HITACHI
Inspire the Next

OpenTP1 Version 7
分散トランザクション処理機能

OpenTP1 プロトコル TP1/NET/User Agent 編

解説・手引・文法・操作書

3000-3-D71-20

前書き

■ 対象製品

・適用 OS：AIX V6.1，AIX V7.1，AIX V7.2

P-1M64-3141 uCosminexus TP1/Message Control 07-51

P-1M64-3241 uCosminexus TP1/NET/Library 07-51

P-F1M64-32411 uCosminexus TP1/NET/User Agent 07-50

・適用 OS：Red Hat Enterprise Linux Server 6 (32-bit x86)，Red Hat Enterprise Linux Server 6 (64-bit x86_64)，Red Hat Enterprise Linux Server 7(64-bit x86_64)

P-8164-3111 uCosminexus TP1/Message Control 07-51

P-8164-3211 uCosminexus TP1/NET/Library 07-51

P-F8164-32111 uCosminexus TP1/NET/User Agent 07-50

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

■ 輸出時の注意

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

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

■ 発行

2017 年 10 月　3000-3-D71-20

■ 著作権

変更内容

変更内容（3000-3-D71-20）uCosminexus TP1/Message Control　07-51，uCosminexus TP1/NET/Library　07-51，uCosminexus TP1/NET/User Agent　07-50

追加・変更内容	変更個所
マニュアル訂正の内容を反映した。	－

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-D71-10）uCosminexus TP1/Message Control　07-51，uCosminexus TP1/NET/Library　07-51，uCosminexus TP1/NET/User Agent　07-50

追加・変更内容
通信管理プログラムに XNF/LS を使用する場合の説明を追加した。
コネクションの確立，解放，および状態表示について，ライブラリ関数および COBOL-UAP 作成用プログラムインタフェースを使用する場合の説明を追加した。
論理端末の閉塞，閉塞解除，および状態表示について，ライブラリ関数および COBOL-UAP 作成用プログラムインタフェースを使用する場合の説明を追加した。
NULL またはヌル文字列設定時のコーディング例を追加した。
リターン値 DCMCFRTN_73001 およびステータスコード 73001 の説明を変更した。
リターン値 DCMCFRTN_NOMSG およびステータスコード 70904 の説明を変更した。
データ操作言語の通信文と C 言語のライブラリ関数の対応の説明を追加した。
ERREVT3 に設定するトランザクションブランチ ID の形式の説明を追加した。
MCF 性能検証用トレースの説明を追加した。
システムサービス共通情報定義の，次に示すオペランドの指定値の上限を拡張した。 • max_socket_descriptors • max_open_fds
定義オブジェクトファイルの出力先に読み取り権限を持つファイルがすでにある場合，定義オブジェクトファイルを上書きするかどうかを指定できるようにした。
OpenTP1 システムを変更する場合に，見直しが必要な定義と，変更手順について説明を追加した。
アプリケーション起動サービスに関する説明を追加した。
スタート関数を呼び出したあとの注意事項を追加した。
次に示すバージョンの変更点を記載した。 • TP1/NET/User Agent 07-50 • TP1/NET/User Agent 07-00

追加・変更内容
バージョン 6 以前のインタフェースの変更一覧を追加した。

はじめに

このマニュアルは，OpenTP1 システムの中で，TP1/NET/User Agent を使用した AP 間通信システムの作成方法，および運用方法について説明したものです。

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

OpenTP1 システムの通信に User Agent プロトコルを使用するシステム管理者，システム設計者，およびプログラマを対象としています。また，オンラインや OpenTP1 システムの基礎的な知識を持っていて，次のマニュアルを理解されていることを前提としています。

- OpenTP1 解説 （3000-3-D50）
- OpenTP1 プログラム作成の手引 （3000-3-D51）
- OpenTP1 システム定義 （3000-3-D52）
- OpenTP1 運用と操作 （3000-3-D53）
- OpenTP1 プログラム作成リファレンス C 言語編 （3000-3-D54）
- OpenTP1 プログラム作成リファレンス COBOL 言語編 （3000-3-D55）

■ マニュアルの構成

このマニュアルは，次に示す章と付録から構成されています。

第 1 章 概要

TP1/NET/User Agent を使用した AP 間通信の概要について説明しています。

第 2 章 機能

TP1/NET/User Agent のコネクション，論理端末，およびメッセージ送受信に関する機能について説明しています。

第 3 章 C 言語のライブラリ関数

TP1/NET/User Agent で使用できる，C 言語のライブラリ関数について説明しています。

第 4 章 COBOL-UAP 作成用プログラムインタフェース

TP1/NET/User Agent で使用できる，COBOL-UAP 作成用プログラムインタフェースについて説明しています。

第５章　ユーザオウンコーディング，MCF イベントインタフェース

TP1/NET/User Agent に関連するユーザオウンコーディング，および MCF イベントインタフェースについて説明しています。

第６章　システム定義

OSAS/UA プロトコルを使用するために必要な，OpenTP1 のシステム定義の中での TP1/NET/User Agent 固有のシステム定義，および定義例について説明しています。

第７章　運用コマンド

TP1/NET/User Agent で使用する運用コマンドについて説明しています。

第８章　組み込み方法

TP1/NET/User Agent を OpenTP1 システムに組み込む方法について説明しています。

第９章　障害対策

TP1/NET/User Agent 運用中に発生する障害と，TP1/NET/User Agent の対応処理，およびメッセージの処理について説明しています。

付録 A　バージョンアップ時の変更点

各バージョンでの関数，定義およびコマンドの変更点について説明しています。

付録 B　旧製品からの移行に関する注意事項

バージョン６以前からバージョン７へ移行する場合の注意事項について説明しています。

付録 C　インタフェースの変更一覧（バージョン６以前から移行する場合）

バージョン６以前からバージョン７に移行する場合のインタフェースの変更個所について説明しています。

付録 D　メッセージ送受信の処理の流れ

メッセージを送受信するときのデータの流れ，ジャーナル取得のタイミングについて説明しています。

付録 E　障害発生時の処理の流れ

障害が発生した場合の処理の流れについて説明しています。

付録 F　MCF 性能検証用トレースの取得

MCF 性能検証用トレースの取得について説明しています。

付録 G　ユーザアプリケーションプログラムの作成例

TP1/NET/User Agent のユーザアプリケーションプログラムの作成例について説明しています。

付録 H　理由コード一覧

障害通知イベントが発生した場合の理由コードについて説明しています。

付録 I　このマニュアルの参考情報

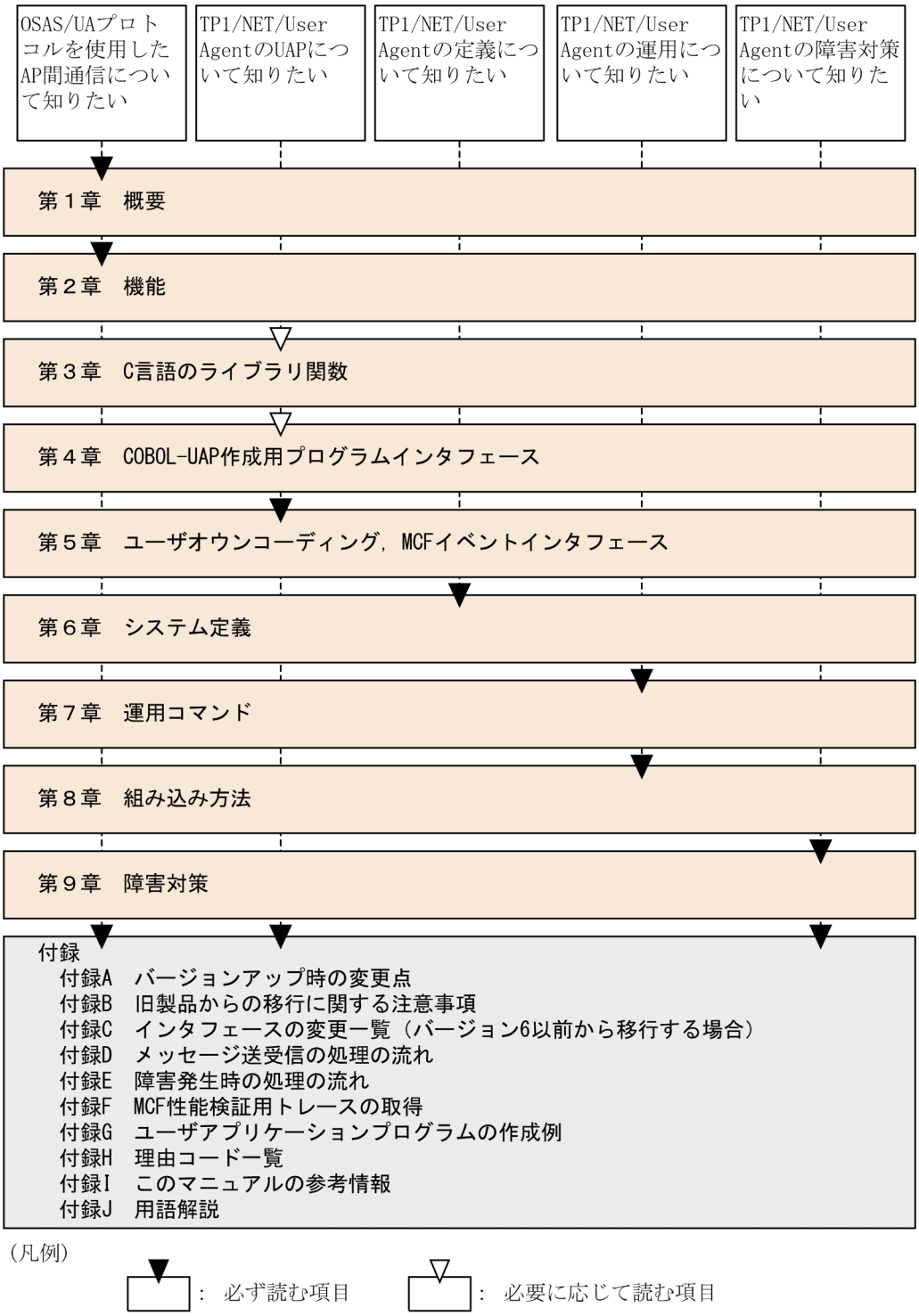
関連マニュアル，このマニュアルで使用している略語の意味などを説明しています。

付録 J　用語解説

TP1/NET/User Agent で使用する用語について説明しています。

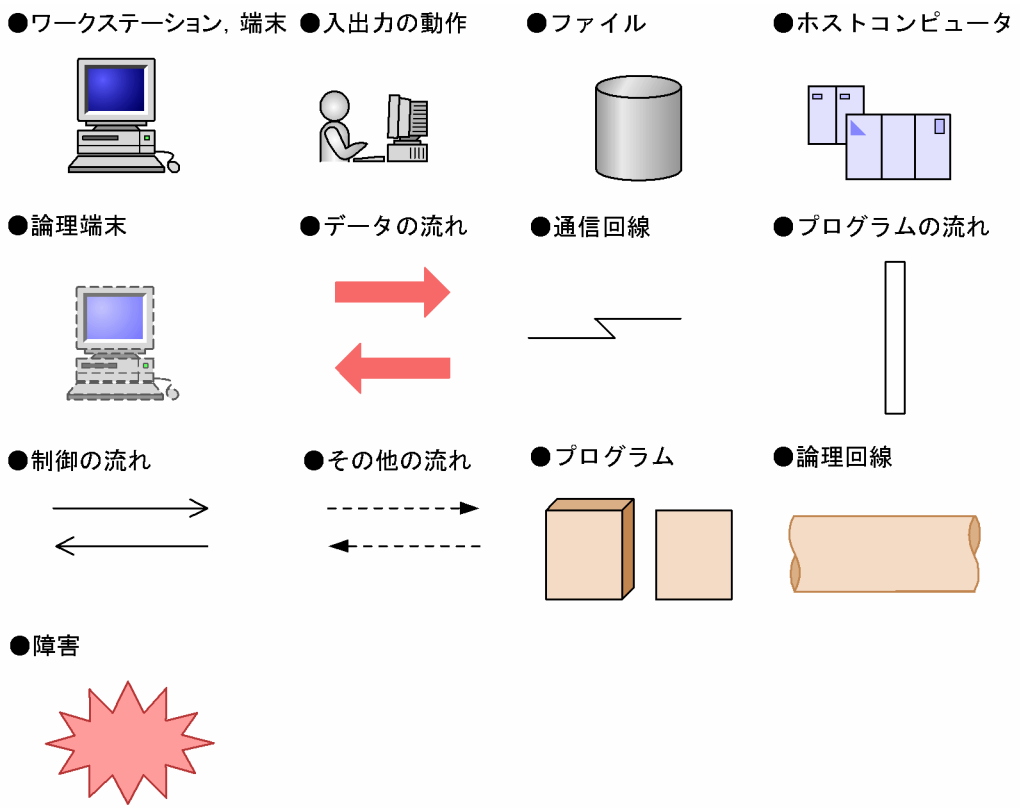
■ 読書手順

このマニュアルは，利用目的に合わせて章を選択して読むことができます。利用目的別に，次の流れに従ってお読みいただくことをお勧めします。



■ 図中で使用する記号

このマニュアルの図中で使用する記号を，次のように定義します。



■ 文法の記号

このマニュアルで使用する各種の記号を説明します。

(1)文法記述記号

文法の記述形式について説明する記号です。

文法記述記号	意味
〔　〕	この記号で囲まれている項目は省略できることを示します。 (例)〔-s MCF 通信プロセス識別子〕 　-s オプションとそのオペランドを指定するか，何も指定しないことを示します。
| (ストローク)	この記号で区切られた項目は選択できることを示します。 (例) -t reply | request 　-t オプションに reply または request を指定できることを示します。 ただし，C 言語のインタフェースの説明でこの記号を使用した場合は，C 言語の文法規則に従います。
{　}	この記号で囲まれている複数の項目のうちから一つを選択できることを示します。 (例) {DCMCFESI | DCMCFEMI} 　DCMCFESI と DCMCFEMI のうち，どちらかを指定できることを示します。
_ (下線)	この記号で示す項目は，オペランド，オプションまたはコマンド引数を省略した場合の省略時解釈値を示します。

文法記述記号	意味
_ （下線）	（例）-i auto | manual -i オプションを省略した場合，manual を省略時解釈値とすることを示します。 ただし，データ操作言語の説明の場合，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。
…	この記号で示す直前の一つの項目を繰り返し指定できることを示します。 ただし，項目が括弧で囲まれている場合，括弧全体が一つの項目となります。
△ （白三角）	空白を示します。 （例）コネクション ID1△コネクション ID2 コネクション ID1 とコネクション ID2 の間に，空白を 1 個入力することを示します。

(2)属性表示記号

ユーザ指定値の範囲などを説明する記号です。

属性表示記号	意味
～	この記号のあとにユーザ指定値の属性を示します。
《　》	ユーザが指定を省略したときの省略時解釈値を示します。
＜　＞	ユーザ指定値の構文要素を示します。
((　))	ユーザ指定値の指定範囲を示します。

(3)構文要素記号

ユーザ指定値の内容を説明する記号です。

構文要素記号	意味
＜英字＞	アルファベット（A～Z，a～z）と_（アンダスコア）
＜英字記号＞	アルファベット（A～Z，a～z）と#，@，¥
＜英数字＞	英字と数字（0～9）
＜英数字記号＞	英字記号と数字（0～9）
＜符号なし整数＞	数字列（0～9）
＜ 10 進数字＞	数字（0～9）
＜ 16 進数字＞	数字（0～9）と（A～F，a～f）
＜識別子＞	先頭がアルファベットの英数字列
＜記号名称＞	先頭が英字記号の英数字記号列
＜文字列＞	任意の文字の配列
＜パス名＞	記号名称，／，および.（ピリオド）

構文要素記号	意味
＜パス名＞	（ただし，パス名は使用する OS に依存）

■ 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 のユーザ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。

目次

1 概要

TP1/NET/User Agent は，OpenTP1 システムの OSAS/UA プロトコルを支援するプログラムです。この章では，TP1/NET/User Agent の概要について説明します。

1.1　AP 間通信の概要

AP 間通信とは，異なるシステムにあるアプリケーションプログラム間でのメッセージ送受信のことです。

TP1/NET/User Agent は，OpenTP1 システムに組み込まれ，OSAS/UA プロトコルを使用した AP 間通信を提供するプログラムです。TP1/NET/User Agent を使用すると，拡張 HNA の OSAS/UA プロトコルによってほかのオンラインシステムや分散機との水平，垂直分散型ネットワークシステムを構築できます。

OSAS/UA プロトコルとは，OSI 上位層（5 層，6 層，および 7 層の一部）の共通基盤である OSAS を使用して，AP 間通信サービスを提供するプロトコルです。

OpenTP1 システムを使用したネットワーク構成例を次の図に示します。

また，TP1/NET/User Agent の OSI 7 層構造の中での機能範囲を図 1-2 に示します。

図 1-1　OpenTP1 システムを使用したネットワーク構成例

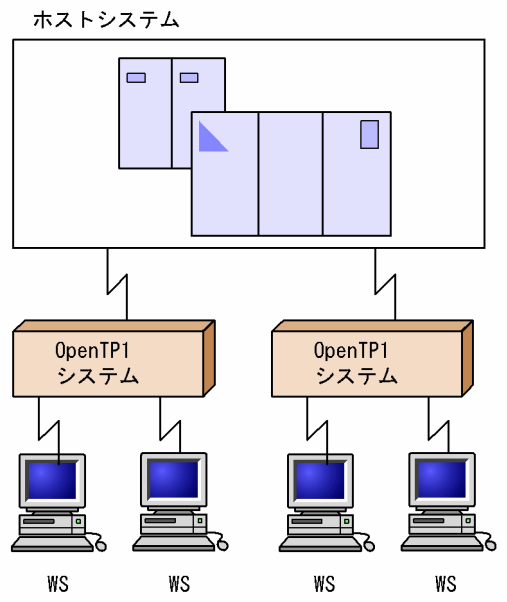
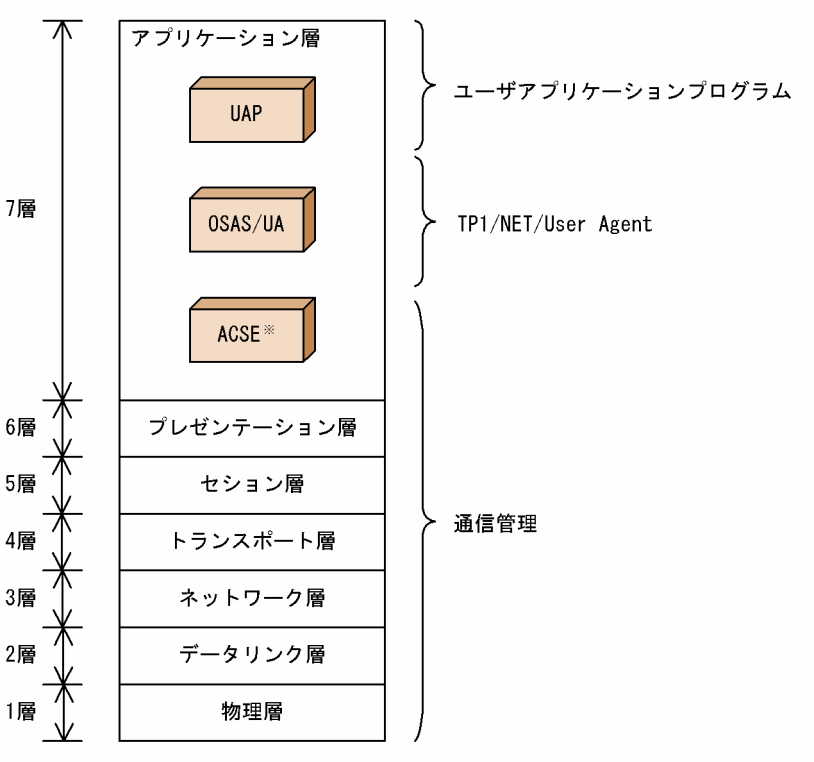


図 1-2　TP1/NET/User Agent の OSI7 層構造の中での機能範囲



注※

ACSE：アソシエーション制御サービス要素

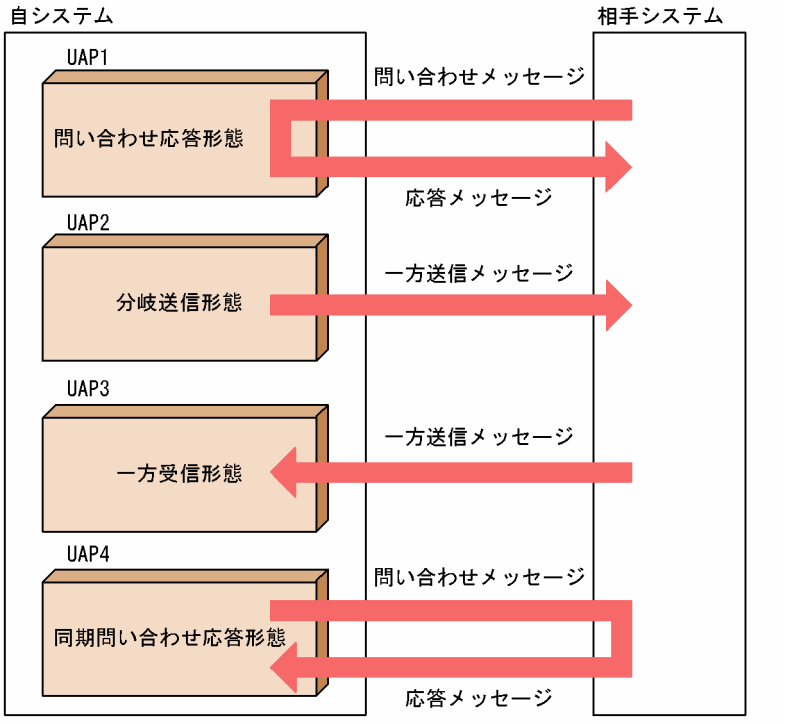
1.2　AP 間通信の形態

AP 間通信を使用すると，自システムで発生したトランザクションを相手システムで処理したり，その結果を受信したりできます。また，相手システムで発生したトランザクションを自システムで処理したり，その結果を送信したりできます。

TP1/NET/User Agent を使用した AP 間通信の形態には，問い合わせ応答形態，分岐送信形態，一方受信形態，および同期問い合わせ応答形態があります。

TP1/NET/User Agent を使用した AP 間通信の例を次の図に示します。

図 1-3　TP1/NET/User Agent を使用した AP 間通信の例



1.2.1　問い合わせ応答形態

問い合わせ応答形態とは，相手システムから問い合わせメッセージを受信して，応答メッセージを送信する通信形態です。

1.2.2　分岐送信形態

分岐送信形態とは，自システムから一方送信メッセージを相手システムへ送信する通信形態です。

1.2.3　一方受信形態

一方受信形態とは，相手システムから一方送信メッセージを受信する通信形態です。

1.2.4　同期問い合わせ応答形態

同期問い合わせ応答形態とは，相手システムに問い合わせメッセージを送信し，応答メッセージを受信する通信形態です。

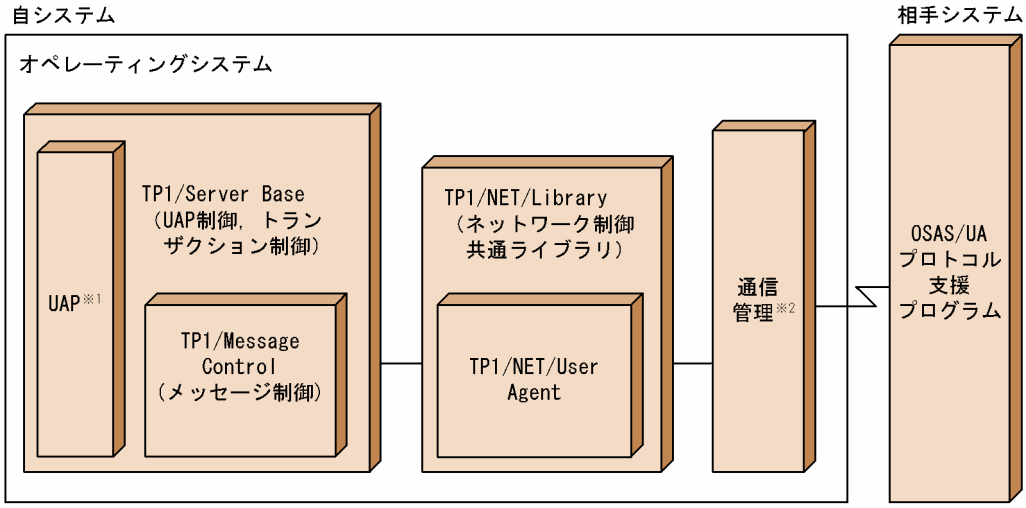
1.3　ソフトウェアの構成の例

TP1/NET/User Agent は，OpenTP1 システムに組み込まれて動作するプログラムです。OpenTP1 のメッセージ送受信機能(TP1/Message Control，TP1/NET/Library)と連携して，メッセージ制御機能（MCF）を実現します。

TP1/NET/User Agent を組み込んだソフトウェア構成の例を次の図に示します。

また，TP1/NET/User Agent が AP 間通信で使用するプロトコル，および主な通信相手プログラムを表 1-1 に示します。

図 1-4　TP1/NET/User Agent を組み込んだソフトウェア構成の例



注※1

TP1/NET/User Agent で扱う UAP は，MHP および SPP です。UAP 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。

注※2

XNF/LS または XNF/AS を使用できます。

表 1-1　TP1/NET/User Agent が適用する AP 間通信のプロトコル

プロトコル	主な通信相手
OSAS/UA プロトコル	VOS3　TMS-4V/SP の OSAS/UA/4VSP VOS3　XDM/DCCM3 の OSAS/UA2/DCCM3

2 機能

TP1/NET/User Agent は AP 間通信を提供し，メッセージの送受信をします。この章では，コネクションの確立や論理端末の端末タイプなどの AP 間通信の仕組み，およびメッセージの送受信について説明します。

2.1　AP 間通信の仕組み

TP1/NET/User Agent は，UA プロトコルデータ単位を最小の単位にして，メッセージの送受信を行います。TP1/NET/User Agent で送受信する UA プロトコルデータ単位について，次の表に示します。

表 2-1　TP1/NET/User Agent の UA プロトコルデータ単位

UA プロトコルデータ単位	機能概要	備考
UINT	初期設定	発呼だけのサポートです。
UOPN	個別開局	発呼だけのサポートです。
UEND	終了	－
UINQ	問い合わせ	UREP の応答が必要です。
UREP	応答	UINQ に対して必要です。
UBRD	一方送信	－
UERR	例外報告	－
UINF	状態通知	－

（凡例）

－：備考はありません。

TP1/NET/User Agent の AP 間通信の仕組みについて説明します。

2.1.1　コネクションの確立と解放

TP1/NET/User Agent は，相手システムとの間で，論理的な通信路（**コネクション**）を確立してメッセージを送受信します。コネクションは，OSAS/UA プロトコルのアソシエーションに対応します。

OSAS/UA プロトコルでは，データ送受信の多重化は UA で行います。

このため自システムと相手システムの同一 PSAP アドレス間では，1 本のコネクションしか確立できません。

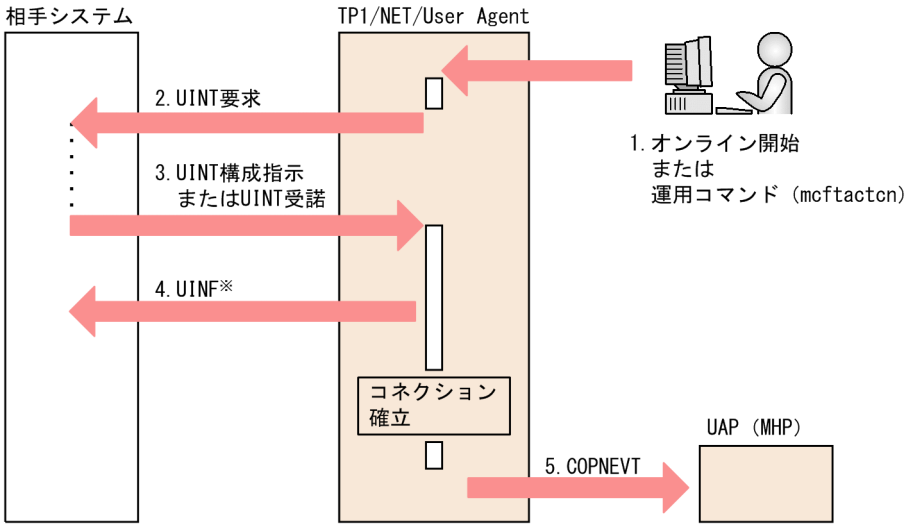
(1)　コネクションの確立

TP1/NET/User Agent からコネクションを確立します。コネクションの確立方法には，次の三つがあります。

- コネクション定義（mcftalccn -i）での auto 指定によるオンラインの開始・再開始時の自動確立
- 運用コマンド（mcftactcn）入力による手動確立
- API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）の発行による手動確立

運用コマンド（mcftactcn）入力時またはオンライン開始時の，TP1/NET/User Agent からのコネクションの確立について，次の図に示します。

図 2-1　TP1/NET/User Agent からのコネクションの確立（運用コマンド入力時またはオンライン開始時）

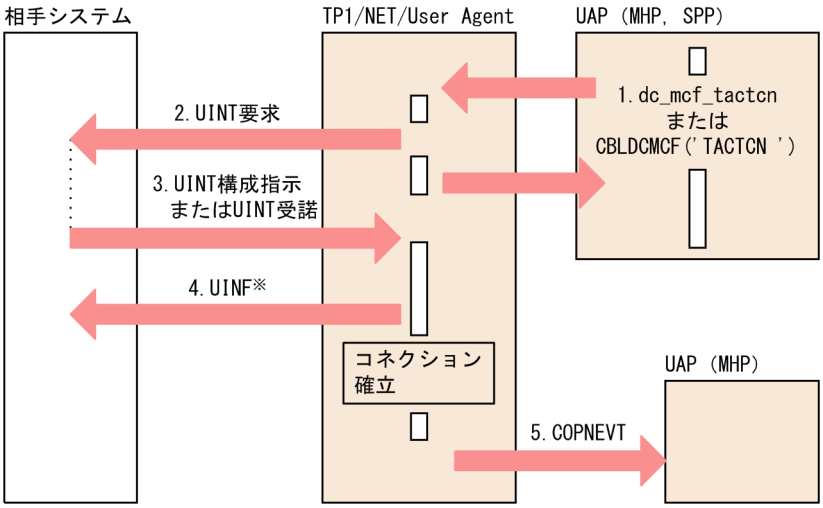


注※　相手システムからUINT構成指示を受信した場合だけ送信します。

1. OpenTP1 システムを開始・再開始します。または運用コマンド（mcftactcn）を入力します。
2. TP1/NET/User Agent は，MCF 通信構成定義情報に基づいて UINT 要求を作成し，相手システムに送信します。
3. 相手システムから UINT 構成指示または UINT 受諾を受信します。
4. 相手システムから UINT 構成指示を受信した場合には，処理完了 UINF を相手システムに送信すると，コネクションが確立します。UINT 受諾を受信した場合には，UINT 受諾受信を契機にコネクションが確立します。
5. TP1/NET/User Agent は，コネクションが確立すると状態通知イベント（COPNEVT）を通知します。

API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）発行時の，TP1/NET/User Agent からのコネクションの確立について，次の図に示します。

図 2-2　TP1/NET/User Agent からのコネクションの確立（API 発行時）



注※ 相手システムからUINT構成指示を受信した場合だけ送信します。

1. API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）を発行します。
2. TP1/NET/User Agent は，MCF 通信構成定義情報に基づいて UINT 要求を作成し，相手システムに送信します。
3. 相手システムから UINT 構成指示または UINT 受諾を受信します。
4. 相手システムから UINT 構成指示を受信した場合には，処理完了 UINF を相手システムに送信すると，コネクションが確立します。UINT 受諾を受信した場合には，UINT 受諾受信を契機にコネクションが確立します。
5. TP1/NET/User Agent は，コネクションが確立すると状態通知イベント（COPNEVT）を通知します。

コネクション確立時に，自システムと相手システムの UA の数を突き合わせます。UA の数が不一致の場合，一致しない UA を使用不可（閉塞状態）として，一致する UA だけが使用可能となります。

コネクション確立時の，UA の数による TP1/NET/User Agent の処理を次の表に示します。

表 2-2　UA の数による TP1/NET/User Agent のコネクション確立

項番	UA の数	TP1/NET/User Agent の処理
1	相手システム＜自システム	不一致となった UA の論理端末を使用不可とし，一致した UA の論理端末を使用可能としてコネクションを確立します。 mcftalccn コマンドに-f オプションを指定した場合は，相手システムの構成に合わせて，自システムを縮退することを通知するメッセージ（KFCA13257-W）が出力されます。
2	相手システム＝自システム	該当するコネクション下の全論理端末を使用してコネクションを確立します。
3	相手システム＞自システム	コネクションの確立を拒否します。
4	相手システム＝ 0	コネクションの確立を拒否します。

ただし，MCF 通信構成定義で「個別開局」と指定された UA に関しては，該当する UA に関する論理端末を閉塞扱いとし，運用コマンド（mcftactle）または API（dc_mcf_tactle 関数または CBLDCMCF('TACTLE△△')）で閉塞を解除するまではメッセージ送受信ができません。また，表 2-2 の項番 1 で使用不可となった論理端末については，閉塞を解除できません。

(2) コネクションの解放

コネクションは，次に示す場合，正常に解放します。

- 相手システムから解放要求された場合
- 運用コマンド（mcftdctcn）を入力した場合
- API（dc_mcf_tdctcn 関数または CBLDCMCF('TDCTCN△△')）を発行した場合
- MCF 通信構成共通定義（mcftcomn -x）の termrls オペランドに nomal を指定して，オンラインを終了した場合

なお，次の場合はコネクションを強制的に解放し，障害通知イベント（CERREVT）を通知します。

- 運用コマンド（mcftdctcn -f）を受け付けた場合
- 強制解放オプション※を指定した API（dc_mcf_tdctcn 関数または CBLDCMCF('TDCTCN△△')）を発行した場合
- 送受信バッファ，および編集バッファの資源不足が発生した場合
- TP1/NET/User Agent 内で回復できない論理矛盾が発生した場合
- MCF 通信プロセスが異常終了した場合（通信管理の機能に依存します。この場合は CERREVT を通知しません）
- 相手システムに何らかの異常が発生し，コネクションが解放された場合

注※

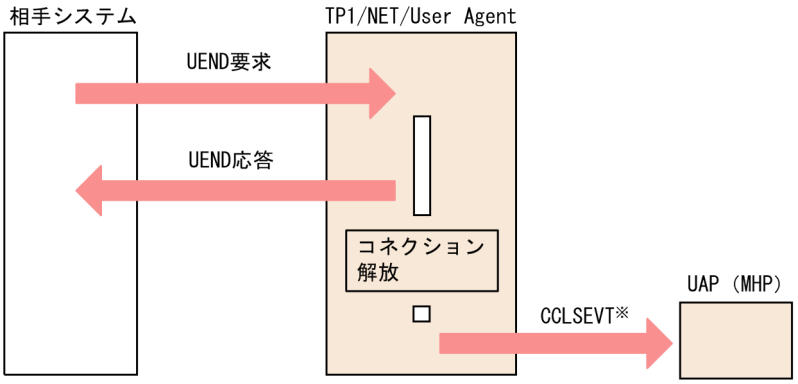
dc_mcf_tdctcn 関数の場合，action 引数に DCMCFFRC を指定します。
CBLDCMCF('TDCTCN△△')の場合，データ名 D1 に'1'を指定します。

コネクションの正常解放はユーザ間で，データ転送の終了の同期合わせ後に行ってください。

相手システムからのコネクションの正常解放を図 2-3 に示します。運用コマンド（mcftdctcn）入力時またはオンライン終了時の自システムからのコネクションの解放を図 2-4 に示します。また，API（dc_mcf_tdctcn 関数または CBLDCMCF('TDCTCN△△')）発行時の自システムからのコネクションの解放を図 2-5 に示します。

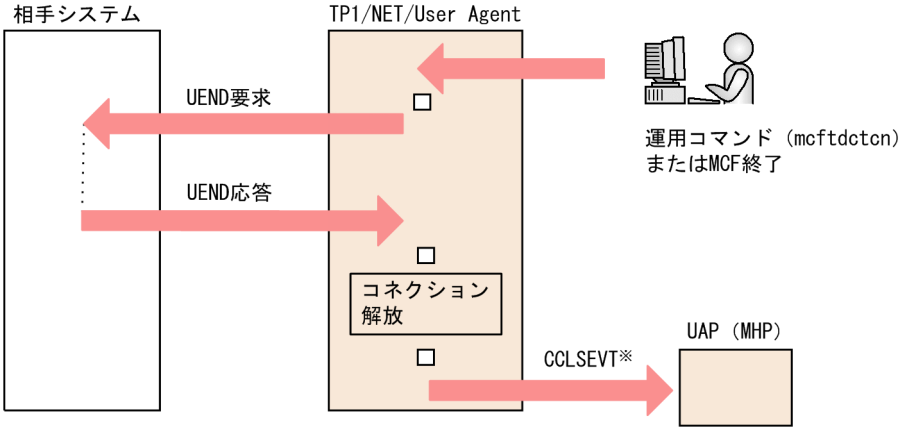
コネクションが正常に解放されると，TP1/NET/User Agent は，状態通知イベント（CCLSEVT）を通知します。

図 2-3　相手システムからのコネクションの解放



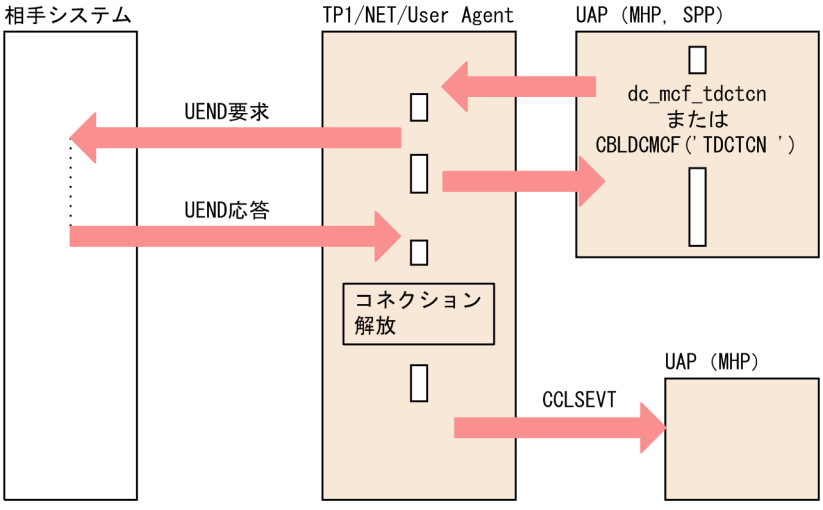
注※　オンライン終了時は通知しません。

図 2-4　自システムからのコネクションの解放（運用コマンド入力時またはオンライン終了時）



注※　オンライン終了時は通知しません。

図 2-5　自システムからのコネクションの解放（API 発行時）

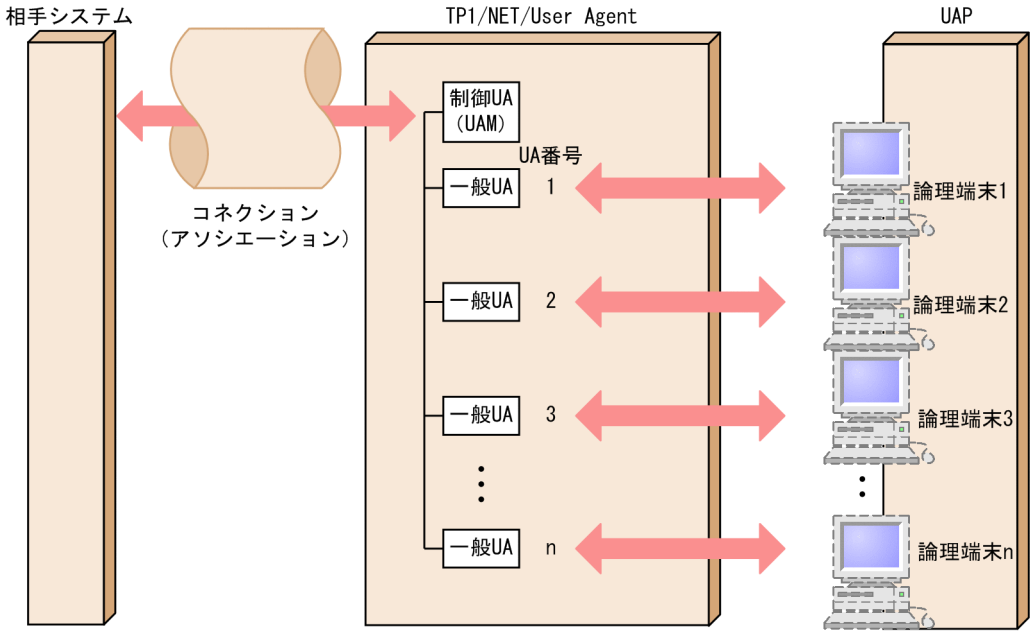


2.1.2　コネクションと論理端末の関係

TP1/NET/User Agent で扱う定義で，ユーザが意識するものとしては，**コネクション**と**論理端末**があります。コネクションは OSAS/UA プロトコルのアソシエーションに対応し，論理端末（LE）は OSAS/UA プロトコルの一般 UA に対応します。

コネクションは TP1/NET/User Agent の相手システム側の通信接点であり，TP1/NET/User Agent と相手システムはコネクション単位にメッセージの送受信をします。論理端末は TP1/NET/User Agent の UAP 側の通信接点であり，TP1/NET/User Agent と UAP は論理端末単位に送受信をします。コネクションと論理端末の関係を次の図に示します。

図 2-6　コネクションと論理端末の関係



2.1.3　論理端末とアプリケーションの型の関係

論理端末は，相手システムとの間でメッセージを送受信するためのユーザインタフェースであり，OSAS/UA プロトコルの UA に対応します。

論理端末は，送受信するメッセージの形態で次に示す四つの端末タイプがあります。**論理端末の端末タイプ**は，MCF の論理端末属性として，論理端末定義（mcftalcle -t）に指定します。

- request 型：問い合わせ型論理端末
- reply 型：応答型論理端末
- send 型：送信型論理端末
- receive 型：受信型論理端末

アプリケーションは，ユーザが送受信データの中で指定した**アプリケーション名**※をキーとして一つのUAP（MHP）プロセスで実行されます。アプリケーションには，処理形態によって二つの型があり，MCFのアプリケーション属性の一つとして，アプリケーション属性定義（mcfaalcap -n）の type オペランドに指定します。受信メッセージによって起動する**アプリケーションの型**を次に示します。

- 応答型：受信メッセージに対して応答する MHP の型
- 非応答型：受信メッセージに対して応答しない MHP の型

注※

アプリケーション名は，メッセージの先頭から空白の手前までの 1 から 8 バイトの識別子です。先頭から 9 バイト目に空白がないとき，または先頭に空白がある場合はアプリケーション名を不正とします。入力メッセージ編集 UOC で決定する場合は，「5.1.1　入力メッセージの編集とアプリケーション名の決定」を参照してください。

論理端末，メッセージ，アプリケーションの型，UAP インタフェース，および通信形態の関係を次の表に示します。

表 2-3　論理端末，メッセージ，アプリケーションの型，UAP インタフェース，および通信形態の関係

論理端末の端末タイプ	メッセージの種類※	受信メッセージのアプリケーションの型	UAP インタフェース	通信形態
reply （応答型論理端末）	問い合わせメッセージ （UINQ）	応答型	receive	問い合わせ応答
	応答メッセージ （UREP）		reply	
request （問い合わせ型論理端末）	一方送信メッセージ （UINQ）	任意	send	分岐送信
	問い合わせメッセージ （UINQ）		sendrecv （セグメント送信時，先頭セグメント受信時）	同期問い合わせ応答
	応答メッセージ （UREP）		recvsync （後続セグメント受信時）	
	一方送信メッセージ （UREP）	非応答型	receive	一方受信
send （送信型論理端末）	一方送信メッセージ （UBRD）	任意	send	分岐送信
receive （受信型論理端末）		非応答型	receive	一方受信

注※

（ ）内は，UA プロトコルデータ単位を示します。

2.1.4 メッセージの分割と組み立て

TP1/NET/User Agent では，次の三つのデータ形式のメッセージを処理します。

- UA プロトコルデータ単位（UPDU）
- UA サービスデータ単位（UASDU）
- 論理メッセージ

以上三つのメッセージの分割，組み立てをしてメッセージの送受信をすると，各種資源（メモリ，回線など）を節約できます。

(1) UA プロトコルデータ単位

メッセージの最小データ送受信単位で，先頭，中間，最終 UPDU から構成されています。問い合わせ応答，および一方送信のデータ単位と，その他の制御用データ単位に分類されます。

(2) UA サービスデータ単位

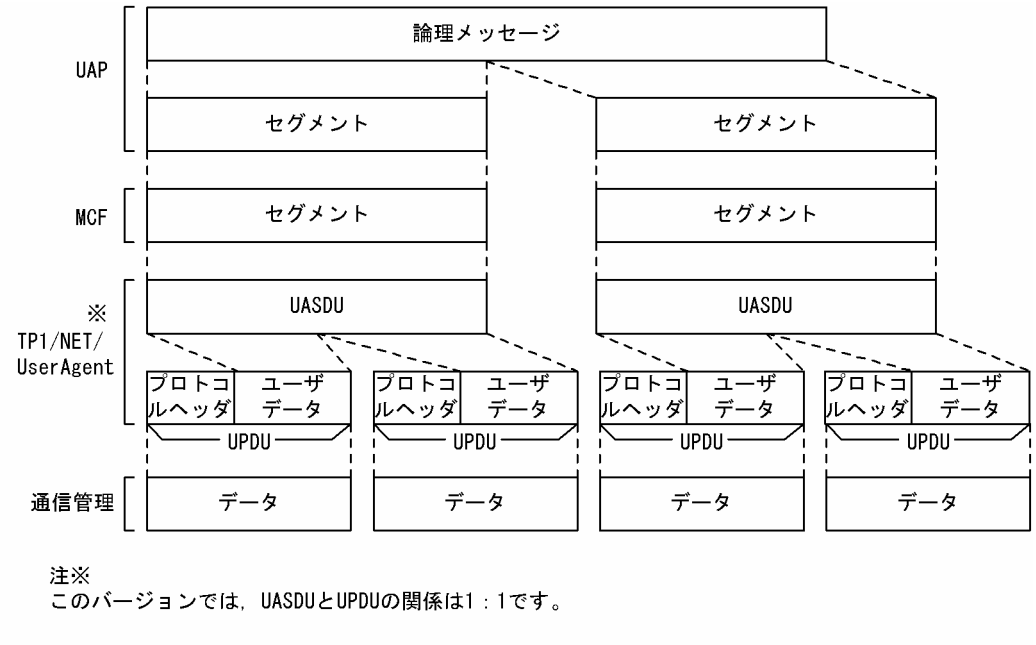
UPDU を組み立てた UA サービス利用者のデータ処理単位，セグメントに相当します。MCF 内のメッセージ送受信の基本単位で，UA サービスプリミティブに対応し，UPDU と同様に，先頭，中間，最終 UASDU で構成されています。

(3) 論理メッセージ

複数セグメントから構成され，UAP が論理的に処理するデータ単位で，MHP の起動単位となります。入出力メッセージ編集 UOC による変換を伴う場合があります。

メッセージのデータ形式を次の図に示します。

図 2-7　メッセージのデータ形式



TP1/NET/User Agent では，論理メッセージ単位に次のようなメッセージの送受信をします。

- 論理メッセージ受信
  UA サービスデータ単位に受信したメッセージを 1 論理メッセージにまとめて受信します。
- 論理メッセージ送信
  1 論理メッセージを UA サービスデータ単位に送信します。

2.1.5　UA の開局と閉局

TP1/NET/User Agent は，相手システムとのコネクションの確立時，UA を開局してメッセージを送受信します。

(1)　UA の開局

UA の開局方法には，次の三つがあります。

- コネクション確立時の自動開局
- 運用コマンド（mcftactle）入力による個別開局
- API（dc_mcf_tactle 関数または CBLDCMCF('TACTLE△△')）の発行による個別開局

コネクション確立時に，「個別開局」指定をした場合，または論理端末に障害があって閉局した場合は，次のどちらかの方法で開局します。

- 運用コマンド（mcftactle）
- API（dc_mcf_tactle 関数または CBLDCMCF('TACTLE△△')）

すべての UA の開局モードが個別開局（コネクション定義（mcftalccn -k）に each を指定）の場合，該当するコネクション内のすべての UA に関連する論理端末を閉塞扱いします。なお，論理端末が閉塞状態であっても，該当する論理端末に対する UAP からの送信要求はできます。ただし，メッセージは出力キューに格納するので，論理端末定義（mcftalcle -m）で指定したメモリ出力メッセージ最大格納数またはディスク出力メッセージ最大格納数を超えない範囲が限度となります。

(a)　コネクション確立時の UA の自動開局

すべての UA の開局モードが一括開局（コネクション定義（mcftalccn -k）に together を指定）の場合，コネクション確立時に，すべての UA がメッセージ送受信可能状態になります。

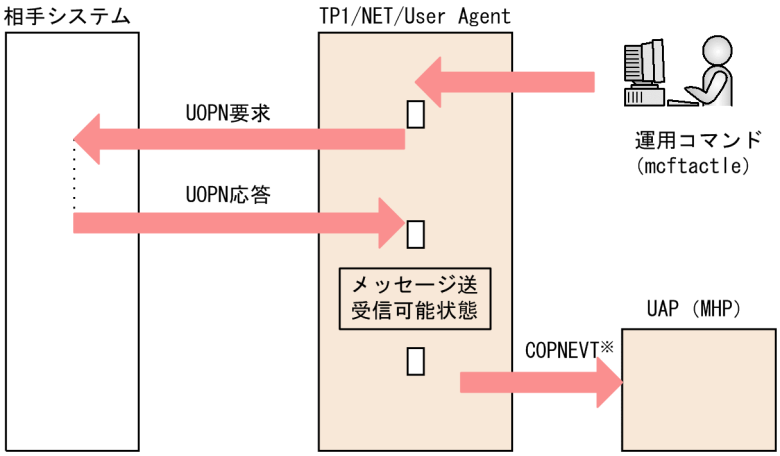
UA が開局すると，該当する UA の論理端末の閉塞状態を解除します。論理端末の閉塞解除時に状態通知イベント（COPNEVT）を起動する場合（コネクション定義（mcftalccn -f）の leopnevt オペランドに use を指定）は，COPNEVT を通知します。

(b)　運用コマンドによる UA の個別開局

TP1/NET/User Agent は，開局されていない UA に対して運用コマンド（mcftactle）を受け付けると，相手システムに UOPN 要求を送信します。そして相手システムから UOPN 応答を受信すると，UA が開局し，メッセージ送受信可能状態になります。

UA が開局すると，該当する UA の論理端末の閉塞状態を解除します。運用コマンドによる UA の個別開局の処理の流れを次の図に示します。

図 2-8　運用コマンドによる UA の個別開局



注※

論理端末の閉塞解除時に状態通知イベント（COPNEVT）を起動する場合（コネクション定義（mcftalccn -f）の leopnevt オペランドに use を指定）だけ通知します。

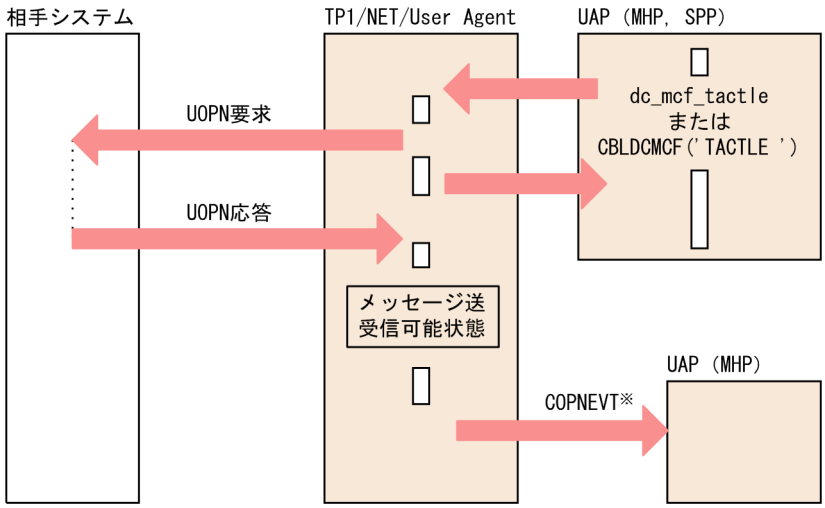
(c)　API による UA の個別開局

TP1/NET/User Agent は，開局されていない UA に対して API（dc_mcf_tactle 関数または CBLDCMCF('TACTLE△△')）を発行すると，相手システムに UOPN 要求を送信します。そして，TP1/

NET/User Agent が相手システムから UOPN 応答を受信すると，UA が開局し，メッセージ送受信可能状態になります。

UA が開局すると，該当する UA の論理端末の閉塞状態を解除します。API による UA の個別開局の処理の流れを次の図に示します。

図 2-9　API による UA の個別開局



注※

論理端末の閉塞解除時に状態通知イベント（COPNEVT）を起動する場合（コネクション定義（mcftalccn -f）の leopnevt オペランドに use を指定）だけ通知します。

(2)　UA の閉局

UA の閉局方法には，次の四つがあります。

- 運用コマンド（mcftdctle）入力による強制閉局
- API（dc_mcf_tdctle 関数または CBLDCMCF('TDCTLE△△')）の発行による強制閉局
- 相手システムからの強制閉局
- MCF 検出障害時の強制閉局

また，TP1/NET/User Agent は UA を閉局するとき，UERR を使用します。

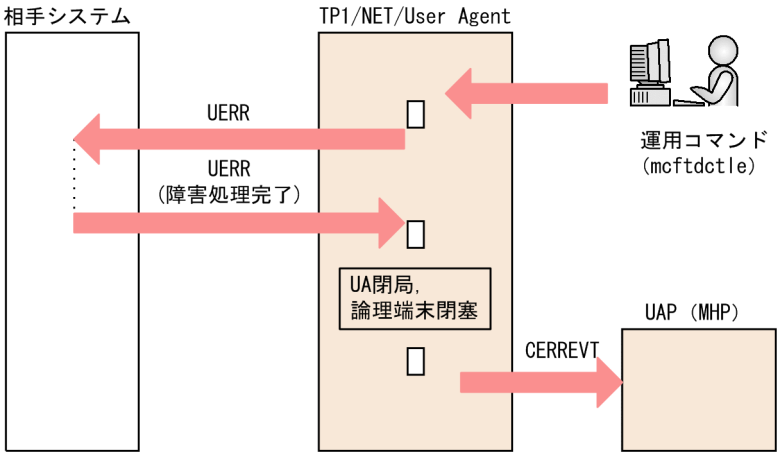
(a)　運用コマンドによる UA の閉局

TP1/NET/User Agent は，運用コマンド（mcftdctle）を受け付けると，UERR を送信します。そして相手システムからの UERR（障害処理完了）を受信し，UA が閉局します。

UA が閉局すると，TP1/NET/User Agent は関連する論理端末を閉塞すると同時に，障害通知イベント（CERREVT）を起動します。

運用コマンドによる UA の閉局の処理の流れを次の図に示します。

図 2-10　運用コマンドによる UA の閉局



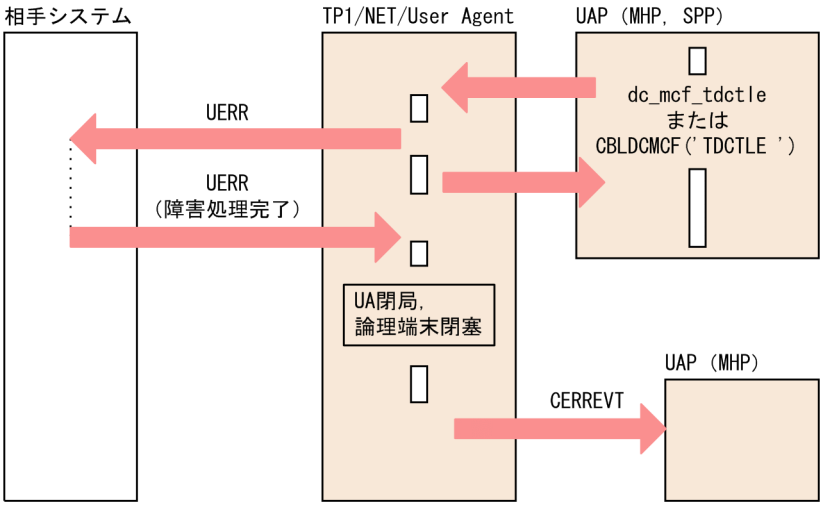
(b)　API による UA の閉局

TP1/NET/User Agent は，API（dc_mcf_tdctle 関数または CBLDCMCF('TDCTLE△△')）を発行すると，UERR を送信します。そして相手システムからの UERR（障害処理完了）を受信し，UA が閉局します。

UA が閉局すると，TP1/NET/User Agent は関連する論理端末を閉塞すると同時に，障害通知イベント（CERREVT）を起動します。

API による UA の閉局の処理の流れを次の図に示します。

図 2-11　API による UA の閉局

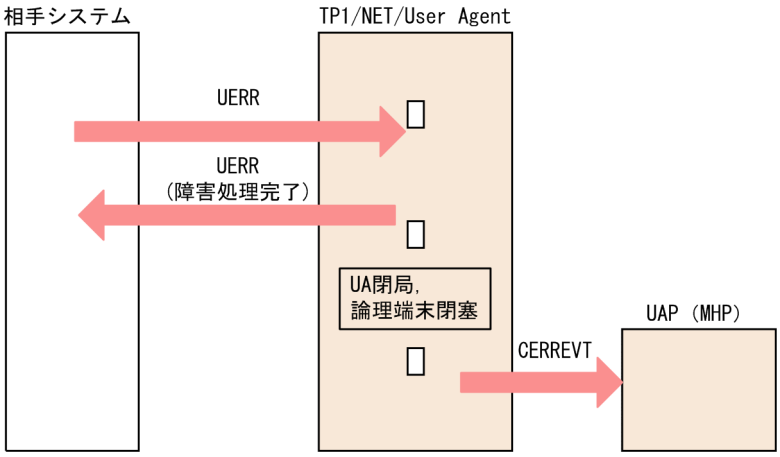


(c)　相手システムからの UA の閉局

相手システムから UERR を受信すると，TP1/NET/User Agent では UERR（障害処理完了）を送信し，UA を閉局します。UA を閉局すると，TP1/NET/User Agent は関連する論理端末を閉塞すると同時に，障害通知イベント（CERREVT）を起動します。

相手システムからの UA の閉局の処理の流れを次の図に示します。

図 2-12　相手システムからの UA の閉局



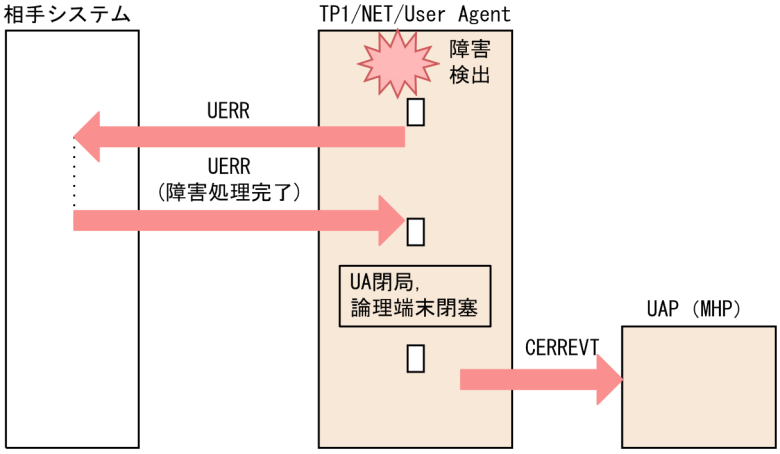
(d)　MCF 検出障害時の UA の閉局

TP1/NET/User Agent は，UAP 異常終了，キュー障害などを検出した場合，UERR を送信します。そして相手システムからの UERR（障害処理完了）を受信し，UA が閉局します。

UA が閉局すると，TP1/NET/User Agent は関連する論理端末を閉塞すると同時に，障害通知イベント（CERREVT）を起動します。

MCF 検出障害時の UA の閉局の処理の流れを次の図に示します。

図 2-13　MCF 検出障害時の UA の閉局



2.2　AP 間通信メッセージの送受信

TP1/NET/User Agent は，AP 間通信メッセージの送受信を制御します。TP1/NET/User Agent で扱う AP 間通信メッセージには次の四つがあります。

- 問い合わせメッセージ
- 応答メッセージ
- 一方送信メッセージ（分岐送信）
- 一方送信メッセージ（一方受信）

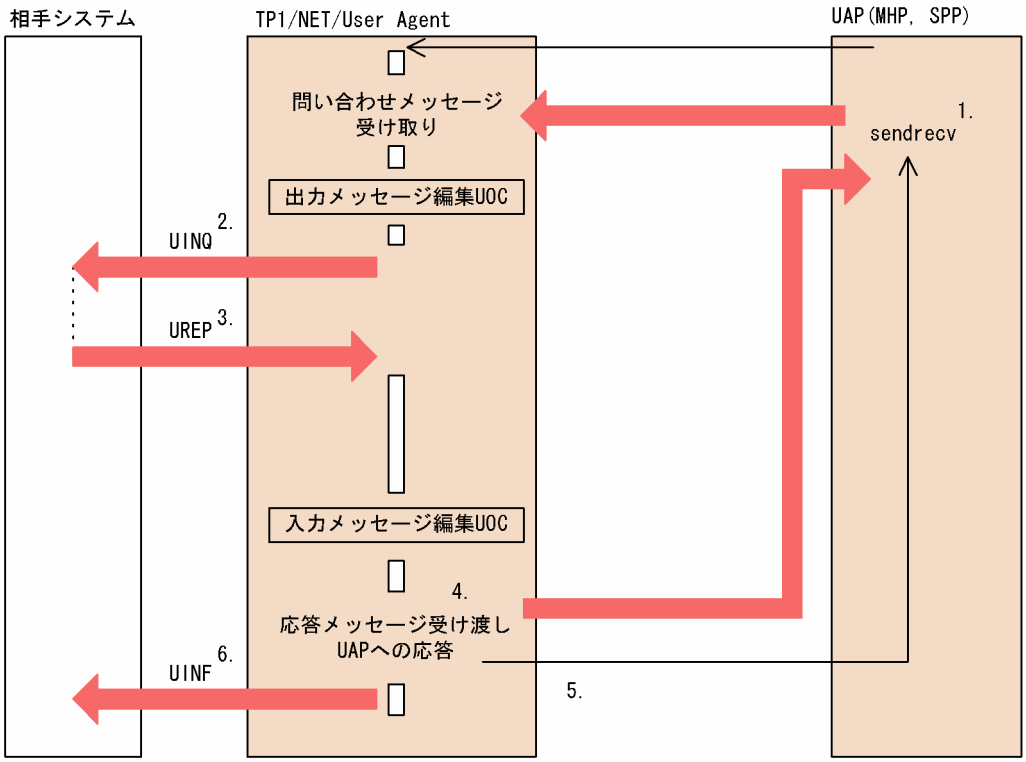
それぞれのメッセージの送受信について説明します。

2.2.1　問い合わせメッセージの送信

TP1/NET/User Agent が，相手システムに問い合わせメッセージ（UINQ）を送信し，相手システムからの応答メッセージ（UREP）を受信する処理の流れを説明します。この場合，問い合わせメッセージ関数（sendrecv）を発行した UAP に，応答メッセージが返ります。

自システムからの問い合わせメッセージの送信と，それに対する相手システムからの応答メッセージを受信する処理の流れを次の図に示します。

図 2-14　問い合わせメッセージの送信と応答メッセージの受信



1. MHP または SPP から request 型論理端末へ，同期型の問い合わせメッセージを送信します。

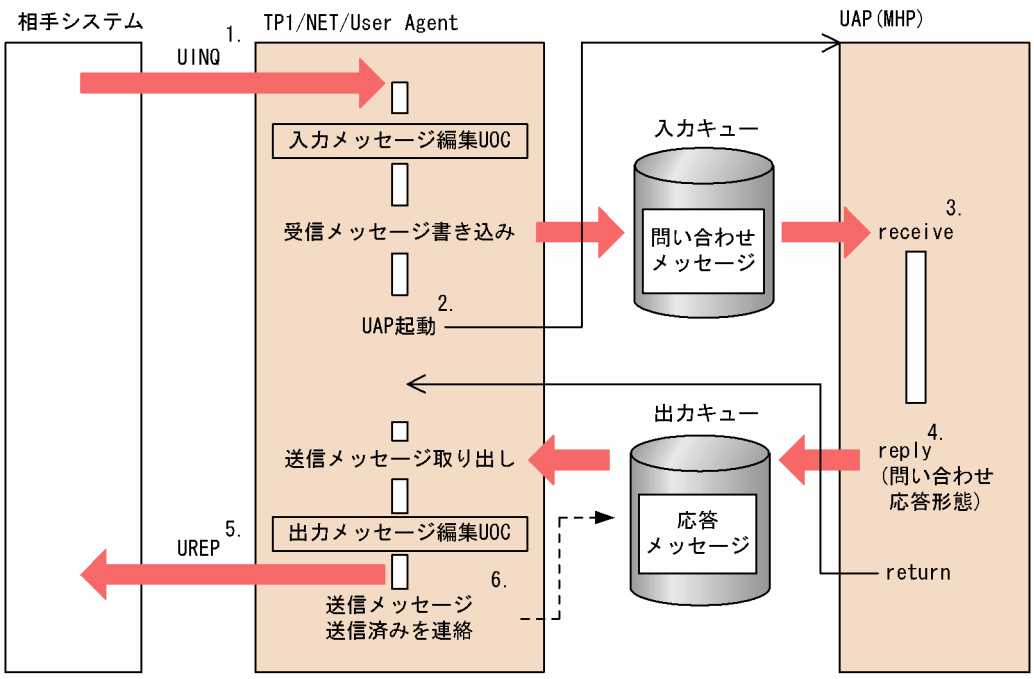
2. TP1/NET/User Agent は相手システムに UINQ を送信します。
3. 相手システムからの UREP を受信します。
4. 相手システムからの応答メッセージを UAP に受け渡します。
5. 問い合わせメッセージを送信した UAP に制御が渡され，応答メッセージを受信します。
6. 3.の UREP の処理完了報告が必要な場合，UAP へのリターンが正常に終了した時点で，UINF を送信します。

2.2.2 応答メッセージの送信

TP1/NET/User Agent が，相手システムからの問い合わせメッセージ（UINQ）を受信し，相手システムに応答メッセージ（UREP）を送信する処理の流れを説明します。

相手システムからの問い合わせメッセージの受信と，それに対する応答メッセージの送信の処理の流れを次の図に示します。

図 2-15 問い合わせメッセージの受信と応答メッセージの送信



1. 相手システムから reply 型論理端末あての UINQ（問い合わせメッセージ）を受信します。
2. TP1/NET/User Agent は相手システムからの問い合わせメッセージを入力キューに書き込み，MHP を起動します。
3. MHP は関数を発行してメッセージを受け取ります。
4. MHP は，応答メッセージを発行します。
5. MHP が正常に return した場合，相手システムに UREP（処理完了報告不要）を送信します。

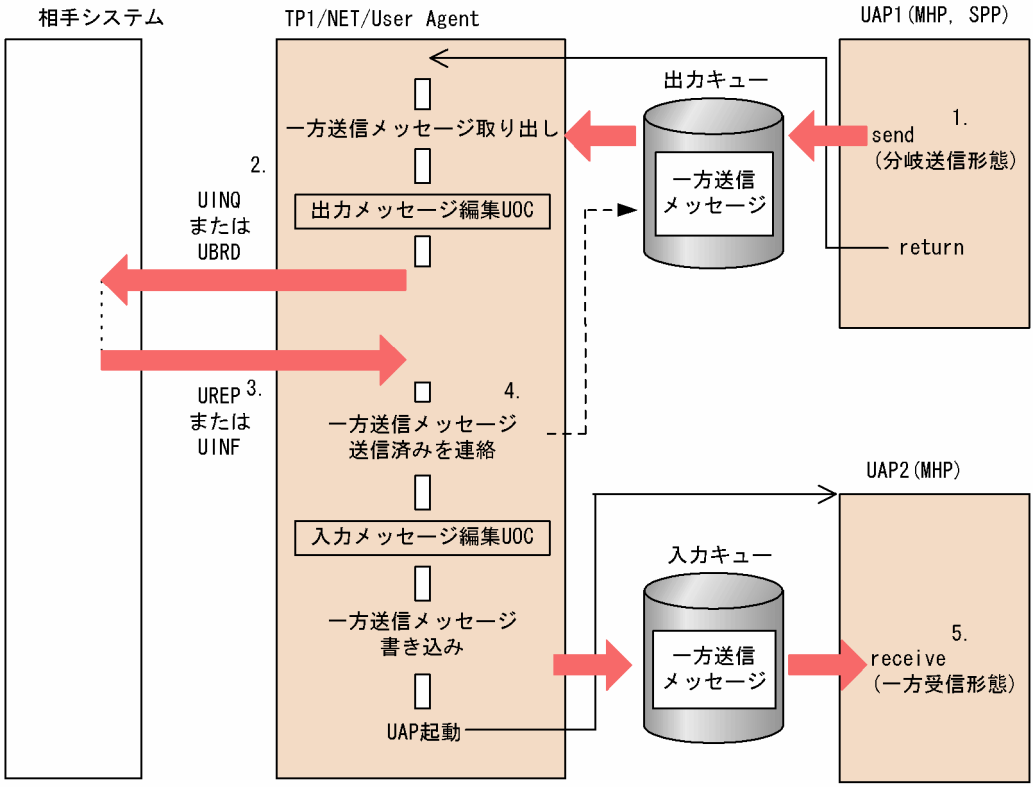
6.5.のあと，出力キューの応答メッセージを送信済みにします。

2.2.3　一方送信メッセージの送信と受信

TP1/NET/User Agent が，相手システムに**一方送信メッセージ**（UINQ または UBRD）を送信し，相手システムからの一方送信メッセージ（UREP または UINF）を受信する場合の処理の流れを説明します。

一方送信メッセージの送信と受信の処理の流れを，次の図に示します。

図 2-16　一方送信メッセージの送信と受信



1. MHP または SPP から request 型論理端末，または send 型論理端末あてに，一方送信メッセージを送信します。
2. TP1/NET/User Agent は相手システムに UINQ（send 型の場合は UBRD）を送信します。
3. 相手システムからの UREP（send 型の場合は UINF）を受信します。
4. 3.の時点で，送信メッセージを送信済みとします。
5. 送信時とは別の MHP に制御が渡され，一方送信メッセージを受信します。

   なお，send 型の場合は，UINF を TP1/NET/User Agent で受信するだけで，MHP での UINF の受信はできません。

3 C 言語のライブラリ関数

この章では，TP1/NET/User Agent で使用できる，C 言語のライブラリ関数について説明します。

C 言語のライブラリ関数の一覧

TP1/NET/User Agent で使用する C 言語のライブラリ関数の一覧を，次の表に示します。

表 3-1　C 言語のライブラリ関数の一覧

関数名	機能
dc_mcf_receive	メッセージの受信
dc_mcf_recvsync	同期型の応答メッセージの受信
dc_mcf_reply	応答メッセージの送信
dc_mcf_resend	メッセージの再送
dc_mcf_send	一方送信メッセージの送信
dc_mcf_sendrecv	同期型のメッセージの送受信
dc_mcf_tactcn	コネクションの確立
dc_mcf_tactle	論理端末の閉塞解除
dc_mcf_tdctcn	コネクションの解放
dc_mcf_tdctle	論理端末の閉塞
dc_mcf_tlscn	コネクションの状態取得
dc_mcf_tlsle	論理端末の状態取得

なお，UAP 作成の詳細については，マニュアル「OpenTP1 プログラム作成の手引」を参照してください。その他の関数については，マニュアル「OpenTP1 プログラム作成リファレンス C 言語編」を参照してください。

NULL またはヌル文字列設定時のコーディング例

C 言語のライブラリ関数の引数に NULL またはヌル文字列を設定する場合のコーディング例を示します。

NULL を設定する場合

```
char *resv01=NULL;
dc_mcf_receive(..., resv01, ...);
```

ヌル文字列を設定する場合

```
char resv01[1]="¥0";
dc_mcf_receive(..., resv01, ...);
```

注

resv01 以外の dc_mcf_receive 関数の引数は省略しています。

dc_mcf_receive － メッセージの受信（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_receive(DCLONG action, DCLONG commform, char *termnam,
                    char *resv01, char *recvdata,
                    DCLONG *rdataleng, DCLONG inbufleng,
                    DCLONG *time)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int  dc_mcf_receive(action, commform, termnam, resv01,
                    recvdata, rdataleng, inbufleng, time)
DCLONG     action;
DCLONG     commform;
char       *termnam;
char       *resv01;
char       *recvdata;
DCLONG     *rdataleng;
DCLONG     inbufleng;
DCLONG     *time;
```

機能

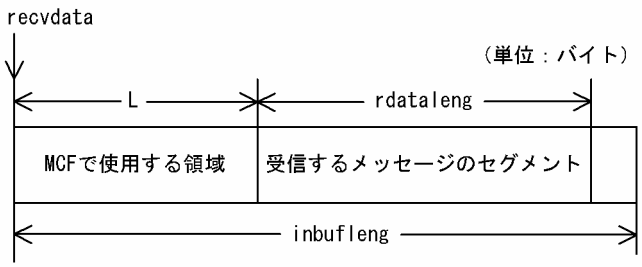
論理端末に届いたメッセージのうち，一つのセグメントを受信します。セグメントの数だけ dc_mcf_receive 関数を呼び出すと，一つの論理メッセージを受信できます。

受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

dc_mcf_receive 関数で受信できるメッセージの種類を次に示します。

- 相手システムから送信されたメッセージ
- MCF イベント
- アプリケーション起動で渡されたメッセージ

セグメントを受信する領域の形式を次に示します。L は，バッファ形式 1 の場合は 8 バイト，バッファ形式 2 の場合は 4 バイトです。



UAP で値を設定する引数

●action

受信するセグメント，および使用するバッファ形式を，次の形式で設定します。

```
{DCMCFFRST|DCMCFSEG} 〔|{DCMCFBUF1|DCMCFBUF2}〕
```

DCMCFFRST

先頭セグメントを受信する場合や，論理メッセージが単一セグメントの場合に設定します。

DCMCFSEG

中間セグメントまたは最終セグメントを受信する場合に設定します。

DCMCFBUF1

バッファ形式 1 を使用する場合に設定します。

DCMCFBUF2

バッファ形式 2 を使用する場合に設定します。

●commform

DCNOFLAGS を設定します。

●termnam

先頭セグメントまたは単一セグメントを受信する場合は，メッセージ入力元の論理端末名称を受け取る領域を設定します。

処理終了後，termnam には OpenTP1 から値が返ります。

中間セグメントまたは最終セグメントを受信する場合は，先頭セグメントの受信時に返されたメッセージ入力元の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

中間セグメントまたは最終セグメントを受信した場合，値は返されません。

●resv01

NULL またはヌル文字列を設定します。

●recvdata

セグメントを受信する領域を設定します。

dc_mcf_receive 関数が終了すると，メッセージのセグメントの一つが返されます。

処理終了後，recvdata には OpenTP1 から値が返ります。

●inbufleng

セグメントを受信する領域の長さを設定します。

OpenTP1 から値が返される引数

●termnam

先頭セグメントまたは単一セグメントを受信する場合，入力元の論理端末名称が返されます。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字が付けられます。

中間セグメントまたは最終セグメントを受信する場合は，ここで返された論理端末名称を termnam に設定してください。

●recvdata

受信したセグメントの内容が返されます。

●rdataleng

受信したセグメントの長さが返されます。

●time

メッセージを受信した時刻が，1970 年 1 月 1 日 0 時 0 分 0 秒からの通算の秒数で返されます。

リターン値

リターン値	リターン値 (数値)	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71000	-12000	先頭セグメントを受信する dc_mcf_receive 関数を 2 回以上呼び出しています。中間セグメントまたは最終セグメントを受信する場合は，action に DCMCFSEG を設定して dc_mcf_receive 関数を呼び出してください。
DCMCFRTN_71001	-12001	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 dc_mcf_receive 関数を呼び出しています。直前に呼び出した dc_mcf_receive 関数でメッセージはすべて受信しました。 このリターン値が返されたあとに，再び dc_mcf_receive 関数を呼び出した場合は，リターン値 DCMCFRTN_72000 が返されます。
DCMCFRTN_71002	-12002	メッセージキューからの入力処理中に障害が発生しました。
		メッセージキューが閉塞されています。
DCMCFRTN_72000	-13000	＜ 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 dc_mcf_receive 関数を呼び出す前に，中間セグメントまたは最終セグメントを受信する dc_mcf_receive 関数を呼び出しています。先頭セグメントを受信する場合は，action に DCMCFFRST を設定して dc_mcf_receive 関数を呼び出してください。

リターン値	リターン値（数値）	意味
DCMCFRTN_72000	-13000	• リターン値 DCMCFRTN_71001 が返されたあとに，再び dc_mcf_receive 関数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 dc_mcf_receive 関数を呼び出せません。
DCMCFRTN_72001	-13001	termnam に設定した論理端末名称が間違っています。
DCMCFRTN_72013	-13013	inbufleng の指定値を超えるセグメントを受信しました。inbufleng の指定値を超えた部分は切り捨てられました。
DCMCFRTN_72016	-13016	action に設定した値が間違っています。
		resv01 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72024	-13024	commform に設定した値が間違っています。
DCMCFRTN_72025	-13025	action に設定したセグメント種別（DCMCFFRST または DCMCFSEG）の値が間違っています。
DCMCFRTN_72036	-13036	inbufleng の指定値が不足しています。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。

（凡例）

－：該当しません。

dc_mcf_recvsync － 同期型のメッセージの受信（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_recvsync(DCLONG action, DCLONG commform, char *termnam,
                     char *resv01, char *recvdata,
                     DCLONG *rdataleng, DCLONG inbufleng,
                     DCLONG *time, DCLONG resv02)
```

K&R 版 C の形式

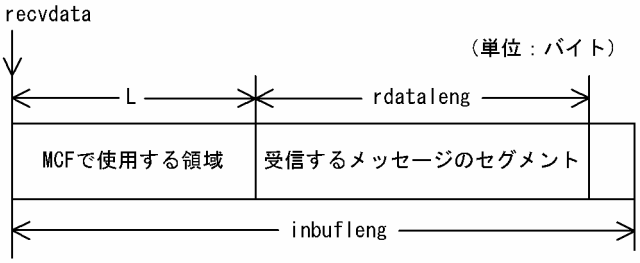
```
#include <dcmcf.h>
int  dc_mcf_recvsync(action, commform, termnam, resv01,
                     recvdata, rdataleng, inbufleng,
                     time, resv02)
DCLONG     action;
DCLONG     commform;
char       *termnam;
char       *resv01;
char       *recvdata;
DCLONG     *rdataleng;
DCLONG     inbufleng;
DCLONG     *time;
DCLONG     resv02;
```

機能

相手システムから届いた同期型の受信メッセージのうち，先頭セグメント以外の後続する一つのセグメントを受信します。先頭セグメントを受信する dc_mcf_sendrecv 関数のあとに，後続するセグメントの数だけ dc_mcf_recvsync 関数を呼び出すと，一つの論理メッセージを受信できます。

受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを受信する領域の形式を次に示します。L は，バッファ形式 1 の場合は 8 バイト，バッファ形式 2 の場合は 4 バイトです。



UAP で値を設定する引数

●action

受信するセグメント，および使用するバッファ形式を，次の形式で設定します。

```
DCMCFSEG〔|{DCMCFBUF1|DCMCFBUF2}〕
```

DCMCFSEG

　中間セグメントまたは最終セグメントの受信を示す DCMCFSEG を設定します。

DCMCFBUF1

　バッファ形式 1 を使用する場合に設定します。

DCMCFBUF2

　バッファ形式 2 を使用する場合に設定します。

●commform

DCNOFLAGS を設定します。

●termnam

dc_mcf_sendrecv 関数の termnam パラメタで指定した入力元の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

●resv01

NULL またはヌル文字列を設定します。

●recvdata

セグメントを受信する領域を設定します。

dc_mcf_recvsync 関数が終了すると，メッセージのセグメントの一つが返されます。

処理終了後，recvdata には OpenTP1 から値が返ります。

●inbufleng

セグメントを受信する領域の長さを設定します。

●resv02

DCNOFLAGS を設定します。

OpenTP1 から値が返される引数

●recvdata

受信したセグメントの内容が返されます。

●rdataleng

受信したセグメントの長さが返されます。

●time

メッセージを受信した時刻が，1970 年 1 月 1 日 0 時 0 分 0 秒からの通算の秒数で返されます。

リターン値

リターン値	リターン値(数値)	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	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 dc_mcf_recvsync 関数を呼び出しています。直前に呼び出した dc_mcf_recvsync 関数でメッセージはすべて受信しました。 このリターン値が返されたあとに，再び次のセグメントを受信する dc_mcf_recvsync 関数を呼び出した場合は，リターン値 DCMCFRTN_72000 が返されます。
DCMCFRTN_71108	-12108	メッセージ受信に必要な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。
DCMCFRTN_72000	-13000	先頭セグメントを受信する dc_mcf_receive 関数を呼び出す前に，dc_mcf_recvsync 関数を呼び出しています。
		リターン値 DCMCFRTN_71001 が返されたあとに，再び dc_mcf_recvsync 関数を呼び出しています。
DCMCFRTN_72001	-13001	termnam に設定した論理端末名称が間違っています。
		termnam に設定した論理端末名称は，定義されていません。
		dc_mcf_recvsync 関数を呼び出せない論理端末を設定しています。
DCMCFRTN_72013	-13013	inbufleng の指定値を超えるセグメントを受信しました。inbufleng の指定値を超えた部分は切り捨てられました。
DCMCFRTN_72016	-13016	action に設定した値が間違っています。
		resv01 または resv02 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72024	-13024	commform に設定した値が間違っています。
DCMCFRTN_72025	-13025	action に設定したセグメント種別(DCMCFFRST または DCMCFSEG)の値が間違っています。
DCMCFRTN_72036	-13036	inbufleng の指定値が不足しています。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。

（凡例）

－：該当しません。

dc_mcf_reply － 応答メッセージの送信（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_reply(DCLONG action, DCLONG commform, char *resv01,
                  char *resv02, char *senddata, DCLONG sdataleng,
                  char *resv03, DCLONG opcd)
```

K&R 版 C の形式

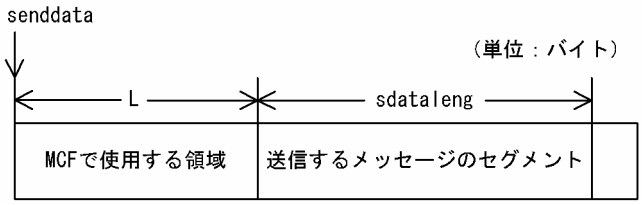
```
#include <dcmcf.h>
int  dc_mcf_reply(action, commform, resv01, resv02, senddata,
                  sdataleng, resv03, opcd)
DCLONG     action;
DCLONG     commform;
char       *resv01;
char       *resv02;
char       *senddata;
DCLONG     sdataleng;
char       *resv03;
DCLONG     opcd;
```

機能

問い合わせをしたシステムへ送る応答メッセージのうち，一つのセグメントを送信します。セグメントの数だけ dc_mcf_reply 関数を呼び出すと，一つの論理メッセージを送信できます。

送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域の形式を次に示します。L は，バッファ形式 1 の場合は 8 バイト，バッファ形式 2 の場合は 4 バイトです。



UAP で値を設定する引数

●action

送信するセグメント，出力通番を付けるかどうか，および使用するバッファ形式を，次の形式で設定します。

```
{DCMCFESI|DCMCFEMI} 〔|{DCMCFSEQ|DCMCFNSEQ}〕 〔|{DCMCFBUF1|DCMCFBUF2}〕
```

DCMCFESI

先頭セグメントまたは中間セグメントを送信する場合に設定します。

DCMCFEMI

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。

DCMCFSEQ

出力通番を付ける場合に設定します。

ただし，非応答型のアプリケーションの場合は，DCMCFSEQ を指定しても，MCF は応答メッセージに出力通番を付けません。

DCMCFNSEQ

出力通番を付けない場合に設定します。

DCMCFBUF1

バッファ形式 1 を使用する場合に設定します。

DCMCFBUF2

バッファ形式 2 を使用する場合に設定します。

●commform

DCNOFLAGS を設定します。

●resv01

ヌル文字列を設定します。

●resv02

NULL またはヌル文字列を設定します。

●senddata

送信するセグメントの内容を設定した領域を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，必ず設定してください。

●sdataleng

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

●resv03

ヌル文字列を設定します。

●opcd

DCNOFLAGS を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71002	-12002	メッセージキューへの出力処理中に障害が発生しました。
		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。
		sdataleng 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，メッセージの送信を受け付けられません。
DCMCFRTN_71003	-12003	メッセージキューが満杯です。
DCMCFRTN_71004	-12004	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。
DCMCFRTN_71108	-12108	メッセージを送信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。
DCMCFRTN_72000	-13000	＜ 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 dc_mcf_receive 関数を呼び出す前に，dc_mcf_reply 関数を呼び出しています。 • 非応答型のアプリケーションから dc_mcf_reply 関数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 dc_mcf_reply 関数を呼び出せません。
DCMCFRTN_72001	-13001	入力元論理端末は，応答メッセージの送信をサポートしていません。
DCMCFRTN_72005	-13005	＜ action で DCMCFESI を設定した場合＞ sdataleng に 0 バイト，またはマイナス値を設定しています。
DCMCFRTN_72008	-13008	最終セグメントを送信する dc_mcf_reply 関数を呼び出したあとで，再び dc_mcf_reply 関数を呼び出しています。
		応答型のアプリケーションから dc_mcf_execap 関数を呼び出して応答型のアプリケーションを起動したあとで，dc_mcf_reply 関数を呼び出しています。
DCMCFRTN_72016	-13016	opcd に設定した値が間違っています。
		resv01，resv02，または resv03 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72026	-13026	action に設定したセグメント種別（DCMCFESI または DCMCFEMI）の値が間違っています。

リターン値	リターン値（数値）	意味
DCMCFRTN_72041	-13041	＜ action で DCMCFEMI を設定した場合＞ • sdataleng に 0 バイト，またはマイナス値を設定しています。 • action に DCMCFESI を設定した dc_mcf_reply 関数を呼び出さないで，メッセージの送信の終了を連絡しています。
DCMCFRTN_72045	-13045	継続問い合わせ応答型のアプリケーションはサポートしていません。
DCMCFRTN_72047	-13047	継続問い合わせ応答型のアプリケーションはサポートしていません。
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。

（凡例）

－：該当しません。

dc_mcf_resend － メッセージの再送（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_resend(DCLONG action, DCLONG commform, char *rtermnam,
                   char *resv01, DCLONG oseqid, DCLONG orgseq,
                   char *otermnam, char *resv02, char *resv03,
                   char *resv04, DCLONG opcd)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int  dc_mcf_resend(action, commform, rtermnam, resv01, oseqid,
                   orgseq, otermnam, resv02, resv03, resv04,
                   opcd)
DCLONG     action;
DCLONG     commform;
char       *rtermnam;
char       *resv01;
DCLONG     oseqid;
DCLONG     orgseq;
char       *otermnam;
char       *resv02;
char       *resv03;
char       *resv04;
DCLONG     opcd;
```

機能

以前に送信したメッセージを，再び送信します。再送するメッセージは，以前に送信したメッセージとは別の，新しいメッセージとして扱います。どのメッセージを再送するかは，次に示す送信済みメッセージの情報で選択できます。

- 出力先の論理端末名称
- メッセージ出力通番
- メッセージ種別（一般の一方送信，優先の一方送信）

対象としたメッセージが以前に送信されていない場合は，dc_mcf_resend 関数はリターン値 DCMCFRTN_NOMSG を返します。また，メッセージキュー（ディスクキュー）内に対象のメッセージがない場合も，リターン値 DCMCFRTN_NOMSG を返します。このため，使用するメッセージキューの種別ではディスクキューを指定するとともに，メッセージキューの大きさの定義は余裕を持った値を指定してください。

UAP で値を設定する引数

●action

再送するメッセージに出力通番を付け直すかどうか，一般か優先か，および最終出力通番のメッセージを再送するかどうかを，次の形式で設定します。

```
{DCMCFSEQ|DCMCFNSEQ} 〔|{DCMCFNORM|DCMCFPRIO}〕 〔|DCMCFLAST〕
```

DCMCFSEQ

再送するメッセージに出力通番を付け直す場合に設定します。

DCMCFNSEQ

再送するメッセージに出力通番を付け直さない場合に設定します。

DCMCFNORM

一般の一方送信メッセージとして再送する場合に設定します。

DCMCFPRIO

優先の一方送信メッセージとして再送する場合に設定します。

DCMCFLAST

最終出力通番を持つメッセージを再送する場合に設定します。この値を設定した場合は，orgseq に設定した値は無視されます。

●commform

一方送信を示す，DCMCFOUT を設定します。

●rtermnam

出力先の論理端末名称を設定します。論理端末名称は最大８バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

●resv01

NULL を設定します。

●oseqid

再送するメッセージを検索するキーとして，以前に送信したメッセージの送信種別を設定します。

DCMCFRID_NORM

一般の一方送信メッセージを対象とする場合に設定します。

DCMCFRID_PRIO

優先の一方送信メッセージを対象とする場合に設定します。

省略した場合は，DCMCFRID_NORM（一般の一方送信メッセージを対象）が設定されます。

DCMCFRID_PRIO を設定した場合は，otermnam に出力先の論理端末名称を設定できません。

●orgseq

再送するメッセージを検索するキーとして，以前に送信したメッセージの出力通番を設定します。action で DCMCFLAST を設定した場合は，ここに設定した値は無視されます。

●otermnam

再送するメッセージを検索するキーとして，以前に送信したメッセージの出力先の論理端末名称を設定します。論理端末名称は最大８バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

●resv02，resv03，resv04

NULL を設定します。

●opcd

DCNOFLAGS を設定します。

リターン値

リターン値	リターン値 (数値)	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_NOMSG	-11904	出力通番使用論理端末数（MCF マネジャ共通定義（mcfmcomn）の-n オプション）を省略，または 0 を指定しています。
		otermnam，oseqid，または orgseq に設定した値が間違っています。
		再送するメッセージは次に示す理由によって，再送できるメッセージの条件を満たしていません。 • 再送対象とするメッセージの送信時に出力通番を付けていません。 • 出力メッセージの割り当て先にメモリキューを使用しています（論理端末定義（mcftalcle -k）の quekind オペランドを省略，または memory を指定）。 • 論理端末が閉塞しているなどの要因によって，送信メッセージが送信済みになっていません。 • 送信済みのメッセージがディスクキューに保持されていません（保持メッセージ数はメッセージキューサービス定義の quegrp コマンドの-m オプションで指定します）。
		otermnam で指定した論理端末に割り当てられているメッセージキューは，システム開始時に障害が発生したため，メモリキューを代用して縮退運転をしています。
DCMCFRTN_BUF_SHORT	-11905	再送するメッセージのセグメントの長さが，UAP 共通定義（mcfmuap -e）で指定した値を超えています。
DCMCFRTN_71002	-12002	メッセージキューへの出力処理中に障害が発生しました。
		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。

リターン値	リターン値（数値）	意味
DCMCFRTN_71002	-12002	MCF が終了処理中のため，メッセージの再送を受け付けられません。
DCMCFRTN_71003	-12003	メッセージキューが満杯です。
DCMCFRTN_71004	-12004	メッセージキューから取り出したメッセージを格納するバッファ（作業領域）を，メモリ上に確保できませんでした。
DCMCFRTN_71108	-12108	メッセージを再送しようとしましたが，再送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。
DCMCFRTN_72000	-13000	＜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 dc_mcf_receive 関数を呼び出す前に，dc_mcf_resend 関数を呼び出しています。 • 非トランザクション属性の MHP から，dc_mcf_resend 関数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，dc_mcf_resend 関数を呼び出しています。
DCMCFRTN_72001	-13001	rtermnam または otermnam に設定した論理端末名称が間違っています。
		dc_mcf_resend 関数を呼び出せない論理端末を設定しています。
DCMCFRTN_72016	-13016	action に設定したメッセージ種別（DCMCFNORM または DCMCFPRIO）の値が間違っています。
		action に設定した値が間違っています。
		oseqid に設定した値が間違っています。
		opcd に設定した値が間違っています。
		resv01，resv02，resv03，または resv04 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72017	-13017	action に設定した出力通番の要否（DCMCFSEQ または DCMCFNSEQ）の値が間違っています。
DCMCFRTN_72024	-13024	commform 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。

（凡例）

－：該当しません。

注意事項

メッセージの再送時には，MCF マネジャ定義の UAP 共通定義（mcfmuap）の-e オプションと-l オプションの指定値に注意してください。

-e オプション

-e オプションでは，dc_mcf_resend 関数で使用する作業領域の大きさを指定します。再送するメッセージのセグメントがこの作業領域より大きい場合，dc_mcf_resend 関数はメッセージを再送しないで，リターン値 DCMCFRTN_BUF_SHORT を返します。このため，-e オプションでは，セグメントの最大長よりも大きな値を設定しておいてください。

-l オプション

-l オプションでは，出力通番に関して指定します。この内容によっては，メッセージキューファイル内に同じ出力通番を持つメッセージが同時に存在する場合があります。この場合は，どのメッセージを再送するか保証できません。

dc_mcf_send － 一方送信メッセージの送信（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_send(DCLONG action, DCLONG commform, char *termnam,
                 char *resv01, char *senddata, DCLONG sdataleng,
                 char *resv02, DCLONG opcd)
```

K&R 版 C の形式

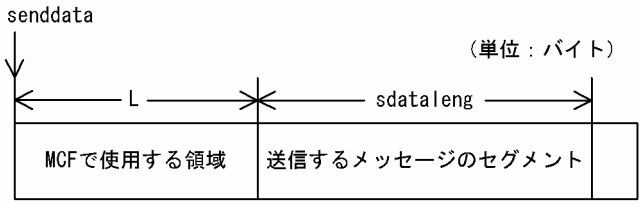
```
#include <dcmcf.h>
int  dc_mcf_send(action, commform, termnam, resv01, senddata,
                 sdataleng, resv02, opcd)
DCLONG     action;
DCLONG     commform;
char       *termnam;
char       *resv01;
char       *senddata;
DCLONG     sdataleng;
char       *resv02;
DCLONG     opcd;
```

機能

相手システムへ送る一方送信メッセージのうち，一つのセグメントを送信します。セグメントの数だけ dc_mcf_send 関数を呼び出すと，一つの論理メッセージを送信できます。

送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域の形式を次に示します。L は，バッファ形式 1 の場合は 8 バイト，バッファ形式 2 の場合は 4 バイトです。



UAP で値を設定する引数

●action

送信するセグメント，優先か一般か，出力通番を付けるかどうか，および使用するバッファ形式を次の形式で設定します。

```
{DCMCFESI|DCMCFEMI} 〔|{DCMCFNORM|DCMCFPRIO}〕
 〔|{DCMCFSEQ|DCMCFNSEQ}〕 〔|{DCMCFBUF1|DCMCFBUF2}〕
```

DCMCFESI

先頭セグメントまたは中間セグメントを送信する場合に設定します。

DCMCFEMI

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。

DCMCFNORM

一般の一方送信メッセージとして送信する場合に設定します。

DCMCFPRIO

優先の一方送信メッセージとして送信する場合に設定します。

DCMCFSEQ

出力通番を付ける場合に設定します。

DCMCFNSEQ

出力通番を付けない場合に設定します。

DCMCFBUF1

バッファ形式 1 を使用する場合に設定します。

DCMCFBUF2

バッファ形式 2 を使用する場合に設定します。

●commform

一方送信を示す，DCMCFOUT を設定します。

●termnam

出力先の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

●resv01

NULL またはヌル文字列を設定します。

●senddata

送信するセグメントの内容を設定した領域を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，必ず設定してください。

●sdataleng

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

●resv02

NULL またはヌル文字列を設定します。

●opcd

DCNOFLAGS を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71002	-12002	メッセージキューへの出力処理中に障害が発生しました。
		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。
		sdataleng 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，メッセージの送信を受け付けられません。
DCMCFRTN_71003	-12003	メッセージキューが満杯です。
DCMCFRTN_71004	-12004	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。
DCMCFRTN_71108	-12108	メッセージを送信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。
DCMCFRTN_72000	-13000	＜ 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 dc_mcf_receive 関数を呼び出す前に，dc_mcf_send 関数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，dc_mcf_send 関数を呼び出しています。
DCMCFRTN_72001	-13001	termnam に設定した論理端末名称が間違っています。
		termnam に設定した論理端末名称は，定義されていません。
		dc_mcf_send 関数を呼び出せない論理端末を設定しています。
DCMCFRTN_72005	-13005	＜ action で DCMCFESI を設定した場合＞ sdataleng に 0 バイト，またはマイナス値を設定しています。

リターン値	リターン値 (数値)	意味
DCMCFRTN_72016	-13016	action に設定したメッセージ種別（DCMCFNORM または DCMCFPRIO）の値が間違っています。
		action に設定した値が間違っています。
		opcd に設定した値が間違っています。
		resv01 または resv02 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72017	-13017	action に設定した出力通番の要否（DCMCFSEQ または DCMCFNSEQ）の値が間違っています。
DCMCFRTN_72024	-13024	commform に設定した値が間違っています。
DCMCFRTN_72026	-13026	action に設定したセグメント種別（DCMCFESI または DCMCFEMI）の値が間違っています。
DCMCFRTN_72041	-13041	＜action で DCMCFEMI を設定した場合＞ • sdataleng に 0 バイト，またはマイナス値を設定しています。 • action に DCMCFESI を設定した dc_mcf_send 関数を呼び出さないで，メッセージの送信の終了を連絡しています。
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。

（凡例）

－：該当しません。

dc_mcf_sendrecv － 同期型のメッセージの送受信（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int  dc_mcf_sendrecv(DCLONG action, DCLONG commform, char *termnam,
                     char *resv01, char *senddata,
                     DCLONG sdataleng, char *recvdata,
                     DCLONG *rdataleng, DCLONG inbufleng,
                     DCLONG *time, DCLONG watchtime)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int  dc_mcf_sendrecv(action, commform, termnam, resv01,
                     senddata, sdataleng, recvdata, rdataleng,
                     inbufleng, time, watchtime)
DCLONG     action;
DCLONG     commform;
char       *termnam;
char       *resv01;
char       *senddata;
DCLONG     sdataleng;
char       *recvdata;
DCLONG     *rdataleng;
DCLONG     inbufleng;
DCLONG     *time;
DCLONG     watchtime;
```

機能

同期型でメッセージを送信したあと，同期型でメッセージを受信します。

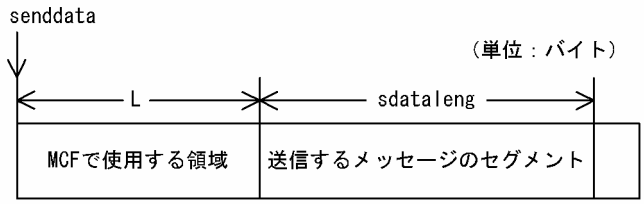
相手システムへ送るメッセージのうち，一つのセグメントを送信します。セグメントの数だけ dc_mcf_sendrecv 関数を呼び出すと，一つの論理メッセージを送信できます。メッセージの最終セグメントを送信すると，dc_mcf_sendrecv 関数は相手システムからの応答を待ちます。応答が届くと，そのメッセージの先頭セグメントを受信します。

中間セグメントまたは最終セグメントを受信する場合は，dc_mcf_recvsync 関数を呼び出してください。

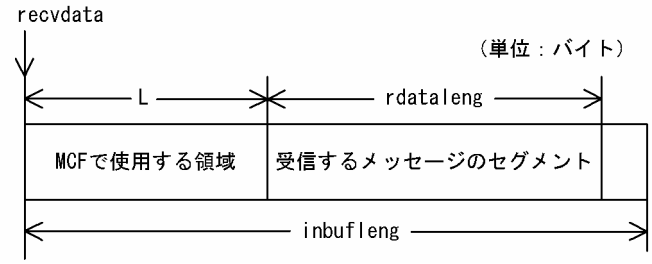
受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。また，送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域と受信する領域の形式をそれぞれ示します。L は，バッファ形式 1 の場合は 8 バイト，バッファ形式 2 の場合は 4 バイトです。

●セグメントを送信する領域



●セグメントを受信する領域



UAP で値を設定する引数

●action

送信するセグメント，および使用するバッファ形式を，次の形式で設定します。

```
{DCMCFESI|DCMCFEMI} 〔|{DCMCFBUF1|DCMCFBUF2}〕
```

DCMCFESI

先頭セグメントまたは中間セグメントを送信する場合に設定します。

DCMCFEMI

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。この値を設定して dc_mcf_sendrecv 関数を呼び出すと，相手システムからの応答を待ちます。

DCMCFBUF1

バッファ形式 1 を使用する場合に設定します。

DCMCFBUF2

バッファ形式 2 を使用する場合に設定します。

●commform

同期型のメッセージの送受信を示す，DCMCFIO を設定します。

●termnam

メッセージを出力して応答を入力する論理端末名称を設定します。論理端末名称は最大８バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

●resv01

NULL またはヌル文字列を設定します。

●senddata

送信するセグメントの内容を設定した領域を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，必ず設定してください。

●sdataleng

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

●recvdata

セグメントを受信する領域を設定します。

単一セグメントまたは最終セグメントを送信する dc_mcf_sendrecv 関数が終了すると，受信したメッセージの先頭セグメントが返されます。

処理終了後，recvdata には OpenTP1 から値が返ります。

●inbufleng

セグメントを受信する領域の長さを設定します。

●watchtime

dc_mcf_sendrecv 関数を呼び出してから終了するまでの最大時間を設定します。0 を設定した場合，MCF マネジャ定義の UAP 共通定義で指定した同期型送受信監視時間（mcfmuap -t sndrcvtim）が設定されます。負の値を設定した場合は，時間を監視しません。

OpenTP1 から値が返される引数

●recvdata

受信したメッセージの先頭セグメントの内容が返されます。

●rdataleng

受信したメッセージの先頭セグメントの長さが返されます。

●time

メッセージを受信した時刻が，1970 年 1 月 1 日 0 時 0 分 0 秒からの通算の秒数で返されます。

リターン値

リターン値	リターン値(数値)	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71002	-12002	メッセージキューへの入出力処理時に障害が発生しました。
		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。
		sdataleng 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，メッセージの送受信を受け付けられません。
DCMCFRTN_71003	-12003	メッセージキューが満杯です。
DCMCFRTN_71004	-12004	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。
DCMCFRTN_71108	-12108	メッセージを送信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。
DCMCFRTN_72000	-13000	先頭セグメントを受信する dc_mcf_receive 関数を呼び出す前に，dc_mcf_sendrecv 関数を呼び出しています。
DCMCFRTN_72001	-13001	termnam に設定した論理端末名称が間違っています。
		termnam に設定した論理端末名称は，定義されていません。
		dc_mcf_sendrecv 関数を呼び出せない論理端末を設定しています。
DCMCFRTN_72005	-13005	＜ action で DCMCFESI を設定した場合＞ sdataleng に 0 バイト，またはマイナス値を設定しています。
DCMCFRTN_72013	-13013	inbufleng の指定値を超えるセグメントを受信しました。inbufleng の指定値を超えた部分は切り捨てられました。
DCMCFRTN_72016	-13016	action に設定した値が間違っています。
		resv01 に設定した値が間違っています。
		引数に設定した値に間違いがあります。
DCMCFRTN_72024	-13024	commform に設定した値が間違っています。
DCMCFRTN_72026	-13026	action に設定したセグメント種別(DCMCFESI または DCMCFEMI)の値が間違っています。
DCMCFRTN_72036	-13036	inbufleng の指定値が不足しています。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。
DCMCFRTN_72041	-13041	＜ action で DCMCFEMI を設定した場合＞ • sdataleng に 0 バイト，またはマイナス値を設定しています。 • action に DCMCFESI を設定した dc_mcf_sendrecv 関数を呼び出さないで，メッセージの送信の終了を連絡しています。

リターン値	リターン値（数値）	意味
DCMCFRTN_73001	-14001	watchtime に 60 秒を加算した時間が経過しましたが，相手システムまたは MCF 通信プロセスからの応答がありません。
		出力キューからのメッセージの読み込み時に障害が発生しました。
		相手システムから受信打ち切りを受信しました。
		dc_mcf_sendrecv 関数を呼び出したあとで，UA 障害またはコネクション障害が発生しました。
DCMCFRTN_73002	-14002	メッセージを送信しようとしましたが，送信バッファまたは編集バッファを確保できませんでした。
		出力先の論理端末で MCF 通信プロセスの内部障害が発生しました。
DCMCFRTN_73008	-14008	論理端末が閉塞中，または MCF が終了処理中に，dc_mcf_sendrecv 関数を呼び出しました。
DCMCFRTN_73010	-14010	入力または出力メッセージ編集 UOC で障害が発生しました。
		メッセージの編集エラーが発生しました。
DCMCFRTN_73015	-14015	出力先の論理端末は，ほかの UAP で仕掛り中です。
DCMCFRTN_73018	-14018	watchtime 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。

（凡例）

－：該当しません。

dc_mcf_tactcn － コネクションの確立（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tactcn (DCLONG action, dcmcf_tactcnopt *cnopt,
                   char *proinf, DCLONG *resv02, char *resv03,
                   char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tactcn (action, cnopt, proinf, resv02, resv03, resv04)
DCLONG             action;
dcmcf_tactcnopt    *cnopt;
char               *proinf;
DCLONG             *resv02;
char               *resv03;
char               *resv04;
```

機能

コネクションを確立します。

なお，dc_mcf_tactcn 関数の正常終了は，コネクション確立要求を TP1/NET/User Agent 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。

dc_mcf_tactcn 関数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，dc_mcf_tlscn 関数を用いてコネクションの状態を確認してください。

UAP で値を設定する引数

●action

確立するコネクションの指定方法を次の形式で設定します。

```
{DCMCFLE | DCMCFCN}
```

DCMCFLE

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。

DCMCFCN

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。

●cnopt

この関数の対象となったコネクションの情報を，構造体 dcmcf_tactcnopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;          …MCF通信プロセス識別子
                char      resv01[4];      …予備領域
                char      idnam[9];       …論理端末名称，コネクションID
                char      resv02[7];      …予備領域
                char      resv03[112];    …予備領域
                char      scnnam[9];      …MCF使用領域
                char      resv04[7];      …予備領域
                char      yournam[9];     …MCF使用領域
                char      resv05[7];      …予備領域
                char      hostnam[143];   …MCF使用領域
                char      resv06[17];     …予備領域
                char      resv07[184];    …予備領域
                } dcmcf_tactcnopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  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 は最大 8 バイトの長さです。論理端末名称，またはコネクション ID の最後にはヌル文字を付けてください。

- resv02，resv03，scnnam，resv04，yournam，resv05，hostnam，resv06，resv07

  領域をヌル文字で埋めます。

●proinf，resv02，resv03，resv04

NULL を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tactcn 関数が受け付けられません。
DCMCFRTN_71002	-12002	MCF が終了処理中のため，dc_mcf_tactcn 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tactcn 関数の処理中にメモリ不足が発生しました。
DCMCFRTN_71005	-12005	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。
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71007	-12007	指定されたコネクション ID は登録されていません。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tactcn 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	コネクションが削除されているため，dc_mcf_tactcn 関数が受け付けられません。
DCMCFRTN_71014	-12014	TP1/NET/NCSB，または 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 のコネクショングループ名を指定しています。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
DCMCFRTN_72051	-13051	cnopt に NULL が設定されています。
DCMCFRTN_72052	-13052	proinf に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72060	-13060	action には DCMCFLE と DCMCFCN を同時に設定できません。
DCMCFRTN_72061	-13061	dcmcf_tactcnopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tactcnopt の resv01 がヌル文字で埋められていません。

リターン値	リターン値（数値）	意味
DCMCFRTN_72063	-13063	dcmcf_tactcnopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tactcnopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tactcnopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72066	-13066	dcmcf_tactcnopt の scnnam がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tactcnopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72068	-13068	dcmcf_tactcnopt の yournam がヌル文字で埋められていません。
DCMCFRTN_72069	-13069	dcmcf_tactcnopt の resv05 がヌル文字で埋められていません。
DCMCFRTN_72070	-13070	dcmcf_tactcnopt の hostnam がヌル文字で埋められていません。
DCMCFRTN_72071	-13071	dcmcf_tactcnopt の resv06 がヌル文字で埋められていません。
DCMCFRTN_72072	-13072	dcmcf_tactcnopt の resv07 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tactleopt の idnam に設定された文字数が 9 以上です。
DCMCFRTN_72074	-13074	dcmcf_tactcnopt の idnam に設定された文字列中に不正な文字があります。

dc_mcf_tactle － 論理端末の閉塞解除（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tactle (DCLONG action, dcmcf_tactleopt *leopt,
                   char *proinf, DCLONG *resv02,
                   char *resv03, char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tactle (action, leopt, proinf, resv02, resv03, resv04)
DCLONG             action;
dcmcf_tactleopt    *leopt;
char               *proinf;
DCLONG             *resv02;
char               *resv03;
char               *resv04;
```

機能

論理端末の閉塞を解除します。

なお，dc_mcf_tactle 関数の正常終了は，論理端末の閉塞解除要求を TP1/NET/User Agent 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。

dc_mcf_tactle 関数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，dc_mcf_tlsle 関数を用いて論理端末の状態を確認してください。

UAP で値を設定する引数

●action

閉塞解除する論理端末の指定方法を次の形式で設定します。

```
DCMCFLE
```

DCMCFLE

　閉塞解除する論理端末を論理端末名称で指定するときに設定します。

●leopt

この関数の対象となった論理端末の情報を，構造体 dcmcf_tactleopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;          …MCF通信プロセス識別子
                char      resv01[4];      …予備領域
                char      idnam[9];       …論理端末名称
                char      resv02[7];      …予備領域
                char      resv03[112];    …予備領域
                char      resv04[376];    …予備領域
                } dcmcf_tactleopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  閉塞解除する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

- resv02，resv03，resv04

  領域をヌル文字で埋めます。

●proinf，resv02，resv03，resv04

NULL を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tactle 関数が受け付けられません。
DCMCFRTN_71002	-12002	MCF が終了処理中のため，dc_mcf_tactle 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tactle 関数の処理中にメモリ不足が発生しました。
DCMCFRTN_71005	-12005	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。

リターン値	リターン値（数値）	意味
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tactle 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	論理端末が削除されているため，dc_mcf_tactle 関数が受け付けられません。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
		action に DCMCFLE が指定されていません。
DCMCFRTN_72051	-13051	leopt に NULL が設定されています。
DCMCFRTN_72052	-13052	proinf に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72061	-13061	dcmcf_tactleopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tactleopt の resv01 がヌル文字で埋められていません。
DCMCFRTN_72063	-13063	dcmcf_tactleopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tactleopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tactleopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tactleopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tactleopt の idnam に設定された文字数が 9 以上です。
DCMCFRTN_72074	-13074	dcmcf_tactleopt の idnam に設定された文字列中に不正な文字があります。

dc_mcf_tdctcn － コネクションの解放（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tdctcn (DCLONG action, dcmcf_tdctcnopt *cnopt,
                   char *proinf, DCLONG *resv02, char *resv03,
                   char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tdctcn (action, cnopt, proinf, resv02, resv03, resv04)
DCLONG               action;
dcmcf_tdctcnopt      *cnopt;
char                 *proinf;
DCLONG               *resv02;
char                 *resv03;
char                 *resv04;
```

機能

コネクションを解放します。

なお，dc_mcf_tdctcn 関数の正常終了は，コネクション解放要求を TP1/NET/User Agent 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。

dc_mcf_tdctcn 関数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，dc_mcf_tlscn 関数を用いてコネクションの状態を確認してください。

UAP で値を設定する引数

●action

解放するコネクションの指定方法を次の形式で設定します。

```
{DCMCFLE | DCMCFCN}  〔 | DCMCFFRC〕
```

DCMCFLE

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。

DCMCFCN

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。

DCMCFFRC

コネクションを強制的に解放するときに設定します。

●cnopt

この関数の対象となったコネクションの情報を，構造体 dcmcf_tdctcnopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;          …MCF通信プロセス識別子
                char      resv01[4];      …予備領域
                char      idnam[9];       …論理端末名称，コネクションID
                char      resv02[7];      …予備領域
                char      resv03[112];    …予備領域
                char      scnnam[9];      …MCF使用領域
                char      resv04[7];      …予備領域
                char      resv05[360];    …予備領域
                } dcmcf_tdctcnopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  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 は最大 8 バイトの長さです。論理端末名称，またはコネクション ID の最後にはヌル文字を付けてください。

- resv02，resv03，scnnam，resv04，resv05

  領域をヌル文字で埋めます。

●proinf，resv02，resv03，resv04

NULL を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tdctcn 関数が受け付けられません。
DCMCFRTN_71002	-12002	MCF が終了処理中のため，dc_mcf_tdctcn 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tdctcn 関数の処理中にメモリ不足が発生しました。
DCMCFRTN_71005	-12005	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。
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71007	-12007	指定されたコネクション ID は登録されていません。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tdctcn 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	コネクションが削除されているため，dc_mcf_tdctcn 関数が受け付けられません。
DCMCFRTN_71014	-12014	TP1/NET/NCSB，または 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 のコネクショングループ名を指定しています。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
DCMCFRTN_72051	-13051	cnopt に NULL が設定されています。
DCMCFRTN_72052	-13052	proinf に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72060	-13060	action には DCMCFLE と DCMCFCN を同時に設定できません。
DCMCFRTN_72061	-13061	dcmcf_tdctcnopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tdctcnopt の resv01 がヌル文字で埋められていません。

リターン値	リターン値（数値）	意味
DCMCFRTN_72063	-13063	dcmcf_tdctcnopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tdctcnopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tdctcnopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72066	-13066	dcmcf_tdctcnopt の scnnam がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tdctcnopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72069	-13069	dcmcf_tdctcnopt の resv05 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tdctcnopt の idnam に設定された文字数が 9 以上です。
DCMCFRTN_72074	-13074	dcmcf_tdctcnopt の idnam に設定された文字列中に不正な文字があります。

dc_mcf_tdctle － 論理端末の閉塞（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tdctle (DCLONG action, dcmcf_tdctleopt *leopt,
                   char *proinf, DCLONG *resv02,
                   char *resv03, char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tdctle (action, leopt, proinf, resv02, resv03, resv04)
DCLONG             action;
dcmcf_tdctleopt    *leopt;
char               *proinf;
DCLONG             *resv02;
char               *resv03;
char               *resv04;
```

機能

論理端末を閉塞します。

なお，dc_mcf_tdctle 関数の正常終了は，論理端末の閉塞要求を TP1/NET/User Agent 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。

dc_mcf_tdctle 関数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，dc_mcf_tlsle 関数を用いて論理端末の状態を確認してください。

UAP で値を設定する引数

●action

閉塞する論理端末の指定方法を次の形式で設定します。

```
DCMCFLE
```

DCMCFLE

閉塞する論理端末を論理端末名称で指定するときに設定します。

●leopt

この関数の対象となった論理端末の情報を，構造体 dcmcf_tdctleopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;          …MCF通信プロセス識別子
                char      resv01[4];      …予備領域
                char      idnam[9];       …論理端末名称
                char      resv02[7];      …予備領域
                char      resv03[112];    …予備領域
                char      resv04[376];    …予備領域
                } dcmcf_tdctleopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  閉塞する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

- resv02，resv03，resv04

  領域をヌル文字で埋めます。

●proinf，resv02，resv03，resv04

NULL を設定します。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tdctle 関数が受け付けられません。
DCMCFRTN_71002	-12002	MCF が終了処理中のため，dc_mcf_tdctle 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tdctle 関数の処理中にメモリ不足が発生しました。
DCMCFRTN_71005	-12005	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。

リターン値	リターン値（数値）	意味
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tdctle 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	論理端末が削除されているため，dc_mcf_tdctle 関数が受け付けられません。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
		action に DCMCFLE が指定されていません。
DCMCFRTN_72051	-13051	leopt に NULL が設定されています。
DCMCFRTN_72052	-13052	proinf に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72061	-13061	dcmcf_tdctleopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tdctleopt の resv01 がヌル文字で埋められていません。
DCMCFRTN_72063	-13063	dcmcf_tdctleopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tdctleopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tdctleopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tdctleopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tdctleopt の idnam に設定された文字数が 9 以上です。
DCMCFRTN_72074	-13074	dcmcf_tdctleopt の idnam に設定された文字列中に不正な文字があります。

dc_mcf_tlscn － コネクションの状態取得（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tlscn (DCLONG action, dcmcf_tlscnopt *cnopt,
                  char *resv01, DCLONG *resv02,
                  char *resv03, DCLONG *infcnt,
                  dcmcf_cninf *inf, char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tlscn (action, cnopt, resv01, resv02, resv03, infcnt,
                  inf, resv04)
DCLONG            action;
dcmcf_tlscnopt    *cnopt;
char              *resv01;
DCLONG            *resv02;
char              *resv03;
DCLONG            *infcnt;
dcmcf_cninf       *inf;
char              *resv04;
```

機能

コネクションの状態を取得します。

UAP で値を設定する引数

●action

状態を取得するコネクションの指定方法を次の形式で設定します。

```
{DCMCFLE | DCMCFCN}
```

DCMCFLE

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。

DCMCFCN

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。

●cnopt

この関数の対象となったコネクションの情報を，構造体 dcmcf_tlscnopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;        …MCF通信プロセス識別子
                char      resv01[4];    …予備領域
                char      idnam[9];     …論理端末名称，コネクションID
                char      resv02[7];    …予備領域
                char      resv03[112];  …予備領域
                char      resv04[376];  …予備領域
                } dcmcf_tlscnopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  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 は最大 8 バイトの長さです。論理端末名称，またはコネクション ID の最後にはヌル文字を付けてください。

- resv02，resv03，resv04

  領域をヌル文字で埋めます。

●resv01，resv02，resv03

NULL を設定します。

●infcnt

コネクション状態を格納する領域 dcmcf_cninf の個数として，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。

●inf

コネクション状態を格納する領域 dcmcf_cninf を設定します。

「構造体 dcmcf_cninf のサイズ×infcnt」バイト数分の領域が必要です。

●resv04

NULL を設定します。

OpenTP1 から値が返される引数

●infcnt

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。

●inf

この関数の対象となったコネクションの情報が，構造体 dcmcf_cninf で返されます。

構造体の形式を次に示します。

```
typedef struct {
                char      idnam[9];     …コネクションID
                char      resv01[7];    …予備領域
                char      pnam[4];      …プロトコル種別
                DCLONG    status;       …コネクション状態
                char      resv02[40];   …予備領域
                } dcmcf_cninf;
```

- idnam

  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。コネクション ID は最大 8 バイトの長さです。コネクション ID の最後にはヌル文字が付けられます。

- resv01

  領域をヌル文字で埋めます。

- pnam

  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。プロトコル種別の最後にはヌル文字が付けられます。

  UA△

  　OSAS/UA プロトコル

- status

  要求したコネクションの状態として，次の値が設定されます。

  DCMCF_CNST_ACT

  　確立状態

  DCMCF_CNST_ACT_B

  　確立処理中状態

  DCMCF_CNST_DCT

  　解放状態

DCMCF_CNST_DCT_B

解放処理中状態

- resv02

領域をヌル文字で埋めます。

リターン値

リターン値	リターン値（数値）	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tlscn 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tlscn 関数の処理中にメモリ不足が発生しました。
DCMCFRTN_71005	-12005	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。
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71007	-12007	指定されたコネクション ID は登録されていません。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tlscn 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	コネクションが削除されているため，dc_mcf_tlscn 関数が受け付けられません。
DCMCFRTN_71014	-12014	TP1/NET/NCSB，または 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 のコネクショングループ名を指定しています。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
DCMCFRTN_72051	-13051	cnopt に NULL が設定されています。
DCMCFRTN_72052	-13052	resv01 に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72056	-13056	infcnt に NULL が設定されています。
DCMCFRTN_72057	-13057	inf に NULL が設定されています。

リターン値	リターン値（数値）	意味
DCMCFRTN_72060	-13060	action には DCMCFLE と DCMCFCN を同時に設定できません。
DCMCFRTN_72061	-13061	dcmcf_tlscnopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tlscnopt の resv01 がヌル文字で埋められていません。
DCMCFRTN_72063	-13063	dcmcf_tlscnopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tlscnopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tlscnopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tlscnopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tlscnopt の idnam に設定された文字数が 9 以上です。
DCMCFRTN_72074	-13074	dcmcf_tlscnopt の idnam に設定された文字列中に不正な文字があります。
DCMCFRTN_72076	-13076	infcnt に 1 が設定されていません。

dc_mcf_tlsle － 論理端末の状態取得（C 言語）

形式

ANSI C，C++の形式

```
#include <dcmcf.h>
int dc_mcf_tlsle (DCLONG action, dcmcf_tlsleopt *leopt,
                  char *resv01, DCLONG *resv02,
                  char *resv03, DCLONG *infcnt,
                  dcmcf_leinf2 *inf, char *resv04)
```

K&R 版 C の形式

```
#include <dcmcf.h>
int dc_mcf_tlsle (action, leopt, resv01, resv02, resv03, infcnt,
                  inf, resv04)
DCLONG            action;
dcmcf_tlsleopt    *leopt;
char              *resv01;
DCLONG            *resv02;
char              *resv03;
DCLONG            *infcnt;
dcmcf_leinf2      *inf;
char              *resv04;
```

機能

論理端末の状態を取得します。

UAP で値を設定する引数

●action

状態を取得する論理端末の指定方法を次の形式で設定します。

```
DCMCFLE
```

DCMCFLE

　状態を取得する論理端末を論理端末名称で指定するときに設定します。

●leopt

この関数の対象となった論理端末の情報を，構造体 dcmcf_tlsleopt に設定します。

構造体の形式を次に示します。

```
typedef struct {
                DCLONG    mcfid;        …MCF通信プロセス識別子
                char      resv01[4];    …予備領域
```

```
char      idnam[9];     …論理端末名称
char      resv02[7];    …予備領域
char      resv03[112];  …予備領域
char      resv04[376];  …予備領域
} dcmcf_tlsleopt;
```

- mcfid

  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。

  注※

  MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

- resv01

  領域をヌル文字で埋めます。

- idnam

  状態を取得する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。論理端末名称の最後にはヌル文字を付けてください。

- resv02，resv03，resv04

  領域をヌル文字で埋めます。

●resv01，resv02，resv03

NULL を設定します。

●infcnt

論理端末の状態を格納する領域 dcmcf_leinf2 の個数として，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。

●inf

論理端末の状態を格納する領域 dcmcf_leinf2 を設定します。

「構造体 dcmcf_leinf2 のサイズ×infcnt」バイト数分の領域が必要です。

●resv04

NULL を設定します。

OpenTP1 から値が返される引数

●infcnt

この関数の対象となった論理端末の個数が返されます。

●inf

この関数の対象となった論理端末の情報が構造体 dcmcf_leinf2 で返されます。

構造体の形式を次に示します。

```
typedef struct {
                char      idnam[9];     …論理端末名称
                char      resv01[7];    …予備領域
                char      resv02[4];    …予備領域
                DCLONG    status;       …論理端末状態
                char      resv03[40];   …予備領域
                } dcmcf_leinf2;
```

- idnam

  要求した論理端末の名称が設定されます。論理端末名称は最大８バイトの長さです。論理端末名称の最後にはヌル文字が付けられます。

- resv01，resv02

  領域をヌル文字で埋めます。

- status

  要求した論理端末の状態として，次の値が設定されます。

  DCMCF_LEST_ACT

  　閉塞解除状態

  DCMCF_LEST_DCT

  　閉塞状態

- resv03

  領域をヌル文字で埋めます。

リターン値

リターン値	リターン値(数値)	意味
DCMCFRTN_00000	0	正常に終了しました。
DCMCFRTN_71001	-12001	MCF が開始処理中のため，dc_mcf_tlsle 関数が受け付けられません。
DCMCFRTN_71004	-12004	dc_mcf_tlsle 関数の処理中にメモリ不足が発生しました。

リターン値	リターン値（数値）	意味
DCMCFRTN_71005	-12005	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。
DCMCFRTN_71006	-12006	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。
DCMCFRTN_71008	-12008	指定された論理端末名称は登録されていません。
DCMCFRTN_71009	-12009	dc_mcf_tlsle 関数が，該当する MCF 通信プロセスではサポートされていません。
DCMCFRTN_71010	-12010	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。
DCMCFRTN_71011	-12011	論理端末が削除されているため，dc_mcf_tlsle 関数が受け付けられません。
DCMCFRTN_72050	-13050	action に未サポートのフラグを設定しています。
		action に DCMCFLE が指定されていません。
DCMCFRTN_72051	-13051	leopt に NULL が設定されています。
DCMCFRTN_72052	-13052	resv01 に NULL が設定されていません。
DCMCFRTN_72053	-13053	resv02 に NULL が設定されていません。
DCMCFRTN_72054	-13054	resv03 に NULL が設定されていません。
DCMCFRTN_72055	-13055	resv04 に NULL が設定されていません。
DCMCFRTN_72056	-13056	infcnt に NULL が設定されています。
DCMCFRTN_72057	-13057	inf に NULL が設定されています。
DCMCFRTN_72061	-13061	dcmcf_tlsleopt の mcfid に 0 未満または 240 以上の値が設定されています。
DCMCFRTN_72062	-13062	dcmcf_tlsleopt の resv01 がヌル文字で埋められていません。
DCMCFRTN_72063	-13063	dcmcf_tlsleopt の idnam の先頭がヌル文字です。
DCMCFRTN_72064	-13064	dcmcf_tlsleopt の resv02 がヌル文字で埋められていません。
DCMCFRTN_72065	-13065	dcmcf_tlsleopt の resv03 がヌル文字で埋められていません。
DCMCFRTN_72067	-13067	dcmcf_tlsleopt の resv04 がヌル文字で埋められていません。
DCMCFRTN_72073	-13073	dcmcf_tlsleopt の idnam に設定された文字数が 9 以上です。

リターン値	リターン値（数値）	意味
DCMCFRTN_72074	-13074	dcmcf_tlsleopt の idnam に設定された文字列中に不正な文字があります。
DCMCFRTN_72076	-13076	infcnt に 1 が設定されていません。

4 COBOL-UAP 作成用プログラムインタフェース

この章では，TP1/NET/User Agent で使用できる，COBOL-UAP 作成用プログラムインタフェースについて説明します。

COBOL-UAP 作成用プログラムインタフェースの一覧

TP1/NET/User Agent で使用する COBOL-UAP 作成用プログラムインタフェースについて，COBOL 言語，およびデータ操作言語に分けて説明します。

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

COBOL 言語のプログラムインタフェース

COBOL 言語で UAP を作成する場合，OpenTP1 システムの関数に対応しているプログラムを，CALL 文で呼び出して UAP を作成します。

COBOL 言語のプログラムインタフェースの一覧を，次の表に示します。

表 4-1　COBOL 言語のプログラムインタフェースの一覧

プログラム名	データ名 A に設定する要求コード	機能
CBLDCMCF	'RECEIVE△'	メッセージの受信
	'RECVSYNC'	同期型の応答メッセージの受信
	'REPLY△△△'	応答メッセージの送信
	'RESEND△△'	メッセージの再送
	'SEND△△△△'	一方送信メッセージの送信
	'SENDRECV'	同期型のメッセージの送受信
	'TACTCN△△'	コネクションの確立
	'TACTLE△△'	論理端末の閉塞解除
	'TDCTCN△△'	コネクションの解放
	'TDCTLE△△'	論理端末の閉塞
	'TLSCN△△△'	コネクションの状態取得
	'TLSLE△△△'	論理端末の状態取得

その他のプログラムについては，マニュアル「OpenTP1 プログラム作成リファレンス COBOL 言語編」を参照してください。

データ操作言語のプログラムインタフェース

データ操作言語（DML）を使用した，通信文について説明します。データ操作言語の形式の詳細については，マニュアル「OpenTP1 プログラム作成リファレンス COBOL 言語編」を参照してください。

データ操作言語のプログラムインタフェースの一覧を，次の表に示します。

表 4-2　データ操作言語のプログラムインタフェースの一覧

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	メッセージの受信	「表 4-4　RECEIVE 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応」，および「表 4-5　SEND 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応」を参照してください。
	SEND	メッセージの送信	

注

dc_mcf_resend（メッセージの再送）に対応するデータ操作言語のインタフェースはありません。

その他の通信文については，マニュアル「OpenTP1 プログラム作成リファレンス COBOL 言語編」を参照してください。

通信記述項について

TP1/NET/User Agent のメッセージ送受信の通信文で，通信記述項に指定できる句の指定要否を，次の表に示します。

表 4-3　通信記述項に指定できる句の指定要否

データ名を指定する句	データ領域の値の設定元						
	1.	2.	3.	4.	5.	6.	7.
STATUS KEY	B	B	B	B	B	B	B
SYMBOLIC TERMINAL	B	U_1※	U_1	U_1	−	−	U_1
MESSAGE DATE	B	B	−	−	−	−	−
MESSAGE TIME	B	B	−	−	−	−	−
SYNCHRONOUS MODE	U_2	U_2	U_2	U_2	U_2	U_2	U_1
SWITCHING MODE	−	−	U_2	U_2	−	−	−
DETAIL MODE	−	−	U_2	U_2	U_2	U_2	−
WAITING TIME	−	−	−	−	−	−	U_2

（凡例）

1.：先頭セグメントの非同期受信（RECEIVE）

2.：中間，最終セグメントの非同期受信（RECEIVE）

3.：一方送信メッセージの先頭，中間セグメントの非同期送信（SEND）

4.：一方送信メッセージの単一，最終セグメントの非同期送信（SEND）

5.：応答メッセージの先頭，中間セグメントの非同期送信（SEND）

6.：応答メッセージの単一，最終セグメントの非同期送信（SEND）

7.：単一セグメントの同期送受信（SEND）

B：OpenTP1 から値が返されます。省略できます。

U_1：UAP で値を設定します。省略できません。

U_2：UAP で値を設定します。省略できます。

－：該当しません。設定しても無効です。

注※

先頭メッセージ受信時の RECEIVE 文と同一の CD 句を用いた場合は省略できます。

データコミュニケーション機能と C 言語のライブラリ関数の対応

RECEIVE 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応を，次の表に示します。

表 4-4　RECEIVE 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応

FOR 句		SYNCHRONOUS MODE 句		対応する C 言語のライブラリ関数
INPUT	I-O	SYNC，または'1'	ASYNC，'0'，'△'，または省略	
○	－	－	○	dc_mcf_receive
－	○	－	○	
－	○	○	－	dc_mcf_recvsync※

（凡例）

○：指定あり

－：指定なし

注※

TP1/NET/User Agent のデータ操作言語ではサポートしていない関数です。

SEND 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応を，次の表に示します。

表 4-5　SEND 文（データコミュニケーション機能）の指定と C 言語のライブラリ関数との対応

FOR 句		SYNCHRONOUS MODE 句		BEFORE 句	対応する C 言語のライブラリ関数
OUTPUT	I-O	SYNC，または'1'	ASYNC，'0'，'△'，または省略		
○	－	－	○	－	dc_mcf_send
－	○	－	○	－	dc_mcf_reply
－	○	○	－	－	dc_mcf_sendsync※

FOR 句		SYNCHRONOUS MODE 句		BEFORE 句	対応する C 言語のライブラリ関数
OUTPUT	I-O	SYNC，または'1'	ASYNC，'0'，'△'，または省略		
－	○	○	－	○	dc_mcf_sendrecv

（凡例）

○：指定あり

－：指定なし

注※

TP1/NET/User Agent ではサポートしていない関数です。

CBLDCMCF('RECEIVE ') － メッセージの受信（COBOL 言語）

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)  VALUE SPACE.
   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Y  PIC X(n).
```

機能

論理端末に届いたメッセージのうち，一つのセグメントを受信します。セグメントの数だけCBLDCMCF('RECEIVE△')を呼び出すと，一つの論理メッセージを受信できます。

受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

CBLDCMCF('RECEIVE△')で受信できるメッセージの種類を次に示します。

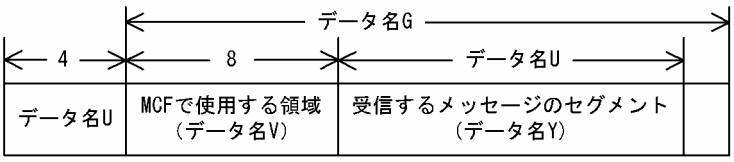
- 相手システムから送信されたメッセージ

- MCF イベント
- アプリケーション起動で渡されたメッセージ

セグメントを受信する領域（一意名 3 で示す領域）の形式を次に示します。

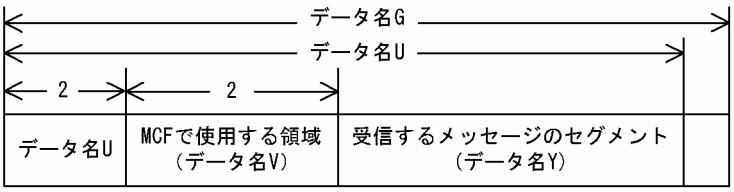
●バッファ形式 1 の場合

（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「VALUE 'RECEIVE△'」を設定します。

●データ名 C

受信するセグメントを設定します。次のどちらかを設定します。

VALUE 'FRST'

先頭セグメントを受信する場合や，論理メッセージが単一セグメントの場合に設定します。

VALUE 'SEG△'

中間セグメントまたは最終セグメントを受信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 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

使用するバッファ形式を設定します。

VALUE '1'

バッファ形式 1 を使用する場合に設定します。

VALUE '2'

バッファ形式 2 を使用する場合に設定します。

空白

省略されたものとして，「VALUE '1'」（バッファ形式 1）が設定されます。

●データ名 N

MCF で使用する領域です。

●データ名 O

空白を設定します。

●データ名 P

中間セグメントまたは最終セグメントを受信する場合は，入力元の論理端末名称を設定します。先頭セグメントの受信時に返された論理端末名称を設定してください。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

先頭セグメントまたは単一セグメントの受信処理終了後，データ名 P には OpenTP1 から値が返されます。

●データ名 Q

MCF で使用する領域です。

●データ名 R

空白を設定します。

●データ名 T

MCF で使用する領域です。

●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 E

メッセージを受信した日付が YYYYMMDD（YYYY：西暦年　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

先頭セグメントまたは単一セグメントを受信する場合，入力元の論理端末名称が返されます。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろが空白で埋められます。

中間セグメントまたは最終セグメントを受信する場合，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 Y

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。中間セグメントまたは最終セグメントを受信する場合は，データ名 C に「VALUE 'SEG△'」を設定して CBLDCMCF('RECEIVE△')を呼び出してください。
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 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	メッセージキューからの入力処理中に障害が発生しました。

ステータスコード	意味
71002	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，中間セグメントまたは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「VALUE '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	データ名 G の指定値を超えるセグメントを受信しました。データ名 G の指定値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	データ名 G の指定値が不足しています。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') − 同期型のメッセージの受信（COBOL 言語）

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 'RECVSYNC'.
   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.
   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)  VALUE SPACE.
   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Y  PIC X(n).
```

機能

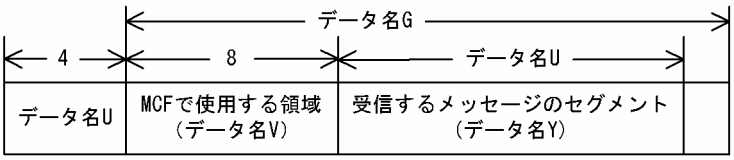
相手システムから届いた同期型の受信メッセージのうち，先頭セグメント以外の後続する一つのセグメントを受信します。先頭セグメントを受信する CBLDBMCF('SENDRECV')のあとに，後続するセグメントの数だけ CBLDCMCF('RECVSYNC')を呼び出すと，一つの論理メッセージを受信できます。

受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを受信する領域（一意名 3 で示す領域）の形式を次に示します。

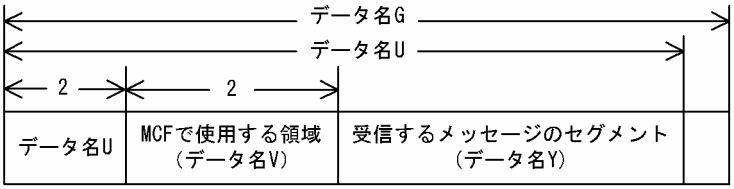
●バッファ形式１の場合

（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

同期型のメッセージの受信を示す要求コード「VALUE 'RECVSYNC'」を設定します。

●データ名 C

中間セグメントまたは最終セグメントの受信を示す「VALUE 'SEG△'」を設定します。

●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

使用するバッファ形式を設定します。

VALUE '1'

バッファ形式 1 を使用する場合に設定します。

VALUE '2'

バッファ形式 2 を使用する場合に設定します。

空白

省略されたものとして，「VALUE '1'」（バッファ形式 1）が設定されます。

●データ名 N

MCF で使用する領域です。

●データ名 O

空白を設定します。

●データ名 P

入力元の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●データ名 Q，データ名 R

空白を設定します。

●データ名 T

MCF で使用する領域です。

●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 E

メッセージを受信した日付が YYYYMMDD（YYYY：西暦年　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 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 Y

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。
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('RECVSYNC')を呼び出しています。直前に呼び出した CBLDCMCF('RECVSYNC')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，再び次のセグメントを受信する CBLDCMCF('RECVSYNC')を呼び出した場合は，ステータスコード 72000 が返されます。
71108	メッセージ受信に必要な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	先頭セグメントを受信する CBLDCMCF('RECEIVE△')を呼び出す前に，CBLDCMCF('RECVSYNC')を呼び出しています。
	ステータスコード 71001 が返されたあとに，再び CBLDCMCF('RECVSYNC')を呼び出しています。
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。
	データ名 P に設定した論理端末名称は，定義されていません。
	CBLDCMCF('RECVSYNC')を呼び出せない論理端末を設定しています。
72013	データ名 G の指定値を超えるセグメントを受信しました。データ名 G の指定値を超えた部分は切り捨てられました。
72016	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 Q 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	データ名 G の指定値が不足しています。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。

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。

CBLDCMCF('REPLY ') － 応答メッセージの送信（COBOL 言語）

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

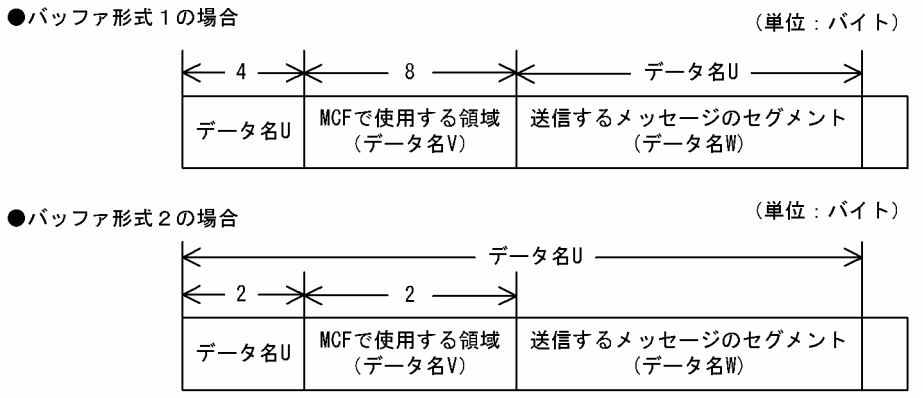
```
01 一意名1.
   02 データ名A  PIC X(8)  VALUE 'REPLY   '.
   02 データ名B  PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C  PIC X(4)  VALUE SPACE.
   02 データ名D  PIC X(4)  VALUE SPACE.
   02 データ名E  PIC 9(8).
   02 データ名F  PIC 9(8).
   02 データ名G  PIC 9(9)  COMP VALUE ZERO.
   02 データ名H  PIC X(4).
   02 データ名I  PIC X(4)  VALUE SPACE.
   02 データ名J  PIC X(4)  VALUE SPACE.
   02 データ名K  PIC X(4).
   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

問い合わせをしたシステムへ送る応答メッセージのうち，一つのセグメントを送信します。セグメントの数だけ CBLDCMCF('REPLY△△△')を呼び出すと，一つの論理メッセージを送信できます。

送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域（一意名 3 で示す領域）の形式を示します。



UAP で値を設定するデータ領域

●データ名 A

応答メッセージの送信を示す要求コード「VALUE 'REPLY△△△'」を設定します。

●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

送信するセグメントを設定します。次のどちらかを設定します。

VALUE 'ESI△'

先頭セグメントまたは中間セグメントを送信する場合に設定します。

VALUE 'EMI△'

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。

●データ名 I，データ名 J

空白を設定します。

●データ名 K

出力通番を付けるかどうかを設定します。

VALUE 'SEQ△'

出力通番を付ける場合に設定します。

ただし，非応答型のアプリケーションの場合は，VALUE 'SEQ△'を指定しても，MCF は応答メッセージに出力通番を付けません。

VALUE 'NSEQ'

出力通番を付けない場合に設定します。

空白

省略されたものとして，「VALUE 'NSEQ'」（出力通番は付けない）が設定されます。

●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。

VALUE '1'

バッファ形式 1 を使用する場合に設定します。

VALUE '2'

バッファ形式 2 を使用する場合に設定します。

空白

省略されたものとして，「VALUE '1'」（バッファ形式 1）が設定されます。

●データ名 N

MCF で使用する領域です。

●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 T

MCF で使用する領域です。

●データ名 U

【バッファ形式 1 の場合】 PIC 9(9)

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

【バッファ形式 2 の場合】 PIC 9(4)

送信するセグメントの長さ＋ 4 を設定します。先頭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で，セグメントの内容がないときは，4 を設定してください。

●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 で使用する領域です。

●データ名 W

送信するセグメントの内容を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，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出力処理中に障害が発生しました。
	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。
	バッファ形式 1 の場合はデータ名 U に 32000 バイトを超える値を設定しています。バッファ形式 2 の場合はデータ名 U に 32004 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，メッセージの送信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。
71108	メッセージを送信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＞ • 先頭セグメントを受信する CBLDCMCF('RECEIVE△')を呼び出す前に，CBLDCMCF('REPLY△△△')を呼び出しています。 • 非応答型のアプリケーションから CBLDCMCF('REPLY△△△')を呼び出しています。

ステータスコード	意味
72000	＜SPP の実行でリターンした場合＞ SPP では CBLDCMCF('REPLY△△△')を呼び出せません。
72001	入力元論理端末は，応答メッセージの送信をサポートしていません。
72005	＜データ名 H で VALUE 'ESI△'を設定した場合＞ バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。
72008	最終セグメントを送信する CBLDCMCF('REPLY△△△')を呼び出したあとで，再び CBLDCMCF('REPLY△△△')を呼び出しています。
	応答型のアプリケーションから CBLDCMCF('EXECAP△△')を呼び出して応答型のアプリケーションを起動したあとで，CBLDCMCF('REPLY△△△')を呼び出しています。
72016	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。
72017	データ名 K に設定した値が間違っています。
72019	データ名 M6 に設定した値が間違っています。
72020	データ名 I に設定した値が間違っています。
72026	データ名 H 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	＜データ名 H で VALUE 'EMI△'を設定した場合＞ • バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。 • データ名 H に VALUE 'ESI△'を設定した CBLDCMCF('REPLY△△△')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。
72045	継続問い合わせ応答型のアプリケーションはサポートしていません。
72047	継続問い合わせ応答型のアプリケーションはサポートしていません。
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。

CBLDCMCF('RESEND ') － メッセージの再送（COBOL 言語）

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 'RESEND  '.
   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).
   02 データ名K  PIC X(4).
   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)  VALUE SPACE.
   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 'OUT '.
   02 データ名P  PIC X(8).
   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 X(8).
   02 データ名U  PIC X(4).
   02 データ名V  PIC 9(9)  COMP.
   02 データ名W  PIC X(4).
   02 データ名X  PIC X(12) VALUE LOW-VALUE.
```

機能

以前に送信したメッセージを，再び送信します。再送するメッセージは，以前に送信したメッセージとは別の，新しいメッセージとして扱います。どのメッセージを再送するかは，次に示す送信済みメッセージの情報で選択できます。

- 出力先の論理端末名称

- メッセージ出力通番
- メッセージ種別（一般の一方送信，優先の一方送信）

対象としたメッセージが以前に送信されていない場合は，CBLDCMCF('RESEND△△')はステータスコード 70904 を返します。また，メッセージキュー（ディスクキュー）内に対象のメッセージがない場合も，ステータスコード 70904 を返します。このため，使用するメッセージキューの種別ではディスクキューを指定するとともに，メッセージキューの大きさの定義では余裕を持った値を指定してください。

UAP で値を設定するデータ領域

●データ名 A

メッセージの再送を示す要求コード「VALUE 'RESEND△△'」を設定します。

●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

空白を設定します。

●データ名 J

一般として再送するか優先として再送するかを設定します。

VALUE 'NORM'

一般の一方送信メッセージとして再送する場合に設定します。

VALUE 'PRIO'

優先の一方送信メッセージとして再送する場合に設定します。

空白

省略されたものとして，「VALUE 'NORM'」（一般の一方送信メッセージとして再送）が設定されます。

●データ名 K

再送するメッセージに出力通番を付け直すかどうかを設定します。

VALUE 'SEQ△'

再送するメッセージに出力通番を付け直す場合に設定します。

VALUE 'NSEQ'

再送するメッセージに出力通番を付け直さない場合に設定します。

空白

省略されたものとして，「VALUE 'NSEQ'」（出力通番を付け直さない）が設定されます。

●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，データ名 M7

空白を設定します。

●データ名 N

MCF で使用する領域です。

●データ名 O

一方送信を示す「VALUE 'OUT△'」を設定します。

●データ名 P

出力先の論理端末名称を設定します。論理端末名称は最大８バイトの長さです。８バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●データ名 Q，データ名 R

空白を設定します。

●データ名 S

MCF で使用する領域です。

●データ名 T

再送するメッセージを検索するキーとして，以前に送信したメッセージの出力先の論理端末名称を設定します。論理端末名称は最大８バイトの長さです。８バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●データ名 U

再送するメッセージを検索するキーとして，以前に送信したメッセージの送信種別を設定します。

VALUE 'NORM'

一般の一方送信メッセージを対象とする場合に設定します。

VALUE 'PRIO'

優先の一方送信メッセージを対象とする場合に設定します。

空白

省略されたものとして，「VALUE 'NORM'」（一般の一方送信メッセージを対象）が設定されます。

「VALUE 'PRIO'」を設定した場合は，データ名 T に出力先の論理端末名称を設定できません。

●データ名 V

再送するメッセージを検索するキーとして，以前に送信したメッセージの出力通番を設定します。データ名 W に「VALUE 'LAST'」を設定した場合は，ここに設定した値は無効となります。

●データ名 W

最終出力通番を持つメッセージを再送するかどうかを設定します。

VALUE 'LAST'

最終出力通番を持つメッセージを再送する場合に設定します。

この値を設定した場合は，データ名 V に設定した値は無効となります。

空白

データ名 V で設定した出力通番を持つメッセージを再送する場合に設定します。

●データ名 X

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。

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。
70904	出力通番使用論理端末数（MCF マネジャ共通定義（mcfmcomn）の-n オプション）を省略，または 0 を指定しています。
	データ名 T，データ名 U，またはデータ名 V に設定した値が間違っています。
	再送するメッセージは次に示す理由により，再送できるメッセージの条件を満たしていません。 • 再送対象とするメッセージの送信時に出力通番を付けていません。 • 出力メッセージの割り当て先にメモリキューを使用しています（論理端末定義（mcftalcle -k）の quekind オペランドを省略，または memory を指定）。 • 論理端末が閉塞しているなどの要因によって，送信メッセージが送信済みになっていません。 • 送信済みのメッセージがディスクキューに保持されていません（保持メッセージ数はメッセージキューサービス定義の quegrp コマンドの-m オプションで指定します）。

ステータスコード	意味
70904	データ名 T で指定した論理端末に割り当てられているメッセージキューは，システム開始時に障害が発生したため，メモリキューを代用して縮退運転をしています。
70905	再送するメッセージのセグメントの長さが，UAP 共通定義（mcfmuap -e）で指定した値を超えています。
71002	メッセージキューへの出力処理中に障害が発生しました。
	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。
	MCF が終了処理中のため，メッセージの再送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	メッセージキューから取り出したメッセージを格納するバッファ（作業領域）を，メモリ上に確保できませんでした。
71108	メッセージを再送しようとしましたが，再送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＞ • 先頭セグメントを受信する CBLDCMCF('RECEIVE△')を呼び出す前に，CBLDCMCF('RESEND△△')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('RESEND△△')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('RESEND△△')を呼び出しています。
72001	データ名 P またはデータ名 T に設定した論理端末名称が間違っています。
	データ名 P に設定した論理端末名称は，定義されていません。
	CBLDCMCF('RESEND△△')を呼び出せない論理端末を設定しています。
72016	データ名 J に設定した値が間違っています。
	データ名 M4 に設定した値が間違っています。
	データ名 U に設定した値が間違っています。
	データ名 N，データ名 S，またはデータ名 X に設定した値が間違っています。
72017	データ名 K に設定した値が間違っています。
72019	データ名 M6 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。
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

メッセージの再送時には，MCF マネジャ定義の UAP 共通定義（mcfmuap）の-e オプションと-l オプションの指定値に注意してください。

-e オプション

-e オプションでは，CBLDCMCF('RESEND△△')で使用する作業領域の大きさを指定します。再送するメッセージのセグメントがこの作業領域より大きい場合，CBLDCMCF('RESEND△△')はメッセージを再送しないで，ステータスコード 70905 を返します。このため，-e オプションでは，セグメントの最大長よりも大きな値を設定しておいてください。

-l オプション

-l オプションでは，出力通番に関して指定します。この内容によっては，メッセージキューファイル内に同じ出力通番を持つメッセージが同時に存在する場合があります。この場合は，どのメッセージを再送するか保証できません。

CBLDCMCF('SEND ') － 一方送信メッセージの送信（COBOL 言語）

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 'SEND    '.
   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).
   02 データ名K  PIC X(4).
   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 'OUT '.
   02 データ名P  PIC X(8).
   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

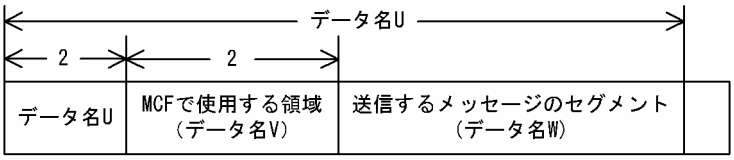
相手システムへ送る一方送信メッセージのうち，一つのセグメントを送信します。セグメントの数だけCBLDCMCF('SEND△△△△')を呼び出すと，一つの論理メッセージを送信できます。

送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域（一意名 3 で示す領域）の形式を示します。

●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

一方送信メッセージの送信を示す要求コード「VALUE 'SEND△△△△'」を設定します。

●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

送信するセグメントを設定します。次のどちらかを設定します。

VALUE 'ESI△'

先頭セグメントまたは中間セグメントを送信する場合に設定します。

VALUE 'EMI△'

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。

●データ名 I

空白を設定します。

●データ名 J

一般として送信するか優先として送信するかを設定します。

VALUE 'NORM'

一般の一方送信メッセージとして送信する場合に設定します。

VALUE 'PRIO'

優先の一方送信メッセージとして送信する場合に設定します。

空白

省略されたものとして，VALUE 'NORM'（一般の一方送信メッセージとして送信）が設定されます。

●データ名 K

出力通番を付けるかどうかを設定します。

VALUE 'SEQ△'

出力通番を付ける場合に設定します。

VALUE 'NSEQ'

出力通番を付けない場合に設定します。

空白

省略されたものとして，VALUE 'NSEQ'（出力通番を付けない）が設定されます。

●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。

VALUE '1'

バッファ形式 1 を使用する場合に設定します。

VALUE '2'

バッファ形式 2 を使用する場合に設定します。

空白

省略されたものとして，「VALUE '1'」（バッファ形式 1）が設定されます。

●データ名 N

MCF で使用する領域です。

●データ名 O

一方送信を示す「VALUE 'OUT△'」を設定します。

●データ名 P

出力先の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●データ名 Q，データ名 R

空白を設定します。

●データ名 T

MCF で使用する領域です。

●データ名 U

【バッファ形式 1 の場合】 PIC 9(9)

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

【バッファ形式 2 の場合】 PIC 9(4)

送信するセグメントの長さ＋ 4 を設定します。先頭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で，セグメントの内容がないときは，4 を設定してください。

●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 で使用する領域です。

●データ名 W

送信するセグメントの内容を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，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出力処理中に障害が発生しました。

ステータスコード	意味
71002	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。
	バッファ形式 1 の場合はデータ名 U に 32000 バイトを超える値を設定しています。バッファ形式 2 の場合はデータ名 U に 32004 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，メッセージの送信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。
71108	メッセージを送信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＞ 先頭セグメントを受信する CBLDCMCF('RECEIVE△')を呼び出す前に，CBLDCMCF('SEND△△△△')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('SEND△△△△')を呼び出しています。
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。
	データ名 P に設定した論理端末名称は，定義されていません。
	CBLDCMCF('SEND△△△△')を呼び出せない論理端末を設定しています。
72005	＜データ名 H で VALUE 'ESI△'を設定した場合＞ バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。
72016	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 J に設定した値が間違っています。
	データ名 M1 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。
72017	データ名 K に設定した値が間違っています。
72019	データ名 M6 に設定した値が間違っています。
72020	データ名 I 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。
72026	データ名 H 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	＜データ名 H で VALUE 'EMI△'を設定した場合＞ • バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。

ステータスコード	意味
72041	• データ名 H に VALUE 'ESI△'を設定した CBLDCMCF('SEND△△△△')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。
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('SENDRECV') － 同期型のメッセージの送受信（COBOL 言語）

形式

PROCEDURE DIVISION の指定

```
CALL  'CBLDCMCF'    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 'SENDRECV'.
   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).
   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※.
   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 'IO  '.
   02 データ名P  PIC X(8).
   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).
01 一意名4.
   02 データ名X  PIC 9(x)  COMP.
   02 データ名Y1 PIC X(x)  VALUE SPACE.
   02 データ名Y2 PIC X(1).
   02 データ名Z  PIC X(n).
```

注※

負の値を設定する場合，「PIC S9(9) COMP」としてください。

機能

同期型でメッセージを送信したあと，同期型でメッセージを受信します。

相手システムへ送るメッセージのうち，一つのセグメントを送信します。セグメントの数だけCBLDCMCF('SENDRECV')を呼び出すと，一つの論理メッセージを送信できます。

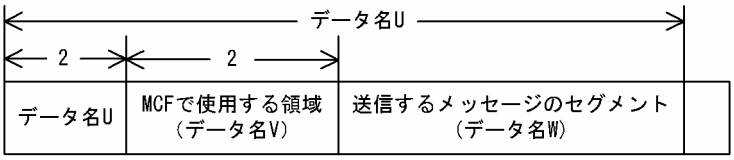
メッセージの最終セグメントを送信すると，CBLDCMCF('SENDRECV')は相手システムからの応答を待ちます。応答が届くと，そのメッセージの先頭セグメントを受信します。中間セグメントまたは最終セグメントを受信する場合は，CBLDCMCF('RECVSYNC')を呼び出してください。

受信できるメッセージの一つのセグメントの最大長は，32000 バイトです。また，送信できるメッセージの一つのセグメントの最大長は，32000 バイトです。

セグメントを送信する領域（一意名 3 で示す領域）の形式を次に示します。

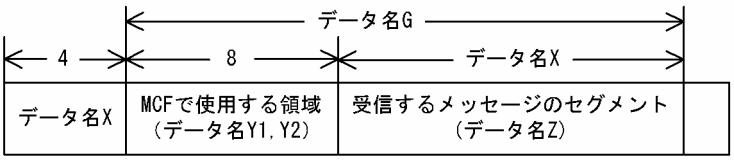
●バッファ形式 1 の場合　（単位：バイト）

●バッファ形式 2 の場合　（単位：バイト）

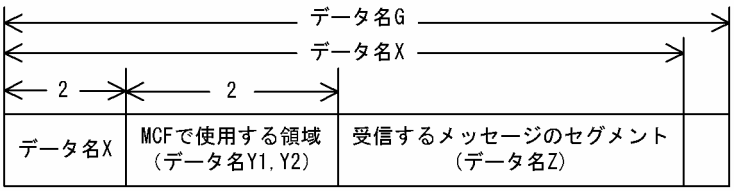


セグメントを受信する領域（一意名 4 で示す領域）の形式を次に示します。

●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

同期型のメッセージの送受信を示す要求コード「VALUE 'SENDRECV'」を設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 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

送信するセグメントを設定します。次のどちらかを設定します。

VALUE 'ESI△'

先頭セグメントまたは中間セグメントを送信する場合に設定します。

VALUE 'EMI△'

最終セグメントを送信する場合や，論理メッセージが単一セグメントの場合に設定します。

メッセージの送信の終了を連絡するために，最後は必ずこの値を設定してください。この値を設定して CBLDCMCF('SENDRECV')を呼び出すと，相手システムからの応答を待ちます。

●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

0 を設定します。

●データ名 M5

CBLDCMCF('SENDRECV')を呼び出してから終了するまでの最大時間を設定します。0 を設定した場合，MCF マネジャ定義の UAP 共通定義で指定した同期型送受信監視時間（mcfmuap -t sndrcvtim）が設定されます。負の値を設定した場合は，時間を監視しません。

●データ名 M6

空白を指定します。

●データ名 M7

使用するバッファ形式を設定します。

VALUE '1'

バッファ形式 1 を使用する場合に設定します。

VALUE '2'

バッファ形式 2 を使用する場合に設定します。

空白

省略されたものとして，「VALUE '1'」（バッファ形式 1）が設定されます。

●データ名 N

MCF で使用する領域です。

●データ名 O

同期型のメッセージの送受信を示す「VALUE 'IO△△'」を設定します。

●データ名 P

メッセージを出力して応答を入力する論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●データ名 Q，データ名 R

空白を設定します。

●データ名 T

MCF で使用する領域です。

●データ名 U

【バッファ形式 1 の場合】 PIC 9(9)

送信するセグメントの長さを設定します。先頭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で，セグメントの内容がないときは，0 を設定してください。

【バッファ形式 2 の場合】 PIC 9(4)

送信するセグメントの長さ＋ 4 を設定します。先頭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で，セグメントの内容がないときは，4 を設定してください。

●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 で使用する領域です。

●データ名 W

送信するセグメントの内容を設定します。一つのセグメントで 32000 バイトまで送信できます。

先頭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で，セグメントの内容がないときも，必ず設定してください。

●データ名 Y1

【バッファ形式 1 の場合】 PIC X(7)

【バッファ形式 2 の場合】 PIC X(1)

空白を設定します。

●データ名 Y2

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 E

メッセージを受信した日付が YYYYMMDD（YYYY：西暦年　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 X

【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 Z

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。
71002	メッセージキューへの入出力処理時に障害が発生しました。
	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。
	バッファ形式 1 の場合はデータ名 U に 32000 バイトを超える値を設定しています。バッファ形式 2 の場合はデータ名 U に 32004 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，メッセージの送受信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。
71108	メッセージを送信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。

ステータスコード	意味
71108	プロセスのローカルメモリが不足しています。
72000	先頭セグメントを受信する CBLDCMCF('RECEIVE△')を呼び出す前に，CBLDCMCF('SENDRECV')を呼び出しています。
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。
	データ名 P に設定した論理端末名称は，定義されていません。
	CBLDCMCF('SENDRECV')を呼び出せない論理端末を設定しています。
72005	＜データ名 H で VALUE 'ESI△'を設定した場合＞ バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。
72013	データ名 G の指定値を超えるセグメントを受信しました。データ名 G の指定値を超えた部分は切り捨てられました。
72016	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 M4 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。
72019	データ名 M6 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。
72026	データ名 H 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 の指定値が不足しています。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。
72041	＜データ名 H で VALUE 'EMI△'を設定した場合＞ • バッファ形式 1 の場合はデータ名 U 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 U に 0 から 4 バイト，またはマイナス値を設定しています。 • データ名 H に VALUE 'ESI△'を設定した CBLDCMCF('SENDRECV')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。
73001	データ名 M5 に 60 秒を加算した時間が経過しましたが，相手システムまたは MCF 通信プロセスからの応答がありません。
	出力キューからのメッセージの読み込み時に障害が発生しました。
	相手システムから受信打ち切りを受信しました。
	CBLDCMCF('SENDRECV')を呼び出したあとで，UA 障害またはコネクション障害が発生しました。
73002	メッセージを送信しようとしましたが，送信バッファまたは編集バッファを確保できませんでした。
	出力先の論理端末で MCF 通信プロセスの内部障害が発生しました。

ステータスコード	意味
73008	論理端末が閉塞中，または MCF が終了処理中に，CBLDCMCF('SENDRECV')を呼び出しました。
73010	入力または出力メッセージ編集 UOC で障害が発生しました。
	メッセージの編集エラーが発生しました。
73015	出力先の論理端末は，ほかの UAP で仕掛り中です。
73018	データ名 M5 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('TACTCN ') ― コネクションの確立（COBOL 言語）

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)   VALUE SPACE.
   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

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 TP1/NET/User Agent 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「VALUE 'TACTCN△△'」を設定します。

●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。

VALUE '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。

VALUE '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，「VALUE 'LE△△'」（論理端末名称を指定）が設定されます。

●データ名 D1，データ名 D2，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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。

例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●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 は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称，またはコネクション ID の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('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 ID 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。
71014	TP1/NET/NCSB，または 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 のコネクショングループを指定しています。
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。
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△△'，または空白以外の値が設定されています。
72059	データ名 D1，データ名 D2，またはデータ名 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 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。

ステータスコード	意味
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 ') ― 論理端末の閉塞解除（COBOL 言語）

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)   VALUE SPACE.
   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 TP1/NET/User Agent 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「VALUE 'TACTLE△△'」を設定します。

●データ名 C，データ名 D1，データ名 D2，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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●データ名 F1

閉塞解除する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の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('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。

ステータスコード	意味
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 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 D1，データ名 D2，またはデータ名 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 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('TDCTCN ') − コネクションの解放（COBOL 言語）

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)   VALUE SPACE.
   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

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 TP1/NET/User Agent 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「VALUE 'TDCTCN△△'」を設定します。

●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。

VALUE '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。

VALUE '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，「VALUE 'LE△△'」（論理端末名称を指定）が設定されます。

●データ名 D1

解放するコネクションの指定方法を設定します。

VALUE '1'

コネクションを強制的に解放します。

VALUE '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）が設定されます。

●データ名 D2，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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。

例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●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 は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称，またはコネクション ID の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('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 ID 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('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，または 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 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。
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△△'，または空白以外の値が設定されています。
72059	データ名 D2，またはデータ名 D3 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
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。
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 ') － 論理端末の閉塞（COBOL 言語）

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)   VALUE SPACE.
   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 TP1/NET/User Agent 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「VALUE 'TDCTLE△△'」を設定します。

●データ名 C，データ名 D1，データ名 D2，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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●データ名 F1

閉塞する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の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('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。

ステータスコード	意味
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 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 D2，またはデータ名 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 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。
72075	データ名 D1 に空白でない値が設定されています。

CBLDCMCF('TLSCN ') － コネクションの状態取得（COBOL 言語）

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「VALUE 'TLSCN△△△'」を設定します。

●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。

VALUE '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。

VALUE '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，「VALUE 'LE△△'」（論理端末名称を指定）が設定されます。

●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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。

例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●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 は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称，またはコネクション ID の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，およびデータ名 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

この命令文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。8 バイトに満たない場合，コネクション ID の後ろが空白で埋められます。

●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。

VALUE 'UA△△'

OSAS/UA プロトコル

●データ名 P

要求したコネクションの状態として，次の値が設定されます。

VALUE 'ACT△'

確立状態

VALUE 'ACTB'

確立処理中状態

VALUE 'DCT△'

解放状態

VALUE 'DCTB'

解放処理中状態

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。

ステータスコード	意味
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 ID 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，または 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 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。
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△△'，または空白以外の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 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。
72076	データ名 M に 1 以外の値が設定されています。

CBLDCMCF('TLSLE ') － 論理端末の状態取得（COBOL 言語）

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)  LOW-VALUE.
      03 データ名P   PIC X(4).
      03 データ名Q   PIC X(40) 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

論理端末の状態取得を示す要求コード「VALUE 'TLSLE△△△'」を設定します。

●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 からこの命令文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。

注※

MCF 環境定義（mcftenv -s）で指定する MCF 通信プロセス識別子は 16 進数と見なしてください。例えば，MCF 通信プロセス識別子が 10 の場合，16 を設定してください。

●データ名 F1

状態を取得する論理端末の名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の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，およびデータ名 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

この命令文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。8 バイトに満たない場合，論理端末名称の後ろが空白で埋められます。

●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。

VALUE 'ACT△'

閉塞解除状態

VALUE 'DCT△'

閉塞状態

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 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。

ステータスコード	意味
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 以外の値が設定されています。

RECEIVE － メッセージの受信（データ操作言語）

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〕
   〔SYNCHRONOUS MODE IS  {ASYNC | データ名6} 〕.
```

DATA DIVISION（データ記述項）の指定

```
01  一意名1.
    02  データ名A   PIC 9(4)  COMP.
    02  データ名B   PIC X(2).
    02  データ名C   PIC X(n).
```

PROCEDURE DIVISION（通信文）の指定

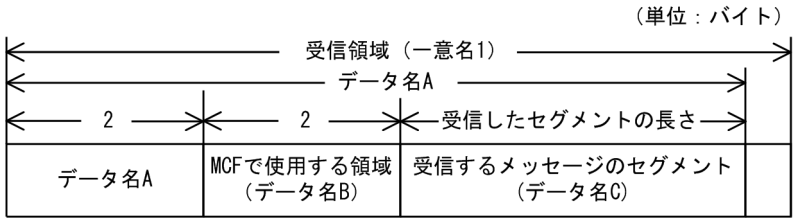
```
RECEIVE 通信記述名
    〔FIRST〕 SEGMENT
      INTO  一意名1.
```

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△')

セグメントを受信する領域（一意名 1 で示す領域）の形式を次に示します。



UAP で値を設定する項目

●FOR 句

次のどちらかの値を設定します。

INPUT

一方送信メッセージの受信

I-O

問い合わせメッセージの受信

●SYMBOLIC TERMINAL 句

中間セグメントまたは最終セグメントを受信する場合，データ名 2 に入力元の論理端末名称を設定します。先頭セグメントの受信時に返された論理端末名称を設定してください。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

先頭セグメントまたは単一セグメントの受信処理終了後，SYMBOLIC TERMINAL 句には OpenTP1 から値が返されます。

●SYNCHRONOUS MODE 句

非同期型のメッセージの受信を示す，次のどちらかの値を設定します。

ASYNC

非同期型のメッセージの受信

データ名 6

次の値を設定したデータ項目

'0'または'△'：非同期型のメッセージの受信

省略した場合は，ASYNC（非同期型のメッセージの受信）が設定されます。

●データ名 B

MCF で使用する領域です。

●FIRST

先頭セグメントを受信する場合に設定します。

OpenTP1 から値が返される項目

●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りません。データ名 1 にステータスコードが返されます。

●SYMBOLIC TERMINAL 句

先頭セグメントまたは単一セグメントを受信する場合，入力元の論理端末名称を受け取りたいときに設定します。省略した場合は，論理端末名称を受け取れません。データ名 2 にメッセージ入力元の論理端末名称が返されます。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろが空白で埋められます。

中間セグメントまたは最終セグメントを受信する場合は，ここで返された論理端末名称を SYMBOLIC TERMINAL 句に設定します。

●MESSAGE DATE 句

メッセージを受信した日付を受け取りたい場合に設定します。省略した場合は，メッセージを受信した日付を受け取れません。データ名 3 にメッセージを受信した日付が YYMMDD（YY：西暦下 2 けた　MM：月　DD：日）の形式で返されます。

●MESSAGE TIME 句

メッセージを受信した時刻を受け取りたい場合に設定します。省略した場合は，メッセージを受信した時刻を受け取れません。データ名 4 に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 A

受信したセグメントの長さ＋ 4 が返されます。

●データ名 C

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 RECEIVE 文を 2 回以上実行しています。中間セグメントまたは最終セグメントを受信する場合は，FIRST を設定しないで 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，再び次のメッセージを受信する 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	メッセージキューからの入力処理中に障害が発生しました。
	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。
	MCF が終了処理中のため，メッセージの受信を受け付けられません。
71108	メッセージの受信に必要な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＞ • 非同期型のメッセージの先頭セグメントを受信する RECEIVE 文を実行する前に，中間セグメントまたは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 を設定して 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，再び非同期型のメッセージを受信する RECEIVE 文を実行しています。
	＜ SPP の実行でリターンした場合＞ SPP では非同期型のメッセージを受信する RECEIVE 文を実行できません。

ステータスコード	意味
72001	SYMBOLIC TERMINAL 句に設定した論理端末名称が間違っています。
	SYMBOLIC TERMINAL 句に設定した論理端末名称は，定義されていません。
	RECEIVE 文を実行できない論理端末を設定しています。
	SYNCHRONOUS MODE 句に SYNC が設定されているか，データ名 6 に'1'が設定されています。
72013	一意名 1 のサイズを超えるセグメントを受信しました。一意名 1 のサイズを超えた部分は切り捨てられました。
72016	通信文に WAITING 句が設定されています。
72020	SYNCHRONOUS MODE 句に設定した値が間違っています。
72024	FOR 句に設定した値が間違っています。
72036	一意名 1 のサイズが不足しています。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（データ操作言語）

形式 1（セグメントの内容を設定して送信する場合）

DATA DIVISION（通信記述項）の指定

```
CD 通信記述名
   FOR {OUTPUT | I-O}
   〔STATUS KEY IS データ名1〕
   〔SYMBOLIC TERMINAL IS データ名2〕
   〔SYNCHRONOUS MODE IS {SYNC | ASYNC | データ名6}〕
   〔SWITCHING MODE IS {NORMAL | PRIOR | データ名7}〕
   〔DETAIL MODE IS データ名10〕
   〔WAITING TIME IS データ名11〕.
```

DATA DIVISION（データ記述項）の指定

```
01  一意名1.
    02  データ名A   PIC 9(4)  COMP.
    02  データ名B   PIC X(2).
    02  データ名C   PIC X(n).
01  一意名3.
    02  データ名D   PIC 9(4)  COMP.
    02  データ名E   PIC X(2).
    02  データ名F   PIC X(n).
```

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}〕
    〔BEFORE RECEIVING MESSAGE INTO 一意名3〕.
```

形式 2（セグメントの内容を設定しないで，メッセージの送信の終了だけ連絡する場合）

DATA DIVISION（通信記述項）の指定

```
CD 通信記述名
   FOR {OUTPUT | I-O}
   〔STATUS KEY IS データ名1〕
   〔SYMBOLIC TERMINAL IS データ名2〕
   〔SWITCHING MODE IS {NORMAL | PRIOR | データ名7}〕.
```

PROCEDURE DIVISION（通信文）の指定

```
SEND 通信記述名  WITH EMI.
```

機能

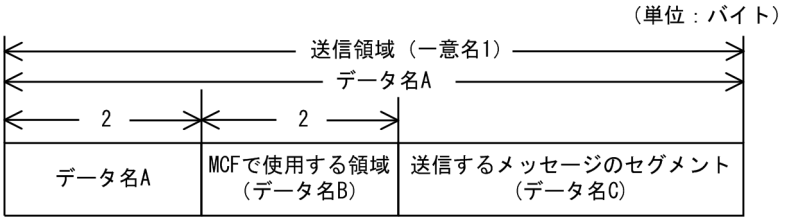
次に示す CALL インタフェースの機能を実現します。

- 一方送信メッセージの送信　CBLDCMCF('SEND△△△△')

- 同期型のメッセージの送受信　CBLDCMCF('SENDRECV')
- 応答メッセージの送信　CBLDCMCF('REPLY△△△')

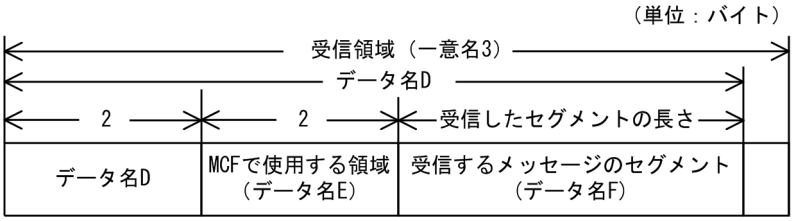
同期型のメッセージの送受信では，単一セグメントの論理メッセージだけを扱えます。

セグメントを送信する領域（一意名 1 で示す領域）の形式を次に示します。



同期型メッセージの送受信の場合，セグメントを受信する領域（一意名 3 で示す領域）も設定します。

セグメントを受信する領域の形式を次に示します。



UAP で値を設定する項目

●FOR 句

次のどちらかの値を設定します。

OUTPUT

一方送信メッセージの送信

I-O

応答メッセージの送信，または同期型のメッセージの送受信

●SYMBOLIC TERMINAL 句

論理端末名称を設定したデータ項目を設定します。データ名 2 に出力先の論理端末名称を設定します。論理端末名称は最大 8 バイトの長さです。8 バイトに満たない場合，論理端末名称の後ろを空白で埋めてください。

●SYNCHRONOUS MODE 句

非同期型でメッセージを送信するか，同期型でメッセージを送信するかを設定します。

SYNC

同期型のメッセージの送受信

同期型のメッセージの送受信のとき設定します。

ASYNC

非同期型のメッセージの送信

一方送信メッセージの送信，または応答メッセージの送信のとき設定します。

データ名 6

次の値を設定したデータ項目

'0'または'△'：非同期型のメッセージの送信

'1'：同期型のメッセージの送受信

省略した場合は，ASYNC（非同期型メッセージの送信）が設定されます。

●SWITCHING MODE 句

一方送信メッセージの場合に，一般か優先かを設定します。

NORMAL

一般の一方送信メッセージ

PRIOR

優先の一方送信メッセージ

データ名 7

次の値を設定したデータ項目

'0'または'△'：一般の一方送信メッセージ

'1'：優先の一方送信メッセージ

省略した場合および非応答型のアプリケーションから応答メッセージを送信した場合は，NORMAL（一般の一方送信メッセージ）が設定されます。

●DETAIL MODE 句

非同期型のメッセージの送信の場合に，出力通番を付けるかどうかを設定します。データ名 10 に次のどちらかを設定します。

データ名 10

次の値を設定したデータ項目

'0'または'△'：出力通番を付けます。

'1'：出力通番を付けません。

省略した場合および非応答型のアプリケーションから応答メッセージを送信した場合は，出力通番を付けません。

●WAITING TIME 句

SEND 文を実行してから終了するまでの，最大時間を設定したデータ項目を設定します。同期型のメッセージの送受信の場合に設定します。

データ名 11

監視時間の値を HHMMSS00（HH：時　MM：分　SS：秒　00 は固定）の形式で設定したデータ項目を設定します。

省略した場合，またはデータ名 11 に'00000000'を設定した場合，MCF マネジャ定義の UAP 共通定義で指定した同期型送受信監視時間（mcfmuap -t sndrcvtim）が設定されます。

●データ名 A

送信するセグメントの長さ＋４を設定します。

●データ名 B

MCF で使用する領域です。

●データ名 C

送信するセグメントの内容を設定します。一つのセグメントで 32000 バイトまで送信できます。

●データ名 E

MCF で使用する領域です。

●WITH 句

送信するセグメントを設定します。非同期型のメッセージ送信の場合だけ設定します。

ESI

先頭セグメントまたは中間セグメントの送信

EMI

最終セグメントまたは単一セグメントの送信

一意名 2

次の値を設定したデータ項目

'1'：ESI（先頭セグメントまたは中間セグメントの送信）

'2'：EMI（最終セグメントまたは単一セグメントの送信）

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の送信）が設定されます。

なお，同期型のメッセージの送受信（SENDRECV）の場合は，EMI を指定するか，または指定を省略してください。

●BEFORE 句

同期型のメッセージの送受信の場合に，セグメントを受信するデータ項目（一意名 3）を設定します。一方送信メッセージの送信，または，応答メッセージの送信の場合，BEFORE 句を省略します。

OpenTP1 から値が返される項目

●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。

●データ名 D

受信したセグメントの長さ＋ 4 が返されます。

●データ名 F

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。
71002	メッセージキューへの出力処理中に障害が発生しました。
	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。
	データ名 A に 32004 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，メッセージの送信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。
71108	メッセージを送信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＞ • 先頭セグメントを受信する RECEIVE 文を実行する前に，SEND 文を実行しています。 • 非応答型のアプリケーションからの問い合わせ応答をしない（UAP 共通定義（mcfmuap -c）の noansreply オペランドに no を指定）場合に，非応答型のアプリケーションから応答メッセージを送信しています。
	＜ 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，SEND 文を実行しています。 • SPP では，応答メッセージを送信する SEND 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。

ステータスコード	意味
72001	SYMBOLIC TERMINAL 句に設定した論理端末名称は，定義されていません。
	SEND 文を実行できない論理端末を設定しています。
	＜応答メッセージの送信を行う（FOR 句に I-O を設定し，かつ，SYNCHRONOUS MODE 句を省略または SYNCHRONOUS MODE 句に ASYNC を設定または SYNCHRONOUS MODE 句に 0 もしくは空白を設定したデータ名 6 を設定）場合＞ 入力元論理端末は，応答メッセージの送信をサポートしていません。
72005	＜ WITH 句に ESI を設定または WITH 句に 1 を設定した一意名 2 を設定した場合＞ データ名 A に 0 から 4 バイト，またはマイナス値を設定しています。
72008	応答メッセージの最終セグメントを送信する SEND 文を実行したあとに，再び応答メッセージを送信する SEND 文を実行しています。
	SEND 文を実行して応答型のアプリケーションを起動させたあとで，再び応答メッセージを送信する SEND 文を実行しています。
72013	データ名 F のサイズを超えるセグメントを受信しました。データ名 F のサイズを超えた部分は切り捨てられました。
72017	DETAIL MODE 句に設定した値が間違っています。
72018	SWITCHING MODE 句に設定した値が間違っています。
72020	SYNCHRONOUS MODE 句に設定した値が間違っています。
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。
72036	データ名 F のサイズが不足しています。5 バイト以上の領域を確保してください。
72037	BEFORE 句に設定した値が間違っています。
72041	＜ WITH 句を省略または WITH 句に EMI を設定または WITH 句に 2 を設定した一意名 2 を設定した場合＞ データ名 A に 0 から 4 バイト，またはマイナス値を設定しています。
	＜メッセージの送信の終了を連絡した場合＞ WITH 句に ESI を設定または WITH 句に 1 を設定した一意名 2 を設定した SEND 文を呼び出さないで，メッセージの送信の終了を連絡しています。
72045	継続問い合わせ応答型のアプリケーションはサポートしていません。
72047	継続問い合わせ応答型のアプリケーションはサポートしていません。
73001	データ名 11 に 60 秒を加算した時間が経過しましたが，相手システムまたは MCF 通信プロセスからの応答がありません。
	出力キューからのメッセージの読み込み時に障害が発生しました。
	相手システムから受信打ち切りを受信しました。
	同期型のメッセージを送信する SEND 文を実行したあとで，UA 障害またはコネクション障害が発生しました。

ステータスコード	意味
73002	メッセージを送信しようとしましたが，送信バッファまたは編集バッファを確保できませんでした。
	出力先の論理端末で MCF 通信プロセスの内部障害が発生しました。
73008	論理端末が閉塞中，または MCF が終了処理中に，SEND 文を呼び出しました。
73010	入力または出力メッセージ編集 UOC で障害が発生しました。
	メッセージの編集エラーが発生しました。
73015	出力先の論理端末は，ほかの UAP で仕掛り中です。
73018	WAITING TIME 句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。

5 ユーザオウンコーディング，MCF イベントインタフェース

この章では，TP1/NET/User Agent に関連するユーザオウンコーディング，およびMCF イベントのインタフェースについて説明します。

5.1　ユーザオウンコーディングインタフェース

メッセージ送受信の UAP を，より多様な業務に対応させるために補助するプログラムを，**ユーザオウンコーディング**（以降，**UOC** と略します）といいます。

TP1/NET/User Agent で使用できる UOC を次に示します。

- 入力メッセージ編集 UOC
- 出力メッセージ編集 UOC
- 送信メッセージ通番編集 UOC

UOC を使用する場合は，あらかじめ MCF メイン関数または UAP のメイン関数に UOC 関数のアドレスを登録し，UOC 関数のオブジェクトファイルを MCF 通信プロセスまたは UAP の実行形式プログラムに結合（リンケージ）しておく必要があります。また，UOC は C 言語で作成します。

5.1.1　入力メッセージの編集とアプリケーション名の決定

入力メッセージ編集 UOC は，受信した論理メッセージをユーザ任意の形式に変換します。そして受信した論理メッセージを基に，ユーザ任意のアプリケーション名を決定できます。

UOC は，UAP を起動するメッセージの最終セグメントを受信すると起動します。ただし，MCF イベント発生時と UAP からのアプリケーションプログラム起動時には，UOC は起動しません。

ユーザは，MCF メイン関数で UOC 関数アドレスを設定します。また，必要に応じてコネクション定義（mcftalccn -e）で，メッセージ編集用バッファグループ番号を定義します。

(1)　入力メッセージの編集

受信したメッセージが格納されている受信バッファ，および MCF 通信構成定義で指定した編集バッファを引き渡します。UOC では，これらのバッファを使用して，入力メッセージの編集ができます。

また，UAP に通知するメッセージのセグメントは，受信バッファ，または編集バッファのどちらかに格納されたものを使用できます。どちらのセグメントを使用するかは，UOC から返されるリターンコードによって選択できます。

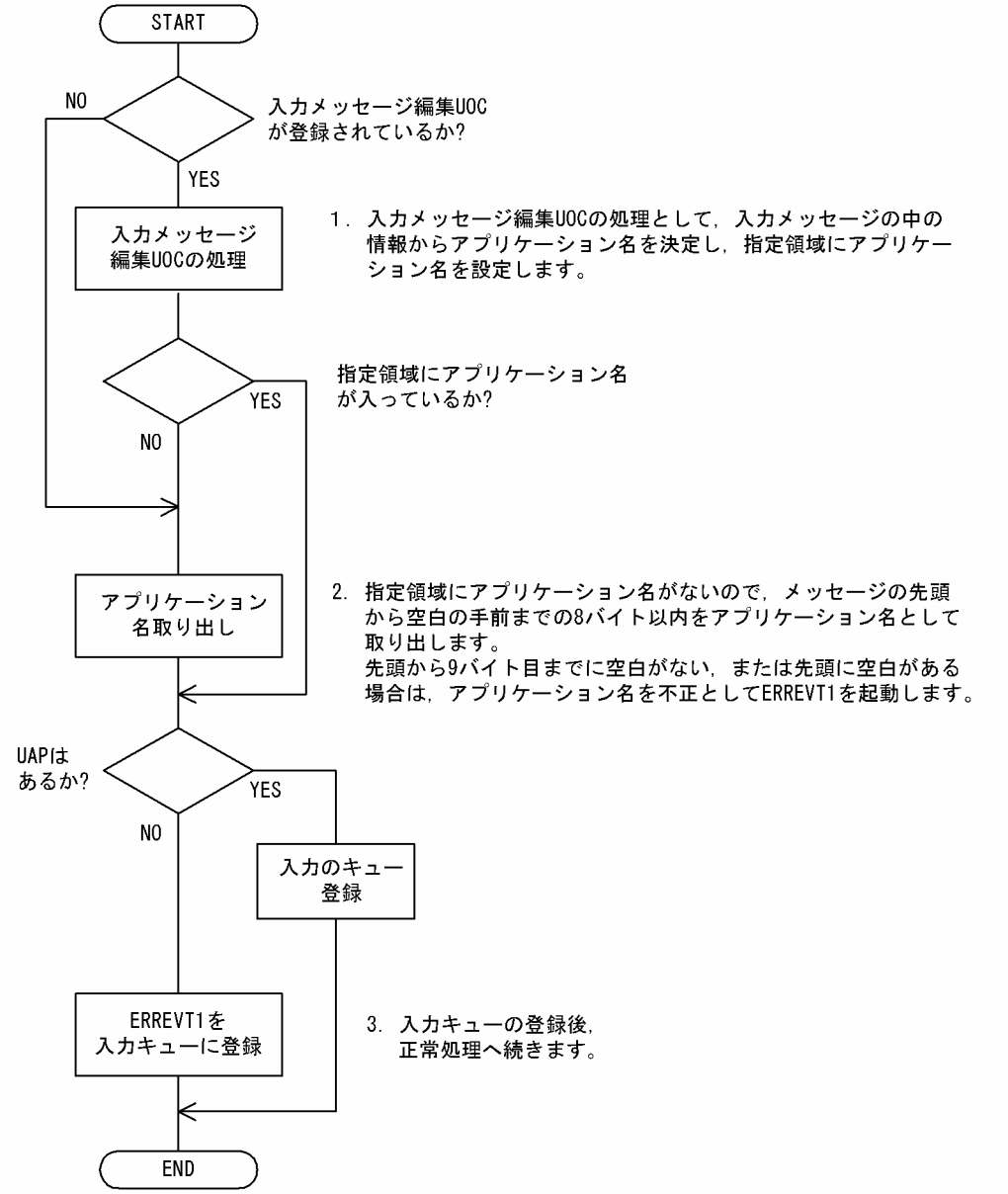
(2)　アプリケーション名の決定

該当する MCF に入力メッセージ編集 UOC が登録されている場合，論理メッセージの受信と同時にアプリケーション名を決定できます。

UOC でアプリケーション名を決定する場合，アプリケーション名の形式は，アプリケーション名格納領域の先頭から，'¥0'の手前までの 1～8 バイトの識別子です。先頭から 9 バイト目までに'¥0'がないときは，アプリケーション名を不正とし，ERREVT1 を起動します。

アプリケーション名の決定の処理を次の図に示します。

図 5-1　アプリケーション名の決定の処理



(3)　UOC エラーリターン処理

UOC から DCMCF_UOC_MSG_NG でリターンした場合，MCF はメッセージログを出力し，相手システムに UERR を送信します。相手システムからの UERR に対する応答を受信したあとに，MCF は障害通知イベント（CERREVT）を起動します。

UOC で障害を検出し，エラー処理 UAP を起動したい場合は，ユーザ任意のエラー処理 UAP のアプリケーション名を設定します。このとき，MCF には DCMCF_UOC_MSG_OK（_RCV）でリターンします。

(4) UOC パラメタ不正の場合の処理

UOC で設定したパラメタに不正があった場合，MCF はメッセージログを出力し，相手システムに UERR を送信します。相手システムからの UERR に対する応答を受信したあとに，MCF は障害通知イベント（CERREVT）を起動します。

(5) OpenTP1 への組み込み方法

スタート関数（dc_mcf_svstart）を発行する MCF メイン関数に，作成した UOC の関数アドレスを設定します。入力メッセージ編集 UOC の関数アドレスは任意に決められます。UOC 関数をコンパイルして生成した UOC オブジェクトファイルを，UOC 関数を登録した MCF メイン関数と結合して，TP1/NET/User Agent の実行形式プログラムを生成します。MCF メイン関数の詳細については，「8.2 MCF メイン関数の作成」を参照してください。

5.1.2 入力メッセージ編集 UOC インタフェース

入力メッセージ編集 UOC は，次に示す形式で呼び出します。

(1) 形式

ANSI C，C++の形式

```
#include <dcmcfuoc.h>
DCLONG uoc_func(dcmcf_uoc_min_n *parm)
```

K&R 版 C の形式

```
#include <dcmcfuoc.h>
DCLONG  uoc_func(parm)

dcmcf_uoc_min_n *parm ;
```

(2) 説明

uoc_func（入力メッセージ編集 UOC）を呼び出すとき，MCF は次に示す所定のパラメタを parm に設定します。

(3) パラメタの内容

(a) dcmcf_uoc_min_n の内容

```
typedef struct {
    DCLONG pro_kind;            …プロトコル種別
    char le_name[9];            …論理端末名称
    char reserve1[7];           …予備
    DCLONG rcv_prim;            …受信サ-ビスプリミテイブ
```

```
    dcmcf_uocbuff_list_n *buflist_adr;
                                …受信バッファリストアドレス
    dcmcf_uocbuff_list_n *ebuflist_adr;
                                …編集バッファリストアドレス
    char aplname[9];            …アプリケーション名称
    char reserve2[7];           …予備
    char *pro_indv_ifa;         …MCF使用領域
    DCLONG rtn_detail;          …詳細リターンコード
    char reserve3[8];           …予備
} dcmcf_uoc_min_n;
```

(b) dcmcf_uocbuff_list_n（バッファリスト）の内容

```
typedef struct {
    DCLONG buf_num;             …バッファ情報数
    DCLONG used_buf_num;        …使用バッファ情報数
    char reserve1[8];           …予備
    dcmcf_uocbufinf_n buf_array[DCMCF_UOC_BUFF_MAX];
                                …バッファ情報
} dcmcf_uocbuff_list_n;
```

(c) dcmcf_uocbufinf_n（バッファ情報）の内容

```
typedef struct {
    char *buf_adr;              …バッファアドレス
    DCULONG buf_size;           …バッファ最大長
    DCULONG seg_size;           …バッファ使用長
    char reserve1[4];           …予備
    dcmcfuoc_w_type buff_id;    …MCF内部情報1
    DCMLONG buff_addr;          …MCF内部情報2
    char reserve2[4];           …予備
} dcmcf_uocbufinf_n;
```

(4) MCF が値を設定する項目

(a) dcmcf_uoc_min_n

- pro_kind

  プロトコル種別として，次の値が設定されます。

  DCMCF_UOC_PRO_OSAS

  OSAS/UA プロトコル

- le_name

  メッセージを入力した論理端末名称が設定されます。

- rcv_prim

  受信サービスプリミティブとして，次の値が設定されます。

  DCMCF_UOC_RCV_INQ

  UINQ

DCMCF_UOC_RCV_REP

UREP

DCMCF_UOC_RCV_BRD

UBRD

DCMCF_UOC_RCV_REP_SR

sendrecv 時の UREP 受信

この場合，アプリケーション名は無効です。

- buflist_adr

  受信用バッファリストのアドレスが設定されます。

- ebuflist_adr

  編集用バッファリストのアドレスが設定されます。

  メッセージ編集用バッファが未定義の場合，つまり，コネクション定義（mcftalccn）の-e オプションを省略した場合，ebuflist_adr には NULL が設定されます。

- aplname

  アプリケーション名が設定されます。

- pro_indv_ifa

  MCF で使用するパラメタです。

(b) dcmcf_uocbuff_list_n（バッファリスト）

- buf_num

  バッファ情報の数が設定されます。

- buf_array

  バッファ情報の配列が設定されます。バッファ情報は，buf_num の数だけ設定されます。

(c) dcmcf_uocbufinf_n（バッファ情報）

- buf_adr

  バッファのアドレスが設定されます。

- buf_size

  バッファの最大長が設定されます。

- seg_size

  送信，または受信用バッファリストの場合だけ，バッファの使用長が設定されます。

- buff_id，buff_addr

  MCF で使用するパラメタです。

(5) ユーザが値を設定する項目

(a) dcmcf_uoc_min_n

- aplname
  UOC で決定したアプリケーション名を設定します。
- rtn_detail
  詳細リターンコードを設定します。
  このコードは，UOC が DCMCF_UOC_MSG_NG をリターンしたときに，MCF に渡されます。MCF は，詳細リターンコードをメッセージログファイルに出力します。詳細リターンコードは-19000～-19999 の範囲で指定してください。

(b) dcmcf_uocbuff_list_n（バッファリスト）

- used_buf_num
  使用したバッファ情報の数を設定します。

(c) dcmcf_uocbufinf_n（バッファ情報）

- seg_size
  バッファの使用長を設定します。

(6) リターン値

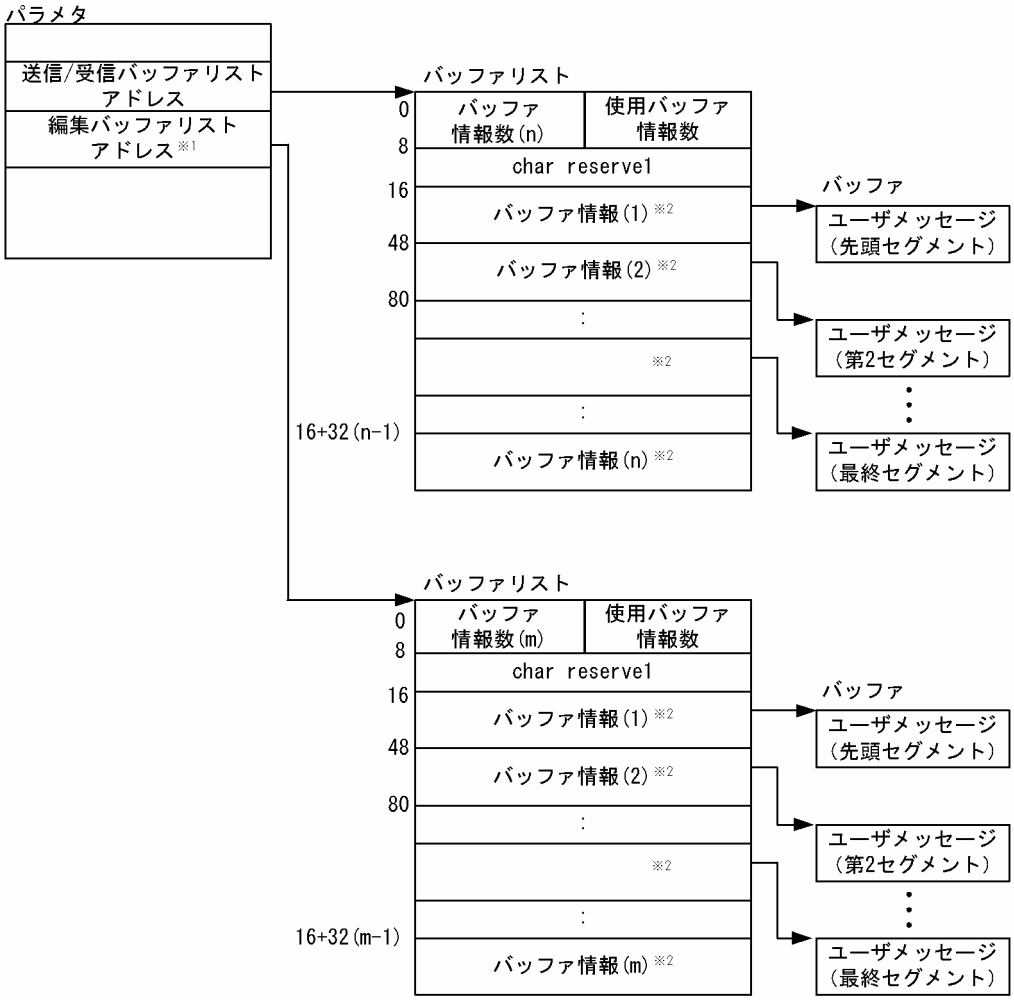
uoc_func()は次のコードでリターンしてください。

リターン値	意味
DCMCF_UOC_MSG_OK	正常リターン（編集バッファでスケジューリング）
DCMCF_UOC_MSG_OK_RCV	正常リターン（受信バッファでスケジューリング）
DCMCF_UOC_MSG_NG	メッセージ編集エラー

(7) パラメタとバッファの関係

UOC インタフェース用のパラメタとバッファの関係を次の図に示します。

図 5-2　UOC インタフェース用のパラメタとバッファの関係



注※1

mcftalccn コマンドの-e オプションを指定しない場合，NULL となり，バッファリストとバッファは確保されません。

注※2

バッファ情報は次の形式をしています。

5.1.3　出力メッセージの編集

出力メッセージの編集 UOC は，応答メッセージまたは一方送信メッセージの編集をする UOC です。出力メッセージの編集 UOC は，UAP が発行した送信メッセージを相手システムに実際に送信する前に処理するように位置させます。出力キューから全セグメントを読み出すと起動します。

ユーザは，MCF メイン関数で UOC 関数アドレスを設定します。また，必要に応じてコネクション定義（mcftalccn -e）で，メッセージ編集用バッファグループ番号を定義します。

(1)　出力メッセージの編集

送信するメッセージが格納されている送信バッファ，およびコネクション定義で指定した編集バッファを引き渡します。UOC では，これらのバッファを使用して，出力メッセージの編集処理ができます。

また，UOC からのリターンコードによって UAP に通知するメッセージとして送信バッファに格納されたものを使用するか，編集バッファに格納されたものを使用するかを選択できます。

(2)　UOC エラーリターン処理

UOC から DCMCF_UOC_MSG_NG でリターンした場合，MCF は次に示す処理をします。

- reply 型論理端末の場合
  メッセージログを出力し，相手システムに UERR を送信します。相手システムからの UERR に対する応答を受信したあとに，MCF は障害通知イベント（CERREVT）を起動します。
- request 型論理端末または send 型論理端末の場合
  メッセージログを出力し，該当するメッセージを破棄します。

(3)　UOC パラメタ不正の場合の処理

UOC で設定した値に不正があった場合，MCF は次に示す処理をします。

- reply 型論理端末の場合
  メッセージログを出力し，相手システムに UERR を送信します。相手システムからの UERR に対する応答を受信したあとに，MCF は障害通知イベント（CERREVT）を起動します。
- request 型論理端末または send 型論理端末の場合
  メッセージログを出力し，該当するメッセージを破棄します。

(4)　OpenTP1 への組み込み方法

入力メッセージ編集 UOC の組み込み方法と同じです。「5.1.1(5)　OpenTP1 への組み込み方法」を参照してください。

5.1.4　出力メッセージ編集 UOC インタフェース

出力メッセージ編集 UOC は，次に示す形式で呼び出します。

(1)　形式

ANSI C，C++の形式

```
#include <dcmcfuoc.h>
DCLONG uoc_func(dcmcf_uoc_mout_n *parm)
```

K&R 版 C の形式

```
#include <dcmcfuoc.h>
DCLONG uoc_func(parm)

dcmcf_uoc_mout_n  *parm ;
```

(2)　説明

uoc_func（出力メッセージ編集 UOC）を呼び出すとき，MCF は次に示す所定のパラメタを parm に設定します。

(3)　パラメタの内容

(a)　dcmcf_uoc_mout_n の内容

```
typedef struct {
    DCLONG pro_kind;            …プロトコル種別
    char le_name[9];            …論理端末名称
    char reserve1[7];           …予備
    dcmcf_uocbuff_list_n *buflist_adr;
                                …送信バッファリストアドレス
    dcmcf_uocbuff_list_n *ebuflist_adr;
                                …編集バッファリストアドレス
    DCLONG output_no;           …メッセージ出力通番
    char msg_type;              …メッセージ種別
    char outputno_flag;         …メッセージ出力通番有効フラグ
    char resend_flag;           …再送フラグ
    char reserve2[1];           …予備
    char *pro_indv_ifa;         …MCF使用領域
    DCLONG rtn_detail;          …詳細リターンコード
    char reserve3[20];          …予備
} dcmcf_uoc_mout_n;
```

(b)　dcmcf_uocbuff_list_n（バッファリスト），dcmcf_uocbufinf_n（バッファ情報）の内容

「5.1.2　入力メッセージ編集 UOC インタフェース」を参照してください。

(4) MCF が値を設定する項目

(a) dcmcf_uoc_mout_n

- pro_kind
  プロトコル種別として，次の値が設定されます。

  DCMCF_UOC_PRO_OSAS
  　OSAS/UA プロトコル
- le_name
  メッセージを入力した論理端末名称が設定されます。
- buflist_adr
  送信用バッファリストのアドレスが設定されます。
- ebuflist_adr
  編集用バッファリストのアドレスが設定されます。

  メッセージ編集用バッファが未定義の場合，つまり，コネクション定義（mcftalccn）の-e オプションを省略した場合，ebuflist_adr には NULL が設定されます。
- output_no
  メッセージ出力通番が設定されます。ただし，outputno_flag が DCMCF_UOC_OUTPUTNO_OK のときだけ有効です。
- msg_type
  メッセージ種別として，次の値が設定されます。

  'o'
  　問い合わせ応答メッセージ

  'n'
  　一般一方送信メッセージ

  'p'
  　優先一方送信メッセージ

  's'
  　同期送信メッセージ
- outputno_flag
  メッセージ出力通番の有効フラグとして，次のどちらかの値が設定されます。

  DCMCF_UOC_OUTPUTNO_OK
  　メッセージ出力通番を有効にします。

  DCMCF_UOC_OUTPUTNO_NG
  　メッセージ出力通番を無効にします。

- resend_flag
  再送メッセージフラグとして，次のどちらかの値が設定されます。
  'r'
  　再送メッセージです。
  'n'
  　再送メッセージではありません。
- pro_indv_ifa
  MCF で使用するパラメタです。

(b)　dcmcf_uocbuff_list_n（バッファリスト），dcmcf_uocbufinf_n（バッファ情報）

「5.1.2　入力メッセージ編集 UOC インタフェース」を参照してください。

(5)　ユーザが値を設定する項目

(a)　dcmcf_uoc_mout_n

- rtn_detail
  詳細リターンコードを設定します。
  このコードは，UOC が DCMCF_UOC_MSG_NG をリターンしたときに，MCF に渡されます。MCF は，詳細リターンコードをメッセージログファイルに出力します。詳細リターンコードは-19000～-19999 の範囲で指定してください。

(b)　dcmcf_uocbuff_list_n（バッファリスト），dcmcf_uocbufinf_n（バッファ情報）

「5.1.2　入力メッセージ編集 UOC インタフェース」を参照してください。

(6)　リターン値

uoc_func()は次のコードでリターンしてください。

リターン値	意味
DCMCF_UOC_MSG_OK	正常リターン（編集バッファでスケジューリング）
DCMCF_UOC_MSG_OK_SND	正常リターン（送信バッファでスケジューリング）
DCMCF_UOC_MSG_NG	メッセージ編集エラー

(7)　パラメタとバッファの関係

UOC インタフェース用のパラメタとバッファの関係は，入力メッセージ編集の場合と同じです。
「5.1.2(7)　パラメタとバッファの関係」を参照してください。

5.1.5　送信メッセージの通番編集

(1)　出力通番編集

送信メッセージの通番編集 UOC では，受け取った出力通番を基に，ユーザ独自の処理ができます。例えば，出力通番を使用して編集した送信メッセージを再送要求します。

送信メッセージの通番編集 UOC を起動する場合，メッセージ送信の関数で，出力通番を付けるように設定してください。UOC は，UAP が send 関数または resend 関数を発行したときに，MCF によって起動されます。したがって，この UOC でメッセージを編集する場合は，先頭セグメントしか編集できません。

(2)　OpenTP1 への組み込み方法

UAP のメイン関数の中に，送信メッセージの通番編集 UOC の関数アドレスを登録しておきます。UAP のメイン関数に登録する dc_mcf_regster 関数の形式を次に示します。

(a)　形式

ANSI C，C++の形式

```
#include<dcmcf.h>
int dc_mcf_regster(DCLONG flags, DCLONG(*uoc_addr)(DCLONG flags,
                    char *termname, DCLONG sendno,
                    DCLONG sendid, DCLONG dataleng,
                    char *senddata))
```

K&R 版 C の形式

```
#include<dcmcf.h>
int dc_mcf_regster(flags, uoc_addr)
DCLONG flags;
DCLONG (*uoc_addr)();
```

(b)　ユーザが値を設定する引数

- flags

  DCMCF_SEND_UOC を指定します。

- uoc_addr

  flags に対応する UOC のアドレスを指定します。

(c)　リターン値

リターン値	意味
DC_OK	正常に終了しました。
DCMCFER_INVALID_ARGS	引数の指定が間違っています。
DCMCFER_NOMEM	ローカルメモリが不足しました。

(d) メイン関数への登録例

- MHP の場合

```
main()
{
extern DCLONG send_uoc();

dc_rpc_open();
dc_mcf_open();
dc_mcf_regster(DCMCF_SEND_UOC,send_uoc);
dc_mcf_mainloop();
dc_mcf_close();
dc_rpc_close();
}
```

- SPP の場合

```
main()
{
extern DCLONG send_uoc();

dc_rpc_open();
dc_mcf_open();
dc_mcf_regster(DCMCF_SEND_UOC,send_uoc);
dc_rpc_mainloop();
dc_mcf_close();
dc_rpc_close();
}
```

メイン関数，サービス関数，スタブ関数に UOC 関数をリンケージして UAP の実行形式プログラムを生成します。

5.1.6 送信メッセージの通番編集 UOC インタフェース

送信メッセージの通番編集 UOC は，次に示す形式で，send_uoc 関数として作成します。なお，UOC の関数名称はユーザの任意です。

(1) 形式

ANSI C，C++の形式

```
#include <dcmcf.h>
DCLONG send_uoc(DCLONG flags, char *termname, DCLONG sendno,
                DCLONG sendid, DCLONG dataleng, char *senddata)
```

K&R 版 C の形式

```
#include <dcmcf.h>
DCLONG send_uoc(flags, termname, sendno, sendid, dataleng,
                senddata)
DCLONG      flags;
```

```
char        *termname;
DCLONG      sendno;
DCLONG      sendid;
DCLONG      dataleng;
char        *senddata;
```

(2) MCF が値を設定する項目

- flags

  送信メッセージの通番編集 UOC がいつ呼ばれたかが設定されます。

  DCMCF_SEND_DML

  メッセージを送信する関数または命令文が呼び出されたときを意味します。

  DCMCF_RESEND_DML

  メッセージを再送する関数または命令文が呼び出されたときを意味します。

- termname

  送信先の論理端末名称が設定されます。

- sendno

  送信メッセージの出力通番が設定されます。

- sendid

  送信するメッセージ種別が設定されます。

  DCMCF_SEND_PRIO

  優先の一方送信メッセージ

  DCMCF_SEND_NORM

  一般の一方送信メッセージ

- dataleng

  送信メッセージ長が設定されます。

- senddata

  セグメントの内容が設定されます。

(3) リターン値

リターン値	意味
DC_OK	正常に終了しました。

5.1.7 UOC 作成上の注意事項

UOC 作成上の注意事項を次に示します。

(1) UOC の構造

UOC で使用するローカル変数のサイズの合計は，各 UOC で 1024 バイト以内になるよう作成してください。また，UOC の中で関数の再帰呼び出しはしないでください。

(2) 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ご注意ください。

- メモリ操作をする関数
  - データ領域管理（例：malloc，free）
  - 共有メモリ管理関数（例：shmctl，shmget，shmop）
  - メモリ操作（例：memcpy）
  - 文字列操作（例：strcpy）
- 時間取得関数

(3) UOC の異常処理

TP1/NET/User Agent の 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()を呼び出すと，MCF が異常終了します。

(4) 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 が呼ばれたりしてもよいように作成してください。

5.2 MCF イベントインタフェース

TP1/NET/User Agent でメッセージ送受信をすると，OpenTP1 の各種システム情報が MHP に通知されます。これを **MCF イベント**といいます。メッセージ送受信処理でエラーや障害が発生した場合，システム内で何が起こっているのかが MCF イベントの内容でわかります。MCF イベントに対応する MHP を MCF イベント処理用 MHP といいます。

MCF イベントは入力キューに渡されて，MCF イベント処理用 MHP で回復処理をします。なお，MCF イベントの発生時は入力メッセージ編集 UOC は呼び出しません。詳細についてはマニュアル「OpenTP1 プログラム作成の手引」を参照してください。

5.2.1 MCF イベントの種類

TP1/NET/User Agent が通知する MCF イベントの種類を次の表に示します。

表 5-1 TP1/NET/User Agent が通知する MCF イベントの種類

MCF イベント名	MCF イベントコード	発生した原因	MCF イベント処理用 MHP の処理の例
不正アプリケーション名検出通知イベント	ERREVT1	メッセージのアプリケーション名が MCF アプリケーション定義にありません。	該当するアプリケーション名がなかったことを報告します。 問い合わせメッセージの場合は，応答メッセージを出力できます。
メッセージ廃棄通知イベント	ERREVT2	次の理由で受信メッセージを破棄しました。 • 入力キューに障害が発生しました。 • 入力メッセージ最大格納数を超過しました。 • 動的共用メモリが不足しました。 • キューファイルが満杯になりました。 • MHP のサービス，サービスグループまたはアプリケーションが閉塞しています。 • スケジュール閉塞されているサービスグループの入力キューに未処理受信メッセージが残った状態で，OpenTP1 を正常終了または計画停止 A で終了しました。 • MHP のサービスグループ，またはアプリケーションがセキュア状態です。 • MHP で呼び出す receive 関数にセグメントを渡す前に，MHP が異常終了しました。	メッセージを廃棄したことを報告します。 問い合わせメッセージの場合は，応答メッセージを出力できます。

MCF イベント名	MCF イベントコード	発生した原因	MCF イベント処理用 MHP の処理の例
メッセージ廃棄通知イベント	ERREVT2	• アプリケーション名に相当する MHP のサービスがありません。 • ユーザサーバ未起動などによって，MHP の起動に失敗しました。 • DBMS の障害などによって，トランザクションの開始に失敗しました。	メッセージを廃棄したことを報告します。 問い合わせメッセージの場合は，応答メッセージを出力できます。
UAP 異常終了通知イベント	ERREVT3	MHP のセグメント受信関数に，セグメントを渡したあとに MHP の異常終了※が発生しました。	UAP 異常終了時の対処障害メッセージを送信します。 問い合わせメッセージの場合は，応答メッセージを出力できます。
タイマ起動メッセージ廃棄通知イベント	ERREVT4	アプリケーションのタイマ起動時に障害が発生しました。	メッセージを廃棄したことを報告します。
未処理送信メッセージ廃棄通知イベント	ERREVTA	次の理由で未処理送信メッセージを破棄しました。 • MCF の正常終了処理時に，未処理送信メッセージの滞留時間監視の時間切れ（タイムアウト）が発生しました。 • 運用コマンド（mcftdlqle）の入力，または API（dc_mcf_tdlqle 関数もしくは CBLDCMCF('TDLQLE△△')）の発行によって，出力キューが削除されました。 • 閉塞されている論理端末の出力キューに未処理送信メッセージが残った状態で，dcstop コマンドが実行されました。	未処理送信メッセージを廃棄したことを報告します。
障害通知イベント	CERREVT	通信管理プログラムのコネクション障害，または論理端末障害が発生しました。	コネクション，または論理端末に障害が発生したことを報告します。
状態通知イベント	COPNEVT	コネクションが確立しました。または，論理端末が閉塞解除しました。	コネクションが確立したことまたは論理端末が閉塞解除したことを報告します。
	CCLSEVT	コネクションが正常に解放されました。	コネクションが解放されたことを報告します。

注※

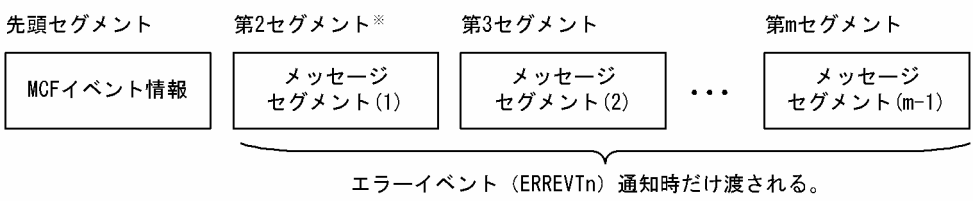
アプリケーション属性定義（mcfaalcap -g）の recvmsg オペランドに r を指定した場合，または dc_mcf_rollback の action に DCMCFRTRY または DCMCFRRTN を指定した場合は除きます。

5.2.2　MCF イベント通知時のセグメント構成

MCF イベントを MHP に通知する場合，先頭セグメントに MCF イベント情報を設定します。エラーイベント（ERREVTn）の場合は，第 2 セグメント以降に処理できなかったメッセージセグメントを最終セグメントまで設定します。

MCF イベント通知時のセグメント構成を次の図に示します。

図 5-3　MCF イベント通知時のセグメント構成



注※　CERREVTの場合，利用者情報を設定します。

MCF イベントは，UAP を作成した言語によって，UAP に通知されるデータの形式が異なります。

エラーイベントの場合はバッファ形式 1 とバッファ形式 2 で先頭の内容が異なります。

このため，それ以降の項目の位置にずれがあります。「5.2.4　MCF イベント情報の形式（COBOL 言語）」のエラーイベントの表では ERREVT1，ERREVT2，ERREVT3，および ERREVTA のバッファ形式別に位置（バイト）を分けて説明しています。

なお，ERREVT4 については，マニュアル「OpenTP1 プログラム作成リファレンス」の該当する言語編を参照してください。

5.2.3　MCF イベント情報の形式（C 言語）

MCF イベント情報は構造体で，MCF イベント処理用 MHP に渡されます。MHP に渡される構造体の形式は，MCF イベントの種類によって異なります。ただし，MCF イベント情報の先頭部分の形式は，各イベントで共通です。

エラーイベント（ERREVTn）で使用する構造体は，<dcmcf.h>で定義してあります。なお，dcmoum_statevt は，<dcmoum.h>で定義されています。<dcmoum.h>の前に<dcmcf.h>を取り込んでおいてください。

(1) MCF イベントの共通ヘッダ

(a) 形式

```
struct dc_mcf_evtheader {
  char mcfevt_name[9];        …MCFイベントコード
  char le_name[16];           …入力元論理端末名称
                               (ERREVT1,ERREVT2,ERREVT3の場合)
                               出力先論理端末名称
                               (ERREVTAの場合)
  char cn_name[9];            …コネクション名
  unsigned char format_kind; …MCF使用領域
  char reserve01;             …予備
  DCLONG time;                …メッセージ入力時刻
};
```

(b) MCF イベントとして設定される項目

- le_name

  ERREVT1，ERREVT2，または ERREVT3 では，メッセージを入力した論理端末名称が設定されます。

  ただし，ERREVT2 または ERREVT3 で，次に示す場合は，'*'が設定されます。

  - SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合
  - 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合

  ERREVTA では，メッセージを出力する論理端末名称が設定されます。

  CERREVT では，論理端末障害が発生した論理端末名称が設定されます。コネクション障害時は無効です。

  COPNEVT では，閉塞解除した論理端末名称が設定されます。コネクション確立時は無効です。

  CCLSEVT では無効です。

- cn_name

  コネクション名が設定されます。

  ただし，ERREVT2 または ERREVT3 で，次に示す場合は，'*'が設定されます。

  - SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合
  - 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合

- time

  メッセージを入力した時刻が，1970 年 1 月 1 日 0 時 0 分 0 秒からの通算の秒数で設定されます。

(2) ERREVT1

(a) 形式

```
struct dc_mcf_evt1_type {
  struct dc_mcf_evtheader  evtheader ;
                             …MCFイベント共通ヘッダ
  char reserve01[12] ;       …予備
  char reserve02[10] ;       …予備
  char reserve03[2] ;        …予備
  char ap_name[10] ;         …アプリケーション名
                               （メッセージに対応する
                                アプリケーション名）
  char reserve04[2] ;        …予備
};
```

(b) MCF イベントとして設定される項目

- ap_name

  次に示すどちらかが設定されます。

  - 形式不正の場合…不正となったアプリケーション名
  - 定義されていない場合…定義されていないアプリケーション名

  アプリケーション名は，MHP から送信されたメッセージの場合に設定されます。MHP 以外から送信された場合は，ヌル文字が設定されます。

(3) ERREVT2

(a) 形式

```
struct dc_mcf_evt2_type {
  struct dc_mcf_evtheader  evtheader ;
                             …MCFイベント共通ヘッダ
  char reserve01[12] ;       …予備
  char reserve02[10] ;       …予備
  char reserve03[2] ;        …予備
  char ap_name[10] ;         …アプリケーション名
                               （メッセージに対応する
                                アプリケーション名）
  short reason_code ;        …理由コード
};
```

(b) MCF イベントとして設定される項目

- ap_name

  エラーになった UAP のアプリケーション名が設定されます。

  アプリケーション名は，MHP から送信されたメッセージの場合に設定されます。MHP 以外から送信された場合は，ヌル文字が設定されます。

- reason_code

  ERREVT2 の理由コードが設定されます。理由コードの詳細については，「付録 H　理由コード一覧」を参照してください。

(4)　ERREVT3

(a)　形式

```
struct dc_mcf_evt3_type {
  struct dc_mcf_evtheader  evtheader ;
                             …MCFイベント共通ヘッダ
  char reserve01[12] ;       …予備
  char map_name[10] ;        …MCF使用領域
  char reserve03[2] ;        …予備
  char ap_name[10] ;         …アプリケーション名
                              （異常が発生したメッセージの
                               アプリケーション名）
  char reserve04[2] ;        …予備
  char service_name[32] ;    …サービス名
  char serv_grp_name[32] ;   …サービスグループ名
  char bid[36] ;             …トランザクションブランチID領域
};
```

(b)　MCF イベントとして設定される項目

- ap_name

  異常が発生した MHP のアプリケーション名が設定されます。

  アプリケーション名は，MHP から送信されたメッセージの場合に設定されます。MHP 以外から送信された場合は，ヌル文字が設定されます。

- service_name

  異常が発生した MHP のアプリケーション名に対応するサービス名が設定されます。

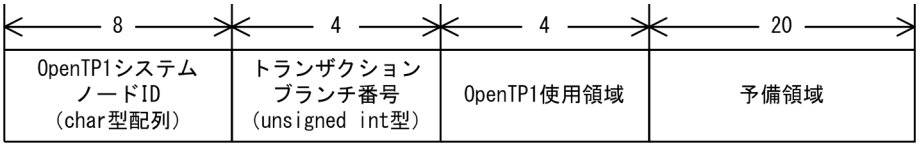
- serv_grp_name

  異常が発生した MHP のサービスが属するサービスグループ名が設定されます。

- bid

  トランザクションブランチ ID が次の形式で設定されます。

（単位：バイト）



(5) ERREVTA

(a) 形式

```
struct dc_mcf_evta_type {
  struct dc_mcf_evtheader  evtheader ;
                             …MCFイベント共通ヘッダ
  char reserve01[12] ;       …予備
  char map_name[10] ;        …MCF使用領域
  char reserve03[2] ;        …予備
  char ap_name[10] ;         …アプリケーション名
                              （正常終了したメッセージの
                               アプリケーション名）
  char reserve04[2] ;        …予備
  char reserve05[32] ;       …予備
  char reserve06[32] ;       …予備
  DCLONG user_leng ;         …他プロトコルの場合の使用領域
  char user_data[16] ;       …他プロトコルの場合の使用領域
  char reserve07[16] ;       …予備
};
```

(b) MCF イベントとして設定される項目

- ap_name

  正常終了したメッセージのアプリケーション名が設定されます。

  アプリケーション名は，MHP から送信されたメッセージの場合に設定されます。MHP 以外から送信された場合は，ヌル文字が設定されます。

(6) CERREVT

(a) 形式

```
typedef struct {
  struct dc_mcf_evtheader  header ;
                             …MCFイベント共通ヘッダ
  DCLONG err_fact ;          …障害要因コード（4バイト）
  DCLONG err_reason1 ;       …理由コード1（4バイト）
  DCLONG err_reason2 ;       …理由コード2（4バイト）
  DCLONG err_rcv_action ;    …MCF使用領域
  char rname[16] ;           …MCF使用領域
  char reserve1[26] ;        …予備
} dcmoum_cerrevt ;
```

(b) MCF イベントとして設定される項目

- err_fact

  CERREVT の障害要因コードが，次に示す値で設定されます。

  $(00000030)_{16}$

  コネクション障害発生

(00000031)$_{16}$

論理端末障害発生

- err_reason1，err_reason2

  CERREVT の理由コードが設定されます。「付録 H　理由コード一覧」を参照してください。

(7)　COPNEVT，CCLSEVT

(a)　形式

```
typedef struct {
  struct dc_mcf_evtheader  header ;
                             …MCFイベント共通ヘッダ
  DCLONG notice_fact ;       …通知要因コード
  char reserve1[12] ;        …予備
  char rname[16] ;           …MCF使用領域
  char reserve2[26] ;        …予備
} dcmoum_statevt ;
```

(b)　MCF イベントとして設定される項目

- notice_fact

  COPNEVT の場合，通知された要因が次に示す値で設定されます。

  (00000030)$_{16}$

  コネクション確立

  (00000031)$_{16}$

  論理端末閉塞解除

5.2.4　MCF イベント情報の形式（COBOL 言語）

COBOL 言語の場合はセグメントの並びとして渡されます。COBOL 言語の UAP の場合，MCF イベント情報の内容を以降の表に示します。

表 5-2　COBOL 言語の MCF イベント情報の内容（ERREVT1）

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。

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
エラーイベントコード	7	3	3	−	−
	10	6	2	英数字	ERREVT1 を示す'1△'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です。
予備	20	16	20	−	−
アプリケーション名	40	36	8	英数字	次に示すどれかが設定されます。 • 形式不正となったアプリケーション名 • 定義されていない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。
予備	80	76	16	−	−
メッセージが入力された日付	96	92	8	外部 10 進数字	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 進数字	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。
予備	112	108	16	−	−

（凡例）

−：該当しません。または，使用されません。

表 5-3　COBOL 言語の MCF イベント情報の内容（ERREVT2）

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	−	−

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 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	ERREVT2 を示す'2△'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です。次に示す場合は，'*' が設定されます。 • SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合 • 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合
予備	20	16	20	−	−
アプリケーション名	40	36	8	英数字	エラーになった UAP のアプリケーション名です。
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。 • SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合 • 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合
予備	80	76	16	−	−
メッセージが入力された日付	96	92	8	外部 10 進数字	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 進数字	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。

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
理由コード※	112	108	4	外部 10 進数字	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。または，使用されません。

注※

理由コードの詳細については，「付録 H　理由コード一覧」を参照してください。

表 5-4　COBOL 言語の MCF イベント情報の内容（ERREVT3）

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	ERREVT3 を示す'3△'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です。次に示す場合は，'*' が設定されます。 • SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合 • 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合
予備	20	16	20	－	－
予備	40	36	8	－	－
マップ名	48	44	8	－	MCF が使用します。
アプリケーション名	56	52	8	英数字	異常が発生したメッセージのアプリケーション名です。
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。

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
コネクション名	72	68	8	英数字	• SPP からアプリケーション起動機能で起動した MHP で，障害が発生した場合 • 上記の障害が発生したあとに，MCF イベントとして起動した MHP からさらにアプリケーション起動機能で起動した MHP で，障害が発生した場合
予備	80	76	16	−	−
メッセージが入力された日付	96	92	8	外部 10 進数字	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 進数字	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。
予備	112	108	16	−	−
サービス名	128	124	31	英数字	異常が発生した UAP のアプリケーション名に対応するサービス名です。
予備	159	155	1	−	−
サービスグループ名	160	156	31	英数字	異常が発生した UAP のサービスグループ名です。
予備	191	187	1	−	−
トランザクションブランチ ID（BID）	192	188	36	英数字	異常が発生したトランザクションの BID です。 トランザクションブランチ ID の形式については，表 5-5 を参照してください。
予備	228	224	28	−	−

（凡例）

−：該当しません。または，使用されません。

表 5-5　トランザクションブランチ ID の形式

項目	位置（バイト）	長さ（バイト）	属性
OpenTP1 システムノード ID	0	8	英数字
トランザクションブランチ番号	8	4	2 進数字

項目	位置（バイト）	長さ（バイト）	属性
OpenTP1 使用領域	12	4	−
予備	16	20	−

表 5-6　COBOL 言語の MCF イベント情報の内容（ERREVTA）

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	ERREVTA を示す'A△'が設定されます。
出力先論理端末名称	12	8	8	英数字	メッセージを出力する論理端末名称です。
予備	20	16	20	−	−
予備	40	36	8	−	−
マップ名	48	44	8	−	MCF が使用します。
アプリケーション名	56	52	8	英数字	正常終了したメッセージのアプリケーション名です。 MHP から送信されたメッセージの場合設定されます。MHP 以外から送信された場合は空白が設定されます。
予備	64	60	8	−	−
予備	72	68	8	−	−
予備	80	76	16	−	−
メッセージが入力された日付	96	92	8	外部 10 進数字	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 進数字	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。

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
予備	112	108	16	−	−
予備	128	124	31	−	−
予備	159	155	1	−	−
予備	160	156	31	−	−
予備	191	187	1	−	−
予備	192	188	36	−	−
予備	228	224	28	−	−

（凡例）

−：該当しません。または，使用されません。

表 5-7　COBOL 言語の MCF イベント情報の内容（CERREVT）

項目	位置（バイト）	長さ（バイト）	属性	内容
イベントコード	0	8	英数字	イベントコード「CERREVT」が設定されます。
入力元論理端末名称	8	8	英数字	障害の発生した論理端末名称が設定されます。コネクション障害時は無効です。
予備	16	8	−	−
入力元コネクション名	24	8	英数字	コネクション名が設定されます。
メッセージ入力日付	32	8	外部 10 進数字	CERREVT を入力した日付です。
メッセージ入力時刻	40	8	外部 10 進数字	CERREVT を入力した時刻です。
障害要因コード	48	4	2 進数字	障害要因が設定されます。 $(00000030)_{16}$：コネクション障害 $(00000031)_{16}$：論理端末障害
理由コード 1※	52	4	2 進数字	理由コード 1 が設定されます。
理由コード 2※	56	4	2 進数字	理由コード 2 が設定されます。
予備	60	48	−	−

（凡例）

−：該当しません。または，使用されません。

注

相手システムから UERR を受信した場合で，UERR の利用者情報が設定されていたとき，利用者情報を CERREVT の第 2 セグメントに設定します。

注※

理由コード 1，および理由コード 2 は，「付録 H　理由コード一覧」を参照してください。

表 5-8　COBOL 言語の MCF イベント情報の内容（COPNEVT，CCLSEVT）

項目	位置（バイト）	長さ（バイト）	属性	内容
イベントコード	0	8	英数字	イベントコード「COPNEVT」，または「CCLSEVT」が設定されます。
入力元論理端末名称	8	8	英数字	論理端末の閉塞解除を通知する COPNEVT の場合は，閉塞解除した論理端末名称が設定されます。 コネクション確立を通知する COPNEVT および CCLSEVT では無効です。
予備	16	8	－	－
入力元コネクション名	24	8	英数字	コネクション名が設定されます。
メッセージ入力日付	32	8	外部 10 進数字	COPNEVT，CCLSEVT を入力した日付です。
メッセージ入力時刻	40	8	外部 10 進数字	COPNEVT，CCLSEVT を入力した時刻です。
通知要因	48	4	2 進数字	COPNEVT の場合に，イベントが通知された要因が設定されます。 $(00000030)_{16}$：コネクション確立 $(00000031)_{16}$：論理端末閉塞解除
予備	52	56	－	－

（凡例）

－：該当しません。または，使用されません。

6 システム定義

この章では，OSAS/UA プロトコルを使用するために必要な，OpenTP1 のシステム定義の中での TP1/NET/User Agent 固有のシステム定義について説明します。また，自システム内にある通信管理プログラムと関連づける内容，相手システムの通信定義と関連づける内容，およびシステム定義例について説明します。

TP1/NET/User Agent の定義の概要

TP1/NET/User Agent のシステム定義は，OpenTP1 のネットワークコミュニケーション定義の中で定義します。また，自システムの通信管理プログラムや相手システムの通信定義と，システム定義内容を関連づける必要があります。

OpenTP1 のネットワークコミュニケーション定義の中での定義

OpenTP1 のネットワークコミュニケーション定義のうち，TP1/NET/User Agent に固有の定義について説明します。

使用する定義ファイル

MCF および TP1/NET/User Agent を起動するには，定義ファイルに環境情報を設定する必要があります。MCF で使用する定義ファイルを次の表に示します。

表 6-1　MCF で使用する定義ファイル

定義の種類	定義のソースファイル	定義の内容
MCF マネジャ定義	MCF マネジャ定義ソースファイル	MCF 全体の実行環境
MCF 通信構成定義	共通定義ソースファイル	プロトコルごとの実行環境
	プロトコル固有定義ソースファイル	
MCF アプリケーション定義	MCF アプリケーション定義ソースファイル	アプリケーションの属性

定義のソースファイルは，定義コマンド，オプション，およびオペランドを指定して作成します。それらの中には，プロトコルで共通のものと，プロトコルに固有のものがあります。次の定義には，TP1/NET/User Agent に固有の定義があります。

- MCF マネジャ定義
- MCF 通信構成定義

この章では，TP1/NET/User Agent に固有の定義コマンド，オプション，およびオペランドについて説明します。プロトコル共通の定義については，マニュアル「OpenTP1 システム定義」を参照してください。ただし，mcftbuf（バッファグループ定義）の length，count オペランドの指定値については，「mcftalccn（コネクション定義の開始）」の注意事項に記載してあります。

TP1/NET/User Agent の組み込み時に必要なファイル

次に示すファイルは，TP1/NET/User Agent を OpenTP1 システムに組み込むときに必要なファイルです。

- システムサービス情報定義ファイル
- システムサービス共通情報定義ファイル

- MCF 定義オブジェクトファイル

この章では，システムサービス情報定義ファイルとシステムサービス共通情報定義ファイルの記述内容，およびMCF 定義オブジェクトファイルを生成するユティリティの起動コマンドについて説明します。TP1/NET/User Agent を組み込む方法については，「8. 組み込み方法」を参照してください。

通信定義の内容の関連づけ

OSAS/UA プロトコルを使用して相手システムと通信するためには，TP1/NET/User Agent のシステム定義内容を自システムの通信管理プログラムや，相手システムの通信定義と関連づける必要があります。

この章では，自システムの通信管理プログラム（XNF/LS，または XNF/AS）と関連づける内容を示します。また，相手システム（TMS-4V/SP または XDM/DCCM3）のネットワーク定義と関連づける内容を示します。

TP1/NET/User Agent 固有のシステム定義の種類

OpenTP1 のネットワークコミュニケーション定義のうち，TP1/NET/User Agent に固有の定義の一覧を次の表に示します。

表 6-2　TP1/NET/User Agent 固有の定義の一覧

定義名		コマンド	オプション・オペランド		定義内容	指定値((値範囲))《省略時解釈値》
MCF マネジャ定義		mcfmuap	-t	sndrcvtim	同期型送受信監視時間	符号なし整数 ((0～65535))《0》(単位：秒)
MCF 通信構成定義	共通定義	mcftcomn	-x	termrls	MCF 終了時のコネクションの解放方法	nomal | error | 《no》
	プロトコル固有定義	mcftalccn (コネクション定義の開始) 指定数：1～512	-c	－	コネクション ID	1～8 文字の識別子
			-p	－	プロトコルの種別	ua
			-n	－	自システムの PSAP アドレス	1～142 けたの 16 進数字
			-g	sndbuf	メッセージ送信用バッファグループ番号	符号なし整数((1～512))
				rcvbuf	メッセージ受信用バッファグループ番号	符号なし整数((1～512))
			-e	msgbuf	メッセージ編集用バッファグループ番号	符号なし整数((1～512))
				count	メッセージ編集用バッファ数	符号なし整数((1～131070))
			-m	mode	使用する通信管理	tli | xnfas
			-i	－	コネクションの確立方法	auto | 《manual》
			-o	－	OSAS/UA プロトコル種別	old | new
			-u	－	UINT を重畳する MPDU	ht | ws
			-w	nomltim	プロトコル監視時間	符号なし整数((0，10～65535))《60》(単位：秒)
				usertim	問い合わせ監視時間	符号なし整数((0，10～65535))《60》(単位：秒)

定義名		コマンド	オプション・オペランド		定義内容	指定値((値範囲))《省略時解釈値》
MCF 通信構成定義	プロトコル固有定義	mcftalccn（コネクション定義の開始） 指定数：1～512	-b	bretry	コネクション確立障害時の確立再試行するかどうかを指定	《yes》｜no
				bretrycnt	コネクション確立障害時の確立再試行回数	符号なし整数((0～65535))《0》（単位：回）
				bretryint	コネクション確立障害時の確立再試行間隔	符号なし整数((0～2550))《60》（単位：秒）
			-k	－	すべての UA の開局モード	《together》｜each
			-d	－	端末識別子の文字数	符号なし整数((0，5～125))《5》
			-y	－	制御 UA の端末識別子	5～125 文字の英数字記号または 10～250 けた（偶数けた）の 16 進数字
			-q	－	相手システムの PSAP アドレス	1～186 けたの 16 進数字
			-z	slot	仮想スロット番号	符号なし整数((0～65535))
			-l	－	TL クラス	0
			-f	leopnevt	論理端末の閉塞解除時に，状態通知イベント（COPNEVT）を起動するかどうかを指定	use｜《nouse》
				negomsg	システム構成の縮退を通知するメッセージを出力するかどうかを指定	use｜《nouse》
			-T	device	デバイス名称	/と-を含む 1～64 文字の識別子
		mcftalcle（論理端末定義） 指定数：1～2048	-l	－	論理端末名称	1～8 文字の識別子
			-t	－	論理端末の端末タイプ	reply｜request｜send｜receive
			-m	mmsgcnt	メモリ出力メッセージ最大格納数	符号なし整数((0～65535))《0》
				dmsgcnt	ディスク出力メッセージ最大格納数	符号なし整数((0～65535))《0》
			-k	quekind	出力メッセージの割り当て先	《memory》｜disk

定義名		コマンド	オプション・オペランド		定義内容	指定値((値範囲))《省略時解釈値》
MCF 通信構成定義	プロトコル固有定義	mcftalcle（論理端末定義） 指定数：1～2048	-k	quegrpid	キューグループ ID	1～8 文字の識別子
			-o	aj	メッセージ送信完了ジャーナルを取得するかどうかを指定	《yes》｜no
		mcftalcua（UA 定義） 指定数：1～2048	-u	－	UA 番号	符号なし整数((1～2048))
			-y	－	一般 UA の端末識別子	5～125 文字の英数字記号または 10～250 けた（偶数けた）の 16 進数字
		mcftalced（コネクション定義の終了） 指定数：mcftalccn と同数	－	－	コネクション定義の終了	－

（凡例）

－：該当する内容がないことを表します。

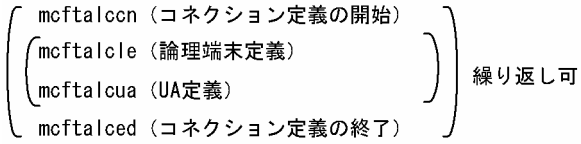
注

TP1/NET/User Agent に固有の定義だけ記載してあります。このほかにも，プロトコルで共通の定義コマンド，オプション，オペランドがあります。それらについては，マニュアル「OpenTP1 システム定義」を参照してください。

定義の指定順序

TP1/NET/User Agent のプロトコル固有定義コマンドの指定順序を次の図に示します。

図 6-1　TP1/NET/User Agent のプロトコル固有定義コマンドの指定順序

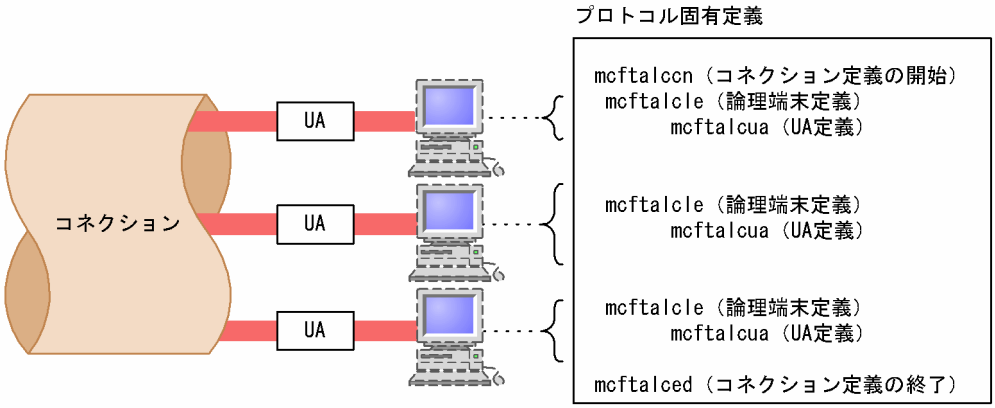


TP1/NET/User Agent のプロトコル固有定義の指定には次の条件があります。

- mcftalcua コマンドは省略できません。
- mcftalccn コマンドと mcftalcle コマンドとの関係は 1 対 n です。
- mcftalcle コマンドと mcftalcua コマンドの関係は 1 対 1 です。

コネクション 1 個に対し，論理端末を 3 個定義した場合のプロトコル固有定義の例を次の図に示します。

図 6-2　プロトコル固有定義の例



mcfmuap（UAP 共通定義）

形式

```
mcfmuap          :
         〔-t ”〔sndrcvtim=同期型送受信監視時間〕”〕
                 :
```

機能

同期型のメッセージの送受信の監視時間を定義します。

オプション

この定義コマンドには，ほかにもオプションがあります。詳細については，マニュアル「OpenTP1 システム定義」を参照してください。

●-t

（オペランド）

sndrcvtim=同期型送受信監視時間　　～〈符号なし整数〉((0～65535))《0》（単位：秒）

同期型のメッセージの送受信の仕掛り開始（sendrecv（EMI）発行）から仕掛り終了（sendrecv 終了）までの限界監視時間を指定します。このオペランドでは，相手システムからの応答時間を監視します。0 を指定した場合，送受信時間の監視はしません。

このオペランドは，問い合わせ監視時間（コネクション定義（mcftalccn -w）の usertim オペランドで指定）より，60 秒多い値を指定してください。

mcftalccn（コネクション定義の開始）

形式

```
mcftalccn  -c コネクションID
           -p ua
           -n x'自システムのPSAPアドレス'
           -g "sndbuf=メッセージ送信用バッファグループ番号
               rcvbuf=メッセージ受信用バッファグループ番号"
          〔-e "msgbuf=メッセージ編集用バッファグループ番号
               count=メッセージ編集用バッファ数"〕
           -m "mode=tli | xnfas"
          〔-i auto | manual〕
           -o old | new
           -u ht | ws
          〔-w "〔nomltim=プロトコル監視時間〕
                〔usertim=問い合わせ監視時間〕"〕
          〔-b "〔bretry=yes | no〕
                〔bretrycnt=コネクション確立障害時の確立再試行回数〕
                〔bretryint=コネクション確立障害時の確立再試行間隔〕"〕
          〔-k together | each〕
          〔-d 端末識別子の文字数〕
          〔-y e'制御UAの端末識別子' | x'制御UAの端末識別子'〕
           -q x'相手システムのPSAPアドレス'
           -z "slot=仮想スロット番号"
          〔-l 0〕
          〔-f "〔leopnevt=use | nouse〕
                〔negomsg=use | nouse〕"〕
          〔-T "device=デバイス名称"〕
```

機能

コネクションに関する環境を定義します。

オプション

●-c コネクション ID　～〈1～8 文字の識別子〉

OpenTP1 システム内で，一意となるコネクション ID を指定します。

●-p ua

プロトコルの種別を指定します。

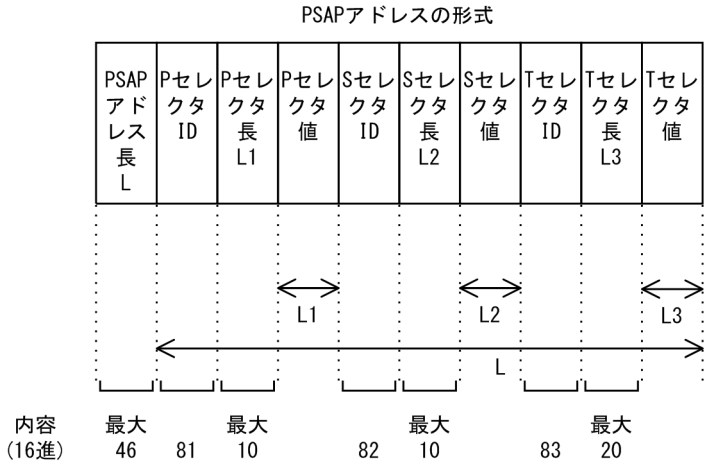
ua

OSAS/UA プロトコル

●-n x'自システムの PSAP アドレス'　～〈1～142 けたの 16 進数字〉

自システムの PSAP アドレスを指定します。x は，16 進数形式で指定することを意味します。

自システムの PSAP アドレスの形式を次に示します。



ほかのコネクションが使用する PSAP アドレスと重複しない PSAP アドレスを指定してください。また，XNF/LS を使用する場合，ほかのプロセスが使用する T セレクタ値と重複しない T セレクタ値を指定してください。

●-g

(オペランド)

sndbuf=メッセージ送信用バッファグループ番号　～〈符号なし整数〉((1～512))

メッセージ送信用バッファのグループ番号を指定します。

mcftbuf コマンドの-g オプションの groupno オペランドに指定されているバッファグループ番号を指定してください。

-n オプションで指定する自システムの PSAP アドレスの値が，ほかの mcftalccn 定義コマンドの値と同じ場合，このオペランドもほかの mcftalccn 定義コマンドと同じ値を指定してください。

rcvbuf=メッセージ受信用バッファグループ番号　～〈符号なし整数〉((1～512))

メッセージ受信用バッファのグループ番号を指定します。

mcftbuf コマンドの-g オプションの groupno オペランドに指定されているバッファグループ番号を指定してください。

-n オプションで指定する自システムの PSAP アドレスの値が，ほかの mcftalccn 定義コマンドの値と同じ場合，このオペランドもほかの mcftalccn 定義コマンドと同じ値を指定してください。

●-e

(オペランド)

msgbuf=メッセージ編集用バッファグループ番号　～〈符号なし整数〉((1～512))

入力，出力メッセージ編集 UOC 呼び出し時に，メッセージ編集用として使用するバッファグループ番号を指定します。

このオペランドを省略した場合は，メッセージ編集用バッファは確保されません。メッセージ編集用バッファグループ番号は，mcftbuf コマンドの-g オプションの groupno オペランドで指定するバッファグループ番号を指定してください。

count=メッセージ編集用バッファ数　～〈符号なし整数〉((1～131070))

入力，出力編集メッセージ UOC 呼び出し時に，メッセージ編集用として使用するバッファの数を指定します。

msgbuf オペランドで指定するメッセージ編集用バッファグループ番号に対応する mcftbuf コマンドの-g オプションの count，および extend オペランドで指定するバッファ数の中から，メッセージ編集用に使用するバッファ数を指定してください。

メッセージ編集用バッファグループ番号に対応する mcftbuf の count オペランドおよび extend オペランドには，すべての論理端末で入力メッセージ編集 UOC および出力メッセージ編集 UOC が，同時に動作した場合を考慮した面数を指定してください。

また，このオペランドの指定は mcftbuf コマンドの-g オプションの count，および extend オペランドで指定されたバッファ数の合計値を超える指定はできません。

msgbuf オペランドを省略した場合は，このオペランドの指定は無効です。

msgbuf オペランドを指定した場合，このオペランドを省略できません。省略した場合，定義オブジェクト生成時に KFCA11519-E メッセージを出力してエラーとなります。

●-m

(オペランド)

mode=tli | xnfas

使用する通信管理を指定します。

tli

XNF/LS を使用します。

xnfas

XNF/AS を使用します。

このオペランドは必ず指定してください。

●-i auto | manual　～《manual》

OpenTP1 システム開始時および再開始時にコネクションを自動的に確立するかどうかを指定します。

auto

OpenTP1 システム開始時および再開始時にコネクションを自動的に確立します。

manual

MCF 起動後，コネクションを確立します。コネクションの確立は，運用コマンド（mcftactcn）の入力，または API（dc_mcf_tactcn 関数もしくは CBLDCMCF('TACTCN△△')）の発行で行います。

●-o old | new

OSAS/UA プロトコルの種別を指定します。

old

旧 OSAS/UA プロトコル

new

新 OSAS/UA プロトコル

TMS-4V/SP のシステムと接続する場合，old を指定してください。また，XDM/DCCM3 のシステムと接続する場合，new を指定してください。

●-u ht | ws

初期設定メッセージ（UINT）を重畳する NIF/OSI 層の MPDU を指定します。

ht

UINT を Invoke（問い合わせ）MPDU に重畳します。

ws

UINT を Invoke（初期設定要求）MPDU および ACSE 層の初期設定要求に重畳します。

●-w

（オペランド）

nomltim=プロトコル監視時間　　～〈符号なし整数〉((0，10～65535))《60》（単位：秒）

usertim オペランド以外の OSAS/UA プロトコルで規定する監視時間を指定します（初期設定監視時間（T12），個別開局監視時間（T11），後続送信指示監視時間（T7），後続送信直後の監視時間（T9），例外報告監視時間（T11））。

0 を指定した場合は，時間監視をしません。

注意事項

監視時間の精度は秒単位です。また，タイマ定義（mcfttim -t）の btim オペランドで指定する時間の間隔でタイムアウトが発生したかどうかを監視しています。このため，このオペランドで指定した監視時間と実際にタイムアウトを検出する時間には秒単位の誤差が生じます。そのため，タイミングによっては，指定した監視時間よりも短い時間でタイムアウトすることがあります。

usertim=問い合わせ監視時間　　～〈符号なし整数〉((0，10～65535))《60》（単位：秒）

問い合わせメッセージ送信後，応答メッセージを受信するまでの監視時間を指定します。

0 を指定した場合は，時間監視をしません。また，このオペランドは，同期型送受信監視時間（コネクション定義（mcfmuap -t）の sndrcvtim オペランドで指定）より少ない値を必ず指定してください。

注意事項

監視時間の精度は秒単位です。また，タイマ定義（mcfttim -t）の btim オペランドで指定する時間の間隔でタイムアウトが発生したかどうかを監視しています。このため，このオペランドで指定した監視時間と実際にタイムアウトを検出する時間には秒単位の誤差が生じます。そのため，タイミングによっては，指定した監視時間よりも短い時間でタイムアウトすることがあります。

●-b

（オペランド）

bretry=yes | no　～《yes》

コネクション確立時に障害が発生した場合，コネクション確立再試行をするかどうかを指定します。

yes

コネクション確立再試行をします。

no

コネクション確立再試行をしません。

bretrycnt=コネクション確立障害時の確立再試行回数　～〈符号なし整数〉((0～65535))《0》(単位：回)

コネクションの確立再試行をする場合の確立再試行回数を指定します。

このオペランドを省略した場合，または 0 を指定した場合は，無限に確立再試行を繰り返します。

bretry オペランドで no を指定した場合，bretrycnt オペランドの指定は無効になります。

通信管理から再試行不可能な障害が通知された場合，または相手システムから確立要求に対する拒否応答を受けた場合は，再試行を中止します。

bretryint=コネクション確立障害時の確立再試行間隔　～〈符号なし整数〉((0～2550))《60》(単位：秒)

コネクションの確立再試行をする場合の確立再試行間隔を指定します。0 を指定した場合，障害が発生するたびにコネクションの確立再試行をします。

bretry オペランドで no を指定した場合，bretryint オペランドの指定は無効になります。

注意事項

再試行間隔の精度は秒単位です。また，タイマ定義（mcfttim -t）の btim オペランドで指定する時間の間隔で再試行するかどうかをチェックします。このため，このオペランドで指定した再試行間隔と実際に再試行する時間には秒単位の誤差が生じます。

●-k together | each　～《together》

コネクション確立時，このコネクションのすべての UA の開局モードを指定します。

together

コネクション確立と同時にすべての UA を開局します（一括開局）。

each

すべての UA を閉局した状態でコネクションを確立します（個別開局）。

●-d 端末識別子の文字数　～〈符号なし整数〉((0，5～125))《5》

制御 UA および一般 UA の端末識別子の文字数を指定します。

旧 OSAS/UA プロトコルを使用する場合（コネクション定義（mcftalccn）の-o オプションで old を指定），このオプションを省略するか，5 以上の値を指定してください。

新 OSAS/UA プロトコルを使用する場合（コネクション定義（mcftalccn）の-o オプションで new を指定），0 を指定してください。

●-y e'制御 UA の端末識別子'｜ x'制御 UA の端末識別子'

制御 UA の端末識別子を指定します。

旧 OSAS/UA プロトコルを使用する場合，このオプションを必ず指定してください。新 OSAS/UA プロトコルを使用する場合，このオプションを省略してください。

この-y オプションで指定する制御 UA の端末識別子の文字数は-d オプションで指定した文字数と一致しなければなりません。

なお，制御 UA の端末識別子は，ほかの mcftalccn 定義コマンドで指定する制御 UA の端末識別子と重複して指定できません。

e　～〈5～125 文字の英数字記号〉

EBCDIC コードに変換することを意味します。

x　～〈10～250 けた（偶数けた）の 16 進数字〉

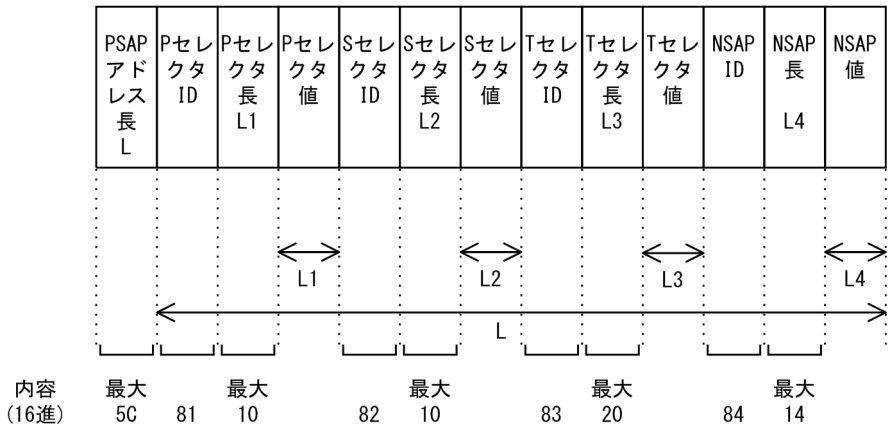
16 進数形式で指定することを意味します。

●-q x'相手システムの PSAP アドレス'　～〈1～186 けたの 16 進数字〉

相手システムの PSAP アドレスを指定します。x は，16 進数形式で指定することを意味します。

相手システムの PSAP アドレスの形式を次に示します。なお，NSAP 値の形式については，マニュアル「通信管理 XNF/AS NSAP アドレス概説編」，またはマニュアル「通信管理 XNF/LS 使用の手引」を参照してください。

PSAPアドレスの形式



-n オプションで指定する自システムの PSAP アドレスと，-q オプションで指定する相手システムの PSAP アドレスの組み合わせは，ほかのコネクションが使用する組み合わせと重複しないでください。

●-z

(オペランド)

slot=仮想スロット番号　～〈符号なし整数〉((0～65535))

仮想スロット番号を指定します。

通信管理の定義で指定した仮想スロット番号を指定してください。

●-l 0

使用する TL クラスを指定します。

0

TL クラス 0

XNF/AS を使用する場合，このオプションを必ず指定してください。XNF/LS を使用する場合，このオプションを省略してください。

●-f

（オペランド）

leopnevt=use | nouse　～《nouse》

論理端末が閉塞解除したときに，状態通知イベント（COPNEVT）を起動するかどうかを指定します。

このオプションの設定値にかかわらず，コネクション確立時の自動開局によって閉塞解除した場合は，COPNEVT は起動しません。

use

論理端末の閉塞解除時に COPNEVT を起動します。

nouse

論理端末の閉塞解除時に COPNEVT を起動しません。

negomsg=use | nouse　～《nouse》

相手システムの構成と自システムの構成が不一致（相手システムの UA ＜自システムの UA）であった場合に，相手システムの構成に合わせて自システムの構成を縮退することを通知するメッセージ（KFCA13257-W）を出力するかどうかを指定します。

use

自システムの構成縮退を通知するメッセージを出力します。

nouse

自システムの構成縮退を通知するメッセージを出力しません。

●-T

（オペランド）

device=デバイス名称　～〈/と-を含む 1～64 文字の識別子〉

通信管理プログラムが使用するデバイス名称を指定します。

XNF/LS を使用する場合，/dev/xnfw/cots を指定してください。

XNF/AS を使用する場合，このオペランドは指定できません。

注意事項

-g オプションおよび-e オプションで指定するバッファグループ番号に対応するバッファグループ定義の mcftbuf コマンドでは，1 コネクション単位に次の表に示す資源が必要です。バッファグループ定義については，マニュアル「OpenTP1 システム定義」を参照してください。

バッファ種別	length オペランド	count オペランド
sndbuf	• 最大セグメント長 または • （端末識別子の文字数※＋ 14）×UA 数の最大値　以上	（最大セグメント分割数＋ 1）×UA 数　以上
rcvbuf	sndbuf と同じ	同上
msgbuf	最大セグメント長以上	同上

注※

mcftalccn コマンドの-d オプションで指定します。

なお，mcftbuf コマンドの length オペランドでは，ユーザレベルでの最大セグメント長を指定します。相手システムによって，相手システムのバッファ資源として，次に示す長さで定義する場合があるので注意してください。

（最大セグメント長）＋（プロトコルヘッダ長）

mcftalced（コネクション定義の終了）

形式

```
mcftalced
```

機能

一つのコネクション定義の終了を示します。

オプション

ありません。

mcftalcle（論理端末定義）

形式

```
mcftalcle -l 論理端末名称
          -t reply | request | send | receive
         〔-m ”〔mmsgcnt=メモリ出力メッセージ最大格納数〕
              〔dmsgcnt=ディスク出力メッセージ最大格納数〕”〕
         〔-k ”〔quekind=memory | disk〕
              〔quegrpid=キューグループID〕”〕
         〔-o ”〔aj=yes | no〕”〕
```

機能

論理端末に関する環境を定義します。

オプション

●-l 論理端末名称　　～〈1～8 文字の識別子〉

OpenTP1 システム内で，一意となる論理端末名称を指定します。

●-t reply | request | send | receive

この論理端末の端末タイプを指定します。

reply
　応答型論理端末

request
　問い合わせ型論理端末

send
　送信型論理端末

receive
　受信型論理端末

●-m

（オペランド）

mmsgcnt=メモリ出力メッセージ最大格納数　　～〈符号なし整数〉((0～65535))《0》
　メモリキューで待ち合わせをする出力メッセージの最大格納数を指定します。
　出力メッセージの待ち合わせ数が指定した最大数になると，それ以後 UAP からの送信要求（dc_mcf_send 関数または SEND 文）はエラーリターン（リターン値 DCMCFRTN_71003 またはステータスコード 71003）となります。

0 を指定した場合，または省略した場合，メモリキューで待ち合わせをする出力メッセージの数は指定可能な最大数（65535）となります。ただし，実際に待ち合わせをできる出力メッセージ数は動的共用メモリの容量に依存します。

dmsgcnt=ディスク出力メッセージ最大格納数　～〈符号なし整数〉((0～65535))《0》

ディスクキューで待ち合わせをする出力メッセージの最大格納数を指定します。

出力メッセージの待ち合わせ数が指定した最大数になると，それ以後 UAP からの送信要求（dc_mcf_send 関数または SEND 文）はエラーリターン（リターン値 DCMCFRTN_71003 またはステータスコード 71003）となります。

0 を指定した場合，または省略した場合，ディスクキューで待ち合わせをする出力メッセージの数は指定可能な最大数（65535）となります。ただし，実際に待ち合わせをできる出力メッセージ数はメッセージキューファイルの容量に依存します。

●-k

(オペランド)

quekind=memory | disk　～《memory》

出力メッセージの割り当て先（メモリキューまたはディスクキュー）を指定します。

memory

メモリキューだけに割り当てます。

disk

ディスクキューおよびメモリキューに割り当てます。

disk を指定した場合，必ず quegrpid オペランドを指定してください。

quegrpid=キューグループ ID　～〈1～8 文字の識別子〉

ディスクキューで待ち合わせをする出力メッセージに使用するキューグループ ID を指定します。MCF マネジャ定義の，mcfmqgid コマンドで指定するキューグループ ID（キューグループ種別は otq）のどれかを指定してください。

この quegrpid オペランドは，quekind オペランドで disk を指定した場合だけ指定します。

●-o

(オペランド)

aj=yes | no　～《yes》

メッセージ送信が完了した場合に，メッセージ送信完了ジャーナル（AJ）を取得するかどうかを指定します。

yes

メッセージ送信完了ジャーナルを取得します。

no

メッセージ送信完了ジャーナルを取得しません。

mcftalcua（UA 定義）

形式

```
mcftalcua -u UA番号
          〔-y e'一般UAの端末識別子'｜x'一般UAの端末識別子'〕
```

機能

一般 UA に関する環境を定義します。

オプション

●-u UA 番号　　～〈符号なし整数〉((1～2048))

UA 番号を指定します。

UA 番号には，一つのコネクション（mcftalccn 定義）に対して 1 から始まる昇順でかつ連続した番号を指定してください。

●-y e'一般 UA の端末識別子'｜ x'一般 UA の端末識別子'

一般 UA の端末識別子を指定します。

旧 OSAS/UA プロトコルを使用する場合，このオプションを必ず指定してください。新 OSAS/UA プロトコルを使用する場合，このオプションを省略してください。

このオプションで指定する一般 UA の端末識別子の文字数は，コネクション定義（mcftalccn -d）で指定した端末識別子の文字数と一致しなければなりません。

なお，一般 UA の端末識別子は，ほかの mcftalcua 定義コマンドで指定する一般 UA の端末識別子と重複して指定できません。

e　　～〈5～125 文字の英数字記号〉

EBCDIC コードに変換することを意味します。

x　　～〈10～250 けた（偶数けた）の 16 進数字〉

16 進数形式で指定することを意味します。

mcftcomn（MCF 通信構成共通定義）

形式

```
mcftcomn          :
        〔-x ”termrls=nomal｜error｜no”〕
```

機能

MCF 終了時のコネクションの解放方法を定義します。

オプション

この定義コマンドには，ほかにもオプションがあります。詳細については，マニュアル「OpenTP1 システム定義」を参照してください。

●-x

（オペランド）

termrls=nomal ｜ error ｜ no　～《no》

MCF 終了時（正常終了，計画停止 A，計画停止 B）のコネクションの解放方法を指定します。

コネクション確立，解放処理中に MCF 終了要求があった場合，異常解放（UERR 送信）します。ただし，コネクション確立再試行をしている場合に，このオプションで no を指定していると，再試行の処理を継続する場合があります。コネクション確立再試行を行う運用の場合には，このオプションで no 以外を指定してください。

なお，このオプションは TP1/NET/User Agent 固有です。

nomal

正常解放（UEND 送信）

error

異常解放（UERR 送信）

no

解放処理はしません。MCF 通信プロセスの終了で解放します。

システムサービス情報定義

MCF サービスはユーザが作るシステムサービスで，OpenTP1 のシステムサービスと同じ位置づけになります。

システムサービス情報定義では，MCF 通信サービスを起動するための環境を定義します。ユーザが MCF サービスを作成するときに定義する必要があります。

システムサービス情報定義は，テキストエディタを使用して作成します。

システムサービス情報定義の完全パス名を次に示します。

```
$DCDIR/lib/sysconf/定義ファイル名
```

定義ファイル名には，システムサービス情報定義の module オペランドで指定する実行形式プログラム名を指定します。この定義ファイル名を MCF マネジャ定義の mcfmcname コマンドに指定します。

形式

set 形式

```
 set module=”TP1/NET/User Agentの実行形式プログラム名”
〔set mcf_prf_trace=Y|N〕
```

機能

プロセスサービスが MCF 通信サービスを起動するための環境を定義します。

各 MCF 通信サービスに対して一つ，システムサービス情報定義を作成できます。また，複数の MCF 通信サービスで一つのシステムサービス情報定義を共用することもできます。

説明

set 形式のオペランド

●module="TP1/NET/User Agent の実行形式プログラム名"　～〈1～8 文字の識別子〉

MCF 通信サービスを起動するための実行形式プログラム名を指定します。

MCF 実行形式プログラムには，MCF 通信プロセスのためのものとアプリケーション起動プロセスのためのものがあります。

MCF 実行形式プログラムは，MCF 通信プロセス同士，アプリケーション起動プロセス同士で共有できます。

TP1/NET/User Agent の実行形式プログラム名には，先頭 4 文字が mcfu で始まる最大 8 文字の名称を指定します。

●mcf_prf_trace=Y¦N　～《Y》

MCF 通信サービスごとに，MCF 性能検証用トレース情報を取得するかどうかを指定します。このオペランドの指定値を有効にするには，システムサービス共通情報定義の mcf_prf_trace_level オペランドに 00000001 を指定してください。

Y

MCF 性能検証用トレース情報を取得します。

N

MCF 性能検証用トレース情報を取得しません。

MCF 通信サービスでの MCF 性能検証用トレース情報取得有無とオペランドの指定値の関係を，次の表に示します。

システムサービス共通情報定義 mcf_prf_trace_level オペランドの指定値	システムサービス情報定義 mcf_prf_trace オペランドの指定値	
	Y	N
00000000	取得しない	取得しない
00000001	取得する	取得しない

このオペランドの使用は，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。

システムサービス共通情報定義

TP1/NET/User Agent で定義したシステム構成の内容によっては，OpenTP1 のシステムサービス共通情報定義を指定する必要があります。

システムサービス共通情報定義の完全パス名を次に示します。

```
$DCDIR/lib/sysconf/mcf
```

形式

set 形式

```
  set max_socket_descriptors=ソケット用ファイル記述子の最大数
  set max_open_fds=MCF通信プロセスでアクセスするファイルの最大数
〔set mcf_prf_trace_level=MCF性能検証用トレース情報の取得レベル〕
```

機能

システムサービス共通情報定義では，複数の MCF 通信サービスに共通する情報を定義します。この定義ファイルは，標準値を定義した状態で製品に含まれています。次に示すオペランドについては，必要に応じて，テキストエディタを使用して定義値を変更してください。ほかのオペランドについては，変更しないでください。

説明

set 形式のオペランド

この定義には，ほかにもオペランドがあります。詳細については，マニュアル「OpenTP1 システム定義」を参照してください。

●max_socket_descriptors=ソケット用ファイル記述子の最大数　～〈符号なし整数〉((64～3596))

各 MCF 通信プロセスでソケット用に使用するファイル記述子数の中の最大値を指定します。

OpenTP1 制御下のプロセスでは，システムサーバやユーザサーバとの間で，ソケットを使用した TCP/IP 通信でプロセス間の情報交換をしています。そのため，同時に稼働する UAP プロセスの数などによって，ソケット用のファイル記述子の最大数を変更する必要があります。

各 MCF 通信プロセスまたはアプリケーション起動プロセスが使用するソケット用ファイル記述子の最大数を求める計算式を次に示します。

```
↑（このMCF通信プロセスに対してメッセージ送信要求を行うUAPプロセス数※1
  ＋システムサービスプロセス数※2
  ＋このMCF通信プロセスまたはアプリケーション起動プロセスに対して同時に処理要求を行う運用コマンド数
  ）／0.8↑
```

（凡例）

↑↑：小数点以下を切り上げます。

注※1

アプリケーション起動プロセスに対するアプリケーション起動要求を行う UAP プロセス数も含みます。

注※2

システムサービスプロセス数とは，自 OpenTP1 内のシステムサービスプロセス数です。自 OpenTP1 内のシステムサービスプロセスは，rpcstat コマンドで表示されるサーバ名をカウントすることで求められます。rpcstat コマンドで表示されるサーバ名のうち，マニュアル「OpenTP1 解説」の OpenTP1 のプロセス構造に記載されているシステムサービスプロセスをカウントしてください。

自 OpenTP1 内の各 MCF 通信プロセスおよびアプリケーション起動プロセスごとに計算し，その結果の中で最大値が 64 より大きい場合は，その値を指定します。64 以下の場合は，64 を指定します。

このオペランドの指定値が小さいと，OpenTP1 制御下の他プロセスとのコネクションが設定できなくなるため，プロセスが KFCA00307-E メッセージを出力して異常終了します。

●max_open_fds=MCF 通信プロセスでアクセスするファイルの最大数　～〈符号なし整数〉((500～4032))

各 MCF 通信プロセスでアクセスするファイル数の中の最大値を指定します。

MCF 通信プロセスが行うメッセージの送受信にもファイル記述子が使われます。この数が不足すると，コネクションの確立ができないなどの障害が発生するため，事前に必要となるファイル記述子の数を設定しておく必要があります。

各 MCF 通信プロセスが使用するファイル記述子の最大数を求める計算式を次に示します。

```
（プロトコル制御で使用するファイル記述子数※1）
＋MCFメイン関数でユーザが使用するファイル記述子数
＋30※2
```

注※1

TP1/NET/User Agent の場合，コネクションの総数を 2 倍した値になります。実際に通信を行う論理端末の総数ではありません。

注※2

MCF 通信プロセスが扱う定義ファイルなどの数の最大値です。

自 OpenTP1 内の各 MCF 通信プロセスごとに計算し，その結果の中で最大値が 500 より大きい場合は，その値を指定します。500 以下の場合は，500 を指定します。

指定値を超えてファイルのアクセスが発生した場合，その超過分はソケット用ファイル記述子使用数として扱われます。この場合，「max_socket_descriptors オペランドの指定値－max_open_fds オペランドの指定値の超過分」が，実際のソケット用ファイル記述子の最大数になりますので，注意してください。

max_socket_descriptors オペランドと max_open_fds オペランドには次の条件を満たす値を指定してください。

```
（「max_socket_descriptorsオペランドの指定値」
＋「max_open_fdsオペランドの指定値」）≦4096
```

ただし，TP1/NET/User Agent の MCF 通信プロセスで使用できるファイル記述子の最大数は，適用 OS によって次のように異なります。

OS	1 プロセスで使用できるファイル記述子の最大数
AIX，HP-UX	2048
Linux	1024

TP1/NET/User Agent の MCF 通信プロセスで，max_socket_descriptors オペランドと max_open_fds オペランドの和が 1 プロセスで使用できるファイル記述子の最大数を超えている場合，TP1/NET/User Agent の MCF 通信プロセスで使用できるファイル記述子数は，1 プロセスで使用できるファイル記述子の最大数に強制的に補正されます。

max_socket_descriptors オペランドと max_open_fds オペランドの和が 1 プロセス当たりでオープンできるファイル数の物理限界値（ハードリミット）を超えていたとき，MCF の開始を中断します。

●mcf_prf_trace_level=MCF 性能検証用トレース情報の取得レベル　～((00000000～00000001)) 《00000001》

MCF 性能検証用トレース情報の取得レベルを指定します。MCF 性能検証用トレースを取得する場合は，システム共通定義の prf_trace オペランドに Y を指定するか，または省略してください。

00000000

MCF 性能検証用トレース情報を取得しません。

00000001

MCF 性能検証用トレース情報（イベント ID：0xa000～0xa0ff）を取得します。イベント ID 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。また，TP1/NET/User Agent 固有の出力情報や取得タイミングについては，「付録 F　MCF 性能検証用トレースの取得」を参照してください。

オペランドの指定に誤りがある場合は，OpenTP1 開始処理中に OpenTP1 が異常終了します。

このオペランドの使用は，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。

注意事項

max_socket_descriptors オペランドの指定値と max_open_fds オペランドの指定値の合計は，OS のシステムパラメタで指定する「1 プロセスでオープンできるファイル数」を超えないようにする必要があり

ます。システム定義の変更などによって，上記のオペランドの指定値の合計が増加する場合は，OS のシステムパラメタの指定を変更してください。

MCF 定義オブジェクトの生成

MCF 定義オブジェクト生成ユティリティでは，MCF の定義ファイルの構文のチェックと定義オブジェクトファイルへの変換をします。ここでは，MCF 定義オブジェクト生成ユティリティの起動コマンドについて説明します。

形式

```
mcfosua -i  〔パス名〕入力ファイル名
        -o  〔パス名〕出力オブジェクトファイル名
       〔-r  {no | rep} 〕
```

機能

MCF 通信構成定義の TP1/NET/User Agent のプロトコル固有定義ファイルの構文をチェックし，定義オブジェクトファイルを作成します。

ただし，開始から再開始の間に定義オブジェクトファイルを変更しないでください。変更した場合，再開始時に正常に動作しないことがあるためご注意ください。

TP1/NET/User Agent のプロトコル固有定義オブジェクトファイル以外の生成ユティリティについては，マニュアル「OpenTP1 システム定義」を参照してください。

オプション

●-i 〔パス名〕入力ファイル名　～〈パス名〉〈1～8 文字の識別子〉

定義ソースが格納されているファイル名を指定します。

●-o 〔パス名〕出力オブジェクトファイル名　～〈パス名〉〈1～8 文字の英数字〉

定義オブジェクトを格納するファイル名を指定します。

次に示す条件を満たした名称を指定してください。

- 先頭 3 文字が_mu で始まる最大 8 文字の名称
- 通信サービス定義（mcfmcname -s）の mcfsvname オペランドで指定する MCF 通信サーバ名

●-r {no | rep}　～《no》

定義オブジェクトファイルの出力先に読み取り権限を持つファイルがすでに存在する場合，定義オブジェクトファイルを上書きするかどうかを指定します。

no

定義オブジェクトファイルを上書きしないで，KFCA10332-E メッセージを出力します。

rep

定義オブジェクトファイルを上書きします。

自システムの通信管理プログラムと関連づける内容

TP1/NET/User Agent の定義には，通信管理プログラムと関連づける項目があります。

通信管理プログラムが XNF/LS の場合（mcftalccn -m "mode=tli"）

相手システムとの接続時に使用したい機能によって，必要となる通信管理プログラムは次のとおり異なります。

- OSI 拡張機能を使用する場合
  XNF/LS/BASE
  XNF/LS/OSI Extension
- 自局 IP アドレス指定機能を使用する場合
  XNF/LS/BASE
  XNF/LS/OSI Extension
  XNF/LS/OSI Extension/Cluster
- OSI 拡張高信頼化機能を使用する場合
  XNF/LS/BASE
  XNF/LS/OSI Extension
  XNF/LS/Host Adaptor

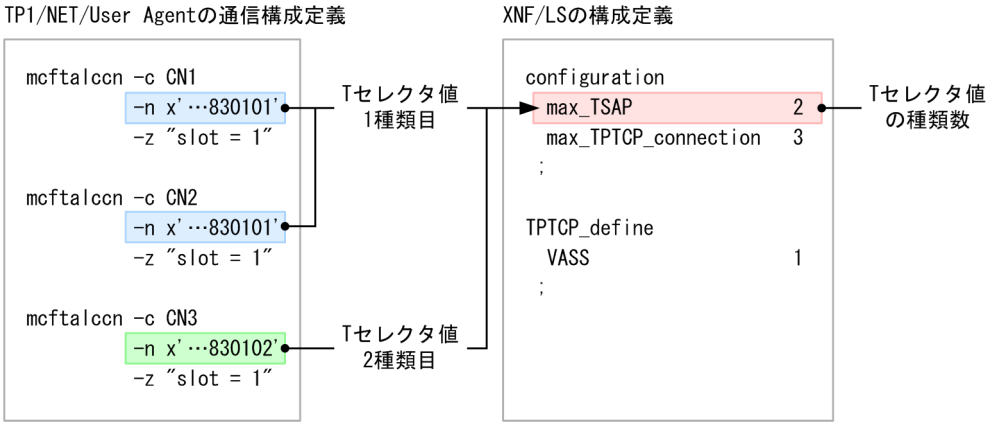
通信管理プログラムの詳細については，マニュアル「通信管理 XNF/LS 使用の手引」を参照してください。

どの通信管理プログラムの機能を使用する場合でも，コネクション定義（mcftalccn）で指定する定義の内容と，XNF/LS のシステム定義時に指定する定義の内容とを関連づける必要があります。関連づける必要のある項目を次に示します。

- コネクション数
- T セレクタ値の種類数※
- 仮想スロット番号

注※

例えば，次の図に示すように三つの mcftalccn コマンドを定義した場合は，T セレクタ値の種類数である 2 を XNF/LS の通信構成定義と関連づけます。



関連づける項目は，使用する機能によって異なります。使用する機能別に，次に示します。

OSI 拡張機能を使用する場合

OSI 拡張機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-3　OSI 拡張機能を使用する場合の XNF/LS の定義との関連づけ

項番	TP1/NET/User Agent での定義内容		XNF/LS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_define	VASS

自局 IP アドレス指定機能を使用する場合

自局 IP アドレス指定機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-4　自局 IP アドレス指定機能を使用する場合の XNF/LS の定義との関連づけ

項番	TP1/NET/User Agent での定義内容		XNF/LS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_slot	VASS

OSI 拡張高信頼化機能を使用する場合

OSI 拡張高信頼化機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-5　OSI 拡張高信頼化機能を使用する場合の XNF/LS の定義との関連づけ

項番	TP1/NET/User Agent での定義内容		XNF/LS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_VC	VASS

通信管理プログラムが XNF/AS の場合（mcftalccn -m "mode=xnfas"）

相手システムとの接続時に使用したい機能によって，使用する通信管理プログラムは次のとおり異なります。

- OSI 拡張機能を使用する場合
  XNF/AS/BASE
  XNF/AS/OSI Extension
- 自局 IP アドレス指定機能を使用する場合
  XNF/AS/BASE
  XNF/AS/OSI Extension
  XNF/AS/OSI Extension/Cluster
- OSI 拡張高信頼化機能を使用する場合
  XNF/AS/BASE
  XNF/AS/OSI Extension
  XNF/AS/Host Adaptor

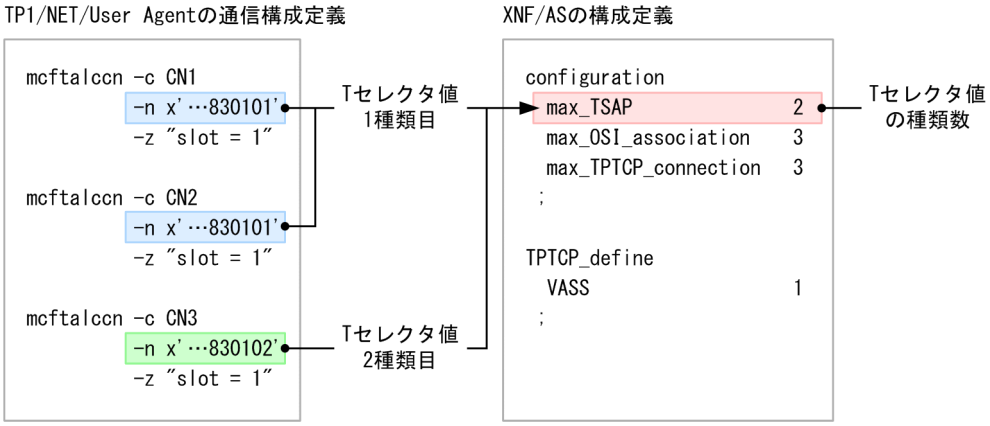
通信管理プログラムの詳細については，マニュアル「通信管理 XNF/AS 解説・運用編」およびマニュアル「通信管理 XNF/AS 構成定義編」を参照してください。

どの通信管理プログラムの機能を使用する場合でも，コネクション定義（mcftalccn）で指定する定義の内容と，XNF/AS のシステム定義時に指定する定義の内容とを関連づける必要があります。関連づける必要のある項目を次に示します。

- コネクション数
- T セレクタ値の種類数※
- 仮想スロット番号

注※

例えば，次の図に示すように三つの mcftalccn コマンドを定義した場合は，T セレクタ値の種類数である 2 を XNF/AS の通信構成定義と関連づけます。



関連づける項目は，使用する機能によって異なります。使用する機能別に，次に示します。

OSI 拡張機能を使用する場合

OSI 拡張機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-6　OSI 拡張機能を使用する場合の XNF/AS の定義との関連づけ

項番	TP1/NET/User Agent での定義内容		XNF/AS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_OSI_association max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_define	VASS

自局 IP アドレス指定機能を使用する場合

自局 IP アドレス指定機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-7　自局 IP アドレス指定機能を使用する場合の XNF/AS の定義との関連づけ

項番	TP1/NET/User Agent での定義内容		XNF/AS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_OSI_association max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_slot	VASS

OSI 拡張高信頼化機能を使用する場合

OSI 拡張高信頼化機能を使用する場合は，次の表に示すとおりに定義を関連づけてください。

表 6-8　OSI 拡張高信頼化機能を使用する場合の XNF/AS の定義との関連づけ

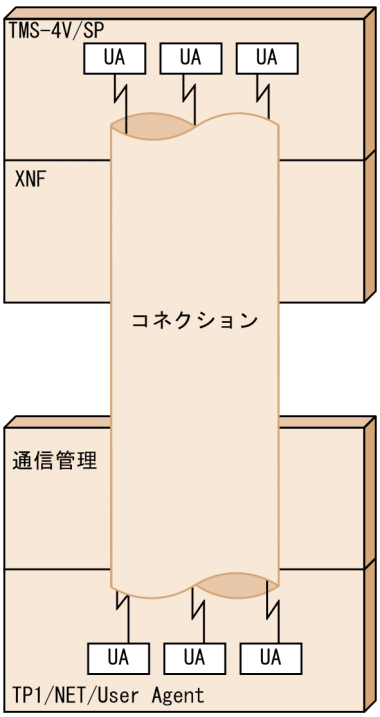
項番	TP1/NET/User Agent での定義内容		XNF/AS での定義内容	
	コマンド	内容	定義文	オペランド
1	mcftalccn	コネクション数（定義数）	configuration	max_OSI_association max_TPTCP_connection
2		T セレクタ値の種類数（-n オプションの T セレクタ値）		max_TSAP
3		仮想スロット番号（-z オプションの slot オペランド）	TPTCP_VC	VASS

相手システムの通信定義と関連づける内容

TMS-4V/SP のシステムとの接続

TMS-4V/SP のシステムと接続して TP1/NET/User Agent を使用する場合のネットワーク構成の例を次の図に示します。

図 6-3　ネットワーク構成の例（TMS-4V/SP のシステムと接続する場合）



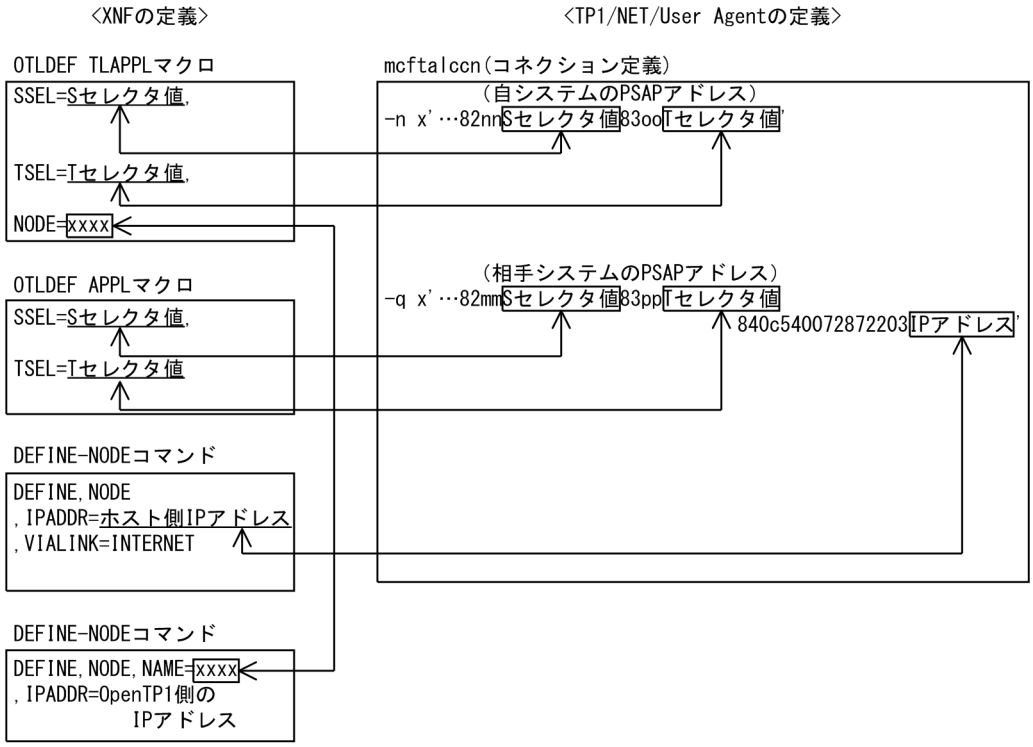
TP1/NET/User Agent は，図 6-3 の例のような場合，XNF，および TMS-4V/SP と定義を関連づける必要があります。

相手システムが XNF と TMS-4V/SP の場合の TP1/NET/User Agent の定義と関連づける内容を示します。

XNF の定義と関連づける内容

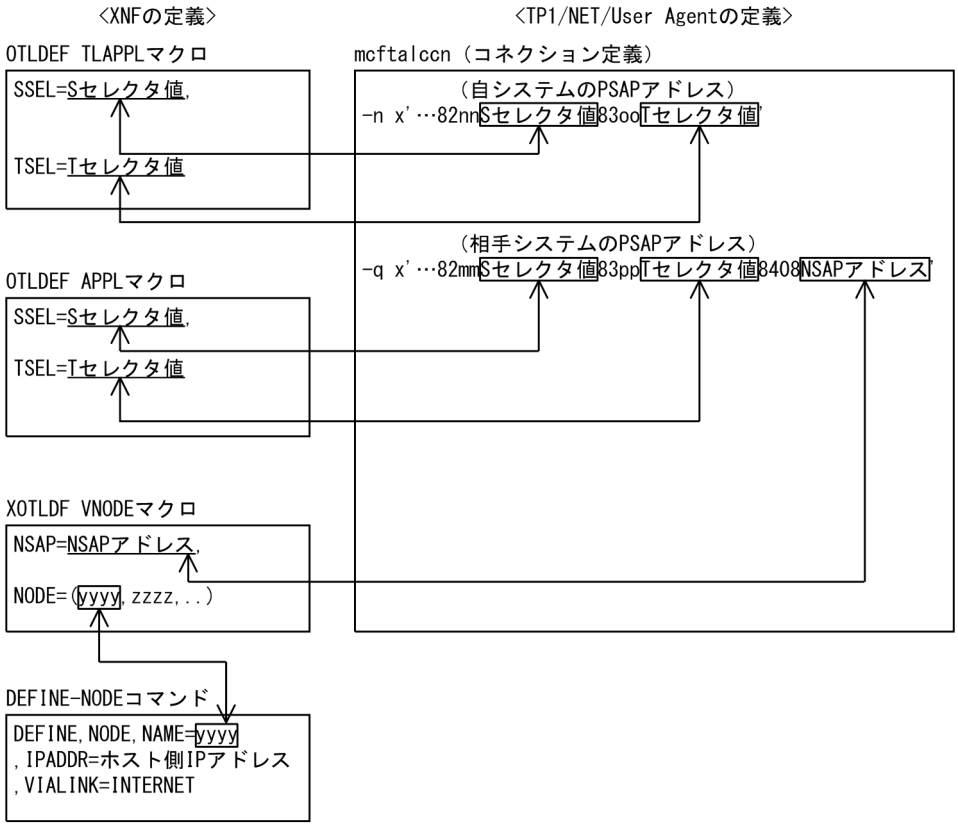
TP1/NET/User Agent のシステム定義と XNF の定義との関係を以降の図に示します。XNF の定義については，マニュアル「XNF TCP/IP 接続機能 XNF/TCP 定義とコマンド」を参照してください。

図 6-4　XNF の定義との関係（OSI 拡張機能または自局 IP アドレス指定機能を使用する場合）



（凡例）
nn, mm：Sセレクタ長
oo, pp：Tセレクタ長
xxxx：ノード名

図 6-5　XNF の定義との関係（OSI 高信頼化機能を使用する場合）

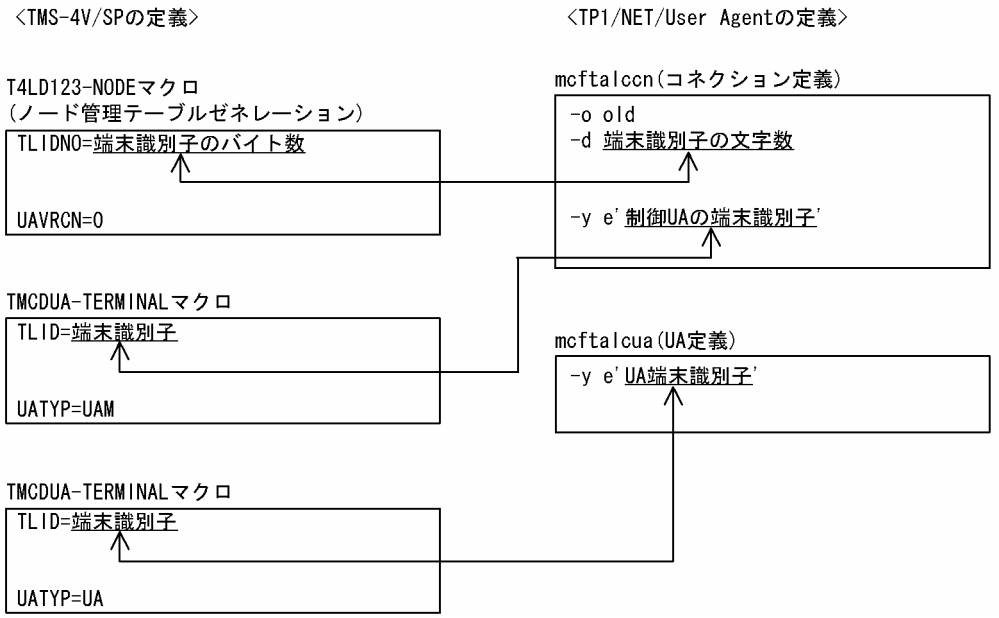


（凡例）
nn，mm：Sセレクタ長
oo，pp：Tセレクタ長
yyyy，zzzz：ノード名

TMS-4V/SP システム定義と関連づける内容

TP1/NET/User Agent のシステム定義と TMS-4V/SP システム定義との関係を次の図に示します。TMS-4V/SP の定義については，マニュアル「TMS-4V/SP OSAS/AP 間通信サービス-UA OSAS/UA/4VSP」を参照してください。

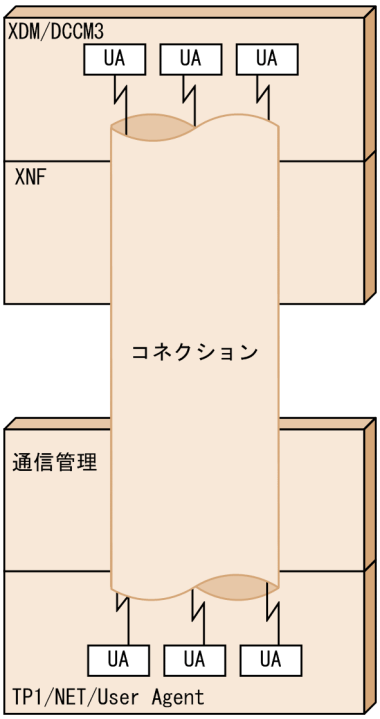
図 6-6　TMS-4V/SP システム定義との関係



XDM/DCCM3 のシステムとの接続

XDM/DCCM3 のシステムと接続して TP1/NET/User Agent を使用する場合のネットワーク構成の例を次の図に示します。

図 6-7　ネットワーク構成の例（XDM/DCCM3 のシステムと接続する場合）



TP1/NET/User Agent は，図 6-7 の例のような場合，XNF および XDM/DCCM3 と定義を関連づける必要があります。

相手システムが XNF，および XDM/DCCM3 の場合の TP1/NET/User Agent の定義と関連づける内容を示します。

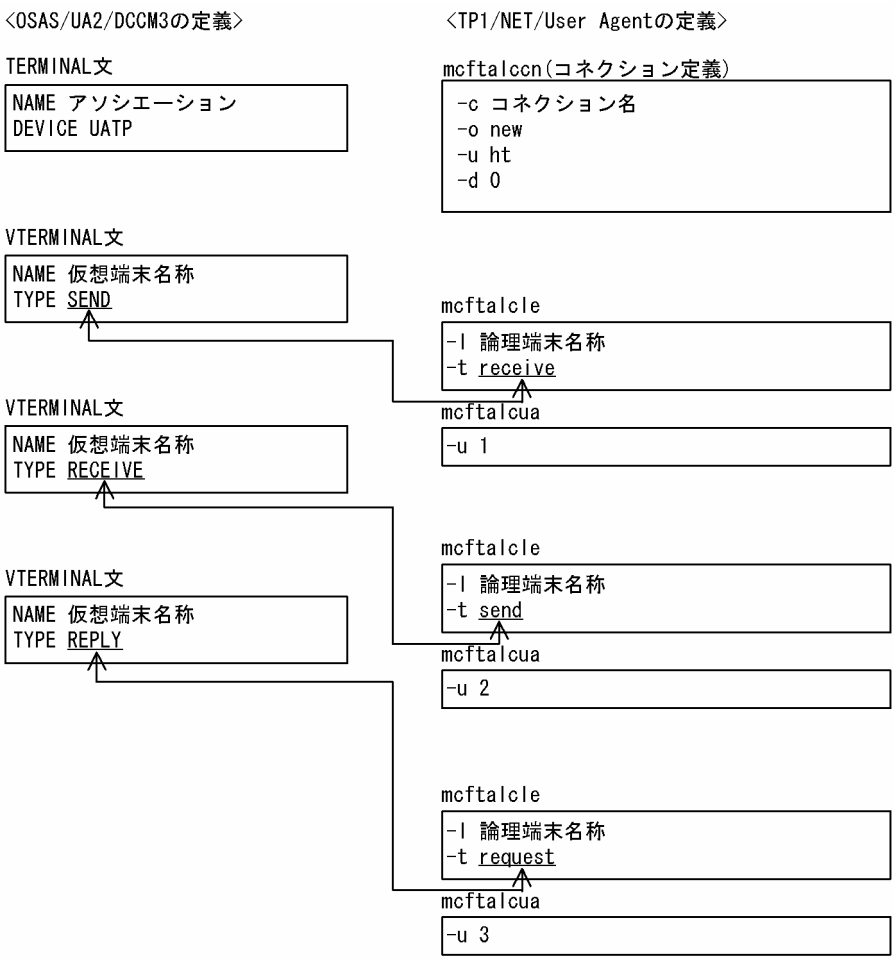
XNF の定義と関連づける内容

TP1/NET/User Agent のシステム定義と XNF の定義との関係については，図 6-4 と図 6-5 を参照してください。

XDM/DCCM3 システム定義と関連づける内容

TP1/NET/User Agent のシステム定義と XDM/DCCM3 システム定義との関係を次の図に示します。XDM/DCCM3 の定義については，マニュアル「OSI アプリケーション共通機能/AP 間通信サービス/DCCM3 OSAS/UA2/DCCM3」を参照してください。

図 6-8　XDM/DCCM3 システム定義との関係



OpenTP1 システムの変更に影響する定義

OpenTP1 システムの変更に伴って見直しが必要となる定義および OpenTP1 ファイルについて説明します。

IP アドレスの変更

IP アドレスを変更する場合に，見直しが必要な定義および変更手順について説明します。

IP アドレスを変更する場合に見直しが必要な定義

IP アドレスを変更する場合，見直す必要のある定義の一覧と発生する条件を次の表に示します。

表 6–9　IP アドレスを変更する場合に見直しが必要な定義の一覧

定義ファイル名	定義	見直しが必要な条件
MCF 通信構成定義	mcftalccn -q※1	OSI 拡張機能を使用する場合（OSI 拡張高信頼化機能を除く）
XNF の構成定義文	TPTCP_slot の IP_address※2	自局 IP アドレス指定機能を使用する場合

注※1

OSI 拡張機能を使用する場合，NSAP アドレスに IP アドレスが含まれています。NSAP アドレスの形式については，マニュアル「通信管理 XNF/AS NSAP アドレス概説編」またはマニュアル「通信管理 XNF/LS 使用の手引」を参照してください。

注※2

詳細については，マニュアル「通信管理 XNF/AS 構成定義編」またはマニュアル「通信管理 XNF/LS 使用の手引」を参照してください。

IP アドレスの変更手順

IP アドレスは，次の手順で変更してください。

1. OpenTP1 を正常停止します。
2. XNF/AS または XNF/LS を停止します。
3. MCF 通信構成定義および XNF の構成定義文について，変更前の IP アドレスを grep コマンドを使用して検索します。
4. 検索の結果，変更前の IP アドレスが見つかった場合には，変更します。
5. MCF 通信構成定義を変更した場合，MCF 通信構成定義の定義オブジェクトファイルを再作成します。XNF の構成定義文を変更した場合，ゼネレーションを行います。

コネクション（論理端末）の追加

コネクション（論理端末）を追加する場合に見直す必要のある定義の一覧，および再見積もりが発生する条件を次の表に示します。

表 6-10　コネクション（論理端末）を追加する場合に見直しが必要な定義の一覧

定義ファイル名	定義	再見積もりが発生する条件
システム環境定義	static_shmpool_size※1	無条件に再見積もりが必要
	dynamic_shmpool_size※1	同時に送受信するメッセージ数が増加する場合
MCF マネジャ定義	mcfmcomn -n	メッセージ出力通番を使用する場合
	mcfmcomn -p※2	無条件に再見積もりが必要
	mcfmexp -l※3	拡張予約定義を定義している場合
MCF 通信構成定義	mcftbuf -g count※4	無条件に再見積もりが必要
	mcftsts -l	状態を引き継ぐ論理端末が増える場合
システムサービス共通情報定義	max_open_fds※5	無条件に再見積もりが必要

注※1

詳細については，マニュアル「OpenTP1 システム定義」の「MCF サービス用の共用メモリの見積もり式」の説明を参照してください。

注※2

詳細については，マニュアル「OpenTP1 システム定義」の「mcfmcomn」と「MCF サービス用の共用メモリの見積もり式」の説明を参照してください。

注※3

詳細については，マニュアル「OpenTP1 システム定義」の「mcfmexp」の説明を参照してください。

注※4

詳細については，「mcftalccn（コネクション定義の開始）」の注意事項とマニュアル「OpenTP1 システム定義」の「mcftbuf」の説明を参照してください。

注※5

詳細については，「システムサービス共通情報定義」を参照してください。

見直す必要のある OpenTP1 ファイルの一覧，および再見積もりが発生する条件を次の表に示します。

表 6-11　コネクション（論理端末）を追加する場合に見直しが必要な OpenTP1 ファイルの一覧

OpenTP1 ファイル	再見積もりが発生する条件
ステータスファイル	次に示すどれかの条件の場合，再見積もりが必要

OpenTP1 ファイル	再見積もりが発生する条件
ステータスファイル	• メッセージ出力通番を使用する場合 • 拡張予約定義を定義している場合 • 状態を引き継ぐ論理端末が増える場合
メッセージキューファイル	入力キューまたは出力キューにディスクキューを割り当てていて，同時に送受信するメッセージ数が増加する場合

また，TP1/NET/User Agent の「リリースノート」を参照し，MCF 通信プロセスが使用するローカルメモリのメモリ所要量も見直してください。

定義例

ここでは，次に示す 2 種類の定義例を示します。

- 新 OSAS/UA プロトコルを使用する場合の定義例
- 旧 OSAS/UA プロトコルを使用する場合の定義例

新 OSAS/UA プロトコルを使用する場合の定義例

新 OSAS/UA プロトコルを使用した場合に，TP1/NET/User Agent を使用したシステム定義の例を示します。

TP1/NET/User Agent のシステム構成例を次の図に，この構成に沿った定義例をそのあとに示します。

なお，この定義のコーディング例を次のファイルで提供しています。

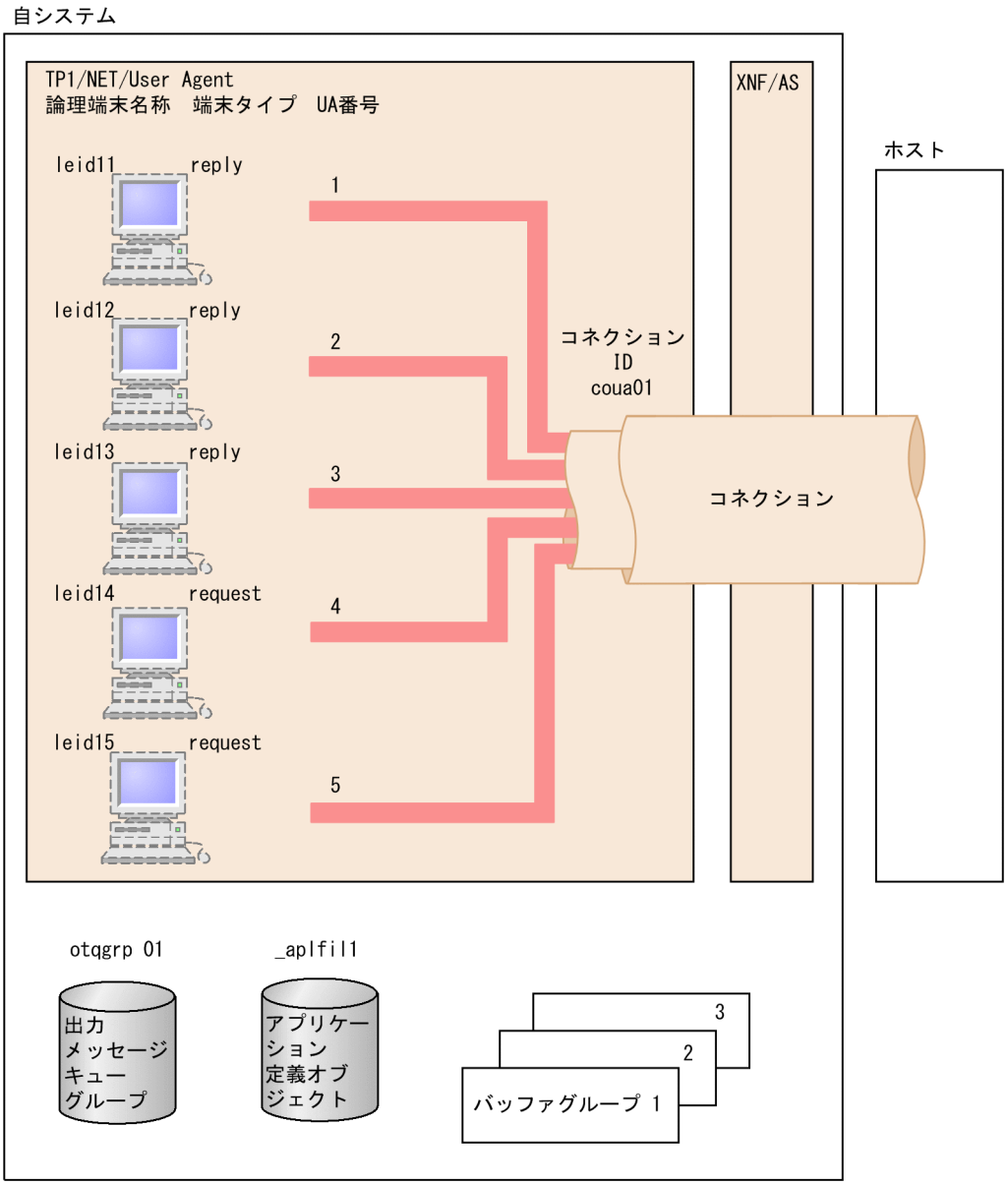
適用 OS が AIX，HP-UX の場合

- /BeTRAN/examples/mcf/UserAgent/conf/com_c2
- /BeTRAN/examples/mcf/UserAgent/conf/com_d2

適用 OS が Linux の場合

- /opt/OpenTP1/examples/mcf/UserAgent/conf/com_c2
- /opt/OpenTP1/examples/mcf/UserAgent/conf/com_d2

図 6-9　TP1/NET/User Agent のシステム構成例（新 OSAS/UA プロトコル使用時）



コーディング例（新 OSAS/UA プロトコルで XNF/AS 使用時の TP1/NET/User Agent の共通定義）

```
##################################################################
### MCF通信構成定義 共通定義                                   ###
##################################################################
#
### MCF環境定義
mcftenv    -s      01                                        ¥
           -a      _aplfil1
#
### MCF通信構成共通定義
mcftcomn
#
### トレース環境定義
mcfttrc
#
### バッファグループ定義（メッセージ送信バッファグループ）
```

```
mcftbuf    -g      ”groupno=1                                  ¥
                   length=4096                                 ¥
                   count=256”
#
### バッファグループ定義（メッセージ受信バッファグループ）
mcftbuf    -g      ”groupno=2                                  ¥
                   length=4096                                 ¥
                   count=256”
#
### バッファグループ定義（メッセージ編集用バッファグループ）
mcftbuf    -g      ”groupno=3                                  ¥
                   length=4096                                 ¥
                   count=256”
```

コーディング例（新 OSAS/UA プロトコルで XNF/AS 使用時の TP1/NET/User Agent のプロトコル固有定義）

```
#################################################################
### MCF通信構成定義 TP1/NET/User Agentプロトコル固有定義###
#################################################################
#
### コネクション定義の開始（coua01）
mcftalccn  -c  coua01                                          ¥
           -p  ua                                              ¥
           -n  x’0a81008202001283020012’                       ¥
           -g  ”sndbuf=1                                       ¥
               rcvbuf=2”                                       ¥
           -e  ”msgbuf=3                                       ¥
               count=5”                                        ¥
           -m  ”mode=xnfas”※1                                   ¥
           -i  auto                                            ¥
           -o  new                                             ¥
           -u  ht                                              ¥
           -b  ”bretrycnt=10”                                  ¥
           -k  each                                            ¥
           -d  0                                               ¥
     -q  x’16810082008302ffff840c5400728722031920660240 01’    ¥
           -z  ”slot=3”                                        ¥
           -l  0※2
### 論理端末/UA定義（reply型：leid11）
   mcftalcle  -l  leid11                                       ¥
              -t  reply                                        ¥
              -m  ”mmsgcnt=20                                  ¥
                  dmsgcnt=20”                                  ¥
              -k  ”quekind=disk                                ¥
                  quegrpid=otqgrp01”                           ¥
              -o  ”aj=no”
       mcftalcua  -u  1
### 論理端末/UA定義（reply型：leid12）
   mcftalcle  -l  leid12                                       ¥
              -t  reply                                        ¥
              -m  ”mmsgcnt=20                                  ¥
                  dmsgcnt=20”                                  ¥
              -k  ”quekind=disk                                ¥
                  quegrpid=otqgrp01”                           ¥
              -o  ”aj=no”
```

```
       mcftalcua  -u  2
### 論理端末/UA定義（reply型：leid13）
   mcftalcle  -l  leid13                                    ¥
              -t  reply                                     ¥
              -m  "mmsgcnt=20                               ¥
                  dmsgcnt=20"                               ¥
              -k  "quekind=disk                             ¥
                  quegrpid=otqgrp01"                        ¥
              -o  "aj=no"
       mcftalcua  -u  3
### 論理端末/UA定義（request型：leid14）
   mcftalcle  -l  leid14                                    ¥
              -t  request                                   ¥
              -m  "mmsgcnt=20                               ¥
                  dmsgcnt=20"                               ¥
              -k  "quekind=disk                             ¥
                  quegrpid=otqgrp01"                        ¥
              -o  "aj=no"
       mcftalcua  -u  4
### 論理端末/UA定義（request型：leid15）
   mcftalcle  -l  leid15                                    ¥
              -t  request                                   ¥
              -m  "mmsgcnt=20                               ¥
                  dmsgcnt=20"                               ¥
              -k  "quekind=disk                             ¥
                  quegrpid=otqgrp01"                        ¥
              -o  "aj=no"
       mcftalcua  -u  5
### コネクション定義の終了（coua01）
   mcftalced
```

注※1

XNF/AS 用に提供するファイルの内容です。

XNF/LS で使用する場合は次のように変更してください。

```
-m  "mode=tli"
```

注※2

XNF/AS 用に提供するファイルの内容です。

XNF/LS で使用する場合は次のように変更してください。

```
-T  "device=/dev/xnfw/cots"
```

旧 OSAS/UA プロトコルを使用する場合の定義例

旧 OSAS/UA プロトコルを使用した場合に，TP1/NET/User Agent を使用したシステム定義の例を示します。

TP1/NET/User Agent のシステム構成例を次の図に，この構成に沿った定義例をそのあとに示します。

なお，この定義のコーディング例を次のファイルで提供しています。

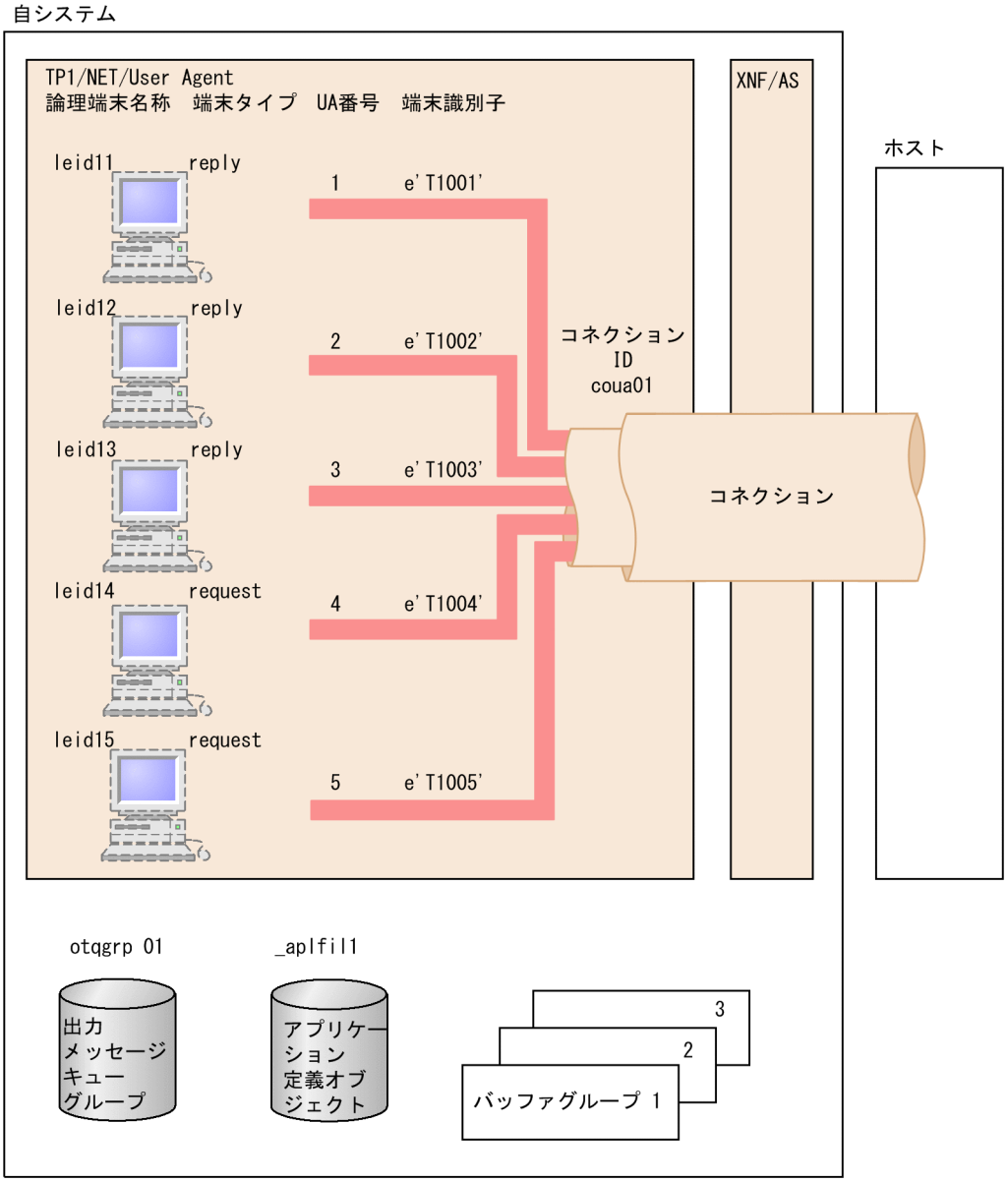
適用 OS が AIX，HP-UX の場合

- /BeTRAN/examples/mcf/UserAgent/conf/com_c1
- /BeTRAN/examples/mcf/UserAgent/conf/com_d1

適用 OS が Linux の場合

- /opt/OpenTP1/examples/mcf/UserAgent/conf/com_c1
- /opt/OpenTP1/examples/mcf/UserAgent/conf/com_d1

図 6-10　TP1/NET/User Agent のシステム構成例（旧 OSAS/UA プロトコル使用時）



コーディング例（旧 OSAS/UA プロトコルで XNF/AS 使用時の TP1/NET/User Agent の共通定義）

```
##------------------------------------------------------------
### MCF通信構成定義 共通定義                                 ###
##------------------------------------------------------------
#
```

```
### MCF環境定義
mcftenv    -s      01                                    ¥
           -a      _aplfil1
#
### MCF通信構成共通定義
mcftcomn
#
### トレース環境定義
mcfttrc
#
### バッファグループ定義（メッセージ送信バッファグループ）
mcftbuf    -g      ”groupno=1                            ¥
                   length=4096                           ¥
                   count=256”
#
### バッファグループ定義（メッセージ受信バッファグループ）
mcftbuf    -g      ”groupno=2                            ¥
                   length=4096                           ¥
                   count=256”
#
### バッファグループ定義（メッセージ編集用バッファグループ）
mcftbuf    -g      ”groupno=3                            ¥
                   length=4096                           ¥
                   count=256”
```

コーディング例（旧 OSAS/UA プロトコルで XNF/AS 使用時の TP1/NET/User Agent のプロトコル固有定義）

```
#################################################################
### MCF通信構成定義 TP1/NET/User Agentプロトコル固有定義###
#################################################################
#
### コネクション定義の開始（coua01）
mcftalccn  -c  coua01                                      ¥
           -p  ua                                          ¥
           -n  x’0a81008202001283020012’                   ¥
           -g  ”sndbuf=1                                   ¥
               rcvbuf=2”                                   ¥
           -e  ”msgbuf=3                                   ¥
               count=5”                                    ¥
           -m  ”mode=xnfas”※1                              ¥
           -i  auto                                        ¥
           -o  old                                         ¥
           -u  ht                                          ¥
           -b  ”bretrycnt=10”                              ¥
           -k  each                                        ¥
           -d  5                                           ¥
           -y  e’T1000’                                    ¥
      -q   x’16810082008302ffff840c5400728722031920660240 01’ ¥
           -z  ”slot=3”                                    ¥
           -l  0※2
### 論理端末/UA定義（reply型：leid11）
   mcftalcle  -l  leid11                                   ¥
              -t  reply                                    ¥
              -m  ”mmsgcnt=20                              ¥
                  dmsgcnt=20”                              ¥
```

```
              -k  ”quekind=disk                          ¥
                  quegrpid=otqgrp01”                     ¥
              -o  ”aj=no”
       mcftalcua  -u  1                                  ¥
                  -y  e’T1001’
### 論理端末/UA定義（reply型：leid12）
   mcftalcle  -l  leid12                                 ¥
              -t  reply                                  ¥
              -m  ”mmsgcnt=20                            ¥
                  dmsgcnt=20”                            ¥
              -k  ”quekind=disk                          ¥
                  quegrpid=otqgrp01”                     ¥
              -o  ”aj=no”
       mcftalcua  -u  2                                  ¥
                  -y  e’T1002’
### 論理端末/UA定義（reply型：leid13）
   mcftalcle  -l  leid13                                 ¥
              -t  reply                                  ¥
              -m  ”mmsgcnt=20                            ¥
                  dmsgcnt=20”                            ¥
              -k  ”quekind=disk                          ¥
                  quegrpid=otqgrp01”                     ¥
              -o  ”aj=no”
       mcftalcua  -u  3                                  ¥
                  -y  e’T1003’
### 論理端末/UA定義（request型：leid14）
   mcftalcle  -l  leid14                                 ¥
              -t  request                                ¥
              -m  ”mmsgcnt=20                            ¥
                  dmsgcnt=20”                            ¥
              -k  ”quekind=disk                          ¥
                  quegrpid=otqgrp01”                     ¥
              -o  ”aj=no”
       mcftalcua  -u  4                                  ¥
                  -y  e’T1004’
### 論理端末/UA定義（request型：leid15）
   mcftalcle  -l  leid15                                 ¥
              -t  request                                ¥
              -m  ”mmsgcnt=20                            ¥
                  dmsgcnt=20”                            ¥
              -k  ”quekind=disk                          ¥
                  quegrpid=otqgrp01”                     ¥
              -o  ”aj=no”
       mcftalcua  -u  5                                  ¥
                  -y  e’T1005’
### コネクション定義の終了（coua01）
   mcftalced
```

注※1

XNF/AS 用に提供するファイルの内容です。

XNF/LS で使用する場合は次のように変更してください。

```
-m  ”mode=tli”
```

注※2

XNF/AS 用に提供するファイルの内容です。

XNF/LS で使用する場合は次のように変更してください。

```
-T  ”device=/dev/xnfw/cots”
```

7 運用コマンド

この章では，TP1/NET/User Agent で使用する運用コマンドについて説明します。

TP1/NET/User Agent の運用コマンド

ここでは，TP1/NET/User Agent に関係のあるオプションについてだけ説明しています。ほかのオプション，運用コマンドの入力方法，およびその他の運用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。

TP1/NET/User Agent で使用する運用コマンドの一覧を，次の表に示します。

表 7-1　TP1/NET/User Agent の運用コマンド

機能		コマンド名称	定義中組み込み	オフライン中に実行	オンライン中に実行	UAP から実行
コネクション管理	コネクションの確立	mcftactcn	×	×	○	○
	コネクションの解放	mcftdctcn	×	×	○	○
	コネクションの状態表示	mcftlscn	×	×	○	○
論理端末管理	論理端末の閉塞解除	mcftactle	×	×	○	○
	論理端末の閉塞	mcftdctle	×	×	○	○
	論理端末の状態表示	mcftlsle	×	×	○	○

（凡例）

○：組み込み，または実行ができます。

×：組み込み，または実行ができません。

mcftactcn（コネクションの確立）

形式

```
mcftactcn 〔-s MCF通信プロセス識別子〕 -c コネクションID
```

機能

コネクションを確立します。

オプション

●-s MCF 通信プロセス識別子　～〈数字（0～9），a～f〉((01～ef))

処理対象のコネクションを制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftactcn コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　～〈1～8 文字の識別子〉

確立するコネクションのコネクション ID を指定します。

コネクション ID は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，コネクション ID とコネクション ID との間を空白で区切ります。同一コネクション ID は，重複して指定できません。

また，コネクション ID は，*を使って一括指定ができます。一括指定と一括指定以外のコネクション ID を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべてのコネクションを確立します。

先行文字列*：先行文字列で始まるすべてのコネクションを確立します。

〈複数指定の例〉 cnn1，cnn2，cnn3 を指定する場合

```
-c "cnn1△cnn2△cnn3"
```

〈一括指定の例〉 cnn で始まるすべてのコネクションを指定する場合

```
-c "cnn*"
```

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftactcn コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftactcn コマンド入力元への応答を失敗しました。	メッセージログファイル
KFCA10371-I	mcftactcn コマンドを正常に受け付けました。	標準出力
KFCA10373-E	mcftactcn コマンドが異常終了しました。	標準エラー出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10391-E	mcftactcn コマンドはサポートされていません。	標準エラー出力
KFCA10500-I	ヘルプメッセージ	標準出力
KFCA13130-E	TP1/NET/User Agent でエラーが発生しました。	標準エラー出力
KFCA13132-E	コネクション確立済みのため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13133-E	コネクション確立処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13134-E	コネクション解放処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13143-E	コネクション確立が着呼指定のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力

注意事項

- mcftactcn コマンドが正常に受け付けられたかどうかは，コマンドのリターン値で判断しないでください。コマンドが出力したメッセージの内容で判断してください。

mcftactle（論理端末の閉塞解除）

形式

```
mcftactle 〔-s MCF通信プロセス識別子〕 〔-c コネクションID〕
          -l 論理端末名称
```

機能

論理端末の閉塞を解除します。

オプション

●-s MCF 通信プロセス識別子　～〈数字（0～9），a～f〉((01～ef))

処理対象の論理端末を制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftactle コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　～〈1～8 文字の識別子〉

閉塞解除したい論理端末に対応するコネクションのコネクション ID を指定します。

コネクション ID は複数指定できません。また，一括指定もできません。

●-l 論理端末名称　～〈1～8 文字の識別子〉

閉塞解除する論理端末の名称を指定します。

-c オプションを指定した場合は，指定したコネクション ID に対応する論理端末の名称を指定します。

論理端末名称は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，論理端末名称と論理端末名称との間を空白で区切ります。同一論理端末名称は，重複して指定できません。

また，論理端末名称は，*を使って一括指定ができます。一括指定と一括指定以外の論理端末名称を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべての論理端末の閉塞を解除します。

先行文字列*：先行文字列で始まるすべての論理端末の閉塞を解除します。

〈複数指定の例〉 len1，len2，len3 を指定する場合

```
-l ”len1△len2△len3”
```

〈一括指定の例〉 len で始まるすべての論理端末を指定する場合

```
-l ”len*”
```

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftactle コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftactle コマンド入力元への応答を失敗しました。	メッセージログファイル，または標準エラー出力
KFCA10371-I	mcftactle コマンドを正常に受け付けました。	標準出力
KFCA10373-E	mcftactle コマンドが異常終了しました。	標準エラー出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10382-E	指定した論理端末は登録されていません。	標準エラー出力
KFCA10390-E	指定されたコネクション ID と論理端末名称の対応が正しくありません。	標準エラー出力
KFCA10391-E	mcftactle コマンドはサポートされていません。	標準エラー出力
KFCA10395-E	指定したコネクションには未接続の論理端末名称が指定されています。	標準エラー出力
KFCA10503-I	ヘルプメッセージ	標準出力
KFCA13131-E	コネクションが未確立のため，運用コマンドは受け付けられません。	標準エラー出力

出力メッセージ ID	内容	出力先
KFCA13136-E	構成合わせで使用不可となった論理端末のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13137-E	論理端末が閉塞解除済みのため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13138-E	論理端末が閉塞解除処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13139-E	論理端末が閉塞処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13142-E	論理端末が相手システムからの受信拒否解除通知待ちのため，運用コマンドは受け付けられません。	標準エラー出力
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力

注意事項

- mcftactle コマンドが正常に受け付けられたかどうかは，コマンドのリターン値で判断しないでください。コマンドが出力したメッセージの内容で判断してください。

mcftdctcn（コネクションの解放）

形式

```
mcftdctcn 〔-s MCF通信プロセス識別子〕 -c コネクションID 〔-f〕
```

機能

コネクションを解放します。

このコマンド入力時，該当するコネクションに仕掛り中の UA がある場合，このコマンド処理は異常終了します。仕掛り中の UA がないことを確認してからコネクションを正常解放してください。

コマンド入力時に送信中であったメッセージの扱いについては，「9.1　障害の種類と対応処理」を参照してください。なお，コネクションが強制的に解放されても，UAP からの送信要求はできます。

オプション

●-s MCF 通信プロセス識別子　　～〈数字（0～9），a～f〉((01～ef))

処理対象のコネクションを制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftdctcn コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　　～〈1～8 文字の識別子〉

解放するコネクションのコネクション ID を指定します。

コネクション ID は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，コネクション ID とコネクション ID との間を空白で区切ります。同一コネクション ID は，重複して指定できません。

また，コネクション ID は，*を使って一括指定ができます。一括指定と一括指定以外のコネクション ID を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべてのコネクションを解放します。

先行文字列*：先行文字列で始まるすべてのコネクションを解放します。

〈複数指定の例〉 cnn1，cnn2，cnn3 を指定する場合

```
-c "cnn1△cnn2△cnn3"
```

〈一括指定の例〉 cnn で始まるすべてのコネクションを指定する場合

```
-c "cnn*"
```

●-f

該当するコネクションを強制的に解放します。

このオプションを指定すると，該当するコネクションが仕掛り中の場合，仕掛り中の処理を終了しないで無条件に解放します。

このオプションの指定を省略すると，該当するコネクションが仕掛り中の場合，このコマンドは異常終了します。

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftdctcn コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftdctcn コマンド入力元への応答を失敗しました。	メッセージログファイル
KFCA10371-I	mcftdctcn コマンドを正常に受け付けました。	標準出力
KFCA10373-E	mcftdctcn コマンドが異常終了しました。	標準エラー出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10391-E	mcftdctcn コマンドはサポートされていません。	標準エラー出力
KFCA10501-I	ヘルプメッセージ	標準出力
KFCA13131-E	コネクションが未確立のため，運用コマンドは受け付けられません。	標準エラー出力

出力メッセージ ID	内容	出力先
KFCA13133-E	コネクション確立処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13134-E	コネクション解放処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13141-E	論理端末が仕掛り中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力

注意事項

- mcftdctcn コマンドが正常に受け付けられたかどうかは，コマンドのリターン値で判断しないでください。コマンドが出力したメッセージの内容で判断してください。

mcftdctle（論理端末の閉塞）

形式

```
mcftdctle 〔-s MCF通信プロセス識別子〕 〔-c コネクションID〕
          -l 論理端末名称
```

機能

論理端末を閉塞します。

コマンド入力時に送信中であったメッセージの扱いについては，「9.1　障害の種類と対応処理」を参照してください。なお，論理端末が閉塞されても，UAP からの送信要求はできます。

オプション

●-s MCF 通信プロセス識別子　　～〈数字（0～9），a～f〉((01～ef))

処理対象の論理端末を制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftdctle コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　　～〈1～8 文字の識別子〉

閉塞したい論理端末に対応するコネクションのコネクション ID を指定します。

コネクション ID は複数指定できません。また，一括指定もできません。

●-l 論理端末名称　　～〈1～8 文字の識別子〉

閉塞する論理端末の名称を指定します。

-c オプションを指定した場合は，指定したコネクション ID に対応する論理端末の名称を指定します。

論理端末名称は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，論理端末名称と論理端末名称との間を空白で区切ります。同一論理端末名称は，重複して指定できません。

また，論理端末名称は，*を使って一括指定ができます。一括指定と一括指定以外の論理端末名称を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべての論理端末を閉塞します。

先行文字列*：先行文字列で始まるすべての論理端末を閉塞します。

〈複数指定の例〉 len1，len2，len3 を指定する場合

```
-l "len1△len2△len3"
```

〈一括指定の例〉 len で始まるすべての論理端末を指定する場合

```
-l "len*"
```

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftdctle コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftdctle コマンド入力元への応答を失敗しました。	メッセージログファイル，または標準エラー出力
KFCA10371-I	mcftdctle コマンドを正常に受け付けました。	標準出力
KFCA10373-E	mcftdctle コマンドが異常終了しました。	標準エラー出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10382-E	指定した論理端末は登録されていません。	標準エラー出力
KFCA10390-E	指定されたコネクション ID と論理端末名称の対応が正しくありません。	標準エラー出力
KFCA10391-E	mcftdctle コマンドはサポートされていません。	標準エラー出力
KFCA10395-E	指定したコネクションには未接続の論理端末名称が指定されています。	標準エラー出力
KFCA10504-I	ヘルプメッセージ	標準出力
KFCA13131-E	コネクションが未確立のため，運用コマンドは受け付けられません。	標準エラー出力

出力メッセージ ID	内容	出力先
KFCA13135-E	論理端末は閉塞済みのため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13136-E	構成合わせで使用不可となった論理端末のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13138-E	論理端末が閉塞解除処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA13139-E	論理端末が閉塞処理中のため，運用コマンドは受け付けられません。	標準エラー出力
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力

注意事項

- 受信仕掛り中に運用コマンド（mcftdctle）を入力した場合，受信仕掛り中のメッセージを破棄します。以降の受信メッセージは入力キューに登録しません。
- 送信仕掛り中に運用コマンド（mcftdctle）を入力した場合は，送信処理が中断されます。UAP からの送信メッセージは，出力キュー上に格納されます。
- mcftdctle コマンドが正常に受け付けられたかどうかは，コマンドのリターン値で判断しないでください。コマンドが出力したメッセージの内容で判断してください。

mcftlscn（コネクションの状態表示）

形式

```
mcftlscn 〔-s MCF通信プロセス識別子〕 -c コネクションID 〔-d〕
```

機能

コネクションの状態を標準出力に出力します。

オプション

●-s MCF 通信プロセス識別子　～〈数字（0～9），a～f〉((01～ef))

処理対象のコネクションを制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftlscn コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　～〈1～8 文字の識別子〉

状態を表示するコネクションのコネクション ID を指定します。

コネクション ID は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，コネクション ID とコネクション ID との間を空白で区切ります。同一コネクション ID は，重複して指定できません。

また，コネクション ID は，*を使って一括指定ができます。一括指定と一括指定以外のコネクション ID を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべてのコネクションの状態を表示します。

先行文字列*：先行文字列で始まるすべてのコネクションの状態を表示します。

〈複数指定の例〉 cnn1，cnn2，cnn3 を指定する場合

```
-c "cnn1△cnn2△cnn3"
```

〈一括指定の例〉 cnn で始まるすべてのコネクションを指定する場合

```
-c "cnn*"
```

●-d

コネクションの状態と該当するコネクションに対応する論理端末の情報を表示します。

このオプションの指定を省略すると，コネクションの状態だけを表示します。

出力形式

```
mmm cccccccc ppp sssss dddd
    llllllll ttt uuuu xxxx
```

注

-d オプションを指定しないで mcftlscn コマンドを実行した場合は，「mmm cccccccc ppp sssss dddd」の行だけ出力されます。

- mmm：MCF 識別子
- cccccccc：コネクション ID
- ppp：プロトコル種別
  UA…OSAS/UA プロトコル
- sssss：コネクションの状態
  ACT…確立
  ACT/B…確立処理中
  DCT…解放
  DCT/B…解放処理中
- dddd：詳細ステータス（保守情報）
- llllllll：論理端末名称
- ttt：論理端末タイプ
  REQ…request 型
  RLY…reply 型
  SND…send 型
  RCV…receive 型
- uuuu：UA 番号
- xxxx：UA ステータス（保守情報）

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftlscn コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力

出力メッセージ ID	内容	出力先
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftlscn コマンド入力元への応答を失敗しました。	メッセージログファイル
KFCA10360-I	状態表示を開始します。	標準出力
KFCA10361-I	標準情報を表示します。	標準出力
KFCA10362-I	詳細情報を表示します。	標準出力
KFCA10369-I	状態表示を終了します。	標準出力
KFCA10373-E	mcftlscn コマンドが異常終了しました。	標準エラー出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10391-E	mcftlscn コマンドはサポートされていません。	標準エラー出力
KFCA10502-I	ヘルプメッセージ	標準出力
KFCA13118-E	バッファ取得に失敗しました。	メッセージログファイル
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力
KFCA16429-I	付加情報を示します。	標準出力

mcftlsle（論理端末の状態表示）

形式

```
mcftlsle 〔-s MCF通信プロセス識別子〕 〔-c コネクションID〕
          -l 論理端末名称 〔-q〕
```

機能

論理端末の状態を標準出力に出力します。

オプション

●-s MCF 通信プロセス識別子　～〈数字（0～9），a～f〉((01～ef))

処理対象の論理端末を制御する MCF 通信サービスの MCF 通信プロセス識別子を指定します。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は指定できません。

MCF 通信プロセス識別子は複数指定できません。

このオプションの指定を省略すると，すべての MCF 通信サービスに対して，mcftlsle コマンドを実行します。したがって，MCF 通信サービスを検索するオーバヘッドが，運用コマンドの処理に加わります。

MCF 通信サービスが多い構成や運用コマンドを多数入力する運用を行う場合は，-s オプションで，MCF 通信プロセス識別子を指定する運用設計を行ってください。

●-c コネクション ID　～〈1～8 文字の識別子〉

状態を表示したい論理端末に対応するコネクションのコネクション ID を指定します。コネクション ID は複数指定できません。また，一括指定もできません。

●-l 論理端末名称　～〈1～8 文字の識別子〉

状態を表示する論理端末の名称を指定します。

-c オプションを指定した場合，指定したコネクション ID に対応する論理端末名称を指定します。

論理端末名称は，一度に 8 個まで指定できます。多数入力する運用を行う場合は，次に示す複数指定または一括指定を使用して，一つの運用コマンドで行う並列処理数を増やし，運用コマンド入力数を減らすように運用設計を行ってください。

複数指定するときは，引用符（"）で囲んで，論理端末名称と論理端末名称との間を空白で区切ります。同一論理端末名称は，重複して指定できません。

また，論理端末名称は，*を使って一括指定ができます。一括指定と一括指定以外の論理端末名称を混在して指定できません。一括指定するときは，引用符（"）で囲んで指定します。

*：すべての論理端末の状態を表示します。

先行文字列*：先行文字列で始まるすべての論理端末の状態を表示します。

〈複数指定の例〉 len1，len2，len3 を指定する場合

```
-l ”len1△len2△len3”
```

〈一括指定の例〉 len で始まるすべての論理端末を指定する場合

```
-l ”len*”
```

●-q

指定した論理端末に対応する出力キューの保留状態を表示します。

このオプションの指定を省略すると，論理端末に対応する出力キューの保留状態は表示しません。

出力形式

```
mmm llllllll sss〔tttt〕
    SYNC xxxxxxxxxx yyyyyyyyyy zzzzzzzzzz
      IO     :          :          :
    PRIO     :          :          :
    NORM     :          :          :
    iii ooo
```

- mmm：MCF 識別子
- llllllll：論理端末名称
- sss：論理端末の状態
  ACT…閉塞解除状態
  DCT…閉塞状態
- tttt：論理端末のテストモード状態（TP1/Message Control/Tester の使用時だけ表示）
  TEST…テストモード
  空白…非テストモード
- SYNC：同期型メッセージ
- IO：非同期型問い合わせ応答メッセージ
- PRIO：非同期型一方送信メッセージ（優先）
- NORM：非同期型一方送信メッセージ（一般）
- xxxxxxxxxx：未送信メッセージ数
- yyyyyyyyyy：未送信メッセージの先頭の出力通番（int の上限値まで表示可能）
- zzzzzzzzzz：未送信メッセージの最後の出力通番（int の上限値まで表示可能）
- iii：出力キューの入力の保留状態（-q オプションの指定時だけ表示）
  NOH…保留解除
  HLD…保留

- ooo：出力キューの出力の保留状態（-q オプションの指定時だけ表示）
  NOH…保留解除
  HLD…保留

出力メッセージ

出力メッセージ ID	内容	出力先
KFCA10350-I	mcftlsle コマンドが入力されました。	標準出力
KFCA10351-E	MCF 開始処理中です。	標準エラー出力
KFCA10352-E	MCF 終了処理中です。	標準エラー出力
KFCA10353-W	入力形式が誤っています。	標準エラー出力
KFCA10354-E	メモリ不足です。	メッセージログファイル，または標準エラー出力
KFCA10355-W	引数の指定が誤っています。	標準エラー出力
KFCA10356-E	プロセス間でタイムアウトが発生しました。	標準エラー出力
KFCA10357-E	MCF 内でタイムアウトが発生しました。	標準エラー出力
KFCA10358-E	内部関数のエラーが発生しました。	メッセージログファイル，または標準エラー出力
KFCA10359-W	mcftlsle コマンド入力元への応答を失敗しました。	メッセージログファイル
KFCA10360-I	状態表示を開始します。	標準出力
KFCA10364-I	表示情報	標準出力
KFCA10365-I	表示情報	標準出力
KFCA10369-I	状態表示を終了します。	標準出力
KFCA10373-E	mcftlsle コマンドが異常終了しました。	標準エラー出力
KFCA10378-I	「出力形式」を参照してください。	標準出力
KFCA10380-E	相手プロセスの検索に失敗しました。	標準エラー出力
KFCA10381-E	指定したコネクションは登録されていません。	標準エラー出力
KFCA10382-E	指定した論理端末は登録されていません。	標準エラー出力
KFCA10390-E	指定したコネクション ID と論理端末名称の対応が正しくありません。	標準エラー出力
KFCA10391-E	mcftlsle コマンドはサポートされていません。	標準エラー出力
KFCA10395-E	指定したコネクションには未接続の論理端末名称が指定されています。	標準エラー出力
KFCA10505-I	ヘルプメッセージ	標準出力
KFCA16402-E	RPC 障害が発生しました。	標準エラー出力

8 組み込み方法

この章では，TP1/NET/User Agent を OpenTP1 システムに組み込む方法について説明します。

8.1　TP1/NET/User Agent の組み込みの流れ

TP1/NET/User Agent を OpenTP1 システムに組み込むときの作業の流れを示します。

1. MCF メイン関数の作成

   TP1/NET/User Agent を起動するためには，MCF メイン関数をコーディングし，コンパイル，およびリンケージしておく必要があります。詳細は，「8.2　MCF メイン関数の作成」を参照してください。

2. MCF サービス名の登録

   TP1/NET/User Agent を実行するために，MCF サービス名をシステムサービス構成定義で定義しておく必要があります。

   MCF サービス名は MCF マネジャ定義オブジェクトファイル名と一致させてください。

   詳細は，マニュアル「OpenTP1 システム定義」を参照してください。

3. システムサービス情報定義ファイルの作成

   システムサービス情報定義ファイルをテキストエディタで作成します。作成するファイルのパス名は，「$DCDIR/lib/sysconf/システムサービス情報定義ファイル名」です。ファイルの定義形式については，「システムサービス情報定義」を参照してください。

4. 定義オブジェクトファイルの生成

   OpenTP1 のネットワークコミュニケーション定義の各ソースファイルから定義オブジェクトファイルを生成します。詳細は，「8.3　定義オブジェクトファイルの生成」を参照してください。

8.2 MCF メイン関数の作成

TP1/NET/User Agent は，OpenTP1 プロセスサービスによって起動されます。

TP1/NET/User Agent を起動するためには，ユーザが MCF メイン関数を作成し，コンパイル，およびリンケージを行って TP1/NET/User Agent の実行形式プログラムを作成する必要があります。リンケージには，mcfplua コマンドを使用します。

MCF メイン関数では，スタート関数（dc_mcf_svstart）を呼び出します。UOC を使用する場合は，MCF メイン関数で UOC の関数アドレスを指定してください。

UOC は，MCF メイン関数と同じ言語（ANSI C，C++または K&R 版 C）で作成してください。

MCF メイン関数のコーディング概要を図 8-1 と図 8-2 に示します。また，ディレクトリへの組み込み方法を図 8-3 に示します。

なお，これらのコーディング例は，次のファイルで提供しています。

適用 OS が AIX，HP-UX の場合

- /BeTRAN/examples/mcf/UserAgent/cmlib/ansi/com.c
- /BeTRAN/examples/mcf/UserAgent/cmlib/c/com.c

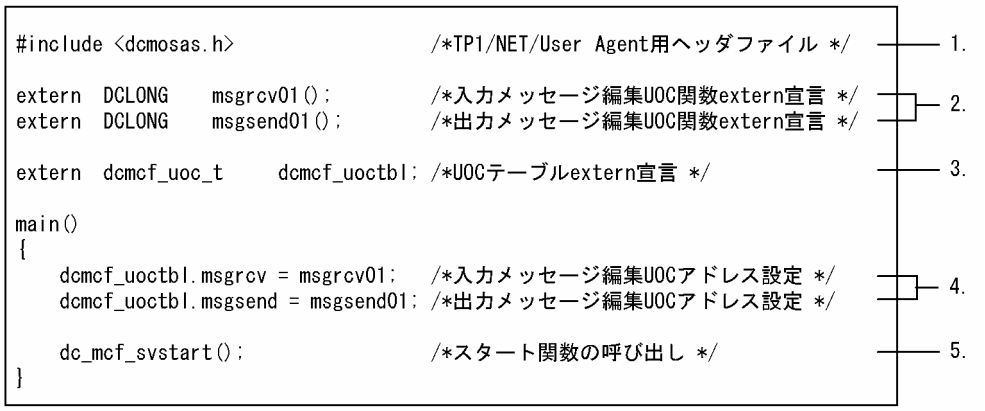
適用 OS が Linux の場合

- /opt/OpenTP1/examples/mcf/UserAgent/cmlib/ansi/com.c
- /opt/OpenTP1/examples/mcf/UserAgent/cmlib/c/com.c

図 8-1 MCF メイン関数のコーディング概要（ANSI C，C++の場合）

```
#include <dcmosas.h>                          /*TP1/NET/User Agent用ヘッダファイル */      1.

extern DCLONG msgrcv01(dcmcf_uoc_min_n *);
                                              /*入力メッセージ編集UOC関数extern宣言 */
extern DCLONG msgsend01(dcmcf_uoc_mout_n *);  /*出力メッセージ編集UOC関数extern宣言 */      2.

extern dcmcf_uoc_t dcmcf_uoctbl;              /*UOCテーブルextern宣言 */                   3.

int main()
{
    dcmcf_uoctbl.msgrcv = (dcmcf_uocfunc)msgrcv01;
                                              /*入力メッセージ編集UOCアドレス設定 */
    dcmcf_uoctbl.msgsend = (dcmcf_uocfunc)msgsend01;                                       4.
                                              /*出力メッセージ編集UOCアドレス設定 */

    dc_mcf_svstart();                         /*スタート関数の呼び出し*/                   5.
    return 0;
 }
```

図 8-2　MCF メイン関数のコーディング概要（K&R 版 C の場合）



1. TP1/NET/User Agent で提供するヘッダファイルを取り込みます。
2. 使用する UOC 関数を extern 宣言します。UOC のリターン値は DCLONG 型にしてください。

   UOC をまったく使用しない場合，このコーディングは必要ありません。
3. UOC テーブルを extern 宣言します。UOC を使用する場合，必ずこのとおりにコーディングしてください。

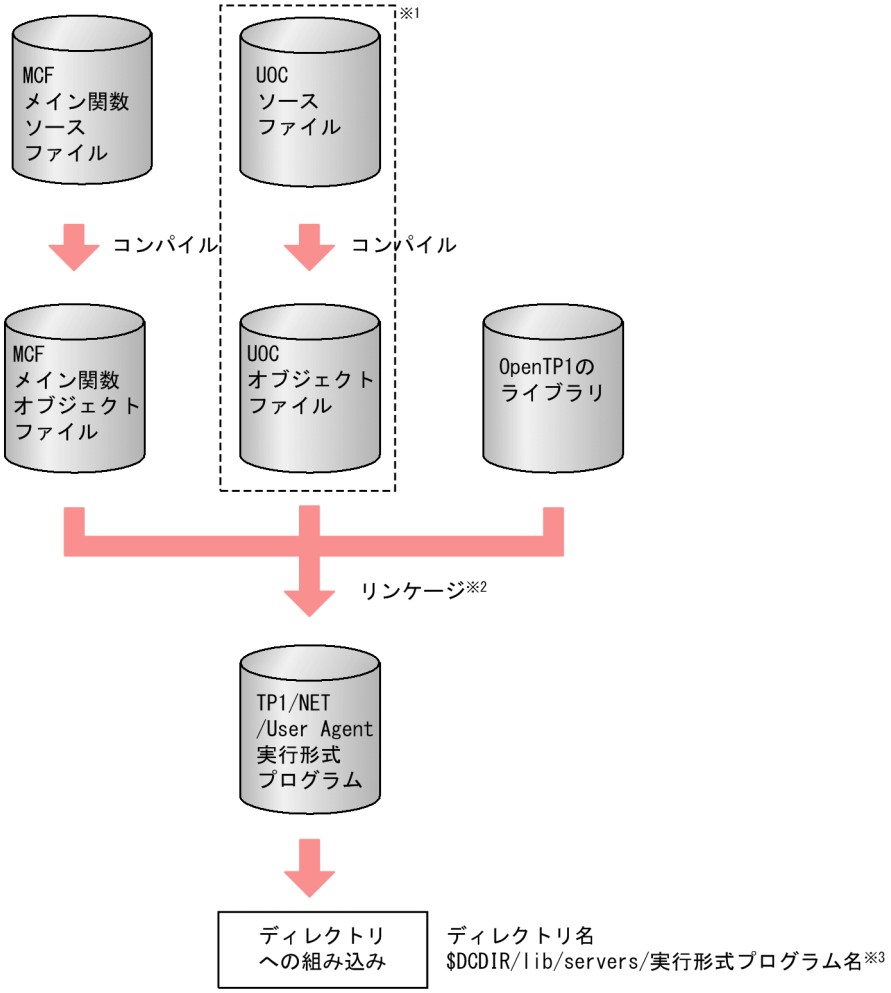
   UOC をまったく使用しない場合，このコーディングは必要ありません。
4. 各 UOC 関数のアドレスを，次に示すシステム提供変数に設定します。使用する UOC だけコーディングしてください。

   ```
   dcmcf_uoctbl.msgrcv  /*入力メッセージ編集UOCアドレス*/
   dcmcf_uoctbl.msgsend /*出力メッセージ編集UOCアドレス*/
   ```

   UOC をまったく使用しない場合，このコーディングは必要ありません。
5. スタート関数を呼び出します。MCF メイン関数には必ずコーディングしてください。

   スタート関数を呼び出したあとは，MCF メイン関数に制御が戻りません。そのため，スタート関数のあとにコーディングした処理は実行されませんので注意してください。

図 8-3　MCF メイン関数のディレクトリへの組み込み方法の概要



注※1

UOC を使用しない場合は，必要ありません。

注※2

mcfplua コマンドでリンケージします。

mcfplua コマンドの詳細については，TP1/NET/User Agent の「リリースノート」を参照してください。

注※3

TP1/NET/User Agent の実行形式プログラム名は，先頭が mcfu で始まる 8 文字以内の名称にしてください。

8.3 定義オブジェクトファイルの生成

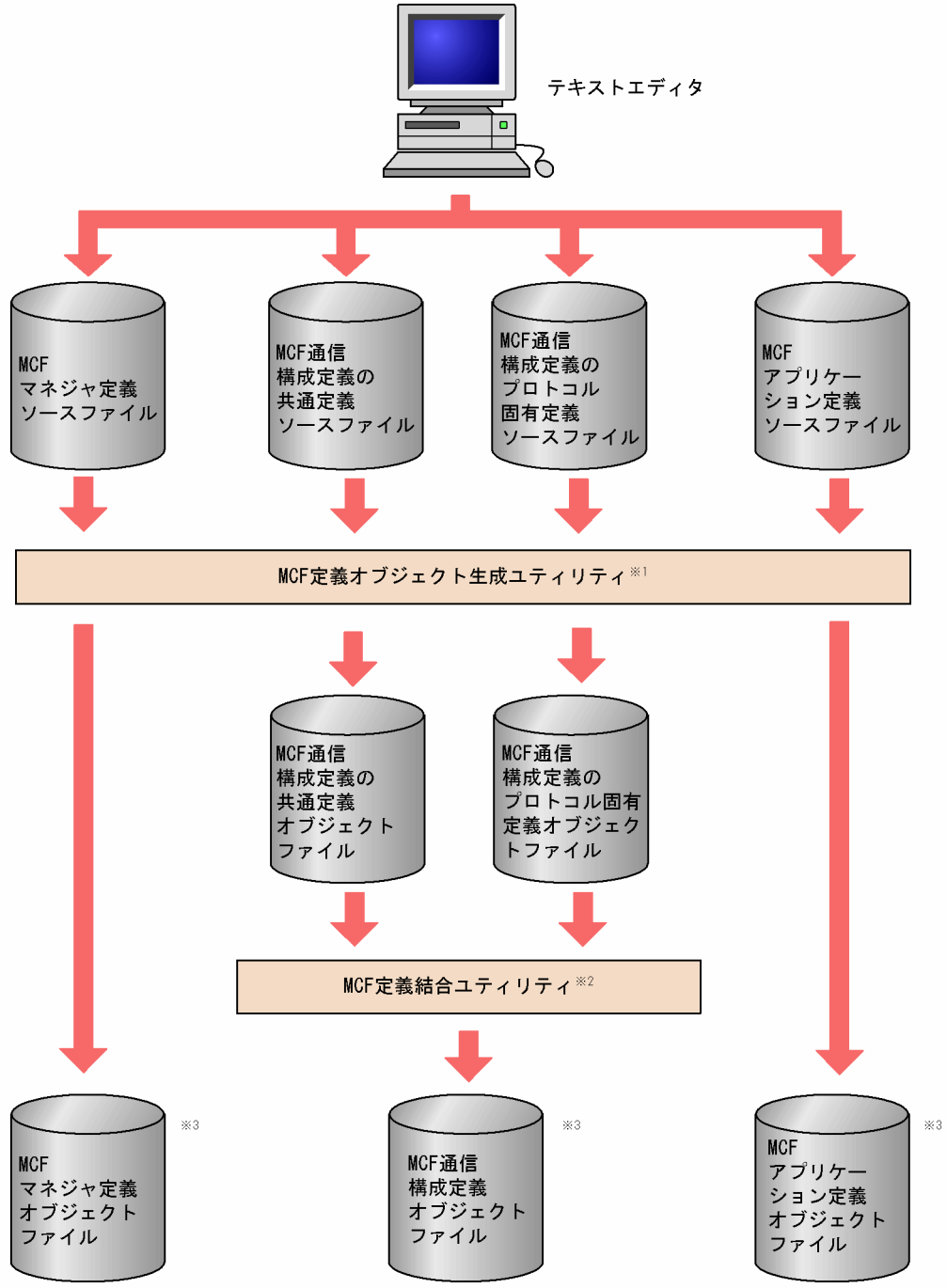
定義オブジェクトファイルを次の手順で生成します。

ただし，開始から再開始の間に定義オブジェクトファイルを変更しないでください。変更した場合，再開始時に正常に動作しない場合がありますのでご注意ください。

1. テキストエディタを使用して，MCF の定義ファイルから，次に示す定義ソースファイルを作成します。
   - MCF マネジャ定義ソースファイル
   - MCF 通信構成定義の共通定義ソースファイル
   - MCF 通信構成定義の TP1/NET/User Agent のプロトコル固有定義ソースファイル
   - MCF アプリケーション定義ソースファイル
2. MCF 定義オブジェクト生成ユティリティを使用して，定義ソースファイルから，次に示すオブジェクトファイルを作成します。
   - MCF マネジャ定義オブジェクトファイル
   - MCF 通信構成定義の共通定義オブジェクトファイル
   - MCF 通信構成定義の TP1/NET/User Agent のプロトコル固有定義オブジェクトファイル
   - MCF アプリケーション定義オブジェクトファイル
3. MCF 定義結合ユティリティを使用して，MCF 通信構成定義の共通定義とプロトコル固有定義のオブジェクトファイルを結合し，次に示すオブジェクトファイルを作成します。
   - MCF 通信構成定義オブジェクトファイル

定義オブジェクトファイルの作成方法の概要を次の図に示します。

図 8-4　定義オブジェクトファイルの作成方法の概要



注※1

次に示すコマンドで生成します。

```
mcfXXXX△-i△〔パス名〕入力ファイル名
        △-o△〔パス名〕出力オブジェクトファイル名
```

mcfXXXX は，ソースファイルごとに異なります。

- mcfmngr：MCF マネジャ定義ソースファイル
- mcfcomn：MCF 通信構成定義のソースファイル

- mcfosua：MCF 通信構成定義のプロトコル（TP1/NET/User Agent）固有定義ソースファイル
- mcfapli：MCF アプリケーション定義ソースファイル

MCF 定義オブジェクト生成ユティリティの mcfosua コマンドについては，「MCF 定義オブジェクトの生成」を，その他のコマ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。

注※2

次に示すコマンドで，MCF 通信構成定義の二つのオブジェクトファイルを結合します。

```
mcflink△-i△共通定義オブジェクトファイル名
       △TP1/NET/User Agent定義オブジェクトファイル名
       △-o△出力オブジェクトファイル名
```

注※3

定義オブジェクトファイルはシステム環境定義の DCCONFPATH で指定したディレクトリに格納してください。システム環境定義については，マニュアル「OpenTP1 システム定義」を参照してください。

9 障害対策

この章では，TP1/NET/User Agetnt 運用中に発生する障害と，TP1/NET/User Agent の対応処理，およびメッセージの処理について説明します。

9.1　障害の種類と対応処理

TP1/NET/User Agent の障害発生時の処理について，次に示す障害の種類ごとに説明します。

- コネクション（アソシエーション）障害
- UA（論理端末）障害
- 入力メッセージ編集 UOC の障害
- 入力キュー，スケジュールサービス，ネームサービス，または RPC の障害
- 出力キューおよび出力メッセージ編集 UOC の障害
- ジャーナル障害
- 送信バッファ障害
- 受信バッファ障害
- 編集バッファ障害
- UAP 障害
- MCF の障害
- OpenTP1 の障害

9.1.1　コネクション（アソシエーション）障害

コネクション障害の発生個所に応じた TP1/NET/User Agent の障害処理については，「9.2　コネクション障害」を参照してください。

表 9-1　コネクション障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・UA 層論理矛盾 ・下位層障害 ・リトライオーバ ・mcftdctcn -f コマンド入力 など	1. 各 UA の障害処理をします（「9.1.2　UA（論理端末）障害」参照）。 2. CERREVT（コネクション障害）を起動します。 3. コネクション障害を通知するメッセージログ（KFCA13102-E）を出力します。 4. コネクションを解放します。	障害の要因を取り除いたあと，次のどちらかの方法で再びコネクションを確立します。 ・運用コマンド（mcftactcn）を入力する ・API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）を発行する

9.1.2 UA（論理端末）障害

表 9-2 UA 障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・UA 障害 ・UA 層検出異常 ・タイムアウト ・mcftdctle コマンド入力 など	1. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 2. CERREVT（UA 障害）を起動します。 3. 論理端末を閉塞します。 4. UA（論理端末）の端末タイプに従って処理します。※1	障害の要因を取り除いたあと，次のどちらかの方法で UA 開局処理をします。 ・運用コマンド（mcftactle）を入力する ・API（dc_mcf_tactle 関数または CBLDCMCF('TACTLE△△')）を発行する
受信メッセージと論理端末の端末タイプの不一致	1. 受信メッセージを破棄します。※2 2. 相手システムに UERR を送信します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 4. CERREVT（UA 障害）を起動します。 5. 論理端末を閉塞します。	相手システムとの構成を見直してください。
送信メッセージと論理端末の端末タイプの不一致	dc_mcf_send，dc_mcf_reply，dc_mcf_sendrecv，dc_mcf_recvsync がエラーリターンします。	UAP で後処理をする必要があります。
UA 再開局処理の失敗	1. 該当する UA を縮退させます。 2. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 3. CERREVT（UA 障害）を起動します。 4. 論理端末を閉塞します。	障害の要因を取り除いたあと，次のどちらかの方法で再びコネクションを確立します。 ・運用コマンド（mcftactcn）を入力する ・API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）を発行する

注※1

論理端末の端末タイプごとの処理内容を次の表に示します。

論理端末の端末タイプ	TP1/NET/User Agent の処理
reply	問い合わせメッセージ（UINQ）受信障害の場合は，受信メッセージを無効にします。
	応答メッセージ（UREP）送信障害の場合は，次に示す処理をします。 1. 送信メッセージを送信済み扱いにします。 2. 送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。
request	一方送信メッセージ（UINQ）送信障害，および一方送信メッセージ（UREP）受信障害の場合は，次に示す処理をします。 1. 一方送信メッセージ（UINQ）の先頭セグメントからの再送準備をします。 2. 送信中断を通知するメッセージログ（KFCA10608-W）を出力します。
	UA 再開局後，一方送信メッセージ（UINQ）を再送します。
	送信完了通知（UINF）送信障害です。

論理端末の端末タイプ	TP1/NET/User Agent の処理
request（send-recv 時）	一方送信メッセージ（UINQ）送信障害，および一方送信メッセージ（UREP）受信障害の場合は，問い合わせメッセージ（UINQ）を送信済みにし，UAP にエラーリターンします。
	送信完了通知（UINF）送信障害です。
send	一方送信メッセージ（UBRD）送信障害，および送信完了通知（UINF）受信障害の場合は，次に示す処理をします。 1. 一方送信メッセージ（UBRD）の先頭セグメントからの再送準備をします。 2. 送信中断を通知するメッセージログ（KFCA13108-W）を出力します。
	UA 再開局後，一方送信メッセージ（UBRD）を再送します。
receive	一方送信メッセージ（UBRD）受信障害の場合は，一方送信メッセージを無効にします。
	処理完了通知（UINF）送信障害です。

注※2

1 論理メッセージの組み立て以前に不一致が検出されるため，ERREVT は起動しません。

9.1.3 入力メッセージ編集 UOC の障害

表 9-3 入力メッセージ編集 UOC の障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・UOC エラーリターン ・UOC 指定誤り	1. UOC エラーを通知するメッセージログ（KFCA10611-E），UOC 指定誤りを通知するメッセージログ（KFCA10620-E），およびアプリケーション不正を通知するメッセージログ（KFCA10610-E）を出力します。 2. UA（論理端末）の端末タイプに従って処理します。※	UOC の処理を見直してください。

注※

論理端末の端末タイプごとの処理内容を次の表に示します。

論理端末の端末タイプ	TP1/NET/User Agent の処理
reply	1. 相手システムに UERR を送信します。 2. CERREVT（UOC 障害）を起動します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 4. 論理端末を閉塞します。
request	1. 相手システムに UERR を送信します。 2. CERREVT（UOC 障害）を起動します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。

論理端末の端末タイプ	TP1/NET/User Agent の処理
request	4. 送信メッセージ（UINQ）は送信済み扱いとします。 5. 論理端末を閉塞します。
request（send-recv 時）	1. 相手システムに UERR を送信します。 2. CERREVT（UOC 障害）を起動します。 3. UAP にエラーリターンします。 4. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 5. 送信メッセージ（UINQ）は送信済み扱いとします。 6. 論理端末を閉塞します。
send	該当しません。
receive	1. 相手システムに UERR を送信します。 2. CERREVT（UOC 障害）を起動します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 4. 論理端末を閉塞します。

9.1.4 入力キュー，スケジュールサービス，ネームサービス，または RPC の障害

表 9-4 入力キュー，スケジュールサービス，ネームサービス，または RPC の障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・アプリケーション閉塞 ・サービス閉塞 ・アプリケーション名不正※1 ・入力キュー書き込み障害（メモリ含む）	1. 入力キュー障害を通知するメッセージログ（KFCA10604-E）を出力します。 2. 入力メッセージの種類に従って次に示す処理をします。 通信メッセージ入力障害の場合は，ERREVT1 または ERREVT2 を起動します。 ERREVT1，ERREVT2，ERREVT3，ERREVT4，または ERREVTA の場合は，メッセージを破棄します。 CERREVT，COPNEVT，または CCLSEVT の場合は，イベントを破棄します。	障害要因を取り除いてください。
アプリケーション名の形式不正	1. アプリケーション不正を通知するメッセージログ（KFCA10610-E）を出力します。 2. 相手システムに UERR を送信します。	アプリケーション名の形式を見直してください。
同期メッセージリターン時障害※2	1. 同期エラーを通知するメッセージログ（KFCA10606-E）を出力します。 2. UAP ではタイムアウトが発生します。 3. リターンの種類に従って次に示す処理をします。	障害の要因を取り除いたあと，次のどちらかの方法で再びコネクションを確立します。 ・運用コマンド（mcftactcn）を入力する

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
同期メッセージリターン時障害※2	正常にリターンした場合は，次のように処理します。 1. 相手システムに UERR を送信します。 2. CERREVT を起動します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。 4. 論理端末を閉塞します。 エラーリターンした場合は，無視して処理を続けます。	・API（dc_mcf_tactcn 関数または CBLDCMCF('TACTCN△△')）を発行する

注※1

アプリケーション名の形式が不正の場合，ERREVT1 を起動します。この ERREVT1 に対する応答メッセージを送信しなかった場合，相手システムに UERR を送信後，CERREVT を起動します。

注※2

RPC だけで発生する障害です。

9.1.5 出力キューおよび出力メッセージ編集 UOC の障害

表 9-5 出力キューおよび出力メッセージ編集 UOC の障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・出力キュー読み込み障害（メモリ含む） ・UOC エラーリターン ・UOC 指定誤り	1. UOC エラーを通知するメッセージログ（KFCA10611-E），UOC 指定誤りを通知するメッセージログ（KFCA10620-E），および出力キュー障害を通知するメッセージログ（KFCA10605-E）を出力します。 2. UA（論理端末）の端末タイプに従って処理します。※	障害要因を取り除いてください。
・メッセージ送信済み障害 ・メッセージリセット障害	メッセージ送信後の障害を通知するメッセージログ（KFCA10617-E）を出力します。	処理を続けます。

注※

論理端末の端末タイプごとの処理内容を次の表に示します。

論理端末の端末タイプ	TP1/NET/User Agent の処理
reply	1. 該当するメッセージを送信済み扱いとし，送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。 2. 相手システムに UERR を送信します。 3. UA 障害を通知するメッセージログ（KFCA13103-E），および UA 閉局を通知するメッセージログ（KFCA13115-I）を出力します。

論理端末の端末タイプ	TP1/NET/User Agent の処理
reply	4. CERREVT（出力キュー障害）を起動します。 5. 論理端末を閉塞します。
request	1. 該当するメッセージを送信済み扱いとし，送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。 2. 該当するメッセージを破棄し，次メッセージのサービスをします。
request（send-recv 時）	1. 該当するメッセージを送信済み扱いとし，送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。 2. UAP にエラーリターンします。 3. 該当するメッセージを破棄し，次メッセージのサービスをします。
send	1. 該当するメッセージを送信済み扱いとし，送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。 2. 該当するメッセージを破棄し，次メッセージのサービスをします。
receive	該当しません。

9.1.6 ジャーナル障害

表 9–6 ジャーナル障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
ジャーナル取得障害（AJ/IJ/OJ）	ジャーナル障害を通知するメッセージログ（KFCA10609-E）を出力します。	処理を続けます。

9.1.7 送信バッファ障害

表 9–7 送信バッファ障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
サイズ不足	出力キュー読み込み障害として扱います。 1. 出力障害を通知するメッセージログ（KFCA10605-E），および送信破棄を通知するメッセージログ（KFCA10607-W）を出力します。 2. 論理端末を閉塞します。 3. CERREVT（メッセージ出力障害）を起動します。	システム定義を修正してください。
バッファ数不足	出力キュー読み込み障害として扱います。 1. バッファ不足を通知するメッセージログ（KFCA10618-E），および送信破棄を通知するメッセージログ（KFCA10607-E）を出力します。 2. コネクションを解放します。 3. CERREVT（バッファ取得障害）を起動します。	システム定義を修正してください。

9.1.8　受信バッファ障害

表 9–8　受信バッファ障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
サイズ不足	1. 不正データ受信を通知するメッセージログ（KFCA13274-E）を出力します。 2. コネクションを解放します。 3. CERREVT（受信障害）を起動します。	相手システムとの構成を見直してください。
バッファ数不足	1. バッファ不足を通知するメッセージログ（KFCA13213-E）を出力します。 2. コネクションを解放します。 3. CERREVT（受信障害）を起動します。	システム定義を修正してください。

9.1.9　編集バッファ障害

表 9–9　編集バッファ障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
サイズ不足	ありません。	必要に応じて，UOC でエラーリターンしてください。
バッファ数不足	1. バッファ不足を通知するメッセージログ（KFCA10618-E）を出力します。 2. 障害の発生した処理に応じて，次に示す処理をします。 送信処理の場合は，次のように処理します。 1. 送信中断を通知するメッセージログ（KFCA10608-W）を出力します。 2. 送信メッセージを再送します。 受信処理の場合は，受信メッセージを破棄します。 3. コネクションを解放します。 4. CERREVT（バッファ取得障害）を起動します。	システム定義を修正してください。

9.1.10　UAP 障害

表 9–10　UAP 障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
• UAP 異常終了 • rollback（UAP 終了指定）呼び出し	reply 型の論理端末を使用している場合は，相手システムに UERR を送信します。	UAP を見直してください。

9.1.11　MCF の障害

表 9–11　MCF の障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
・内部論理矛盾 ・他 PP とのインタフェースエラー	1. KFCA13197-E，KFCA13198-E，または KFCA13199-E を出力します。 2. メモリダンプを出力します。 3. 必要に応じて，MCF のプロセスを異常終了します（回復できる場合は終了しません）。	保守情報（$DCDIR/spool ディレクトリ以下）を退避してください。
・MCF プログラムエラー ・UOC プログラムエラー	OS によるコアダンプを出力します。	保守情報（$DCDIR/spool ディレクトリ以下）を退避してください。

9.1.12　OpenTP1 の障害

表 9–12　OpenTP1 の障害発生時の処理

障害の内容	TP1/NET/User Agent の処理	ユーザの処理
OpenTP1 異常終了	OS によるコアダンプを出力します。	保守情報（$DCDIR/spool ディレクトリ以下）を退避してください。

9.2　コネクション障害

コネクションの処理中に，障害の発生する個所の区分を次の図に示します。また，障害発生個所に応じたTP1/NET/User Agent の障害処理について，次の表に示します。

図 9-1　コネクション障害の発生個所

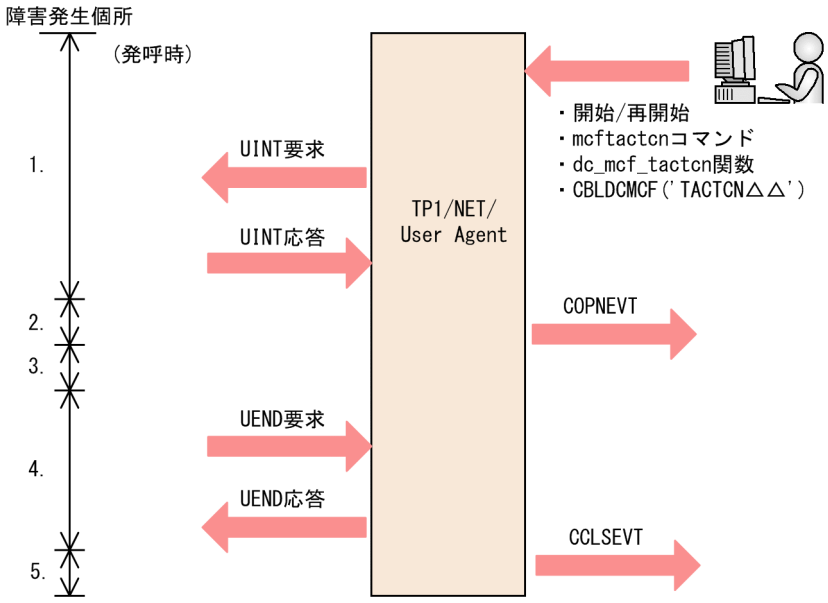


表 9-13　コネクション障害の発生個所に応じた障害処理

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理
1.	コネクション確立時のリトライオーバ（下位層の障害，またはタイムアウトによるリトライ）	コネクション障害	• CERREVT を起動します。 • コネクションを解放します。 解放後は運用コマンド（mcftactcn）または API（dc_mcf_tactcn 関数もしくは CBLDCMCF('TACTCN△△')）で再確立できます。
2.	入力キュー書き込みエラー（COPNEVT）	メッセージ入力障害	• メッセージを破棄します。
3.	コネクション確立後のコネクション障害	コネクション障害	• UA 障害を処理します。 • CERREVT を起動します。 • コネクションを解放します。 解放後は運用コマンド（mcftactcn）または API（dc_mcf_tactcn 関数もしくは CBLDCMCF('TACTCN△△')）で再確立できます。
4.	コネクション解放中のコネクション障害	コネクション障害	• CERREVT を起動します。 • コネクションを解放します。

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理
4.	コネクション解放中のコネクション障害	コネクション障害	解放後は運用コマンド（mcftactcn）または API（dc_mcf_tactcn 関数もしくは CBLDCMCF('TACTCN△△')）で再確立できます。
5.	入力キュー書き込みエラー（CCLSEVT）	メッセージ入力障害	• メッセージを破棄します。

9.3　問い合わせメッセージ，応答メッセージ障害

9.3.1　問い合わせメッセージ送信時の障害

TP1/NET/User Agent からの問い合わせメッセージ，およびそれに対する相手システムからの応答メッセージ処理中に，障害の発生する個所の区分を次の図に示します。また，障害発生個所に応じた TP1/NET/User Agent の障害処理について，次の表に示します。

図 9-2　問い合わせメッセージ送信時，応答メッセージ受信時の障害の発生個所

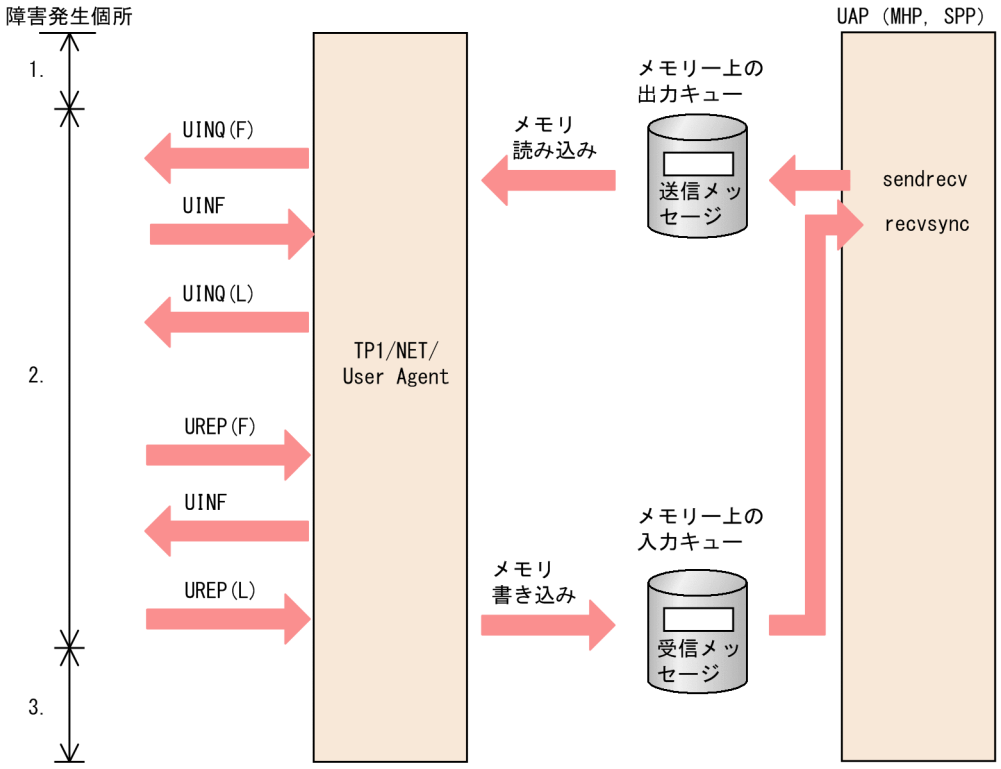


表 9-14　問い合わせメッセージ送信時，応答メッセージ受信時の障害の発生個所に応じた障害処理

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
1.	メモリ読み込みエラー	メッセージ出力障害	• UAP へエラーリターンします。 • 論理端末の閉塞処理はしません。	送信メッセージを破棄します。	UAP へリターン後に再送できます。または障害処理ができます。
2.	問い合わせ応答時の UA 障害（UA 障害通知，コネクション障害通知）	UA 障害（コネクション障害時を除く）	• UAP へエラーリターンします。 • CERREVT を起動します。	送信メッセージを破棄します。	• CERREVT 処理 • UAP へリターン後に再送できます。または障害処理ができます。

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
2.	問い合わせ応答時の UA 障害（UA 障害通知，コネクション障害通知）	UA 障害（コネクション障害時を除く）	• 論理端末を閉塞します。	送信メッセージを破棄します。	• CERREVT 処理 • UAP へリターン後に再送できます。または障害処理ができます。
	受信打ち切り受信時	送信メッセージの受信打ち切り	• UAP へエラーリターンします。 • 論理端末の閉塞処理はしません。	送信メッセージを破棄します。	UAP へリターン後に再送できます。または障害処理ができます。
	受信拒否受信時	送信メッセージの受信拒否	• UAP へエラーリターンします。 • CERREVT を起動します。 • 論理端末を閉塞します。	送信メッセージを破棄します。	• CERREVT 処理 • UAP へリターン後に再送できます。または障害処理ができます。
3.	応答エラー	メッセージ入力障害	• UAP へエラーリターンします。 • 論理端末を閉塞します。	受信メッセージを破棄します。	UAP へリターン後に再送できます。または障害処理ができます。

9.3.2 応答メッセージ送信時の障害

相手システムからの問い合わせメッセージ，および TP1/NET/User Agent からの応答メッセージ処理中に，障害の発生する個所の区分を次の図に示します。また，障害発生個所に応じた TP1/NET/User Agent の障害処理について，次の表に示します。

図 9-3　問い合わせメッセージ受信時，応答メッセージ送信時の障害の発生個所

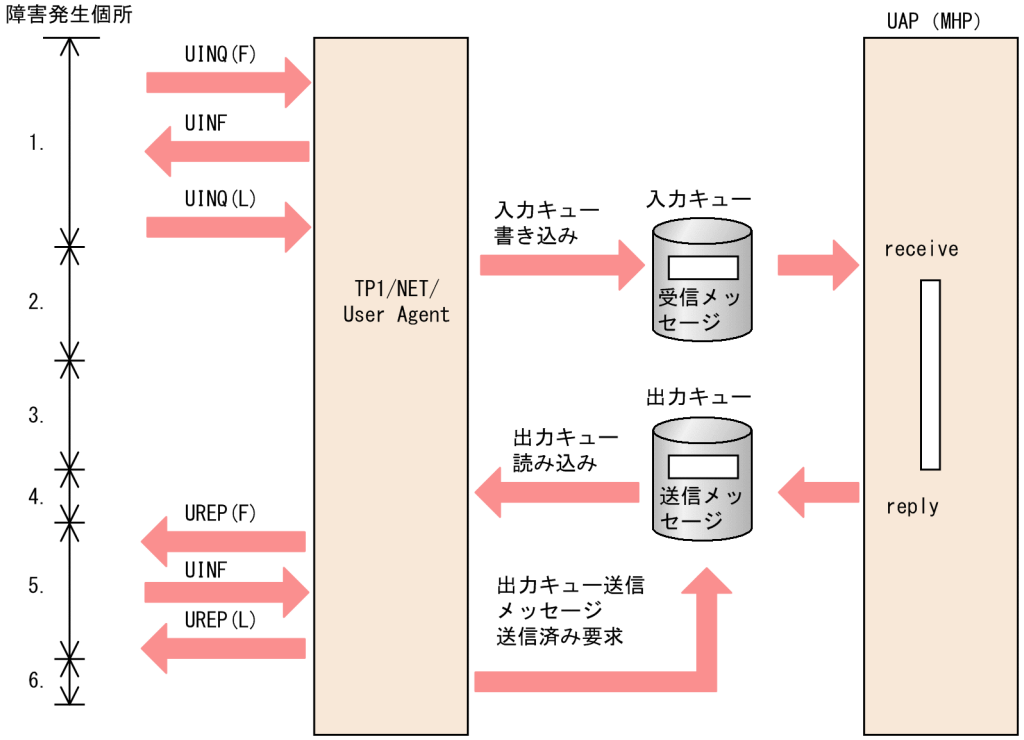


表 9-15　問い合わせメッセージ受信時，応答メッセージ送信時の障害の発生個所に応じた障害処理

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
1.	問い合わせメッセージ受信時 UA 障害 (UA 障害通知またはコネクション障害通知)	UA 障害（コネクション障害時を除く）	• CERREVT を起動します。 • 論理端末を閉塞します。	該当しません。	CERREVT 処理
	送信中断受信時	相手局がメッセージの送信を中断	論理端末の閉塞処理はしません。	受信メッセージを破棄します。	ありません。
2.	入力キュー書き込みエラー	メッセージ入力障害	• ERREVT1 または ERREVT2 を起動します。 • 論理端末の閉塞処理はしません。	ERREVT1，または ERREVT2 で受信メッセージの障害を通知します。	ERREVT1，または ERREVT2 で応答メッセージを送信できるようになります。
3.	UAP 異常終了	ありません。	ERREVT3 を起動します。	該当しません。	ERREVT3 で応答メッセージを送信できるようになります。
4.	出力キュー読み込みエラー	メッセージ出力障害	• CERREVT を起動します。	送信済みとします。	CERREVT 処理

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
4.	出力キュー読み込みエラー	メッセージ出力障害	• 論理端末を閉塞します。	送信済みとします。	CERREVT 処理
5.	応答メッセージ送信時 UA 障害 （UA 障害通知，コネクション障害通知）	UA 障害（コネクション障害時を除く）	• CERREVT を起動します。 • 論理端末を閉塞します。	送信済みとします。	CERREVT 処理
6.	出力キュー送信 メッセージ送信済み要求エラー	メッセージ送信後処理障害	処理を続けます。	送信済みとします。	ありません。

9.4　一方送信メッセージ障害

TP1/NET/User Agent からの一方送信メッセージ，および相手システムからの一方送信メッセージの処理中に，障害の発生する個所の区分を次の図に示します。また，障害発生個所に応じた TP1/NET/User Agent の障害処理について，次の表に示します。

図 9-4　一方送信メッセージ障害の発生個所

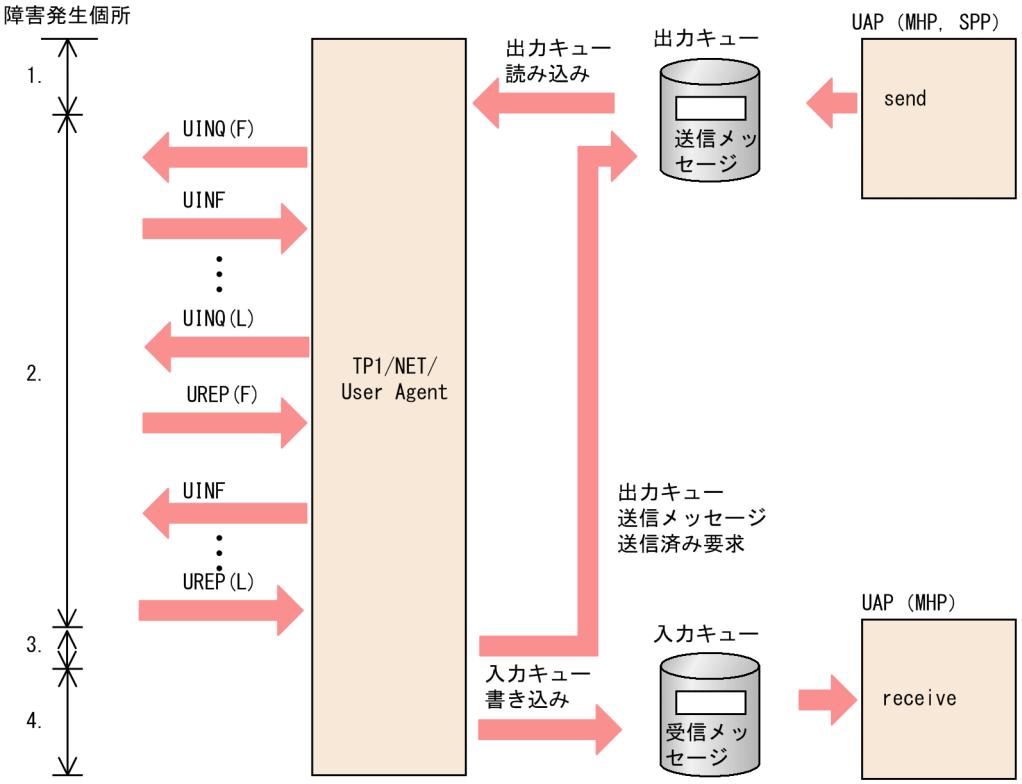


表 9-16　一方送信メッセージ障害の発生個所に応じた障害処理

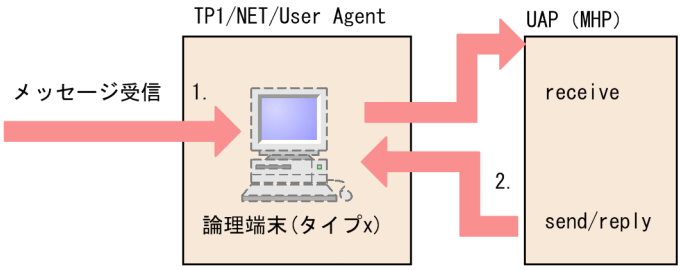
障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
1.	出力キュー読み込みエラー	メッセージ出力障害	次の問い合わせメッセージの送信処理をします。	送信済みとします。	ありません。
2.	一方送信メッセージ送受信時 UA 障害 （UA 障害通知，コネクション障害通知）	UA 障害（コネクション障害時を除く）	• CERREVT を起動します。 • 論理端末を閉塞します。	論理端末の閉塞を解除したあと，メッセージを再送します。	CERREVT 処理
	受信打ち切り受信時	送信メッセージの受信打ち切り	• ERREVTA を起動します。	送信メッセージを破棄します。	ERREVTA 処理

障害発生個所	内容	メッセージログの出力内容	TP1/NET/User Agent の処理	メッセージの扱い	UAP でできる処理
2.	受信拒否受信時	送信メッセージの受信拒否	論理端末を閉塞します。	相手局からの受信拒否解除受信後，メッセージを再送します。	ありません。
3.	出力キュー送信メッセージ送信済み要求エラー	メッセージ送信後処理障害	処理を続けます。	送信済みとします。	ありません。
4.	入力キュー書き込みエラー	メッセージ入力障害	• ERREVT1 または ERREVT2 を起動します。 • CERREVT を起動します。 • 論理端末を閉塞します。	ERREVT1，または ERREVT2 で受信メッセージを通知します。	• ERREVT1 処理または ERREVT2 処理 • CERREVT 処理

9.5　論理端末の端末タイプ不正

論理端末ごとに端末タイプを定義できるため，次の図の 1.と 2.に示す個所で論理端末の端末タイプ不正が発生することがあります。

図 9-5　論理端末の端末タイプ不正



（凡例）　1. ：受信メッセージ不正
2. ：送信先論理端末の端末タイプ不正

受信メッセージ不正

図 9-5 の 1.の個所では，論理端末で受信できるメッセージと，実際に受信したメッセージが整合しない場合に，受信メッセージ不正が発生します。

例えば，request 型の論理端末で問い合わせメッセージ（UINQ）を受信した場合，受信メッセージが該当する論理端末でサービスできないときは，UERR を送信し，論理端末を閉塞します。

送信先論理端末の端末タイプ不正

図 9-5 の 2.の個所では，送信先の論理端末の端末タイプと送信種別が整合しない場合に，送信先論理端末の端末タイプ不正が発生します。

例えば，reply 型の論理端末に対する一方送信要求では，送信先論理端末の端末タイプ不正の発生時，UAP にエラーリターンします。

付録

付録 A　バージョンアップ時の変更点

各バージョンでの変更点を次に示す分類ごとに示します。

- 関数，定義およびコマンドの追加・変更・削除
- 動作の変更
- 関数，定義およびコマンドのデフォルト値の変更

付録 A.1　07-50 での変更点

TP1/NET/User Agent 07-50 での関数，定義およびコマンドの追加・変更・削除を次の表に示します。

表 A–1　TP1/NET/User Agent 07-50 での関数，定義およびコマンドの追加・変更・削除

種別	分類	内容
追加	関数	なし
	定義	コネクション定義の開始（mcftalccn） • -T オプションの device オペランド
	コマンド	mcfosua コマンド • -r オプション
変更		なし
削除	関数	なし
	定義	コネクション定義の開始（mcftalccn） • -z オプションの dtea オペランド • -z オプションの alt1_dtea オペランド • -z オプションの alt2_dtea オペランド • -z オプションの alt3_dtea オペランド • -z オプションの alt4_dtea オペランド • -z オプションの alt5_dtea オペランド • -z オプションの alt6_dtea オペランド • -z オプションの alt7_dtea オペランド • -z オプションの alt1_slot オペランド • -z オプションの alt2_slot オペランド • -z オプションの alt3_slot オペランド • -z オプションの alt4_slot オペランド • -z オプションの alt5_slot オペランド • -z オプションの alt6_slot オペランド • -z オプションの alt7_slot オペランド
	コマンド	なし

TP1/NET/User Agent 07-50 での動作の変更点はありません。

TP1/NET/User Agent 07-50 でのデフォルト値の変更はありません。

付録 A.2　07-00 での変更点

TP1/NET/User Agent 07-00 での関数，定義およびコマンドの追加・変更・削除はありません。

TP1/NET/User Agent 07-00 での動作の変更点はありません。

TP1/NET/User Agent 07-00 でのデフォルト値の変更はありません。

付録 B　旧製品からの移行に関する注意事項

旧製品から移行する場合の注意事項を示します。

付録 B.1　ソースの互換性

バージョン 6 以前からバージョン 7 へ移行する場合の各種ソースファイルの互換性について説明します。

表 B-1　バージョン 6 以前で使用していたソースファイルの互換性

ソースファイルの種類	ソースファイルを作成した言語	互換性
UAP	C 言語	ソースファイルを変更しないで使用できます。※
	COBOL 言語	ソースファイルを変更しないで使用できます。
UOC	C 言語	ソースファイルを変更しないで使用できます。※
MCF 通信構成定義（プロトコル固有の定義）	－	ソースファイルを変更しないで使用できます。

（凡例）

－：該当する内容がないことを表します。

注※

バージョン 7 では，メッセージ送受信インタフェース，UOC，および MCF イベントインタフェースのそれぞれの引数ならびにパラメタの型が変更されていますが，この変更による UAP や UOC の処理への影響はありません。

詳細については，「付録 C　インタフェースの変更一覧（バージョン 6 以前から移行する場合）」を参照してください。

付録 C　インタフェースの変更一覧（バージョン 6 以前から移行する場合）

バージョン 6 以前のインタフェースの変更一覧を示します。

ここで説明するインタフェースを次に示します。

表 C-1　インタフェースの変更一覧

変更されたインタフェース		バージョン 7 のマニュアルの該当個所
メッセージ送受信インタフェース	dc_mcf_receive	「dc_mcf_receive － メッセージの受信（C 言語）」
	dc_mcf_recvsync	「dc_mcf_recvsync － 同期型のメッセージの受信（C 言語）」
	dc_mcf_reply	「dc_mcf_reply － 応答メッセージの送信（C 言語）」
	dc_mcf_resend	「dc_mcf_resend － メッセージの再送（C 言語）」
	dc_mcf_send	「dc_mcf_send － 一方送信メッセージの送信（C 言語）」
	dc_mcf_sendrecv	「dc_mcf_sendrecv － 同期型のメッセージの送受信（C 言語）」
ユーザオウンコーディング	入力メッセージ編集 UOC	「5.1.2　入力メッセージ編集 UOC インタフェース」
	出力メッセージ編集 UOC	「5.1.4　出力メッセージ編集 UOC インタフェース」
	送信メッセージの通番編集 UOC	「5.1.6　送信メッセージの通番編集 UOC インタフェース」
MCF イベントインタフェース		「5.2.3　MCF イベント情報の形式（C 言語）」
MCF メイン関数のコーディング概要		「8.2　MCF メイン関数の作成」

以降，バージョン 6 以前のインタフェースと，バージョン 7 のインタフェースの変更一覧を示します。変更個所には，下線を付与しています。

付録 C.1　メッセージ送受信インタフェース

(1)　dc_mcf_receive － メッセージの受信

(a)　ANSI C，C++の形式

バージョン 6 以前

```
#include <dcmcf.h>
int  dc_mcf_receive(long action,
                    long commform,
                    char *termnam,
                    char *resv01,
                    char *recvdata,
                    long *rdataleng,
```

バージョン 7

```
#include <dcmcf.h>
int  dc_mcf_receive(DCLONG action,
                    DCLONG commform,
                    char *termnam,
                    char *resv01,
                    char *recvdata,
                    DCLONG *rdataleng,
```

バージョン 6 以前	バージョン 7
long inbufleng, long *time)	DCLONG inbufleng, DCLONG *time)

(b) K&R 版 C の形式

バージョン 6 以前	バージョン 7
#include <dcmcf.h> int dc_mcf_receive(action, commform, termnam, resv01, recvdata, rdataleng, inbufleng, time) long action; long commform; char *termnam; char *resv01; char *recvdata; long *rdataleng; long inbufleng; long *time;	#include <dcmcf.h> int dc_mcf_receive(action, commform, termnam, resv01, recvdata, rdataleng, inbufleng, time) DCLONG action; DCLONG commform; char *termnam; char *resv01; char *recvdata; DCLONG *rdataleng; DCLONG inbufleng; DCLONG *time;

(2) dc_mcf_recvsync － 同期型のメッセージの受信

(a) ANSI C，C++の形式

バージョン 6 以前	バージョン 7
#include <dcmcf.h> int dc_mcf_recvsync(long action, long commform, char *termnam, char *resv01, char *recvdata, long *rdataleng, long inbufleng, long *time, long resv02)	#include <dcmcf.h> int dc_mcf_recvsync(DCLONG action, DCLONG commform, char *termnam, char *resv01, char *recvdata, DCLONG *rdataleng, DCLONG inbufleng, DCLONG *time, DCLONG resv02)

(b) K&R 版 C の形式

バージョン 6 以前	バージョン 7
#include <dcmcf.h> int dc_mcf_recvsync(action, commform, termnam, resv01, recvdata, rdataleng, inbufleng, time, resv02) long action;	#include <dcmcf.h> int dc_mcf_recvsync(action, commform, termnam, resv01, recvdata, rdataleng, inbufleng, time, resv02) DCLONG action;

バージョン６以前	バージョン７
long commform; char *termnam; char *resv01; char *recvdata; long *rdataleng; long inbufleng; long *time; long resv02;	DCLONG commform; char *termnam; char *resv01; char *recvdata; DCLONG *rdataleng; DCLONG inbufleng; DCLONG *time; DCLONG resv02;

(3) dc_mcf_reply － 応答メッセージの送信

(a) ANSI C，C++の形式

バージョン６以前	バージョン７
#include <dcmcf.h> int dc_mcf_reply(long action, long commform, char *resv01, char *resv02, char *senddata, long sdataleng, char *resv03, long opcd)	#include <dcmcf.h> int dc_mcf_reply(DCLONG action, DCLONG commform, char *resv01, char *resv02, char *senddata, DCLONG sdataleng, char *resv03, DCLONG opcd)

(b) K&R 版 C の形式

バージョン６以前	バージョン７
#include <dcmcf.h> int dc_mcf_reply(action, commform, resv01, resv02, senddata, sdataleng, resv03, opcd) long action; long commform; char *resv01; char *resv02; char *senddata; long sdataleng; char *resv03; long opcd;	#include <dcmcf.h> int dc_mcf_reply(action, commform, resv01, resv02, senddata, sdataleng, resv03, opcd) DCLONG action; DCLONG commform; char *resv01; char *resv02; char *senddata; DCLONG sdataleng; char *resv03; DCLONG opcd;

(4) dc_mcf_resend − メッセージの再送

(a) ANSI C，C++の形式

バージョン６以前

```
#include <dcmcf.h>
int  dc_mcf_resend(long action,
                   long commform,
                   char *rtermnam,
                   char *resv01,
                   long oseqid,
                   long orgseq,
                   char *otermnam,
                   char *resv02,
                   char *resv03,
                   char *resv04,
                   long opcd)
```

バージョン７

```
#include <dcmcf.h>
int  dc_mcf_resend(DCLONG action,
                   DCLONG commform,
                   char *rtermnam,
                   char *resv01,
                   DCLONG oseqid,
                   DCLONG orgseq,
                   char *otermnam,
                   char *resv02,
                   char *resv03,
                   char *resv04,
                   DCLONG opcd)
```

(b) K&R 版 C の形式

バージョン６以前

```
#include <dcmcf.h>
int  dc_mcf_resend(action,
                   commform,
                   rtermnam,
                   resv01,
                   oseqid,
                   orgseq,
                   otermnam,
                   resv02,
                   resv03,
                   resv04,
                   opcd)
long      action;
long      commform;
char      *rtermnam;
char      *resv01;
long      oseqid;
long      orgseq;
char      *otermnam;
char      *resv02;
char      *resv03;
char      *resv04;
long      opcd;
```

バージョン７

```
#include <dcmcf.h>
int  dc_mcf_resend(action,
                   commform,
                   rtermnam,
                   resv01,
                   oseqid,
                   orgseq,
                   otermnam,
                   resv02,
                   resv03,
                   resv04,
                   opcd)
DCLONG    action;
DCLONG    commform;
char      *rtermnam;
char      *resv01;
DCLONG    oseqid;
DCLONG    orgseq;
char      *otermnam;
char      *resv02;
char      *resv03;
char      *resv04;
DCLONG    opcd;
```

(5) dc_mcf_send − 一方送信メッセージの送信

(a) ANSI C，C++の形式

バージョン６以前

```
#include <dcmcf.h>
int  dc_mcf_send(long action,
                 long commform,
                 char *termnam,
                 char *resv01,
                 char *senddata,
```

バージョン７

```
#include <dcmcf.h>
int  dc_mcf_send(DCLONG action,
                 DCLONG commform,
                 char *termnam,
                 char *resv01,
                 char *senddata,
```

バージョン６以前	バージョン７
long sdataleng, char *resv02, long opcd)	DCLONG sdataleng, char *resv02, DCLONG opcd)

(b)　K&R 版 C の形式

バージョン６以前	バージョン７
#include <dcmcf.h> int dc_mcf_send(action, commform, termnam, resv01, senddata, sdataleng, resv02, opcd) long action; long commform; char *termnam; char *resv01; char *senddata; long sdataleng; char *resv02; long opcd;	#include <dcmcf.h> int dc_mcf_send(action, commform, termnam, resv01, senddata, sdataleng, resv02, opcd) DCLONG action; DCLONG commform; char *termnam; char *resv01; char *senddata; DCLONG sdataleng; char *resv02; DCLONG opcd;

(6)　dc_mcf_sendrecv － 同期型のメッセージの送受信

(a)　ANSI C，C++の形式

バージョン６以前	バージョン７
#include <dcmcf.h> int dc_mcf_sendrecv(long action, long commform, char *termnam, char *resv01, char *senddata, long sdataleng, char *recvdata, long *rdataleng, long inbufleng, long *time, long watchtime)	#include <dcmcf.h> int dc_mcf_sendrecv(DCLONG action, DCLONG commform, char *termnam, char *resv01, char *senddata, DCLONG sdataleng, char *recvdata, DCLONG *rdataleng, DCLONG inbufleng, DCLONG *time, DCLONG watchtime)

(b)　K&R 版 C の形式

バージョン６以前	バージョン７
#include <dcmcf.h> int dc_mcf_sendrecv(action, commform, termnam, resv01, senddata, sdataleng, recvdata,	#include <dcmcf.h> int dc_mcf_sendrecv(action, commform, termnam, resv01, senddata, sdataleng, recvdata,

バージョン 6 以前	バージョン 7
rdataleng, inbufleng, time, watchtime) long action; long commform; char *termnam; char *resv01; char *senddata; long sdataleng; char *recvdata; long *rdataleng; long inbufleng; long *time; long watchtime;	rdataleng, inbufleng, time, watchtime) DCLONG action; DCLONG commform; char *termnam; char *resv01; char *senddata; DCLONG sdataleng; char *recvdata; DCLONG *rdataleng; DCLONG inbufleng; DCLONG *time; DCLONG watchtime;

付録 C.2　ユーザオウンコーディング

(1)　入力メッセージ編集 UOC

(a)　形式

ANSI C，C++の形式

バージョン 6 以前	バージョン 7
#include <dcmcfuoc.h> long uoc_func(dcmcf_uoc_min_n *parm)	#include <dcmcfuoc.h> DCLONG uoc_func(dcmcf_uoc_min_n *parm)

K&R 版 C の形式

バージョン 6 以前	バージョン 7
#include <dcmcfuoc.h> long uoc_func(parm) dcmcf_uoc_min_n　*parm ;	#include <dcmcfuoc.h> DCLONG uoc_func(parm) dcmcf_uoc_min_n　*parm ;

(b)　パラメタの内容

dcmcf_uoc_min_n の内容

バージョン 6 以前	バージョン 7
typedef struct { 　　long pro_kind; 　　char le_name[9]; 　　char reserve1[7]; 　　long rcv_prim; 　　dcmcf_uocbuff_list_n *buflist_adr; 　　dcmcf_uocbuff_list_n *ebuflist_adr; 　　char aplname[9]; 　　char reserve2[7];	typedef struct { 　　DCLONG pro_kind; 　　char le_name[9]; 　　char reserve1[7]; 　　DCLONG rcv_prim; 　　dcmcf_uocbuff_list_n *buflist_adr; 　　dcmcf_uocbuff_list_n *ebuflist_adr; 　　char aplname[9]; 　　char reserve2[7];

バージョン 6 以前	バージョン 7

`    char *pro_indv_ifa;`
`    long rtn_detail;`
`    char reserve3[8];`
`} dcmcf_uoc_min_n;`
	`    char *pro_indv_ifa;`
	`    DCLONG rtn_detail;`
	`    char reserve3[8];`
	`} dcmcf_uoc_min_n;`

dcmcf_uocbuff_list_n（バッファリスト）の内容

バージョン 6 以前	バージョン 7

`typedef struct {`
`    long buf_num;`
`    long used_buf_num;`
`    char reserve1[8];`
`    dcmcf_uocbufinf_n`
`        buf_array[DCMCF_UOC_BUFF_MAX];`
`} dcmcf_uocbuff_list_n;`
	`typedef struct {`
	`    DCLONG buf_num;`
	`    DCLONG used_buf_num;`
	`    char reserve1[8];`
	`    dcmcf_uocbufinf_n`
	`        buf_array[DCMCF_UOC_BUFF_MAX];`
	`} dcmcf_uocbuff_list_n;`

dcmcf_uocbufinf_n（バッファ情報）の内容

バージョン 6 以前	バージョン 7

`typedef struct {`
`    char *buf_adr;`
`    unsigned long buf_size;`
`    unsigned long seg_size;`
`    char reserve1[4];`
`    dcmcfuoc_w_type buff_id;`
`    long buff_addr;`
`    char reserve2[4];`
`} dcmcf_uocbufinf_n;`
	`typedef struct {`
	`    char *buf_adr;`
	`    DCULONG buf_size;`
	`    DCULONG seg_size;`
	`    char reserve1[4];`
	`    dcmcfuoc_w_type buff_id;`
	`    DCMLONG buff_addr;`
	`    char reserve2[4];`
	`} dcmcf_uocbufinf_n;`

(2)　出力メッセージ編集 UOC

(a)　形式

ANSI C，C++の形式

バージョン 6 以前	バージョン 7
`#include <dcmcfuoc.h>` `long uoc_func(dcmcf_uoc_mout_n *parm)`	`#include <dcmcfuoc.h>` `DCLONG uoc_func(dcmcf_uoc_mout_n *parm)`

K&R 版 C の形式

バージョン 6 以前	バージョン 7

`#include <dcmcfuoc.h>`
`long uoc_func(parm)`
 	
`dcmcf_uoc_mout_n *parm ;`
	`#include <dcmcfuoc.h>`
	`DCLONG uoc_func(parm)`
	
	`dcmcf_uoc_mout_n *parm ;`

(b) パラメタの内容

dcmcf_uoc_mout_n の内容

バージョン 6 以前

```
typedef struct {
    long pro_kind;
    char le_name[9];
    char reserve1[7];
    dcmcf_uocbuff_list_n *buflist_adr;
    dcmcf_uocbuff_list_n *ebuflist_adr;
    long output_no;
    char msg_type;
    char outputno_flag;
    char resend_flag;
    char reserve2[1];
    char *pro_indv_ifa;
    long rtn_detail;
    char reserve3[20];
} dcmcf_uoc_mout_n;
```

バージョン 7

```
typedef struct {
    DCLONG pro_kind;
    char le_name[9];
    char reserve1[7];
    dcmcf_uocbuff_list_n *buflist_adr;
    dcmcf_uocbuff_list_n *ebuflist_adr;
    DCLONG output_no;
    char msg_type;
    char outputno_flag;
    char resend_flag;
    char reserve2[1];
    char *pro_indv_ifa;
    DCLONG rtn_detail;
    char reserve3[20];
} dcmcf_uoc_mout_n;
```

dcmcf_uocbuff_list_n（バッファリスト），dcmcf_uocbufinf_n（バッファ情報）の内容

入力メッセージ編集 UOC のパラメタの内容と同じです。「付録 C.2(1)(b)　パラメタの内容」を参照してください。

(3) 送信メッセージの通番編集 UOC

(a) 形式

ANSI C，C++の形式

バージョン 6 以前

```
#include <dcmcf.h>
long send_uoc(long flags,
              char *termname,
              long sendno,
              long sendid,
              long dataleng,
              char *senddata)
```

バージョン 7

```
#include <dcmcf.h>
DCLONG send_uoc(DCLONG flags,
                char *termname,
                DCLONG sendno,
                DCLONG sendid,
                DCLONG dataleng,
                char *senddata)
```

K&R 版 C の形式

バージョン 6 以前

```
#include <dcmcf.h>
long send_uoc(flags,
              termname,
              sendno,
              sendid,
              dataleng,
              senddata)
long    flags;
char    *termname;
long    sendno;
long    sendid;
```

バージョン 7

```
#include <dcmcf.h>
DCLONG send_uoc(flags,
                termname,
                sendno,
                sendid,
                dataleng,
                senddata)
DCLONG    flags;
char      *termname;
DCLONG    sendno;
DCLONG    sendid;
```

バージョン６以前	バージョン７
long dataleng; char *senddata;	DCLONG dataleng; char *senddata;

付録 C.3 MCF イベントインタフェース

(1) MCF イベントの共通ヘッダの形式

バージョン６以前	バージョン７

struct dc_mcf_evtheader {
char mcfevt_name[9] ;
char le_name[16] ;
char cn_name[9] ;
unsigned char format_kind;
char reserve01;
long time ;
};
	struct dc_mcf_evtheader {
	char mcfevt_name[9] ;
	char le_name[16] ;
	char cn_name[9] ;
	unsigned char format_kind;
	char reserve01;
	DCLONG time ;
	};

(2) ERREVT1 の形式

バージョン６以前とバージョン７で，差異はありません。

(3) ERREVT2 の形式

バージョン６以前とバージョン７で，差異はありません。

(4) ERREVT3 の形式

バージョン６以前とバージョン７で，差異はありません。

(5) ERREVTA の形式

バージョン６以前	バージョン７
struct dc_mcf_evta_type { struct dc_mcf_evtheader evtheader ; char reserve01[12] ; char reserve02[10] ; char reserve03[2] ; char ap_name[10] ; char reserve04[2] ; char reserve05[32] ; char reserve06[32] ; long user_leng ; char user_data[16] ; char reserve07[16] ; } ;	struct dc_mcf_evta_type { struct dc_mcf_evtheader evtheader ; char reserve01[12] ; char reserve02[10] ; char reserve03[2] ; char ap_name[10] ; char reserve04[2] ; char reserve05[32] ; char reserve06[32] ; DCLONG user_leng ; char user_data[16] ; char reserve07[16] ; } ;

(6) CERREVT の形式

バージョン 6 以前	バージョン 7

typedef struct {
  struct dc_mcf_evtheader header ;
  long err_fact ;
  long err_reason1 ;
  long err_reason2 ;
  long err_rcv_action ;
  char rname[16] ;
  char reserve1[26] ;
} dcmoum_cerrevt ;
	typedef struct {
	  struct dc_mcf_evtheader header ;
	  DCLONG err_fact ;
	  DCLONG err_reason1 ;
	  DCLONG err_reason2 ;
	  DCLONG err_rcv_action ;
	  char rname[16] ;
	  char reserve1[26] ;
	} dcmoum_cerrevt ;

(7) COPNEVT，CCLSEVT の形式

バージョン 6 以前	バージョン 7

typedef struct {
  struct dc_mcf_evtheader header ;
  long notice_fact ;
  char reserve1[12] ;
  char rname[16] ;
  char reserve2[26] ;
} dcmoum_statevt ;
	typedef struct {
	  struct dc_mcf_evtheader header ;
	  DCLONG notice_fact ;
	  char reserve1[12] ;
	  char rname[16] ;
	  char reserve2[26] ;
	} dcmoum_statevt ;

付録 C.4 MCF メイン関数のコーディング概要

MCF メイン関数のコーディング概要の変更一覧を示します。変更個所は，図中の網掛け部分です。

(1) ANSI C，C++の場合

(a) バージョン 6 以前

```
#include <dcmosas.h>                          /*TP1/NET/User Agent用ヘッダファイル */

extern long msgrcv01(dcmcf_uoc_min_n *);      /*入力メッセージ編集UOC関数extern宣言 */
extern long msgsend01(dcmcf_uoc_mout_n *);    /*出力メッセージ編集UOC関数extern宣言 */

extern dcmcf_uoc_t dcmcf_uoctbl;              /*UOCテーブルextern宣言 */

int main()
{
    dcmcf_uoctbl.msgrcv = (dcmcf_uocfunc)msgrcv01;
                                              /*入力メッセージ編集UOCアドレス設定 */
    dcmcf_uoctbl.msgsend = (dcmcf_uocfunc)msgsend01;
                                              /*出力メッセージ編集UOCアドレス設定 */
    dc_mcf_svstart();                         /*スタート関数の呼び出し*/
    return 0;
 }
```

(b) バージョン7

```
#include <dcmosas.h>                          /*TP1/NET/User Agent用ヘッダファイル */

extern DCLONG msgrcv01(dcmcf_uoc_min_n *);    /*入力メッセージ編集UOC関数extern宣言 */
extern DCLONG msgsend01(dcmcf_uoc_mout_n *); /*出力メッセージ編集UOC関数extern宣言 */

extern dcmcf_uoc_t dcmcf_uoctbl;              /*UOCテーブルextern宣言 */

int main()
{
    dcmcf_uoctbl.msgrcv = (dcmcf_uocfunc)msgrcv01;
                                              /*入力メッセージ編集UOCアドレス設定 */
    dcmcf_uoctbl.msgsend = (dcmcf_uocfunc)msgsend01;
                                              /*出力メッセージ編集UOCアドレス設定 */
    dc_mcf_svstart();                         /*スタート関数の呼び出し*/
    return 0;
 }
```

(2) K&R 版 C の場合

(a) バージョン6以前

```
#include <dcmosas.h>                          /*TP1/NET/User Agent用ヘッダファイル */

extern   long     msgrcv01();                 /*入力メッセージ編集UOC関数extern宣言 */
extern   long     msgsend01();                /*出力メッセージ編集UOC関数extern宣言 */

extern  dcmcf_uoc_t     dcmcf_uoctbl;         /*UOCテーブルextern宣言 */

main()
{
    dcmcf_uoctbl.msgrcv = msgrcv01;           /*入力メッセージ編集UOCアドレス設定 */
    dcmcf_uoctbl.msgsend = msgsend01;         /*出力メッセージ編集UOCアドレス設定 */

    dc_mcf_svstart();                         /*スタート関数の呼び出し*/
 }
```

(b) バージョン7

```
#include <dcmosas.h>                          /*TP1/NET/User Agent用ヘッダファイル */

extern  DCLONG    msgrcv01();                 /*入力メッセージ編集UOC関数extern宣言 */
extern  DCLONG    msgsend01();                /*出力メッセージ編集UOC関数extern宣言 */

extern  dcmcf_uoc_t     dcmcf_uoctbl;         /*UOCテーブルextern宣言 */

main()
{
    dcmcf_uoctbl.msgrcv = msgrcv01;           /*入力メッセージ編集UOCアドレス設定 */
    dcmcf_uoctbl.msgsend = msgsend01;         /*出力メッセージ編集UOCアドレス設定 */

    dc_mcf_svstart();                         /*スタート関数の呼び出し*/
 }
```

付録 D　メッセージ送受信の処理の流れ

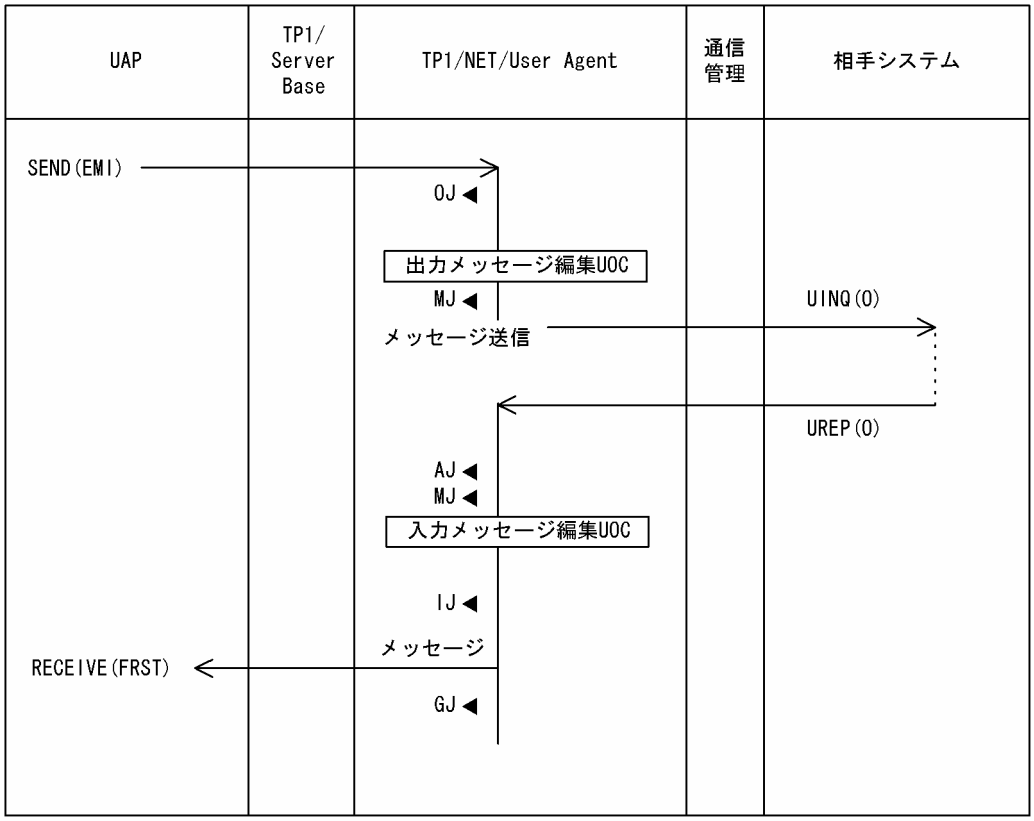
メッセージを送受信するときのデータの流れ，ジャーナルの取得タイミングなどを，メッセージの種類ごとに示します。

付録 D.1　一方送信メッセージ

(1)　一方送信メッセージ送信時の処理の流れ（request 型論理端末）

(a)　UA サービスデータ単位分割なしの場合

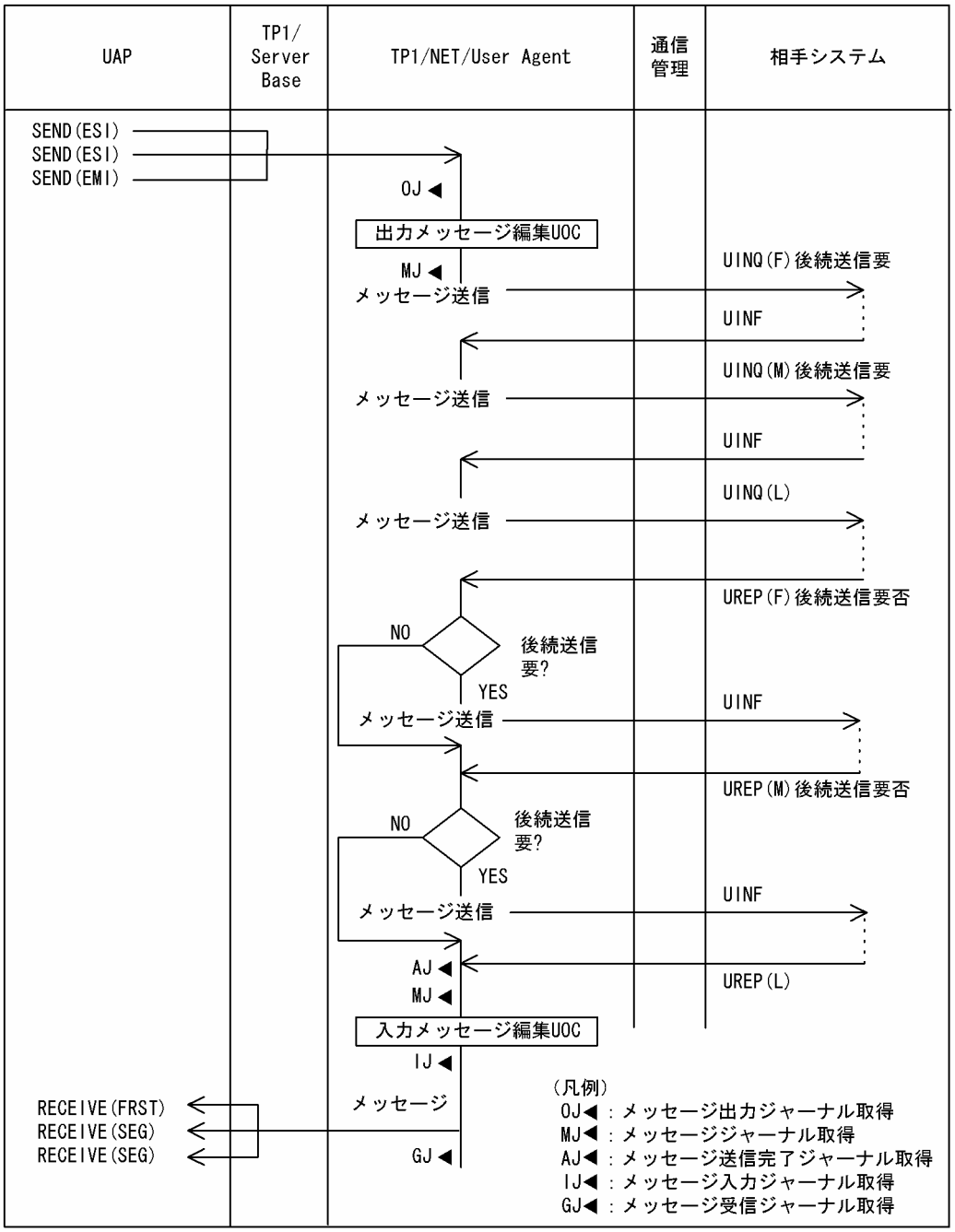
図 D-1　request 型論理端末からの一方送信メッセージの処理の流れ（UA サービスデータ単位分割なしの場合）



（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得
AJ◀：メッセージ送信完了ジャーナル取得
IJ◀：メッセージ入力ジャーナル取得
GJ◀：メッセージ受信ジャーナル取得

(b) UA サービスデータ単位分割ありの場合

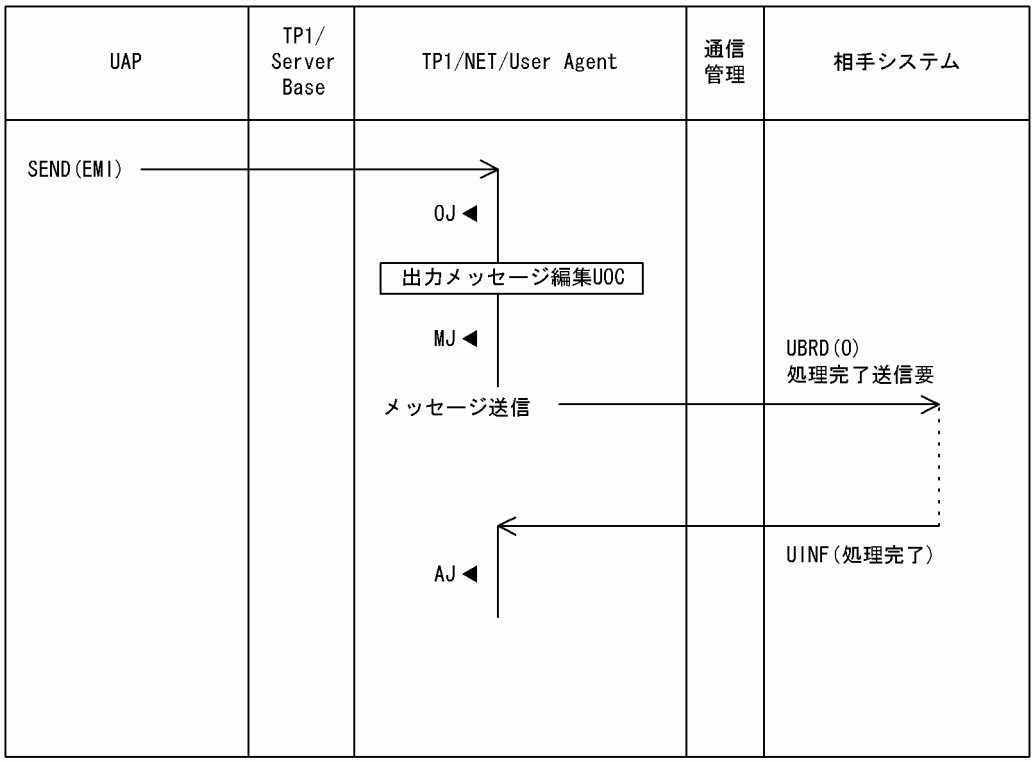
図 D-2 request 型論理端末からの一方送信メッセージの処理の流れ（UA サービスデータ単位分割ありの場合）



(2)　一方送信メッセージ送信時の処理の流れ（send 型論理端末）

(a)　UA サービスデータ単位分割なしの場合

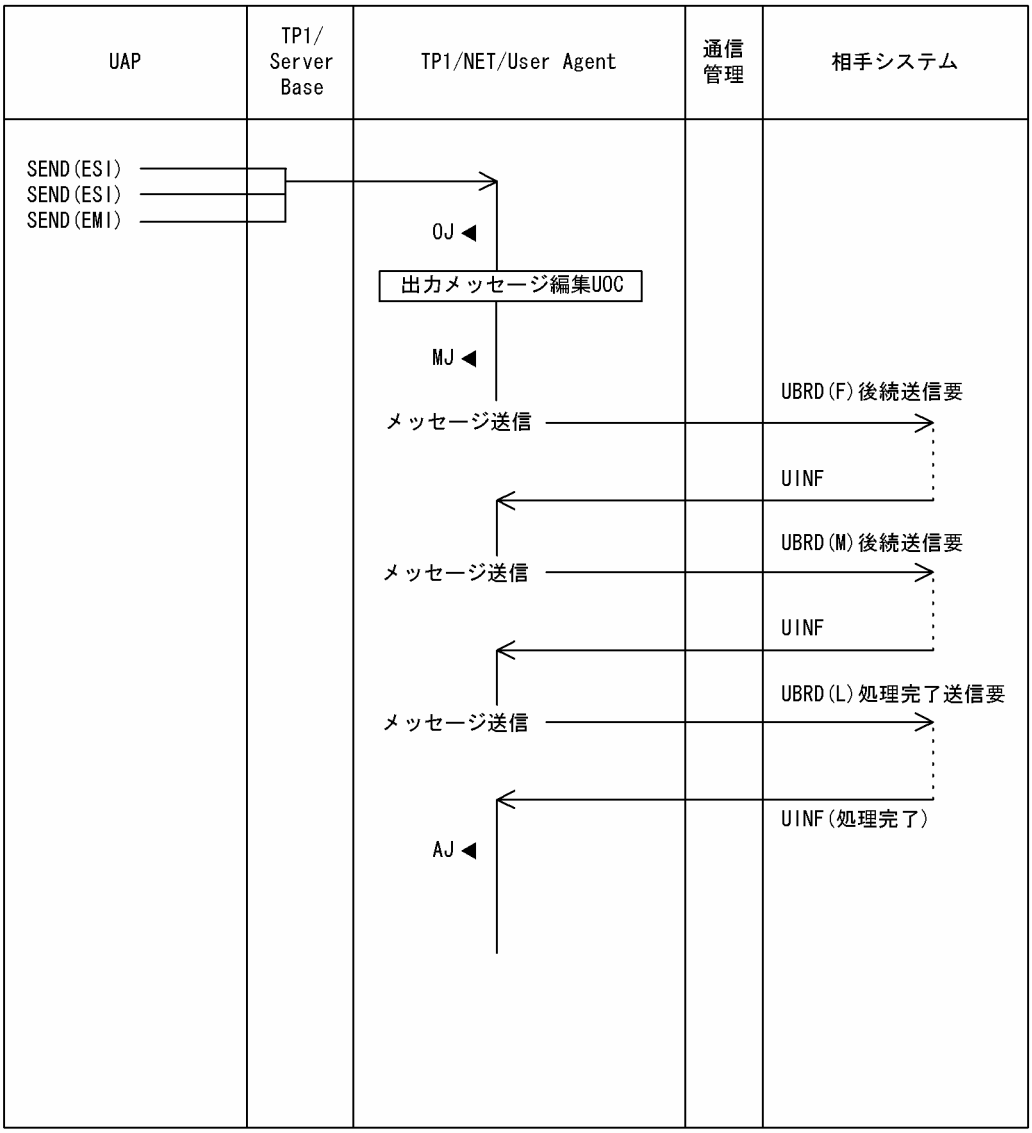
図 D-3　send 型論理端末からの一方送信メッセージの処理の流れ（UA サービスデータ単位分割なしの場合）



（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得
AJ◀：メッセージ送信完了ジャーナル取得

(b)　UA サービスデータ単位分割ありの場合

図 D-4　send 型論理端末からの一方送信メッセージの処理の流れ（UA サービスデータ単位分割ありの場合）

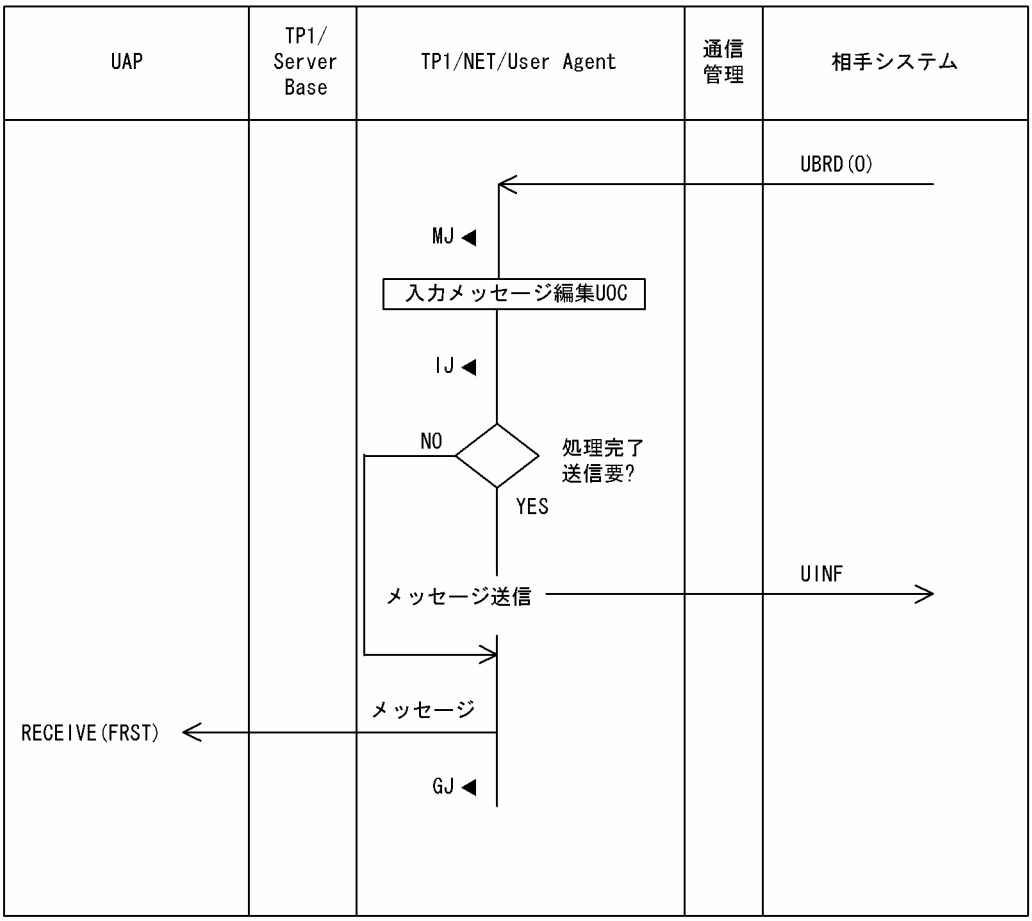


（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得
AJ◀：メッセージ送信完了ジャーナル取得

(3)　一方送信メッセージ受信時の処理の流れ

(a)　UA サービスデータ単位分割なしの場合

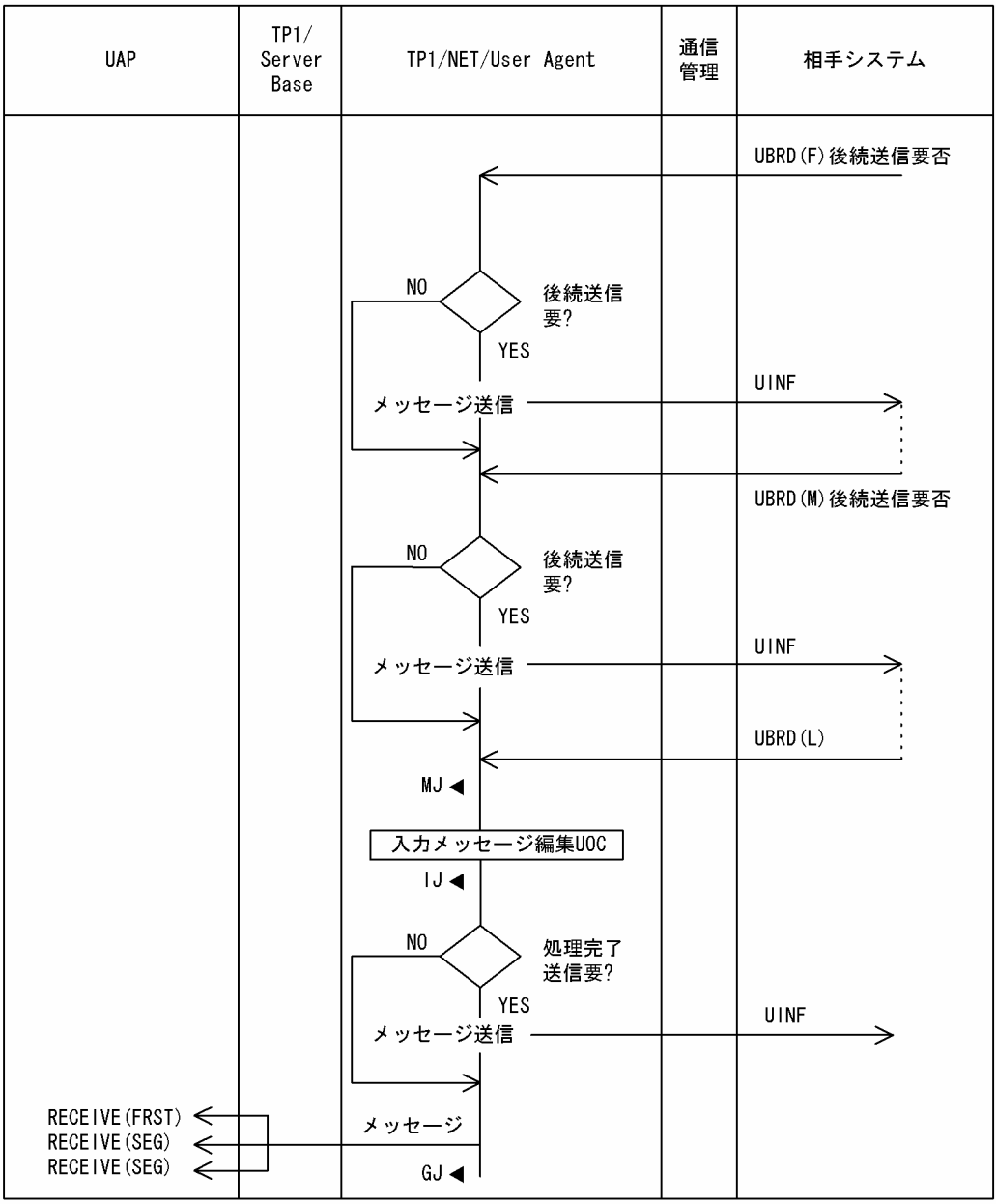
図 D-5　receive 型論理端末への一方送信メッセージの処理の流れ（UA サービスデータ単位分割なしの場合）



（凡例）
MJ◀：メッセージジャーナル取得
IJ◀：メッセージ入力ジャーナル取得
GJ◀：メッセージ受信ジャーナル取得

(b) UA サービスデータ単位分割ありの場合

図 D-6 receive 型論理端末への一方送信メッセージの処理の流れ（UA サービスデータ単位分割ありの場合）



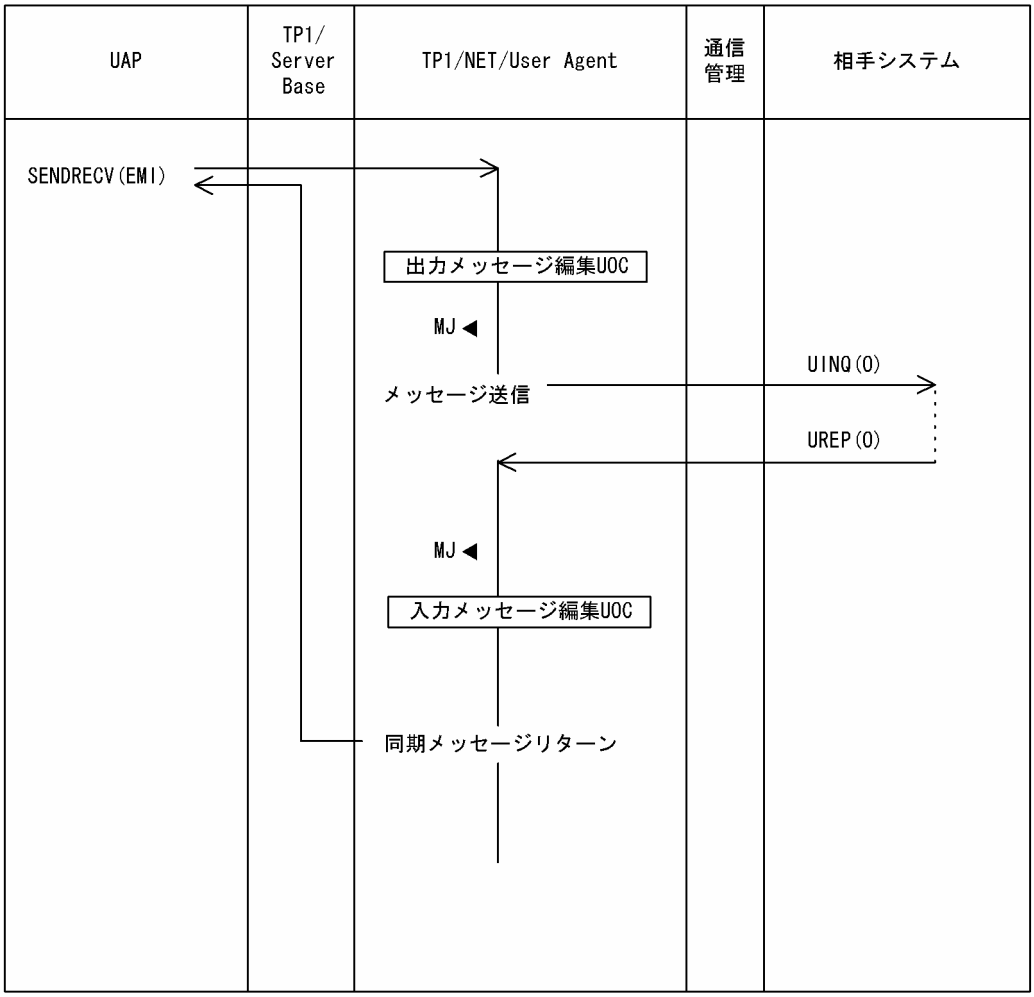
（凡例）
MJ ◀: メッセージジャーナル取得
IJ ◀: メッセージ入力ジャーナル取得
GJ ◀: メッセージ受信ジャーナル取得

付録 D.2　問い合わせメッセージ，および応答メッセージ

(1)　問い合わせメッセージ送信時の処理の流れ

(a)　UA サービスデータ単位分割なしの場合

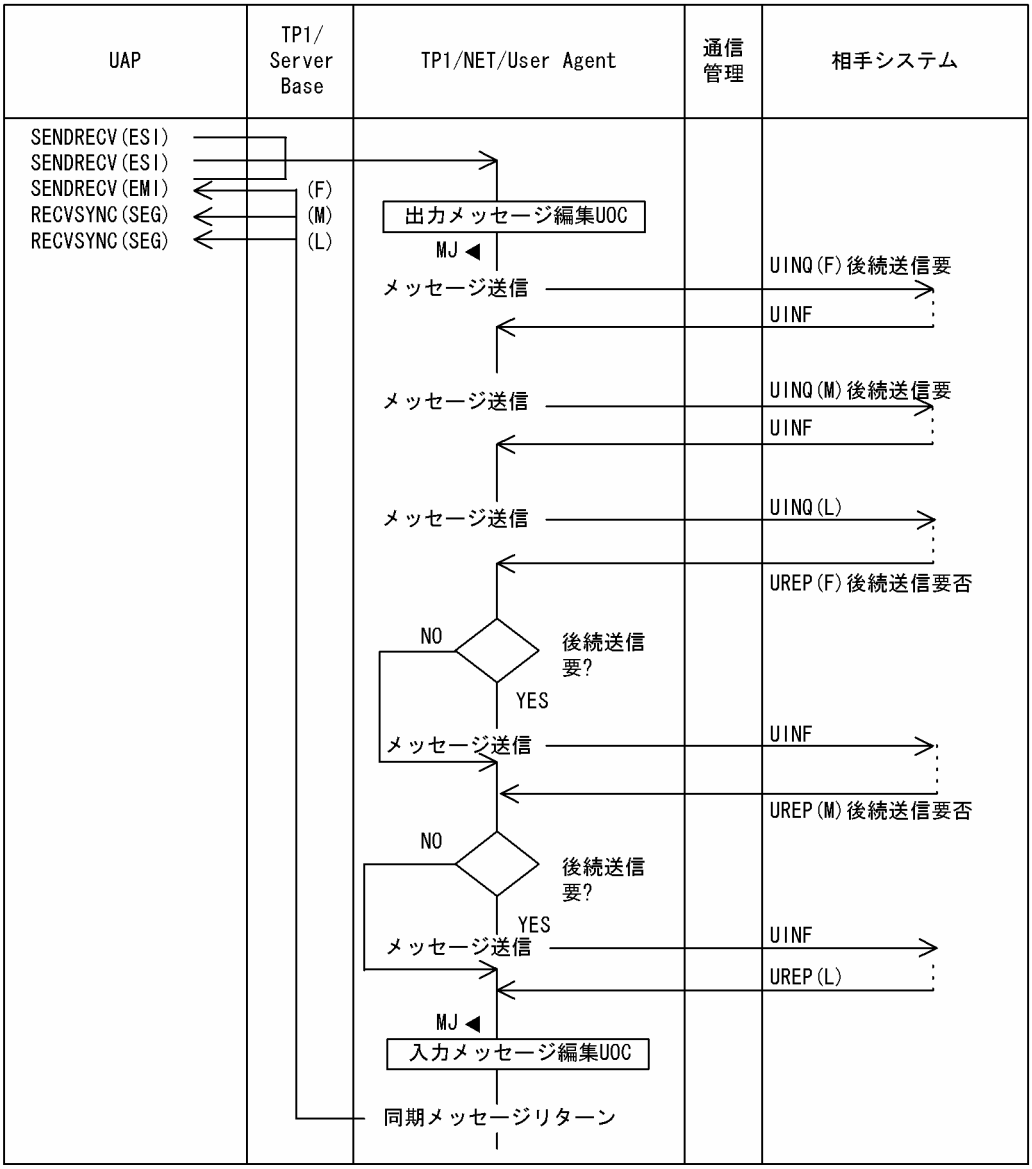
図 D-7　問い合わせメッセージ送信時の処理の流れ（UA サービスデータ単位分割なしの場合）



（凡例）
MJ◀：メッセージジャーナル取得

(b)　UA サービスデータ単位分割ありの場合

図 D-8　問い合わせメッセージ送信時の処理の流れ（UA サービスデータ単位分割ありの場合）

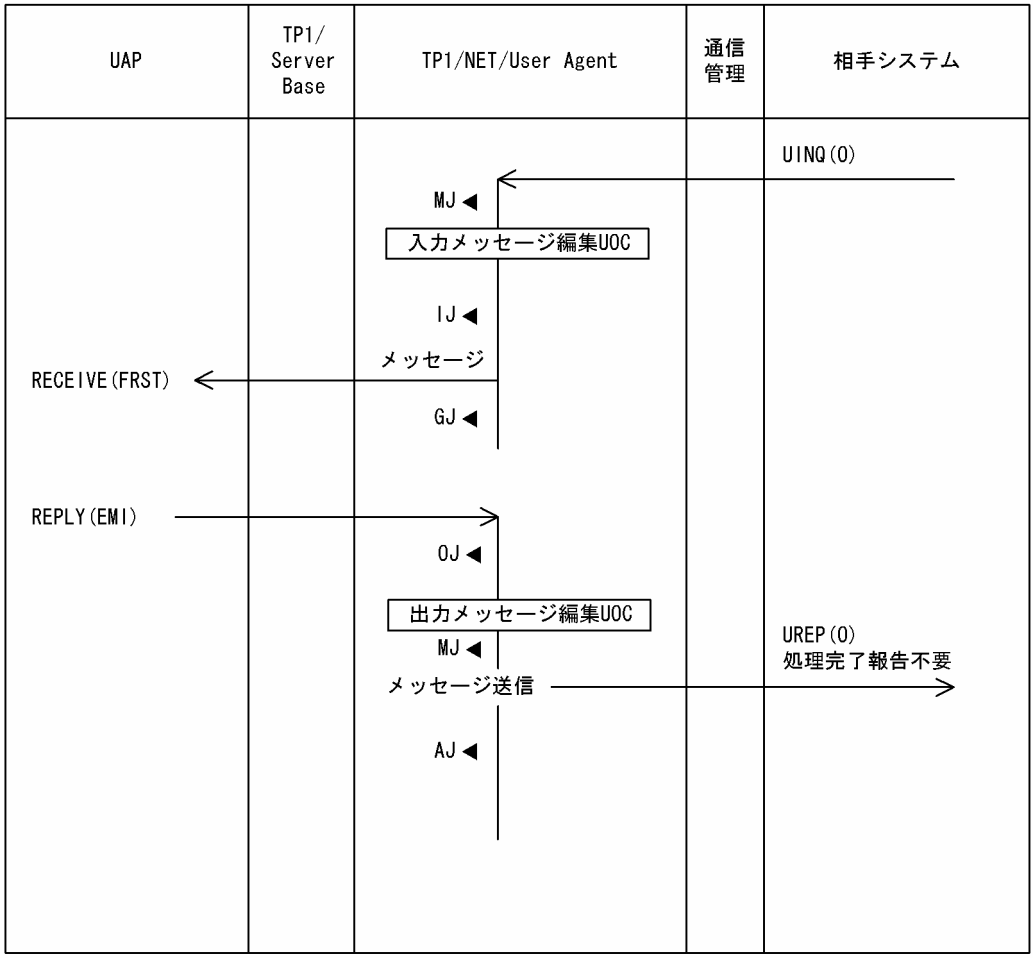


（凡例）
MJ◀：メッセージジャーナル取得

(2)　応答メッセージ送信時の処理の流れ

(a)　UA サービスデータ単位分割なしの場合

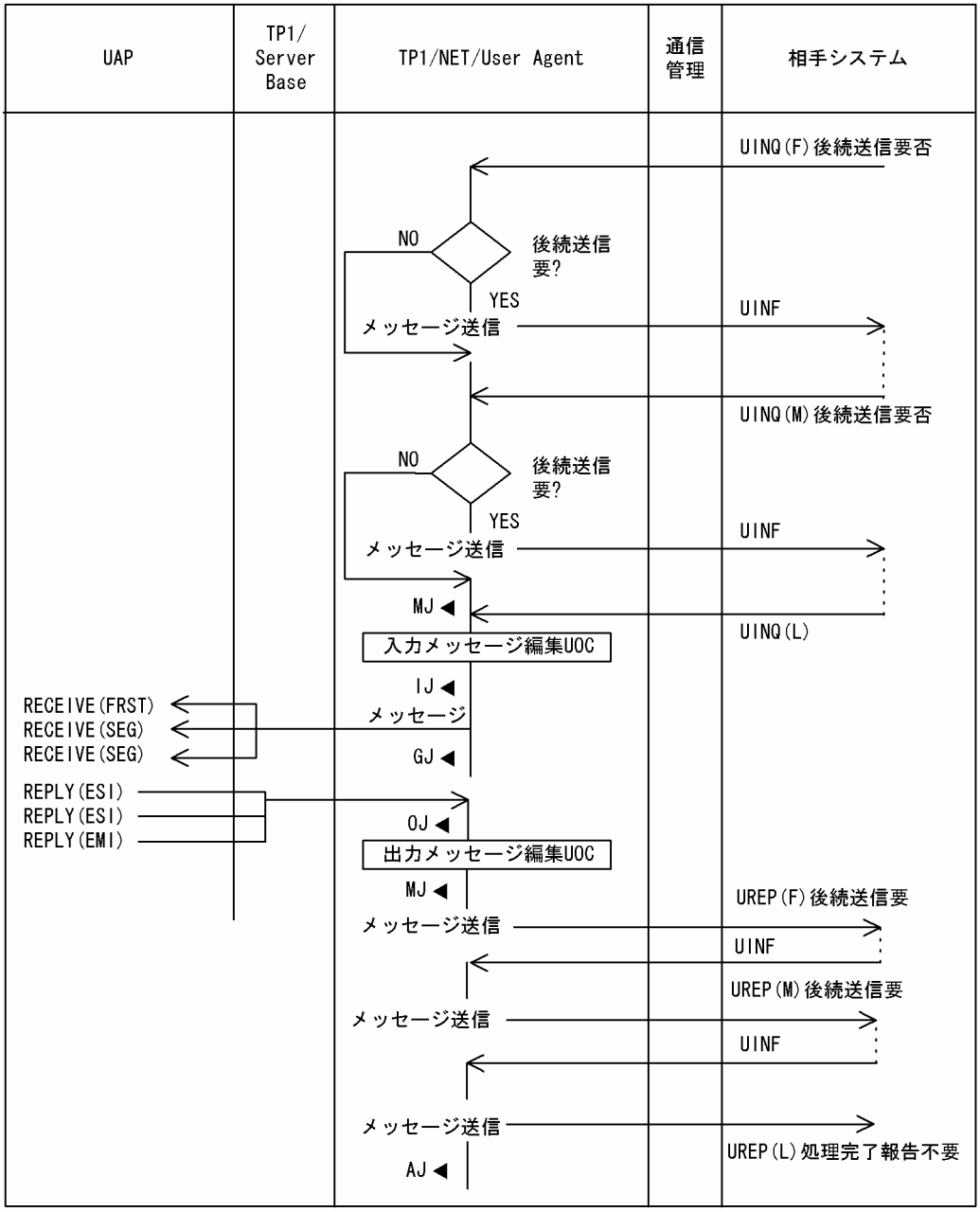
図 D-9　応答メッセージ送信時の処理の流れ（UA サービスデータ単位分割なしの場合）



（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得
AJ◀：メッセージ送信完了ジャーナル取得
IJ◀：メッセージ入力ジャーナル取得
GJ◀：メッセージ受信ジャーナル取得

(b)　UA サービスデータ単位分割ありの場合

図 D-10　応答メッセージ送信時の処理の流れ（UA サービスデータ単位分割ありの場合）



（凡例）
OJ◀ ：メッセージ出力ジャーナル取得
MJ◀ ：メッセージジャーナル取得
AJ◀ ：メッセージ送信完了ジャーナル取得
IJ◀ ：メッセージ入力ジャーナル取得
GJ◀ ：メッセージ受信ジャーナル取得

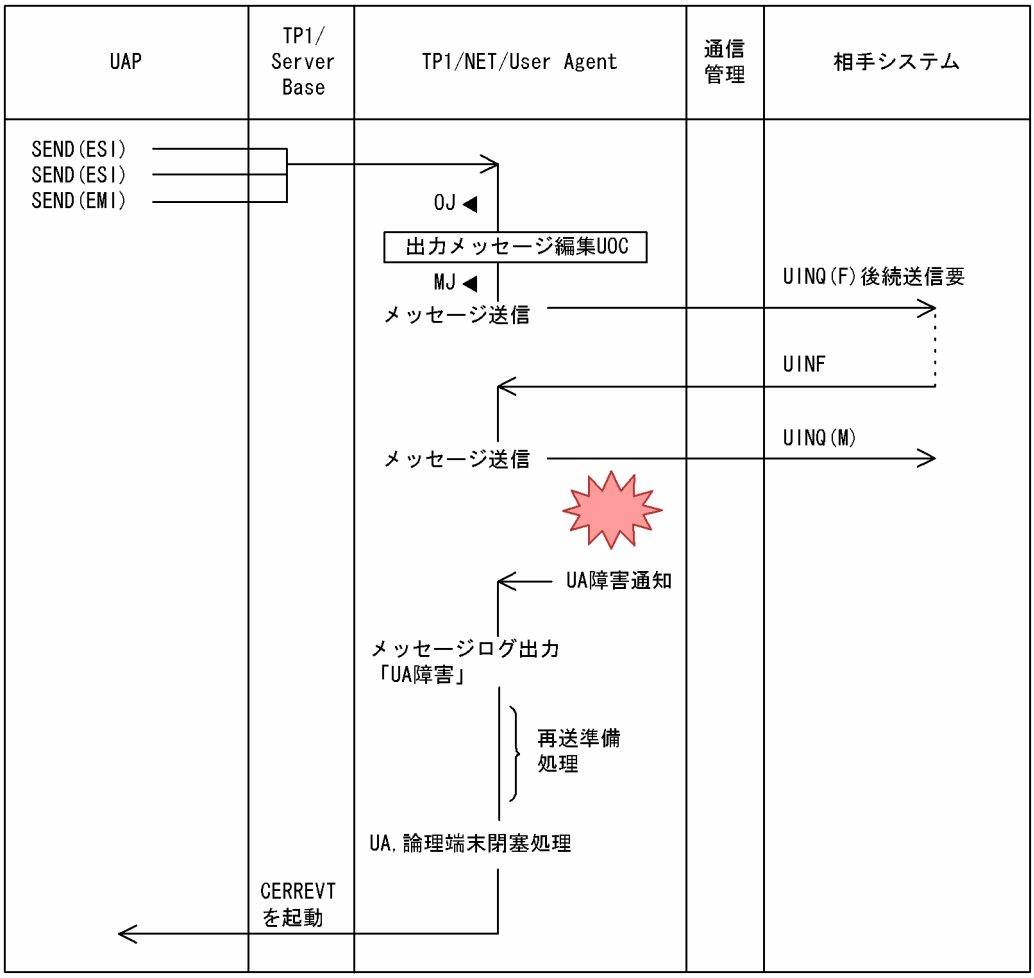
付録 E　障害発生時の処理の流れ

メッセージを送受信するときに発生した障害処理の流れについて示します。

付録 E.1　一方送信メッセージ障害

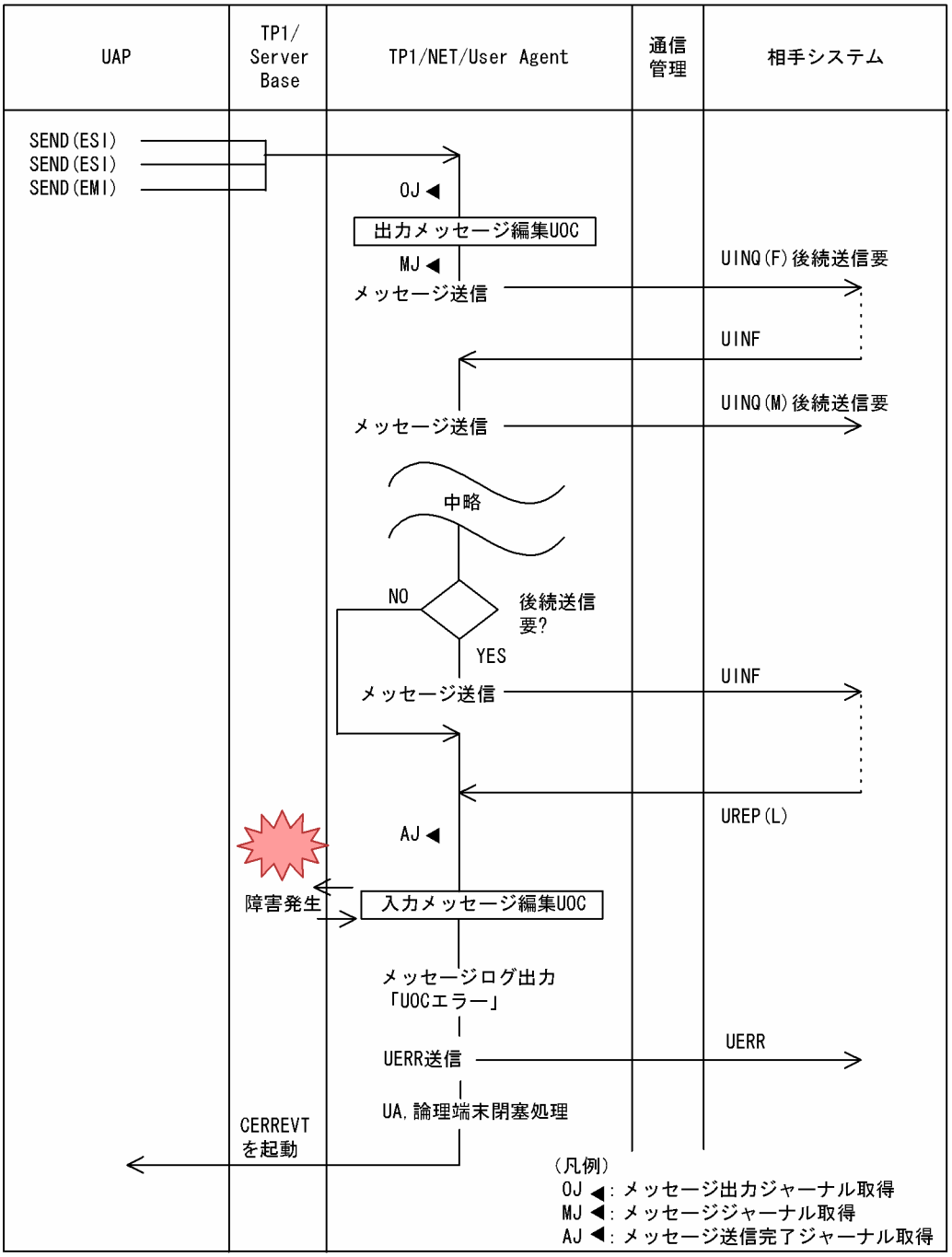
(1)　UINQ 送信時の UA 障害

図 E-1　UINQ 送信時の UA 障害の処理の流れ（一方送信メッセージ障害の場合）



(2)　入力メッセージ編集 UOC エラー

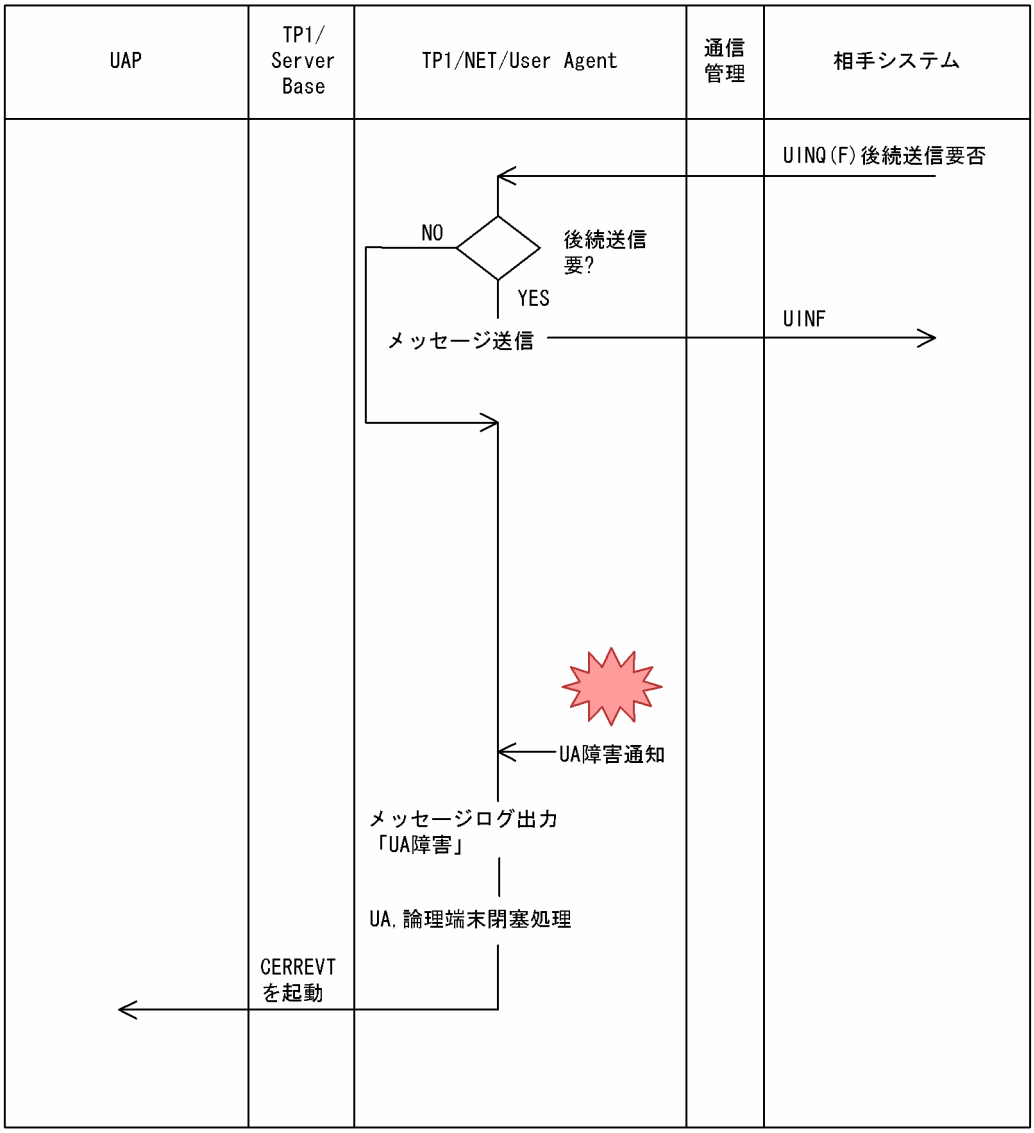
図 E-2　入力メッセージ編集 UOC エラーの処理の流れ（一方送信メッセージ障害の場合）



付録 E.2　応答メッセージ障害

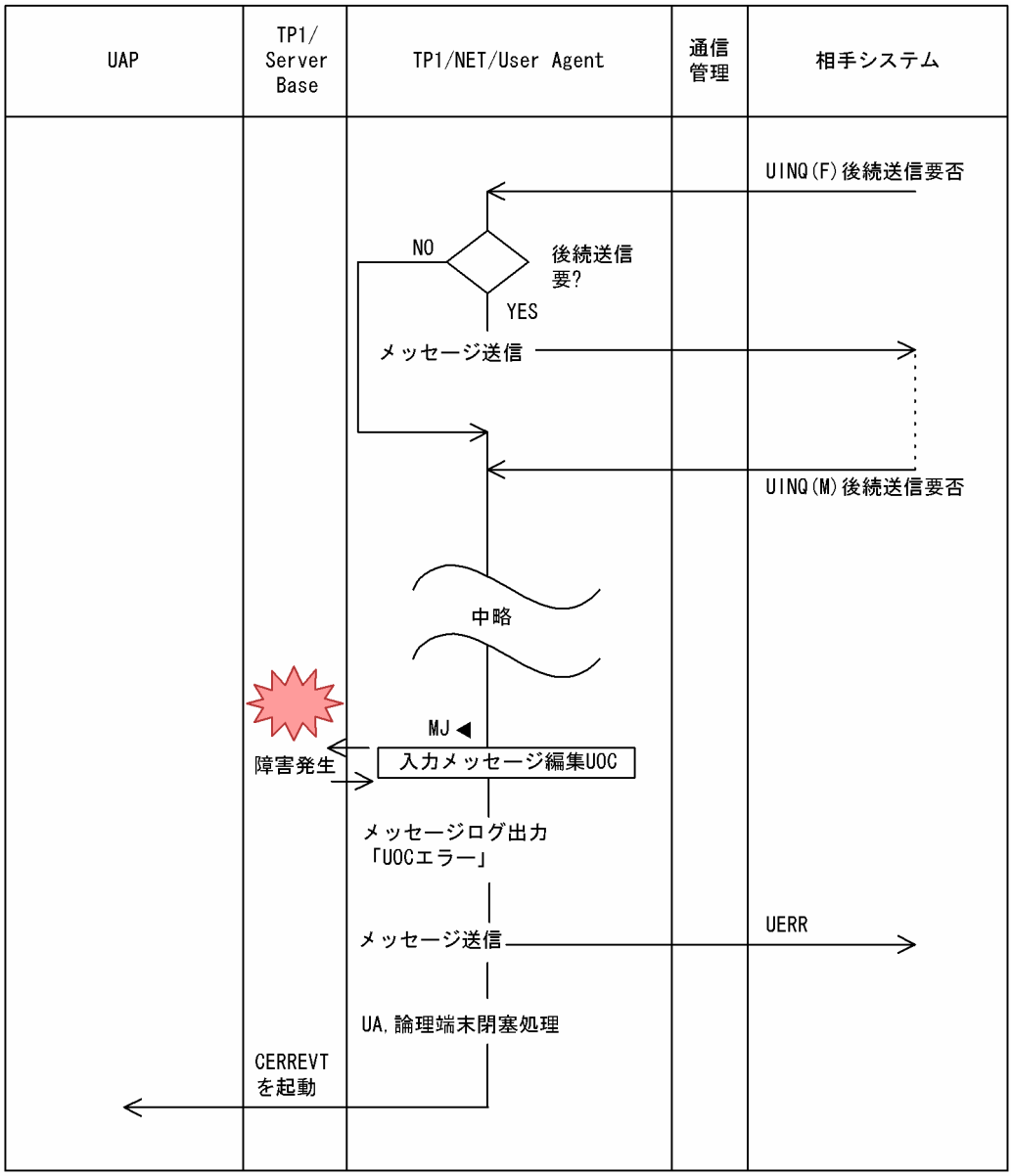
(1)　UINQ 受信時の UA 障害

図 E-3　UINQ 受信時 UA 障害の処理の流れ（応答メッセージ障害の場合）



(2)　入力メッセージ編集 UOC エラー

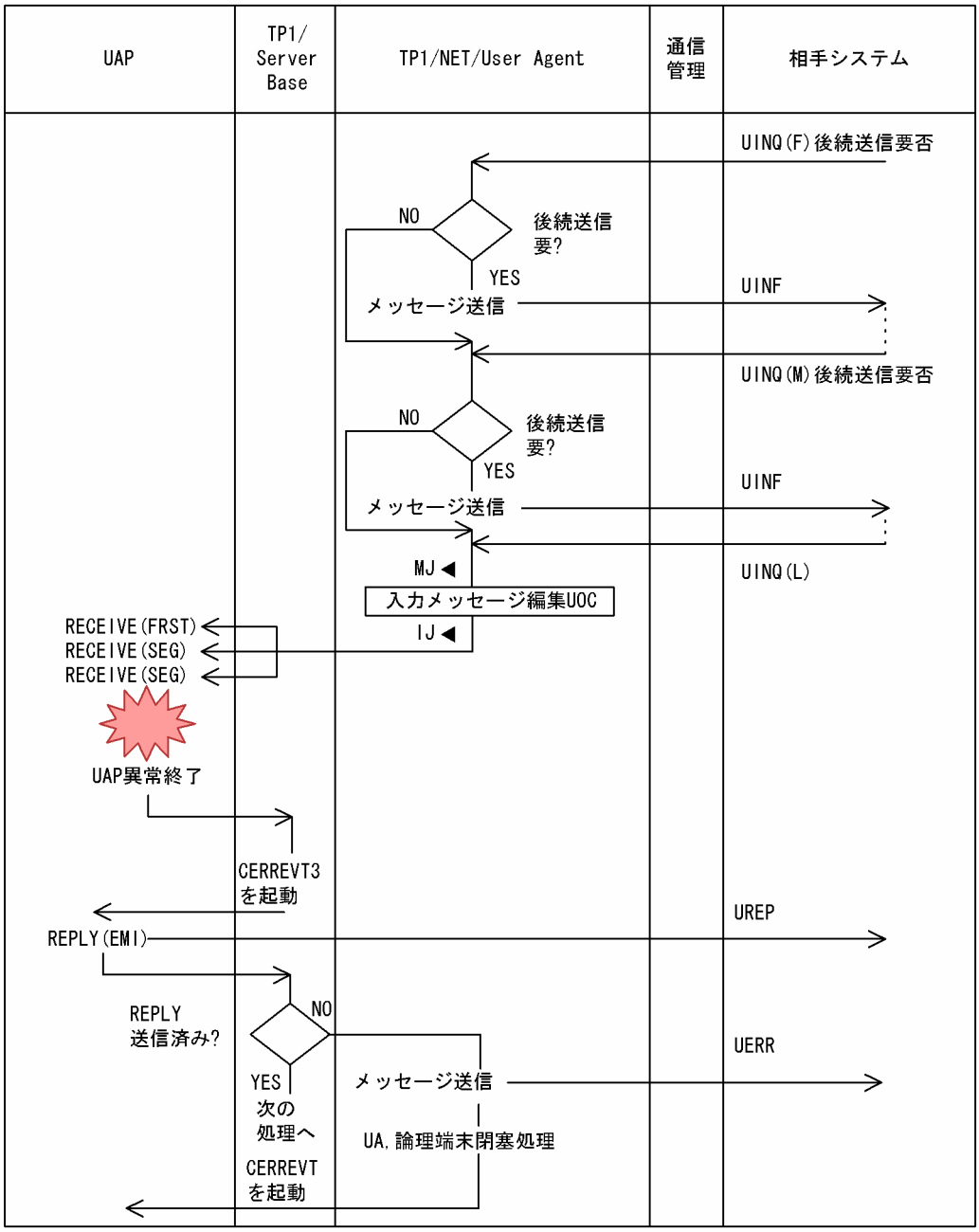
図 E-4　入力メッセージ編集 UOC エラーの処理の流れ（応答メッセージ障害の場合）



（凡例）
MJ◀：メッセージジャーナル取得

(3)　UAP 異常終了

図 E-5　UAP 異常終了の処理の流れ（応答メッセージ障害の場合）

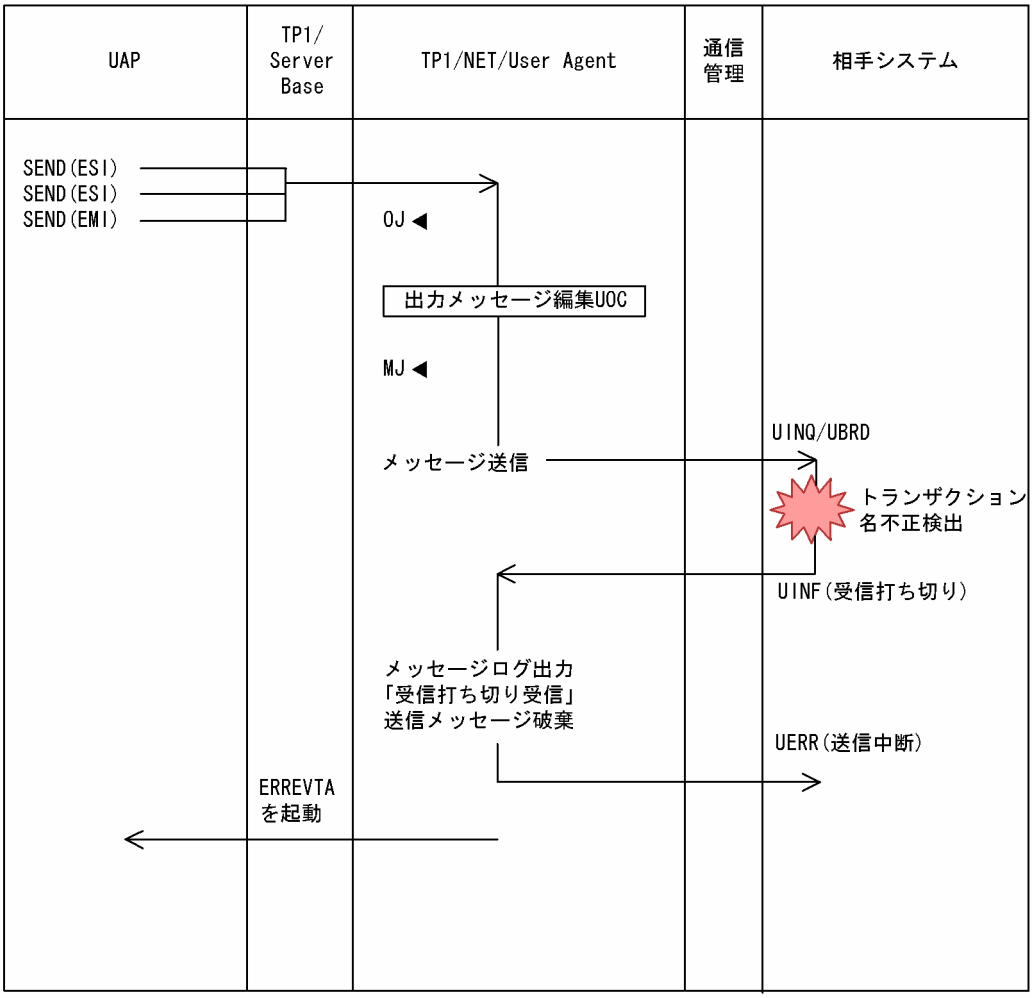


（凡例）
MJ◀：メッセージジャーナル取得
IJ◀：メッセージ入力ジャーナル取得

付録 E.3　相手システム（XDM/DCCM3）接続時の障害

(1)　受信打ち切りを受信

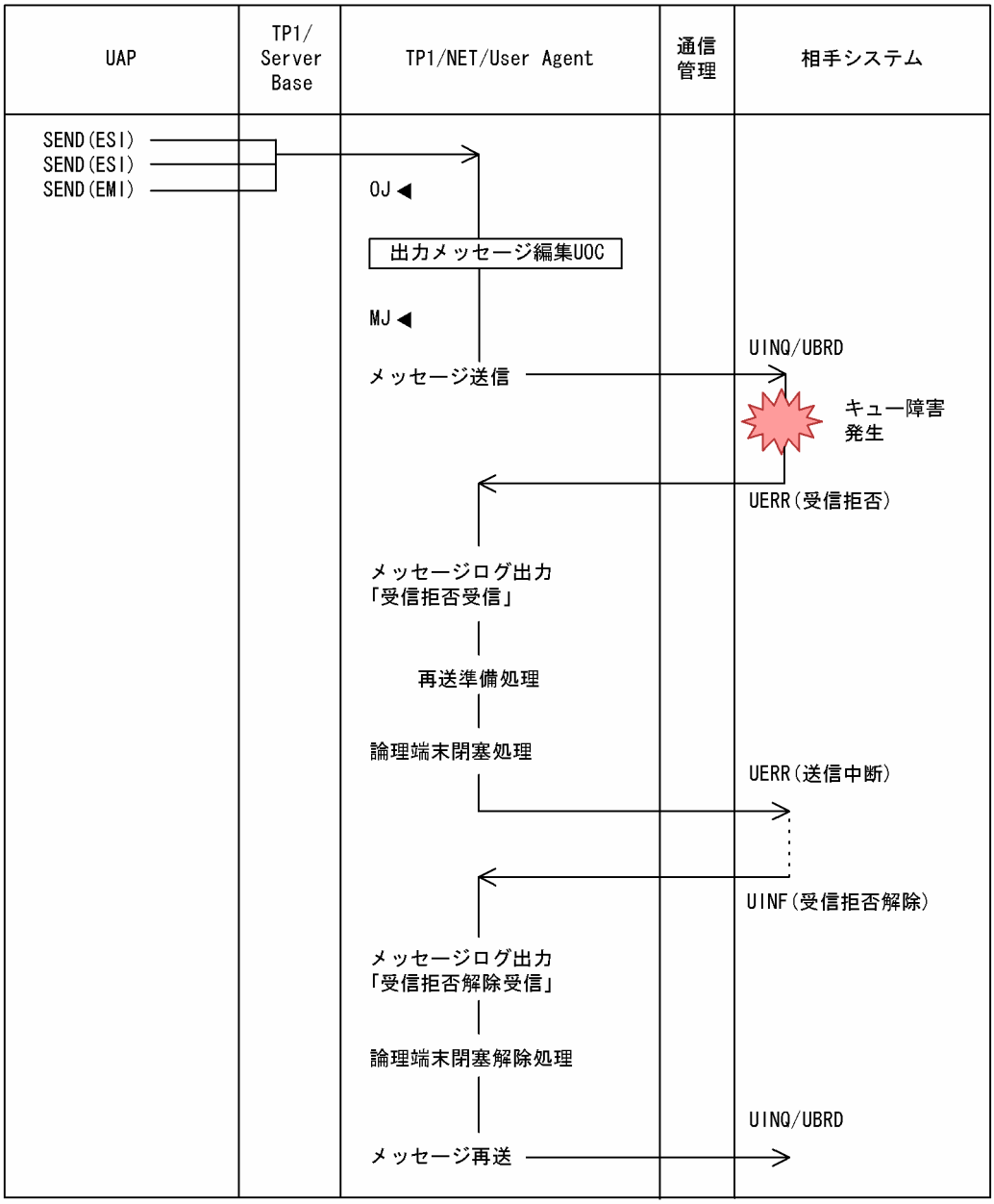
図 E-6　受信打ち切り受信時の処理の流れ（相手システム接続時の障害の場合）



（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得

(2)　受信拒否・受信拒否解除を受信

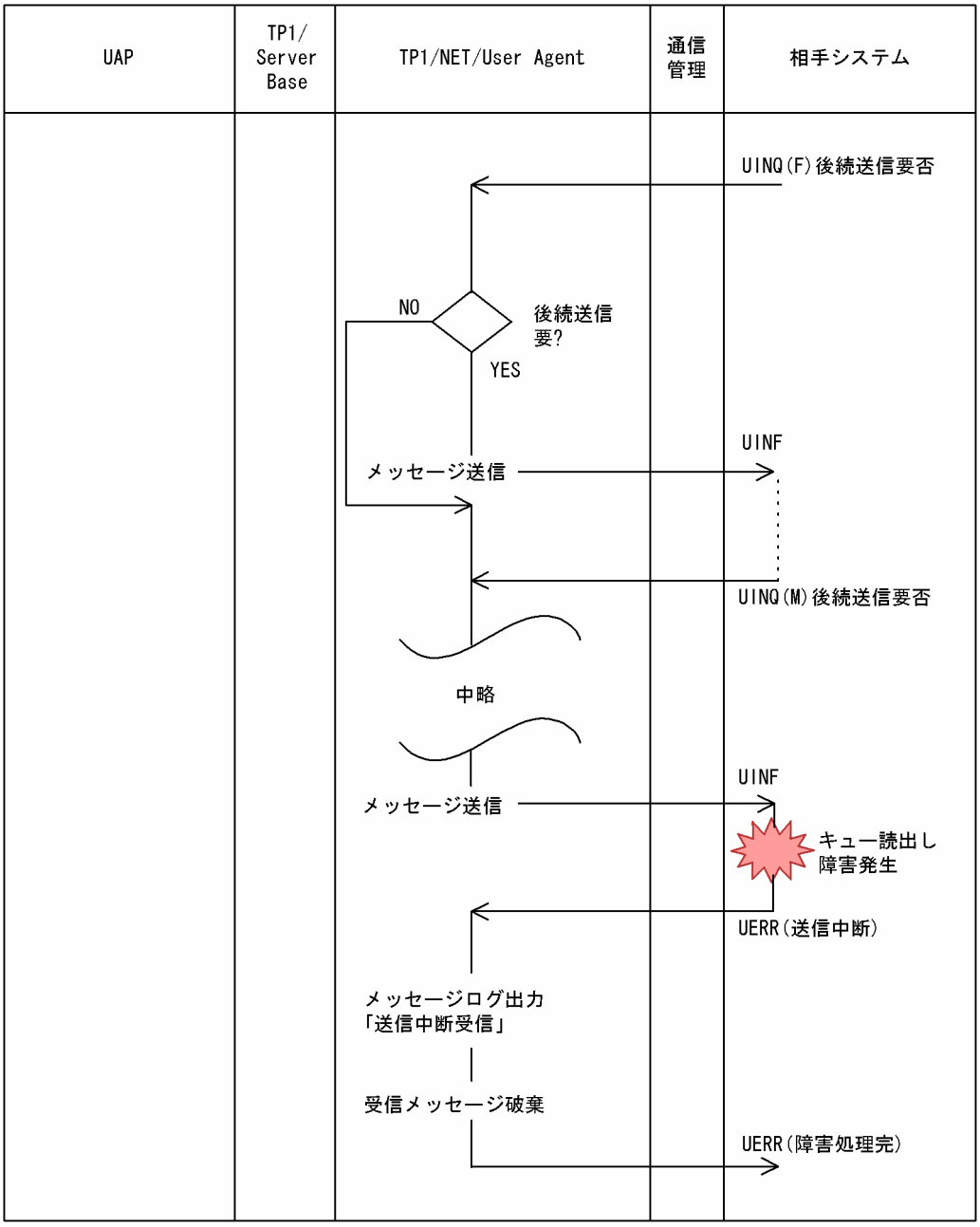
図 E-7　受信拒否・受信拒否解除を受信時の処理の流れ（相手システム接続時の障害の場合）



（凡例）
OJ◀：メッセージ出力ジャーナル取得
MJ◀：メッセージジャーナル取得

(3)　送信中断を受信

図 E-8　送信中断受信時の処理の流れ（相手システム接続時の障害の場合）



付録 F　MCF 性能検証用トレースの取得

TP1/Message Control を使用したメッセージ送受信での主なイベントで，MCF 識別子などのトレース情報を取得しています。これを MCF 性能検証用トレースと呼びます。

ここでは，MCF 性能検証用トレースの MCF 固有情報の出力情報，取得タイミング，および取得量について説明します。

付録 F.1　MCF 固有情報の出力情報

ここでは，UOC 呼び出し時の MCF 性能検証用トレースのダンプ出力情報について説明します。

(1)　UOC 呼び出し時

TP1/NET/User Agent で使用する UOC の情報を取得します。MCF 固有情報のダンプ出力情報，および UOC 名称の出力情報を，以降の表に示します。

表 F-1　UOC 呼び出し時の MCF 固有情報のダンプ出力情報

イベント ID	オフセット				
	0x0000～0x0003	0x0004～0x0007	0x0008～0x000f	0x0010～0x0017	0x0018～0x001f
0xa070	MCF 識別子	スレッド ID	－	－	UOC 名称
0xa071			－	－	

（凡例）

－：情報を取得しません。

表 F-2　UOC 名称の出力情報

UOC の種類	UOC 名称出力情報
入力メッセージ編集 UOC	"MSGRCV"
出力メッセージ編集 UOC	"MSGSEND"
送信メッセージ通番編集 UOC	"SEND_UOC"

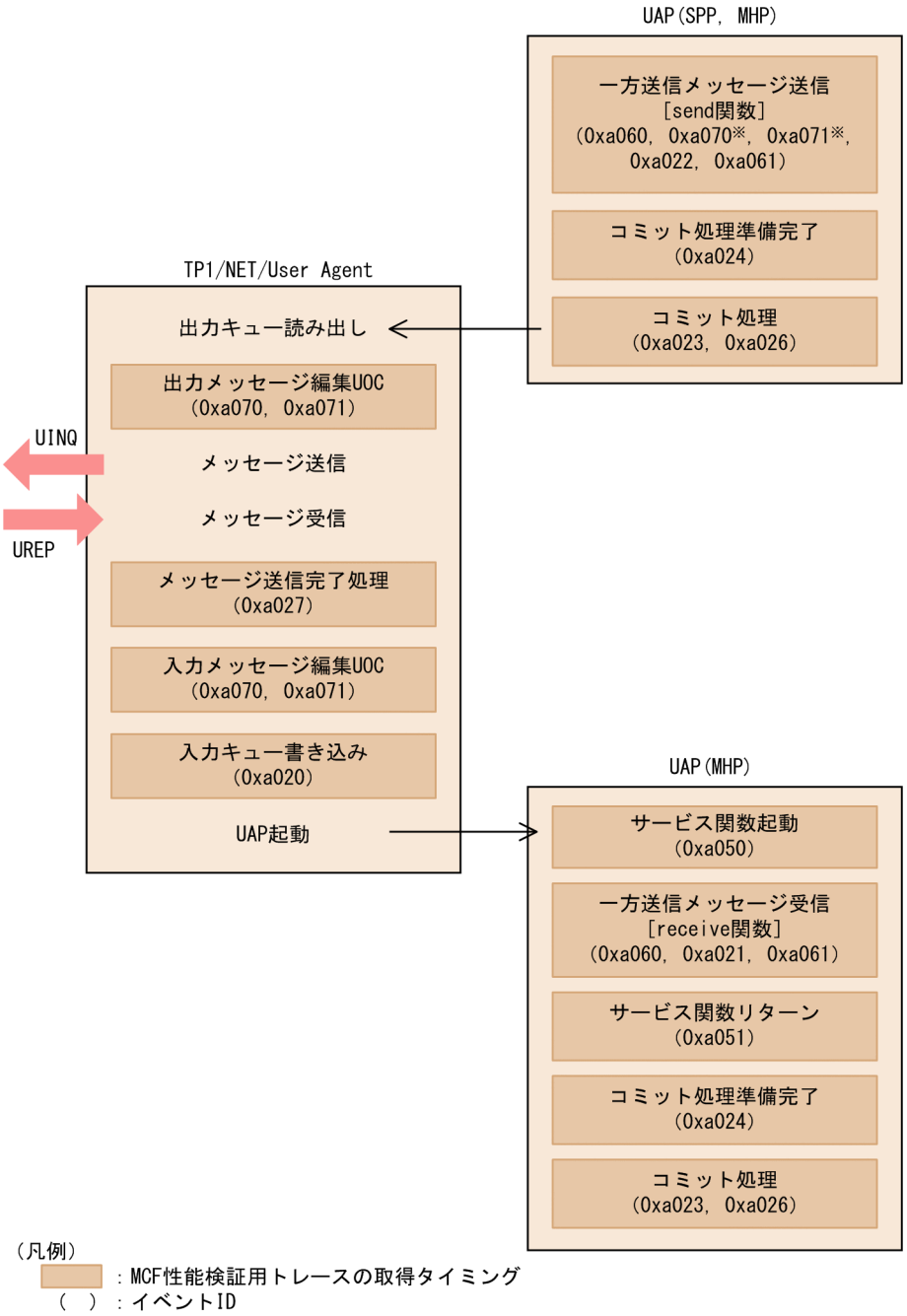
付録 F.2　MCF 性能検証用トレースの取得タイミング

MCF 性能検証用トレースの取得タイミングを，使用する送受信形態別に説明します。

(1)　一方送信メッセージ送信時

一方送信メッセージ送信時の MCF 性能検証用トレースの取得タイミングについて，次の図に示します。

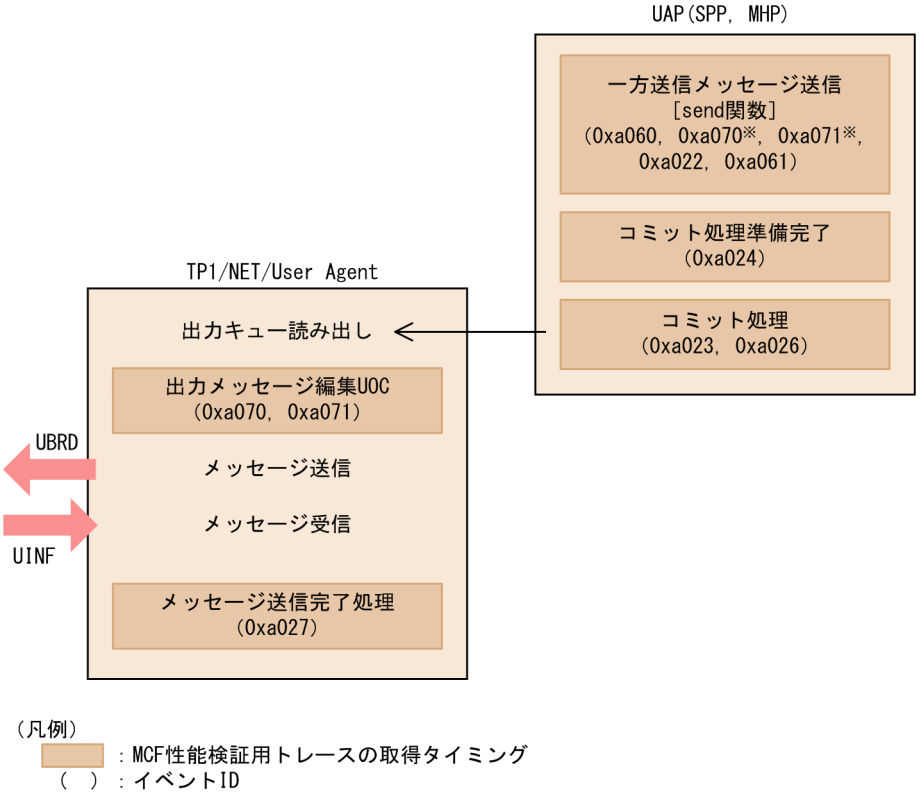
図 F-1　一方送信メッセージ送信時の MCF 性能検証用トレースの取得タイミング（request 型論理端末）



注※

送信メッセージの通番編集 UOC 使用時に該当します。

図 F-2　一方送信メッセージ送信時の MCF 性能検証用トレースの取得タイミング（send 型論理端末）



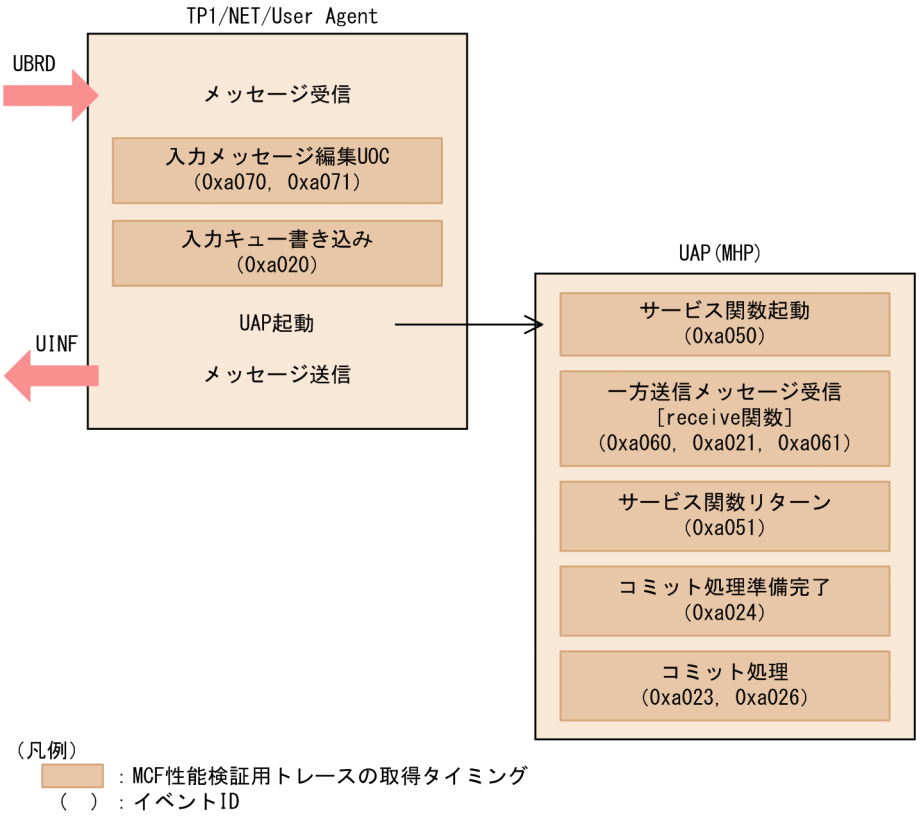
注※

送信メッセージの通番編集 UOC 使用時に該当します。

(2)　一方送信メッセージ受信時

一方送信メッセージ受信時の MCF 性能検証用トレースの取得タイミングについて，次の図に示します。

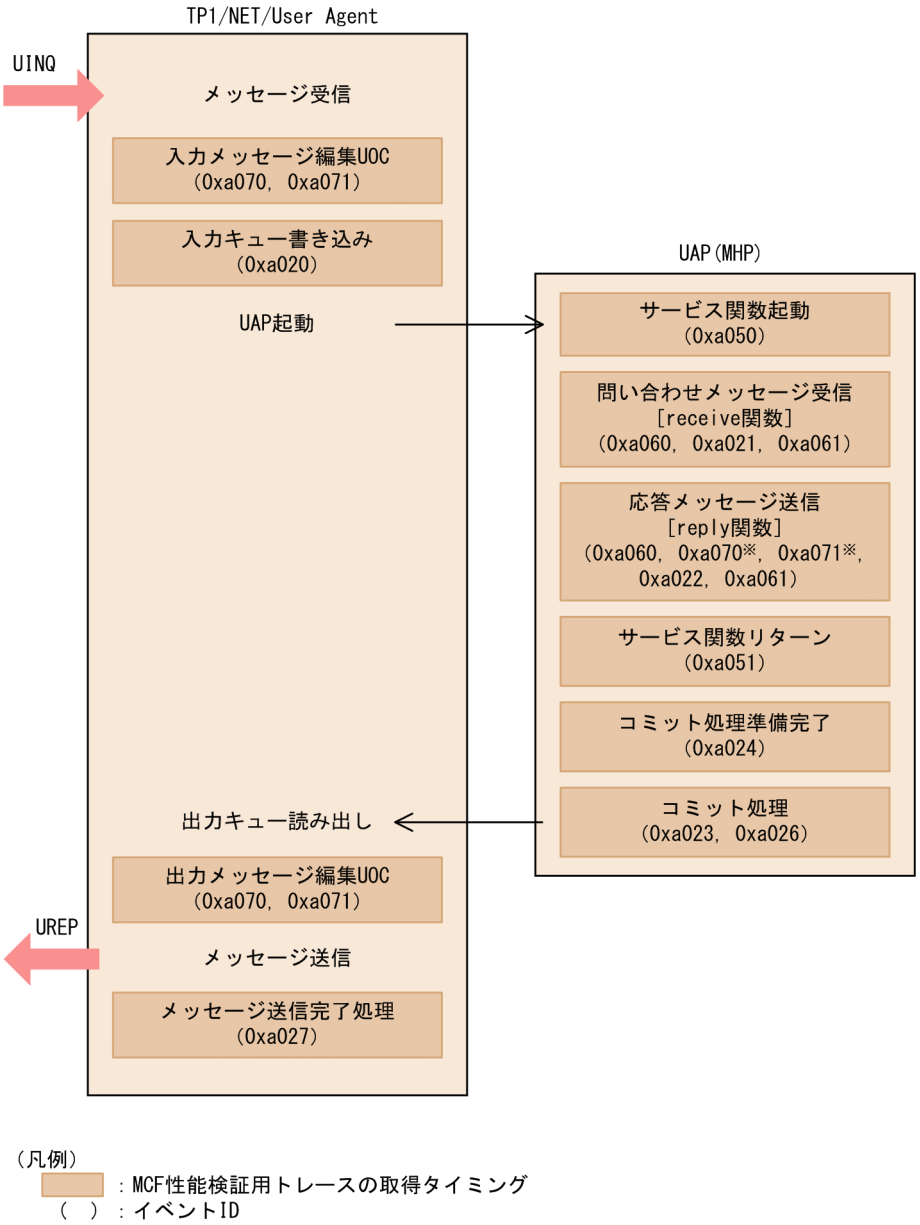
図 F-3　一方送信メッセージ受信時の MCF 性能検証用トレースの取得タイミング



(3)　問い合わせメッセージ受信時

問い合わせメッセージ受信時の MCF 性能検証用トレースの取得タイミングについて，次の図に示します。

図 F-4　問い合わせメッセージ受信時の MCF 性能検証用トレースの取得タイミング



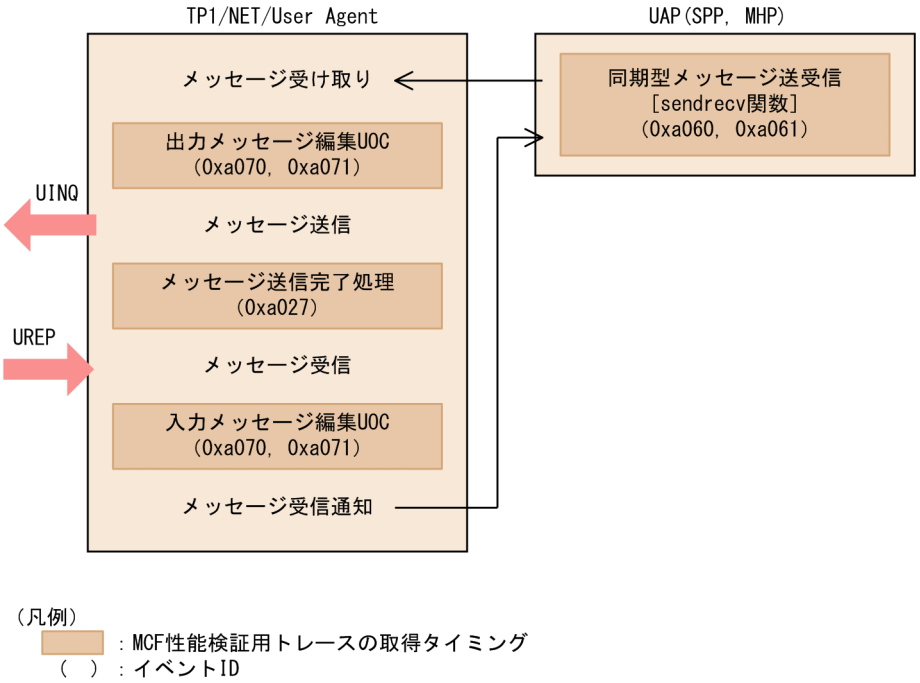
注※

送信メッセージの通番編集 UOC 使用時に該当します。

(4)　同期型メッセージ送受信時

同期型メッセージ送受信時の MCF 性能検証用トレースの取得タイミングについて，次の図に示します。

図 F-5　同期型メッセージ送受信時の MCF 性能検証用トレースの取得タイミング



付録 F.3　MCF 性能検証用トレースの取得量

1 回のメッセージ送受信で取得する MCF 性能検証用トレースのトレース取得量を，次の表に示します。

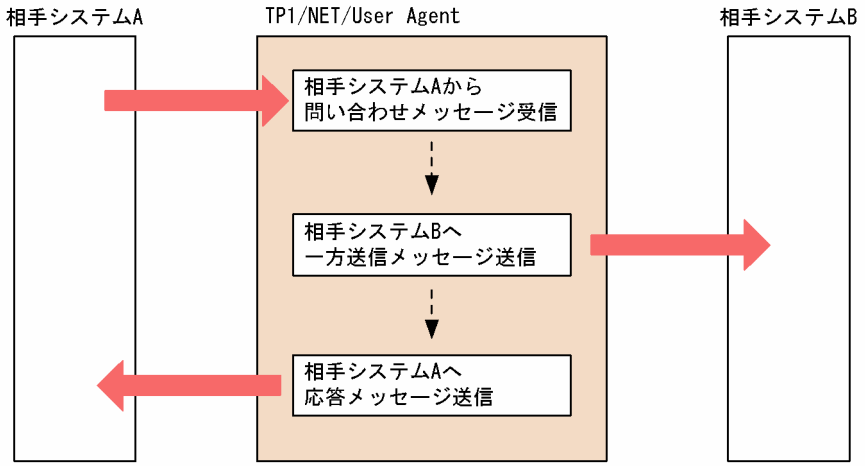
表 F-3　MCF 性能検証用トレースの取得量

メッセージの送受信形態		トレース取得量 (単位：キロバイト)
一方送信メッセージの送信	request 型論理端末	3.7
	send 型論理端末	1.7
一方送信メッセージの受信		2.0
問い合わせメッセージの受信		3.0
同期型のメッセージの送受信		0.9

付録 G　ユーザアプリケーションプログラムの作成例

UAP の作成例として，処理の流れを次の図に示し，その流れに沿ったコーディング例を C 言語（K&R 版）および COBOL 言語で示します。

図 G-1　処理の流れ



付録 G.1　コーディング例

図 G-1 の処理の流れについて，使用する言語ごとに UAP のコーディング例を示します。

(1)　C 言語（K&R 版）

C 言語（K&R 版）を使用した UAP のコーディング例を次に示します。

```
#include <dcmcf.h>
ex_uap1()
{
   char termnam[9] ;
   char recvdata[2048] ;
   DCLONG rdataleng ;
   DCLONG time ;
   DCLONG inbufleng=2048 ;
   char senddata[512] ;
   static char resv01[9]="¥0" ;
   static char resv02[9]="¥0" ;
   static char resv03[9]="¥0" ;
   char *workadd = recvdata ;
                              /* 問い合わせメッセージの受信 */
   dc_mcf_receive(DCMCFFRST,DCNOFLAGS,termnam,resv01,
                  workadd,&rdataleng,2048,&time) ;
   workadd = workadd + rdataleng + 8 ;
   inbufleng = 2048 - (workadd - recvdata) ;
   while(dc_mcf_receive(DCMCFSEG,DCNOFLAGS,termnam,resv01,
                        workadd,&rdataleng,inbufleng,&time)
         == DCMCFRTN_00000)
```

```
      {
         workadd = workadd + rdataleng + 8 ;
         inbufleng = inbufleng - (workadd - recvdata) ;
       }

   /*        データの処理                          */

   dc_mcf_send(DCMCFEMI,DCMCFOUT,"HOSTB",
               resv01,senddata,504,resv02,DCNOFLAGS) ;
                              /* 一方送信メッセージの送信 */
   dc_mcf_reply(DCMCFEMI,DCNOFLAGS,resv03,
                resv01,senddata,504,resv02,DCNOFLAGS) ;
                              /* 応答メッセージの送信 */
}
```

(2) COBOL 言語

COBOL 言語を使用した UAP のコーディング例を次に示します。

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. EXUAP1.

 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  RCV.
     02  MSG-REC       PIC X(8) VALUE 'RECEIVE '.
     02  STATUS-CODE1  PIC X(5).
     02  FILLER        PIC X(3).
     02  SEG-CODE      PIC X(4) VALUE 'FRST'.
     02  RTN-CODE      PIC X(4) VALUE SPACE.
     02  DAY-ID        PIC 9(8).
     02  TIME-ID       PIC 9(8).
     02  SEG-LENG      PIC 9(9) COMP VALUE 2048.
     02  MCFUSE1       PIC X(4) VALUE SPACE.
     02  MCFUSE2       PIC X(4) VALUE SPACE.
     02  MCFUSE3       PIC X(4) VALUE SPACE.
     02  MCFUSE4       PIC X(4) VALUE SPACE.
     02  MCFUSE5       PIC X(8) VALUE SPACE.
     02  MCFUSE6       PIC X(4) VALUE SPACE.
     02  MCFUSE7       PIC X(8) VALUE SPACE.
     02  MCFUSE8       PIC X(4) VALUE SPACE.
     02  MCFUSE9       PIC 9(9) COMP VALUE ZERO.
     02  MCFUSE10      PIC 9(9) COMP VALUE ZERO.
     02  MCFUSE11      PIC X(1) VALUE SPACE.
     02  MCFUSE12      PIC X(1) VALUE SPACE.
     02  MCFUSE13      PIC X(14) VALUE LOW-VALUE.
 01  CD1.
     02  SEG-CODE1     PIC X(4) VALUE SPACE.
     02  TERM-CODE     PIC X(8).
     02  MCFUSE14      PIC X(8) VALUE SPACE.
     02  MCFUSE15      PIC X(8) VALUE SPACE.
     02  MCFUSE16      PIC X(28) VALUE LOW-VALUE.
 01  DATA1.
     02  MSGSEG-LENG1  PIC 9(9) COMP.
     02  MCFUSE17      PIC X(4).
```

```
    02  MCFUSE18      PIC X(2).
    02  MCFUSE19      PIC X(1).
    02  MCFUSE20      PIC X(1).
    02  REC-MSGSEG1   PIC X(2048).
01  SND.
    02  MSG-SEND      PIC X(8) VALUE 'SEND    '.
    02  STATUS-CODE2  PIC X(5).
    02  FILLER        PIC X(3).
    02  MCFUSE21      PIC X(4) VALUE SPACE.
    02  MCFUSE22      PIC X(4) VALUE SPACE.
    02  MCFUSE23      PIC 9(8).
    02  MCFUSE24      PIC 9(8).
    02  MCFUSE25      PIC 9(9) COMP VALUE ZERO.
    02  SEND-SEG      PIC X(4) VALUE 'EMI '.
    02  SEND-SYNC     PIC X(4) VALUE 'ASYN'.
    02  SEND-NORM     PIC X(4) VALUE 'NORM'.
    02  SEND-SEQ      PIC X(4) VALUE 'NSEQ'.
    02  MCFUSE26      PIC X(8) VALUE SPACE.
    02  SEND-CODE     PIC X(4) VALUE SPACE.
    02  MCFUSE27      PIC X(8) VALUE SPACE.
    02  MCFUSE28      PIC X(4) VALUE SPACE.
    02  MCFUSE29      PIC 9(9) COMP VALUE ZERO.
    02  MCFUSE30      PIC 9(9) COMP.
    02  MCFUSE31      PIC X(1) VALUE SPACE.
    02  MCFUSE32      PIC X(1) VALUE SPACE.
    02  MCFUSE33      PIC X(14) VALUE LOW-VALUE.
01  CD2.
    02  SENDSEG-CODE2 PIC X(4) VALUE 'OUT '.
    02  TERM-CODE2    PIC X(8) VALUE 'HOSTB'.
    02  MCFUSE34      PIC X(8) VALUE SPACE.
    02  MCFUSE35      PIC X(8) VALUE SPACE.
    02  MCFUSE36      PIC X(28) VALUE LOW-VALUE.
01  DATA2.
    02  MSGSEG-LENG2  PIC 9(9) COMP VALUE 512.
    02  MCFUSE37      PIC X(8).
    02  REC-MSGSEG2   PIC X(512).
01  RPLY.
    02  MSG-REPLY     PIC X(8) VALUE 'REPLY   '.
    02  STATUS-CODE3  PIC X(5).
    02  FILLER        PIC X(3).
    02  MCFUSE41      PIC X(4) VALUE SPACE.
    02  MCFUSE42      PIC X(4) VALUE SPACE.
    02  MCFUSE43      PIC 9(8).
    02  MCFUSE44      PIC 9(8).
    02  MCFUSE45      PIC 9(9) COMP VALUE ZERO.
    02  SEND-SEG1     PIC X(4) VALUE 'EMI '.
    02  SEND-SYNC1    PIC X(4) VALUE 'ASYN'.
    02  MCFUSE46      PIC X(4) VALUE SPACE.
    02  SEND-SEQ1     PIC X(4) VALUE SPACE.
    02  MCFUSE47      PIC X(8) VALUE SPACE.
    02  SEND-CODE1    PIC X(4) VALUE SPACE.
    02  MCFUSE48      PIC X(8) VALUE SPACE.
    02  MCFUSE49      PIC X(4) VALUE SPACE.
    02  MCFUSE50      PIC 9(9) COMP VALUE ZERO.
    02  MCFUSE51      PIC 9(9) COMP.
    02  MCFUSE52      PIC X(1) VALUE SPACE.
    02  MCFUSE53      PIC X(1) VALUE SPACE.
    02  MCFUSE54      PIC X(14) VALUE LOW-VALUE.
```

```
 01  CD3.
     02  SEND-CODE3    PIC X(4) VALUE SPACE.
     02  TERM-CODE3    PIC X(8) VALUE SPACE.
     02  MCFUSE55      PIC X(8) VALUE SPACE.
     02  MCFUSE56      PIC X(8) VALUE SPACE.
     02  MCFUSE57      PIC X(28) VALUE LOW-VALUE.

 PROCEDURE DIVISION.
 CALL 'CBLDCMCF'USING RCV CD1 DATA1.
*(問い合わせメッセージの受信)
*
*   処理 1
*
 CALL 'CBLDCMCF'USING SND CD2 DATA2.
*(一方送信メッセージの送信)
*
*   処理 2
*
 CALL 'CBLDCMCF'USING RPLY CD3 DATA2.
*(応答メッセージの送信)
*
*   処理 3
*
 EXIT PROGRAM.
```

付録 G.2　提供するサンプルコーディング

ここでは，TP1/NET/User Agent が提供するサンプルコーディングの格納場所について言語ごとに示します。

(1)　C 言語

適用 OS が Linux の場合

/opt/OpenTP1/examples/mcf/UserAgent/aplib/c/ap.c

適用 OS が AIX または HP-UX の場合

/BeTRAN/examples/mcf/UserAgent/aplib/c/ap.c

(2)　COBOL 言語

適用 OS が Linux の場合

/opt/OpenTP1/examples/mcf/UserAgent/aplib/cobol/ap.cbl

適用 OS が AIX または HP-UX の場合

/BeTRAN/examples/mcf/UserAgent/aplib/cobol/ap.cbl

付録 H　理由コード一覧

ERREVT2 発生時の理由コードを表 H-1 に，CERREVT 発生時の理由コードを表 H-2 に示します。

表 H-1　ERREVT2 の理由コード一覧

C 言語の理由コード（16 進数）	COBOL 言語の理由コード（外部 10 進）	ERREVT2 の通知理由
DCMCF_NO_SERV (0010)	0010	アプリケーション名に相当する MHP のサービスがありません。
DCMCF_SCD_ERR (0020)	0020	ユーザサーバ未起動などによって，MHP の起動に失敗しました。
DCMCF_QUE_BUF_ERR (0030)	0030	動的共用メモリが不足しました。
DCMCF_QUE_FIL_OVER (0031)	0031	キューファイルが満杯になりました。
DCMCF_QUE_LIMIT_OVER (0032)	0032	入力メッセージ最大格納数を超過しました。
DCMCF_QUE_IO_ERR (0033)	0033	入力キューに障害が発生しました。
DCMCF_AP_CLOSE (0040)	0040	MHP のアプリケーションが閉塞しています。
DCMCF_AP_SECURE (0041)	0041	MHP のアプリケーションがセキュア状態です。
DCMCF_SERV_CLOSE (0042)	0042	• MHP のサービスまたはサービスグループが閉塞しています。 • スケジュール閉塞されているサービスグループの入力キューに未処理受信メッセージが残った状態で，OpenTP1 を正常終了または計画停止 A で終了しました。
DCMCF_SERV_SECURE (0043)	0043	MHP のサービスグループがセキュア状態です。
DCMCF_ABNORMAL_END (0050)	0050	• MHP で呼び出す receive 関数にセグメントを渡す前に，MHP が異常終了しました。 • DBMS の障害などによって，トランザクションの開始に失敗しました。

表 H-2　CERREVT の理由コード一覧

理由コード 1 （16 進数）	理由コード 2 （16 進数）	発生条件	発生個所
DCMOUM_RSN1_MCF （00000001）	DCMOUM_RSN2_ITQ （00000000）	メッセージ入力障害	UA，論理端末
	DCMOUM_RSN2_APL （00000001）	アプリケーション名取得障害	UA，論理端末
	DCMOUM_RSN2_OTGET （00000002）	メッセージ出力障害	UA，論理端末
	DCMOUM_RSN2_OTCMP （00000003）	メッセージ送信完了処理障害	UA，論理端末
	DCMOUM_RSN2_DCTLE （00000004）	mcftdctle による UA，論理端末の閉塞	UA，論理端末
	DCMOUM_RSN2_UAPAB （00000005）	reply 型 UAP 異常終了による UERR 送信	UA，論理端末
	DCMOUM_RSN2_SYCER （00000006）	UAP への同期リターン失敗	UA，論理端末
	DCMOUM_RSN2_ENDER （00000007）	終了処理中のメッセージ受信による UERR 送信（入力抑止）	UA，論理端末
	DCMOUM_RSN2_OPNRJ （00000008）	UA 開局要求拒否	UA，論理端末
	（11ff000a）	UINQ を受信できない端末タイプ（論理端末定義（mcftalcle -t）で指定）の論理端末で UINQ を受信	論理端末
	（11ff000e）	UBRD を受信できない端末タイプ（論理端末定義（mcftalcle -t）で指定）の論理端末で UBRD を受信	論理端末
	DCMOUM_RSN2_DCTCN_F （20000000）	mcftdctcn -f による強制解放	コネクション
	DCMOUM_RSN2_NOBUF （20000002）	バッファ取得失敗による強制解放	コネクション
	その他	上記以外の障害 （理由コード 2 は保守情報）	UA，論理端末，コネクション
DCMOUM_RSN1_UERR （00000002）	RRRRDDDD （受信障害コード）	UERR 送受信，または下位層障害	UA，論理端末，コネクション

理由コード 1 (16 進数)	理由コード 2 (16 進数)	発生条件	発生個所
DCMOUM_RSN1_UOC (00000003)	詳細リターンコード	ユーザ（UOC）検出障害	論理端末，コネクション
DCMOUM_RSN1_UAP (00000004)	RRRRDDDD (送信障害コード)	send（エラー送信）による閉局	UA，論理端末
DCMOUM_RSN1_ACTER (00000005)	RRRRDDDD (受信障害コード)	コネクションの確立失敗 (UERR 送受信，または下位層障害)	コネクション
DCMOUM_RSN1_ACTLERJ (00000006)	RRRR0000 (拒否理由)	個別開局の拒否応答の受信	UA，論理端末
その他	不定	上記以外の障害 (理由コード 1，2 は保守情報)	UA，論理端末，コネクション

(凡例)

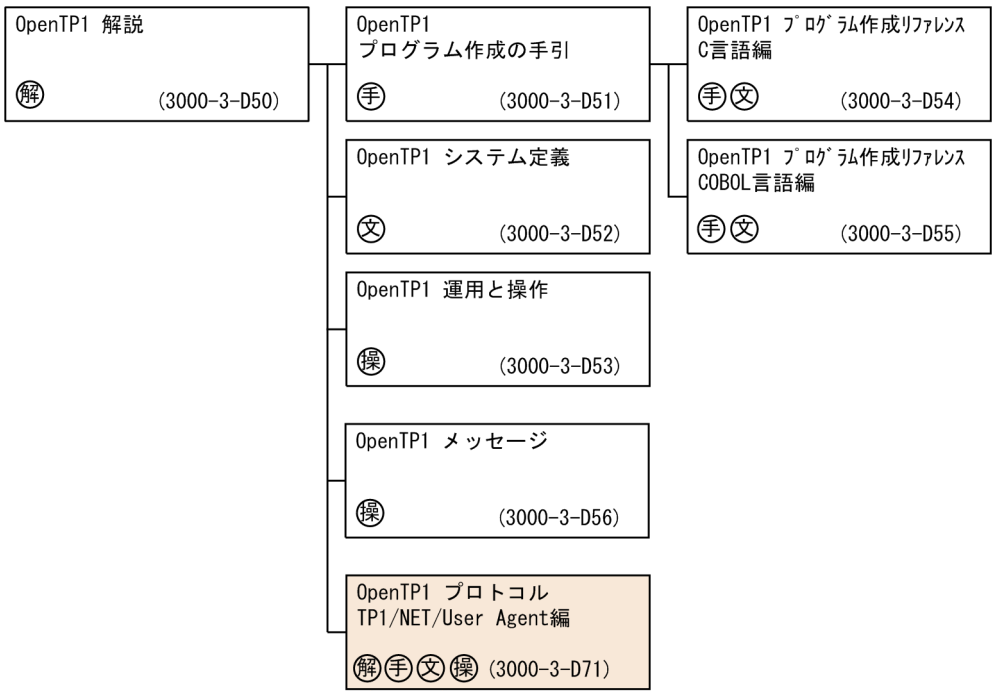
RRRR：障害理由または拒否理由（下位層障害時は 0000 を設定します。）

DDDD：OSAS/UA プロトコルで規定される障害理由詳細（下位層障害時は 0000 を設定します。）

付録 I　このマニュアルの参考情報

付録 I.1　関連マニュアル

●OpenTP1 Version 7



●通信管理

- 通信管理 XNF/AS 解説・運用編（3000-3-B61）
- 通信管理 XNF/AS 構成定義編（3000-3-B62）
- 通信管理 XNF/AS NSAP アドレス概説編(3000-3-B43)
- 通信管理 XNF/LS 使用の手引（3000-3-B51）

●相手システム（VOS3)

- TMS-4V/SP OSAS/AP 間通信サービス-UA OSAS/UA/4VSP（6190-6-126）
- OSI アプリケーション共通機能/AP 間通信サービス/DCCM3 OSAS/UA2/DCCM3（6190-6-371）
- XNF TCP/IP 接続機能 XNF/TCP 定義とコマンド（6190-3-584）

付録 I.2　このマニュアルでの表記

(1)　製品名

このマニュアルでの表記と正式名称を次の表に示します。

表記		製品名
IPF		Itanium(R) Processor Family
TP1/Message Control		uCosminexus TP1/Message Control
TP1/NET/Library		uCosminexus TP1/NET/Library
TP1/NET/User Agent		uCosminexus TP1/NET/User Agent
UNIX	AIX	AIX V6.1
		AIX V7.1
		AIX V7.2
	HP-UX	HP-UX 11i
		HP-UX 11i V2（PA-RISC）
		HP-UX 11i V2（IPF）
	Linux	Red Hat Enterprise Linux(R) Server 6 (32-bit x86)
		Red Hat Enterprise Linux(R) Server 6 (64-bit x86_64)
		Red Hat Enterprise Linux(R) Server 7 (64-bit x86_64)

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

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

コード	JIS コード配列	ASCII コード配列
$(5c)_{16}$	'¥'（円記号）	'＼'（バックスラッシュ）
$(7e)_{16}$	'￣'（オーバライン）	'~'（チルダ）

付録 I.3　英略語

このマニュアルで使用する英略語を次に示します。

英略語	英字での表記
ACSE	Association Control Service Element
AP	Application Program
LE	Logical Entity
MCF	Message Control Facility
MHP	Message Handling Program
NIF/OSI	Network Interface Feature/Open Systems Interconnection
OS	Operating System
OSAS/UA	Open Systems Interconnection Application Support Common Facility/User Agent
OSI	Open Systems Interconnection
SPP	Service Providing Program
TCP/IP	Transmission Control Protocol/Internet Protocol
UA	User Agent
UAM	UA Manager
UAP	User Application Program
UASDU	User Agent Service Data Unit
UOC	User Own Cording
UPDU	User Agent Protocol Data Unit

付録 I.4　KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

付録 J　用語解説

(英字)

AJ（メッセージ送信完了ジャーナル）

MCF が取得する統計用の履歴情報の一つです。メッセージ出力通番，出力先論理端末名などで構成されます。

AJ の取得タイミングは，メッセージの種類で異なります。一方送信メッセージ（UINQ）の場合は，一方送信メッセージ（UREP）の最終セグメントを受信した直後です。一方送信メッセージ（UBRD）の場合は，一方送信メッセージ（UBRD）の最終セグメントを送信したあと，相手システムから UINF を受信した直後です。応答メッセージ（UREP）の場合は，応答メッセージ（UREP）の最終セグメントを送信した直後です。

GJ（メッセージ受信ジャーナル）

MCF が取得する統計用の履歴情報の一つです。入力メッセージサイズ，入力論理端末名などで構成されます。

GJ の取得タイミングは，非同期受信関数を発行して，MHP が受信メッセージを入力キューから取り出した直後です。

IJ（メッセージ入力ジャーナル）

MCF が取得する統計用の履歴情報の一つです。メッセージ入力通番，入力論理端末名，入力メッセージ種別，および入力メッセージなどで構成されます。

IJ の取得タイミングは，相手システムから受信したメッセージを入力キューに格納する直前です。

MJ（メッセージジャーナル）

MCF が取得する統計用の履歴情報の一つです。入力論理端末名または，出力論理端末名，メッセージジャーナル種別，入力または出力メッセージ（入力メッセージ編集前のデータ，または出力メッセージ編集後のデータ）などで構成されます。

MJ の取得タイミングは，入力メッセージの場合，入力メッセージ編集 UOC を呼び出す直前です。また，出力メッセージの場合，出力メッセージ編集 UOC を呼び出した直後です。

OJ（メッセージ出力ジャーナル）

MCF が取得する統計用の履歴情報の一つです。メッセージ出力通番，出力論理端末名，出力メッセージ種別，出力メッセージなどで構成されます。

OJ の取得タイミングは，非同期送信関数を発行して，UAP が送信メッセージを出力キューに格納した直後です。

OSAS/UA プロトコル

拡張 HNA のアプリケーションプロファイルの一つとして規定した，分散処理ネットワークでのトランザクション処理を実現するためのプロトコルです。

UA

OSAS/UA プロトコルで指定された手順を実行するエンティティです。データの送受信をします。

UA には，一般 UA と制御 UA があります。一般 UA は，一般 UA の開局，閉局，およびデータの送受信に使用します。制御 UA は，コネクションの確立と解放に使用します。

（カ行）

コネクション

MCF の通信管理側の通信接点であり，MCF と通信管理はコネクション単位に送受信をします。OSAS/UA プロトコルのアソシエーションに対応します。

（ラ行）

論理端末

TP1/NET/User Agent と UAP との通信接点です。TP1/NET/User Agent と UAP は，論理端末単位にメッセージの送受信をします。

索引

O

R

S

T

U

X

あ

い

う

お

か

き

く

ろ