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/Secondary Logical Unit - TypeP2 編

解説・手引・文法・操作書

3000-3-D80

マニュアルの購入方法

このマニュアル，および関連するマニュアルをご購入の際は，巻末の「ソフトウェアマニュアルのサービス ご案内」をご参照ください。

HITACHI

■対象製品

・適用 OS：AIX 5L V5.1，AIX 5L V5.2，AIX 5L V5.3

P-1M64-3131 uCosminexus TP1/Message Control 07-00

P-1M64-3231 uCosminexus TP1/NET/Library 07-00

P-F1M64-3231B uCosminexus TP1/NET/Secondary Logical Unit - TypeP2 07-00

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。

これらの製品は，ISO9001 および TickIT の認証を受けた品質マネジメントシステムで開発されました。

■輸出時の注意

本製品を輸出される場合には，外国為替および外国貿易法ならびに米国の輸出管理関連法規などの規制をご確認の上，必要な手続きをお取りください。

なお，ご不明な場合は，弊社担当営業にお問い合わせください。

■商標類

AIX は，米国における米国 International Business Machines Corp. の登録商標です。

IMS は，米国における米国 International Business Machines Corp. の商標です。

SNA は，米国 International Business Machines Corp. のプロトコル名称です。

UNIX は，X/Open Company Limited が独占的にライセンスしている米国ならびに他の国における登録商標です。

■発行

2006 年 11 月（第 1 版）3000-3-D80

■著作権

はじめに

このマニュアルは，TP1/NET/Secondary Logical Unit - TypeP2 の概要，機能，操作，および運用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 SLUTYPE-P 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

SLU - TypeP2 を使用したシステム間の通信（AP 間通信）の概要について説明しています。

第 2 章　機能

SLU - TypeP2 が設定するコネクションの確立方法，メッセージの種類と送受信の方法について説明しています。

第 3 章　メッセージ送受信インタフェース

UAP の作成方法，および作成例について説明しています。

第 4 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

SLU - TypeP2 に関連するユーザオウンコーディングインタフェース，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 5 章　システム定義

SLUTYPE-P 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 SLU - TypeP2 固有のシステム定義について説明しています。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，および定義例について説明しています。

第 6 章　運用コマンド

SLU - TypeP2 で使用する運用コマンドについて説明しています。

第 7 章　組み込み方法

SLU - TypeP2 を OpenTP1 システムへ組み込む方法について説明しています。

第 8 章　障害対策

SLU - TypeP2 運用中に発生する障害と，SLU - TypeP2 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　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 B　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。

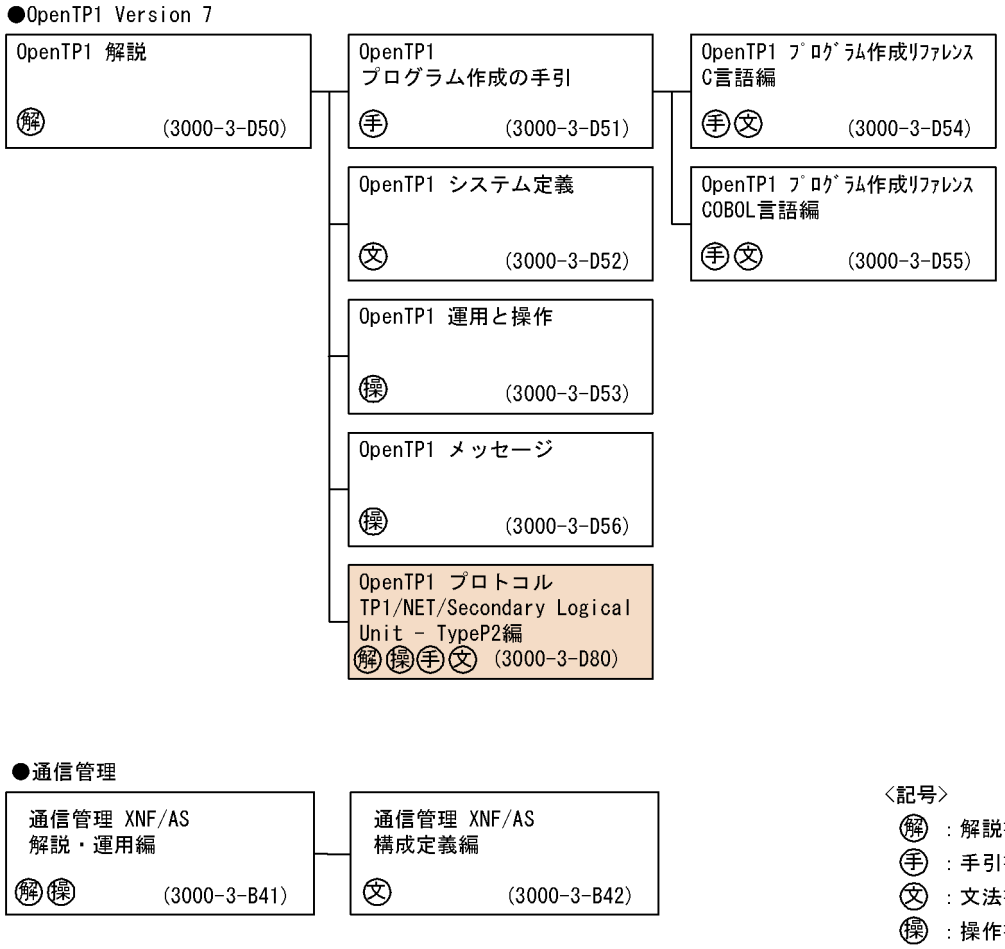
付録 C　理由コードおよびセンスコード一覧

障害通知イベントが発生した場合の理由コード，および -RSP 送信時のセンスコードについて説明しています。

付録 D　用語解説

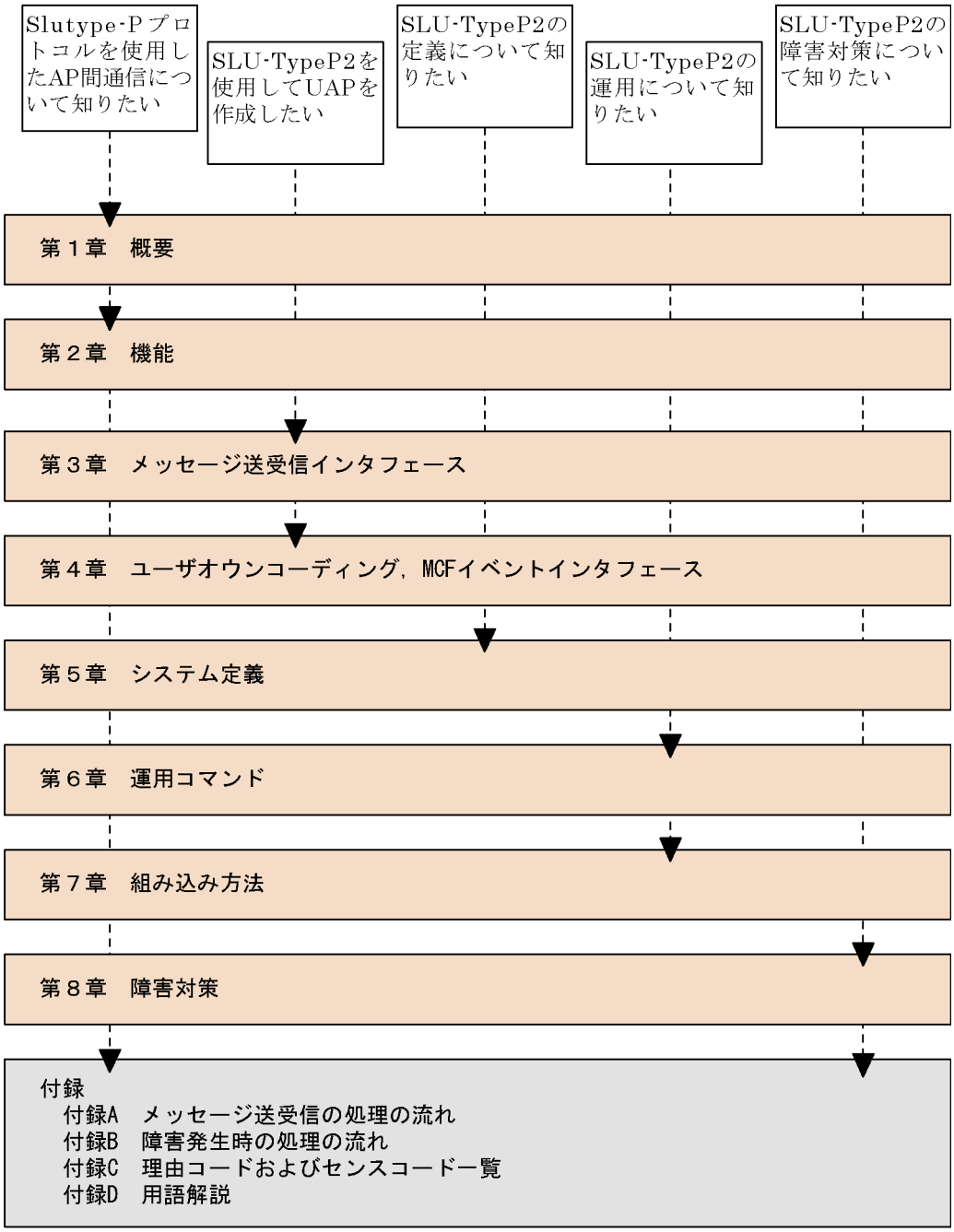
SLU - TypeP2 で使用する用語について説明しています。

■関連マニュアル



■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。


Slutype-Pプロトコルを使用したAP間通信について知りたい
SLU-TypeP2を使用してUAPを作成したい
SLU-TypeP2の定義について知りたい
SLU-TypeP2の運用について知りたい
SLU-TypeP2の障害対策について知りたい
第１章　概要
第２章　機能
第３章　メッセージ送受信インタフェース
第４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
第５章　システム定義
第６章　運用コマンド
第７章　組み込み方法
第８章　障害対策
付録
付録A　メッセージ送受信の処理の流れ
付録B　障害発生時の処理の流れ
付録C　理由コードおよびセンスコード一覧
付録D　用語解説

（凡例）

：　必ず読む項目

■図中で使用する記号

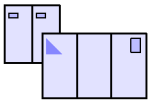
このマニュアルの図中で使用する記号を，次のように定義します。

●ワークステーション，端末

●論理端末（端末状態）

●論理回線

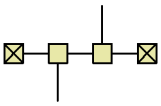
●ホストコンピュータ



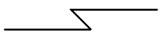
●ファイル

●ネットワーク（WAN）

●ネットワーク（LAN）



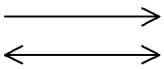
●通信回線



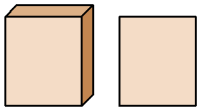
●プログラムの流れ

●データ，メッセージの流れ

●制御の流れ



●プログラム



●入出力の動作

●障害

■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。

文法記述記号	意味
{　}	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。 （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，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）
＜ 16 進数字＞	数字（0 ～ 9）とアルファベット（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

構文要素記号	意味
＜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） （ただし，パス名は使用する OS に依存）

■このマニュアルでの表記

（1）製品名

このマニュアルで使用する製品名称の略称を次に示します。

製品名称	略称
AIX 5L V5.1	AIX
AIX 5L V5.2	
AIX 5L V5.3	
uCosminexus TP1/Message Control	TP1/Message Control
uCosminexus TP1/NET/Library	TP1/NET/Library
uCosminexus TP1/NET/Secondary Logical Unit - TypeP2	TP1/NET/Secondary Logical Unit - TypeP2 または SLU - TypeP2

（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）	'～'（チルド）

■略語一覧

このマニュアルで使用する英略語の一覧を次に示します。

英略語	英字での表記
AP	Application Program
FM	Function Management
HNA	Hitachi Network Architecture
LAN	Local Area Network
LU	Logical Unit
MCF	Message Control Facility
MHP	Message Handling Program

英略語	英字での表記
OS	Operating System
PLU	Primary Logical Unit
RU	Request Unit または Response Unit
SPP	Service Providing Program
TS	Transmission Services
UAP	User Application Program
UOC	User Own Coding
WAN	Wide Area Network

■常用漢字以外の漢字の使用について

このマニュアルでは，常用漢字を使用することを基本としていますが，次に示す用語については，常用漢字以外の漢字を使用しています。
個所（かしょ） 閉塞（へいそく）

■ 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。

■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

図目次

表目次

1 概要

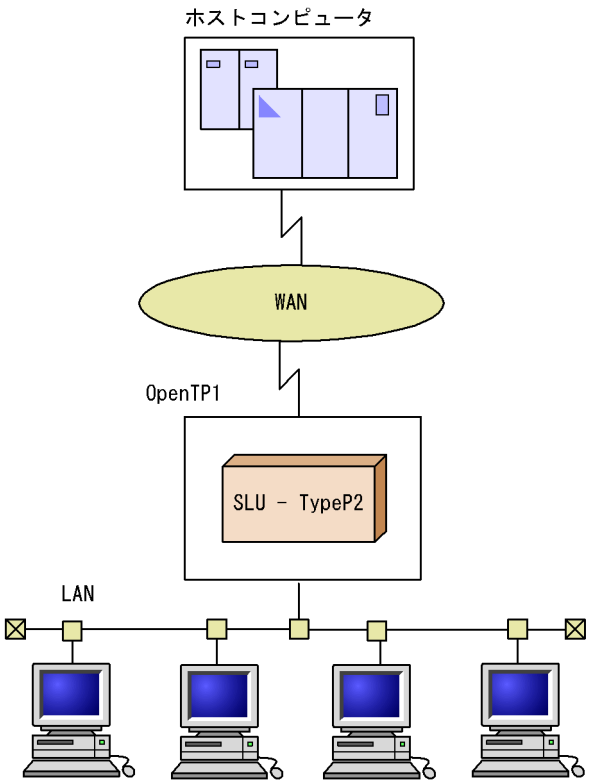
SLU - TypeP2 は，OpenTP1 システムを構成するプログラムの一つです。ホストコンピュータ，端末などを SLUTYPE-P プロトコルによって論理的に接続し，メッセージを送受信します。
この章では，SLU - TypeP2 を使用したシステム間の通信（AP 間通信）の概要について説明します。

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をいいます。SLU - TypeP2 は，SLUTYPE-P プロトコルの 2 次局側として，1 次局側のホストシステムと AP 間通信をするプログラムです。SLU - TypeP2 を使用した AP 間通信では，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。

SLU - TypeP2 を使用したネットワーク構成の例を次の図に示します。

図 1-1　SLU - TypeP2 を使用したネットワーク構成の例



1.2 AP 間通信の形態

1.2.1 通信形態

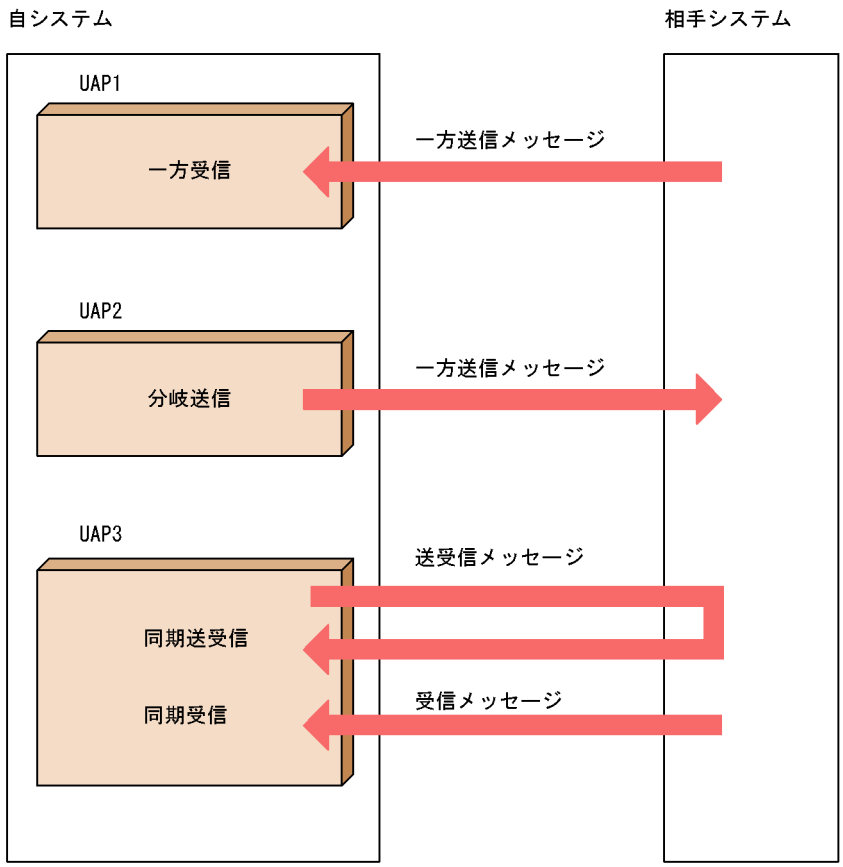
AP 間通信を使用すると，自システムで発生したトランザクションを相手システムで処理したり，その結果を受信したりできます。また，相手システムで発生したトランザクションを自システムで処理したり，その結果を送信したりできます。

SLU - TypeP2 を使用した AP 間通信の形態には，次の四つがあります。

- 一方受信
- 分岐送信
- 同期送受信
- 同期受信

SLU - TypeP2 を使用した AP 間通信の例を次の図に示します。

図 1-2 SLU - TypeP2 を使用した AP 間通信の例



(1) 一方受信

相手システムから AP 間通信の開始要求を受信する形態です。開始要求を受信すると，SLU - TypeP2 はメッセージ送受信のためのアプリケーションを起動します。

(2) 分岐送信

相手システムに対して，メッセージを送信する形態です。

(3) 同期送受信

自システムからメッセージを送信し，応答を受信する形態です。このとき，自システムからメッセージを送信後，相手システムからの応答を待ちます。応答を受信したときに，同期送受信を要求した UAP に制御を返します。

(4) 同期受信

同期送受信でメッセージが複数に分割されている場合，2 番目以降を受信する形態です。メッセージの先頭部分は同期送受信で受け取ります。メッセージの最終部分の受信が完了すると，同期受信を要求した UAP に制御を返します。

この形態は，同期送受信のメッセージが単一の場合は適用されません。

1.3 ソフトウェアの構成

SLU - TypeP2 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。

この節では，SLU - TypeP2 を使用した MCF を実現するための前提プログラムと，SLU - TypeP2 を組み込んだソフトウェア構成について説明します。

1.3.1 前提プログラム

SLU - TypeP2 を使用した MCF を実現するための前提プログラムは次のとおりです。

適用 OS：AIX

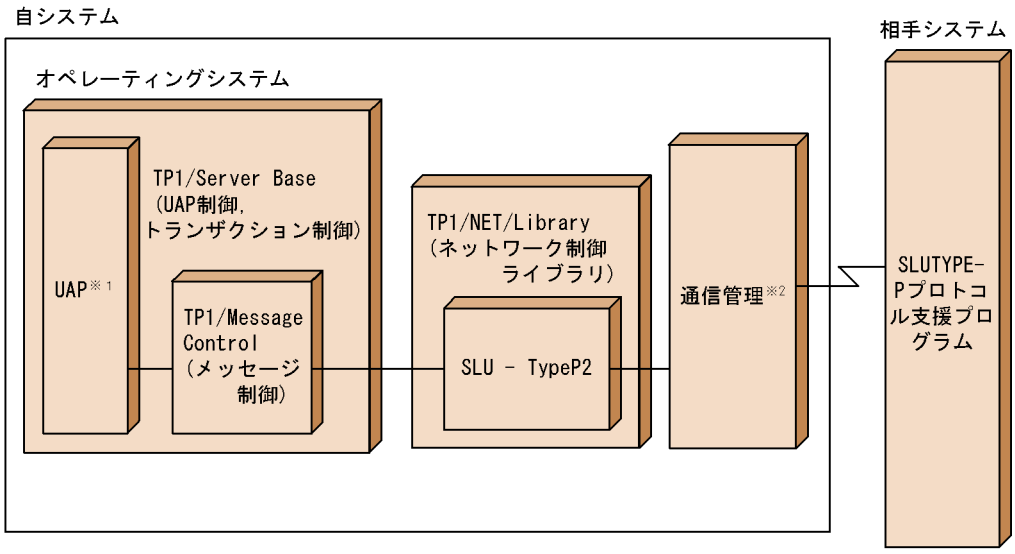
- P-1M64-2131　uCosminexus TP1/Server Base　07-00 以降
- P-1M14-511　XNF/AS/BASE ※
- P-F1M14-5111　XNF/AS/WAN ※
- P-F1M14-5116　XNF/AS/HNA2 ※

注※
前提となる XNF のバージョンは，AIX V5.1 以降に対応するバージョン以降です。

1.3.2 ソフトウェア構成の例

SLU - TypeP2 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，SLU - TypeP2 が AP 間通信で使用するプロトコルおよび通信相手プログラムを表 1-1 に示します。

図 1-3　SLU - TypeP2 を組み込んだソフトウェア構成の例



注※ 1

SLU - TypeP2 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/AS を使用できます。

表 1-1　SLU - TypeP2 に適用する AP 間通信のプロトコル

プロトコル	通信相手
SLUTYPE-P プロトコル	IMS IBM ホストシステム

2 機能

一般に，AP 間通信をするときには，自システムと相手システムとの間であらかじめ通信上の規約（プロトコル）を決める必要があります。SLU - TypeP2 は，二つのシステムの間にコネクションという論理的通信路を設定し，メッセージを送受信します。
この章では，SLU - TypeP2 が設定するコネクションの確立方法，メッセージの種類と送受信の方法について説明します。

2.1　AP 間通信の仕組み

2.1.1　コネクションの確立と解放

SLU - TypeP2 では，相手システムとの間に論理的通信路（**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，SLUTYPE-P プロトコルのセションに対応します。

(1) コネクションの確立

コネクションの確立方法には，次の三つがあります。

- ホストからの要求による確立
- SLU - TypeP2 の定義による自動確立
- 運用コマンド入力による手動確立

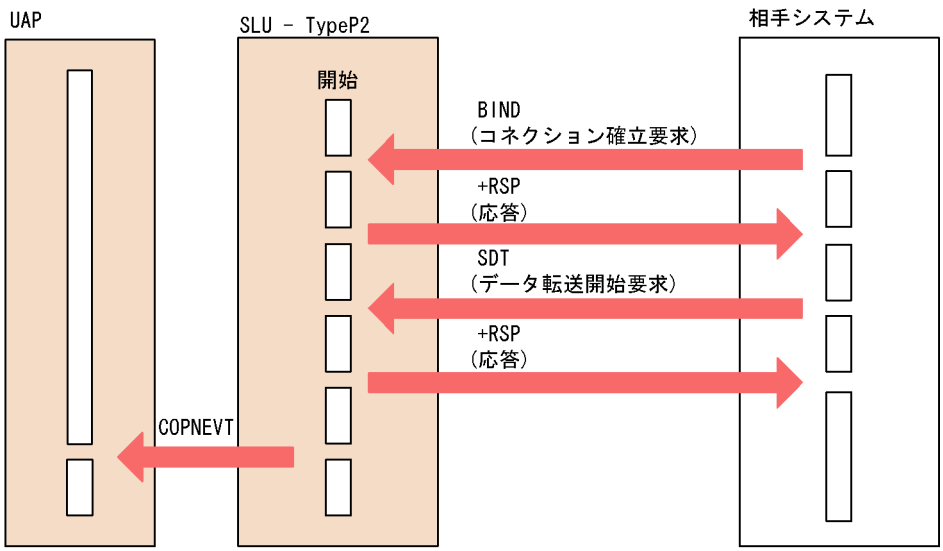
(a) ホストからの要求による確立

ホスト（相手システム）からコネクションの確立の要求を受ける方法です。

システム定義時，コネクション定義の起動種別（mcftalccn -k）にホスト起動（host）が指定された場合，SLU - TypeP2 はホストからの確立要求（BIND）を待ちます。コネクション確立後，すべての論理端末を閉塞解除し，UAP にコネクションの確立（COPNEVT）を通知します。

ホストからの要求による確立を次の図に示します。

図 2-1　ホストからの要求による確立

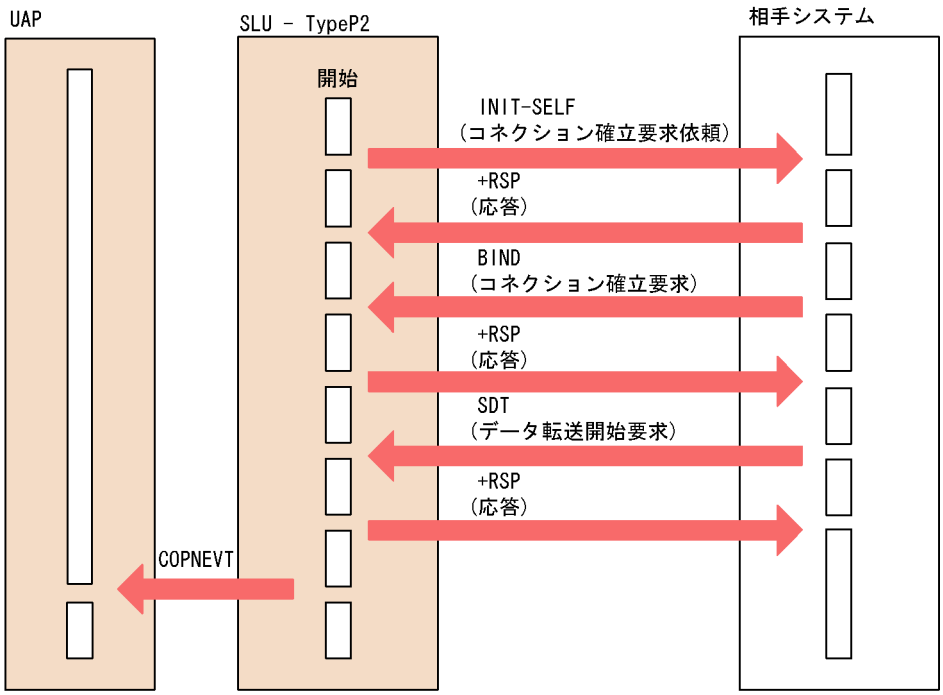


(b) SLU - TypeP2 の定義による自動確立

システム定義時，コネクション定義の起動種別（mcftalccn -k）に端末起動（ws）が指定され，かつ自動確立（-i auto）が指定された場合，SLU - TypeP2 はホストに対して確立要求を送信します。コネクション確立後，すべての論理端末を閉塞解除し，UAP にコネクションの確立（COPNEVT）を通知します。

SLU - TypeP2 の定義による自動確立を次の図に示します。

図 2-2　SLU - TypeP2 の定義による自動確立

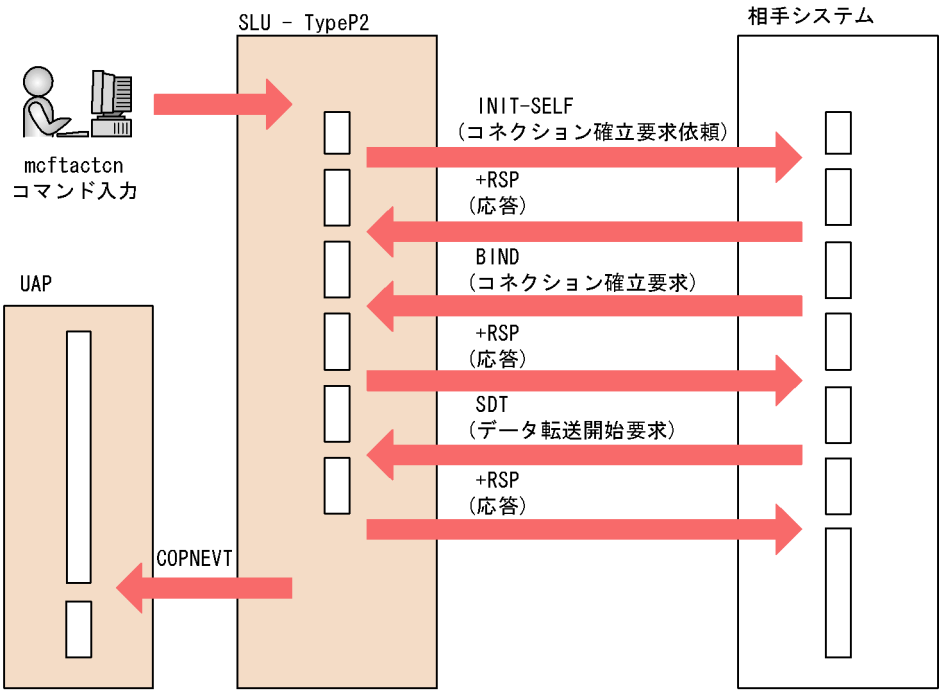


(c) 運用コマンド入力による手動確立

システム定義時，コネクション定義の起動種別（mcftalccn -k）に端末起動（ws）が指定され，かつ手動確立（-i manual）が指定された場合，または，コネクション解放後に再確立する場合，運用コマンド（mcftactcn）を入力することでコネクションを確立できます。このとき，SLU - TypeP2 はホストに対して確立要求を送信します。コネクション確立後，すべての論理端末を閉塞解除し，UAP にコネクションの確立（COPNEVT）を通知します。

運用コマンド入力による手動確立を次の図に示します。

図 2-3 運用コマンド入力による手動確立



(2) コネクション確立時のチェック項目

ホストからの要求によってコネクションを確立する場合，SLU - TypeP2 はホストから受信した確立要求（BIND コマンド）のパラメタをチェックします。このチェックの結果，エラーが検出されると否定応答（センスコード =0821）が出力されます。否定応答（センスコード =0821）は，次のエラーが検出された場合に出力されます。

- BIND 指令中のセションパラメタが不正の場合
- 2 次局側でサポートしていない内容を含んでいる場合

SLU - TypeP2 で固有にチェックするパラメタの内容を次の表に示します。

表 2-1 パラメタのチェック内容

内容	位置（バイト）	長さ（バイト）	チェック基準値（16 進数字）
フォーマット，BIND 指令種別	1	1	$(01)_{16}$，または $(00)_{16}$
FM プロファイル番号	2	1	$(04)_{16}$
TS プロファイル番号	3	1	$(04)_{16}$
PLU プロトコル	4	1	$(b1)_{16}$
SLU プロトコル	5	1	$(b1)_{16}$

内容	位置（バイト）	長さ（バイト）	チェック基準値（16 進数字）
共通プロトコル	6	2	$(6080)_{16}$
SLU 送信最大 RU 長※	10	1	$(00)_{16}$，または MCF 通信構成定義の送信最大 RU 長（mcftalccn -r sndrusiz）の指定値
SLU 受信最大 RU 長※	11	1	$(00)_{16}$，または MCF 通信構成定義の受信最大 RU 長（mcftalccn -r rcvrusiz）の指定値
LU タイプ	14	1	$(00)_{16}$

注※

SLU 送信最大 RU 長と SLU 受信最大 RU 長は，相手システム（ホストシステム）で定義される最大 RU 長です。これらの値は，自システム（OpenTP1 システム）の MCF 通信構成定義（mcftalccn）の -r sndrusiz と -r rcvrusiz の指定値と，次に示す大小関係にある場合に有効となります。

- 自システムの送信最大 RU 長（-r sndrusiz）の指定値が，ホストシステムで指定されている SLU 送信最大 RU 長以下の場合
- 自システムの受信最大 RU 長（-r rcvrusiz）の指定値が，ホストシステムで指定されている SLU 受信最大 RU 長以上の場合

(3) コネクション確立時の再試行

SLU - TypeP2 は，コネクション確立時，コネクション定義の起動種別（mcftalccn -k）に端末起動（ws）が指定された場合，コネクション定義のコネクション確立再試行（mcftalccn -b）で指定された値に基づいてコネクション確立の再試行をします。

コネクション定義の詳細については，5 章の「mcftalccn（コネクション定義の開始）」を参照してください。

(4) コネクションの正常解放

コネクションの正常解放には，次の三つがあります。

- ホストからの解放
- システム終了時の解放
- 運用コマンド（mcftdctcn）の入力による解放

コネクションが正常解放された場合，論理端末は自動的に閉塞され，SLU - TypeP2 は UAP にコネクションの解放（CCLSEVT）を通知します。なお，システム終了時は，CCLSEVT は通知されません。

コネクションの正常解放は，ユーザ間のデータ送受信が終了したあとに行ってください。

ホストからの解放を図 2-4 に，システム終了時の解放を図 2-5 に，運用コマンド入力による解放を図 2-6 に示します。

図 2-4　ホストからの解放

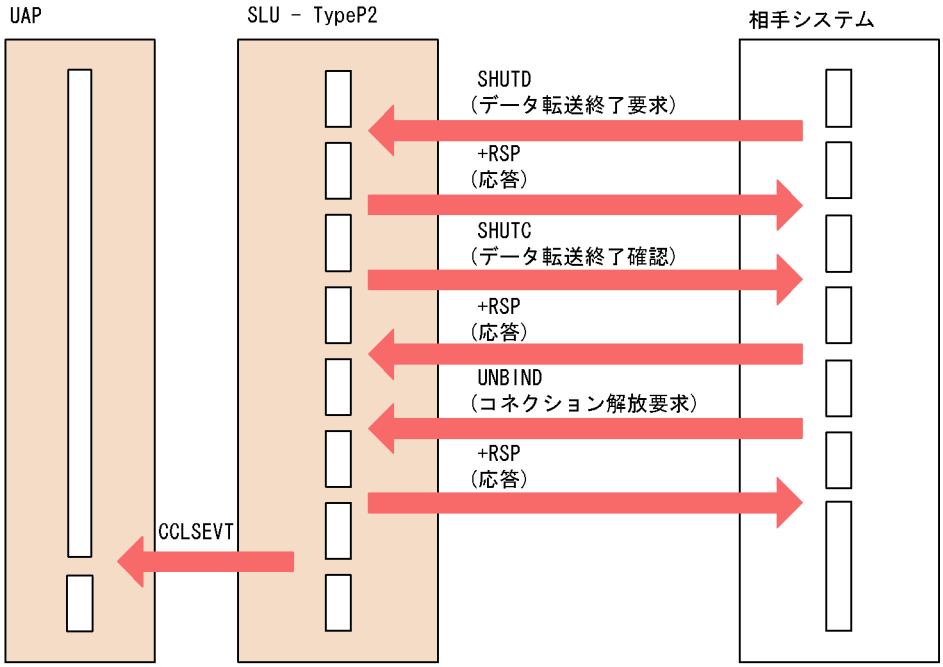
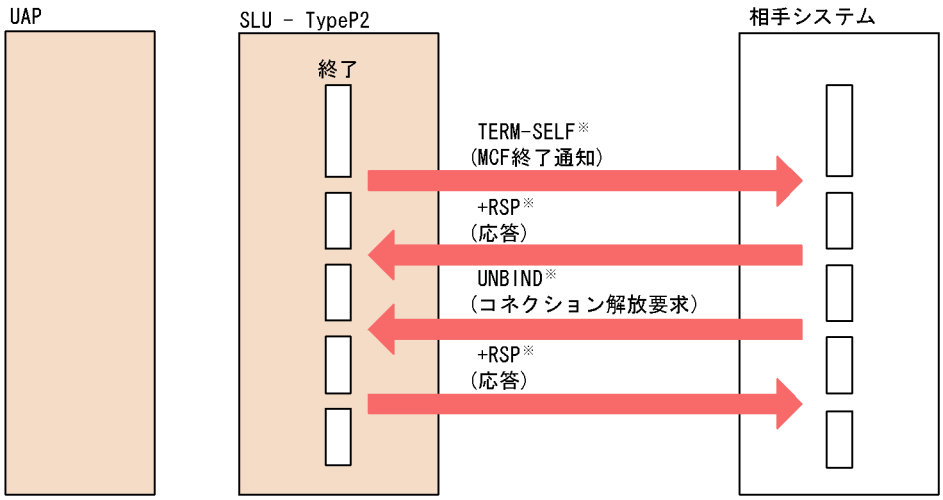


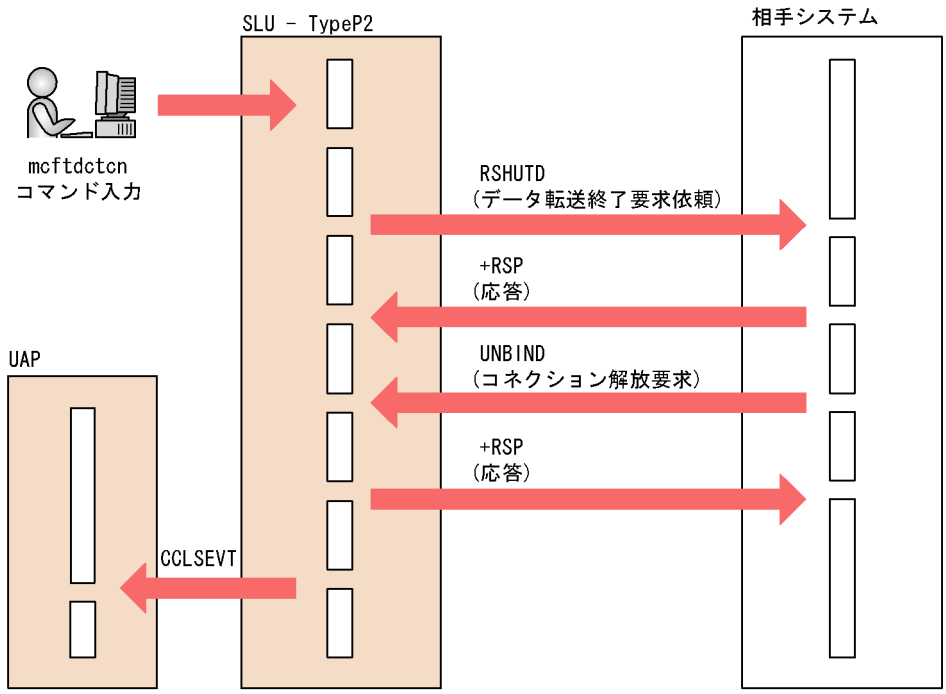
図 2-5　システム終了時の解放



注　システム終了時，CCLSEVTは通知されません。

注※　通信管理で発行されます。

図 2-6　運用コマンド（mcftdctcn）入力による解放



(5) コネクションの強制解放

コネクションの強制解放には，次の三つがあります。

- 下位障害発生による解放
- 運用コマンド（mcftdctcn -f）の入力による解放
- バッファ不足などの内部矛盾による解放

コネクションが強制解放された場合，論理端末は自動的に閉塞され，SLU - TypeP2 は UAP にコネクションの解放（CERREVT）を通知します。下位障害，内部矛盾などの障害対策については，「8.　障害対策」を参照してください。

下位障害発生による解放を図 2-7 に，運用コマンド（mcftdctcn -f）入力による解放を図 2-8 に，内部矛盾による解放を図 2-9 に示します。

図 2-7 下位障害発生による解放

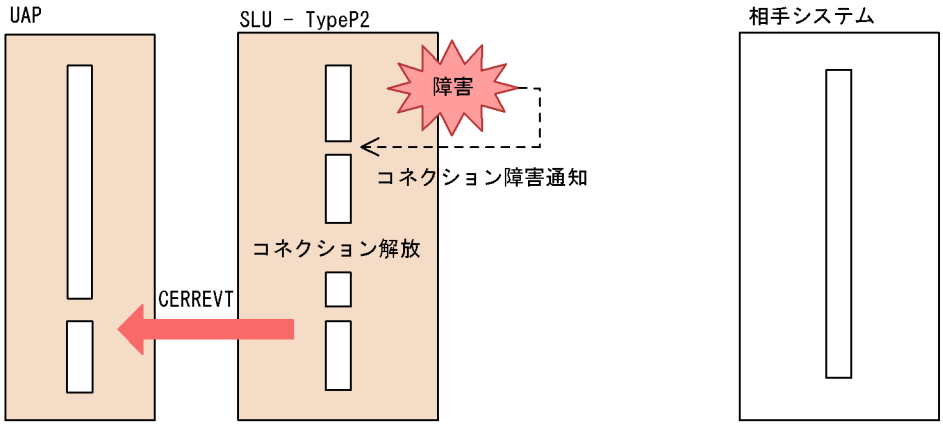
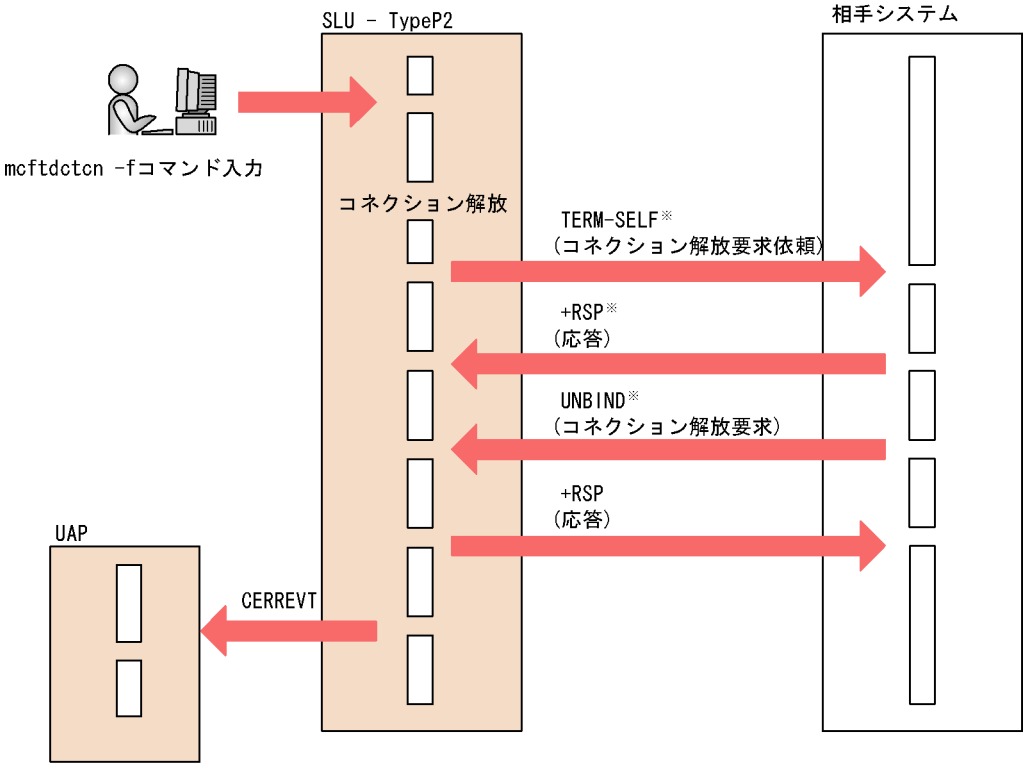


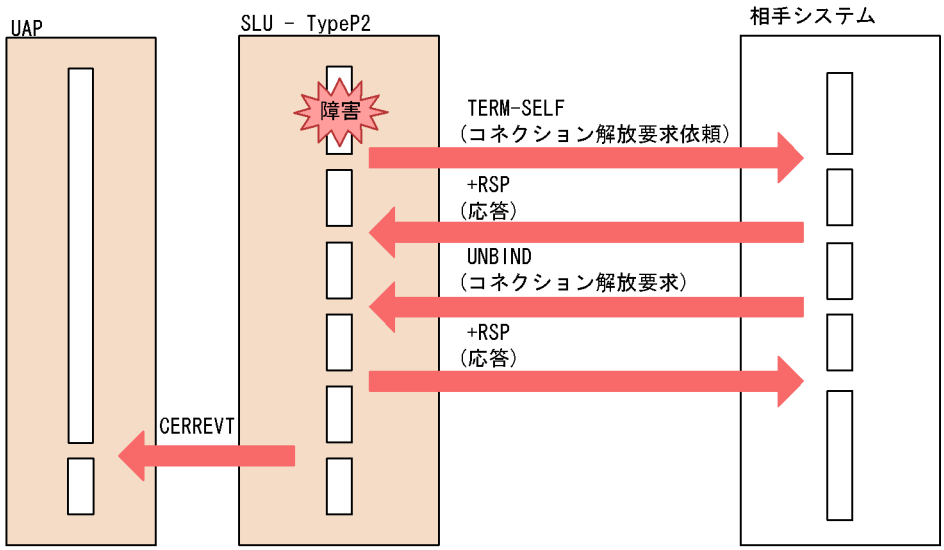
図 2-8 運用コマンド（mcftdctcn -f）入力による解放



注※

通信管理で発行されます。

図 2-9　内部矛盾による解放



(6) コネクション解放後の回復動作

システム定義のコネクション定義の起動種別（mcftalccn -k）に指定するオペランドによって，SLU - TypeP2 では解放したコネクションの回復動作が異なります。

コネクション定義の起動種別（mcftalccn -k）の指定内容と SLU - TypeP2 の動作を次の表に示します。

表 2-2　-k オプションの指定内容と SLU - TypeP2 の動作

-k オプションの指定内容	SLU - TypeP2 の動作
host（ホスト起動）	コネクション解放後，ホストからのコネクション確立要求を待ちます。
ws（端末起動）	コネクション解放後，ユーザからの運用コマンド（mcftactcn）の入力を待ちます。

なお，コマンド入力による手動回復なのか，システムによる自動回復なのかを示す回復動作情報は，SLU - TypeP2 が CERREVT または CCLSEVT に設定します。これらのイベントの種類および詳細については，「4.2　MCF イベントインタフェース」を参照してください。

コネクション定義の起動種別（mcftalccn -k）にホスト起動（host）が指定されたコネクションに対して運用コマンド（mcftactcn）が入力された場合，mcftactcn コマンドはエラーリターンして，SLU - TypeP2 がメッセージログ（KFCA15342-E）を出力します。

また，コネクション定義の起動種別（mcftalccn -k）に端末起動（ws）が指定された場合，該当するコネクションに対して運用コマンド（mcftactcn）が入力されるまで，ホス

トからのコネクション確立要求は通信管理によって拒否されます。

2.1.2 コネクションと論理端末の関係

SLU - TypeP2 は，論理端末を通して，自システムの UAP とメッセージを送受信します。この論理端末は，SLU - TypeP2 と UAP との通信接点に当たります。

これに対してコネクションとは，SLU - TypeP2 が通信管理プログラムを介して，相手システムの UAP とメッセージを送受信するときに確立するものです。コネクションと論理端末の指定を対応させると，自システムと相手システムとの論理的通信路が確立でき，AP 間通信ができるようになります。コネクションと論理端末は，システム定義時に対応させます。

2.1.3 論理端末とアプリケーションの型の関係

SLU - TypeP2 で扱う論理端末の端末タイプには，次の三つがあります。

- send（送信型論理端末）
- receive（受信型論理端末）
- request（問い合わせ型論理端末）

アプリケーションは，ユーザが送受信データの中に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。アプリケーションは，サービスの方式によって型が異なります。この型を MCF の属性の一つとして，システム定義時に指定します。SLU - TypeP2 で使用するアプリケーションの型は，非応答型（noans）です。

論理端末の端末タイプとメッセージ，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 論理端末の端末タイプとメッセージ，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	アプリケーションの型	UAP インタフェース	通信形態
send（送信型論理端末）	一方送信メッセージ	非応答型	send	分岐送信
			resend	
receive（受信型論理端末）	一方送信メッセージ	非応答型	receive	一方受信
request（問い合わせ型論理端末）	問い合わせメッセージ	非応答型	send	分岐送信
			resend	

論理端末の端末タイプ	メッセージ	アプリケーションの型	UAP インタフェース	通信形態
			sendrecv（セグメント送信時，先頭セグメント受信時）	同期送受信（同期問い合わせ応答）
			recvsync（後続セグメント受信時）	同期受信（同期問い合わせ応答）
			receive	一方受信

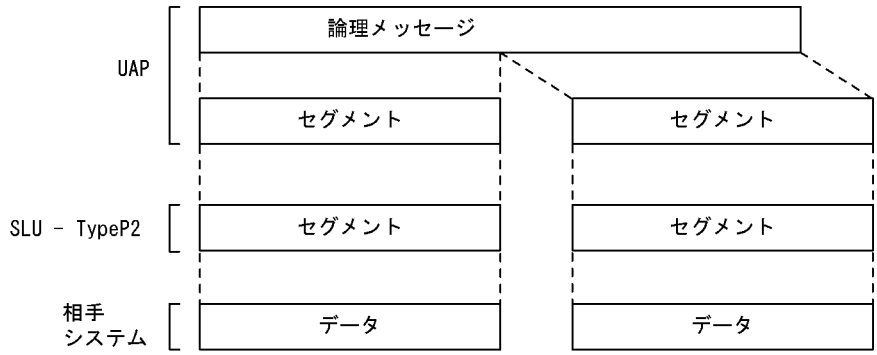
2.1.4 メッセージの分割と組み立て

SLU - TypeP2 と UAP との間では，メッセージを**セグメント**と呼ばれる単位に分割して扱います。一つの業務で処理するメッセージを**論理メッセージ**といいます。論理メッセージは，一つまたは複数のセグメントで構成されます。

UAP で送受信命令を発行すると，一つのセグメントを受信または送信します。論理メッセージが複数のセグメントで構成される場合，セグメントの数だけ送受信命令を発行してください。

セグメントと論理メッセージの関係を次の図に示します。

図 2-10 セグメントと論理メッセージ



また，メッセージ送受信の関数で処理するセグメントの先頭には，MCF で使用するヘッダ領域があります。このヘッダ領域の長さによって，バッファ形式 1 とバッファ形式 2 があります。通常，バッファ形式 1 を使用します。

メッセージの形式，パラメタの詳細，データ名などについては，「3. メッセージ送受信インタフェース」を参照してください。

2.2 AP 間通信メッセージの送受信

SLU - TypeP2 では，次に示すメッセージの送受信をします。

- 問い合わせメッセージの送信と応答メッセージの受信
- 一方送信メッセージの送信と受信

2.2.1 問い合わせメッセージの送信と応答メッセージの受信

ホスト（相手システム）へ同期型の問い合わせメッセージを送信し，応答を同期型のメッセージで受信します。

ホストへ問い合わせメッセージを送信するとき，UAP から sendrecv 関数を呼び出します。問い合わせメッセージが複数のセグメントで構成される場合，セグメントの数だけ sendrecv 関数を呼び出す必要があります。UAP がメッセージのセグメントを送信し終わると，SLU - TypeP2 は，ホストからの応答を待ちます。

ホストからの応答メッセージを受信すると，SLU - TypeP2 はメッセージの先頭セグメントを UAP に返します。応答メッセージが複数のセグメントで構成される場合，UAP は recvsync 関数を呼び出して後続セグメントを受信できます。

SLU - TypeP2 では，問い合わせ応答のデータ送受信時，ホストへ送信する応答識別（RQD または RQE）を指定することで，送信に対する応答（+RSP または -RSP）が必要かどうかを決定できます。応答を不要と指定すると，エラーが発生した場合だけ応答（-RSP）が送信されます。応答識別の指定については，5 章の「mcftalccn（コネクション定義の開始）」の -d オプションを参照してください。

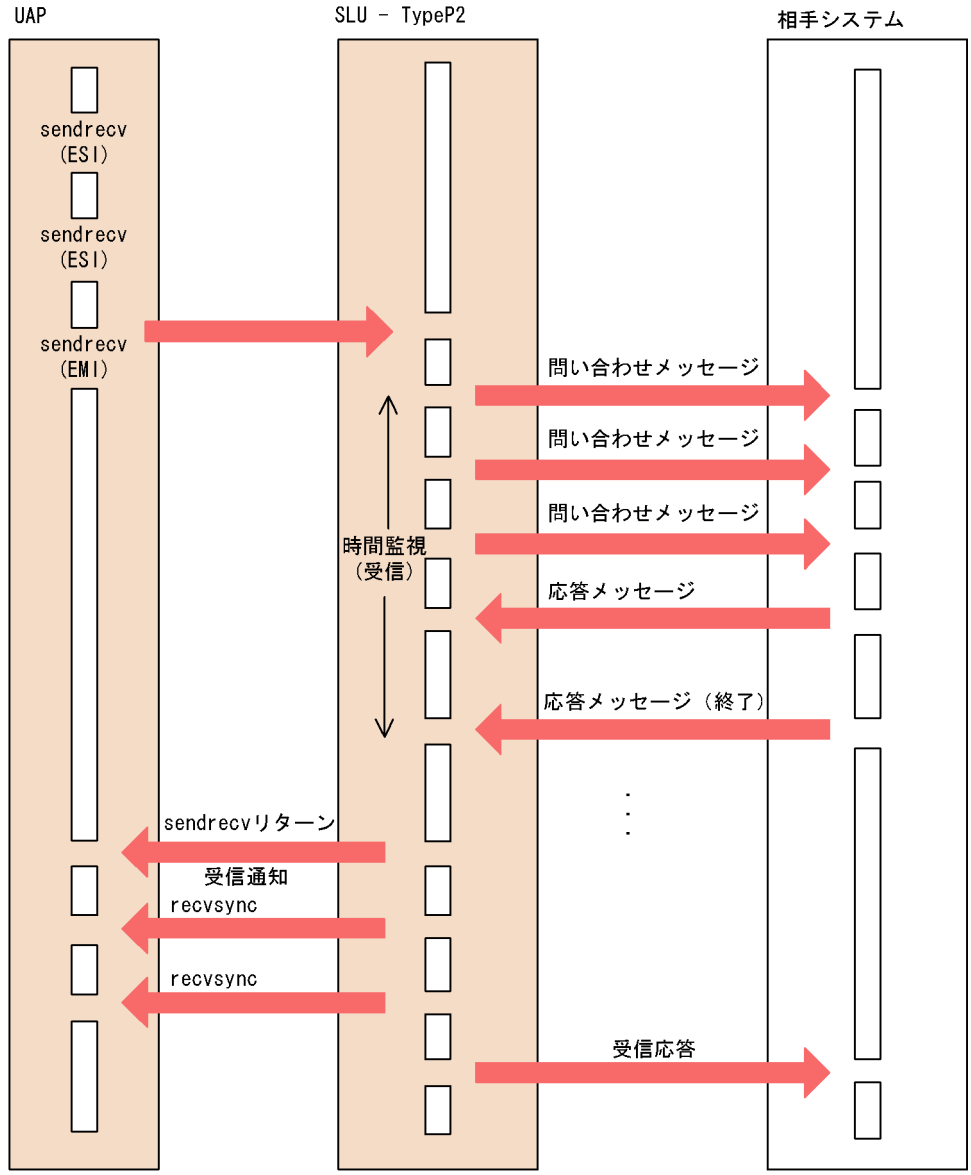
また，ユーザの指定によっては，問い合わせメッセージ送信完了から応答メッセージ受信までの間，時間監視ができます。監視時間内に応答がない場合，同期送受信要求がエラーリターンします。

ホストからの応答メッセージがダミーメッセージ（0 バイトデータ）の場合，データは UAP に通知されません。

自システムから複数セグメントメッセージを送信する場合，先頭または中間セグメントに対して否定応答（-RSP）を返さないで，最終セグメントに対して否定応答（-RSP）を返す相手システムと接続してください。

問い合わせメッセージの送信と応答メッセージの受信を次の図に示します。

図 2-11　問い合わせメッセージの送信と応答メッセージの受信



2.2.2　一方送信メッセージの送信と受信

一方送信メッセージを送信し，また，ホストから一方送信メッセージを受信します。

ホストへ一方送信メッセージを送信するとき，UAP から send 関数を呼び出します。一方送信メッセージが複数のセグメントで構成される場合，セグメントの数だけ send 関数を呼び出してください。一方送信メッセージの送信が完了すると，SLU - TypeP2 は，出力キューにある一方送信メッセージを送信済みとします。

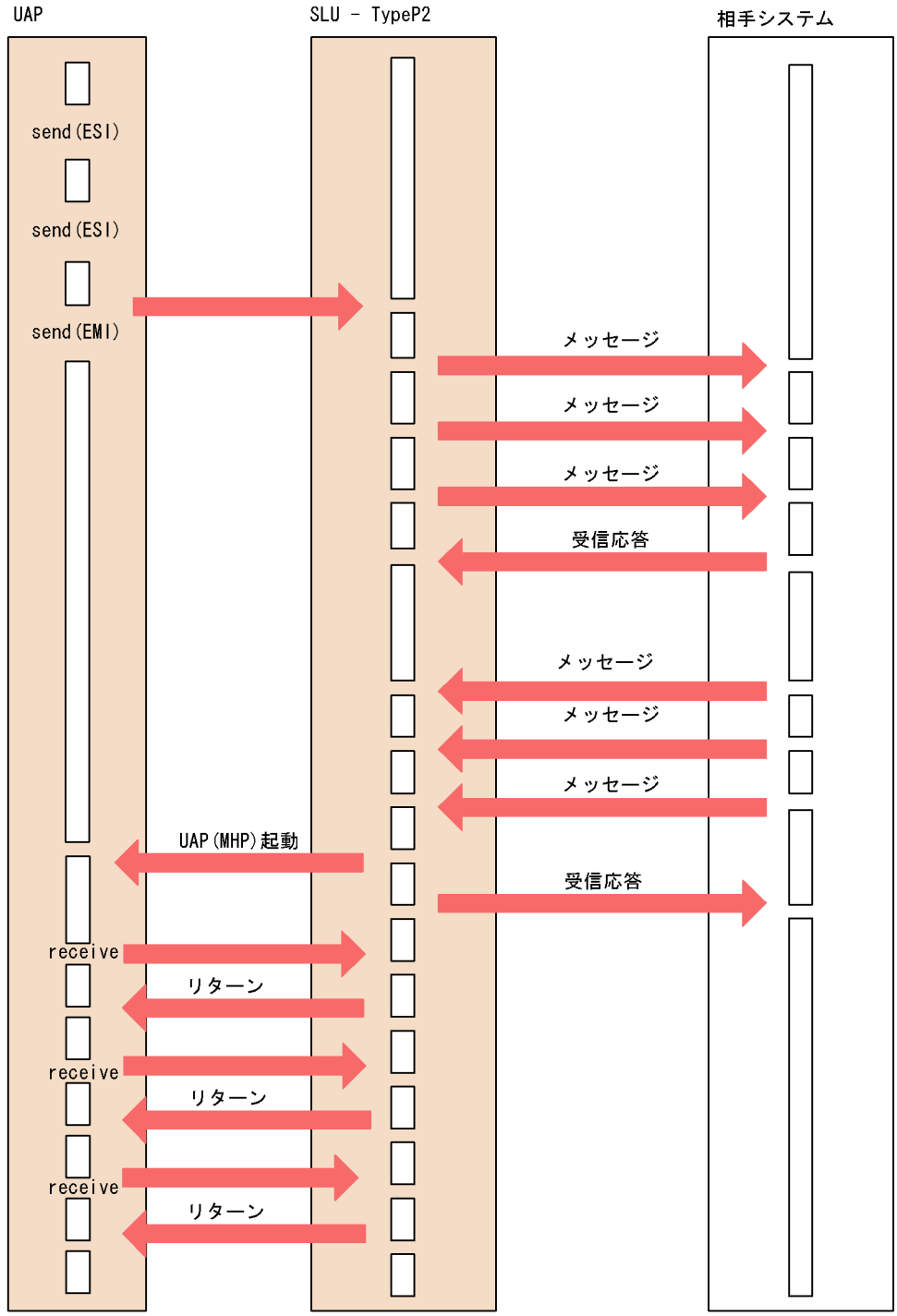
ホストからの応答も，一方送信メッセージとして受信します。一方送信メッセージを受信すると，SLU - TypeP2 は，メッセージに対応するアプリケーションを起動します。アプリケーションに該当する UAP は，receive 関数を呼び出してメッセージを受信できます。一方送信メッセージが複数のセグメントで構成される場合，セグメントの数だけ receive 関数を呼び出してください。

ホストから受信した一方送信メッセージがダミーメッセージ（0 バイトデータ）の場合，データは UAP に通知されません。

自システムから複数セグメントメッセージを送信する場合，先頭または中間セグメントに対して否定応答（-RSP）を返さないで，最終セグメントに対して否定応答（-RSP）を返す相手システムと接続してください。

一方送信メッセージの送信と受信を次の図に示します。

図 2-12　一方送信メッセージの送信と受信

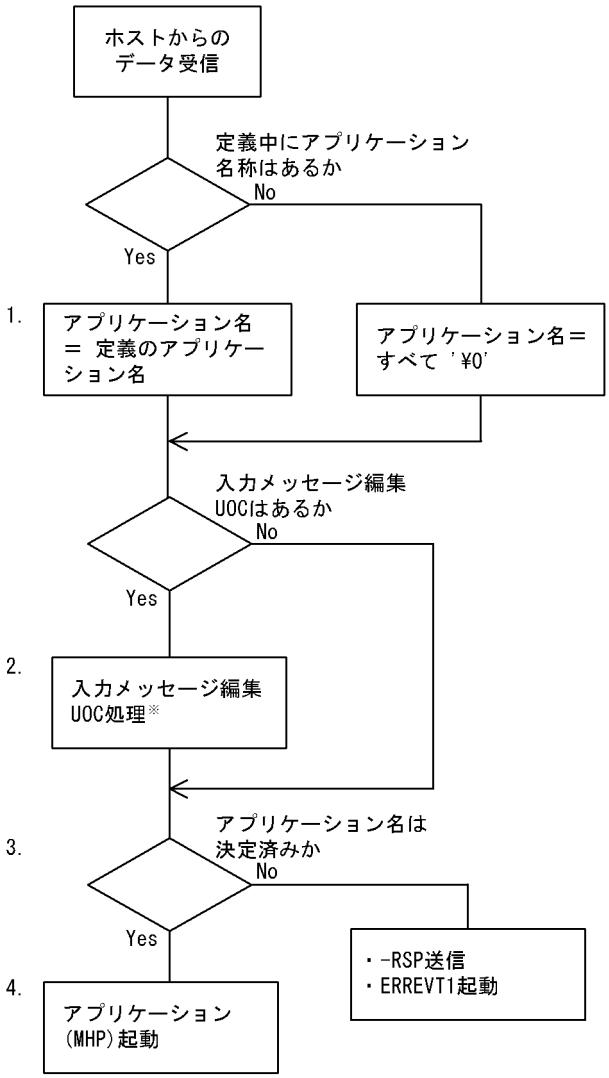


2.2.3　アプリケーション名の決定

SLU - TypeP2 では，システム定義時に，論理端末定義（mcftalcle）の -v オプションで指定された値を，ホストからの一方送信メッセージ受信時に，アプリケーション名とします。論理端末定義で指定を省略した場合は，入力メッセージ編集 UOC を使用してアプリケーション名を決定する必要があります。入力メッセージ編集 UOC については，「4.1.2　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。

アプリケーション名決定の流れを次の図に示します。

図 2-13　アプリケーション名決定の流れ



注※

入力メッセージ編集 UOC によるアプリケーション名の決定については，「4.1.1　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。

1. 入力メッセージの中の情報からアプリケーション名を決定します。
2. 入力メッセージがある場合，入力メッセージ編集 UOC による処理をします。
3. アプリケーション名が決定しているかどうかを判定します。
4. 決定されたアプリケーション名に基づいてアプリケーションを起動します。

3 メッセージ送受信インタフェース

SLU - TypeP2 を使用してメッセージを送受信する場合，ユーザは個別の業務に対応させるため，UAP を作成します。この章では，UAP の作成方法，および作成例について説明します。

メッセージ送受信インタフェースの一覧

SLU - TypeP2 で使用するメッセージ送受信インタフェースについて，C 言語，COBOL 言語，およびデータ操作言語に分けて説明します。

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。

C 言語のメッセージ送受信

C 言語（K&R 版 C，ANSI C，C++）でメッセージ送受信をする場合は，OpenTP1 システムで提供する関数を使用して UAP を作成します。

メッセージ送受信の関数を次の表に示します。

表 3-1　メッセージ送受信の関数（C 言語）

関数名	機能
dc_mcf_receive	一方送信メッセージの受信
dc_mcf_recvsync	同期型の応答メッセージの受信
dc_mcf_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	同期型の問い合わせメッセージの送受信

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。

COBOL 言語のメッセージ送受信

COBOL 言語でメッセージを送受信する場合は，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。

メッセージ送受信の関数に対応するプログラムを次の表に示します。

表 3-2　メッセージ送受信の関数に対応するプログラム（COBOL 言語）

プログラム名	データ名 A に設定する要求コード	意味
CBLDCMCF	'RECEIVE △ '	一方送信メッセージの受信
	'RECVSYNC'	同期型の応答メッセージの受信
	'RESEND △△ '	メッセージの再送
	'SEND △△△△ '	一方送信メッセージの送信
	'SENDRECV'	同期型の問い合わせメッセージの送受信

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（COBOL 言語）のメッセージ送受信

データ操作言語（COBOL 言語）を使用した，メッセージ送受信の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。

SLU - TypeP2 で固有の通信文を次の表に示します。

表 3-3　メッセージ送受信の通信文（データ操作言語）

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	CBLDCMCF('RECEIVE △ ')
	SEND	メッセージの送信	CBLDCMCF('SEND △△△△ ')
			CBLDCMCF('SENDRECV')

注

同期型の応答メッセージの受信（CBLDCMCF('RECVSYNC')）およびメッセージの再送（CBLDCMCF('RESEND △△ ')）に相当するデータ操作言語のインタフェースはありません。

dc_mcf_receive － 一方送信メッセージの受信（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;
```

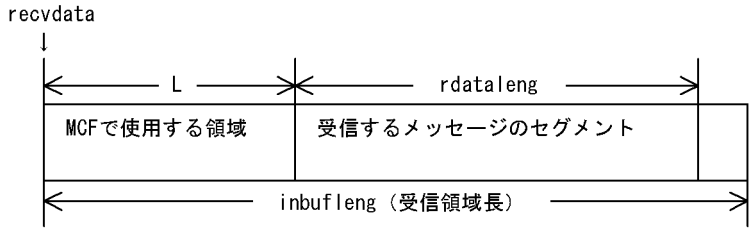
機能

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。

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

メッセージの先頭セグメントを受信するかどうか，および使用するバッファ形式を次の形式で設定します。

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

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

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

先頭セグメントを受信する場合は，入力元の論理端末名称が返される領域を設定します。dc_mcf_receive 関数が終了すると，入力元の論理端末名称が返されます。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろ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，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。

先頭セグメントの受信処理終了後，termnam には OpenTP1 から値が返されます。

● resv01

ヌル文字を設定します。

● recvdata

セグメントを受信する領域を設定します。受信できるセグメントの最大長は，通信管理に依存します。

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

● 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

先頭セグメントを受信する場合だけ，入力元の論理端末名称が返されます。

ここで返された論理端末名称を，中間セグメントまたは最終セグメントの受信時に 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	意味
DCMCFRTN_00000	正常に終了しました。
DCMCFRTN_71000	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	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。

リターン値	意味
DCMCFRTN_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。
DCMCFRTN_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 dc_mcf_receive 関数を呼び出す前に，中間セグメントおよび最終セグメントを受信する 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 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	termnam に設定した論理端末名称が間違っています。
DCMCFRTN_72013	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	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	commform に設定した値が間違っています。
DCMCFRTN_72025	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	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。

dc_mcf_recvsync ― 同期型の応答メッセージの受信（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)
```

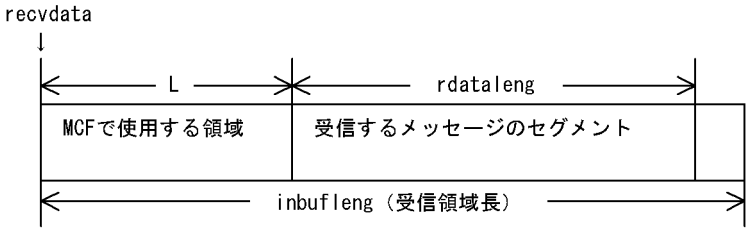
K&R 版 C の形式

```
#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

相手システムから同期型で受信し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_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。

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

メッセージの先頭セグメントを受信するかどうか，および使用するバッファ形式を次の形式で設定します。

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

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

DCMCFSEG
中間セグメントおよび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

入力元の論理端末名称を設定します。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろにはヌル文字を付けてください。

● resv01

ヌル文字を設定します。

● recvdata

セグメントを受信する領域を設定します。受信できるセグメントの最大長は，通信管理に依存します。

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

● 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	意味
DCMCFRTN_00000	正常に終了しました。
DCMCFRTN_71001	メッセージの最終セグメントを受信したあとで，その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 関数でその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_71002	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。
DCMCFRTN_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。
DCMCFRTN_72000	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	termnam に設定した論理端末名称が間違っています。
	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	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	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	commform に設定した値が間違っています。
DCMCFRTN_72025	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	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。

リターン値	意味
DCMCFRTN_73001	入力元の論理端末で障害が発生しました。
DCMCFRTN_73002	dc_mcf_recvsync 関数の処理中に，入力元の論理端末と MCF との間のコネクションまたは経路が切り離されました。
	dc_mcf_recvsync 関数が呼び出されるまでに，入力元の論理端末と MCF との間のコネクションまたは経路が切り離されました。
DCMCFRTN_73008	入力元の論理端末は，次のコマンドで停止しています。 • mcftdctle • mcftdctcn
DCMCFRTN_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。
	メッセージの読み込み時に障害が発生しました。
DCMCFRTN_73018	resv02 に設定した値が間違っています。
DCMCFRTN_73020	設定した論理端末は停止しています。
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。

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を基に選択できます。

- 出力先の論理端末名称
- メッセージ通番
- メッセージ種別（一般の一方送信または優先の一方送信）

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。この値を設定したときは，orgseq に設定した値は無効となります。

● 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。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろにはヌル文字を付けてください。

● resv01

ヌル文字を設定します。

● 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効となります。

● 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。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろにはヌル文字を付けてください。

● resv02，resv03，resv04

ヌル文字を設定します。

● opcd

DCNOFLAGS を設定します。

リターン値

リターン値	意味
DCMCFRTN_00000	正常に終了しました。
DCMCFRTN_NOMSG	該当するメッセージがありません。
DCMCFRTN_BUF_SHORT	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	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。
DCMCFRTN_71003	メッセージキューが満杯です。
DCMCFRTN_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。
DCMCFRTN_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。
DCMCFRTN_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 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。

リターン値	意味
	＜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	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	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	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	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。

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，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)
```

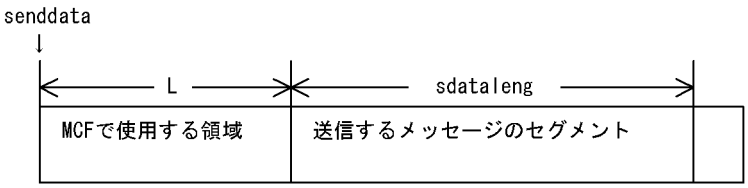
K&R 版 C の形式

```
#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。

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

メッセージの最終セグメントを送信するかどうか，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，メッセージが単一セグメントの場合に，DCMCFEMI を設定します。
　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
　出力通番が必要な場合に設定します。

DCMCFNSEQ
　出力通番が必要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。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろにはヌル文字を付けてください。

● resv01

ヌル文字を設定します。

● 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。

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場合も，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。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場合は，0 を設定します。

● resv02

ヌル文字を設定します。

● opcd

DCNOFLAGS を設定します。

リターン値

リターン値	意味
DCMCFRTN_00000	正常に終了しました。
DCMCFRTN_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。
	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	メッセージキューが満杯です。
DCMCFRTN_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。
DCMCFRTN_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。
DCMCFRTN_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 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	termnam に設定した論理端末名称が間違っています。
	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	先頭セグメントまたは中間セグメント送信時，sdataleng に 0 以下の値を設定しています。
DCMCFRTN_72016	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。

リターン値	意味
	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	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	commform に設定した値が間違っています。
DCMCFRTN_72026	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	単一セグメント送信時，sdataleng に 0 以下の値を設定しています。
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。

dc_mcf_sendrecv ― 同期型の問い合わせ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。

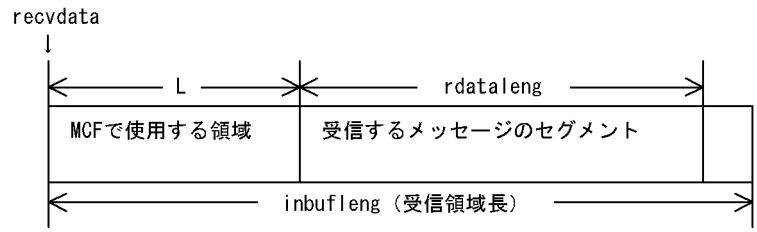
dc_mcf_sendrecv 関数では，相手システムへ送るメッセージのうち，一つのセグメントを送信できます。セグメントの数だけ dc_mcf_sendrecv 関数を呼び出すと，一つの論理メッセージを送信できます。

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。中間セグメントおよび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。

セグメントを送信する領域と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。

●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

メッセージの最終セグメントを送信するかどうか，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}〔｜{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，メッセージが単一セグメントの場合に，DCMCFEMI を設定します。
　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，論理端末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

同期型メッセージの送受信を示す，DCMCFIO を設定します。

● termnam

出力先の論理端末名称を設定します。論理端末名称の長さは最大 8 バイトです。論理端末名称の後ろにはヌル文字を付けてください。

● resv01

ヌル文字を設定します。

● 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。

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場合も，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。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場合は，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。受信できるセグメントの最大長は，通信管理に依存します。

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，recvdata には受信したメッセージの先頭セグメントが返されます。

● 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（mcfmuap -t）で指定した，同期型送受信監視時間が設定されます。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	意味
DCMCFRTN_00000	正常に終了しました。
DCMCFRTN_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。
	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	メッセージキューが満杯です。
DCMCFRTN_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。
DCMCFRTN_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。
DCMCFRTN_72000	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	termnam に設定した論理端末名称が間違っています。
	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	先頭セグメントまたは中間セグメント送信時，sdataleng に 0 以下の値を設定しています。
DCMCFRTN_72013	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	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	commform に設定した値に間違いがあります。
DCMCFRTN_72026	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	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	単一セグメント送信時，sdataleng に 0 以下の値を設定しています。
DCMCFRTN_72073	非同期型のメッセージの送信処理が仕掛り中です。
DCMCFRTN_73001	出力先の論理端末で障害が発生しました。
DCMCFRTN_73002	MCF 通信サービスで障害が発生しました。

リターン値	意味
DCMCFRTN_73005	watchtime に設定した時間が経過しましたが，論理端末からの応答がありません。
	応答ダミーデータ（空白メッセージ）を受信しました。
DCMCFRTN_73008	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	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。
	メッセージの読み込み時に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	出力先の論理端末は，ほかの UAP で仕掛り中です。
DCMCFRTN_73018	watchtime に設定した値が間違っています。
DCMCFRTN_73020	出力先の論理端末は停止しています。
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('RECEIVE ') ― 一方送信メッセージの受信（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).
   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

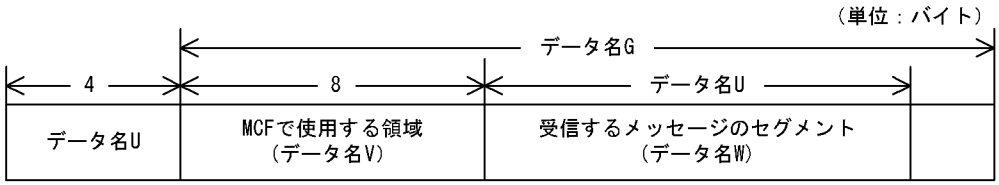
論理端末に届いたメッセージのうち，一つのセグメントを受信します。セグメントの数だけ CBLDCMCF('RECEIVE △') を呼び出すと，一つの論理メッセージを受信できます。

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

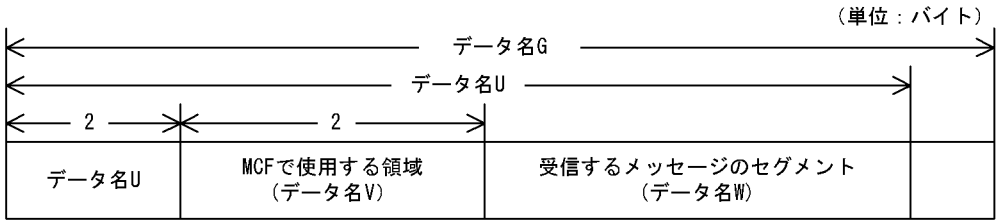
- 相手システムから送信されたメッセージ
- MCF イベント
- アプリケーション起動で渡されたメッセージ

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



●バッファ形式 2 の場合



UAP で値を設定するデータ領域

●データ名 A

メッセージの受信を示す要求コードを「VALUE 'RECEIVE △'」と設定します。

●データ名 C

メッセージの先頭セグメントを受信するかどうかを設定します。次のどちらかを設定してください。

VALUE 'FRST'

先頭セグメントを受信する場合や，メッセージが単一セグメントの場合に，「VALUE 'FRST'」を設定します。

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

先頭セグメントを受信する場合は，この領域には入力元の論理端末名称が返されます。

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，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。

先頭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

リターンコードが，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

先頭セグメントを受信する場合だけ，入力元の論理端末名称が返されます。

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データ名 P に設定してください。

●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に +4 を加算した値が返されます。

●データ名 W

受信したセグメントの内容が返されます。

ステータスコード

ステータスコード	意味
00000	正常に終了しました。
71000	先頭セグメントを受信する CBLDCMCF('RECEIVE △ ') を 2 回以上呼び出しています。中間セグメントおよび最終セグメントを受信する場合は，データ名 C に「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。このリターン値が返されたあとに，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。
	メッセージキューが閉塞されています。
71108	プロセスのローカルメモリが不足しています。
72000	＜ MHP の実行でリターンした場合＞ • 先頭セグメントを受信する CBLDCMCF('RECEIVE △ ') を呼び出す前に，中間セグメントおよび最終セグメントを受信する CBLDCMCF('RECEIVE △ ') を呼び出しています。先頭セグメントを呼び出す場合は，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。
	バッファ形式 2 の場合で，32763 バイトを超えるセグメントを受信しました。32763 バイトを超えた部分は切り捨てられました。
72016	データ名 D に設定した値が間違っています。
	データ名 N またはデータ名 T に設定した値が間違っています。
	データ名 M7 に設定した値が間違っています。
72024	データ名 O に設定した値が間違っています。
72025	データ名 C に設定した値が間違っています。
72028	データ名 A に設定した値が間違っています。
72036	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') ー同期型の応答メッセージの受信（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 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X   PIC X(1).
   02 データ名Z   PIC X(n).
```

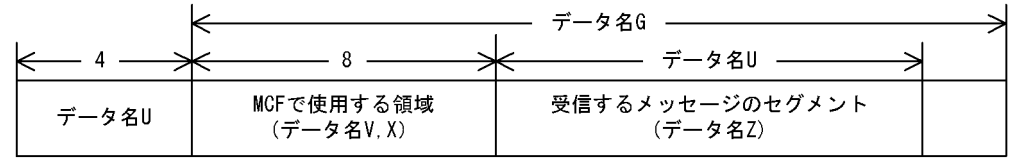
機能

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。

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

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を「VALUE 'RECVSYNC'」と設定します。

●データ名 C

メッセージの後続セグメントを受信する「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次の値で設定します。

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 を使用することを設定します。

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 バイトです。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 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(7)`

【バッファ形式 2 の場合】 `PIC X(1)`
　MCF で使用する領域です。

●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。

●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)
　受信したメッセージの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)
　受信したメッセージのセグメントの長さに +4 を加算した値が返されます。

●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。
71001	メッセージの最終セグメントを受信したあとで，その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') でその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。
71002	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	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。
	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。
	バッファ形式 2 の場合で，32763 バイトを超えるセグメントを受信しました。32763 バイトを超えた部分は切り捨てられました。

ステータスコード	意味
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。
73001	入力元の論理端末で障害が発生しました。
73002	CBLDCMCF('RECVSYNC') の処理中に，入力元の論理端末と MCF との間のコネクションまたは経路が切り離されました。
	CBLDCMCF('RECVSYNC') が呼び出されるまでに，入力元の論理端末と MCF との間のコネクションまたは経路が切り離されました。
73008	入力元の論理端末は，次のコマンドで停止しています。 • mcftdctle • mcftdctcn
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。
	メッセージの読み込み時に障害が発生しました。
73018	データ名 M5 に設定した値が間違っています。
73020	設定した論理端末は停止しています。
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('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を基に選択できます。

- 出力先の論理端末名称
- メッセージ通番
- メッセージ種別（一般の一方送信または優先の一方送信）

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を「VALUE 'RESEND △△'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 E，データ名 F

MCF で使用する領域です。

●データ名 G

0 を設定します。

●データ名 H，データ名 I

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

メッセージ出力先の論理端末名称を設定します。論理端末名称の長さは最大 8 バイトです。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 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。論理端末名称の長さは最大 8 バイトです。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 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	該当するメッセージがありません。
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。

ステータスコード	意味
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。
	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，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

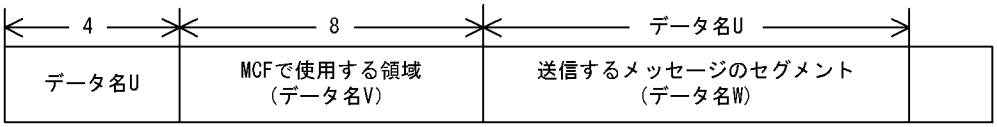
```
01 一意名1.
   02 データ名A   PIC X(8)  VALUE 'SEND    '.
   02 データ名B   PIC X(5).
   02 FILLER    PIC X(3).
   02 データ名C   PIC X(4)  VALUE SPACE.
   02 データ名D   PIC X(4)  VALUE SPACE.
   02 データ名E   PIC 9(8).
   02 データ名F   PIC 9(8).
   02 データ名G   PIC 9(9)  COMP VALUE ZERO.
   02 データ名H   PIC X(4).
   02 デー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

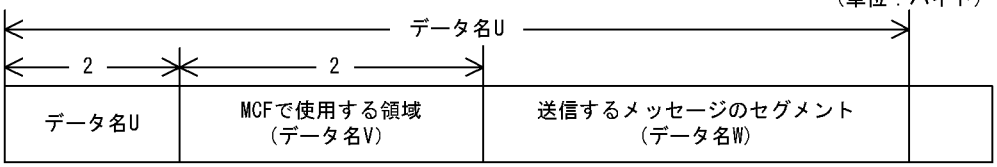
相手システムへ送る一方送信メッセージのうち，一つのセグメントを送信します。セグメントの数だけ CBLDCMCF('SEND △△△△') を呼び出すと，一つの論理メッセージを送信できます。

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

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

●データ名 C，データ名 D

空白を設定します。

●データ名 E，データ名 F

MCF で使用する領域です。

●データ名 G

0 を設定します。

●データ名 H

メッセージの最終セグメントを送信するかどうかを設定します。次のどちらかを設定します。

VALUE 'ESI △ '

先頭セグメントおよび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，メッセージが単一セグメントの場合に，「VALUE 'EMI △ '」を設定します。

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 △'
　出力通番が必要な場合に設定します。

VALUE 'NSEQ'
　出力通番が必要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。論理端末名称の長さは最大 8 バイトです。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 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。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に +4 を加算した値を設定します。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。先頭セグメントまたは中間セグメントの送信後，メッセージの送信の終了を連絡する場合にも必ず設定してください。一つのセグメントで 32000 バイトまで送信できます。

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。
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 U 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。
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。
	CBLDCMCF('SEND △△△△ ') を呼び出せない論理端末を設定しています。
72005	バッファ形式 1 の先頭セグメントまたは中間セグメント送信時，データ名 U に 0 以下の値を設定しています。または，バッファ形式 2 の先頭セグメントまたは中間セグメント送信時，データ名 U に 4 以下の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。

ステータスコード	意味
72041	バッファ形式 1 の単一セグメント送信時，データ名 U に 0 以下の値を設定しています。または，バッファ形式 2 の単一セグメント送信時，データ名 U に 4 以下の値を設定しています。
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') ― 同期型の問い合わせ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).
```

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。

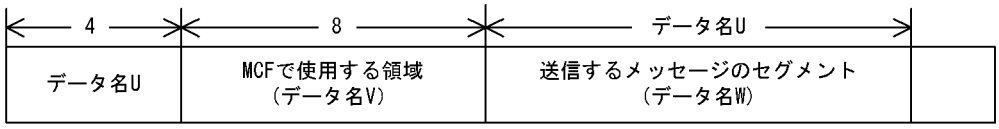
CBLDCMCF('SENDRECV') では，相手システムへ送るメッセージのうち，一つのセグメントを送信できます。セグメントの数だけ CBLDCMCF('SENDRECV') を呼び出すと，一つの論理メッセージを送信できます。

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。

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。

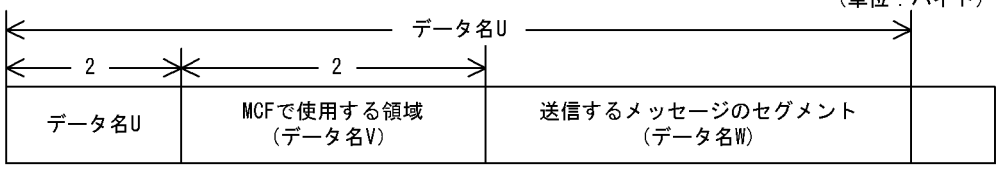
●バッファ形式 1 の場合

（単位：バイト）



●バッファ形式 2 の場合

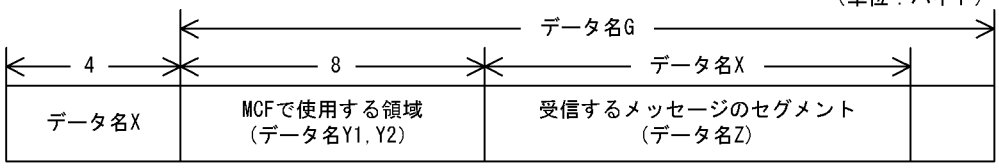
（単位：バイト）



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。

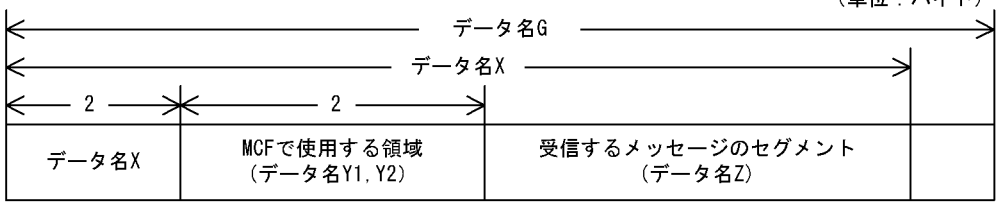
●バッファ形式 1 の場合

（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を「VALUE 'SENDRECV'」と設定します。

●データ名 C，データ名 D

空白を設定します。

●データ名 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

メッセージの最終セグメントを送信するかどうかを設定します。次のどちらかを設定してください。

VALUE 'ESI △ '
　先頭セグメントおよび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，メッセージが単一セグメントの場合に，「VALUE 'EMI △ '」を設定します。
　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，論理端末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（mcfmuap -t）で指定した，同期型送受信監視時間が設定されます。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

一方送信を示す「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。論理端末名称の長さは最大 8 バイトです。8 バイトに満たない名称を設定する場合は，後ろを空白で埋めてください。

●データ名 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。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に＋ 4 を加算した値を設定します。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。先頭セグメントまたは中間セグメントの送信後，メッセージの送信の終了を連絡する場合にも必ず設定してください。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 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。
	データ名 U 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。
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	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。
	CBLDCMCF('SENDRECV') を呼び出せない論理端末を設定しています。
72005	バッファ形式 1 の先頭セグメントまたは中間セグメント送信時，データ名 U に 0 以下の値を設定しています。または，バッファ形式 2 の先頭セグメントまたは中間セグメント送信時，データ名 U に 4 以下の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 の指定値の長さを超えた部分は切り捨てられました。
	バッファ形式 2 の場合で，32763 バイトを超えるセグメントを受信しました。32763 バイト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	バッファ形式 1 の単一セグメント送信時，データ名 U に 0 以下の値を設定しています。または，バッファ形式 2 の単一セグメント送信時，データ名 U に 4 以下の値を設定しています。
72073	非同期型のメッセージの送信処理が仕掛り中です。

ステータスコード	意味
73001	出力先の論理端末で障害が発生しました。
73002	MCF 通信サービスで障害が発生しました。
73005	データ名 M5 に設定した時間が経過しましたが，論理端末からの応答がありません。
	応答ダミーデータ（空白メッセージ）を受信しました。
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。
	メッセージの読み込み時に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。
73020	出力先の論理端末は停止しています。
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。

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〕.
```

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。

- 一方送信メッセージの受信　CBLDCMCF('RECEIVE △ ')

通信記述項に設定する項目

● FOR 句

次の値を指定します。

INPUT
　一方送信メッセージの受信

● STATUS KEY 句

ステータスコードを受け取りたい場合に指定します。省略した場合は，ステータスコードを受け取りません。

● SYMBOLIC TERMINAL 句

入力元の論理端末名称を参照するデータ項目を指定します。

● MESSAGE DATE 句

メッセージを受信した日付を参照するデータ項目を指定します。YYMMDD（YY：西暦の下 2 けた　MM：月　DD：日）の形式で参照できます。

● MESSAGE TIME 句

メッセージを受信した時刻を参照するデータ項目を指定します。HHMMSS00（HH：時　MM：分　SS：秒　00 は固定）の形式で参照できます。

通信文に指定する項目

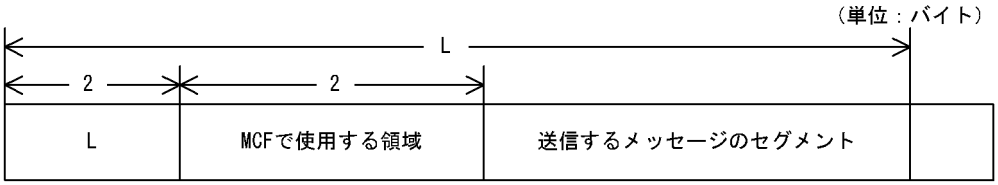
● FIRST

先頭セグメントを受信する場合に指定します。

●一意名 1

セグメントを受信するデータ項目を指定します。

一意名 1 の形式を次に示します。



ステータスコード

ステータスコード	意味
00000	正常に終了しました。
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 を指定しないで RECEIVE 文を実行してください。
71001	メッセージの最終セグメントを受信したあとで，その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 文でその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。

ステータスコード	意味
72000	＜ MHP の実行でリターンした場合＞ • 先頭セグメントを受信する 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。 • ステータスコード 71001 が返されたあとで，再び RECEIVE 文を実行しています。
	＜ SPP の実行でリターンした場合＞ SPP では RECEIVE 文を実行できません。
72001	SYMBOLIC TERMINAL 句に設定した論理端末名称が間違っています。
	RECEIVE 文を実行できない論理端末を設定しています。
72013	受信領域の長さを超えるセグメントを受信しました。受信領域の長さを超えた部分は切り捨てられました。
	一意名 1 の L の指定値を超えるセグメントを受信しました。一意名 1 の L の指定値を超えた部分は切り捨てられました。
72024	FOR 句の指定が間違っています。
72036	一意名 1 の L の指定値が不足しています。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〕.
```

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 △△△△ ')
- 同期型の問い合わせメッセージの送受信　CBLDCMCF('SENDRECV')

同期型の問い合わせ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。

通信記述項に設定する項目

● FOR 句

次のどちらかの値を指定します。

OUTPUT
　一方送信メッセージの送信

I-O
　同期型のメッセージの送受信

● STATUS KEY 句

ステータスコードを受け取りたい場合に指定します。この指定を省略した場合は，ステータスコードを受け取れません。

● SYMBOLIC TERMINAL 句

論理端末名称を設定したデータ項目を指定します。

● SYNCHRONOUS MODE 句

メッセージ送信が，同期型か非同期型かを指定します。

SYNC
　同期型のメッセージ送受信。
　同期型のメッセージ送受信の場合に設定します。

ASYNC
　非同期型のメッセージ送信。
　一方送信メッセージ送信の場合に設定します。

データ名 6
　次の値を設定したデータ項目
　'0'：非同期型のメッセージ送信
　'1'：同期型のメッセージ送受信
　指定を省略した場合は，非同期のメッセージ送信（ASYNC）が仮定されます。

● SWITCHING MODE 句

一方送信メッセージの場合に，一般か優先かを指定します。

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
指定を省略した場合は，一般（NORMAL）が仮定されます。

● DETAIL MODE 句

一方送信メッセージの場合に，出力通番を付けるかどうかを指定します。

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信，または同期型のメッセージを送受信する場合に指定します。

指定を省略した場合，または '00000000' を設定した場合は，MCF マネジャ定義の UAP 共通定義（mcfmuap -t）で指定した同期型送信監視時間または同期型送受信監視時間が設定されます。

データ名 11

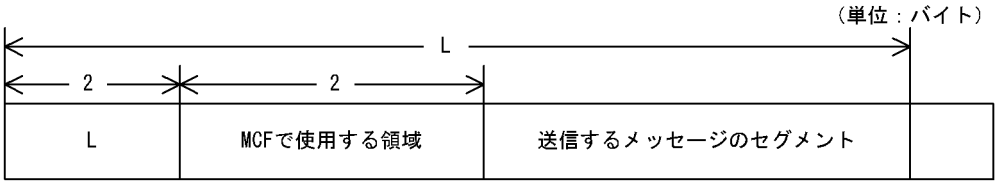
監視時間の値を HHMMSS00（HH：時　MM：分　SS：秒　00 は固定）の形式で設定したデータ項目

通信文に指定する項目

●一意名 1

セグメントを送信するデータ項目を指定します。一つのセグメントで 32000 バイトまで送信できます。

一意名 1 の形式を次に示します。



● WITH 句

メッセージの最終セグメントを送信するかどうかを指定します。非同期型のメッセージ送信の場合だけ指定します。

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

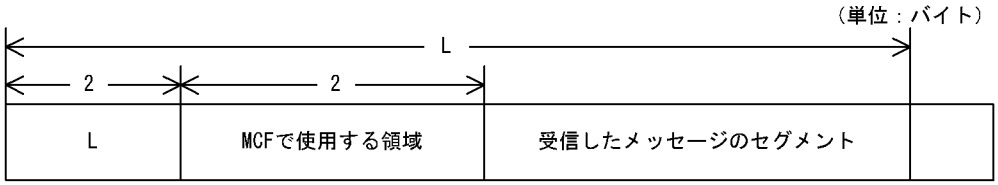
一意名 2

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）
'2'：EMI（最終セグメントまたは単一セグメント）
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を送受信する場合に，セグメントを受信するデータ項目（一意名 3）を指定します。

一意名 3 の形式を次に示します。



ステータスコード

ステータスコード	意味
00000	正常に終了しました。
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。
	一意名 1 の L 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。
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。

ステータスコード	意味
72000	＜MHP の実行でリターンした場合＞ • 応答型でない MHP から応答メッセージを送信しています。 • 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。
	＜SPP の実行でリターンした場合＞ • 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。 • 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。
72001	SYMBOLIC TERMINAL 句に設定した論理端末名称が間違っています。
	SEND 文を実行できない論理端末を設定しています。
72005	先頭セグメントまたは中間セグメント送信時，一意名 1 の L に 4 以下の値を設定しています。
72013	一意名 3 の L の指定値を超えるセグメントを受信しました。一意名 3 の L の指定値を超えた部分は切り捨てられました。
	32763 バイトを超えるセグメントを受信しました。32763 バイト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	一意名 3 の L の指定値が不足しています。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	単一セグメント送信時，一意名 1 の L に 4 以下の値を設定しています。
72073	非同期型のメッセージの送信処理で仕掛り中です。
73001	出力先の論理端末で障害が発生しました。
73002	MCF 通信サービスで障害が発生しました。
73005	WAITING TIME 句に設定した時間が経過しましたが，論理端末からの応答がありません。
73008	論理端末が閉塞中，または MCF が終了処理中に，SEND 文を実行しました。
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。
	メッセージの読み込み時に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。
73020	出力先の論理端末は停止しています。
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。

ユーザアプリケーションプログラム作成例

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）と COBOL 言語で示します。

また，TP1/NET/SLU-TypeP2 では，このコーディング例を次のファイルで提供しています。

C 言語（K&R 版）のコーディング例

- /BeTRAN/examples/mcf/SLUP2/aplib/c/ap1.c
- /BeTRAN/examples/mcf/SLUP2/aplib/c/sv1.c
- /BeTRAN/examples/mcf/SLUP2/aplib/c/sv2.c

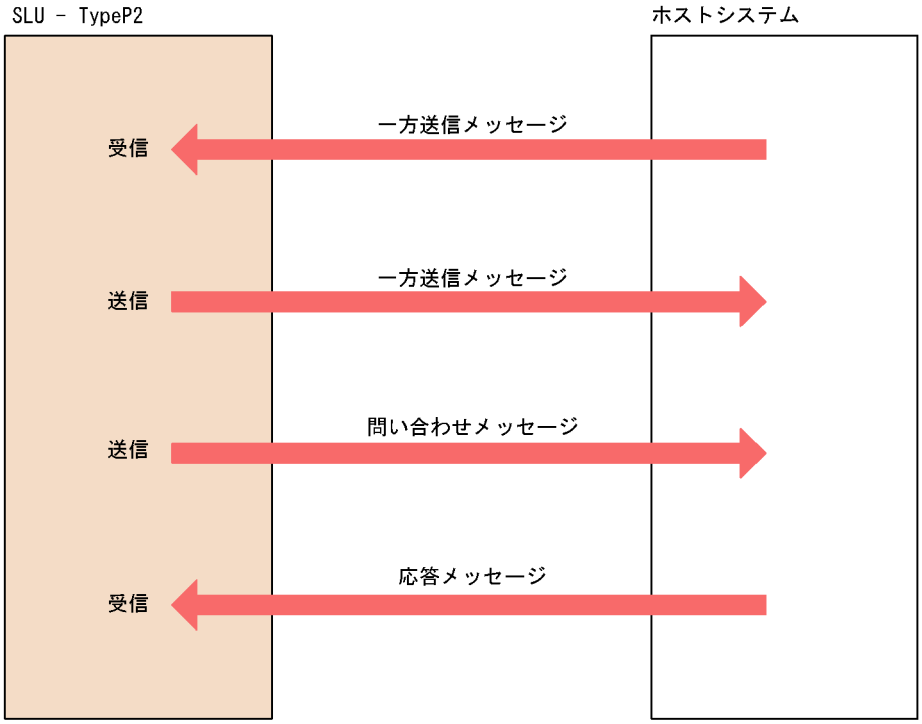
COBOL 言語のコーディング例

- /BeTRAN/examples/mcf/SLUP2/aplib/cobol/ap1.cbl
- /BeTRAN/examples/mcf/SLUP2/aplib/cobol/sv1.cbl
- /BeTRAN/examples/mcf/SLUP2/aplib/cobol/sv2.cbl

データ操作言語のコーディング例

- /BeTRAN/examples/mcf/SLUP2/aplib/dml/ap1.cbl
- /BeTRAN/examples/mcf/SLUP2/aplib/dml/sv1.cbl

図 3-1　処理の流れの例



C 言語

```
#include<dcmcf.h>

#define rcvmax 5120
void slmhprcv1()
{
   char            termnam[16];
   char            mapname[16];
   char            sdatarea[2048];
   char            rdatarea[rcvmax];
   DCLONG          sdataleng;
   DCLONG          rdataleng;
   DCLONG          time;

   int             rtn;

   memset(mapname, 0, 16);
   memset(sdatarea, 0, 2048);
   memset(rdatarea, 0, rcvmax);

   rtn =dc_mcf_receive(   DCMCFFRST,
                          DCNOFLAGS,
                          termnam,
                          mapname,
                          rdatarea,
                          &rdataleng,
```

```
                        rcvmax,
                        &time   );

   /*****************************************************/
   /*                                                   */
   /*   User Service Part                               */
   /*                                                   */
   /*****************************************************/

   if(rtn !=DCMCFRTN_00000)
   {
      goto ERROR;
   }
ERROR:;
   return;
}


#include<dcmcf.h>

#define sndmax   448
#define rcvmax   5120
#define segmax   (sndmax -9 -3 -4)          /*432 bytes */

/*448 bytes */
typedef   struct senddata_s{
   char    mcf_use[8];
   char    ap_name[9];
   char    snd_dmy[3];
   char    snd_lng[4];
   char    snd_seg[segmax];
}sdata_def;

void   slsppsnd1()
{
   static char   *lename ="SL2REQ01";
   char            termnam[9];
   char            resv01[9];
   char            resv02[9];
   char            resv03[9];
   DCLONG          rdataleng;
   DCLONG          sdataleng;
   DCLONG          inbufleng;
   sdata_def       senddata;
   int     rtn_code;

   memset(resv01, 0, 9);
   memset(resv02, 0, 9);
   memset(resv03, 0, 9);

   memset(&senddata, 0x33, sizeof(senddata));

   memcpy(senddata.ap_name, "slup2snd ", 9);
   memcpy(senddata.snd_lng, "0448", 4);
   memcpy(senddata.snd_seg, "send:aaaaaaaaaa1aaaaaaaaaaaaaaa2
aaaaaaaaaaaaaaa3aaaaaaaaaaaaaaa4aaaaaaaaaaaaaaa5
aaaaaaaaaaaaaaa6aaaaaaaaaaaaaaa7aaaaaaaaaaaaaaa8
aaaaaaaaaaaaaaa9aaaaaaaaaaaaaaaAaaaaaaaaaaaaaaaB
```

```
aaaaaaaaaaaaaaaCaaaaaaaaaaaaaaaDaaaaaaaaaaaaaaaE
aaaaaaaaaaaaaaaFaaaaaaaaaaaaaaa0", 256);
   memcpy(senddata.snd_seg + segmax -3, "end", 3);

   rtn_code =dc_mcf_send(   DCMCFEMI,
                            DCMCFOUT,
                            lename,
                            resv01,
                            &senddata,
                            sndmax,
                            resv02,
                            DCNOFLAGS   );
   if(rtn_code !=DCMCFRTN_00000)
   {
      goto ERROR;
   }
ERROR:;
   return;
}


#include <dcmcf.h>

#define sndmax   448
#define rcvmax   5120
#define segmax   (sndmax -9 -3 -4)/*432 bytes */

/*448 bytes */
typedef   struct senddata_s{
   char    mcf_use[8];
   char    ap_name[9];
   char    snd_dmy[3];
   char    snd_lng[4];
   char    snd_seg[segmax];
}sdata_def;

void   slsppreq1()
{
   static char   rcv_buf[rcvmax];
   static char   *lename ="SL2REQ01";
   char            termnam[9];
   char            resv01[9];
   char            resv02[9];
   char            resv03[9];
   DCLONG          rdataleng;
   DCLONG          sdataleng;
   DCLONG          inbufleng;
   sdata_def       senddata;
   DCLONG          rcvtime;
   int             rtn_code;

   memset(resv01, 0, 9);
   memset(resv02, 0, 9);
   memset(resv03, 0, 9);

   memset(&senddata, 0x33, sizeof(senddata));

   memcpy(senddata.ap_name, "slup2req ", 9);
```

```
   memcpy(senddata.snd_lng, "0448", 4);
   memcpy(senddata.snd_seg, "*FMH*aaaaaaaaaa1aaaaaaaaaaaaaaa2
aaaaaaaaaaaaaaa3aaaaaaaaaaaaaaa4aaaaaaaaaaaaaaa5
aaaaaaaaaaaaaaa6aaaaaaaaaaaaaaa7aaaaaaaaaaaaaaa8
aaaaaaaaaaaaaaa9aaaaaaaaaaaaaaaAaaaaaaaaaaaaaaaB
aaaaaaaaaaaaaaaCaaaaaaaaaaaaaaaDaaaaaaaaaaaaaaaE
aaaaaaaaaaaaaaaFaaaaaaaaaaaaaaa0", 256);
   memcpy(senddata.snd_seg + segmax -3, "end", 3);

   rtn_code =dc_mcf_sendrecv (DCMCFEMI,
                              DCMCFIO,
                              lename,
                              resv01,
                              &senddata,
                              sndmax,
                              rcv_buf,
                              &rdataleng,
                              rcvmax,
                              &rcvtime,
                              0);

   if(rtn_code != DCMCFRTN_00000)
   {
      goto ERROR;
   }
ERROR:;
   return;
}
```

COBOL 言語

```
 IDENTIFICATION    DIVISION.
 PROGRAM-ID.       UAPCBL.

 ENVIRONMENT       DIVISION.

 DATA              DIVISION.
 WORKING-STORAGE   SECTION.
 01   RCV.
      02  MSG-REQ          PIC X(8)   VALUE   'RECEIVE '.
      02  RTN              PIC X(5).
      02  FILLER           PIC X(3).
      02  RSV1             PIC X(4)   VALUE   'FRST'.
      02  RSV2             PIC X(4)   VALUE   SPACE.
      02  MCFUSING1        PIC 9(8).
      02  MCFUSING2        PIC 9(8).
      02  RSV3             PIC 9(9)   COMP    VALUE   2048.
      02  SEGKIND          PIC X(4)   VALUE   SPACE.
      02  RSV4             PIC X(4)   VALUE   SPACE.
      02  MSGKIND          PIC X(4)   VALUE   SPACE.
      02  OUTPUTNO         PIC X(4)   VALUE   SPACE.
      02  RSV5             PIC X(8)   VALUE   SPACE.
      02  RSV6             PIC X(4)   VALUE   SPACE.
      02  RSV7             PIC X(8)   VALUE   SPACE.
      02  RSV8             PIC X(4)   VALUE   SPACE.
      02  RSV9             PIC 9(9)   COMP    VALUE   ZERO.
```

```
     02  RSV10            PIC 9(9)   COMP    VALUE   ZERO.
     02  RSV11            PIC X(1)   VALUE   SPACE.
     02  RSV12            PIC X(1)   VALUE   SPACE.
     02  RSV13            PIC X(14)  VALUE   LOW-VALUE.
 01  CD1.
     02  SEG-CODE1        PIC X(4)   VALUE   SPACE.
     02  TERM-CODE        PIC X(8)   VALUE   SPACE.
     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).

 PROCEDURE DIVISION.
************************************************************
 CALL 'CBLDCMCF'  USING   RCV    CD1    DATA1.
*
************************************************************
*                                                          *
*  User Service Part                                       *
*                                                          *
************************************************************
*
 EXIT PROGRAM.

 IDENTIFICATION    DIVISION.
 PROGRAM-ID.       SPPSND.

 ENVIRONMENT       DIVISION.
 CONFIGURATION     SECTION.

 DATA              DIVISION.
 WORKING-STORAGE   SECTION.

 01   SND-AREA1.
      02  DATA-A           PIC X(8)   VALUE   'SEND    '.
      02  DATA-B           PIC X(5).
      02  FILLER           PIC X(3).
      02  DATA-C           PIC X(4)   VALUE   SPACE.
      02  DATA-D           PIC X(4)   VALUE   SPACE.
      02  DATA-E           PIC 9(8).
      02  DATA-F           PIC 9(8).
      02  DATA-G           PIC 9(9)   COMP    VALUE   ZERO.
      02  DATA-H           PIC X(4)   VALUE   'EMI '.
      02  DATA-I           PIC X(4)   VALUE   SPACE.
      02  DATA-J           PIC X(4)   VALUE   'NORM'.
      02  DATA-K           PIC X(4)   VALUE   'NSEQ'.
      02  DATA-L           PIC X(8)   VALUE   SPACE.
      02  DATA-M1          PIC X(4)   VALUE   SPACE.
      02  DATA-M2          PIC X(8)   VALUE   SPACE.
      02  DATA-M3          PIC X(4)   VALUE   SPACE.
      02  DATA-M4          PIC 9(9)   COMP    VALUE   ZERO.
      02  DATA-M5          PIC 9(9)   COMP    VALUE   ZERO.
```

```
     02  DATA-M6          PIC X(1)   VALUE   SPACE.
     02  DATA-M7          PIC X(1)   VALUE   '1'.
     02  DATA-N           PIC X(14)  VALUE   LOW-VALUE.
01   SND-AREA2.
     02  DATA-O           PIC X(4)   VALUE   'OUT '.
     02  DATA-P           PIC X(8)   VALUE   'SL2REQ01'.
     02  DATA-Q           PIC X(8)   VALUE   SPACE.
     02  DATA-R           PIC X(8)   VALUE   SPACE.
     02  DATA-T           PIC X(28)  VALUE   LOW-VALUE.
01   SND-AREA3.
     02  DATA-U           PIC 9(9)   COMP   VALUE   248.
     02  DATA-V           PIC X(8).
     02  DATA-W           PIC X(248).

PROCEDURE         DIVISION.
CALL 'CBLDCMCF'  USING   SND-AREA1    SND-AREA2
     SND-AREA3.
EXIT PROGRAM.

IDENTIFICATION    DIVISION.
PROGRAM-ID.       SPPSR.

ENVIRONMENT       DIVISION.
CONFIGURATION     SECTION.

DATA              DIVISION.
WORKING-STORAGE   SECTION.

01   SR-AREA1.
     02  DATA-A           PIC X(8)   VALUE   'SENDRECV'.
     02  DATA-B           PIC X(5).
     02  FILLER           PIC X(3).
     02  DATA-C           PIC X(4)   VALUE   SPACE.
     02  DATA-D           PIC X(4)   VALUE   SPACE.
     02  DATA-E           PIC 9(8).
     02  DATA-F           PIC 9(8).
     02  DATA-G           PIC 9(9)   COMP    VALUE   248.
     02  DATA-H           PIC X(4)   VALUE   'EMI '.
     02  DATA-I           PIC X(4)   VALUE   SPACE.
     02  DATA-J           PIC X(4)   VALUE   SPACE.
     02  DATA-K           PIC X(4)   VALUE   SPACE.
     02  DATA-L           PIC X(8)   VALUE   SPACE.
     02  DATA-M1          PIC X(4)   VALUE   SPACE.
     02  DATA-M2          PIC X(8)   VALUE   SPACE.
     02  DATA-M3          PIC X(4)   VALUE   SPACE.
     02  DATA-M4          PIC 9(9)   COMP    VALUE   ZERO.
     02  DATA-M5          PIC 9(9)   COMP    VALUE   ZERO.
     02  DATA-M6          PIC X(1)   VALUE   SPACE.
     02  DATA-M7          PIC X(1)   VALUE   SPACE.
     02  DATA-N           PIC X(14)  VALUE   LOW-VALUE.
01   SR-AREA2.
     02  DATA-O           PIC X(4)   VALUE   'IO  '.
     02  DATA-P           PIC X(8)   VALUE   'SL2REQ01'.
     02  DATA-Q           PIC X(8)   VALUE   SPACE.
     02  DATA-R           PIC X(8)   VALUE   SPACE.
     02  DATA-T           PIC X(28)  VALUE   LOW-VALUE.
01   SR-AREA3.
     02  DATA-U           PIC 9(9)   COMP    VALUE   248.
```

```
      02  DATA-V           PIC X(8).
      02  DATA-W           PIC X(248).
 01   SR-AREA4.
      02  DATA-X           PIC 9(9)   COMP.
      02  DATA-Y1          PIC X(7)   VALUE   SPACE.
      02  DATA-Y2          PIC X(1).
      02  DATA-Z           PIC X(248).

 PROCEDURE         DIVISION.
 CALL 'CBLDCMCF'  USING   SR-AREA1  SR-AREA2
      SR-AREA3  SR-AREA4.

 EXIT PROGRAM.
```

データ操作言語

```
 IDENTIFICATION    DIVISION.
 PROGRAM-ID.       UAPDML.

 ENVIRONMENT       DIVISION.
 CONFIGURATION     SECTION.
*
************************************************************
*
 DATA              DIVISION.
 WORKING-STORAGE   SECTION.
*
************************************************************
*
 01   RECV-AREA1.
      02  RE-DATALENG1     PIC 9(4)   COMP   VALUE   1028.
      02  RE-RSV1          PIC X(2).
      02  RE-DATA1         PIC X(1024).
*
************************************************************
*
 COMMUNICATION SECTION.
*
************************************************************
*


     CD  RECV-IN1
     FOR  INPUT
     STATUS KEY IS                   RE-STATUS1
     SYMBOLIC TERMINAL IS            RE-TERMNAM
     MESSAGE DATE IS                 RE-DATE1
     MESSAGE TIME IS                 RE-TIME1.
*
************************************************************
*
 PROCEDURE DIVISION.
*
************************************************************
*
     RECEIVE  RECV-IN1
         FIRST  SEGMENT
```

```
          INTO  RECV-AREA1.
*
 EXIT PROGRAM.


 IDENTIFICATION    DIVISION.
 PROGRAM-ID.       SNDDML.

 ENVIRONMENT       DIVISION.
 CONFIGURATION     SECTION.
*
************************************************************
*
 DATA              DIVISION.
 WORKING-STORAGE   SECTION.
*
************************************************************
*
 01   SEND-AREA1.
      02  SE-DATALENG1   PIC 9(4)   COMP   VALUE   1028.
      02  SE-RSV1        PIC X(2).
      02  SE-DATA1       PIC X(1024).
*
************************************************************
*
 COMMUNICATION SECTION.
*
************************************************************
*

    CD  SEND-IN1
        FOR  OUTPUT
        STATUS KEY IS                 SE-STATUS1
        SYMBOLIC TERMINAL IS          SE-TERMNAM
        SYNCHRONOUS MODE IS           ASYNC
        SWITCHING MODE IS             NORMAL.
*
************************************************************
*
 PROCEDURE DIVISION.
*
************************************************************
*
    MOVE 'SL2REQ01'TO SE-TERMNAM.
    SEND SEND-IN1
        FROM SEND-AREA1
                WITH EMI.
*
 EXIT PROGRAM.
```

4 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，SLU - TypeP2 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。

4.1 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。

SLU - TypeP2 で使用する 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
- 送信メッセージの通番編集 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。

4.1.1 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。次に，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 MCF 通信構成定義（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

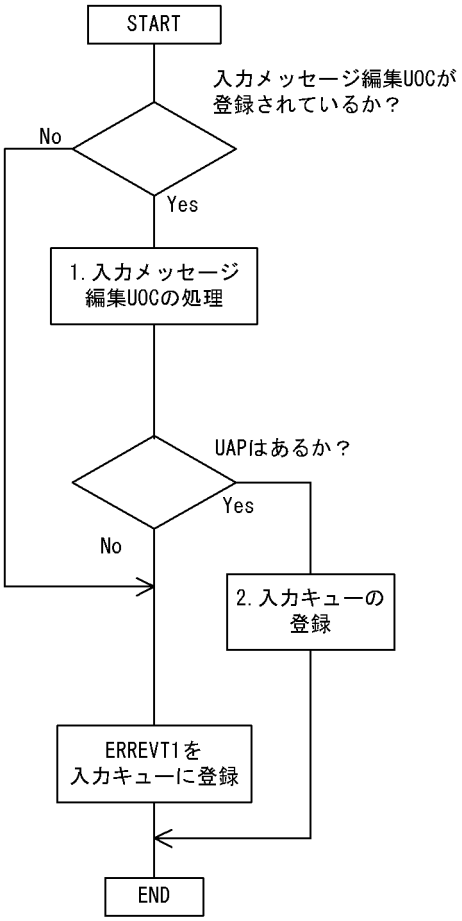
該当する MCF に入力メッセージ編集 UOC が登録されている場合，論理メッセージの受信と同時にアプリケーション名を決定できます。

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 バイトの英数字です。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 4-1　アプリケーション名の決定の処理



1. 入力メッセージの中の情報からアプリケーション名を決定し，指定領域（aplname）にアプリケーション名を設定します。
2. 入力キューの登録後，正常処理へ続きます。

（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，SLU - TypeP2 は論理端末を閉塞し，障害通知イベント（CERREVT）を通知します。

UOC で障害を検出し，MCF イベント処理用 MHP を起動したい場合は，ユーザ任意の MCF イベント処理用 MHP のアプリケーション名を設定します。MCF には

DCMCF_UOC_MSG_OK，または DCMCF_UOC_MSG_OK_RCV でリターンします。ただし，この場合は SLU - TypeP2 は正常なメッセージとして処理するため，UAP 側で障害処理をしてください。

（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，SLU - TypeP2 は論理端末を閉塞し，障害通知イベント（CERREVT）を通知します。

（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，SLU - TypeP2 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，「7.2　MCF メイン関数の作成」を参照してください。

4.1.2　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++ の場合

```
#include<dcmcfuoc.h>

DCLONG    uoc_func(dcmcf_uoc_min_n *parm)
```

K&R 版 C の場合

```
#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;            … 受信サービスプリミティブ
    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
    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
    DCLONG buf_size;            … バッファ最大長
    DCLONG 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
    DCLONG 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_SLUP2

SLUTYPE-P プロトコル（2 次局）

● le_name

メッセージを入力した論理端末の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_BRD
一方送信メッセージ受信

DCMCF_UOC_RCV_REP_RE
応答メッセージ受信

● 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。
メッセージ編集バッファが未定義の場合，つまり，MCF 通信構成定義の mcftalccn コマンドの -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
MCF 通信構成定義の mcftalcle コマンドの -v オプションで指定した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，-19999 ～ -19000 の範囲で設定してください。

(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

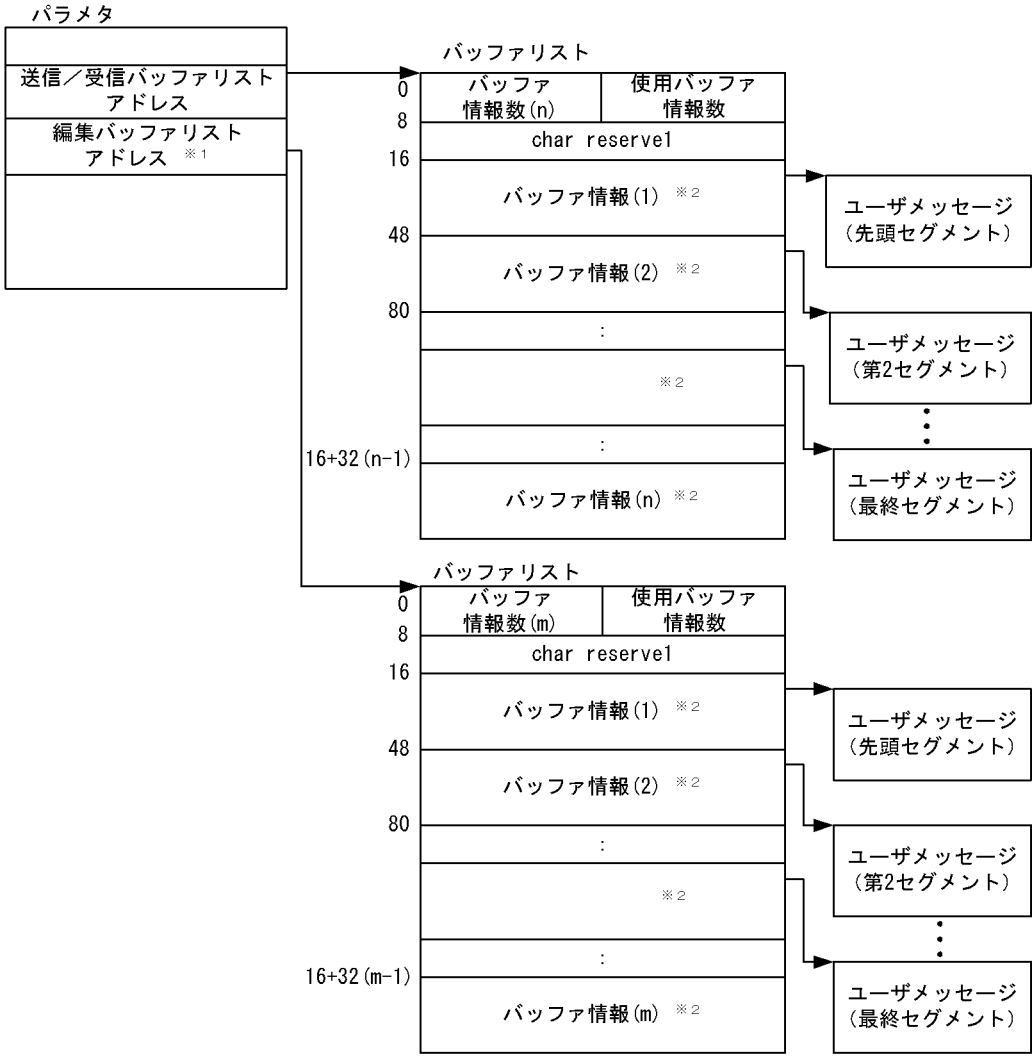
uoc_func() は次のコードでリターンしてください。

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 4-2　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 -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。

4.1.3 出力メッセージの編集

出力メッセージ編集 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 MCF 通信構成定義（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，SLU - TypeP2 は論理端末を閉塞します。なお，このとき MCF イベント（ERREVTn）は通知しません。

（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 はメッセージログを出力し，障害通知イベント（CERREVT）を通知します。該当するメッセージは破棄します。

（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。「4.1.1（5） OpenTP1 への組み

込み方法」を参照してください。

4.1.4 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++ の場合

```
#include<dcmcfuoc.h>

DCLONG    uoc_func(dcmcf_uoc_mout_n *parm)
```

K&R 版 C の場合

```
#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

「4.1.2 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_SLUP2
SLUTYPE-P プロトコル（2 次局）

● le_name

メッセージを出力した論理端末の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。
メッセージ編集バッファが未定義の場合，つまり，MCF 通信構成定義の mcftalccn コマンドの -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，次のどちらかの値が設定されます。

'n'
一般の一方送信メッセージ

'p'
優先の一方送信メッセージ

's'
同期問い合わせメッセージ

● 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）

「4.1.2　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，-19999 ～ -19000 の範囲で設定してください。

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

「4.1.2　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，入力メッセージの編集の場合と同じです。「4.1.2(7)　パラメタとバッファの関係」を参照してください。

4.1.5 送信メッセージの通番編集

送信メッセージの通番編集 UOC では，受け取った通番を基に，ユーザ独自の処理ができます。例えば，通番を使用して送信メッセージを編集できます。

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 が先頭セグメントを送信する send 関数または resend 関数を呼び出したときに，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，先頭セグメントしか編集できません。

(1) 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++ の場合

```
#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 の場合

```
#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 を設定します。

● uoc_addr

flags に対応する UOC のアドレスを設定します。

(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();
}
```

4.1.6 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++ の場合

```
#include<dcmcf.h>

DCLONG send_uoc(DCLONG flags, char *termname, DCLONG sendno,
                DCLONG sendid,DCLONG dataleng, char *senddata)
```

K&R 版 C の場合

```
#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 が値を設定する引数

● 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。次の値が設定されます。

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

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

● 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。次のどちらかが設定されます。

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

先頭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。

4.1.7 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

SLU - TypeP2 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() を発行すると，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。したがって，UAP より先に UOC が実行されたり，UAP がすべて終了してから UOC が呼び出されたりしてもよいように作成してください。

（5） ユーザセグメントの操作方法

UOC でユーザセグメントを参照または設定する場合，ユーザセグメントの先頭アドレスがバウンダリ調整されていないことを確認してください。ユーザセグメントの参照・設定方法によっては，バウンダリアクセス例外が発生する場合があります。必要に応じて，メモリ操作関数（memcpy，memset など）を使用してください。

4.2 MCF イベントインタフェース

SLU - TypeP2 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 を呼び出しません。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。

4.2.1 MCF イベントの種類

SLU - TypeP2 が通知する MCF イベントの種類を次の表に示します。

表 4-1 SLU - TypeP2 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 での処理の例
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。 • 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ました。 • MHP のセグメント受信関数に，セグメントを渡す前に MHP の異常終了が発生しました。 • 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。 • アプリケーションの即時起動時に障害が発生しました。 • MHP のアプリケーション，サービスグループがセキュア状態です。	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。

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 での処理の例
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）の入力によって，出力キューが削除されました。	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。	コネクションが確立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※

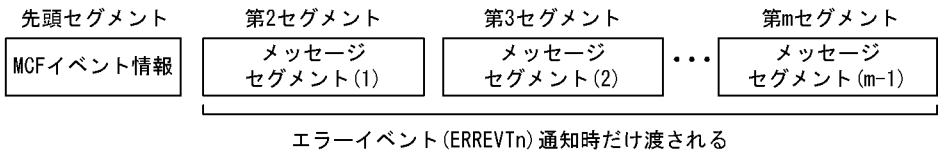
MCF アプリケーション定義 (mcfaalcap -g) の recvmsg オペランドに r を指定した場合，または dc_mcf_rollback の action に DCMCFRTRY または DCMCFRRTN を指定した場合は除きます。

4.2.2 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 4-3 MCF イベント通知時のセグメント構成



MCF イベントは，作成した UAP が C 言語の場合と COBOL 言語の場合で，UAP に通知されるデータの形式が異なります。

COBOL 言語を使用したエラーイベント（ERREVTn）の場合は，バッファ形式 1 とバッファ形式 2 と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。「4.2.4　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。

4.2.3　MCF イベント情報の形式（C 言語）

C 言語の場合は，イベント別の情報が，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に共通です。

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，dc_mcf_evtheader は <dcmcf.h> で定義されています。<dcmslm.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，
                                      ERREVT4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，ERREVT3 で，次に示す場合には，'*' が設定されます。

- SPP がアプリケーション起動をした MHP で，障害が発生した場合
- 上記の障害発生時に MCF イベントとして起動された MHP によって，さらにアプリケーション起動をされた 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，COPNEVT，または CCLSEVT の場合は無効です。

● 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，次に示す場合には，'*' が設定されます。

- SPP がアプリケーション起動をした MHP で，障害が発生した場合
- 上記の障害発生時に MCF イベントとして起動された MHP によって，さらにアプリケーション起動をされた 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イベント情報共通ヘッダ
    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イベント情報共通ヘッダ
    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。理由コードの内容については，「付録 C　理由コードおよびセンスコード一覧」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イベント情報共通ヘッダ
    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

トランザクションのブランチ ID が設定されます。

(5) ERREVTA

(a) 形式

```
struct dc_mcf_evta_type {
    struct dc_mcf_evtheader evtheader;
                                  … MCFイベント情報共通ヘッダ
    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イベント情報共通ヘッダ
    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;        … 回復動作情報
    char reserve1[42];            … 予備
}dcmslm_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 C　理由コードおよびセンスコード一覧」を参照してください。

● err_rcv_action

CERREVT 通知時に，回復動作情報として次の値が設定されます。

DCMSLM_RSV_MANUAL

コマンド入力による手動回復

DCMSLM_RSV_AUTO

システムによる自動回復

回復動作情報は，該当するコネクションが端末起動の場合は手動回復になり，ホスト起動の場合は自動回復になります。

（7）COPNEVT，CCLSEVT

(a) 形式

```
typedef struct{
    struct dc_mcf_evtheader header;
                                  … MCFイベント情報共通ヘッダ
    DCLONG cls_rcv_action;        … 回復動作情報
                                    （CCLSEVTの場合だけ有効）
    char reserve1[54];            … 予備
}dcmslm_statevt;
```

(b) MCF イベントとして設定される項目

● cls_rcv_action

CCLSEVT 通知時に，回復動作情報として次の値が設定されます。

DCMSLM_RSV_MANUAL

コマンド入力による手動回復

DCMSLM_RSV_AUTO

システムによる自動回復

回復動作情報は，該当するコネクションが端末起動の場合は手動回復になり，ホスト起動の場合は自動回復になります。

4.2.4　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表 4-2 ～表 4-7 に示します。

表 4-2 COBOL 言語の UAP に通知される MCF イベント情報の内容（ERREVT1）

項目	位置（バイト）		長さ	属性	内容
	形式 1	形式 2	（バイト）		
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	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	次に示すどちらかが設定されます。 • 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 4-3 COBOL 言語の UAP に通知される MCF イベント情報の内容（ERREVT2）

項目	位置（バイト）		長さ	属性	内容
	形式 1	形式 2	（バイト）		
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	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。次に示す場合には，'*' が設定されます。 • SPP がアプリケーション起動をした MHP で障害が発生した場合 • 上記の障害発生時に MCF イベントとして起動された MHP によって，さらにアプリケーション起動された 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が設定されます。
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。次に示す場合には，'*' が設定されます。 • SPP がアプリケーション起動をした MHP で障害が発生した場合 • 上記の障害発生時に MCF イベントとして起動された MHP によって，さらにアプリケーション起動された 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

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
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。
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※

理由コードの内容については，「付録 C　理由コードおよびセンスコード一覧」を参照してください。

表 4-4　COBOL 言語の UAP に通知される 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 がアプリケーション起動をした MHP で，障害が発生した場合 • 上記の障害発生時に MCF イベントとして起動された MHP が，さらにアプリケーション起動をした MHP で，障害が発生した場合
予備	20	16	20	—	—

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
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。 • SPP がアプリケーション起動をした MHP で，障害が発生した場合 • 上記の障害発生時に MCF イベントとして起動された MHP が，さらにアプリケーション起動をした MHP で，障害が発生した場合
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 ID（BID）	192	188	36	2 進数字	異常が発生したトランザクションの BID です。
予備	228	224	28	—	—

（凡例）

—：該当しません。または，使用されません。

表 4-5 COBOL 言語の UAP に通知される MCF イベント情報の内容（ERREVTA）

項目	位置（バイト）		長さ	属性	内容
	形式 1	形式 2	（バイト）		
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	—	—
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	—	—
予備	128	124	31	—	—
予備	159	155	1	—	—
予備	160	156	31	—	—
予備	191	187	1	—	—

項目	位置（バイト）		長さ（バイト）	属性	内容
	形式 1	形式 2			
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 4-6　COBOL 言語の UAP に通知される 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	障害要因コードが設定されます。 $(00000030)_{16}$：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。
回復動作情報	60	4	2 進数字	障害時，システムを回復する方法を示す値が設定されます。 $(00000000)_{16}$：システムによる自動回復 $(ffffffff)_{16}$：コマンド入力による手動回復
予備	64	44	—	—

（凡例）

—：該当しません。または，使用されません。

注※

理由コード 1 および理由コード 2 の内容については，「付録 C　理由コードおよびセ

ンスコード一覧」を参照してください。

表 4-7 COBOL 言語の UAP に通知される 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	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。
回復動作情報	48	4	2 進数字	障害時，システムを回復する方法を示す値が設定されます。 $(00000000)_{16}$：システムによる自動回復 $(ffffffff)_{16}$：コマンド入力による手動回復 回復動作情報は，該当するコネクションが端末起動の場合は手動回復になり，ホスト起動の場合は自動回復になります。
予備	52	56	—	—

（凡例）

—：該当しません。または，使用されません。

5 システム定義

この章では，SLUTYPE-P 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 SLU - TypeP2 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，および定義例について説明します。

SLU - TypeP2 の定義の概要

SLU - TypeP2 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，SLU - TypeP2 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 SLU - TypeP2 を起動するには，定義ファイルに環境情報を指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 5-1　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。表 5-1 の定義の中で，SLU - TypeP2 に固有の定義があるものを次に示します。

- MCF マネジャ定義
- MCF 通信構成定義

この章では，SLU - TypeP2 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。プロトコルで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 の注意事項に記載してあります。

SLU - TypeP2 の組み込み時に必要なファイル

次に示すファイルは，SLU - TypeP2 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。SLU - TypeP2 を組み込む方法については，「7. 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

SLUTYPE-P 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，SLU - TypeP2 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/AS）と関連づける内容を示します。

SLU - TypeP2 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，SLU - TypeP2 に固有の定義の一覧を次の表に示します。

表 5-2　SLU - TypeP2 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（SENDRECV）	符号なし整数 ((0 ～ 65535)) 《0》（単位：秒）
MCF 通信構成定義	共通定義	プロトコル共通のコマンドだけで指定できます。共通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。				
	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	slup2
			-g	sndbuf	送信用バッファグループ番号	符号なし整数 ((1 ～ 512))
				rcvbuf	受信用バッファグループ番号	符号なし整数 ((1 ～ 512))
			-e	msgbuf	編集用バッファグループ番号	符号なし整数 ((1 ～ 512))
				count	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	xnfas
			-i	—	コネクションの確立方法	auto ｜ 《manual》
			-b	bretry	障害時にコネクション確立の再試行をするかどうか	《yes》 ｜ no
				bretrycnt	コネクション確立再試行の回数	符号なし整数 ((0 ～ 65535)) 《0》（単位：回）
				bretryint	コネクション確立再試行の間隔	符号なし整数 ((0 ～ 2550)) 《60》（単位：秒）
			-k	—	起動種別	host ｜ ws
			-n	ownnode	自局ノード名称	符号なし整数 ((0 ～ 253))

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》
			-q	hostnode	ホストノード名称	1 ～ 8 けたの 16 進数字
			-r	sndrusiz	送信最大 RU 長	符号なし整数 ((0 ～ 32767))
				rcvrusiz	受信最大 RU 長	符号なし整数 ((8 ～ 32767))
			-o	—	LOGON モード名称	1 ～ 8 文字の識別子 | space
			-j	pluname	PLU 名称	1 ～ 8 文字の識別子
			-d	—	問い合わせ応答識別	《rqd》 | rqe
			-w	cmderrstp	コマンドエラーリターン抑止種別	yes | 《no》
				firststsn	OpenTP1 の開始後，初めての STSN コマンドに対する応答の内容	positive | 《negative》
			-f	—	ホストへ送信するデータで，FMH を使用するかどうか	《yes》 | no
			-t	termself	TERM-SELF の終了方式	《orderly》 | forced
				setplu	TERM-SELF に PLU 名称を設定するかどうか	yes | 《no》
				dactlu	TERM-SELF の DACTLU 送信フィールドに設定する送信種別	on | 《off》
		mcftalcle（論理端末定義） 指定数：1 ～ 512	-l	—	論理端末名称	1 ～ 8 文字の識別子
			-t	—	論理端末タイプ	send | receive | request
			-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	出力キューの媒体の種別	《memory》 | disk
				quegrpid	出力キューグループ ID	1 ～ 8 文字の識別子

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》
			-o	aj	送信完了時の情報を取得するかどうか	《yes》｜ no
			-v	―	メッセージ受信時に起動するアプリケーション名	1 ～ 8 文字の識別子
		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）

―：該当しません。

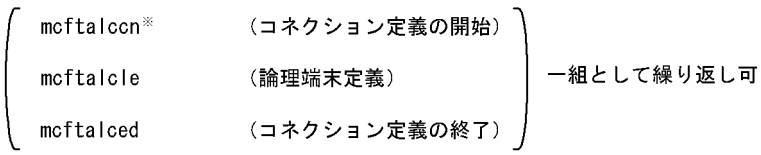
注※

SLU - TypeP2 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

SLU - TypeP2 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。MCF 通信構成定義コマンドを指定するときは，必ずこの順序に従ってください。

図 5-1　SLU - TypeP2 のプロトコル固有定義コマンドの指定順序



注※

mcftalccn と mcftalcle の指定は，1 対 1 になるようにしてください。

mcfmuap（UAP 共通定義）

形式

```
mcfmuap          :
        〔-t "〔sndrcvtim = 同期送受信監視時間〕"〕
                 :
```

機能

UAP に共通する環境を定義します。

オプション

この定義コマンドには，ほかにもオプションおよび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。

● -t

（オペランド）

sndrcvtim= **同期送受信監視時間　　～＜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。

mcftalccn（コネクション定義の開始）

形式

```
mcftalccn  -c コネクションID
           -p slup2
           -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 = xnfas"
          〔-i auto｜manual〕
          〔-b "〔bretry = yes｜no〕
                〔bretrycnt = コネクション確立再試行回数〕
                〔bretryint = コネクション確立再試行間隔〕"〕
           -k host｜ws
           -n "ownnode = 自局ノード名称"
           -q "hostnode = x'ホストノード名称'"
           -r "sndrusiz = 送信最大RU長
               rcvrusiz = 受信最大RU長"
           -o e'LOGONモード名称'｜space
           -j "pluname = e'PLU名称'"
          〔-d rqd｜rqe〕
          〔-w "〔cmderrstp = yes｜no〕
                〔firststsn = positive｜negative〕"〕
          〔-f yes｜no〕
          〔-t "〔termself = orderly｜forced〕
                〔setplu = yes｜no〕
                〔dactlu = on｜off〕"〕
```

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＞**

このコネクション ID は，ほかの mcftalccn コマンドの -c オプションで指定するコネクション ID と重複して指定できません。

● -p slup2

プロトコルの種別を指定します。

slup2
　SLUTYPE-P プロトコル（2 次局）

● -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 オペランドで指定するバッファグループ番号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 オペランドで指定するバッファグループ番号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，および出力メッセージ編集 UOC を使用する場合に，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には，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，および出力メッセージ編集 UOC 使用時に，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。
また，この count オペランドで指定するメッセージ編集用バッファ数は，mcftbuf コマンドの -g オプションの count，および extend オペランドで指定する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。

● -m

（オペランド）

mode=xnfas

使用する通信管理を指定します。

xnfas

XNF/AS を使用します。

● -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，運用コマンド（mcftactcn）を入力してコネクションを確立します。

● -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，コネクションの確立再試行をするかどうかを指定します。

yes

コネクションの確立再試行をします。

no

コネクションの確立再試行をしません。

bretrycnt= **コネクション確立再試行回数　〜＜符号なし整数＞** ((0 〜 65535))《0》**（単位：回）**

コネクション確立時に障害が発生した場合，MCF が行う確立再試行の回数を指定します。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= **コネクション確立再試行間隔　〜＜符号なし整数＞** ((0 〜 2550))《60》**（単位：秒）**

コネクション確立時に障害が発生した場合，MCF が行う確立再試行の間隔を指定します。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。

● -k host ｜ ws

起動種別を指定します。

host

ホスト（1 次局）側から SLU - TypeP2 へコネクションを確立します。SLU - TypeP2 は，ホスト（1 次局）側からのコネクション確立要求 (BIND) を待ち合わせます。

ws

SLU - TypeP2 からホスト（1 次局）側へコネクションを確立します。

● -n

（オペランド）

ownnode= **自局ノード名称　　～＜符号なし整数＞** ((0 ～ 253))

自局ノード名称を指定します。XNF/AS の構成定義で指定した LU 番号を指定してください。

詳細については，この章の「自システムの通信管理プログラムと関連づける内容」を参照してください。

● -q

（オペランド）

hostnode=x' **ホストノード名称 '　　～＜** 1 ～ 8 **けたの** 16 **進数字＞**

ホストノード名称を指定します。x は，16 進数形式で指定することを意味します。XNF/AS の構成定義で指定した PU 番号と同じ値を指定してください。

詳細については，この章の「自システムの通信管理プログラムと関連づける内容」を参照してください。

● -r

（オペランド）

sndrusiz= **送信最大** RU **長　　～＜符号なし整数＞** ((0 ～ 32767))

送信最大 RU 長を指定します。

rcvrusiz= **受信最大** RU **長　　～＜符号なし整数＞** ((8 ～ 32767))

受信最大 RU 長を指定します。

● -o

（オペランド）

e'LOGON **モード名称 '　　～＜** 1 ～ 8 **文字の識別子＞**

LOGON モード名称を指定します。e は，EBCDIC コードに変換されることを意味します。

指定した文字数が 8 文字に満たない場合，指定領域の右側はスペースで埋められます。LOGON モード名称は EBCDIC コードに変換されます。

space

LOGON モード名称に 8 文字のスペース（EBCDIC コードの $(4040404040404040)_{16}$）を設定します。

● -j

（オペランド）

pluname=e'PLU **名称** '　　～< 1 ～ 8 **文字の識別子**>

PLU 名称を指定します。e は，EBCDIC コードに変換されることを意味します。

● -d rqd ｜ rqe　　～《rqd》

ホスト（1 次局）への問い合わせ応答時の，応答識別を指定します。複数セグメントの問い合わせメッセージを送信する場合，このオプションの指定値は，最終セグメントだけ有効です。先頭セグメントおよび中間セグメントの応答識別は，このオプションの指定内容に関係なく，RQE になります。

rqd

問い合わせメッセージの応答識別に RQD を指定します。RQD を指定すると，問い合わせメッセージに対するホストからの応答（+RSP または -RSP）が送信されます。

rqe

問い合わせメッセージの応答識別に RQE を指定します。RQE を指定すると，エラーが発生した場合だけホストからの応答（-RSP）が送信されます。

● -w

（オペランド）

cmderrstp=yes ｜ no　　～《no》

SLU - TypeP2 が運用コマンド（mcftactcn，mcftactle，mcftdctcn，および mcftdctle）による状態変更ができない場合，運用コマンドを正常に受け付けたことにするかどうかを指定します。状態変更ができない場合とは，コネクションが確立済みのときに mcftactcn を入力したなどが該当します。

ただし，コネクションおよび論理端末の状態と受け付けた運用コマンドの組み合わせが以下の場合，このオペランドの指定に関係なくエラーとします。

- コネクション確立処理障害時（mcftactcn）
- コネクション使用中（mcftdctcn）
- コネクション解放処理障害（正常解放時，および強制解放時）（mcftdctcn）
- 論理端末閉塞解除処理障害（mcftactle）
- 論理端末閉塞処理障害（mcftdctle）
- 論理端末使用中（mcftdctle）

yes

入力された運用コマンドを正常に受け付けたことにします。

no

入力された運用をエラーとします。運用コマンドはエラーメッセージを出力します。

firststsn=positive ｜ negative 〜《negative》

OpenTP1 の開始後初めてのコネクションの確立において，ホストから STSN コマンドのアクションコードが "SET&TEST 要求 " の送達確認がされた場合に，ホストに送信する応答の種類を指定します。

positive

ホストに肯定応答（TEST POSITIVE）を送信します。

negative

ホストに否定応答（TEST NEGATIVE）を送信します。

● -f yes ｜ no 〜《yes》

ホストに送信するデータで，FMH を使用するかどうかを指定します。

yes

ホストへのデータ送信で FMH を使用します（RH 内の FI フィールドに ON を設定する）。
ただし，FMH データは UAP で送信メッセージ内に設定する必要があります。

no

ホストへのデータ送信で FMH を使用しません（RH 内の FI フィールドに OFF を設定する）。

ただし，このオペランドの指定値に関係なく，ホスト側から受信する BIND セションパラメタは，FMH を使用する設定（バイト 6，ビット 1=ON）になっている必要があります。FMH を使用しない設定の場合，BIND パラメタ不正として -RSP（センスコード：0x0821）が応答され，コネクションの確立が拒否されます。

● -t

（オペランド）

termself =orderly ｜ forced 〜《orderly》

TERM-SELF コマンドの終了方式フィールド（バイト 3，ビット 2）に設定する終了方式を指定します。

orderly

計画終了（'1'）を設定します。

forced

強制終了（'0'）を設定します。

setplu=yes ｜ no 〜《no》

TERM-SELF コマンドに PLU 名称を設定するかどうかを指定します。設定する PLU 名称は，-j オプションに指定した PLU 名称です。

yes

PLU 名称を設定します。

no

PLU 名称を設定しません。

なお，TERM-SELF コマンドの解放セションフィールド（バイト 3，ビット 5 ～ 6）への値の設定は，SLU - TypeP2 が setplu オペランドの指定値を基に決定します。

setplu オペランドの指定	解放セションフィールドへの設定値
yes の場合	'00'
no の場合	'10'
-t オプションの setplu は省略し，そのほかのオペランドを指定した場合	'10'
-t オプション全体を省略した場合	'00'：SLU - TypeP2 が TERM-SELF コマンドを送信する場合 '10'：通信管理が TERM-SELF コマンドを送信する場合

注 1

解放セションフィールドの設定値の意味は次のとおりです。

- '00' が設定されると，TERM-SELF コマンドの送信先が PLU に限定されます。
- '10' が設定されると，PLU，SLU の区別に関係なく TERM-SELF コマンドが送信されます。

注 2

解放セションフィールドの設定値は，-t オプション全体を省略する場合と -t オプションの各オペランドに省略時解釈値を明示指定する場合で異なります。解放セションフィールドの設定値をこのオプションが未サポートの SLU - TypeP2 旧バージョンを使用する場合と同じにするには，-t オプション全体を省略してください。

dactlu=on ｜ off　～《off》

TERM-SELF コマンドの DACTLU 送信フィールド（バイト 3，ビット 3）に設定する送信種別を指定します。

on

送信可（'1'）を設定します。

off

送信不可（'0'）を設定します。

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は，バッファグループ定義の mcftbuf コマンドに対応しています。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	バッファサイズ	バッファ面数
sndbuf	ユーザデータ長（送信最大 RU 長）	最大セグメント分割数＋ 1
rcvbuf	ユーザデータ長（受信最大 RU 長）	最大セグメント分割数＋ 2

mcftalced（コネクション定義の終了）

形式

```
mcftalced
```

機能

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 send｜receive｜request
         〔-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〕"〕
         〔-v アプリケーション名〕
```

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＞**

論理端末名称を指定します。

この論理端末名称は，ほかの mcftalcle コマンドの -l オプションで指定する論理端末名称とは異なる指定をしてください。

● -t send ｜ receive ｜ request

この論理端末の端末タイプを指定します。

send
　送信型論理端末

receive
　受信型論理端末

request
　問い合わせ型論理端末

● -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（SEND）はエラーリターンとなります。
　0 を指定した場合，メモリで待ち合わせをする出力メッセージの数は無制限になります。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（SEND）はエラーリターンとなります。
0 を指定した場合，ディスクで待ち合わせをする出力メッセージの数は無制限になります。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てください。

● -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。

● -v **アプリケーション名　〜＜** 1 〜 8 **文字の識別子＞**

入力メッセージを受信した場合に起動するアプリケーション名（MHP）を指定します。

MCF アプリケーション定義（mcfaalcap -n オプションの name オペランド）で指定した名称を指定してください。MCF アプリケーション定義については，マニュアル「OpenTP1 システム定義」を参照してください。

このオプションを省略した場合は，入力メッセージ編集 UOC で指定された値がアプリケーション名となります。

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 module = "SLU - TypeP2の実行形式プログラム名"
```

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。

オペランド

● module="SLU - TypeP2 **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。

SLU - TypeP2 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。

システムサービス共通情報定義

SLU - TypeP2 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
```

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

● 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 〜 2047))

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数を指定します。

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。↑↑は，小数点以下を切り上げることを意味します。

```
↑（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通信プロセスに対して同時に処理要求を行う運用コマンド数
  ）／0.8↑
```

注※ 1

アプリケーション起動サーバ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 MCF 通信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。

● 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〉((100 ～ 2016))

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。

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

SLU - TypeP2 の場合，MCF 通信構成定義に定義したコネクションの総数を 2 倍した値になります。実際に通信を行うコネクションの総数ではありませんので，注意してください。

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から max_open_fds オペランドの指定値を減算した超過分が，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，ご注意ください。

（「このオペランドの指定値」
＋同定義内の「max_socket_descriptorsオペランドの指定値」）≦2048

条件を満たさない指定をした場合は，このオペランドの指定値は次に示すように強制的に補正されます。

2048－（同定義内の「max_socket_descriptorsオペランドの指定値」）

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，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

```
mcfslup2 -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
```

機能

MCF 通信構成定義の SLU - TypeP2 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てはいけません。変更した場合，再開始時に正常に動作しないことがありますのでご注意ください。

SLU - TypeP2 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。

自システムの通信管理プログラムと関連づける内容

SLU - TypeP2 の定義には，LU の定義について，通信管理と関連づける項目があります。

自局ノード名称とホストノード名称

自局ノード名称とホストノード名称は，通信管理の構成定義で指定した値と一致させてください。通信管理の構成定義については，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。

ownnode オペランド（mcftalccn -n）

ownnode オペランドで指定する自局ノード名称は，HNA2 用全体定義文（HNA2_configuration max_SLUS_LU）で指定した SLUS 用の LU 番号と一致させます。

hostnode オペランド（mcftalccn -q）

hostnode オペランドで指定するホストノード名称は，HNA2 用 PU 定義文（HNA2_PU PU_number）で指定した PU 番号と一致させます。

自システムと相手システムの LU の構成に応じた定義例

自システムの通信管理プログラムのローカルアドレス割り当て機能を使用しない場合

ローカルアドレス割り当て機能を使用しない場合は，通信管理プログラムが「LU 番号＋2」をローカルアドレスとして自動的に割り当てます。

SLUS 用 LU だけで構成する例と，SLUS 用 LU 以外の LU と混在する例をそれぞれ示します。

図 5-2　SLUS 用 LU だけの構成例（ローカルアドレス割り当て機能を使用しない場合）

定義する LU	LU 数	使用できる LU の番号	1 次局のローカルアドレス
SLUS 用 LU	10	0～9	2～11

自システム(HNA2次局)の通信管理プログラムの定義文

```
HNA2_configuration
    default_slot_no     1
    max_SLUS_LU         10
    max_SLU_count       10
;

HNA2_PU
    PU_number           0
    destination_name    mcf02_vc
;
```

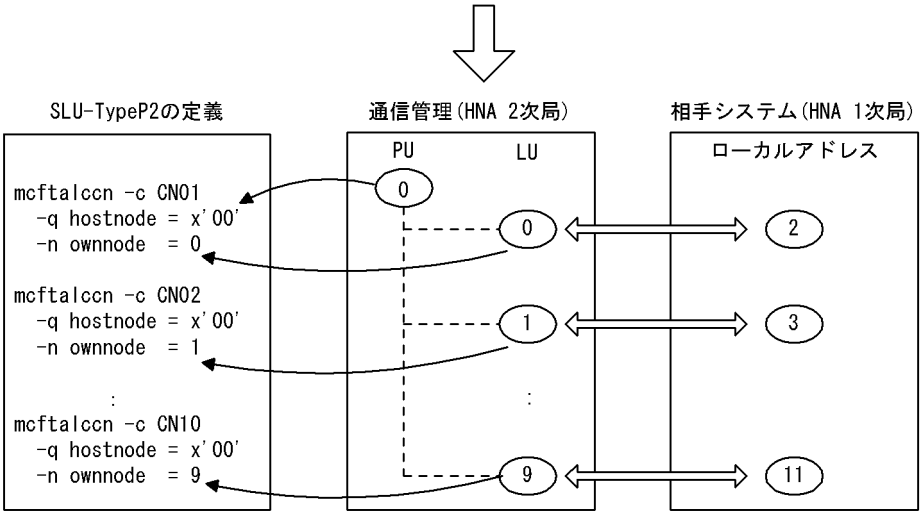
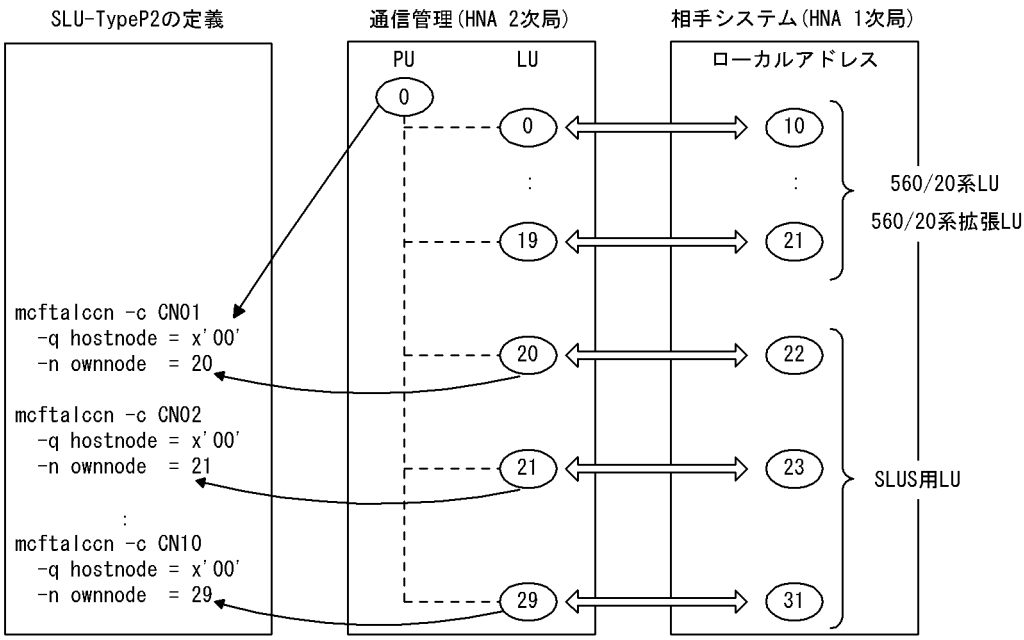


図 5-3 SLUS 用 LU 以外の LU と混在する構成例（ローカルアドレス割り当て機能を使用しない場合）

定義する LU	LU 数	使用できる LU の番号	1 次局のローカルアドレス
560/20 系 LU	10	0～9	2～11
560/20 系拡張 LU	10	10～19	12～21
SLUS 用 LU	10	20～29	22～31

自システム(HNA2次局)の通信管理プログラムの定義文

```
HNA2_configuration
   default_slot_no      1
   max_560_LU           10
   max_extend_LU        10
   max_SLUS_LU          10
   max_SLU_count        10
;

HNA2_PU
   PU_number            0
   destination_name     mcf02_vc
;
```



自システムの通信管理プログラムのローカルアドレス割り当て機能を使用する場合

ローカルアドレス割り当て機能を使用する場合に，SLUS 用 LU だけで構成する例と，SLUS 用 LU 以外の LU と混在する例をそれぞれ示します。

図 5-4　SLUS 用 LU だけの構成例（ローカルアドレス割り当て機能を使用する場合）

定義する LU	LU 数	使用できる LU の番号	1 次局のローカルアドレス
SLUS 用 LU	10	0～9	1～10

自システム(HNA2次局)の通信管理プログラムの定義文

```
HNA2_configuration
    default_slot_no     1
    max_SLUS_LU         10
    max_SLU_count       10
;
HNA2_PU
    PU_number           0
    destination_name
    mcf02_vc
    assign_LA           yes
    first_SLUS_LA       1
;
```

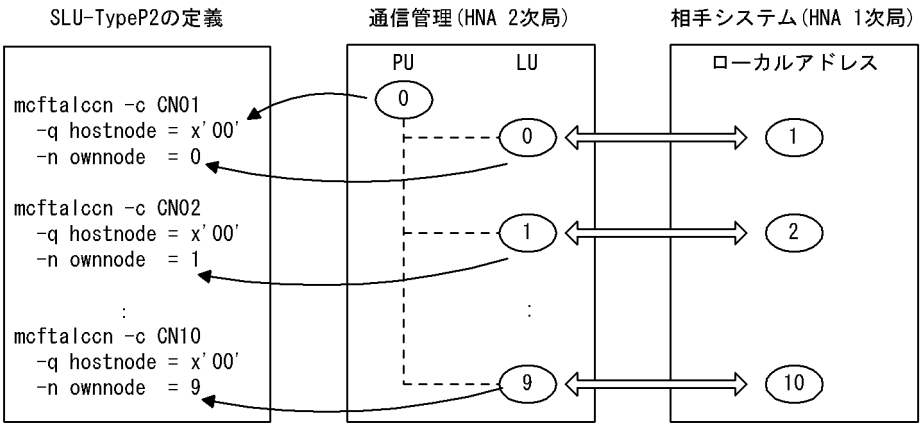
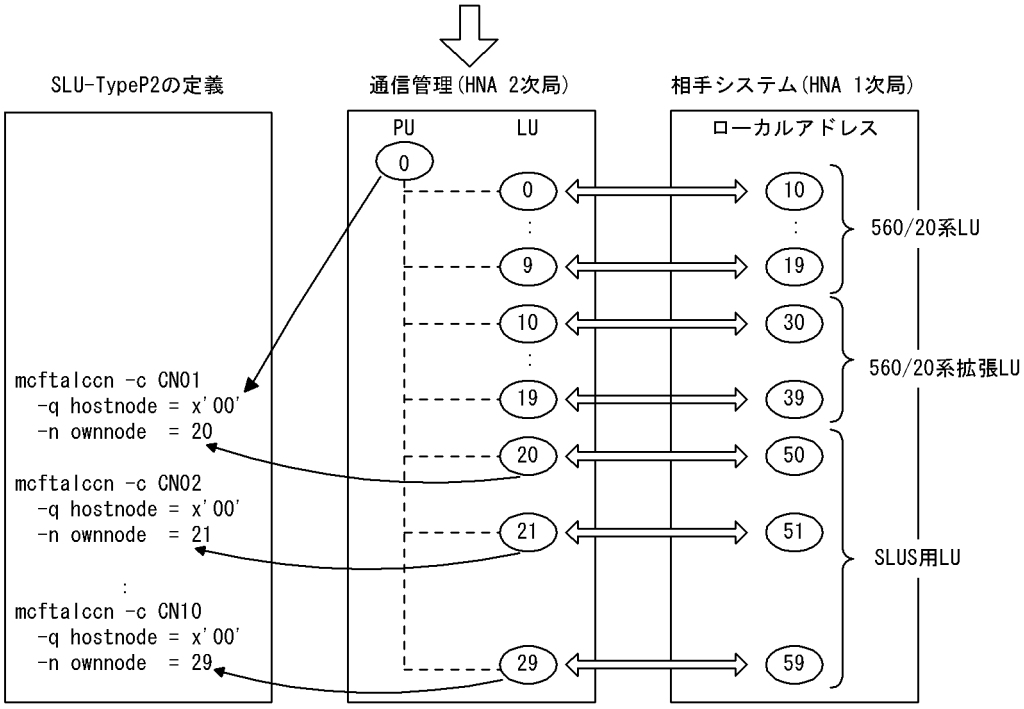


図 5-5 SLUS 用 LU 以外の LU と混在する構成例（ローカルアドレス割り当て機能を使用する場合）

定義する LU	LU 数	使用できる LU の番号	1 次局のローカルアドレス
560/20 系 LU	10	0～9	10～19
560/20 系拡張 LU	10	10～19	30～39
SLUS 用 LU	10	20～29	50～59

自システム(HNA2次局)の通信管理プログラムの定義文

```
HNA2_configuration
    default_slot_no     1
    max_560_LU          10
    max_extend_LU       10
    max_SLUS_LU         10
    max_SLU_count       10
;
HNA2_PU
    PU_number           0
    destination_name
    mcf02_vc
    assign_LA           yes
    first_560_LA        10
    first_extend_LA     30
    first_SLUS_LA       50
;
```



通信管理プログラムの自動ログオン機能

SLU - TypeP2 は，NOTIFY コマンドが未サポートのため，HNA2 全体定義文

（HNA2_configuration），HNA2 用 PU 定義文（HNA2_PU）および HNA2 用 LU 定義文（HNA2_LU）のどれにも，自動ログオン定義（auto_logon）に yes を指定できません。自動ログオン定義を省略するか，no を指定する必要があります。

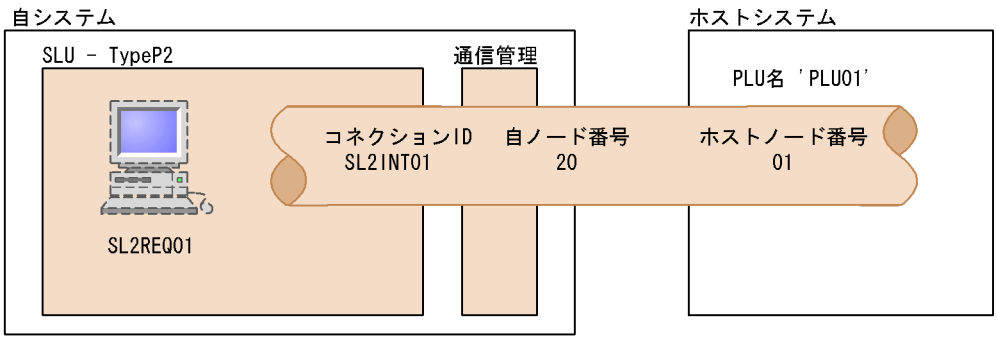
定義例

SLU - TypeP2 のシステム構成例を次の図に，その構成に沿った定義例をそのあとに示します。

SLU - TypeP2 では，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。

- /BeTRAN/examples/mcf/SLUP2/conf/com_d

図 5-6　SLU - TypeP2 のシステム構成例




```
###################################################################
#          MCF COMMUNICATION CONFIGURATION DEFINITION             #
#                  for SLU - TypeP2 protocol                      #
###################################################################
#--------------Connection definition (mcftalccn)------------#
mcftalccn    -c    SL2INT01                  ¥
             -p    slup2                     ¥
             -n   "ownnode    = 20"          ¥
             -q   "hostnode   = x'01'"       ¥
             -o  e'LOGONMOD'                 ¥
             -j   "pluname    = e'PLU01'"    ¥
             -g   "sndbuf     = 1            ¥
                   rcvbuf     = 2"           ¥
             -e   "msgbuf     = 3            ¥
                   count      = 10"          ¥
             -m   "mode       = xnfas"       ¥
             -i    auto                      ¥
             -b   "bretry     = yes          ¥
                   bretrycnt  = 10           ¥
                   bretryint  = 2"           ¥
             -k    ws                        ¥
             -d    rqe                       ¥
             -w   "firststsn = positive"     ¥
             -t   "termself = forced"        ¥
             -r   "sndrusiz   = 1024         ¥
                   rcvrusiz   = 1024"
#-------------------------------------------------------------#
```

```
########LE definition  (SL2REQ01)
mcftalcle   -l    SL2REQ01                ¥
            -t    request                 ¥
            -m   "mmsgcnt    = 0          ¥
                  dmsgcnt    = 0"         ¥
            -k   "quekind    = disk       ¥
                  quegrpid   = quegrp01"  ¥
            -o   "aj         = yes"       ¥
            -v    slmhprv1
#-------------------------------------------------------------#
###Connection definition end  (SL2INT01)
mcftalced
```

6 運用コマンド

OpenTP1 システムは，運用コマンドを使用してシステムを運用できます。この章では，SLU - TypeP2 で使用する運用コマンドについて説明します。

SLU - TypeP2 **の運用コマンド**
mcftactcn**（コネクションの確立）**
mcftactle**（論理端末の閉塞解除）**
mcftdctcn**（コネクションの解放）**
mcftdctle**（論理端末の閉塞）**
mcftlscn**（コネクションの状態表示）**
mcftlsle**（論理端末の状態表示）**

SLU - TypeP2 の運用コマンド

SLU - TypeP2 は，オンライン中に運用コマンドを入力できます。SLU - TypeP2 で使用する運用コマンドを次の表に示し，それぞれについて説明します。

なお，ここでは，SLU - TypeP2 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。

表 6-1　SLU - TypeP2 で使用する運用コマンドの一覧

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　　～＜** 16 **進数字＞** ((01 ～ ef))

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 通信プロセスに対して mcftactcn コマンドを実行します。

● -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名称を指定します。

コネクション ID は一度に 8 個まで指定できます。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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

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。	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。	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
KFCA10500-I	ヘルプメッセージ	標準出力
KFCA15330-E	MCF 運用コマンド処理中に異常が発生しました。	標準エラー出力
KFCA15332-E	コネクションが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
KFCA15333-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
KFCA15334-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
KFCA15342-E	ホスト起動方式のコネクション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

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　　～＜ 16 進数字＞ ((01 ～ ef))

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 通信プロセスに対して mcftactle コマンドを実行します。

● -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てください。

論理端末名称は一度に 8 個まで指定できます。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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。	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。	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
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
KFCA10503-I	ヘルプメッセージ	標準出力
KFCA15330-E	MCF 運用コマンド処理中に異常が発生しました。	標準エラー出力
KFCA15337-E	論理端末が活性化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
KFCA15339-E	論理端末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
KFCA15396-E	内部処理実行中に障害が発生しました。論理端末を閉塞します。	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

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。

オプション

● -s MCF **通信プロセス識別子　　～＜ 16 進数字＞** ((01 ～ ef))

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 通信プロセスに対して mcftdctcn コマンドを実行します。

● -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。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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した場合，該当するコネクションが仕掛り中のとき，仕掛り中の処理を終了しないで強制的に解放します。

このオプションの指定を省略した場合，該当するコネクションが仕掛り中の場合，

mcftdctcn コマンドはエラーとなります。

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。	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。	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
KFCA10501-I	ヘルプメッセージ	標準出力
KFCA15330-E	MCF 運用コマンド処理中に異常が発生しました。	標準エラー出力
KFCA15331-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
KFCA15333-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
KFCA15334-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
KFCA15341-E	コネクション使用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

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。

オプション

● -s MCF 通信プロセス識別子　　～＜ 16 進数字＞ ((01 ～ ef))

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 通信プロセスに対して mcftdctle コマンドを実行します。

● -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てください。

論理端末名称は一度に 8 個まで指定できます。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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。	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。	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
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
KFCA10395-E	指定したコネクションには未接続の論理端末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
KFCA15330-E	MCF 運用コマンド処理中に異常が発生しました。	標準エラー出力
KFCA15335-E	論理端末が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
KFCA15338-E	論理端末使用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
KFCA15339-E	論理端末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）を入力した場合は，論理端末が閉塞状態でも，メッセージを受信します。論理端末閉塞によるメッセージ受信処理への影響はありません。

● 送信仕掛り中に運用コマンド（mcftdctle）を入力した場合は，運用コマンドがエラーリターンします。送信仕掛り中でない場合は，論理端末は閉塞されます。

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表示します。

オプション

● -s MCF **通信プロセス識別子　　～＜ 16 進数字＞** ((01 ～ ef))

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 通信プロセスに対して mcftlscn コマンドを実行します。

● -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。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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
  SL2…SLUTYPE-P プロトコル
- sssss：コネクション状態
  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：論理端末の端末タイプ
  SND…send 型
  RCV…receive 型
  REQ…request 型

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
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。	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。	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

出力メッセージ ID	内容	出力先
KFCA10369-I	状態表示を終了します。	標準出力
KFCA10371-I	mcftlscn コマンドを正常に受け付けました。	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
KFCA10502-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

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〕〔-t〕
```

機能

論理端末の状態を標準出力に表示します。

オプション

● -s MCF **通信プロセス識別子　　～＜** 16 **進数字＞** ((01 ～ ef))

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 通信プロセスに対して mcftlsle コマンドを実行します。

● -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てください。

論理端末名称は一度に 8 個まで指定できます。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。一括指定は一つだけ指定できます。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。一括指定をするときは，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。

● -t

指定した論理端末がセキュア状態かどうかを表示します。

出力形式

```
mmm llllllll sss〔ggg〕〔tttt〕
    SYNC xxxxxxxxxx yyyyyyyyyy zzzzzzzzzz
    IO    :     :     :
    PRIO  :     :     :
    NORM  :     :     :
    iii ooo
```

- mmm：MCF 識別子
- llllllll：論理端末名称
- sss：論理端末状態
  ACT…閉塞解除状態
  DCT…閉塞状態
- ggg：論理端末のセキュア状態（-t オプション指定時だけ表示）
  NOS…非セキュア状態
  SEC…セキュア状態
- tttt：論理端末のテストモード状態（TP1/Message Control/Tester 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：未送信メッセージ最小通番
- zzzzzzzzzz：未送信メッセージ最大通番
- iii：出力キューの入力の保留状態（-q オプション指定時だけ表示）
  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。	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。	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
KFCA10371-I	mcftlsle コマンドを正常に受け付けました。	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
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
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

7 組み込み方法

この章では，SLU - TypeP2 を OpenTP1 システムに組み込む方法について説明します。

7.1 SLU - TypeP2 の組み込みの流れ

SLU - TypeP2 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

SLU - TypeP2 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，「7.2　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

SLU - TypeP2 を実行するためには，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。

(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，5 章の「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，「7.3　定義オブジェクトファイルの生成」を参照してください。

7.2 MCF メイン関数の作成

SLU - TypeP2 は，OpenTP1 プロセスサービスによって起動されます。

SLU - TypeP2 を起動するためには，ユーザが MCF メイン関数をコーディングし，コンパイル，およびリンケージを行って SLU - TypeP2 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，mcfplslup2 コマンドを使用します。

MCF メイン関数からは，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。

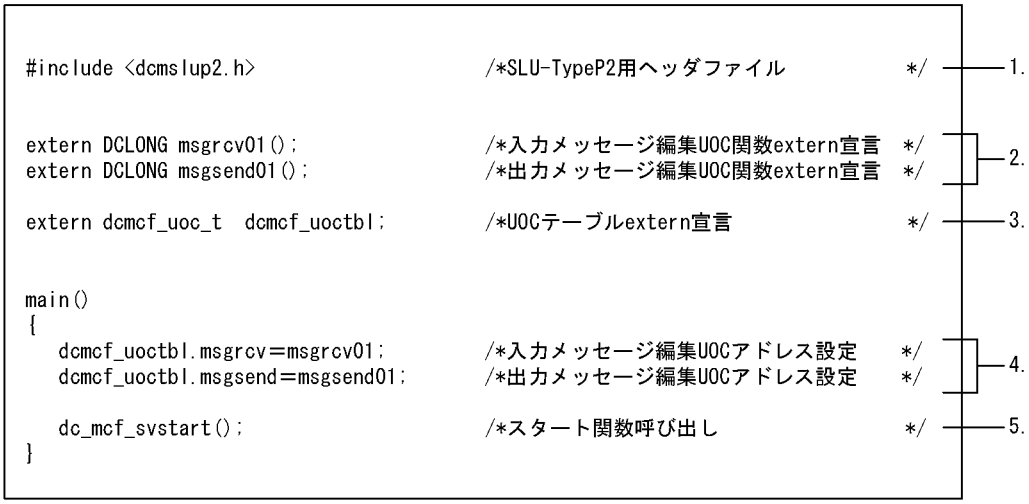
MCF メイン関数のコーディング概要を図 7-1 と図 7-2 に示します。また，ディレクトリへの組み込み方法を図 7-3 に示します。なお，これらのコーディング例を次のファイルで提供しています。

- /BeTRAN/examples/mcf/SLUP2/cmlib/ansi/com.c
- /BeTRAN/examples/mcf/SLUP2/cmlib/c/com.c

図 7-1 MCF メイン関数のコーディング概要（ANSI C，C++ の場合）

```
#include <dcmslup2.h>                          /*SLU-TypeP2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();                            /*スタート関数呼び出し               */ ―― 5.
  return 0;
}
```

図 7-2 MCF メイン関数のコーディング概要（K&R 版 C の場合）

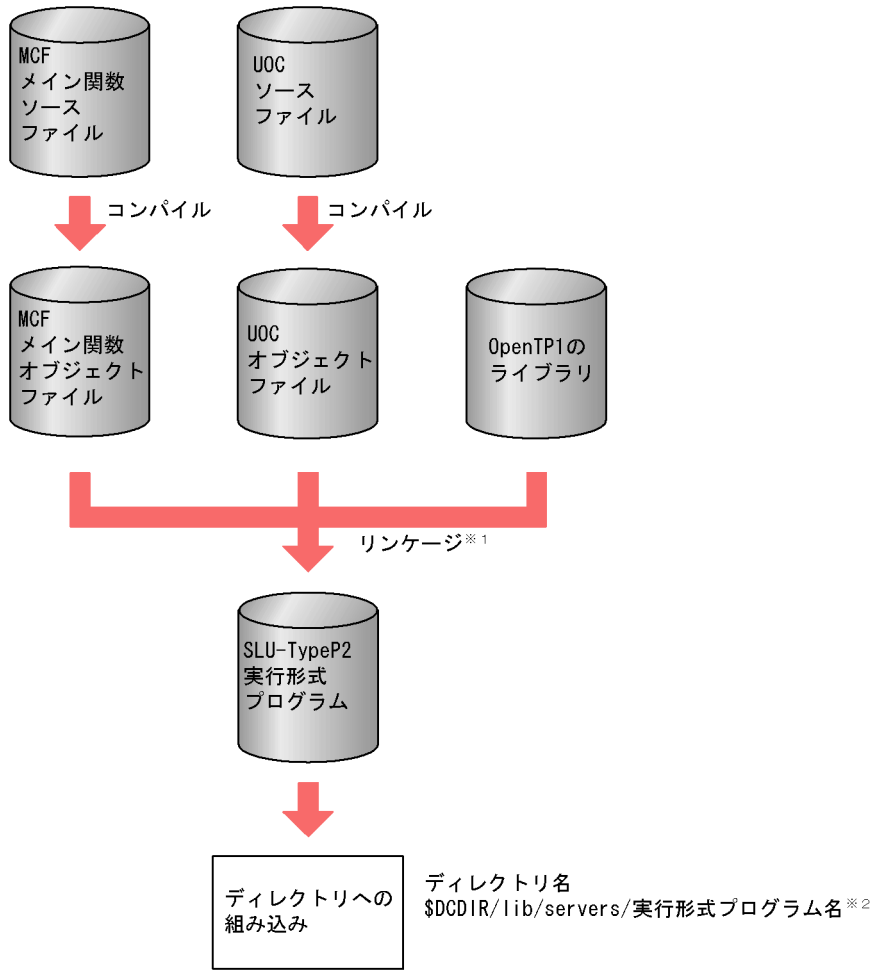


1. SLU - TypeP2 で提供するヘッダファイルを取り込みます。
2. 使用する UOC 関数を extern 宣言します。UOC のリターン値は DCLONG 型にしてください。
   UOC をまったく使用しない場合，このコーディングは必要ありません。
3. UOC テーブルを extern 宣言します。UOC を使用する場合，必ずこのとおりにコーディングしてください。
   UOC をまったく使用しない場合，このコーディングは必要ありません。
4. 各 UOC 関数のアドレスを，次に示すシステム提供変数に設定します。使用する UOC だけコーディングしてください。

   ```
   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。

図 7-3　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

mcfplslup2 コマンドでリンケージします。

注※ 2

SLU - TypeP2 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。

7.3 定義オブジェクトファイルの生成

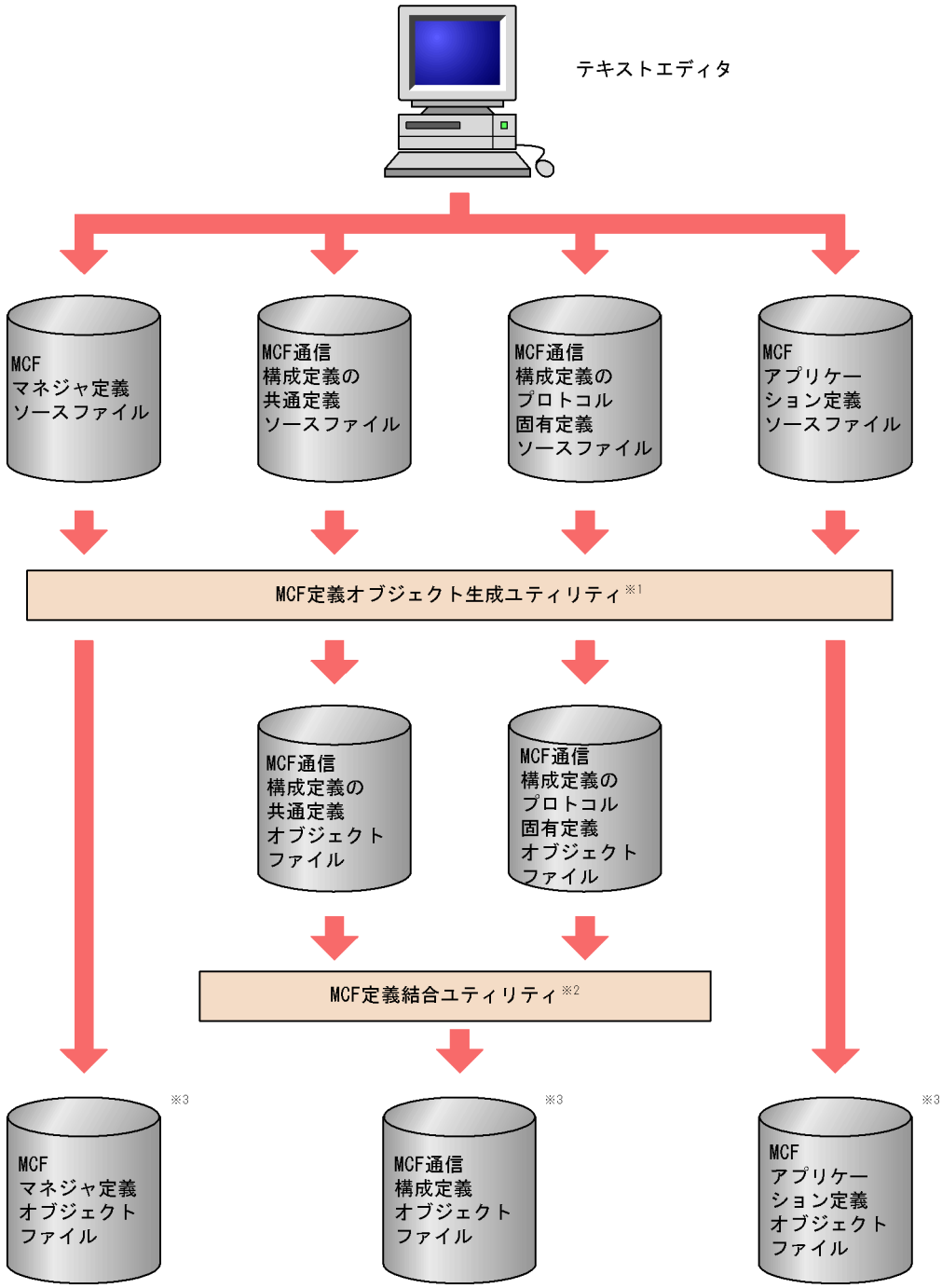
定義オブジェクトファイルを次の手順で生成します。

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てはいけません。変更した場合，再開始ができなくなることがあります。

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 SLU - TypeP2 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 SLU - TypeP2 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ます。**

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 7-4　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
- mcfslup2：MCF 通信構成定義のプロトコル（SLU - TypeP2）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 mcfslup2 コマンドについては 5 章の「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
        △SLU - TypeP2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。

8 障害対策

この章では，SLU - TypeP2 運用中に発生する障害と，SLU - TypeP2 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。

8.1 障害の種類と対応処理

運用中に障害が発生すると，SLU - TypeP2 はシステムを回復します。このとき，システム定義を指定すると，MCF イベント処理用 MHP を起動することもできます。

8.1.1 SLU - TypeP2 運用中の障害と対応処理

SLU - TypeP2 運用中の障害と対応処理について，障害の種類ごとに示します。

また，センスコードの詳細は，「付録 C.3　SLU-TypeP2 が使用するセンスコード」を参照してください。

(1) コネクション障害

コネクション障害の発生個所に応じた SLU - TypeP2 の障害処理については，「8.2　コネクション障害」を参照してください。

表 8-1　コネクション障害発生時の処理

障害の内容	SLU - TypeP2 の処理	ユーザの処置
通信管理からの H2_ABORT 受信など	1. 該当する論理端末ごとの障害処理をします（「8.1.2　論理端末ごとの障害処理」参照）。 2. コネクション障害を通知するメッセージログ（KFCA15121-E）を出力します。	端末起動の場合は運用コマンド（mcftactcn）を入力して，再び確立処理をします。ホスト起動の場合はホスト側から確立要求をするようにしてください。
通信管理マクロエラーリターン	1. 該当する論理端末ごとの障害処理をします（「8.1.2　論理端末ごとの障害処理」参照）。 2. コネクション障害を通知するメッセージログ（KFCA15120-E）を出力します。	
セションパラメタ不正の BIND コマンドを受信	1. -RSP を送信します。センスコードを次に示します。 セションパラメタ不正：08210000 2. -RSP 送信したことを通知するメッセージログ（KFCA15204-E）を出力します。 3. CERREVT（コネクション障害，下位層障害）を起動します。 4. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 5. コネクション確立処理を中断します。	SLU-TypeP2 の定義誤りの場合，誤りを訂正し，定義オブジェクトを作り直して OpenTP1 を再開始します。

(2) 論理端末障害

論理端末の端末タイプごとに処理が異なります。

(a) request 型

表 8-2 request 型論理端末の障害発生時の処理

障害の内容	SLU - TypeP2 の処理	ユーザの処置
メッセージ送信完了エラー，拒否応答（-RSP）受信	1. UAP にエラーリターンします。リターン値 DCMCFRTN_73001（同期）を返します。 2. CERREVT（論理端末障害，送信中断）を起動します。 3. 出力キューを削除します（同期）。 4. 再送準備を要求します（非同期）。 5. 論理端末を閉塞します。 6. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 7. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。	障害要因を取り除いたあと，運用コマンド（mcftactle）を入力して，論理端末を閉塞解除します。
出力キュー読み込み障害，出力メッセージ編集 UOC エラーリターン，送信バッファサイズ不足	1. UAP にエラーリターンします。リターン値 DCMCFRTN_73001（同期）を返します。 2. CERREVT（論理端末障害）を起動します。 3. 出力キューを削除します（同期）。 4. 再送準備を要求します（非同期）。 5. 論理端末を閉塞します。 6. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 7. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 8. 内部処理障害を通知するメッセージログ（KFCA15396-E）を出力します。	• 出力キュー読み込み障害 障害要因を取り除いてください。 • 出力メッセージ編集 UOC エラーリターン 出力メッセージ編集 UOC 処理を見直してください。 • 送信バッファサイズ不足 システム定義を修正してください。
送信メッセージ削除失敗	処理を続行します。	ありません。
タイムアウト	1. UAP にエラーリターンします。リターン値 DCMCFRTN_73005（同期）を返します。 2. 出力キューを削除します（同期）。 3. 論理端末を閉塞します。 4. コネクションを切断します。 5. CERREVT（コネクション障害，タイムアウト）を起動します。 6. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 7. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 8. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。	端末起動の場合は運用コマンド（mcftactcn）を入力して，再び確立処理をします。ホスト起動の場合はホスト側から確立要求をするようにしてください。

障害の内容	SLU - TypeP2 の処理	ユーザの処置
不正応答メッセージ受信（ブラケット違反，チェイン違反，RU 長不正）	1. UAP にエラーリターンします。 リターン値 DCMCFRTN_73001（同期）を返します。 2. 出力キューを削除します（同期）。 3. -RSP を送信します。 センスコードを次に示します。 ・ブラケット違反：20030000 ・チェイン違反：20020000 ・RU 長不正：10020000 4. -RSP 送信したことを通知するメッセージログ（KFCA15204-E）を出力します。 5. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。	相手システムへ不正応答メッセージを受信したことを連絡してください。
入力キュー書き込み障害，アプリケーションスケジュール失敗，入力メッセージ編集 UOC エラーリターン	1. CERREVT（論理端末障害，入力キュー障害）を起動します。 2. 論理端末を閉塞します。 3. -RSP を送信します。 センスコードを次に示します。 ・要求不実行：08000000 4. -RSP 送信したことを通知するメッセージログ（KFCA15204-E）を出力します。 5. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 6. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 7. 内部処理障害を通知するメッセージログ（KFCA15396-E）を出力します。	運用コマンド（mcftactle）を入力して，論理端末を閉塞解除します。
送信仕掛り中の mcftdctle 入力	1. コマンドをエラーリターンします。 2. 論理端末の使用中を通知するメッセージログ（KFCA15338-E）を出力します。	ありません。
応答ダミーデータ受信	1. UAP にエラーリターンします。 リターン値 DCMCFRTN_73005 を返します。	ありません。
メッセージコンテンション（ブラケット開始拒否）	1. -RSP を送信します。 センスコードを次に示します。 ・ブラケット開始拒否：08130000，08140000 2. -RSP 送信したことを通知するメッセージログ（KFCA15204-E）を出力します。	ありません。

(b) send 型

表 8-3 send 型論理端末の障害発生時の処理

障害の内容	SLU - TypeP2 の処理	ユーザの処置
メッセージ送信完了エラー，拒否応答（-RSP）受信	1. -RSP 受信したことを通知するメッセージログ（KFCA15205-E）を出力します。 2. CERREVT（論理端末障害，送信中断）を起動します。 3. 再送準備を要求します（非同期）。 4. 論理端末を閉塞します。 5. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 6. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。	障害要因を取り除いたあと，運用コマンド（mcftactle）を入力して，論理端末を閉塞解除します。
出力キュー読み込み障害，出力メッセージ編集 UOC エラーリターン，送信バッファサイズ不足	1. CERREVT（論理端末障害）を起動します。 2. 論理端末を閉塞します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 5. 内部処理障害を通知するメッセージログ（KFCA15396-E）を出力します。 6. 再送準備を要求します。	• 出力キュー読み込み障害 障害要因を取り除いてください。 • 出力メッセージ編集 UOC エラーリターン 出力メッセージ編集 UOC 処理を見直してください。 • 送信バッファサイズ不足 システム定義を修正してください。
送信メッセージ削除失敗	処理を続行します。	ありません。
送信仕掛り中の mcftdctle 入力	1. コマンドをエラーリターンします。 2. 論理端末の使用中を通知するメッセージログ（KFCA15338-E）を出力します。	ありません。
メッセージコンテンション（ブラケット開始拒否）	1. -RSP を送信します。 センスコードを次に示します。 ・ブラケット開始拒否：08130000，08140000 2. -RSP 送信したことを通知するメッセージログ（KFCA15204-E）を出力します。	ありません。

(c) receive 型

表 8-4 receive 型論理端末の障害発生時の処理

障害の内容	SLU - TypeP2 の処理	ユーザの処置
入力キュー書き込み障害，アプリケーションスケジュール失敗，入力メッセージ編集 UOC エラーリターン	1. CERREVT（論理端末障害）を起動します。 2. -RSP を送信します。 センスコードを次に示します。 ・要求不実行：08000000 3. -RSP 受信したことを通知するメッセージログ（KFCA15205-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 6. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 7. 内部処理障害を通知するメッセージログ（KFCA15396-E）を出力します。 8. 再送準備を要求します。	障害要因を取り除いたあと，運用コマンド（mcftactle）を入力して，論理端末を閉塞解除します。

(3) その他の障害

表 8-5 論理端末の型に共通する，その他の障害発生時の処理

障害の種類	SLU - TypeP2 の処理	ユーザの処置
受信メッセージと論理端末の端末タイプの不一致※1	1. 受信メッセージを破棄します。 2. -RSP を送信します。 センスコードを次に示します。 ・要求不実行：08000000 3. -RSP 受信したことを通知するメッセージログ（KFCA15205-E）を出力します。 4. CERREVT（論理端末障害）を起動します。 5. 論理端末を閉塞します。 6. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。	ホスト側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※2	UAP に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。
送信メッセージ消去失敗	処理を続行します。	ありません。
送信バッファ面数不足	1. メモリダンプを出力します。 2. コネクションを切断します。 3. CERREVT（コネクション障害）を起動します。 4. コネクション強制解放を通知するメッセージログ（KFCA15398-E）を出力します。 5. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。	バッファ面数を増やして OpenTP1 を再開始します。

障害の種類	SLU - TypeP2 の処理	ユーザの処置
受信バッファ面数不足	1. メモリダンプを出力します。 2. コネクションを切断します。 3. CERREVT（コネクション障害）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。	バッファ面数を増やしてOpenTP1 を再開始します。
同期送受信（sendrecv）中にUAP 終了	1. 論理端末障害を通知する通知するメッセージログ（KFCA15303-E）を出力します。 2. 論理端末を閉塞するメッセージログ（KFCA15396-E）または処理を続行するメッセージログ（KFCA15397-E）を出力します。 3. メモリダンプを取得します。（論理端末を閉塞する場合） 4. CERREVT（論理端末障害）を起動します。 5. 論理端末を閉塞します。	UAP を見直し，障害要因を取り除いたあと，論理端末が閉塞されている場合は，運用コマンド（mcftactle）を入力して，論理端末を閉塞解除します。
内部矛盾	1. メモリダンプを出力します。 2. 内部処理実行中異常を通知するメッセージログ（KFCA15399-E）を出力します。 3. プロセスを異常終了します。	OpenTP1 を再開始します。

注※ 1

例えば，send 型の論理端末にメッセージ受信した場合に発生します。

注※ 2

例えば，次の場合に発生します。

- receive 型の論理端末に対して送信要求（send，sendrecv）を実行した場合
- send 型の論理端末に対して同期送受信関数（sendrecv）を実行した場合

8.1.2 論理端末ごとの障害処理

SLU - TypeP2 運用中の障害と対応処理について，論理端末ごとに示します。

（1） request 型

表 8-6 request 型論理端末の障害処理

障害の内容		SLU - TypeP2 の処理
下位切断	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，下位層障害）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，下位層障害）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。
	同期送受信仕掛り中	1. 同期送受信タイマを解除します。 2. UAP をエラーリターンします。 リターン値 DCMCFRTN_73001（同期）を返します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，下位層障害）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。
	上記以外の場合	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，下位層障害）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。
mcftdctcn -f 入力による強制解放	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，強制解放）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，強制解放）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。

障害の内容		SLU - TypeP2 の処理
	同期送受信仕掛り中	1. 同期送受信タイマを解除します。 2. UAP をエラーリターンします。 リターン値 DCMCFRTN_73001（同期）を返します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，強制解放）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302 -E）を出力します。 8. コネクションを切断します。
	上記以外の場合	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，強制解放）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。
LU-LU セション解放	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，送信中断）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，送信中断）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。
	同期送受信仕掛り中	1. 同期送受信タイマを解除します。 2. UAP をエラーリターンします。 リターン値 DCMCFRTN_73001（同期）を返します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，送信中断）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。

（2） send 型

表 8-7 send 型論理端末の障害処理

障害の内容		SLU - TypeP2 の処理
下位切断	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，下位層障害）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，下位層障害）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。
	上記以外の場合	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，下位層障害）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。
mcftdctcn -f 入力による強制解放	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，強制解放）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，強制解放）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。

障害の内容		SLU - TypeP2 の処理
	上記以外の場合	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，強制解放）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。
LU-LU セション解放	非同期送信仕掛り中	1. 再送準備を要求します。 2. CERREVT（論理端末障害，送信中断）を起動します。 3. 論理端末障害を通知するメッセージログ（KFCA15303-E）を出力します。 4. 論理端末を閉塞します。 5. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 6. CERREVT（コネクション障害，送信中断）を起動します。 7. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 8. コネクションを切断します。

（3） recieve 型

表 8-8　receive 型論理端末の障害処理

障害の内容	SLU - TypeP2 の処理
下位切断	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，下位層障害）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。
mcftdctcn -f 入力による強制解放	1. 論理端末を閉塞します。 2. 論理端末閉塞を通知するメッセージログ（KFCA15312-I）を出力します。 3. CERREVT（コネクション障害，強制解放）を起動します。 4. コネクション障害を通知するメッセージログ（KFCA15302-E）を出力します。 5. コネクションを切断します。

8.2 コネクション障害

コネクションが切断された場合，端末起動型のシステムでは，運用コマンド（mcftactcn）を入力してコネクションを再確立します。ホスト起動型のシステムでは，自動的に相手システムからのコネクション確立を待ちます。自システムで管理している通信機器に障害が発生した場合も，同様にコネクション障害として処理します。

メッセージ送受信時にコネクション障害が発生した場合，メッセージは破棄されます。

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，障害発生個所による SLU - TypeP2 の障害処理について表 8-9 に示します。

図 8-1　コネクション障害

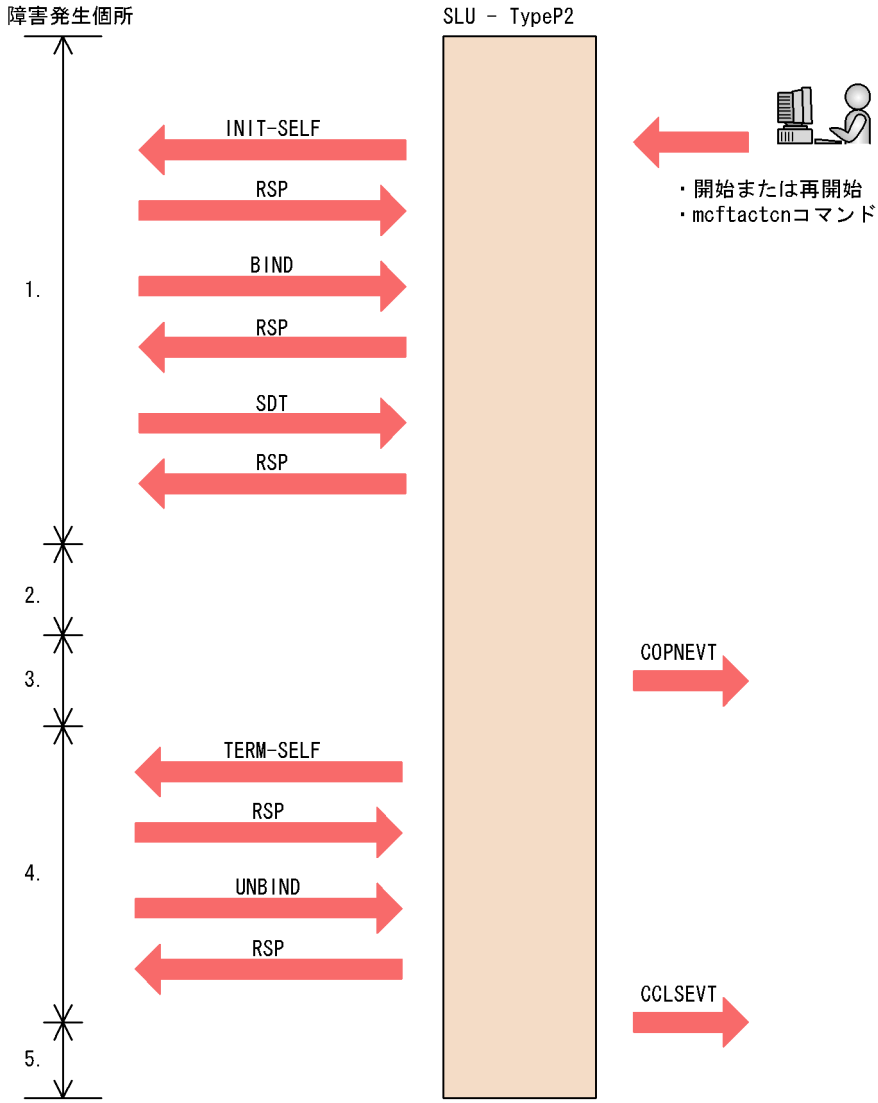


表 8-9　コネクション障害の処理

障害発生個所	内容	メッセージログの出力内容	SLU - TypeP2 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）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。

障害発生個所	内容	メッセージログの出力内容	SLU - TypeP2 の処理
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	• CERREVT を起動します。 • コネクションを閉塞します。 閉塞後は運用コマンド（mcftactcn）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。 • コネクションを閉塞します。 閉塞後は運用コマンド（mcftactcn）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。

8.3　論理端末障害

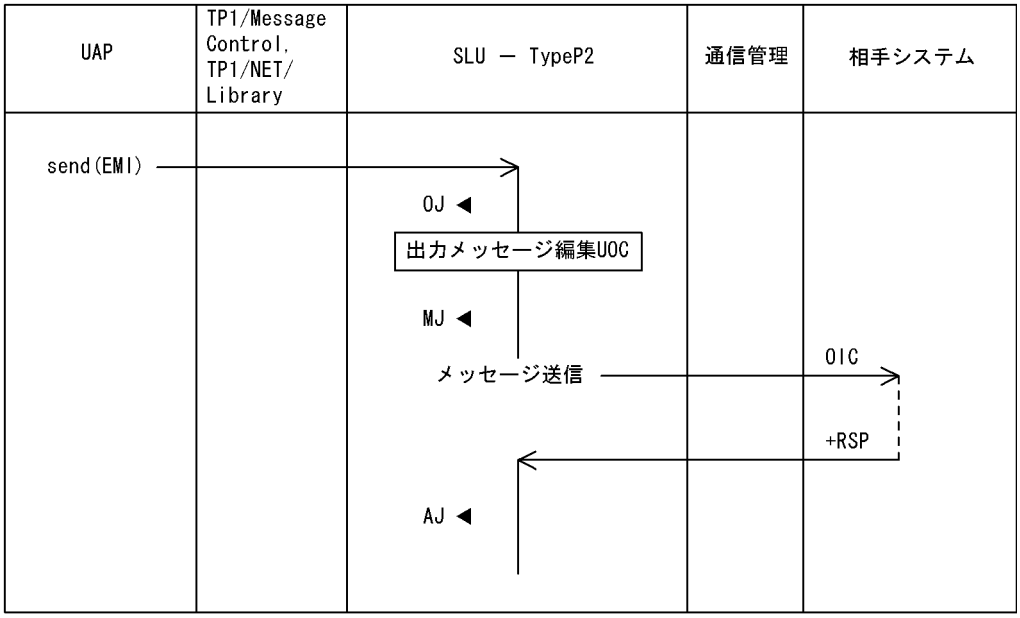
論理端末障害が発生したときの処理は，コネクション障害の場合の処理と同じです。処理の詳細については，「8.1　障害の種類と対応処理」および「8.2　コネクション障害」を参照してください。

付録

付録 A　メッセージ送受信の処理の流れ

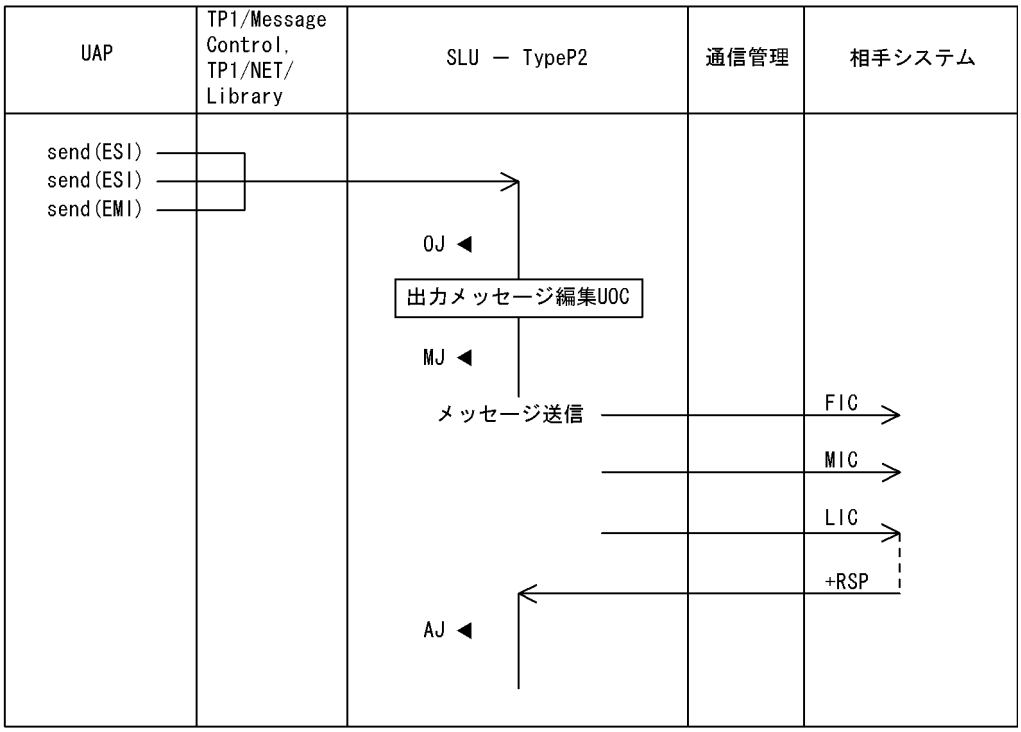
メッセージを送受信するときの処理の流れを図 A-1 ～図 A-6 に示します。

図 A-1　一方送信メッセージ（send 型／単独セグメントの場合）



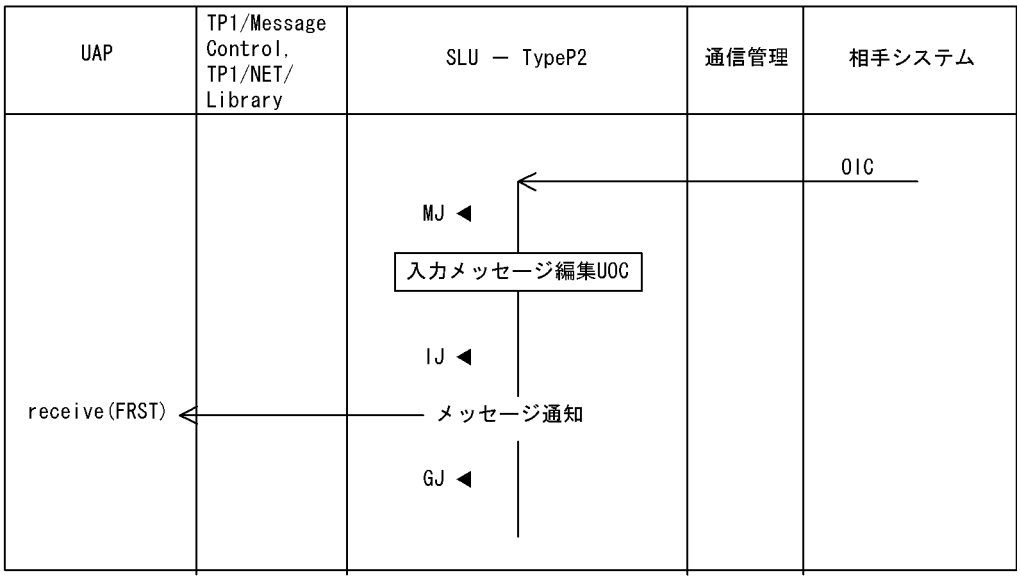
（凡例）
AJ ◀：メッセージ送信完了ジャーナル取得
MJ ◀：メッセージジャーナル取得
OJ ◀：メッセージ出力ジャーナル取得

図 A-2　一方送信メッセージ（send 型／複数セグメントの場合）



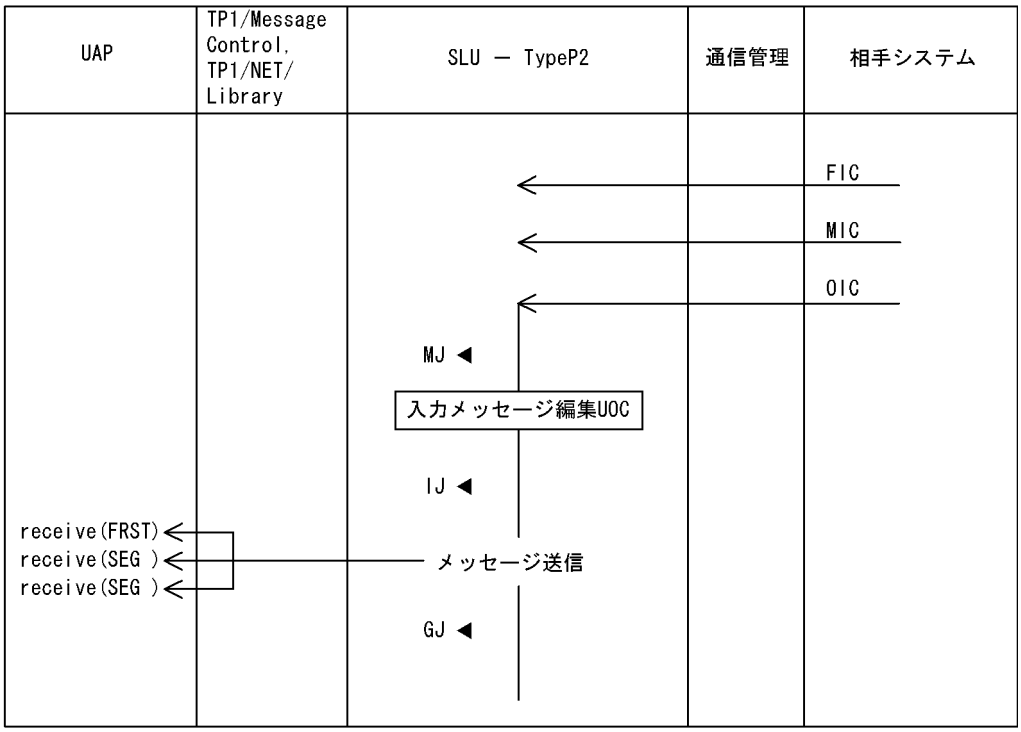
（凡例）
AJ ◀：メッセージ送信完了ジャーナル取得
MJ ◀：メッセージジャーナル取得
OJ ◀：メッセージ出力ジャーナル取得

図 A-3　一方送信メッセージ（receive 型／単独セグメントの場合）



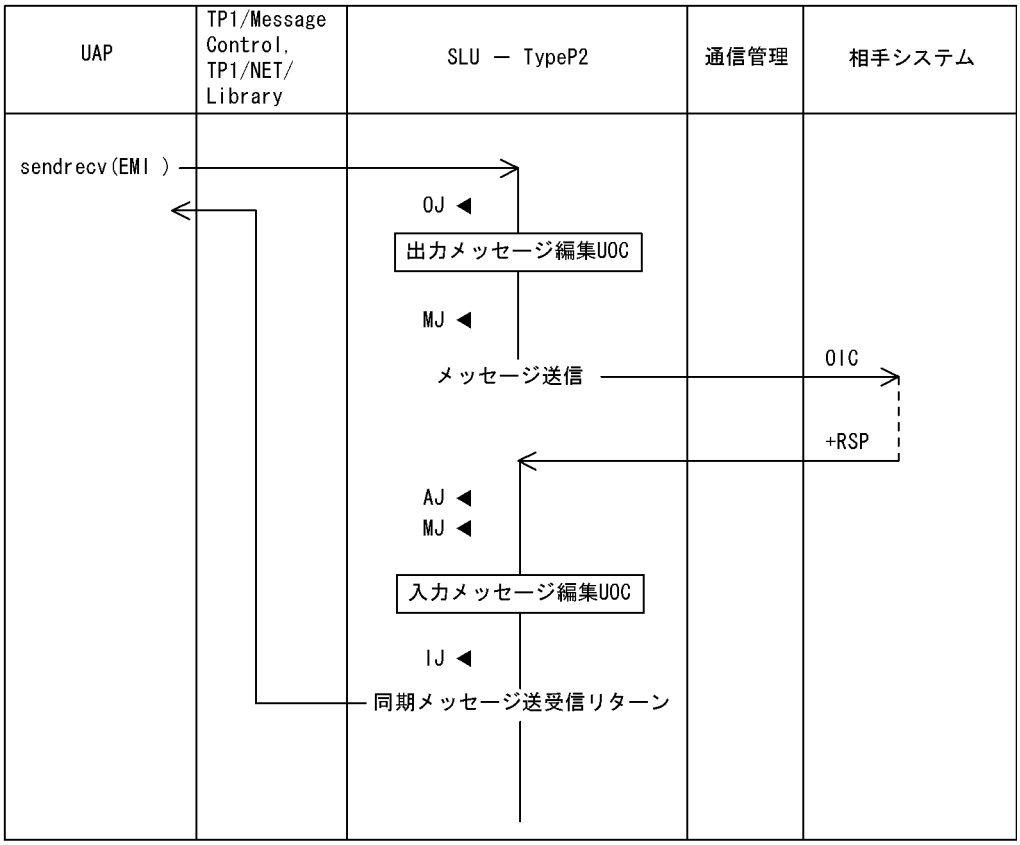
（凡例）
GJ ◀：メッセージ受信ジャーナル取得
IJ ◀：メッセージ入力ジャーナル取得
MJ ◀：メッセージジャーナル取得

図 A-4　一方送信メッセージ（receive 型／複数セグメントの場合）



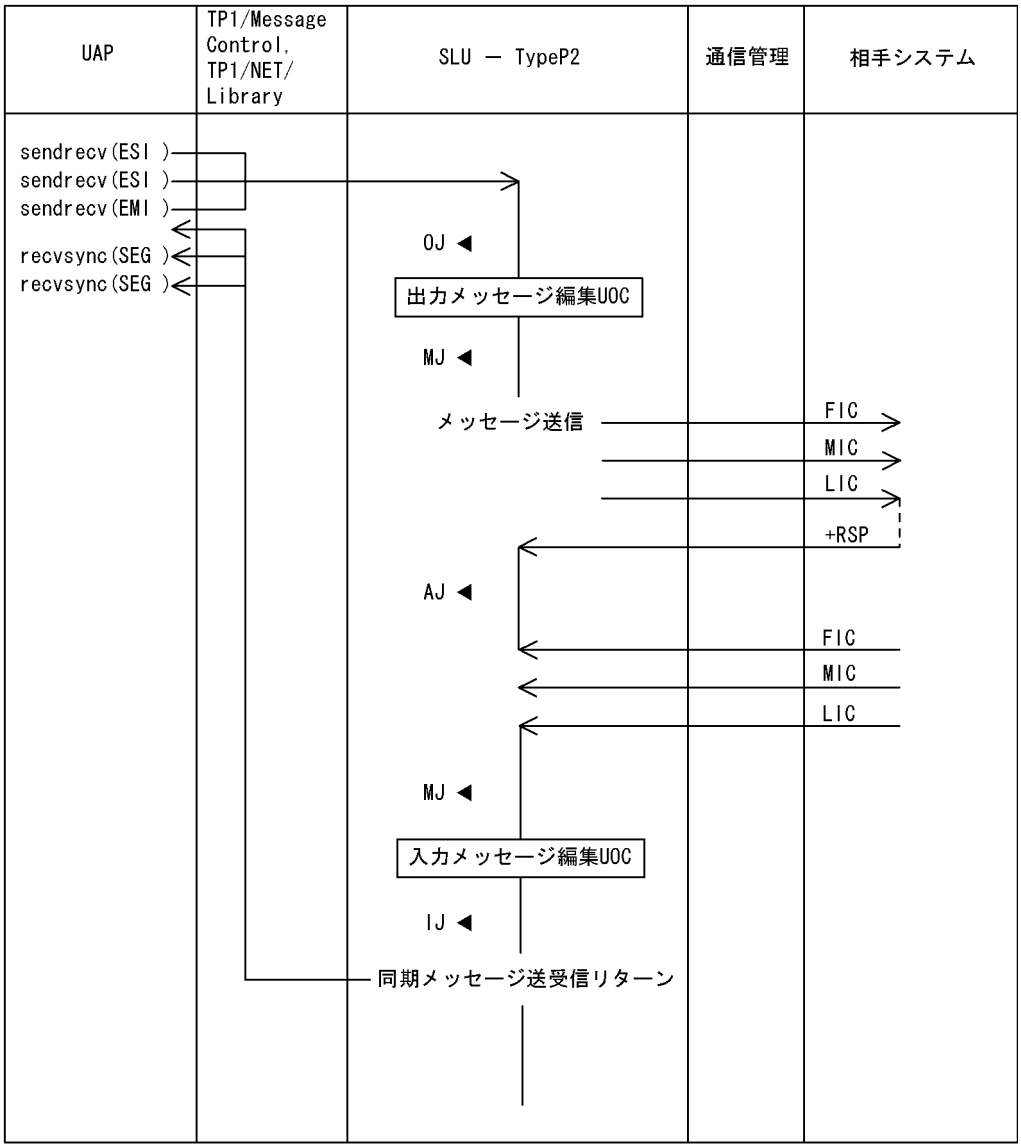
（凡例）
GJ ◀：メッセージ受信ジャーナル取得
IJ ◀：メッセージ入力ジャーナル取得
MJ ◀：メッセージジャーナル取得

図 A-5　問い合わせメッセージ（request 型／単独セグメントの場合）



（凡例）
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
MJ◀：メッセージジャーナル取得
OJ◀：メッセージ出力ジャーナル取得

図 A-6　問い合わせメッセージ（request 型／複数セグメントの場合）



（凡例）
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
MJ ◀：メッセージジャーナル取得
OJ ◀：メッセージ出力ジャーナル取得

付録 B　障害発生時の処理の流れ

障害発生時の処理の流れを図 B-1 ～図 B-5 に示します。

図 B-1　コネクション確立失敗

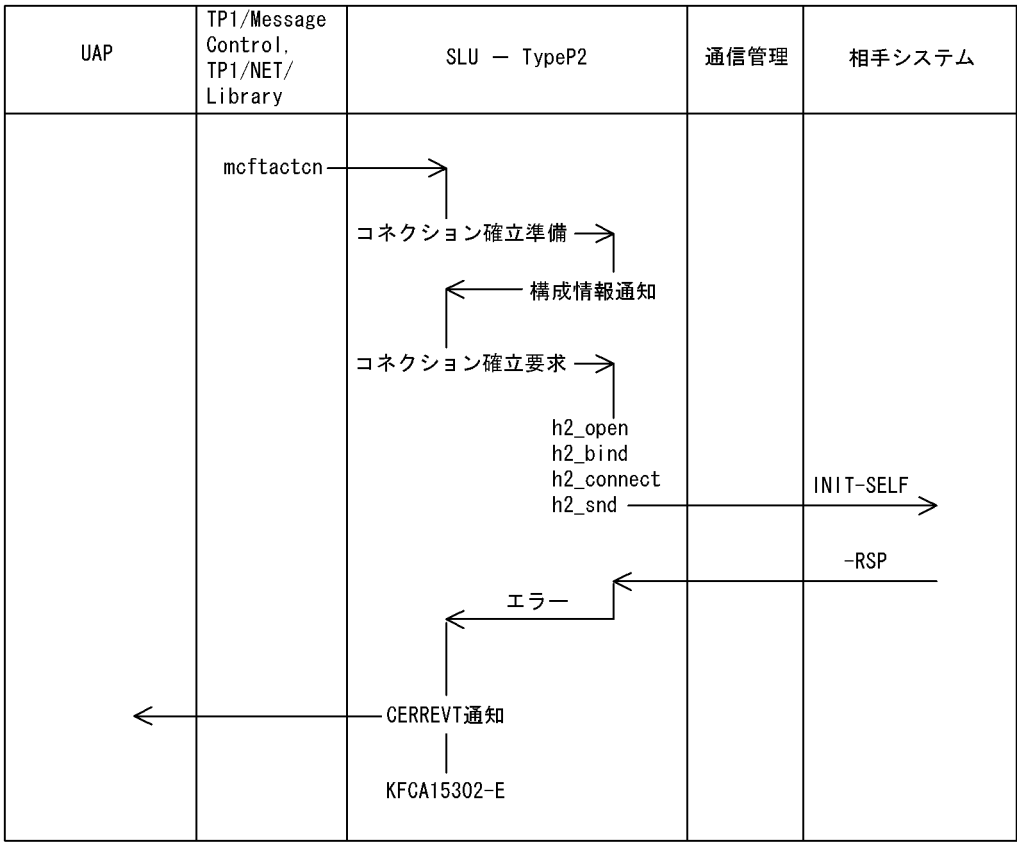


図 B-2　コネクション強制解放

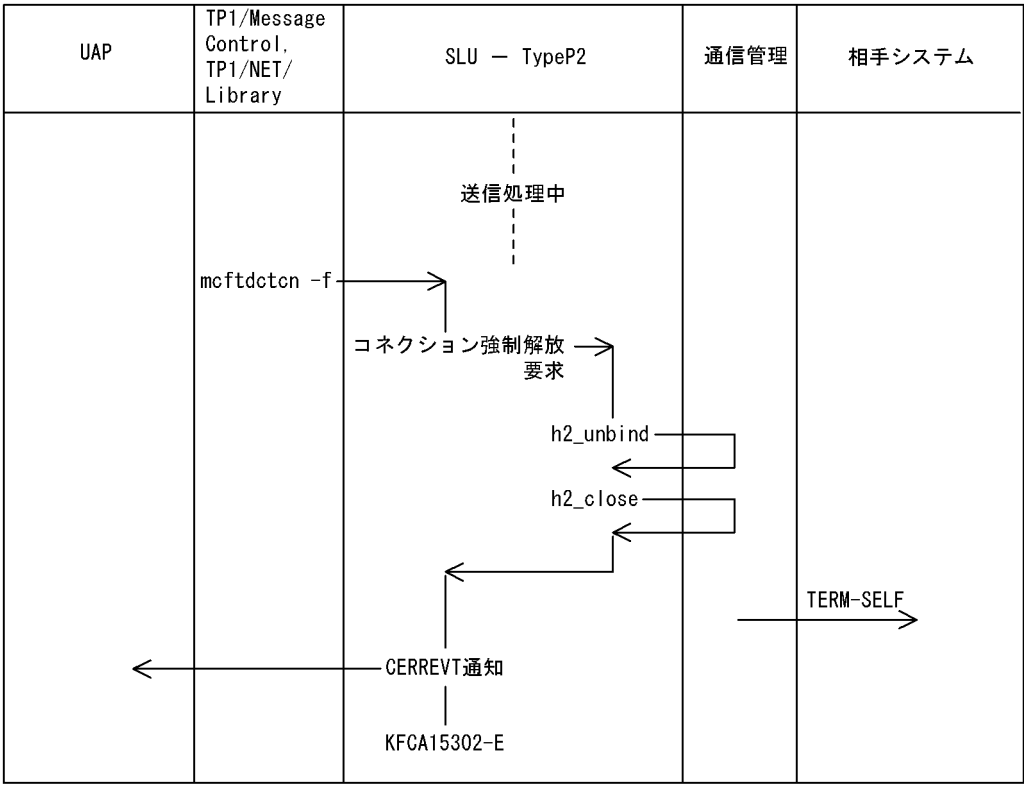
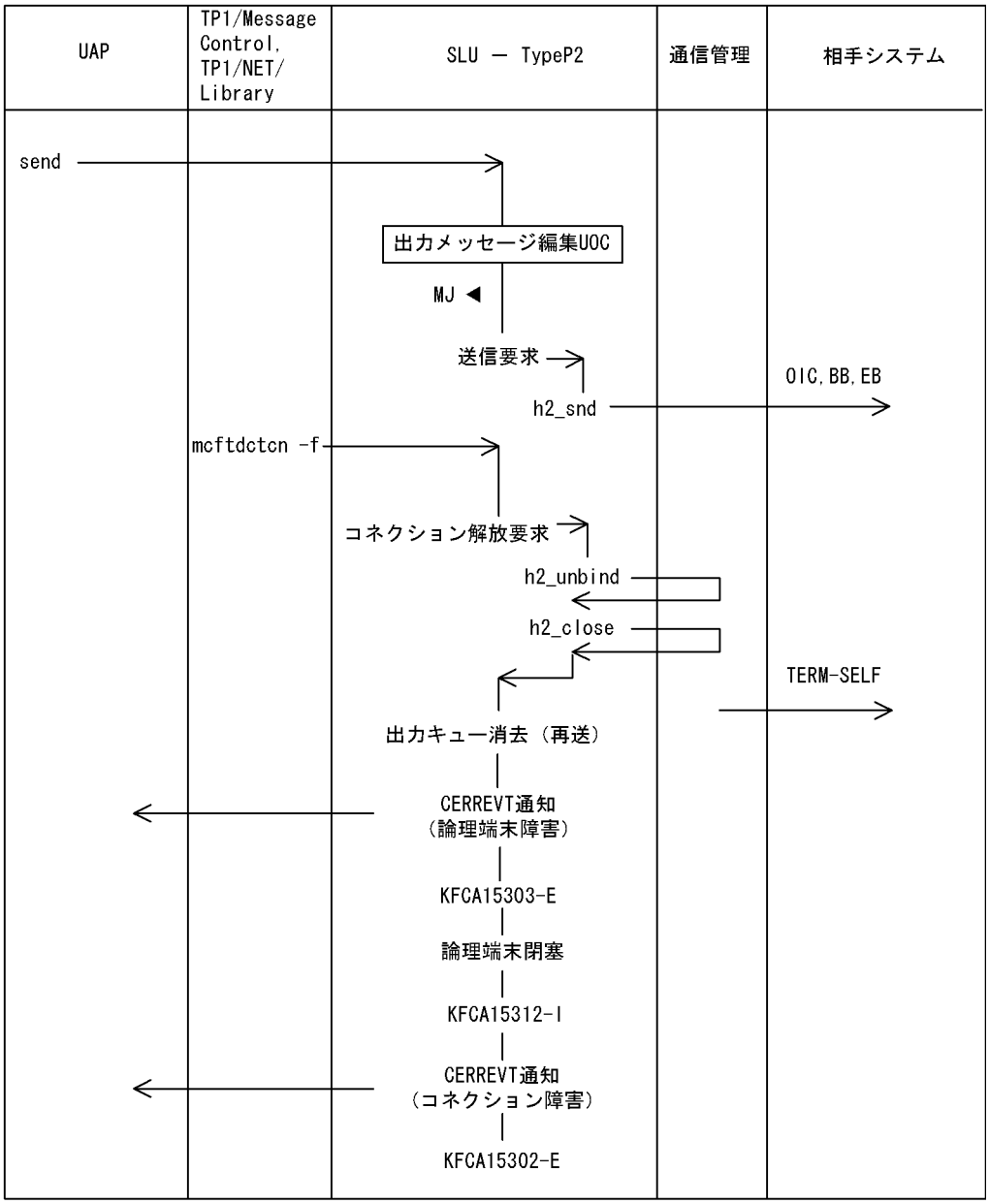
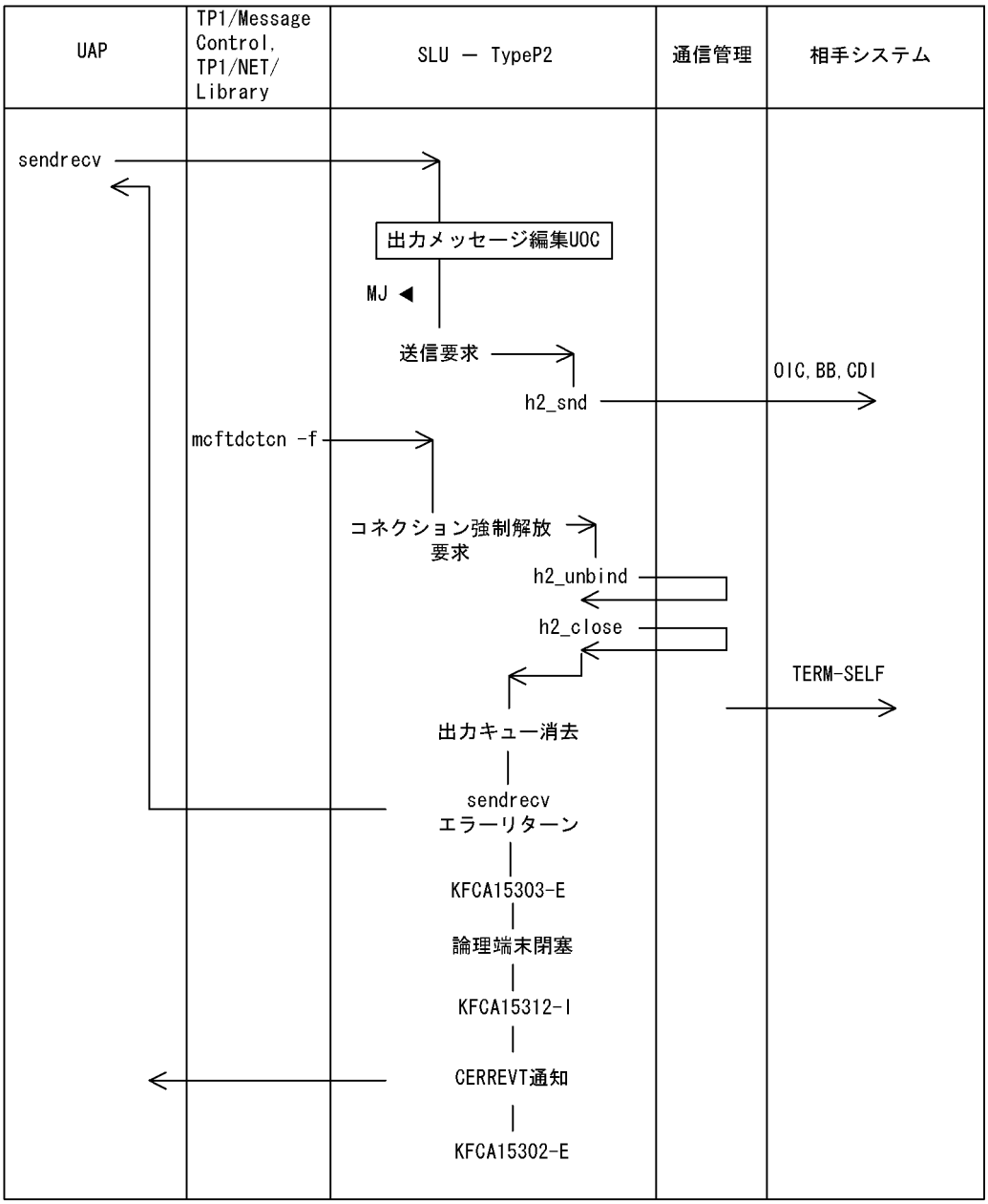


図 B-3　コネクション強制解放（send 実行中）



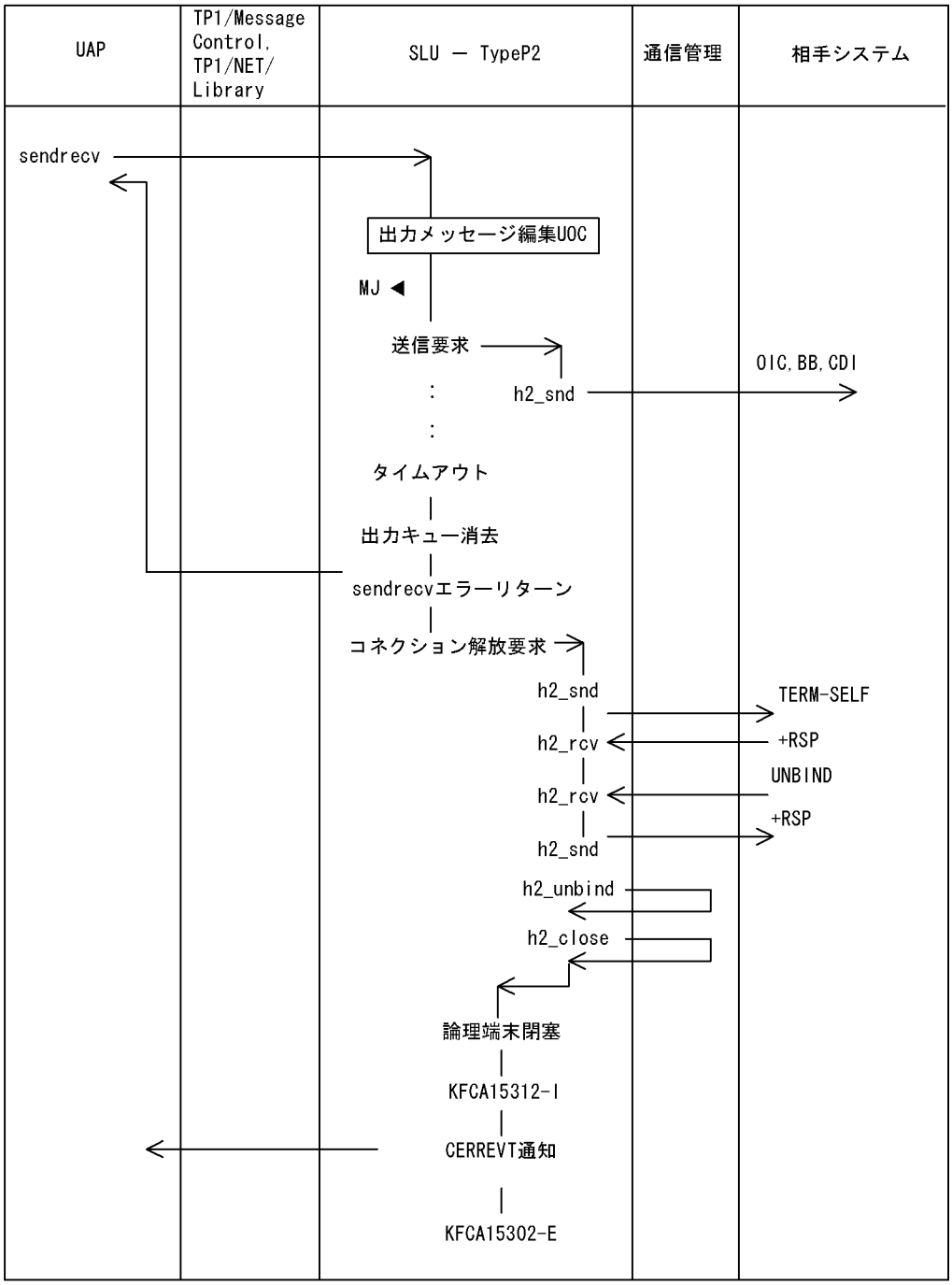
（凡例）
MJ◀：メッセージジャーナル取得

図 B-4　コネクション強制解放（sendrecv 実行中）



（凡例）
MJ ◀：メッセージジャーナル取得

図 B-5　タイムアウト



（凡例）
MJ◀：メッセージジャーナル取得

付録 C　理由コードおよびセンスコード一覧

付録 C.1　ERREVT2 の理由コード

ERREVT2 通知時の理由コードを次の表に示します。

表 C-1　ERREVT2 の理由コード

C 言語の理由コード（16 進数字）	COBOL 言語の理由コード（外部 10 進数字）	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	RPC 障害，サーバ未起動などによって，MHP または SPP の起動に失敗しました。
DCMCF_QUE_BUF_ERR（0030）	0030	メモリ不足のため，入力キューへの書き込みに失敗しました。
DCMCF_QUE_FIL_OVER（0031）	0031	キューファイルが満杯のため，入力キューへの書き込みに失敗しました。
DCMCF_QUE_LIMIT_OVER（0032）	0032	入力メッセージ最大格納数の定義指定値を超えたため，入力キューに書き込みませんでした。
DCMCF_QUE_IO_ERR（0033）	0033	入力キューへの書き込み時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中です。
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中です。
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 のセグメント受信関数にセグメントを渡す前に，MHP の異常が発生しました。

付録 C.2　CERREVT の理由コード

CERREVT 通知時の理由コードを次の表に示します。

表 C-2　CERREVT の理由コード

理由コード 1 （10 進数字）	理由コード 2 （10 進数字）	発生条件	障害レベル
DCMSLM_RSN1_MCF （MCF 障害） （1）	DCMSLM_RSN2_ITQ （0）	メッセージ入力障害	コネクション
	DCMSLM_RSN2_APL （1）	アプリケーション名取得障害	コネクション
	DCMSLM_RSN2_OTGET （2）	メッセージ出力障害	コネクション
	DCMSLM_RSN2_SLCMP （3）	メッセージ送信処理障害	コネクションまたは論理端末※
	DCMSLM_RSN2_UAPAB （5）	UAP 異常による強制解放	コネクション
	DCMSLM_RSN2_SYCER （6）	UAP への同期リターン失敗	コネクション
	DCMSLM_RSN2_ENDER （7）	終了処理中のメッセージ拒否	コネクション
	DCMSLM_RSN2_DCTLE （8）	mcftdctle による論理端末閉塞	論理端末
	DCMSLM_RSN2_LETYPE （9）	論理端末の型不正	論理端末
	DCMSLM_RSN2_ERRIND （11）	エラーデータ受信	論理端末
	DCMSLM_RSN2_DCTCN_F （536870912）	mcftdctcn -f による強制解放	コネクション
	DCMSLM_RSN2_ENDTO （536870913）	強制解放（応答メッセージ受信監視時間超過）	コネクション
	DCMSLM_RSN2_NOBUF （536870914）	バッファ取得失敗による強制解放	コネクション
	DCMSLM_RSN2_ERROR （-1）	MCF 内部矛盾による強制解放	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）	コネクション

理由コード 1 （10 進数字）	理由コード 2 （10 進数字）	発生条件	障害レベル
DCMSLM_RSN1_ABORT （プロトコル障害） （6）	DCMSLM_RSN2_XNF （16）	下位層障害	コネクション
	DCMSLM_RSN2_SLM （17）	メッセージ制御プロトコル障害	コネクション
DCMSLM_RSN1_UOC （UOC 検出障害） （3）	UOC からの詳細リターンコード	ユーザ（UOC）検出障害	コネクション
DCMSLM_RSN1_ACTER （コネクション障害） （5）	不定	コネクションの確立失敗	コネクション
その他	不定	上記以外の障害（理由コード 1，2 は保守情報）	不定

注※

コネクションに発生するのは，送信完了エラーおよび -RSP 受信時の場合です。そのほかの発生条件の場合は，論理端末に発生します。

付録 C.3　SLU - TypeP2 が使用するセンスコード

SLU - TypeP2 が使用するセンスコード（KFCA15204-E（-RSP 送信時ログ）出力時のセンスコード）を次の表に示します。

表 C-3　SLU - TypeP2 が使用するセンスコード

センスコード	意味	発生条件
0x08000000	要求不実行	入力キュー書き込み障害の発生
		メッセージ受信に対するアプリケーションスケジュール失敗
		入力メッセージ編集 UOC エラーリターン
0x08090000	モード不一致	RQD 指定のメッセージ受信後，レスポンス送信前にデータ受信
		レスポンス受信待ちで応答（EB）メッセージ受信
		チェイン受信途中ではないのに CANCEL 受信
0x08130000	ブラケット開始拒否（RTR 送信なし）	RTR のレスポンス受信前に一方（BB，EB）メッセージ受信

センスコード	意味	発生条件
		ホストに送信権委譲後に BID 受信
0x08140000	ブラケット開始拒否（RTR 送信あり）	メッセージ送信に対するレスポンス待ちのときに，一方（BB，EB）メッセージまたは BID 受信
0x08210000	セションパラメタ不正	BIND 受信時にセションパラメタ不正検出
0x10020000	RU 長不正	受信したメッセージのレングスが，セションパラメタで指定された最大 RU 長を超過
0x10030000	機能未サポート	次に示す未サポートコマンドを受信 • LUSTAT • CHASE • QEC • RELQ • SIGNAL • SBI • BIS
0x20010000	シーケンス不正	NSPE を BIND 受信待ち以外で受信
		BIND を BIND 受信待ち以外で受信
		SHUTD の二重受信
0x20020000	チェイン状態違反	FIC の二重受信
		FIC，OIC と受信
		FIC 受信に対して，-RSP（2002 または 2003）送信後，MIC または LIC 受信
0x20030000	ブラケット違反	一方（BB，EB）MIC，LIC 受信待ちのときに応答（EB）メッセージ受信
		応答（EB）メッセージ受信待ちのときに一方（BB，EB）メッセージ受信
		ホストから BB 有り EB 無しのメッセージ受信
		ホストから BB，EB 無しのメッセージ受信

付録 D　用語解説

（英字）

AJ（メッセージ送信完了ジャーナル）

SLU - TypeP2 で取得する履歴情報の一つです。メッセージの送信完了情報である送信通番と送信先論理端末名称とで構成されます。
AJ の取得タイミングは，相手システムにメッセージを送信し，その送信完了を受信した直後です。

GJ（メッセージ受信ジャーナル）

SLU - TypeP2 で取得する履歴情報の一つです。メッセージの受信情報である，メッセージ長，論理端末名称などの情報です。
GJ の取得タイミングは，receive を発行して，入力キューから取り出したメッセージを UAP に渡した直後です。

IJ（メッセージ入力ジャーナル）

入力キューに入力された情報です。論理端末名称と入力メッセージの情報で構成されます。
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入力する直前です。

MJ（メッセージジャーナル）

SLU - TypeP2 で取得する履歴情報の一つです。端末名称，メッセージ種別，入力メッセージ編集前のデータ，および出力メッセージ編集後のデータの情報です。
MJ の取得タイミングは，メッセージ送信前，入力メッセージ編集 UOC 処理前，および出力メッセージ編集 UOC 処理後です。

OJ（メッセージ出力ジャーナル）

出力キューに入力された情報です。論理端末名称，出力メッセージ，セグメント種別などで構成されます。
OJ の取得タイミングは，UAP から送信要求を受け付けた時です。

SLUTYPE-P プロトコル

ホストシステムとインテリジェント端末（OpenTP1 システムの論理端末など）とを接続するためのプロトコルです。米国 IBM 社が開発したネットワークアーキテクチャである SNA に準拠しています。

SNA

米国 IBM 社が開発したネットワークアーキテクチャの名称です。SLU - TypeP2 は，このアーキテクチャに従ったプロトコルを実装しています。

（カ行）

コネクション

SLU - TypeP2 が AP 間通信をするときに，自システムと相手システムの間に確立する論理的通信路

です。

（サ行）

セション

SLUTYPE-P プロトコルでの論理的通信路です。SLU - TypeP2 ではコネクションと呼びます。

（ラ行）

論理端末

SLU - TypeP2 と UAP との通信接点です。SLU - TypeP2 と UAP は，論理端末単位にメッセージを送受信します。

索引

記号

A

C

D

E

G

I

L

M

O

P

R

S

T

U

あ

い

う

お

き

く

こ

し

せ

そ

た

つ

て

と

に

ね

は

ふ

へ

ほ

み

め

ゆ

よ

り

ろ

ソフトウェアマニュアルのサービス ご案内

ソフトウェアマニュアルについて，３種類のサービスをご案内します。ご活用ください。

１．マニュアル情報ホームページ

ソフトウェアマニュアルの情報をインターネットで公開しております。

URL　http://www.hitachi.co.jp/soft/manual/

ホームページのメニューは次のとおりです。

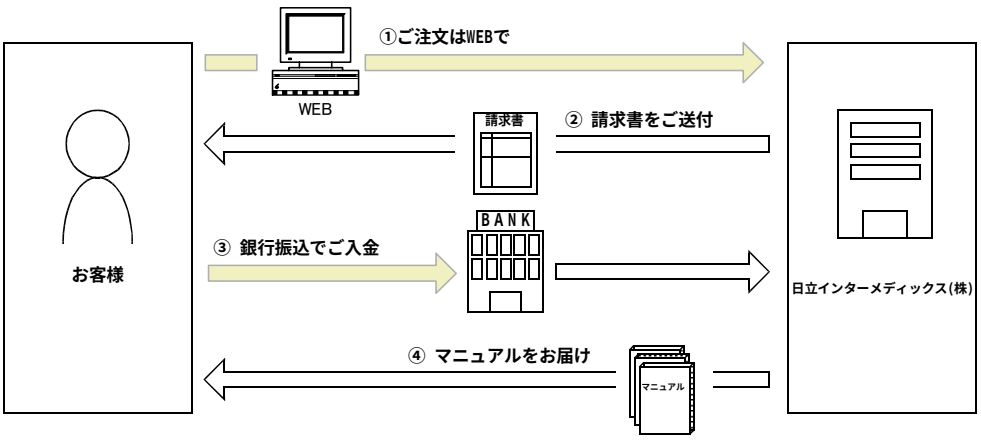
■マニュアル一覧	日立コンピュータ製品マニュアルを製品カテゴリ，マニュアル名称，資料番号のいずれかから検索できます。
■CD-ROMマニュアル情報	複数マニュアルを格納したCD-ROMマニュアルを提供しています。どの製品に対応したCD-ROMマニュアルがあるか，を参照できます。
■マニュアルのご購入	日立インターメディックス(株)の「日立コンピュータ製品マニュアルサイト」からお申し込みできます。 (詳細は「３．マニュアルのご注文」を参照してください。)
■Web提供マニュアル一覧	インターネットで参照できるマニュアルの一覧を提供しています。 (詳細は「２．インターネットからのマニュアル参照」を参照してください。)
■ご意見・お問い合わせ	マニュアルに関するご意見，ご要望をお寄せください。

２．インターネットからのマニュアル参照(ソフトウェアサポートサービス)

ソフトウェアサポートサービスの契約をしていただくと，インターネットでマニュアルを参照できます。
本サービスの対象となる契約の種別，及び参照できるマニュアルは，マニュアル情報ホームページでご確認ください。なお，ソフトウェアサポートサービスは，マニュアル参照だけでなく，対象製品に対するご質問への回答，問題解決支援，バージョン更新版の提供など，お客様のシステムの安定的な稼働のためのサービスをご提供しています。まだご契約いただいていない場合は，ぜひご契約いただくことをお勧めします。

３．マニュアルのご注文

日立インターメディックス(株)の「日立コンピュータ製品マニュアルサイト」からご注文ください。



① 下記 URL にアクセスして必要事項を入力してください。

URL　http://www2.himdx.net/manual/privacy.asp?purchase_flag=1

② ご注文いただいたマニュアルについて，請求書をお送りします。

③ 請求書の金額を指定銀行へ振り込んでください。なお，送料は弊社で負担します。

④ 入金確認後，７日以内にお届けします。在庫切れの場合は，納期を別途ご案内いたします。