
1190          02   RP-RSV17          PIC X(14) VALUE LOW-VALUE.
1200        01  RPLY-PARM2.
1210          02   RP-RSV18          PIC X(4) VALUE SPACE.
1220          02   RP-RSV19          PIC X(8) VALUE SPACE.
1230          02   RP-RSV20          PIC X(8) VALUE SPACE.
1240          02   RP-RSV21          PIC X(8) VALUE SPACE.
1250          02   RP-RSV22          PIC X(28) VALUE LOW-VALUE.
1260        01  RPLY-PARM3.
```

```
1270          02   RP-DATALENG       PIC 9(9) COMP VALUE 16.
1280          02   RP-RSV23          PIC X(8).
1290          02   RP-DATA           PIC X(16) VALUE 'SVRA REPLY DATA1'.
1300        *
1310        ************************************************************
1320        *    MCF-ROLLBACK データ領域                               *
1330        ************************************************************
1340        *
1350         01   RBK-PARM1.
1360          02   RBK-NAME          PIC X(8) VALUE 'ROLLBACK'.
1370          02   RBK-STATUS        PIC X(5).
1380          02   FILLER            PIC X(3).
1390          02   RBK-ACTION        PIC X(4) VALUE 'NRTN'.
1400          02   RBK-RSV1          PIC X(12) VALUE LOW-VALUE.
1410
1420         PROCEDURE DIVISION.
1430        *
1440        ************************************************************
1450        *    MCF-RECEIVE（メッセージの受信）                       *
1460        ************************************************************
1470        *
1480         CALL 'CBLDCMCF' USING RECV-PARM1 RECV-PARM2 RECV-PARM3.
1490         IF RECV-STATUS IS NOT EQUAL TO '00000'
1500        *
1510        ************************************************************
1520        *    MCF-ROLLBACK（エラー処理）                            *
1530        ************************************************************
1540        *
1550         CALL 'CBLDCMCF' USING RBK-PARM1.
1560        *
1570        ************************************************************
1580        *    MCF-EXECAP（アプリケーションプログラム起動）          *
1590        ************************************************************
1600        *
1610         CALL 'CBLDCMCF' USING EXEC-PARM1 EXEC-PARM2 EXEC-PARM3.
1620         IF EXEC-STATUS IS NOT EQUAL TO '00000'
1630        *
1640        ************************************************************
1650        *    MCF-ROLLBACK（エラー処理）                            *
1660        ************************************************************
1670        *
1680         CALL 'CBLDCMCF' USING RBK-PARM1.
1690        *
1700        ************************************************************
1710        *    MCF-REPLY（応答メッセージの送信）                     *
1720        ************************************************************
1730        *
1740         CALL 'CBLDCMCF' USING RPLY-PARM1 RPLY-PARM2 RPLY-PARM3.
1750         IF RPLY-STATUS IS NOT EQUAL TO '00000'
1760        *
1770        ************************************************************
1780        *    MCF-ROLLBACK（エラー処理）                            *
1790        ************************************************************
1800        *
1810         CALL 'CBLDCMCF' USING RBK-PARM1.
1820        *
1830        ************************************************************
1840        *    終了処理                                              *
```

```
1850       ************************************************************
1860       *
1870        EXIT PROGRAM.
```

6.3.3 MHP の例（サービスプログラム　DML の例）

MHP のサービスプログラムをデータ操作言語（DML）で作成した場合のコーディング例を次に示します。

```
  10       *
  20       ************************************************************
  30       *    MHPサービスプログラム                                 *
  40       ************************************************************
  50       *
  60        IDENTIFICATION DIVISION.
  70
  80        PROGRAM-ID. SVRA.
  90
 100        ENVIRONMENT DIVISION.
 110        CONFIGURATION SECTION.
 120       *
 130       ************************************************************
 140       *    ワーク変数                                            *
 150       ************************************************************
 160       *
 170        DATA DIVISION.
 180        WORKING-STORAGE SECTION.
 190       *
 200       ************************************************************
 210       *    メッセージ受信領域                                    *
 220       ************************************************************
 230       *
 240         01  RECV-AREA.
 250           02  RE-DATALENG       PIC 9(4) COMP VALUE 1028.
 260           02  RE-RSV1           PIC X(2).
 270           02  RE-DATA           PIC X(1024).
 280       *
 290       ************************************************************
 300       *    アプリケーション起動メッセージ領域                    *
 310       ************************************************************
 320       *
 330         01  SEND-PRO-AREA.
 340           02  PRO-DATALENG       PIC 9(4) COMP VALUE 20.
 350           02  PRO-RSV1           PIC X(2).
 360           02  PRO-DATA           PIC X(16) VALUE 'SVRA EXECAP DATA'.
 370       *
 380       ************************************************************
 390       *    応答メッセージ送信領域                                *
 400       ************************************************************
 410       *
 420         01  SEND-IO-AREA.
 430           02  IO-DATALENG       PIC 9(4) COMP VALUE 20.
 440           02  IO-RSV1           PIC X(2).
 450           02  IO-DATA           PIC X(16) VALUE 'SVRA REPLY DATA1'.
 460       *
```

```
470       ********************************************************
480       *    通信記述項                                        *
490       ********************************************************
500       *
510        COMMUNICATION SECTION.
520       *
530       ********************************************************
540       *    メッセージの受信                                  *
550       ********************************************************
560        *
570         CD RECV-INF
580           FOR INPUT
590           STATUS KEY IS        RE-STATUS
600           SYMBOLIC TERMINAL IS RE-TERMNAM
610           MESSAGE DATE IS      RE-DATE
620           MESSAGE TIME IS      RE-TIME.
630        *
640        ********************************************************
650        *    アプリケーションプログラム起動                    *
660        ********************************************************
670        *
680         CD SEND-PRO
690           FOR OUTPUT PROGRAM
700           STATUS KEY IS        SE-STATUS-PRO
710           SYMBOLIC TERMINAL IS SE-TERMNAM-PRO.
720        *
730        ********************************************************
740        *    応答メッセージの送信                              *
750        ********************************************************
760        *
770         CD SEND-IO
780           FOR I-O
790           STATUS KEY IS        SE-STATUS-IO
800           SYNCHRONOUS MODE IS  ASYNC.
810
820         PROCEDURE DIVISION.
830
840        *
850        ********************************************************
860        *    メッセージの受信                                  *
870        ********************************************************
880        *
890           RECEIVE RECV-INF
900                   FIRST SEGMENT
910                   INTO RECV-AREA.
920           IF RE-STATUS IS NOT EQUAL '00000'
930        *
940        ********************************************************
950        *    部分回復                                          *
960        ********************************************************
970        *
980           ROLLBACK WITH STOPPING.
990        *
1000       ********************************************************
1010       *    アプリケーションプログラム起動                    *
1020       ********************************************************
1030       *
1040          MOVE 'aprepB  ' TO SE-TERMNAM-PRO
```

```
1050           SEND SEND-PRO
1060                FROM SEND-PRO-AREA
1070                WITH EMI.
1080           IF SE-STATUS-PRO IS NOT EQUAL '00000'
1090         *
1100         ************************************************************
1110         *    部分回復                                              *
1120         ************************************************************
1130         *
1140           ROLLBACK WITH STOPPING.
1150         *
1160         ************************************************************
1170         *    応答メッセージの送信                                  *
1180         ************************************************************
1190         *
1200           SEND SEND-IO
1210                FROM SEND-IO-AREA
1220                WITH EMI.
1230           IF SE-STATUS-IO IS NOT EQUAL '00000'
1240         *
1250         ************************************************************
1260         *    部分回復                                              *
1270         ************************************************************
1280         *
1290           ROLLBACK WITH STOPPING.
1300         *
1310         ************************************************************
1320         *    終了処理                                              *
1330         ************************************************************
1340         *
1350           EXIT PROGRAM.
```

6.4　X/Open に準拠した UAP のコーディング例

6.4.1　XATMI インタフェースの例

(1)　リクエスト／レスポンス型サービスの通信の例

(a)　処理の概要

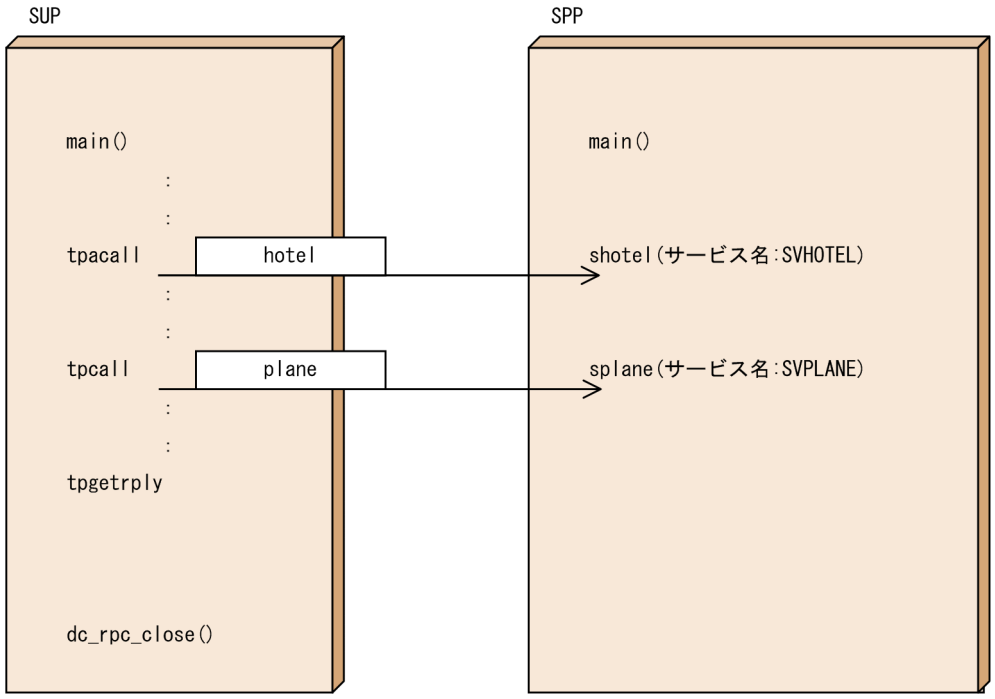
ここで示す例題の処理概要を，次に説明します。

[説明]宿泊施設の空き状況を調べるサービスと，飛行機の空き状況を調べるサービスを，SUP から呼びます。前者は非同期に，後者は同期的に応答を受信します。

(b)　UAP の構成

例題の UAP の構成を次の図に示します。

図 6-4　同期的に応答を受信するリクエスト／レスポンス型サービスの通信形態



(c)　通信に使う型付きバッファ

通信に使う型付きバッファの構造体を次に示します。

```
HOTEL.cbl
           05 RDATE            PIC S9(9) COMP-5.
           05 PLACE            PIC X(128).
           05 HNAME            PIC X(128).
           05 RSTATUS          PIC S9(9) COMP-5.
```

```
PLANE.cbl
           05 RDATE            PIC S9(9) COMP-5.
           05 DEST             PIC X(128).
           05 DEPARTURE        PIC S9(9) COMP-5.
           05 RSTATUS          PIC S9(9) COMP-5.
```

(d) SUP の例

- XATMI インタフェース定義の例

  リクエスト／レスポンス型サービスの例題で示す SUP の XATMI インタフェース定義を次に示します。

```
10  /* SUPのXATMIインタフェース定義の例（rrsupcb.defファイル）*/
20  called_servers = { "rrsppcb.def" };
```

- SUP のコーディング例

  リクエスト／レスポンス型サービスの例題で示す SUP のコーディング例を次に示します。

```
 10      *
 20      ************************************************************
 30      * SUPの例(rrsup.cblファイル)
 40      ************************************************************
 50      *
 60       IDENTIFICATION DIVISION.
 70      *
 80       PROGRAM-ID.  MAIN.
 90      *
100      ************************************************************
110      * データ領域の設定
120      ************************************************************
130      *
140       DATA DIVISION.
150       WORKING-STORAGE SECTION.
160      ************************************************************
170      * 変数の宣言
180      ************************************************************
190      ***** typed record for SVHOTEL *****
200       01  HOTEL-REQ.
210           COPY HOTEL.
220      ***** type infomation for SVHOTEL *****
230       01  HOTELTYPE-REC.
240           COPY TPTYPE.
250      ***** typed record for SVPLANE *****
260       01  PLANE-REQ.
270           COPY PLANE.
280      ***** type infomation for SVPLANE *****
290       01  PLANETYPE-REC.
300           COPY TPTYPE.
310      ***** WERRMSG *****
320       01  WERRMSG-REC.
330           COPY ERRMSG.
340      ***** service definition for SVHOTEL *****
350       01  HOTELDEF-REC.
360           COPY TPSVCDEF.
```

```
370      ***** service definition for SVPLANE *****
380       01  PLANEDEF-REC.
390           COPY TPSVCDEF.
400      ***** return record *****
410       01  STATUS-REC.
420           COPY TPSTATUS.
430      ***** working area is used for replies *****
440       01  WK-AREA    PIC X(264).
450      ***** redefine working area 1 *****
460       01  HOTEL-REP REDEFINES WK-AREA.
470           COPY HOTEL.
480      ***** redefine working area 2 *****
490       01  PLANE-REP REDEFINES WK-AREA.
500           COPY PLANE.
510      ***** redefine working area 3 *****
520       01  ERRMSG-REP REDEFINES WK-AREA.
530           COPY ERRMSG.
540      ***** typed infomation  *****
550       01  TYPE-REC.
560           COPY TPTYPE.
570      ***** others *****
580       01  WSTATUS PIC S9(9) COMP-5.
590      ***** dc_rpc_open *****
600       01  RPC-OP-ARG.
610           02  REQEST        PIC X(8) VALUE 'OPEN    '.
620           02  STATUS-CODE   PIC X(5) VALUE SPACE.
630           02  FILLER        PIC X(3).
640           02  FLAGS         PIC S9(9) COMP VALUE ZERO.
650      ***** dc_rpc_close *****
660       01  RPC-CL-ARG.
670           02  REQEST        PIC X(8) VALUE 'CLOSE   '.
680           02  STATUS-CODE   PIC X(5) VALUE SPACE.
690           02  FILLER        PIC X(3).
700           02  FLAGS         PIC S9(9) COMP VALUE ZERO.
710      *
720       01  ADM-ARG.
730           02    REQUEST     PIC X(8) VALUE 'COMPLETE'.
740           02    STATUS-CODE PIC X(5) VALUE SPACE.
750           02    FILLER      PIC X(3).
760           02    FLAGS       PIC S9(9) COMP VALUE ZERO.
770           02    FILLER      PIC X(3).
780      *
790       01  FLAG              PIC S9(9) COMP VALUE ZERO.
800      *
810       PROCEDURE DIVISION.
820      *
830      ************************************************************
840      * RPC-OPEN(UAPの開始)
850      ************************************************************
860      *
870       CALL 'CBLDCRPC' USING RPC-OP-ARG.
880           IF STATUS-CODE OF RPC-OP-ARG NOT = '00000' THEN
890             DISPLAY 'CLIENT: RPC-OPEN FAILED. CODE = '
900                      STATUS-CODE OF RPC-OP-ARG
910             GO TO PROG-END
920           END-IF.
930      *
940      ************************************************************
```

```
 950      * ADM-COMPLETE(ユーザサーバの開始処理完了の報告)
 960      ************************************************************
 970      *
 980       CALL 'CBLDCADM' USING ADM-ARG.
 990           IF STATUS-CODE OF ADM-ARG NOT = '00000' THEN
1000             DISPLAY 'CLIENT: ADM-COMPLETE FAILED. CODE = '
1010                     STATUS-CODE OF ADM-ARG
1020             GO TO PROG-END
1030           END-IF.
1040      *
1050      ************************************************************
1060      * TPACALL(サービスの要求 (SVHOTEL))
1070      ************************************************************
1080      *
1090      ***** set parameters *****
1100      *
1110      ***** set up HOTELDEF-REC *****
1120      *
1130       MOVE LOW-VALUES            TO HOTELDEF-REC.
1140       MOVE "SVHOTEL"             TO SERVICE-NAME OF HOTELDEF-REC.
1150      *
1160      ***** set up HOTELTYPE-REC *****
1170      *
1180       MOVE "X_COMMON"            TO REC-TYPE    OF HOTELTYPE-REC.
1190       MOVE "hotel"               TO SUB-TYPE    OF HOTELTYPE-REC.
1200       COMPUTE LEN OF HOTELTYPE-REC = FUNCTION LENGTH(HOTEL-REQ).
1210      *
1220      ***** set up HOTEL-REQ *****
1230      *
1240       MOVE 940415                TO RDATE        OF HOTEL-REQ.
1250       MOVE "SAPPRO"              TO PLACE        OF HOTEL-REQ.
1260       MOVE "PRINCE"              TO HNAME        OF HOTEL-REQ.
1270       MOVE 0                     TO RSTATUS      OF HOTEL-REQ.
1280      *
1290      ***** CALL TPACALL *****
1300       CALL "TPACALL" USING
1310           HOTELDEF-REC HOTELTYPE-REC HOTEL-REQ STATUS-REC.
1320           IF NOT TPOK OF STATUS-REC THEN
1330             DISPLAY 'CLIENT: SVHOTEL SERVICE REQ WAS FAIL:ERROR = '
1340                     TP-STATUS OF STATUS-REC
1350             GO TO PROG-END
1360           END-IF.
1370      *
1380       DISPLAY 'CLIENT: SVHOTEL SERVICE REQ WAS SUCCESS '.
1390      *
1400      *
1410      ************************************************************
1420      * TPCALL(サービスの要求 (SVPLANE))
1430      ************************************************************
1440      *
1450      ***** set parameters *****
1460      *
1470      ***** set up PLANEDEF-REC *****
1480      *
1490       MOVE LOW-VALUES            TO PLANEDEF-REC.
1500       MOVE "SVPLANE"             TO SERVICE-NAME OF PLANEDEF-REC.
1510      *
1520      ***** set up PLANETYPE-REC *****
```

```
1530      *
1540       MOVE "X_COMMON"            TO REC-TYPE    OF PLANETYPE-REC.
1550       MOVE "plane"               TO SUB-TYPE    OF PLANETYPE-REC.
1560       COMPUTE LEN OF PLANETYPE-REC = FUNCTION LENGTH(PLANE-REQ).
1570      *
1580      ***** set up PLANE-REQ *****
1590      *
1600       MOVE 940415                TO RDATE        OF PLANE-REQ.
1610       MOVE "CHITOSE"             TO DEST         OF PLANE-REQ.
1620       MOVE 1540                  TO DEPARTURE    OF PLANE-REQ.
1630       MOVE 0                     TO RSTATUS      OF PLANE-REQ.
1640      *
1650      *
1660      ***** set up TYPE-REC *****
1670      *
1680       MOVE "X_COMMON"            TO REC-TYPE     OF TYPE-REC.
1690       MOVE "plane"               TO SUB-TYPE     OF TYPE-REC.
1700       COMPUTE LEN OF TYPE-REC = FUNCTION LENGTH(WK-AREA).
1710      *
1720      ***** CALL TPCALL *****
1730       CALL "TPCALL" USING PLANEDEF-REC PLANETYPE-REC PLANE-REQ
1740            TYPE-REC WK-AREA STATUS-REC.
1750      *
1760      * FAILURE CASE
1770      *
1780           IF NOT TPOK OF STATUS-REC THEN
1790             DISPLAY 'CLIENT: SVPLANE SERVICE REQ WAS FAILED'
1800             DISPLAY 'CLIENT: TPCALL WAS FAILED:ERROR='
1810                      TP-STATUS OF STATUS-REC
1820             IF TPESVCFAIL OF STATUS-REC THEN
1830               MOVE ERRMESSAGE IN ERRMSG-REP
1840                    TO ERRMESSAGE OF WERRMSG-REC
1850               DISPLAY 'CLIENT: USER  CODE = '
1860                        ERRMESSAGE OF WERRMSG-REC
1870               GO TO PROG-END
1880             END-IF
1890             GO TO PROG-END
1900           END-IF.
1910      *
1920      * SUCCESS CASE
1930      *
1940       DISPLAY 'CLIENT: SVPLANE SERVICE REQ WAS SUCCESS '.
1950           MOVE RSTATUS IN PLANE-REP TO WSTATUS.
1960           IF WSTATUS = 1 THEN
1970             DISPLAY 'CLIENT: NO BORDING TICKET'
1980           ELSE
1990             DISPLAY 'CLIENT: A BORDING TICKET WAS FOUND'
2000           END-IF.
2010      *
2020      ************************************************************
2030      * TPGETRPLY(応答メッセージの受信)
2040      ************************************************************
2050      *
2060      ***** set parameters *****
2070      *
2080      ***** set up TYPE-REC *****
2090      *
2100       MOVE "X_COMMON"            TO REC-TYPE    OF TYPE-REC.
```

```
2110       MOVE "hotel"               TO SUB-TYPE    OF TYPE-REC.
2120       COMPUTE LEN OF TYPE-REC = FUNCTION LENGTH(WK-AREA).
2130      *
2140      ***** CALL TPGETRPLY *****
2150       CALL "TPGETRPLY" USING HOTELDEF-REC TYPE-REC WK-AREA
2160            STATUS-REC.
2170      *
2180      * FAILURE CASE
2190      *
2200           IF NOT TPOK OF STATUS-REC THEN
2210             DISPLAY 'CLIENT: SVHOTEL SERVICE RSP WAS FAILED '
2220             DISPLAY 'CLIENT: TPGETRPLY WAS FAILED:ERROR='
2230                     TP-STATUS OF STATUS-REC
2240             IF TPESVCFAIL OF STATUS-REC THEN
2250               MOVE ERRMESSAGE IN ERRMSG-REP
2260                    TO ERRMESSAGE OF WERRMSG-REC
2270               DISPLAY 'CLIENT: USER  CODE = '
2280                       ERRMESSAGE OF WERRMSG-REC
2290               GO TO PROG-END
2300             END-IF
2310             GO TO PROG-END
2320           END-IF.
2330      *
2340      * SUCCESS CASE
2350      *
2360       DISPLAY 'CLIENT: SVHOTEL SERVICE RSP WAS SUCCESS '.
2370           MOVE RSTATUS IN HOTEL-REP TO WSTATUS.
2380           IF WSTATUS = 1 THEN
2390             DISPLAY 'CLIENT: NO ROOM'
2400           ELSE
2410             DISPLAY 'CLIENT: A ROOM WAS FOUND'
2420           END-IF.
2430      *
2440      ************************************************************
2450      * 処理の終了
2460      ************************************************************
2470      *
2480       PROG-END.
2490      *
2500       DISPLAY 'CLIENT: SEE YOU LATER'
2510      *
2520      ************************************************************
2530      * RPC-CLOSE(UAPの終了)
2540      ************************************************************
2550      *
2560       CALL 'CBLDCRPC' USING RPC-CL-ARG.
2570      *
2580       STOP RUN.
```

- ユーザサービス定義の例

  リクエスト／レスポンス型サービスの例題で示す SUP のユーザサービス定義例を次に示します。

```
  10  #ユーザサービス定義の例(rrsupファイル)
  20  set  module              = "rrsup"
  30  set  receive_from        = none
```

```
 40  set  trn_expiration_time = 180
 50  set  trn_expiration_time_suspend = Y
```

(e) SPP の例

- XATMI インタフェース定義の例

  リクエスト／レスポンス型サービスの例題で示す SPP の XATMI インタフェース定義を次に示します。

```
 10    /* XATMIインタフェース定義の例（rrsppcb.defファイル）*/
 20    X_COMMON hotel {
 30        long   rdate;
 40        char   place[128];
 50        char   hname[128];
 60        long   rstatus;
 70    };
 80    X_COMMON plane {
 90        long   rdate;
100        char   dest[128];
110        long   departure;
120        long   rstatus;
130    };
140    X_COMMON errmsg {
150        char   errmessage[128];
160    };
170    service SHOTEL(X_COMMON hotel) ;
180    service SPLANE(X_COMMON plane) ;
```

- SPP のコーディング例（メインプログラム）

  リクエスト／レスポンス型サービスの例題で示す SPP のコーディング例（メインプログラム）を次に示します。

```
 10      *
 20      ************************************************************
 30      * SPPの例(rrspp.cblファイル)
 40      ************************************************************
 50      *
 60       IDENTIFICATION DIVISION.
 70      *
 80       PROGRAM-ID.     MAIN.
 90      *
100      ************************************************************
110      * データ領域の設定
120      ************************************************************
130      *
140       DATA DIVISION.
150       WORKING-STORAGE SECTION.
160       01  RPC-OP-ARG.
170           02  REQEST       PIC X(8) VALUE 'OPEN    '.
180           02  STATUS-CODE  PIC X(5) VALUE SPACE.
190           02  FILLER       PIC X(3).
200           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
210      *
220       01  RPC-CL-ARG.
230           02  REQEST       PIC X(8) VALUE 'CLOSE   '.
```

```
240           02  STATUS-CODE  PIC X(5) VALUE SPACE.
250           02  FILLER       PIC X(3).
260           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
270      *
280       01  RSV-ARG.
290           02  REQUEST      PIC X(8) VALUE 'MAINLOOP'.
300           02  STATUS-CODE  PIC X(5) VALUE SPACE.
310           02  FILLER       PIC X(3).
320           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
330      *
340       PROCEDURE DIVISION.
350      *
360      ************************************************************
370      * RPC-OPEN(UAPの開始)
380      ************************************************************
390      *
400       CALL 'CBLDCRPC' USING RPC-OP-ARG.
410           IF STATUS-CODE OF RPC-OP-ARG NOT = '00000' THEN
420             DISPLAY 'SERVER: RPC-OPEN FAILED. CODE = '
430                     STATUS-CODE OF RPC-OP-ARG
440             GO TO PROG-END
450           END-IF.
460      *
470      ************************************************************
480      * RPC-MAINLOOP(SPPサービスの開始)
490      ************************************************************
500      *
510       DISPLAY 'SERVER: ENTERING MAINLOOP...'
520       CALL 'CBLDCRSV' USING RSV-ARG.
530           IF STATUS-CODE OF RSV-ARG NOT = '00000' THEN
540             DISPLAY 'SERVER: RPC-MAINLOOP FAILED. CODE = '
550                     STATUS-CODE OF RSV-ARG
560           END-IF.
570      *
580      ************************************************************
590      * プログラムの終わり
600      ************************************************************
610       PROG-END.
620      *
630      ************************************************************
640      * RPC-CLOSE(UAPの終了)
650      ************************************************************
660      *
670       CALL 'CBLDCRPC' USING RPC-CL-ARG.
680      *
690      ************************************************************
700      * 処理の終了
710      ************************************************************
720      *
730       STOP RUN.
```

- SPP のコーディング例（サービスプログラム）

リクエスト／レスポンス型サービスの例題で示す SPP のコーディング例（サービスプログラム）を次に示します。

＜SVHOTEL サービス用のコーディング例＞

```
      *
      ************************************************************
      * SPPのサービス関数例(shotel.cblファイル)
      ************************************************************
      *
       IDENTIFICATION DIVISION.
      *
       PROGRAM-ID.     SHOTEL.
      *
      ************************************************************
      * データ領域の設定
      ************************************************************
      *
       DATA DIVISION.
       WORKING-STORAGE SECTION.
      *
      ************************************************************
      * 変数の宣言
      ************************************************************
      *
      * TPSVCDEF record
      *
       01  SVCDEF-REC.
           COPY TPSVCDEF.
      *
      * TPTYPE record
      *
       01  TYPE-REC.
           COPY TPTYPE.
      *
      * TPSTATUS record
      *
       01  STATUS-REC.
           COPY TPSTATUS.
      *
      * TPSVCRET record
      *
       01  SVCRET-REC.
           COPY TPSVCRET.
      *
      * WK-AREA is where service requests are read into
      *
       01  WK-AREA PIC X(264).
      *
       01  HOTEL-REC REDEFINES WK-AREA.
           COPY HOTEL.
      *
      ************************************************************
      * shotel processing
      ************************************************************
      *
       PROCEDURE DIVISION.
      *
      ***** set length *****
       COMPUTE LEN OF TYPE-REC = FUNCTION LENGTH(WK-AREA).
```

```
560      *
570      ************************************************************
580      * TPSVCSTART
590      ************************************************************
600      *
610       CALL "TPSVCSTART" USING
620            SVCDEF-REC TYPE-REC WK-AREA STATUS-REC.
630      *
640      * Shotel return status=0 if the specified hotel can be made
650      * a reservation. Shotel return status=1 if there are no room
660      * in that specified hotel.
670      *   In this case, shotel return status=1 because there are no
680      * room.
690      *
700        MOVE 1 TO RSTATUS IN HOTEL-REC.
710      *
720      ************************************************************
730      * TPRETURN
740      ************************************************************
750      *
760       SET TPSUCCESS OF SVCRET-REC TO TRUE.
770       MOVE 1 TO APPL-CODE OF SVCRET-REC.
780      *
790       COPY TPRETURN
800            REPLACING TPSVCRET-REC BY SVCRET-REC
810                      TPTYPE-REC BY TYPE-REC
820                      DATA-REC BY WK-AREA.
830      *
840      ************************************************************
850      * 処理の終了
860      ************************************************************
870      *
880      *
890       END PROGRAM SHOTEL.
```

＜SVPLANE サービス用のコーディング例＞

```
 10      *
 20      ************************************************************
 30      * SPPのサービス関数例(splane.cblファイル)
 40      ************************************************************
 50      *
 60       IDENTIFICATION DIVISION.
 70      *
 80       PROGRAM-ID.     SPLANE.
 90      *
100      ************************************************************
110      * データ領域の設定
120      ************************************************************
130      *
140       DATA DIVISION.
150       WORKING-STORAGE SECTION.
160      *
170      ************************************************************
180      * 変数の宣言
190      ************************************************************
200      *
```

```
210      * TPSVCDEF record
220      *
230       01  SVCDEF-REC.
240           COPY TPSVCDEF.
250      *
260      * TPTYPE record
270      *
280       01  TYPE-REC.
290           COPY TPTYPE.
300      *
310      * TPSTATUS record
320      *
330       01  STATUS-REC.
340           COPY TPSTATUS.
350      *
360      * TPSVCRET record
370      *
380       01  SVCRET-REC.
390           COPY TPSVCRET.
400      *
410      * WK-AREA is where service requests are read into
420      *
430       01  WK-AREA PIC X(264).
440      *
450       01  PLANE-REC REDEFINES WK-AREA.
460           COPY PLANE.
470      *
480      ************************************************************
490      * splane processing
500      ************************************************************
510      *
520       PROCEDURE DIVISION.
530      *
540      ***** set length *****
550       COMPUTE LEN OF TYPE-REC = FUNCTION LENGTH(WK-AREA).
560      *
570      ************************************************************
580      * TPSVCSTART
590      ************************************************************
600      *
610       CALL "TPSVCSTART" USING
620            SVCDEF-REC TYPE-REC WK-AREA STATUS-REC.
630      *
640      * Splane return status=0 if the specified plane can be made
650      * a reservation. Splane return status=1 if there aren't any
660      * ticket in that specified plane.
670      *   In this case, splane return status=1 because there are no
680      * room.
690      *
700        MOVE 1 TO RSTATUS IN PLANE-REC.
710      *
720      ************************************************************
730      * TPRETURN
740      ************************************************************
750      *
760       SET TPSUCCESS OF SVCRET-REC TO TRUE.
770       MOVE 0 TO APPL-CODE OF SVCRET-REC.
780      *
```

```
790       COPY TPRETURN
800            REPLACING TPSVCRET-REC BY SVCRET-REC
810                      TPTYPE-REC BY TYPE-REC
820                      DATA-REC BY WK-AREA.
830      *
840      ************************************************************
850      * 処理の終了
860      ************************************************************
870      *
880      *
890       END PROGRAM SPLANE.
```

- ユーザサービス定義の例

  リクエスト／レスポンス型サービスの例題で示す SPP のユーザサービス定義例を次に示します。

```
10  #ユーザサービス定義の例(rrsppファイル)
20  set    service_group     = "rrspp_svg"
30  set    module            = "rrspp"
40  set    service           = "SVHOTEL=SHOTEL","SVPLANE=SPLANE"
50  set    trn_expiration_time = 180
60  set    trn_expiration_time_suspend = Y
70  set    server_type = "xatmi_cbl"
```

(2) 会話型サービスの通信の例

(a) 処理の概要

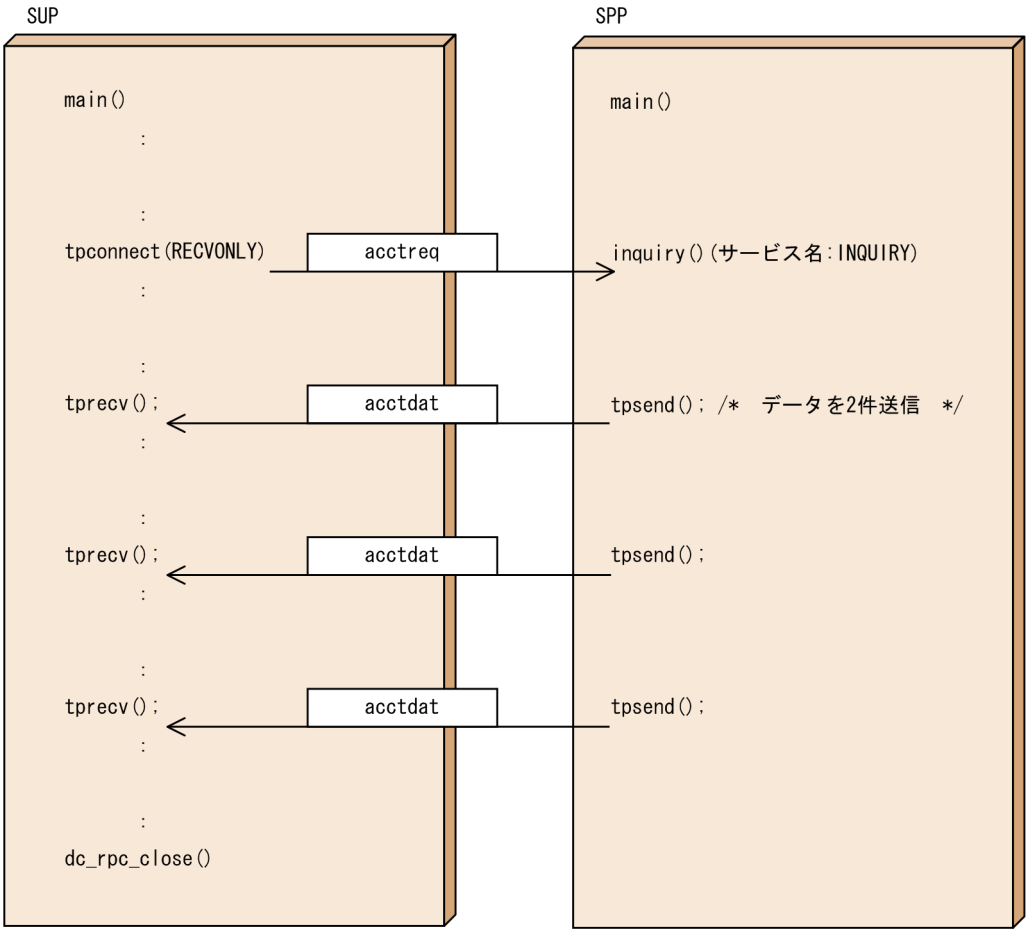
ここで示す例題の処理概要を，次に説明します。

[説明] acctreq 構造体の構造を持つ型付きバッファサービスでサービスプログラムを起動します。acctreq のメンバは口座番号の上限と下限を示します。サービスプログラムでは，この範囲にある口座データを acctreq 構造体の構造を持つ型付きバッファに設定して，会話のオリジネータに送信します。

(b) UAP の構成

例題の UAP の構成を次の図に示します。

図 6-5　会話型サービスの通信形態



(c)　通信に使う型付きバッファ

通信に使う型付きバッファの構造体を次に示します。

●サービスプログラム起動時のデータ

```
ACCTREQ.cbl
           05 UPPERNO          PIC S9(9) COMP-5.
           05 LOWERNO          PIC S9(9) COMP-5.
```

●会話型サービスの通信をするときのデータ

```
           05 ACCTNO           PIC S9(9) COMP-5.
           05 NAME             PIC X(128).
           05 AMOUNT           PIC S9(4) COMP-5.
           05 FILLER           PIC X(2).
```

(d)　SUP の例

- XATMI インタフェース定義の例

  会話型サービスの例題で示す SUP の XATMI インタフェース定義を次に示します。

```
10  /*  SUPのXATMIインタフェース定義の例（cvsupcb.defファイル）*/
20  called_servers = { ”cvsppcb.def” } ;
```

- SUP のコーディング例

  会話型サービスの例題で示す SUP のコーディング例を次に示します。

```
10      *
20      ************************************************************
30      * SUPの例（convsup.cblファイル）
40      ************************************************************
50      *
60       IDENTIFICATION DIVISION.
70      *
80       PROGRAM-ID. MAIN.
90      *
100     ************************************************************
110     * データ領域の設定
120     ************************************************************
130     *
140      DATA DIVISION.
150      WORKING-STORAGE SECTION.
160     ************************************************************
170     * 定数の宣言
180     ************************************************************
190     ************************************************************
200     * 変数の宣言
210     ************************************************************
220     ***** typed record for INQUARY when inquary service *****
230      01  ACCTREQ-REC.
240          COPY ACCTREQ.
250     ***** type infomation for INQUARY *****
260      01  ACCTREQTYPE-REC.
270          COPY TPTYPE.
280     ***** service definition for INQUARY *****
290      01  ACCTREQDEF-REC.
300          COPY TPSVCDEF.
310     ***** return record *****
320      01  STATUS-REC.
330          COPY TPSTATUS.
340     ***** received record between INQUARY and CONVSUP
350      01  ACCTDATA-REQ.
360          COPY ACCTDATA.
370     **** type information received record between INQUARY and CONVSUP
380      01  ACCTDATATYPE-REC.
390          COPY TPTYPE.
400     ***** service definition for INQUARY *****
410      01  ACCTDATADEF-REC.
420          COPY TPSVCDEF.
430     ***** working area is used for replies *****
440      01  WK-AREA    PIC X(136).
450     ***** redefine working area 1 *****
460      01  ACCTREQ-REP REDEFINES WK-AREA.
470          COPY ACCTREQ.
480     ***** redefine working area 2 *****
490      01  ACCTDATA-REP REDEFINES WK-AREA.
500          COPY ACCTDATA.
```

```
      ***** typed infomation  *****
       01  TYPE-REC.
           COPY TPTYPE.
      ***** others *****
       01  WSTATUS PIC S9(9) COMP-5.
      ***** dc_rpc_open *****
       01  RPC-OP-ARG.
           02  REQEST        PIC X(8) VALUE 'OPEN    '.
           02  STATUS-CODE   PIC X(5) VALUE SPACE.
           02  FILLER        PIC X(3).
           02  FLAGS         PIC S9(9) COMP VALUE ZERO.
      ***** dc_rpc_close *****
       01  RPC-CL-ARG.
           02  REQEST        PIC X(8) VALUE 'CLOSE   '.
           02  STATUS-CODE   PIC X(5) VALUE SPACE.
           02  FILLER        PIC X(3).
           02  FLAGS         PIC S9(9) COMP VALUE ZERO.
      ***** dc_adm_complete ****
       01  ADM-ARG.
           02    REQUEST     PIC X(8) VALUE 'COMPLETE'.
           02    STATUS-CODE PIC X(5) VALUE SPACE.
           02    FILLER      PIC X(3).
           02    FLAGS       PIC S9(9) COMP VALUE ZERO.
           02    FILLER      PIC X(3).
      *
       01  FLAG              PIC S9(9) COMP VALUE ZERO.
      *
      ****** for TX interface *****
      *
       01  TX-RETURN-STATUS.
           COPY TXSTATUS.
      *
       01  RS REDEFINES TX-RETURN-STATUS.
           05 RSVAL         PIC S9(9) COMP-5.
      *
       01 TX-INFO-AREA.
           COPY TXINFDEF.
      *
       PROCEDURE DIVISION.
      *
      ************************************************************
      * RPC-OPEN(UAPの開始)
      ************************************************************
      *
       CALL 'CBLDCRPC' USING RPC-OP-ARG.
           IF STATUS-CODE OF RPC-OP-ARG NOT = '00000' THEN
             DISPLAY 'CLIENT: RPC-OPEN FAILED. CODE = '
                      STATUS-CODE OF RPC-OP-ARG
             GO TO PROG-END
           END-IF.
      *
      ************************************************************
      * TXOPEN(リソースマネジャのオープン)
      ************************************************************
      *
       CALL "TXOPEN" USING TX-RETURN-STATUS.
           IF RSVAL OF RS NOT = 0 THEN
             DISPLAY 'CLIENT:TX-OPEN FAILED. CODE = '
```

```
1090             RSVAL OF RS
1100             GO TO PROG-END
1110           END-IF.
1120      *
1130      ************************************************************
1140      * TX-SET-TRANSACTION-TIMEOUT(トランザクション監視時間の設定)
1150      ************************************************************
1160      *
1170       MOVE 180 TO TRANSACTION-TIMEOUT OF TX-INFO-AREA.
1180       CALL "TXSETTIMEOUT" USING TX-INFO-AREA TX-RETURN-STATUS.
1190           IF RSVAL OF RS NOT = 0 THEN
1200             DISPLAY 'CLIENT:TX-SET-TRANSACTION-TIMEOUT FAILED. CODE = '
1210             RSVAL OF RS
1220             GO TO PROG-END
1230           END-IF.
1240      *
1250      ************************************************************
1260      * ADM-COMPLETE(ユーザサーバの開始処理完了の報告)
1270      ************************************************************
1280      *
1290       CALL 'CBLDCADM' USING ADM-ARG.
1300           IF STATUS-CODE OF ADM-ARG NOT = '00000' THEN
1310             DISPLAY 'CLIENT: ADM-COMPLETE FAILED. CODE = '
1320                      STATUS-CODE OF ADM-ARG
1330             GO TO PROG-END
1340           END-IF.
1350      *
1360      ************************************************************
1370      * TX-SET-TRANSACTION-CONTROL(非連鎖モード設定)
1380      ************************************************************
1390      *
1400       MOVE 0 TO TRANSACTION-CONTROL OF TX-INFO-AREA.
1410       CALL "TXSETTRANCTL" USING TX-INFO-AREA TX-RETURN-STATUS.
1420           IF RSVAL OF RS NOT = 0 THEN
1430             DISPLAY 'CLIENT:TX-SET-TRANSACTION-CONTROL FAILED. CODE ='
1440             RSVAL OF RS
1450           END-IF.
1460      *
1470      ************************************************************
1480      * TPXBEGIN(トランザクションの始まり)
1490      ************************************************************
1500      *
1510       CALL "TXBEGIN" USING TX-RETURN-STATUS.
1520           IF RSVAL OF RS NOT = 0 THEN
1530             DISPLAY 'CLIENT:TX-BEGIN FAILED. CODE ='
1540             RSVAL OF RS
1550             GO TO PROG-END
1560           END-IF.
1570      *
1580      ************************************************************
1590      * TPCONNECT(サービス要求 (INQUARY))
1600      ************************************************************
1610      *
1620      ***** set parameters *****
1630      *
1640      ***** set up ACCTREQDEF-REC *****
1650      *
1660       MOVE LOW-VALUES            TO ACCTREQDEF-REC.
```

```
1670       MOVE 1                     TO TPSENDRECV-FLAG OF ACCTREQDEF-REC.
1680       MOVE "INQUIRY"             TO SERVICE-NAME    OF ACCTREQDEF-REC.
1690      *
1700      ***** set up ACCTREQTYPE-REC *****
1710      *
1720       MOVE "X_COMMON"            TO REC-TYPE    OF ACCTREQTYPE-REC.
1730       MOVE "acctreq"             TO SUB-TYPE    OF ACCTREQTYPE-REC.
1740       COMPUTE LEN OF ACCTREQTYPE-REC = FUNCTION LENGTH(ACCTREQ-REC).
1750      *
1760      ***** set up ACCTREQ-REC *****
1770      *
1780       MOVE "100000000" TO LOWERNO OF ACCTREQ-REC.
1790       MOVE "200000000" TO UPPERNO OF ACCTREQ-REC.
1800      *
1810      ***** CALL TPCONNECT *****
1820       CALL "TPCONNECT" USING
1830           ACCTREQDEF-REC ACCTREQTYPE-REC ACCTREQ-REC STATUS-REC.
1840           IF NOT TPOK OF STATUS-REC THEN
1850             DISPLAY 'CLIENT: INQUARY SERVICE REQ WAS FAIL. CODE = '
1860                     TP-STATUS OF STATUS-REC
1870             DISPLAY  'CLIENT:TX-ROLLBACK STARTED'
1880             CALL "TXROLLBACK" USING TX-RETURN-STATUS
1890             DISPLAY  'CLIENT:TX-ROLLBACK ENDED'
1900             IF RSVAL OF RS NOT = 0 THEN
1910               DISPLAY 'CLIENT:TX-ROLLBACK FAILED. CODE ='
1920               RSVAL OF RS
1930             END-IF
1940             GO TO PROG-END
1950           END-IF.
1960      *
1970       DISPLAY 'CLIENT: INQUARY SERVICE REQ WAS SUCCESS '.
1980      *
1990      ***** set up ACCTDATA-REC *****
2000      *
2010       MOVE 0 TO TP-STATUS OF STATUS-REC.
2020       MOVE LOW-VALUES  TO ACCTDATADEF-REC.
2030       MOVE COMM-HANDLE OF ACCTREQDEF-REC TO
2040            COMM-HANDLE OF ACCTDATADEF-REC.
2050       MOVE "X_COMMON"            TO REC-TYPE    OF ACCTDATATYPE-REC.
2060       MOVE "acctdata"            TO SUB-TYPE    OF ACCTDATATYPE-REC.
2070       COMPUTE LEN OF ACCTDATATYPE-REC = FUNCTION LENGTH(ACCTDATA-REQ).
2080      *
2090       PERFORM WITH TEST AFTER UNTIL NOT TPOK OF STATUS-REC
2100      *
2110      ************************************************************
2120      * TPRECV(データの受信)
2130      ************************************************************
2140         MOVE 0 TO TP-STATUS OF STATUS-REC
2150         CALL "TPRECV" USING
2160             ACCTDATADEF-REC ACCTDATATYPE-REC WK-AREA STATUS-REC
2170             IF TPOK OF STATUS-REC THEN
2180               DISPLAY 'CLIENT: RECEIVED ACOUNT INFORMATION '
2190               DISPLAY 'CLIENT: ACCOUNT NUMBER ='
2200                       ACCTNO IN ACCTDATA-REP
2210               DISPLAY 'CLIENT: NAME =' ANAME IN ACCTDATA-REP
2220               DISPLAY 'CLIENT: AMOUNT =' AMOUNT IN ACCTDATA-REP
2230             END-IF
2240       END-PERFORM.
```

```
2250      *
2260           IF TPEEVENT OF STATUS-REC THEN
2270             IF TPEV-SVCSUCC OF STATUS-REC THEN
2280               DISPLAY 'CLIENT:INQUARY SERVICE SUCCESS'
2290      *
2300      ************************************************************
2310      *   TX-COMMIT(トランザクションのコミット)
2320      ************************************************************
2330      *
2340               DISPLAY  'CLIENT:TX-COMMIT STARTED'
2350               CALL "TXCOMMIT" USING TX-RETURN-STATUS
2360               DISPLAY  'CLIENT:TX-COMMIT ENDED'
2370                 IF RSVAL OF RS NOT = 0 THEN
2380                   DISPLAY 'CLIENT:TX-COMMIT FAILED. CODE ='
2390                   RSVAL OF RS
2400                 END-IF
2410             ELSE
2420               DISPLAY 'CLIENT:EVENT OCCURED IN INQUARY SERVICE'
2430               DISPLAY 'CODE =' TPEVENT OF STATUS-REC
2440      *
2450      ************************************************************
2460      *   TX-ROLLBAK(トランザクションのロールバック)
2470      ************************************************************
2480      *
2490               DISPLAY  'CLIENT:TX-ROLLBACK STARTED'
2500               CALL "TXROLLBACK" USING TX-RETURN-STATUS
2510               DISPLAY  'CLIENT:TX-ROLLBACK ENDED'
2520               IF RSVAL OF RS NOT = 0 THEN
2530                 DISPLAY 'CLIENT:TX-ROLLBACK FAILED. CODE ='
2540                 RSVAL OF RS
2550               END-IF
2560             END-IF
2570           ELSE
2580             DISPLAY 'CLIENT:EVENT OCCURED IN INQUARY SERVICE'
2590             DISPLAY 'CODE =' TPEVENT OF STATUS-REC
2600      *
2610      ************************************************************
2620      *   TX-ROLLBAK(トランザクションのロールバック)
2630      ************************************************************
2640      *
2650             DISPLAY  'CLIENT:TX-ROLLBACK STARTED'
2660             CALL "TXROLLBACK" USING TX-RETURN-STATUS
2670             DISPLAY  'CLIENT:TX-ROLLBACK ENDED'
2680             IF RSVAL OF RS NOT = 0 THEN
2690               DISPLAY 'CLIENT:TX-ROLLBACK FAILED. CODE ='
2700               RSVAL OF RS
2710             END-IF
2720           END-IF.
2730      *
2740      ************************************************************
2750      * 処理の終了
2760      ************************************************************
2770      *
2780       PROG-END.
2790      *
2800       DISPLAY 'CLIENT: SEE YOU LATER'
2810      *
2820      ************************************************************
```

```
2830      * RPC-CLOSE(UAPの終了)
2840      ************************************************************
2850      *
2860       CALL 'CBLDCRPC' USING RPC-CL-ARG.
2870      *
2880       STOP RUN.
```

- ユーザサービス定義の例

  会話型サービスの例題で示す SPP のユーザサービス定義例を次に示します。

```
 10  #ユーザサービス定義の例(convsupファイル)
 20  set module              = "convsup"          #実行形式ファイル名
 30  set watch_time          = 180                #最大応答待ち時間
 40  set receive_from        = none               #受信方法
 50  set trn_expiration_time = 180
 60                           #トランザクションブランチ限界経過時間
 70  set trn_expiration_time_suspend = Y          #必ず Y を指定
```

(e) SPP の例

- XATMI インタフェースの例

  会話型サービスの例題で示す SPP の XATMI インタフェース定義を次に示します。

```
 10  /* SPPのXATMIインタフェース定義の例（cvsppcb.defファイル）*/
 20  X_COMMON acctreq {
 30      long   upperno;
 40      long   lowerno;
 50  };
 60  X_COMMON acctdata {
 70      long   acctno;
 80      char   aname[128];
 90      short  amount;
100  };
110  service INQUIRY(X_COMMON acctreq) ;
```

- SPP のコーディング例（メインプログラム）

  会話型サービスの例題で示す SPP のコーディング例（メインプログラム）を次に示します。

```
 10      *
 20      ************************************************************
 30      * SPPの例（convspp.cblファイル）
 40      ************************************************************
 50      *
 60       IDENTIFICATION DIVISION.
 70      *
 80       PROGRAM-ID.     MAIN.
 90      *
100      ************************************************************
110      * データ領域の設定
120      ************************************************************
130      *
140       DATA DIVISION.
```

```
150       WORKING-STORAGE SECTION.
160       01  RPC-OP-ARG.
170           02  REQEST       PIC X(8) VALUE 'OPEN    '.
180           02  STATUS-CODE  PIC X(5) VALUE SPACE.
190           02  FILLER       PIC X(3).
200           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
210       *
220       01  RPC-CL-ARG.
230           02  REQEST       PIC X(8) VALUE 'CLOSE   '.
240           02  STATUS-CODE  PIC X(5) VALUE SPACE.
250           02  FILLER       PIC X(3).
260           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
270       *
280       01  RSV-ARG.
290           02  REQUEST      PIC X(8) VALUE 'MAINLOOP'.
300           02  STATUS-CODE  PIC X(5) VALUE SPACE.
310           02  FILLER       PIC X(3).
320           02  FLAGS        PIC S9(9) COMP VALUE ZERO.
330       *
340       PROCEDURE DIVISION.
350       *
360       ************************************************************
370       * RPC-OPEN(UAPの開始)
380       ************************************************************
390       *
400       CALL 'CBLDCRPC' USING RPC-OP-ARG.
410           IF STATUS-CODE OF RPC-OP-ARG NOT = '00000' THEN
420             DISPLAY 'SERVER: RPC-OPEN FAILED. CODE = '
430                     STATUS-CODE OF RPC-OP-ARG
440             GO TO PROG-END
450           END-IF.
460       *
470       ************************************************************
480       * RPC-MAINLOOP(SPPサービスの開始)
490       ************************************************************
500       *
510       DISPLAY 'SERVER: ENTERING MAINLOOP...'
520       CALL 'CBLDCRSV' USING RSV-ARG.
530           IF STATUS-CODE OF RSV-ARG NOT = '00000' THEN
540             DISPLAY 'SERVER: RPC-MAINLOOP FAILED. CODE = '
550                     STATUS-CODE OF RSV-ARG
560           END-IF.
570       *
580       ************************************************************
590       * プログラムの終了
600       ************************************************************
610       PROG-END.
620       *
630       ************************************************************
640       * RPC-CLOSE(UAPの終了)
650       ************************************************************
660       *
670       CALL 'CBLDCRPC' USING RPC-CL-ARG.
680       *
690       ************************************************************
700       * 処理の終了
710       ************************************************************
```

```
720      *
730       STOP RUN.
```

- SPP のコーディング例（サービスプログラム）

  会話型サービスの例題で示す SPP のコーディング例（サービスプログラム）を次に示します。

```
 10      *
 20      ************************************************************
 30      * INQUIRY    サービスプログラム(convsvc.cblファイル)
 40      ************************************************************
 50      *
 60       IDENTIFICATION DIVISION.
 70      *
 80       PROGRAM-ID.     INQUIRY.
 90      *
100      ************************************************************
110      * データ領域の設定
120      ************************************************************
130      *
140       DATA DIVISION.
150       WORKING-STORAGE SECTION.
160      *
170      ************************************************************
180      * 変数の宣言
190      ************************************************************
200      *
210      * TPSVCDEF record
220      *
230       01  SVCDEF-REC.
240           COPY TPSVCDEF.
250      *
260      * TPTYPE record
270      *
280       01  TYPE-REC.
290           COPY TPTYPE.
300      *
310      * TPSTATUS record
320      *
330       01  STATUS-REC.
340           COPY TPSTATUS.
350      *
360      * TPSVCRET record
370      *
380       01  SVCRET-REC.
390           COPY TPSVCRET.
400      *
410      * WK-AREA is where service requests are read into
420      *
430       01  WK-AREA PIC X(136).
440      *
450       01  ACCTREQ-REC REDEFINES WK-AREA.
460           COPY ACCTREQ.
470      *
480       01  ACCTDATA-REC.
490           COPY ACCTDATA.
500      *
```

```
510      * TPSVCDEF record for TPSEND
520      *
530       01  ACCTDATADEF-REC.
540           COPY TPSVCDEF.
550      *
560      * TPTYPE record for TPSEND
570      *
580       01  ACCTDATATYPE-REC.
590           COPY TPTYPE.
600      *
610      *
620      ************************************************************
630      * 照会処理
640      ************************************************************
650      *
660       PROCEDURE DIVISION.
670      *
680      ***** set length *****
690       COMPUTE LEN OF TYPE-REC = FUNCTION LENGTH(WK-AREA).
700      *
710      ************************************************************
720      * TPSVCSTART
730      ************************************************************
740      *
750       CALL "TPSVCSTART" USING
760            SVCDEF-REC TYPE-REC WK-AREA STATUS-REC.
770      *
780      * find user data files between lower and upper account number.
790      * In this case 2 data was found, and was replied.
800      *
810      ************************************************************
820      * 送信データの設定
830      ************************************************************
840       MOVE LOW-VALUES        TO ACCTDATADEF-REC.
850       MOVE COMM-HANDLE OF SVCDEF-REC TO COMM-HANDLE OF ACCTDATADEF-REC.
860       MOVE "X_COMMON"        TO REC-TYPE OF ACCTDATATYPE-REC.
870       MOVE "acctdata"        TO SUB-TYPE OF ACCTDATATYPE-REC.
880       COMPUTE LEN OF ACCTDATATYPE-REC =
890               FUNCTION LENGTH(ACCTDATA-REC).
900      *
910      ************************************************************
920      * TPSEND(第1データの送信)
930      ************************************************************
940      *
950       MOVE "10000001"       TO ACCTNO  OF ACCTDATA-REC.
960       MOVE "HITACHI HANAKO" TO ANAME   OF ACCTDATA-REC.
970       MOVE "2000"           TO AMOUNT  OF ACCTDATA-REC.
980       CALL "TPSEND" USING ACCTDATADEF-REC ACCTDATATYPE-REC
990                           ACCTDATA-REC STATUS-REC.
1000          IF TPOK OF STATUS-REC THEN
1010            MOVE 0 TO TP-RETURN-VAL OF SVCRET-REC
1020          ELSE
1030            MOVE 1 TO TP-RETURN-VAL OF SVCRET-REC
1040            GO TO PROG-END
1050          END-IF
1060     *
1070     ************************************************************
1080     * TPSEND(第2データの送信)
```

```
1090      ************************************************************
1100      *
1110       MOVE "10000002"       TO ACCTNO  OF ACCTDATA-REC.
1120       MOVE "HITACHI TAROU"  TO ANAME   OF ACCTDATA-REC.
1130       MOVE "1000"           TO AMOUNT  OF ACCTDATA-REC.
1140       CALL "TPSEND" USING ACCTDATADEF-REC ACCTDATATYPE-REC
1150                           ACCTDATA-REC STATUS-REC.
1160           IF TPOK OF STATUS-REC THEN
1170             MOVE 0 TO TP-RETURN-VAL OF SVCRET-REC
1180           ELSE
1190             MOVE 1 TO TP-RETURN-VAL OF SVCRET-REC
1200             GO TO PROG-END
1210           END-IF
1220      *
1230      ************************************************************
1240      * TPRETURN(受信プログラムの終了)
1250      ************************************************************
1260      *
1270       SET TPSUCCESS OF SVCRET-REC TO TRUE.
1280       MOVE 1 TO APPL-CODE OF SVCRET-REC.
1290      *
1300       PROG-END.
1310      *
1320       MOVE "        "         TO REC-TYPE OF TYPE-REC.
1330       MOVE "                " TO SUB-TYPE OF TYPE-REC.
1340       MOVE 0 TO  LEN OF TYPE-REC.
1350       COPY TPRETURN
1360            REPLACING TPSVCRET-REC BY SVCRET-REC
1370                      TPTYPE-REC BY TYPE-REC
1380                      DATA-REC BY WK-AREA.
1390      *
1400      ************************************************************
1410      * 処理の終了
1420      ************************************************************
1430      *
1440      *
1450       END PROGRAM INQUIRY.
```

- ユーザサービス定義の例

  会話型サービスの例題で示す SPP のユーザサービス定義例を次に示します。

```
  10  # ユーザサービス定義の例(convsppファイル)
  20    set  service_group     = "convspp_svg"    #サービスグループ名
  30    set  module            = "convspp"        #実行形式ファイル名
  40    set  service = "INQUIRY=INQUIRY"
  50                          #サービス名=エントリポイント名
  60    set  watch_time        = 180              #最大応答待ち時間
  70    set  trn_expiration_time   =  240
  80                          #トランザクションブランチ限界経過時間
  90    set  trn_expiration_time_suspend = Y      #必ず Y を指定
 100    set  server_type = "xatmi_cbl"              #サーバタイプ
 110    set  receive_from = "socket"               #受信方法
```

6.4.2　TX インタフェースの例

X/Open の TX インタフェースを使用した，SUP のコーディング例を次に示します。この SUP は，6.1 で示す SUP の処理を，TX インタフェースでトランザクション制御したものです。処理の構成図と，サービス要求先の SPP の処理については，「6.1　クライアント／サーバ形態の UAP のコーディング例（SUP, SPP　DAM アクセス）」を参照してください。なお，行番号 460，470 で示す TX-RETURN-STATUS は，処理から直接呼び出すと正しく参照できないため，行番号 490，500 で示すように，RS REDEFINES TX-RETURN-STATUS と再定義しています。

```
 10  *
 20  ********************************************************
 30  * SUP01                                                *
 40  ********************************************************
 50  *
 60   IDENTIFICATION DIVISION.
 70  *
 80   PROGRAM-ID. MAIN.
 90  *
100  ********************************************************
110  * データ領域の設定                                     *
120  ********************************************************
130  *
140   DATA DIVISION.
150   WORKING-STORAGE SECTION.
160   01  RPC-ARG1.
170       02 REQUEST      PIC X(8) VALUE SPACE.
180       02 STATUS-CODE  PIC X(5) VALUE SPACE.
190       02 FILLER       PIC X(3).
200       02 FLAGS        PIC S9(9) COMP VALUE ZERO.
210  *
220   01  RPC-ARG2.
230       02 REQUEST      PIC X(8) VALUE SPACE.
240       02 STATUS-CODE  PIC X(5) VALUE SPACE.
250       02 FILLER       PIC X(3).
260       02 FLAGS        PIC S9(9) COMP VALUE ZERO.
270       02 DESCRIPTOR   PIC S9(9) COMP VALUE ZERO.
280       02 S-NAME       PIC X(32) VALUE SPACE.
290       02 G-NAME       PIC X(32) VALUE SPACE.
300  *
310   01  RPC-ARG3.
320       02  SEND-DATA-LENG  PIC S9(9) COMP VALUE ZERO.
330       02  SEND-DATA       PIC X(32) VALUE SPACE.
340  *
350   01  RPC-ARG4.
360       02  RECEIVE-DATA-LENG PIC S9(9) COMP VALUE ZERO.
370       02  RECEIVE-DATA      PIC X(32) VALUE SPACE.
380  *
390   01  ADM-ARG1.
400       02  REQUEST      PIC X(8) VALUE SPACE.
410       02  STATUS-CODE  PIC X(5) VALUE SPACE.
420       02  FILLER       PIC X(3).
430       02  FLAGS        PIC S9(9) COMP VALUE ZERO.
440       02  FILLER       PIC X(3).
450  *
460   01  TX-RETURN-STATUS.
```

```
 470       COPY TXSTATUS.
 480  *
 490   01  RS REDEFINES TX-RETURN-STATUS.
 500       05 RSVAL         PIC S9(9) COMP-5.
 510  *
 520   01  TX-INFO-AREA.
 530       COPY TXINFDEF.
 540  *
 550   PROCEDURE DIVISION.
 560  *
 570  ******************************************************
 580  * RPC-OPEN(UAPの開始)                                *
 590  ******************************************************
 600  *
 610   MOVE 'OPEN' TO REQUEST OF RPC-ARG1.
 620   MOVE ZERO   TO FLAGS  OF RPC-ARG1.
 630   CALL 'CBLDCRPC' USING RPC-ARG1.
 640       IF STATUS-CODE OF RPC-ARG1 NOT = '00000' THEN
 650         DISPLAY 'SUP01:RPC-OPEN FAILED. CODE = '
 660         STATUS-CODE OF RPC-ARG1
 670         GO TO PROG-END
 680       END-IF.
 690  *
 700  ******************************************************
 710  * TX-OPEN(リソースマネジャのオープン)                *
 720  ******************************************************
 730  *
 740   CALL 'TXOPEN' USING TX-RETURN-STATUS.
 750       IF RSVAL OF RS NOT = 0 THEN
 760         DISPLAY 'SUP01:TX-OPEN FAILED. CODE = '
 770         RSVAL OF RS
 780         GO TO PROG-END
 790       END-IF.
 800  *
 810  ************************************************************
 820  * TX-SET-TRANSACTION-TIMEOUT(トランザクション監視時間の設定)*
 830  ************************************************************
 840  *
 850   MOVE 180 TO TRANSACTION-TIMEOUT OF TX-INFO-AREA.
 860   CALL 'TXSETTIMEOUT' USING TX-INFO-AREA TX-RETURN-STATUS.
 870       IF RSVAL OF RS NOT = 0 THEN
 880         DISPLAY 'SUP01:TX-SET-TRANSACTION-TIMEOUT FAILED. CODE = '
 890         RSVAL OF RS
 900         GO TO PROG-END
 910       END-IF.
 920  *
 930  ******************************************************
 940  * ADM-COMPLETE(ユーザサーバの開始処理完了報告)       *
 950  ******************************************************
 960  *
 970   MOVE 'COMPLETE' TO REQUEST OF ADM-ARG1.
 980   CALL 'CBLDCADM' USING ADM-ARG1.
 990       IF STATUS-CODE OF ADM-ARG1 NOT = '00000' THEN
1000         DISPLAY 'SUP01:ADM-COMPLETE FAILED. CODE = '
1010         STATUS-CODE OF ADM-ARG1
1020         GO TO PROG-END
1030       END-IF.
1040  *
```

```
1050  ******************************************************
1060  * TX-BEGIN(トランザクションの開始)                    *
1070  ******************************************************
1080  *
1090   CALL 'TXBEGIN' USING TX-RETURN-STATUS.
1100       IF RSVAL OF RS NOT = 0 THEN
1110         DISPLAY 'SUP01:TX-BEGIN FAILED. CODE = '
1120         RSVAL OF RS
1130         GO TO TRAN-END
1140       END-IF.
1150  *
1160  ******************************************************
1170  * TX-INFO(トランザクション情報の取得)                 *
1180  ******************************************************
1190  *
1200   CALL 'TXINFORM' USING TX-INFO-AREA TX-RETURN-STATUS.
1210       IF RSVAL OF RS <= 0 THEN
1220         DISPLAY 'SUP01:NOT IN TRANSACTION. CODE = '
1230         RSVAL OF RS
1240         GO TO PROG-END
1250       ELSE
1260         IF RSVAL OF RS = 1 THEN
1270           DISPLAY 'SUP01:RETURN = ' COMMIT-RETURN
1280           DISPLAY 'SUP01:CONTROL = ' TRANSACTION-CONTROL
1290           DISPLAY 'SUP01:TIMEOUT = ' TRANSACTION-TIMEOUT
1300           DISPLAY 'SUP01:STATE = ' TRANSACTION-STATE
1310         END-IF
1320       END-IF.
1330  ******************************************************
1340  * RPC-CALL(遠隔サービスの要求)                        *
1350  ******************************************************
1360  *
1370   MOVE 'CALL' TO REQUEST OF RPC-ARG2.
1380   MOVE 'SPP01' TO G-NAME OF RPC-ARG2.
1390   MOVE 'SVR01' TO S-NAME OF RPC-ARG2.
1400   MOVE 'SUP01:DATA OpenTP1' TO SEND-DATA OF RPC-ARG3.
1410   MOVE 32 TO SEND-DATA-LENG OF RPC-ARG3.
1420   MOVE 32 TO RECEIVE-DATA-LENG OF RPC-ARG4.
1430   CALL 'CBLDCRPC' USING RPC-ARG2 RPC-ARG3 RPC-ARG4.
1440       IF STATUS-CODE OF RPC-ARG2 NOT = '00000' THEN
1450         DISPLAY 'SUP01:RPC-CALL RETURN CODE = '
1460         STATUS-CODE OF RPC-ARG2
1470  *       GO TO TRAN-END
1480       END-IF.
1490   DISPLAY 'SERVICE FUNCTION RETURN = ' RECEIVE-DATA.
1500   TRAN-END.
1510  *
1520  ******************************************************
1530  * TX-SET-TRANSACTION-CONTROL(非連鎖モード設定)        *
1540  ******************************************************
1550  *
1560   MOVE 0 TO TRANSACTION-CONTROL OF TX-INFO-AREA.
1570   CALL 'TXSETTRANCTL' USING TX-INFO-AREA TX-RETURN-STATUS.
1580       IF RSVAL OF RS NOT = 0 THEN
1590         DISPLAY 'SUP01:TX-SET-TRANSACTION-CONTROL FAILED. CODE = '
1600         RSVAL OF RS
1610       END-IF.
1620  *
```

```
1630  ******************************************************
1640  * TX-COMMIT(非連鎖モードのコミット)                   *
1650  ******************************************************
1660  *
1670   CALL 'TXCOMMIT' USING TX-RETURN-STATUS.
1680       IF RSVAL OF RS NOT = 0 THEN
1690         DISPLAY 'SUP01:TX-COMMIT FAILED. CODE = '
1700         RSVAL OF RS
1710       END-IF.
1720   PROG-END.
1730  *
1740  ******************************************************
1750  * RPC-CLOSE(UAPの終了)                               *
1760  ******************************************************
1770  *
1780   MOVE 'CLOSE' TO REQUEST OF RPC-ARG1.
1790   MOVE ZERO    TO FLAGS  OF RPC-ARG1.
1800   CALL 'CBLDCRPC' USING RPC-ARG1.
1810   DISPLAY 'SUP01:SUP PROCESS ENDED'.
1820   STOP RUN.
```

7 アプリケーション起動関連のリファレンス

この章では，TP1/Message Control を使った場合の，アプリケーションプログラムを起動する機能に関連するユーザオウンコーディング（C 言語の形式）と MCF イベントのリファレンス情報について説明します。

タイマ起動引き継ぎ決定 UOC の関数形式

タイマ起動引き継ぎ決定 UOC は，次に示す形式で呼び出します。

形式

ANSI C，C++ の形式

```
#include <dcmpsv.h>
DCLONG  uoc_func(dcmpsv_uoc_rtime  *parm)
```

K&R 版 C の形式

```
#include <dcmpsv.h>
DCLONG  uoc_func(parm)

dcmpsv_uoc_rtime     *parm ;
```

説明

タイマ起動のアプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）を呼び出したあとで，障害が起こって OpenTP1 を再開始（リラン）した場合に，タイマ起動の環境を変更する UOC です。この UOC を作成しておくと，次に示すことができます。

- タイマ起動を引き継ぐか，または取り消す。
- 引き継いだタイマ起動を即時起動とする。
- 起動するアプリケーション名を変更する。

タイマ起動引き継ぎ決定 UOC を MCF に組み込むときは，アプリケーション起動サービス用の MCF メイン関数に，UOC 関数のアドレスを設定します。アプリケーション起動サービス用の MCF メイン関数は，通信プロトコルに依存しません。

アプリケーション起動サービス用の MCF メイン関数の作成方法については，マニュアル「OpenTP1 運用と操作」を参照してください。

uoc_func（タイマ起動引き継ぎ決定 UOC）を呼び出すときには，parm には次に示すパラメタが MCF から設定されます。

パラメタの内容

dcmpsv_uoc_rtime の内容

```
typedef  struct {
     char    le_name[9] ;   … 入力元論理端末名称
     char    reserve1[7] ;  … 予備
     char    ap_name[9] ;   … アプリケーション名
     char    reserve2[7] ;  … 予備
     DCLONG  exec_time ;    … タイマ起動時刻
```

```
    char    ap_type ;       … アプリケーションの型
                              'a':ans型  'n':noans型
    char    time_type ;     … タイマ起動の種別
                              'i':経過時間指定のタイマ起動
                              't':時刻指定のタイマ起動
    char    reserve3[26] ; … 予備
} dcmpsv_uoc_rtime ;
```

MCF から UOC へ値が渡される引数

●le_name

入力元の論理端末名称が設定されます。アプリケーションプログラムの起動

(CBLDCMCF('EXECAP ')) が SPP から呼び出されている場合は，'*' が設定されます。

●ap_name

UAP がタイマ起動のアプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）に設定した，アプリケーション名が設定されます。

●exec_time

UAP がタイマ起動のアプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）に設定した，MHP を起動する時間が，1970 年 1 月 1 日 0 時 0 分 0 秒からの通算秒で設定されます。

●ap_type

タイマ起動のアプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）を呼び出した UAP の，アプリケーションの型が設定されます。

'a'：ans 型

'n'：noans 型

●time_type

UAP がタイマ起動のアプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）に設定した，タイマ起動の種別が設定されます。

'i'：経過時間指定のタイマ起動

't'：時刻指定のタイマ起動

UOC で値を設定する引数

●ap_name

タイマ起動で開始させるアプリケーション名を変更するときに，変更後のアプリケーション名を設定します。ここに指定した名称は，リターン値に DCMPSV_UOC_TIME_JUST を設定したときに有効となります。

リターン値

uoc_func()は，次に示す値でリターンしてください。

リターン値	意味
DCMPSV_UOC_TIME_CONTINUE	タイマ起動を継続します。
DCMPSV_UOC_TIME_JUST	即時起動として起動します。
DCMPSV_UOC_TIME_DEQ	タイマ起動を取り消します。

uoc_func()からリターンした値による MCF の処理を次に示します。

- DCMPSV_UOC_TIME_CONTINUE

  UOC からこの値をリターンすると，MCF は時刻を 1970 年 0 時 0 分 0 秒からの通算秒で取得して，アプリケーションプログラムの起動（CBLDCMCF('EXECAP ')）に設定された時刻と比べます。比べた結果，関数に設定した時刻を過ぎていた場合，該当する MHP を即時に起動します。設定した時刻を過ぎていない場合は，タイマ起動となります。

- DCMPSV_UOC_TIME_JUST

  UOC からこの値をリターンすると，MCF は該当する MHP を即時に起動します。この値をリターンする場合は，即時起動する MHP のアプリケーション名を，UOC で変更できます。ただし，MCF イベント処理用 MHP を起動するような変更はできません。変更したアプリケーション名が定義されていない名称の場合は，ERREVT4 が通知されます。

  UOC で即時起動する UAP のアプリケーション名を変更した場合に，変更前と変更後で起動する MHP のアプリケーションの型が異なるときは，タイマ起動するセグメントは出力キューから削除されて，警告メッセージ（KFCA10711-W）が出力されます。

- DCMPSV_UOC_TIME_DEQ

  UOC からこの値をリターンすると，MCF はタイマ起動を取り消します。タイマ起動するセグメントは出力キューから削除されて，情報メッセージ（KFCA10700-I）が出力されます。

上記以外の値を UOC からリターンすると，タイマ起動するセグメントは出力キューから削除されて，警告メッセージ（KFCA10710-W）が出力されます。

UOC 作成上の注意事項

- UOC で使用できる関数

  UOC を作成する場合，UOC では次に示す関数だけが使用できます。ほか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注意してください。

  - メモリ操作をする関数

    データ領域管理（例：malloc，free）

    共有メモリ管理（例：shmctl，shmget，shmop）

    メモリ操作（例：memcpy）

    文字列操作（例：strcpy）

- 時間取得関数

- **UOC の異常処理**

  UOC で異常を検知した場合，MCF の所定のリターンコードを使用して MCF に異常の発生を通知してください。UOC でプロセス終了となるシグナル，または abort()を発行すると，MCF が異常終了します。

- **UOC の実行タイミング**

  MCF が起動する UOC の実行タイミングは，OpenTP1 システム，および UAP の開始，終了シーケンスと同期しない場合があります。UAP より先に UOC が実行されたり，UAP がすべて終了してから UOC が呼び出されたりしてもよいように作ってください。

- **UOC のローカル変数サイズ**

  UOC で使用するローカル変数のサイズの合計は，各 UOC で 1 キロバイト以内になるよう設計してください。また，UOC の中で関数の再帰呼び出しはしないでください。

タイマ起動メッセージ廃棄通知イベント（ERREVT4）のデータ形式

タイマ起動メッセージ廃棄通知イベント（ERREVT4）の先頭セグメントとして渡される，データの形式を次に示します。ERREVT4 以外の MCF イベント情報の形式については，マニュアル「OpenTP1 プロトコル」の該当するプロトコル編を参照してください。

ERREVT4 の MCF イベント情報の内容

ERREVT4 の MCF イベント情報の内容を表 7-1 に，ERREVT4 の理由コードを表 7-2 に示します。形式 1 とはバッファ形式 1 を，形式 2 とはバッファ形式 2 を示します。

表 7-1　ERREVT4 の MCF イベント情報の内容

項目	位置(バイト)		長さ(バイト)	属性	内容
	形式 1	形式 2			
予備（形式 1 のときだけ）	0	−	2	−	−
予備（形式 1 のときだけ）	2	−	2	−	−
エラーイベントコード	4	0	3	英数字	'ERR'が設定されます。
	7	3	3	−	−
	10	6	2	英数字	ERREVT4 を示す'4△'が設定されます。
入力元論理端末名称	12	8	8	英数字	メッセージを入力した論理端末名称 次に示す場合は，'*'が設定されます。 1. SPP がアプリケーション起動をした MHP で，障害が起こった場合 2. 上記の障害時に MCF イベントとして起動された MHP が，さらにアプリケーション起動をした MHP で，障害が起こった場合
予備	20	16	20	−	−
アプリケーション名	40	36	8	英数字	異常が発生したタイマ起動のアプリケーション名
予備	48	44	8	−	−
予備	56	52	8	−	−
予備	64	60	8	−	−
コネクション名	72	68	8	英数字	コネクション名です。 次に示す場合は，'*'が設定されます。 1. SPP がアプリケーション起動をした MHP で，障害が起こった場合 2. 上記の障害時に MCF イベントとして起動された MHP が，さらにアプリケーション起動をした MHP で，障害が起こった場合

項目	位置(バイト)		長さ(バイト)	属性	内容
	形式1	形式2			
予備	80	76	16	−	−
メッセージが入力された日付	96	92	8	外部 10 進	端末入力メッセージを入力した日付です。 「yyyymmdd」の形式です。 yyyy：西暦の年 mm：月 dd：日
メッセージが入力された時刻	104	100	8	外部 10 進	端末入力メッセージを入力した時刻です。 「hhmmss00」の形式です。 hh：時 mm：分 ss：秒 「00」は固定です。
理由コード	112	108	4	外部 10 進	理由コードが設定されます。
予備	116	112	12	−	−

（凡例）

−：該当しません。

表 7-2　ERREVT4 の理由コード

COBOL 言語の理由コード（外部 10 進）	ERREVT4 の通知理由
0020	RPC 障害，サーバ未起動などが原因で，MHP の起動に失敗しました。
0030	メモリ不足のため，入力キューへの書き込みに失敗しました。
0031	キューファイルが満杯のため，入力キューへの書き込みに失敗しました。
0032	入力メッセージ最大格納数の定義指定値を超えたため，入力キューに書き込みませんでした。
0033	入力キューへの書き込み時に障害が発生しました。
0040	MHP のアプリケーションが閉塞中です。
0041	MHP のアプリケーションがセキュア状態です。
0042	MHP のサービスまたはサービスグループが閉塞中です。
0043	MHP のサービスグループがセキュア状態です。

付録

付録 A　OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合

OpenTP1 の通信のうち，OpenTP1 独自のインタフェース（CBLDCRPC('CALL ')）と XATMI インタフェースを併用する方法について説明します。

XATMI インタフェースと併用できるのは，OpenTP1 独自のインタフェースだけです。TxRPC と XATMI インタフェースは併用できません。

付録 A.1　併用する形態

併用する形態には，次に示す 2 とおりがあります。

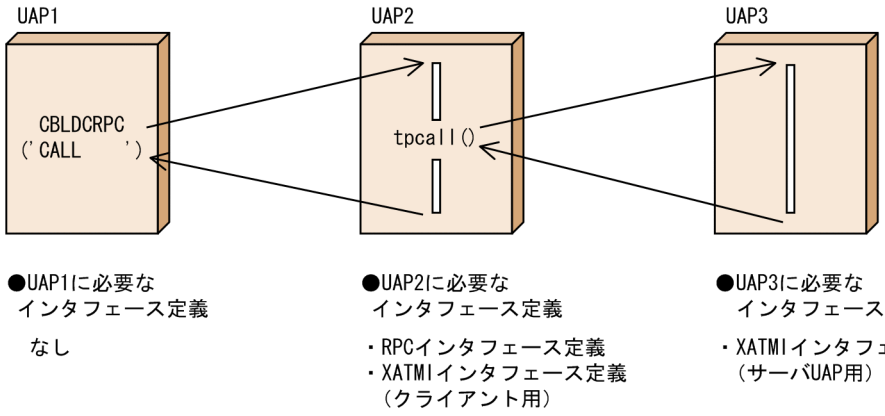
1. OpenTP1 の RPC のサーバであり，XATMI インタフェース通信のクライアントである場合
2. XATMI インタフェース通信のサーバであり，OpenTP1 の RPC のクライアントである場合

このうち，1.の形態のスタブを作成するときには，RPC インタフェース定義と XATMI インタフェース定義を一つのファイルに指定して，stbmake コマンドまたは tpstbmk コマンドを実行します。

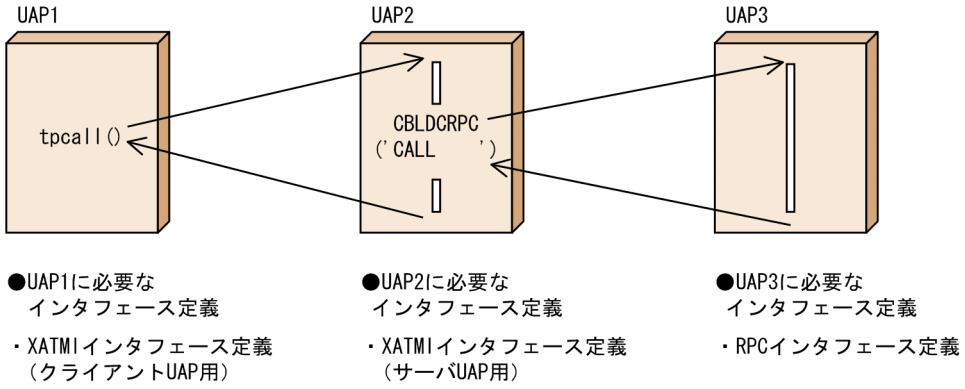
通信を併用する形態と必要なインタフェース定義を次の図に示します。

図 A-1　通信を併用する形態と必要なインタフェース定義

1. OpenTP1のRPCのサーバであり，XATMIインタフェース通信のクライアントである場合



2. XATMIインタフェース通信のサーバであり，OpenTP1のRPCのクライアントである場合



付録 A.2　併用するアプリケーションプログラムのスタブの作成手順

CBLDCRPC('CALL ')で呼ばれて，XATMI インタフェースの API（TPCALL など）を呼び出す UAP のスタブを作成する方法について説明します。

1. インタフェース定義ファイルを作成します

作成するファイルには，RPC インタフェース定義と XATMI インタフェース定義のクライアント用の定義を指定します。ファイル名は ".def" で終わらせてください。

2. stbmake コマンドまたは tpstbmk コマンドを実行します

スタブを生成するコマンド（stbmake コマンドまたは tpstbmk コマンド）に，必要なコマンド引数を指定して実行します。コマンドを実行すると，次に示すファイルが作成されます（xxxxx は，インタフェース定義ファイルのファイル名から ".def" を除いた文字列を示します）。

- OpenTP1 の RPC 用スタブソースファイル
  （デフォルトのファイル名：xxxxx_sstb．c）

- XATMI スタブソースファイル（デフォルトのファイル名：xxxxx_stbx.c）
- XATMI スタブヘッダファイル（デフォルトのファイル名：xxxxx_stbx.h）
- XATMI スタブコピーファイル（サブタイプ名に ".cbl" が付いた名称）

RPC インタフェース定義と XATMI インタフェース定義が混在している場合には，XATMI スタブソースファイル，XATMI スタブヘッダファイルおよび XATMI スタブコピーファイルが作成されます。

3. スタブソースファイルをコンパイルして，UAP にリンケージさせます

2.で作成したソースファイルを C コンパイラでコンパイルして，UAP にリンケージさせます。

付録 A.3　呼び出せる XATMI インタフェースの API

CBLDCRPC('CALL ')で呼び出された SPP が使える XATMI インタフェースの API を次の表に示します。これらの API を呼び出す SPP にはスタブを結合していることが前提となります。詳細については，「付録 A.2　併用するアプリケーションプログラムのスタブの作成手順」を参照してください。

表 A-1　CBLDCRPC('CALL ')で呼び出された SPP が使える XATMI インタフェースの API

XATMI インタフェースの API	使用可否
TPACALL	○
TPADVERTISE	－
TPCALL	○
TPCANCEL	○
TPCONNECT	○
TPDISCON	○
TPGETRPLY	○
TPRECV	○
TPRETURN	○
TPSEND	○
TPSVCSTART	○
TPUNADVERTISE	－

（凡例）

○：API を使えます。

－：API を使えません。

索引

記号

.def 58

A

ACTIVE 句 434
ap_name 581
ap_type 581

B

BEFORE 句 442

C

called_servers 55
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 72
CBLDCADM('COMMAND ') 84
CBLDCADM('COMPLETE') 87
CBLDCADM('STATUS ') 89
CBLDCADT('PRINT ') 92
CBLDCDAM('CLOS') 98
CBLDCDAM('END ') 101
CBLDCDAM('HOLD') 103
CBLDCDAM('OPEN') 105
CBLDCDAM('READ') 110
CBLDCDAM('REWT') 116
CBLDCDAM('RLES') 120
CBLDCDAM('STAT') 123
CBLDCDAM('STRT') 127
CBLDCDAM('WRIT') 129
CBLDCDMB('BSEK') 134
CBLDCDMB('CLOS') 136
CBLDCDMB('CRAT') 138
CBLDCDMB('DGET') 142
CBLDCDMB('DPUT') 145
CBLDCDMB('GET ') 148
CBLDCDMB('OPEN') 151
CBLDCDMB('PUT ') 154
CBLDCIST('CLOS') 158
CBLDCIST('OPEN') 160
CBLDCIST('READ') 162
CBLDCIST('WRIT') 165
CBLDCJNL('UJPUT ') 169
CBLDCJUP('CLOSERPT') 172
CBLDCJUP('OPENRPT ') 174
CBLDCJUP('RDGETRPT') 177
CBLDCLCK('GET ') 187
CBLDCLCK('RELALL ') 190
CBLDCLCK('RELNAME ') 192
CBLDCLOG('PRINT ') 195
CBLDCMCF('ADLTAP ') 200
CBLDCMCF('APINFO ') 203
CBLDCMCF('CLOSE ') 209
CBLDCMCF('COMMIT ') 211
CBLDCMCF('CONTEND ') 214
CBLDCMCF('EXECAP ') 216
CBLDCMCF('MAINLOOP') 224
CBLDCMCF('OPEN ') 226
CBLDCMCF('RECEIVE ') 228
CBLDCMCF('RECVSYNC') 234
CBLDCMCF('REPLY ') 235
CBLDCMCF('RESEND ') 236
CBLDCMCF('ROLLBACK') 237
CBLDCMCF('SENDRECV') 241
CBLDCMCF('SENDSYNC') 242
CBLDCMCF('SEND ') 240
CBLDCMCF('TACTCN ') 243
CBLDCMCF('TACTLE ') 247
CBLDCMCF('TDCTCN ') 251
CBLDCMCF('TDCTLE ') 255
CBLDCMCF('TDLQLE ') 259
CBLDCMCF('TEMPGET ') 262
CBLDCMCF('TEMPPUT ') 267
CBLDCMCF('TIMERCAN') 271
CBLDCMCF('TIMERSET') 273

CBLDCMCF('TLSCN ') 277
CBLDCMCF('TLSCOM ') 282
CBLDCMCF('TLSLE ') 286
CBLDCMCF('TLSLN ') 289
CBLDCMCF('TOFLN ') 292
CBLDCMCF('TONLN ') 294
CBLDCPRF('PRFGETN ') 297
CBLDCPRF('PRFPUT ') 299
CBLDCRAP('CONNECTX') 305
CBLDCRAP('CONNECT ') 302
CBLDCRAP('DISCNCT ') 308
CBLDCRPC('CALL ') 311
CBLDCRPC('CALL ')で呼び出された SPP が使える XATMI インタフェースの API 589
CBLDCRPC('CLOSE ') 328
CBLDCRPC('CLTSEND ') 330
CBLDCRPC('DISCARDF') 333
CBLDCRPC('DISCARDS') 335
CBLDCRPC('GETCLADR') 337
CBLDCRPC('GETERDES') 339
CBLDCRPC('GETGWADR') 341
CBLDCRPC('GETSVPRI') 343
CBLDCRPC('GETWATCH') 345
CBLDCRPC('OPEN ') 347
CBLDCRPC('POLLANYR') 349
CBLDCRPC('SETSVPRI') 355
CBLDCRPC('SETWATCH') 357
CBLDCRPC('SVRETRY ') 359
CBLDCRSV('MAINLOOP') 361
CBLDCRTS('RTSPUT ') 364
CBLDCTAM('ERS '/'ERSR'/'ZRS '/'ZRSR') 368
CBLDCTAM('FxxR'/'FxxU'/'VxxR'/'VxxU') 372
CBLDCTAM('GST ') 378
CBLDCTAM('INFO') 381
CBLDCTAM('MFY '/'MFYS'/'STR '/'WFY '/'WFYS'/'YTR ') 386
CBLDCTRN('BEGIN ') 391
CBLDCTRN('C-COMMIT') 393
CBLDCTRN('C-ROLL ') 396
CBLDCTRN('INFO ') 399
CBLDCTRN('RMSELECT') 401
CBLDCTRN('U-COMMIT') 404
CBLDCTRN('U-ROLL ') 407
CBLDCUTO('T-STATUS') 410
CBLDCXAT('CONNECT ') 526
COBOL 言語用テンプレート 71
COBOL データ構造 500
COBOL-UAP 作成用プログラムの説明形式 71
COBOL-UAP 作成用プログラムの文法 70
commit_return 特性の設定 517
COMMIT － MHP のコミット 426
CUP への一方通知 330

D

DAM アクセス機能を使う場合 50
DAM ファイルサービス（CBLDCDAM, CBLDCDMB） 97
DCADM.cbl 83
DCCONFPATH 68
DCDAM.cbl 97
DCDIR 68
DCDMB.cbl 97
DCIST.cbl 157
DCJNL.cbl 168
DCJUP.cbl 171
DCLCK.cbl 186
DCLOG.cbl 194
DCMCF.cbl 199
DCMPSV_UOC_TIME_CONTINUE 582
DCMPSV_UOC_TIME_DEQ 582
DCMPSV_UOC_TIME_JUST 582
DCPRF.cbl 296
DCRAP.cbl 301
DCRPC.cbl 310
DCRSV.cbl 310
DCSVGNAME 68
DCSVNAME 68
DCTAM.cbl 367

DCTRN.cbl 390
DCUAPCONFPATH 68
DCUTO.cbl 409
DC で始まる環境変数 68
DISABLE － 継続問い合わせ応答の終了 427
DML 414
DML インタフェース 413
DML のサービス機能 425

E

entry 47
ERREVT4 584
exec_time 581
EXIT PROGRAM 39

F

FIRST 421
FOR 句 420, 430, 434, 439, 442, 444

I

ISAM 機能を使う場合 50
IST サービス（CBLDCIST） 157
IST テーブルからレコードの入力 162
IST テーブルのオープン 160
IST テーブルのクローズ 158
IST テーブルへレコードの出力 165

J

jnlrput 出力ファイルからジャーナルデータの入力177
jnlrput 出力ファイルのオープン 174
jnlrput 出力ファイルのクローズ 172

L

le_name 581

M

MCF 環境のオープン 226
MCF 環境のクローズ 209
MCF 通信サービスの状態取得 282
MESSAGE 430
MESSAGE DATE 句 421
MESSAGE TIME 句 421
MHP 74
MHP で使える機能とプログラム 32
MHP で使える機能と要求コード 32
MHP のコミット 211
MHP のコミット〔COMMIT〕 426
MHP のサービス開始 224
MHP の作成手順 42
MHP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラム 76
MHP のメインプログラムの場合 76
MHP のロールバック〔CBLDCMCF('ROLLBACK')〕 237
MHP のロールバック〔ROLLBACK〕 432

O

OpenTP1 から値が返されるデータ領域 72
OpenTP1 で設定する UAP のシグナルの一覧 67
OpenTP1 の COBOL-UAP 作成用プログラムの文法（DML インタフェース） 413
OpenTP1 の COBOL-UAP 作成用プログラムの文法（アソシエーションの状態の通知） 524
OpenTP1 の UAP を COBOL 言語で作成するとき71
OpenTP1 の機能と COBOL-UAP 作成用プログラムの対応 20
OpenTP1 のリモートプロシジャコールと XATMI インタフェースの API を併用する場合 587
OSI TP 通信をする UAP の場合 54

R

rap リスナーとのコネクションの解放 308
rap リスナーとのコネクションの確立〔CBLDCRAP('CONNECTX')〕 305
rap リスナーとのコネクションの確立〔CBLDCRAP('CONNECT ')〕 302
RECEIVE － 一時記憶データの受け取り 429
RECEIVE － メッセージの受信 420
ROLLBACK － MHP のロールバック 432

RPC インタフェース定義　46
RPC インタフェース定義に注釈文を記述するとき　47
RPC インタフェース定義の作成例　47
RPC インタフェース定義ファイル　46
RPC インタフェース定義ファイルの作成　46
RPC インタフェース定義ファイルを格納するディレクトリ　47
RPC インタフェース定義ファイルを示すサフィックス　47

S

SEGMENT　430
SEND − アプリケーションプログラムの起動　433
SEND − 一時記憶データの更新　439
SEND − 運用コマンドの実行　441
SEND − メッセージの送信　424
SEND − ユーザジャーナルの取得　444
service　57
SPP　74
SPP，MHP に結合させるファイル　50
SPP，MHP の開始と終了　65
SPP で使える機能とプログラム　27
SPP で使える機能と要求コード　27
SPP のサービス開始　361
SPP の作成手順　40
SPP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラム　75
SPP のメインプログラムの場合　75
STATUS KEY 句　421, 427, 430, 434, 440, 442, 444
stbmake（XATMI インタフェース用スタブの作成 TCP/IP 通信）　61
stbmake（スタブのソースファイルの作成）　48
stbmake コマンド〔XATMI インタフェース用スタブの作成方法〕　54
stbmake コマンド〔スタブの作成方法〕　46
stbmake コマンドで作成されるファイル　47
stbmake コマンドで入出力できるファイル名の長さ　63
STOP　RUN　39
SUP　74
SUP で使える機能とプログラム　24
SUP で使える機能と要求コード　25
SUP に結合させるファイル　50
SUP の開始と終了　65
SUP の作成手順　40
SUP のメインプログラムで必ず呼び出す OpenTP1 の COBOL-UAP 作成用プログラム　75
SUP のメインプログラムの場合　75
SUP を強制停止させたい場合　65
SYMBOLIC TERMINAL 句　421, 434

T

TAM アクセス機能を使う場合　50
TAM テーブルからレコードの入力　372
TAM テーブルの状態の取得　378
TAM テーブルの情報の取得　381
TAM テーブルのレコードの更新/追加　386
TAM テーブルのレコードの削除　368
TAM ファイルサービス（CBLDCTAM）　367
TCP/IP 通信をする UAP の場合　54
time_type　581
TPACALL　454
TPADVERTISE　458
TPCALL　460
TPCANCEL　466
TPCONNECT　468
TPDISCON　473
TPGETRPLY　475
TPINTRO　451
TPRECV　480
TPRETURN　485
TPSEND　489
tpstbmk（XATMI インタフェース用スタブの作成 OSI TP 通信）　63
tpstbmk コマンド〔XATMI インタフェース用スタブの作成方法〕　54
tpstbmk コマンドで入出力できるファイル名の長さ64
TPSVCSTART　494
TPUNADVERTISE　497

transaction_control 特性の設定　522
transaction_timeout 特性の設定　520
trnmkobj コマンド　50
TXBEGIN　503
TXCLOSE　505
TXCOMMIT　507
TXINFORM　510
TXINTRO　500
TXOPEN　512
TXROLLBACK　514
TXSETCOMMITRET　517
TXSETTIMEOUT　520
TXSETTRANCTL　522
TX インタフェースのアプリケーションプログラミングインタフェース（TX～）　499
TX インタフェースの例　575

U

UAP 共用ライブラリ　45
UAP 作成時の注意　49
UAP で値を設定するデータ領域　71
UAP の実行環境を設定する方法　77
UAP のプロセスが開始して，OS から最初に呼ばれるプログラム　75
uoc_func()　582
uoc_func（タイマ起動引き継ぎ決定 UOC）　580

W

Windows を使う場合〔コーディング規約〕　39
WITH STOPPING 句　432
WITH 句　435

X

X_C_TYPE　57
X_COMMON　56
X_OCTET　56
X/Open に準拠した API と機能の対応　447
X/Open に準拠した UAP のコーディング例　552
X/Open に準拠したアプリケーションプログラミングインタフェース　446
XATMI インタフェース　587
XATMI インタフェース定義　588
XATMI インタフェース定義（クライアント UAP 用）　54
XATMI インタフェース定義（サーバ UAP 用）　55
XATMI インタフェース定義ファイル　54
XATMI インタフェース定義ファイルの名称　58
XATMI インタフェース定義ファイルを示すサフィックス　58
XATMI インタフェースのアプリケーションプログラミングインタフェース（TP～）　450
XATMI インタフェースの通信を OSI TP 通信で使う場合　63
XATMI インタフェースの例　552
XATMI インタフェース用スタブの作成方法　53
XATMI インタフェース用の COPY ファイル　451
XATMI インタフェースを使う UAP の作成方法　52
XATMI スタブコピーファイル　60
XATMI スタブソースファイル　60
XATMI スタブヘッダファイル　60
XATMI 用のスタブ　60

あ

アソシエーションの確立　526
アソシエーションの状態を示す通信イベントの形式　528
アソシエーションの操作（CBLDCXAT）　525
アプリケーション起動関連のリファレンス　579
アプリケーション情報通知　203
アプリケーションに関するタイマ起動要求の削除　200
アプリケーションプログラムと COBOL-UAP 作成用プログラムの対応　20
アプリケーションプログラムの開始　347
アプリケーションプログラムの開始と終了　65
アプリケーションプログラムの環境変数　68
アプリケーションプログラムの起動〔CBLDCMCF('EXECAP ')〕　216
アプリケーションプログラムの起動〔SEND〕　433
アプリケーションプログラムのコーディング　20
アプリケーションプログラムの作成　19

アプリケーションプログラムの作成（TCP/IP 通信，OSI TP 通信） 52
アプリケーションプログラムの作成（TCP/IP 通信） 40
アプリケーションプログラムの作成手順 52
アプリケーションプログラムの実行 65
アプリケーションプログラムの終了 328
アプリケーションプログラムの翻訳と結合 49

い

一意名 1 421, 435, 442, 444
一意名 2 442
一意名およびデータ名 415
一意名で示すファイル名とデータ名 71
一時記憶データの受け取り〔CBLDCMCF('TEMPGET ')〕 262
一時記憶データの受け取り〔RECEIVE〕 429
一時記憶データの更新〔CBLDCMCF('TEMPPUT ')〕 267
一時記憶データの更新〔SEND〕 439
入り口点 47
入り口名〔コーディング規約〕 39

う

運用コマンドの実行〔CBLDCADM('COMMAND ')〕 84
運用コマンドの実行〔SEND〕 441

え

エラーが発生した非同期応答型 RPC 要求の記述子の取得 339
遠隔サービスの要求 311

お

応答メッセージの送信〔CBLDCMCF('REPLY ')〕 235
オフラインの業務をする UAP で使える機能とプログラム 37
オフラインの業務をする UAP で使える機能と要求コード 37
オフラインの業務をする UAP に結合させるファイル 51
オフラインの業務をする UAP の開始と終了 66
オフラインの業務をする UAP の作成手順 44
オンラインテスタの管理（CBLDCUTO） 409

か

開始方法 65
回復対象外 DAM ファイル使用の開始 127
回復対象外 DAM ファイル使用の終了 101
外部変数名〔コーディング規約〕 39
会話型サービスからのメッセージの受信 480
会話型サービスとのコネクションの確立 468
会話型サービスとのコネクションの切断 473
会話型サービスへのメッセージの送信 489
環境変数 68
監査ログの出力 92
監査ログの出力（CBLDCADT） 91

き

記述形式に使う記号 415
起動するアプリケーション名を変更する 580
機能と要求コード 24

く

クライアント／サーバ形態の UAP のコーディング例（SPP　TAM アクセス） 539
クライアント／サーバ形態の UAP のコーディング例（SUP，SPP　DAM アクセス） 531
クライアント UAP（SUP，または SPP）用の XATMI インタフェース定義の形式 54
クライアント UAP から値が渡されるデータ領域 72
クライアント UAP のノードアドレスの取得 337

け

継続問い合わせ応答の終了〔CBLDCMCF('CONTEND ')〕 214
継続問い合わせ応答の終了〔DISABLE〕 427
ゲートウェイのノードアドレスの取得 341
結合 50
現在のトランザクションに関する情報の返却 510
現在のトランザクションに関する情報の報告 399

こ

コーディング規約　37
コーディング上の注意　37
コーディングする COBOL 言語　71
コーディング例　530
異なるプロセスで同じレコード型を使うとき　59
コネクションの解放　251
コネクションの確立　243
コネクションの状態取得　277
コンパイル　49

さ

サーバ UAP から，さらに別のサーバ UAP を呼び出す場合　58
サーバ UAP から値が返されるデータ領域　72
サーバ UAP の XATMI インタフェース定義に指定する項目　55
サーバ型コネクションの確立要求の受付開始　294
サーバ型コネクションの確立要求の受付終了　292
サーバ型コネクションの確立要求の受付状態取得　289
サービス関数動的ローディング機能を使用する SPP，MHP に結合させるファイル　51
サービス機能　425
サービスグループ名　76
サービスの応答待ち時間の更新　357
サービスの応答待ち時間の参照　345
サービスプログラムの作成（MHP）　81
サービスプログラムの作成（SPP）　77
サービスプログラムのリトライ　359
サービスプログラム名〔コーディング規約〕　39
サービス名の広告　458
サービス名の広告の取り消し　497
サービス要求のスケジュールプライオリティの参照　343
サービス要求のスケジュールプライオリティの設定　355
サービスルーチンからのリターン　485
サービスルーチンの開始　494
サービスを実行して結果を返す MHP のサービスプログラム　81
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　77
作成手順　40

し

シグナル　67
資源の排他　187
資源の排他制御（CBLDCLCK）　186
資源名称を指定した排他の解除　192
システム運用の管理（CBLDCADM）　83
システム運用の管理（CBLDCADM）の COBOL 言語用テンプレート　83
システムコールや任意のプログラムのライブラリを使う UAP を COBOL 言語で作成するとき　38
ジャーナルデータの編集（CBLDCJUP）　171
終了のしかたについて〔コーディング規約〕　39
終了方法　65
受信する通信イベントの形式　528
障害が起こって OpenTP1 を再開始（リラン）した場合に，タイマ起動の環境を変更する UOC　580
障害に備えて　68
処理結果の受信の拒否　333
処理結果の非同期受信　349

す

スタブ　45
スタブが必要になるアプリケーションプログラム　45
スタブのオブジェクトファイルを作成するには　49
スタブの作成手順　46
スタブの作成手順〔XATMI インタフェース用〕　54
スタブの作成手順〔併用するアプリケーションプログラム〕　588
スタブの作成方法　45
スタブのソースファイルの作成　47
スタブの翻訳（コンパイル）　49
スタブを作成する　46
スタブを作成するとき　46
スタブを作成するとき〔XATMI インタフェース用〕　60
ステータスコード　72

せ

性能検証用トレース（CBLDCPRF）　296
性能検証用トレース取得通番の通知　297
全資源の排他の解除　190

そ

ソースファイルの作成　60

た

タイプで使えるデータ型の一覧　56
タイマ起動引き継ぎ決定 UOC の関数形式　580
タイマ起動メッセージ廃棄通知イベント（ERREVT4）のデータ形式　584
タイマ起動を引き継ぐか，または取り消す　580
他社のリソースマネジャのオブジェクトファイルの結合方法　50

つ

通信記述項　414
通信記述項（CD）の記述規則　416
通信節（COMMUNICATION SECTION）の記述規則　415
通信文　414
通信文の記述規則　416

て

データ型の一覧　56
データコミュニケーション機能　419
データ操作言語　414
データ操作言語を使ったコーディング　414
データ名として指定する文字の長さ　71

と

同期型のメッセージの受信　234
同期型のメッセージの送受信　241
同期型のメッセージの送信　242
動作環境　66
特定の処理結果の受信の拒否　335
トラブルシュート関連　68
トランザクション制御（CBLDCTRN）　390
トランザクション制御用オブジェクトファイル　50
トランザクションとサービスプログラムの関係　79
トランザクションの開始〔CBLDCMCF('BEGIN ')〕　391
トランザクションの開始〔TXBEGIN〕　503
トランザクションのコミット　507
トランザクションのロールバック　514

に

任意区間でのリアルタイム統計情報の取得　364

は

バインドモード　38

ひ

引き継いだタイマ起動を即時起動とする　580
非連鎖モードのコミット　404
非連鎖モードのロールバック　407

ふ

物理ファイルからブロックの直接入力　142
物理ファイルからブロックの入力　148
物理ファイルのオープン　151
物理ファイルのクローズ　136
物理ファイルのブロックの検索　134
物理ファイルの割り当て　138
物理ファイルへブロックの出力　154
物理ファイルへブロックの直接出力　145
プログラム ID　47
プログラムの翻訳　49
プログラム名〔コーディング規約〕　39

ほ

翻訳　49
翻訳と結合　49

め

名称の付け方の注意〔XATMI インタフェース用スタブの作成〕　59
名称の付け方の注意〔コーディング規約〕　39

メインプログラム　39
メインプログラムとサービスプログラムの作成　74
メインプログラムの作成（SUP，SPP，MHP）　75
メッセージキューイング機能を使う場合　50
メッセージ送受信（CBLDCMCF）　198
メッセージ送受信機能を使う場合　50
メッセージ送受信形態の UAP のコーディング例（MHP）　543
メッセージの再送　236
メッセージの受信〔CBLDCMCF('RECEIVE ')〕　228
メッセージの受信〔RECEIVE〕　420
メッセージの送信〔CBLDCMCF('SEND ')〕　240
メッセージの送信〔SEND〕　424
メッセージログの出力　195
メッセージログの出力（CBLDCLOG）　194

ゆ

ユーザ固有の性能検証用トレースの取得　299
ユーザサーバの開始処理完了の報告　87
ユーザサーバの状態の報告　89
ユーザサーバのテスト状態の報告　410
ユーザサーバのテスト状態を報告する場合　50
ユーザジャーナルの取得（CBLDCJNL）　168
ユーザジャーナルの取得〔CBLDCJNL('UJPUT ')〕　169
ユーザジャーナルの取得〔SEND〕　444
ユーザタイマ監視の設定　273
ユーザタイマ監視の取り消し　271

り

リクエスト／レスポンス型サービスからの非同期応答の受信　475
リクエスト／レスポンス型サービスのキャンセル　466
リクエスト／レスポンス型サービスの呼び出し　454
リクエスト／レスポンス型サービスの呼び出しと応答の受信　460
リソースマネジャ集合のオープン　512
リソースマネジャ集合のクローズ　505
リソースマネジャ接続先選択　401
リモートプロシジャコール（CBLDCRPC，CBLDCRSV）　310
リンケージ　50

る

ルートトランザクションブランチ　32

れ

レコード型　55
連鎖モードのコミット　393
連鎖モードのロールバック　396

ろ

論理端末の出力キュー削除　259
論理端末の状態取得　286
論理端末の閉塞　255
論理端末の閉塞解除　247
論理ファイルからブロックの入力　110
論理ファイルのオープン　105
論理ファイルのクローズ　98
論理ファイルの状態の参照　123
論理ファイルのブロックの更新　116
論理ファイルの閉塞　103
論理ファイルの閉塞の解除　120
論理ファイルへブロックの出力　129