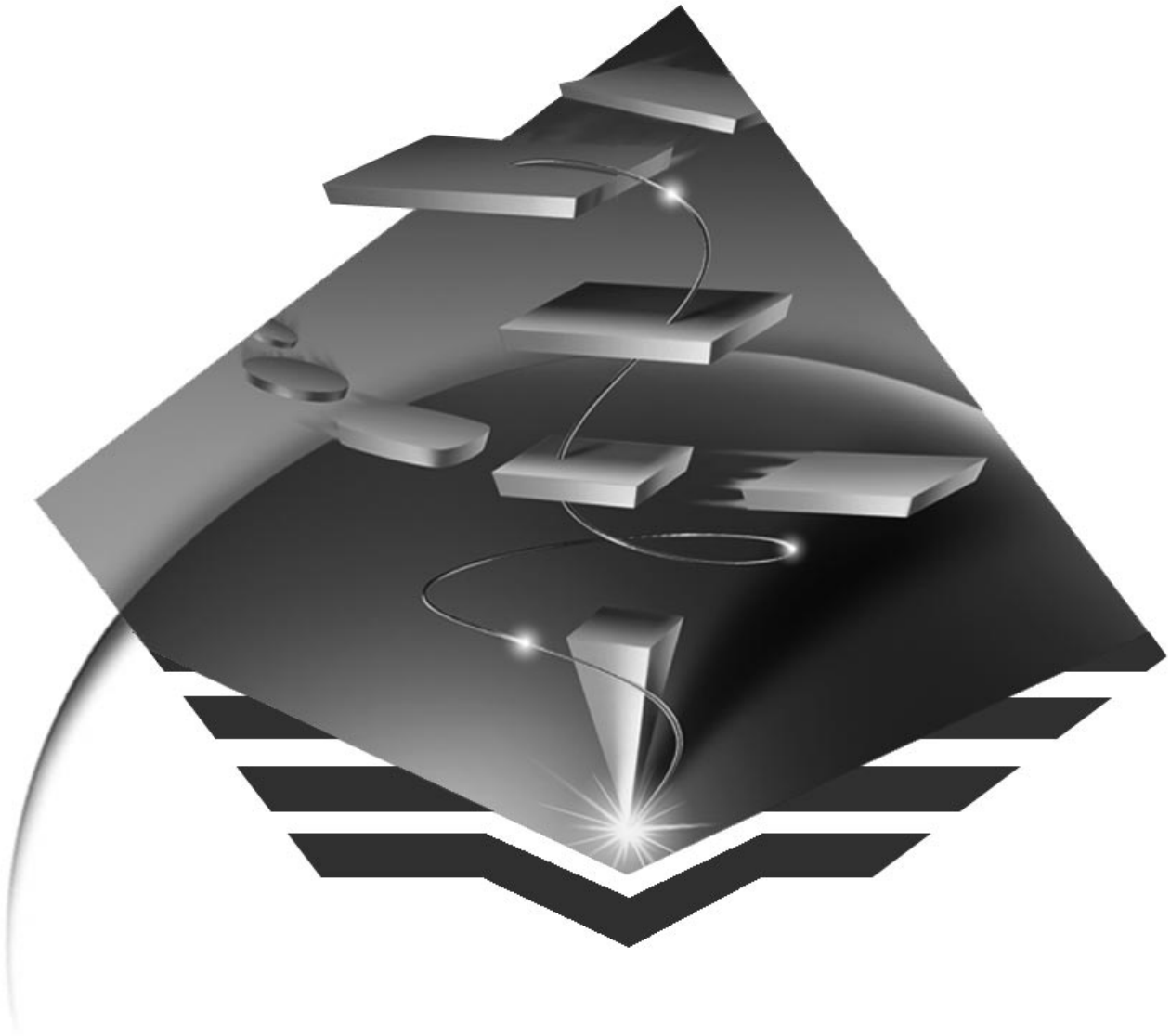
Communications Server for Windows** NT

システム管理プログラミング

バージョン 5.0



SC88-7728-00
英文原典：SC31-8426-00

Communications Server for Windows** NT

システム管理プログラミング

バージョン 5.0

英文原典：SC31-8426-00
SC88-7728-00

ご注意

本書の情報およびそれによってサポートされる製品を使用する前に、xiiiページの『特記事項』に記載する一般情報をお読みください。

本書は、Communications Server バージョン 5.0 に適用されます。

原　典：	SC31-8426-00 Communications Server for Windows** NT Client/Server System Management Programming Version 5.0
発　行：	日本アイ・ビー・エム株式会社
担　当：	ナショナル・ランゲージ・サポート

第1刷 1997.4

この文書では、平成明朝体™W3 および平成角ゴシック体™W5を使用しています。この(書体*)は、(財) 日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成角ゴシック体™W5

目次

図

表

特記事項

本書において、日本では発表されていない IBM 製品、プログラム、およびサービスについて言及または説明する場合があります。しかし、このことは、 IBM がこのような製品、プログラム、およびサービスを、日本で発表する意図があることを必ずしも示すものではありません。本書で IBM 製品、プログラム、またはサービスに言及している部分があっても、このことは IBM 製品,プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらのプログラムまたは製品に代えて、IBM の知的所有権を侵害することのない機能的に同等のプログラムまたは製品を使用することができます。ただし、 IBM によって明示的に指定されたものを除き、これらの製品、プログラム、またはサービスに関連する動作の評価および検査は、お客様の責任で行っていただきます。

IBMは、本書で説明する主題に関する特許権(特許出願を含む）、商標権、または著作権を所有している場合があります。本書は、これらの特許権、商標権、および著作権について、本書で明示されている場合を除き、実施権、使用権等を許諾することを意味するものではありません。実施権、使用権等の許諾については、下記の宛先に、書面による照会状を送付してください。

〒106 東京都港区六本木 3 丁目 2-31

AP 事業所

IBM World Trade Asia Corporation

Intellectual Property Law & Licensing

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および(ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

Site Counsel

IBM Corporation

P.O. Box 12195

3039 Cornwallis Road

Research Triangle Park, NC 27709-2195

USA

本プログラムに関する上記の情報は、適切な条件の下で使用することができますが、有償の場合もあります。

本書に記載するライセンス・プログラムおよびそのライセンス資料のすべては、「IBM プログラム使用契約書」の契約条項にもとづいて弊社から提供されるものです。

本書は、実働面での使用を意図したものではなく、いかなる種類の保証も含まずそのままの形でお届けするものです。したがって、市販性および特定目的への適合性を含めたいかなる保証も認められません。

商標

以下の用語は、米国 IBM Corp.の商標です。

ACF/VTAM

拡張対等通信ネットワーキング

AFP

AIX

AnyNet

APPN

AS/400

CICS

Common User Access

CUA

IBM

IMS

MVS

MVS/ESA

MVS/XA

NetView

Operating System/2

OS/2

OS/400

RACF

System/370

Virtual Machine/Enterprise Systems Architecture

VM/ESA

VTAM

PC Direct は Ziff Communications Company の商標で、ライセンスにもとづき IBM Corporation が使用しています。

UNIX はX/Open カンパニーリミテッドが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。

C-bus は Corollary, Inc の商標です。

Microsoft, Windows, Windows 95 ロゴは、Microsoft Corporation の商標または登録商標です。

Java および HotJava は、Sun Microsystems, Inc. の商標です。

二重アスタリスク (**) で示されている会社名、製品名、サービス名は、他社の商標またはサービス・マークです。

本書について

本書では、IBM Communications Server for Windows NT** を使用するプログラムの開発方法について説明します。 本文中の *Windows* という用語は、Windows NT を指します。 本文中のワークステーション という用語は、サポートされているすべてのパーソナル・コンピューターを指します。 言及されるパーソナル・コンピューターのモデルやアーキテクチャーが 1 つだけの場合には、そのタイプだけが示されています。

本書の対象読者

本書は、ノード・オペレーター機能 (NOF) API メッセージを使用して、Communications Server の操作の管理や照会をしようと考えているプログラマーおよび開発者を対象としています。

本書は、また、リモート（ホスト・フォーカル・ポイント）ネットワーク管理アプリケーションと通信するために、Communications Server によって提供される基本管理サービス・サポートを使用するネットワーク管理アプリケーションを作成している開発者も対象にしています。

本書の使用方法

本書は、3 部編成となっています。 1ページの『第1部 Communications Server ノード・オペレーター機能』は、以下の章から構成されています。

- 7ページの『第1章 はじめに』では、本書の目的を説明します。
- 11ページの『第2章 本書の verb についての概要』では、ノード・オペレーター機能 API 構造およびそれがサポートする verb について説明します。 この章では、Communications Server が実現する verb のカテゴリーと、Communications Server が提供する追加シグナルのカテゴリーの概要を述べます。
- 21ページの『第3章 ノード・オペレーター機能エントリー・ポイント』では、エントリー・ポイントの拡張について説明します。
- 第 4 章から第 11 章までは、各 verb の構文について説明しています。 各 verb の情報をもった構造のコピーを示し、各項目について説明して、続いて、戻される可能性のある戻りコードをリストしています。

529ページの『第2部 Communications Server 管理サービス API』は、以下の章から構成されています。

- 531ページの『第12章 管理サービス API について』では、管理サービス API について説明します。
- 535ページの『第13章 管理サービスのエントリー・ポイント』では、管理サービス verb に対するエントリー・ポイントを説明します。

- 547ページの『第14章 管理サービス verb』では、各 verb の構文について説明します。 各 verb の情報をもった構造のコピーを示し、各項目について説明して、続いて、戻される可能性のある戻りコードをリストしています。

561ページの『第3部 Communications Server ASCII 構成』は、以下の章から構成されています。

- 563ページの『第15章 ASCII 構成について』では、Communications Server ASCII の構成および検証ユーティリティーについて説明します。
- 567ページの『第16章 ASCII 構成キーワード』では、キーワードの種類とタイプについて説明します。

アイコン

本書では、特別情報を参照する必要がある場合、次のアイコンが現れます。

鳴っている電話。

本書で使用している表記法

Communications Server ライブラリーを通じて、以下の表記法が使用されています。リストされた表記法には、本書では使用されないものもあります。

テキストの表記規則

Bold	太字は、プログラムまたはコマンド・プロンプトで使用することのできる verb、関数、およびパラメーターを示します。 これらの値には大文字小文字の区別があり，テキストに記載されているとおりに正確に入力しなければなりません。
Italics	イタリック体は以下のものを表します。 • ユーザーが値を与える変数。 • リスト、チェック・ボックス、入力フィールド、押しボタン、およびメニュー選択などのウィンドウ制御の名前。 これらは、ウィンドウに表示されるとおりにテキストに表示されます。 • 資料のタイトル。 • 文字として使用されている文字、または語として使用されている語。 **例:** *a* と記載されている場合、それが *an* であるとみなされないことを確認してください。
Bold italics	太字のイタリック体は、語を強調するために使用されます。
UPPERCASE	英大文字は、定数，ファイル名，キーワード，およびプログラム内またはコマンド・プロンプトで使用することができるオプションを指します。これらの値は大文字で入力することも小文字入力することもできます。
二重引用符	二重引用符は、ウィンドウに表示されるメッセージを示します。 この一例には、エミュレーター・セッションの操作員情報域 (OIA) に表示されるメッセージがあります。
`Example type`	この字体は、コマンド・プロンプトまたはウィンドウにタイプ（入力）すべき情報を示します。

数字の表記規則

2 進数	テキスト（たとえば、『2 進数xxxx xxxx の値は ... です』）で表される場合を除き、BX'xxxx xxxx' または BX'x' のように表されます。
ビット位置	右端 (最小有効ビット) の 0 から始まります。
10 進数	5 桁以上の 10 進数は、メートル法形式で表されます。 3 桁のグループを区切るには、コンマではなくスペースが使用されます。 たとえば、16147 という数は、16 147 と表記されます。
16 進数	テキスト中では 16 進数 xxxx または X'xxxx' と表記されます（『隣接ノードのアドレスは 16 進数 5D で、これは X'5d' を示します』）。

関連情報について

SNA、APPN、あるいは LU 6.2 アーキテクチャーに関する情報については、以下の IBM 資料を参照してください。

- *IBM Systems Network Architecture: LU 6.2 Reference: Peer Protocols,* SC31-6808（ソフトコピーのみ）
- *IBM Systems Network Architecture: APPN Architecture Reference,* SC30-3422
- *IBM Systems Network Architecture: Management Services,* SC30-3346
- *IBM Systems Network Architecture: Formats,* GA27-3136
- *IBM APPN Architecture and Product Implementations Tutorial,* GG24-3669
- *IBM コミュニケーション・マネージャー/2 システム管理プログラミング解説書*、SC88-5528（ソフトコピーのみ）
- *IBM コミュニケーション・マネージャー/2 APPC プログラミングの手引き*、SC88-5521（ソフトコピーのみ）
- *IBM システム/370 解説書*、N:GA22-7000
- *IBM Systems Network Architecture: Technical Overview,* GC30-3073
- *IBM Systems Network Architecture: VTAM Programming for LU Type 6.2,* SC30-3400
- *システム・ネットワーク体系　概念と諸製品*、GC30-3072
- *IBM Systems Network Architecture Format and Protocol Reference Manual: Architecture Logic for LU Type 6.2,* SC30-3269
- *IBM Systems Network Architecture: Transaction Programmer's Reference Manual for LU Type 6.2,* GC30-3084
- *IBM Systems Network Architecture Format and Protocol Reference Manual: Architectural Logic,* SC30-3112

第1部 Communications Server ノード・オペレーター機能

第1章 はじめに

このパートでは、Communications Server によって提供されるノード・オペレーター機能 (NOF) API について説明します。

本書の目的

本書の目的は以下のとおりです。

- ノード・オペレーター機能 API の構造の概要を簡単に説明する。
- インターフェースを渡って送られるシグナルの構文を定義する。

Communications Server ノード・オペレーター機能

Communications Server ノード・オペレーター機能により、ノード・オペレーターと、制御点 (CP) および論理装置 (LU) との通信が可能になります。 ノード・オペレーター機能は、ノードの開始時に制御点を初期化するために使用するノードの構成情報をオペレーターから受け取ります。ノード・オペレーター機能は、また、ノードの構成情報を照会し、表示する要求も受け取ります。 ノード・オペレーターは、以下のことができます。

- LU、DLC、ポートおよびリンクの定義ならびに削除。
- リンクおよびセッションの活動化ならびに非活動化
- データベースおよび状況情報を、制御点および LU に照会。

ノード・オペレーターとは、対話型ディスプレイを使って作業する操作員、ファイル・インターフェースによってアクセスされるコマンド・ファイル、またはトランザクション・プログラムということになります。ノード・オペレーター機能は、verb インターフェースを使って、このノード・オペレーターと通信します。

エントリー・ポイント

Communications Server は、ノード・オペレーター機能 verb を扱うライブラリー・ファイルを提供します。

ノード・オペレーター機能の verb は、簡単な言語によるインターフェースを採ります。ユーザー・プログラムが、*verb 制御ブロック*と呼ばれる記憶域にあるブロック内のフィールドを埋めます。 それから、プログラムはエントリー・ポイントを呼び出して、verb 制御ブロックへのポインターを渡します。この操作が完了すると、ノード・オペレーター機能は元に戻り、verb 制御ブロック内のフィールドは使用されて、修正されています。 そこで、プログラムは、戻りパラメーターを verb 制御ブロックから参照することができます。

ノード・オペレーター機能 verb に対するエントリー・ポイントのリストは以下のとおりです。

- WinAsyncNOF()
- WinAsyncNOFEx()

- WinNOFCancelAsyncRequest()
- WinNOFCleanup()
- WinNOFStartup()
- WinNOFRegisterIndicationSink()
- WinNOFUnregisterIndicationSink()
- WinNOFGetIndication()

エントリー・ポイントの詳細説明については、第3章 ノード・オペレーター機能エントリー・ポイントを参照してください。

verb 制御ブロック (VCB)

プログラミング上の注意事項 : 基本オペレーティング・システムは、呼出し側アプリケーションのアドレス空間でいくつかのサブシステムを実行することで、パフォーマンスを最適化します。これは、アプリケーション・プログラムによってローカル記述子テーブル (LDT) セレクターが誤って使用されると、誤操作または、場合によってはシステム障害の原因となることを意味します。したがって、アプリケーション・プログラムでは、ポインターの LDT セレクター・フィールドの変更を伴うようなポインターの算術演算を実行しないようにする必要があります。

verb 制御ブロック (VCB) に使用するセグメントは、読取り/書込みデータ・セグメントでなければなりません。 ユーザー・プログラムは、VCB をそのプログラム内の変数として宣言すること、VCB を割り振ること、またはより大きいセグメントからVCB を 2 次割振りすることができます。 このセグメントには、プログラムが発行する verb のすべてのフィールドを入れる十分な大きさが必要です。

アプリケーション・プログラムは、verb が発行されてから完了するまでは、verb 制御ブロック のどの部分も変更しません。 ノード・オペレーター機能は、ある verb の実行を終了すると、元のブロックに修正済みの完全な VCB を複写して戻します。したがって、プログラムが変数として verb 制御ブロックを宣言する場合は、内部プロシージャーのスタック上ではなく、静的記憶域で宣言することを考慮してください。

各 VCB の中の予約済で未使用のフィールドは、すべてゼロ (X'00') で埋めてください。実際には、プログラムが値をパラメーターに割り当てる前に、verb 制御ブロック全体をゼロに設定する方が時間効率がよい場合があります。 予約済みフィールドをゼロに設定しておくことは、特に重要です。

注: VCB が読取り/書込みでない場合、または VCB が 10 バイト（つまり、ノード・オペレーター機能の 1 次戻りコードおよび 2 次戻りコードを収容するために十分な大きさ）より短い場合は、ノード・オペレーター機能はその VCB にアクセスできないため、基本オペレーティング・システムはこのプロセスを異常終了させます。 このような終了は、*一般保護障害*、プロセッサー例外トラップ D として認識されます。

VCB が短かすぎる場合、または間違ったタイプのセグメントが使用された場合、ノード・オペレーター機能は、1 次戻りコード INVALID_VERB_SEGMENT を戻します。

ノード・オペレーター機能 (NOF) プログラムの作成

Communications Server は、NOF verb を扱う動的リンク・ライブラリー (DLL) ファイルを提供します。

この DLL は再入可能で、複数のアプリケーション・プロセスおよびスレッドが、DLL を同時に呼び出すことができます。

NOF verb は、簡単な言語によるインターフェースを採ります。 ユーザー・プログラムが、*verb 制御ブロック* (VCB) と呼ばれる記憶域にあるブロック内のフィールドを埋めます。それから、プログラムは NOF DLL を呼び出して、 verb 制御ブロックへのポインターを渡します。この操作が完了すると、NOF は元に戻り、VCB 内のフィールドは使用されて、修正されています。 そこで、プログラムは、戻りパラメーターを verb 制御ブロックから参照することができます。

9ページの表 1は、NOF プログラムをコンパイルしてリンクするために必要な、提供ヘッダー・ファイルおよびライブラリーのソース・モジュール使用法を示します。ヘッダー・ファイルのいくつかは、その他の必須ヘッダー・ファイルを含んでいる場合があります。

表 1. NOF 用のヘッダー・ファイルおよびライブラリー

オペレーティング・システム	ヘッダー・ファイル	ライブラリー	DLL 名
WINNT & WIN95	WINNOF.H	WINNOF32.LIB	WINNOF32.DLL
WIN3.1	WINNOF.H	WINNOF.LIB	WINNOF.DLL
OS/2	APPC_C.H	APPC.LIB	APPC.DLL

SNA API クライアントによるサポート

SNA API クライアントは、完全なノード・オペレーター機能の 1 つのサブセットだけをサポートできます。 具体的には、**WINNOF** が、Windows クライアント (95、NT、3.1) 上でサポートされる唯一の API です。以下のリストは、サポートされる NOF verb です。

- QUERY_LOCAL_LU
- QUERY_LU_0_TO_3
- QUERY_LU_POOL
- QUERY_MODE
- QUERY_MODE_DEFINITION
- QUERY_PARTNER_LU
- QUERY_PARTNER_LU_DEFINITION
- QUERY_PU

- QUERY_SESSION
- QUERY_TP
- QUERY_TP_DEFINITION

第2章 本書の verb についての概要

本書で説明されている verb インターフェースにより、プログラムは、Communications Server ネットワーク環境に関連する大部分の構成、システム管理、およびノード定義機能を実行することができます。 この章では、これらの各機能およびそれと関連のある verb についての概要を述べます。

verb 記述の読み方

第 4 章から第 11 章までは、構成、システム管理、および接続マネージャーの verb について説明しています。

指定パラメーター

各 verb 記述ごとに、プログラムで指定するパラメーターとそれに関連するパラメーター値について詳細に説明する項があります。

場合によっては，パラメーターに変数値を与える必要があります。

戻りパラメーター

各 verb 記述ごとに、プログラムに戻されるパラメーターとそれに関連するパラメーター値について詳細に説明する項があります。

戻りコード

本書で説明している構成、システム管理、および接続マネージャーの verb には、それらに関連する戻りコードがあり、この戻りコードが、verb の実行が正常終了した場合の情報またはエラー情報を提供します。 これらのコードは各 verb の『戻りパラメーター』の項にリストされています。

追加情報

verb の説明の多くには、『追加情報』という項もあります。 この項には，その verb について、役に立つ追加情報が記載されています。

共通 VCB フィールド

この章では、ノード・オペレーター機能 API を通して渡される各 verb の構文について説明します。 ここでは、各 verb ごとに、渡したり、戻されたりするパラメーターについても説明します。

```
typedef struct nof_hdr
{
        unsigned short  opcode;
        unsigned char   reserv2;        /* reserved             */
```

```
        unsigned char   format;
        unsigned short  primary_rc;
        unsigned long   secondary_rc;
} NOF_HDR;
```

各 VCB ごとに数多くの共通フィールドをもっています。 これらを以下にリストして、説明します。

opcode
: verb 操作コード。 このフィールドにより verb が識別されます。

形式
: VCB の形式を示します。 VCB の現行バージョンを指定するために、このフィールドに設定しなければならない値は、各 verb の下にそれぞれ説明されています。

primary_rc
: 1 次戻りコード。 各 verb が取り得る値は、verb の各項ごとにリストされています。

secondary_rc
: 2 次戻りコード。 これは、1 次戻りコードによって与えられる情報を補足するものです。

verb の要約

ノード・オペレーター機能 API は、以下のことを行うために使用することのできる verb から構成されています。

- ノード資源の構成
- リンクおよびセッションの活動化ならびに非活動化
- ノードが保有する情報の照会
- セッション数の変更
- 非送信請求指示の取扱い
- パスワード・サポートの提供
- リモート LU の 『ping』
- CPI-C 側の情報の定義、照会および削除

ノード構成

以下の verb は、資源を定義するために使用することができます。

- DEFINE_ADJACENT_NODE
- DEFINE_CN
- DEFINE_COS
- DEFINE_DEFAULT_PU
- DEFINE_DLC
- DEFINE_DLUR_DEFAULTS
- DEFINE_DOWNSTREAM_LU

- DEFINE_DOWNSTREAM_LU_RANGE
- DEFINE_FOCAL_POINT
- DEFINE_INTERNAL_PU
- DEFINE_LOCAL_LU
- DEFINE_LS
- DEFINE_LU_0_TO_3
- DEFINE_LU_0_TO_3_RANGE
- DEFINE_LU_POOL
- DEFINE_MODE
- DEFINE_PARTNER_LU
- DEFINE_PORT
- DEFINE_TP

以下の verb は、資源を削除するために使用することができます。

- DELETE_ADJACENT_NODE
- DELETE_CN
- DELETE_COS
- DELETE_DLC
- DELETE_DOWNSTREAM_LU
- DELETE_DOWNSTREAM_LU_RANGE
- DELETE_FOCAL_POINT
- DELETE_INTERNAL_PU
- DELETE_LOCAL_LU
- DELETE_LS
- DELETE_LU_0_TO_3
- DELETE_LU_0_TO_3_RANGE
- DELETE_LU_POOL
- DELETE_MODE
- DELETE_PARTNER_LU
- DELETE_PORT
- DELETE_TP

活動化および非活動化

以下の verb は、リンク・レベルで使用されます。

- START_DLC
- START_LS
- START_PORT
- STOP_DLC

- STOP_LS
- STOP_PORT

以下の verb は、独立 LU リクエスター機能に使用されます。

- START_INTERNAL_PU
- STOP_INTERNAL_PU

以下の verb はセッション・レベルで使用されます。

- ACTIVATE_SESSION
- DEACTIVATE_CONV_GROUP
- DEACTIVATE_SESSION

次の verb により、高性能経路指定 (HPR) RTP 接続がパスを切り替えるようにします。

PATH_SWITCH

ノードの照会

以下の verb は、名前付きフィールドにノードの情報を戻します。

- QUERY_DEFAULT_PU
- QUERY_MDS_STATISTICS
- QUERY_NN_TOPOLOGY_STATS
- QUERY_NODE
- QUERY_STATISTICS

以下の verb は、1 つまたは複数単位の情報を戻すことができます。

- QUERY_ADJACENT_NN
- QUERY_CN
- QUERY_CN_PORT
- QUERY_COS
- QUERY_DEFAULTS
- QUERY_DLUS
- QUERY_DOWNSTREAM_PU
- QUERY_FOCAL_POINT
- QUERY_LU_POOL
- QUERY_MDS_APPLICATION
- QUERY_MODE_TO_COS_MAPPING
- QUERY_NMVT_APPLICATION
- QUERY_PU
- QUERY_TP

この情報は、リストの形式で格納されていると考えることができます。verb はリスト中の名前付き項目を指定することができます。この項目は、そのリストにおける

プレース・マーカー（または、索引値）であると考えられるものです。 これらの verb 上の **list_options** フィールドは、リストのどの個所から情報を戻すのかを指定します。

- **list_options** を AP_FIRST_IN_LIST に設定した場合、索引値を指定しているフィールドは無視され、戻りリストは最初から始まります。
- **list_options** を AP_LIST_INCLUSIVE に設定した場合、戻りリストは指定された索引値から始まります。
- **list_options** を AP_LIST_FROM_NEXT に設定した場合、戻りリストは指定された索引値の後の項目から始まります。

索引値は、戻り情報の開始点を指定します。 一度これを決めると、照会 verb のいくつかは、戻りリストに対する追加のフィルター処理オプションも提供します。 これらは、索引値とは独立して指定されます。 他の指定がなければ、戻りリストは、IBM の 6611 APPN MIB に従って並ぶことに注意してください。 （verb パラメーターが MIB テーブル項目に対してどのようにマップされるかについては、623ページの『付録A. IBM APPN MIB テーブル』を参照してください）。

戻される項目の数または埋められるバッファー・サイズが設定されます。 （両方とも設定されると、2 つの指定情報量のうちの小さいほうとともに verb が戻されます）。一般的に、アプリケーション・バッファー・サイズは戻すことのできる情報量を制限するので、ノード・オペレーター機能は、要求された情報を戻すのに必要な全バッファー・スペースの大きさ、およびこれが表す項目の総数を指示する追加情報を戻します。

1 つまたは複数単位の情報を戻すだけでなく、以下の verb は、異なるレベルの情報を戻すこともできます。 **list_options** フィールドに AP_DETAIL または AP_SUMMARY のいずれかを設定することによって、要約情報または詳細情報のいずれを戻すのかを **list_options** フィールドは指定します。 この **list_options** は、たとえば AP_DETAIL | AP_FIRST_IN_LIST というように、**OR 結合（論理和）**で指定されます。

- QUERY_DIRECTORY_LU
- QUERY_DLC
- QUERY_DLUR_LU
- QUERY_DLUR_PU
- QUERY_DOWNSTREAM_LU
- QUERY_ISR_SESSION
- QUERY_LOCAL_LU
- QUERY_LOCAL_TOPOLOGY
- QUERY_LS
- QUERY_LU_0_TO_3
- QUERY_MODE
- QUERY_MODE_DEFINITION
- QUERY_NN_TOPOLOGY_NODE

- QUERY_NN_TOPOLOGY_TG
- QUERY_PARTNER_LU
- QUERY_PARTNER_LU_DEFINITION
- QUERY_PORT
- QUERY_RTP_CONNECTION
- QUERY_SESSION
- QUERY_TP_DEFINITION

セッション限度 Verb

- CHANGE_SESSION_LIMIT
- INITIALIZE_SESSION_LIMIT
- RESET_SESSION_LIMIT

非送信請求指示

ノード情報を表示するアプリケーションは、これらの指示（変更が生じて、要約情報だけを戻すときに発行される）を使用して、照会 verb（詳細情報を戻すもの）をトリガーします。情報を受け取るように登録されたアプリケーションがある場合、名前付きイベントの非送信請求指示として、ノードは、以下にリストしているシグナルだけを作成します。したがって、情報がもう必要でなくなった場合、アプリケーションの登録を抹消しなければなりません。

- DLC_INDICATION
- DLUR_LU_INDICATION
- DLUS_INDICATION
- DOWNSTREAM_LU_INDICATION
- DOWNSTREAM_PU_INDICATION
- FOCAL_POINT_INDICATION
- ISR_INDICATION
- LOCAL_LU_INDICATION
- LOCAL_TOPOLOGY_INDICATION
- LS_INDICATION
- LU_0_TO_3_INDICATION
- MODE_INDICATION
- NN_TOPOLOGY_NODE_INDICATION
- NN_TOPOLOGY_TG_INDICATION
- PLU_INDICATION
- PORT_INDICATION
- PU_INDICATION
- REGISTRATION_FAILURE
- RTP_INDICATION

- SESSION_INDICATION

指示のために使用されるエントリー・ポイントは、以下のとおりです。

WinNOFRegisterIndicationSink
指示を受け取れるように登録する。

WinNOFUnregisterIndicationSink
指示を受け取らないように、登録を抹消する。

WinNOFGetIndication
指示を受け取る。

これらの指示は、ノード・オペレーター機能により登録済みであるすべての指示受信側に渡されます。指示を生成するコンポーネントが、指示を送信できない場合、そのコンポーネントは、自分が次に発行する指示上に **data_lost** 標識を設定します。指示上の **data_lost** フラグが AP_YES に設定されている場合、その後のデータ・フィールドはヌルに設定されている可能性があります。このフラグを使用して、その詳細個所が失われるという変更が生じたことをアプリケーションに通知し、アプリケーションに対して、適切な照会 verb を発行して応答するように指示を与えます。

機密保護 verb

以下の機密保護 verb により、LU-LU 検査または会話機密保護のためのパスワードの管理ができます。

- DEFINE_LU_LU_PASSWORD
- DEFINE_USERID_PASSWORD
- DELETE_LU_LU_PASSWORD
- DELETE_USERID_PASSWORD

APING verb

次の verb により、管理アプリケーションは、ネットワーク内のリモート LU を『ping』できます。

APING

CPI-C verb

以下の verb により、CPI-C 側の情報の定義、照会、削除ができます。

- DEFINE_CPIC_SIDE_INFO
- DELETE_CPIC_SIDE_INFO
- QUERY_CPIC_SIDE_INFO

Communications Server for Windows NT によって提供される CPI-C サポートに関する詳細については、*CPI-C Reference* を参照してください。

接続マネージャー verb

以下の verb を使用して、接続マネージャーを制御することができます。

- DISABLE_ATTACH_MANAGER
- ENABLE_ATTACH_MANAGER
- QUERY_ATTACH_MANAGER

DLC プロセス、ポート、およびリンク・ステーション

DLC プロセス

Communications Server は、複数の DLC プロセスを作成することができます。ノード・オペレーター機能 API で発行された START_DLC verb に応答して、各 DLC プロセスが Communications Server により作成されます。各 DLC は、特定のデータ・リンク・プロトコル（SDLC やトークンリングなど）を使用して、単一リンクもしくはリンクの集合を介した通信を担います。

各 DLC プロセスは、1 つまたは複数のポートを管理することができます。ポートの説明は下記のとおりです。

ポート

ポートは、ローカル・マシンの固有のアクセス・ポイント（MAC/SAP アドレス・ペアなど）を表し、DLC プロセスと関連があります。各 DLC は、1 つまたは複数のポートを保有することができます。ポートのタイプは、以下のいずれかです。

交換ポート
同時に活動状態にある 1 つまたは複数の隣接リンク・ステーションを保有できる。（これは *SNA APPN Architecture Reference* における定義と異なるので注意してください。）

非交換ポート
2 地点間リンク接続と分岐リンク接続の両方を保有できる。非交換リンク接続上の隣接リンク・ステーションは、ノード・オペレーター機能のコンポーネントによって定義しなければなりません。予測不能な結果を避けるために、分岐非交換リンクは、1 次と 2 次の関係がすべてのノード上に適切に定義されていることが必要です。

SATF ポート
トークンリングのような共用アクセス・トランスポート機能を使用する。これにより、この機構に接続するどの 2 つのリンク・ステーション間でも接続が可能となります。任意のリンク・ステーションを通じてリンクの活動化を開始できるように、トークンリング上で活動化されるすべてのリンク・ステーションの初期任務を、常に交渉可能となるように定義しなければなりません。

注: SATF ポートは、接続ネットワークと関連付けられることもあります。この場合、トポロジー更新を使用して、固有のアクセス・ポイントのアドレスを同報通信します。

リンク・ステーション

リンク・ステーションは、ポートと関連があり、隣接ノードへの接続を表します。ポートは、複数のリンク・ステーションをもつことができます。リンク・ステーションは、以下のように分類されます。

定義済みリンク・ステーション
明示的に定義された（DEFINE_LS verb を使用）リンク・ステーション。

動的リンク・ステーション
接続ネットワークを通じて動的接続の活動化の結果として作成されたリンク・ステーション（仮想経路指定ノード (VRN) とも呼ばれる）。

暗黙リンク・ステーション
交換ポートまたは SATF ポート上の未知のパートナー・ノードから受け取った呼出しの結果として作成されたリンク・ステーション。（このタイプのポートは、*SNA APPN Architecture Reference* には定義されていません。）

一時的リンク・ステーション
交換ポートまたは SATF ポート上で DLC インターフェースを介して CONNECT_IN を受け取ったときに作成されるリンク・ステーション。リモート・ノードのアイデンティティーが判別されたとき、削除されるか、動的または暗黙状態になります。

第3章 ノード・オペレーター機能エントリー・ポイント

この章では、ノード・オペレーター機能の verb のエントリー・ポイントについて説明します。

WinNOF()

この関数は、ノード・オペレーター機能のすべての verb に対する同期エントリー・ポイントを提供します。

構文

```
void WINAPI WinNOF(long vcb,
                   unsigned short vcb_size)
```

パラメーター

説明

vcb verb 制御ブロックへのポインター

vcb_size

verb 制御ブロックのバイト数

戻り

戻り値なし。 verb 制御ブロック内の **primary_rc** フィールドおよび **secondary_rc** フィールドはエラーを示します。

注釈

これは、ノード・オペレーター機能 API に対する主な同期エントリー・ポイントです。 この呼出しは、verb が完了するまでブロック化されます。

WinAsyncNOF()

この関数は、ノード・オペレーター機能のすべての verb に対する非同期エントリー・ポイントを提供します。

構文

```
HANDLE WINAPI WinAsyncNOF(HWND hwnd,
                          long vcb,
                          unsigned short vcb_size)
```

パラメーター

説明

hwnd 完了メッセージを受け取るウィンドウ・ハンドル

vcb verb 制御ブロックへのポインター

vcb_size
verb 制御ブロックのバイト数

戻り

戻り値は、非同期要求が正常に行われたかどうかを示します。 関数が正常に実行された場合、実際の戻り値は handle です。 関数が正常に実行されなかった場合、ゼロが戻されます。

注釈

各アプリケーション・スレッドは、このエントリー・ポイントの使用時に、未解決要求を一度に 1 つしかもつことができません。

非同期操作が完了したとき、アプリケーションのウィンドウ *hWnd* は、入力ストリングとして **WinAsyncNOF** をもつ戻りメッセージ **RegisterWindowMessage** を受け取ります。 *wParam* 引数には、元の関数呼出しによって戻された非同期タスク・ハンドルが含まれます。

関数が正常に戻った場合、操作の完了時または会話の取消し時に、**WinAsyncNOF()** メッセージがアプリケーションに通知されます。

注: WinNOFCancelAsyncRequest() も参照してください。

WinAsyncNOFEx()

この関数は、ノード・オペレーター機能のすべての verb に対する非同期エントリー・ポイントを提供します。 ブロック化呼出しではなく、エントリー・ポイントを使用すると、複数のセッションが同一スレッド上で処理できるようになります。

構文

```
HANDLE WINAPI WinAsyncNOFEx(HANDLE handle,
                            long vcb,
                            unsigned short vcb_size);
```

パラメーター

説明

ハンドル (handle)
アプリケーションが待機するイベントのハンドル

vcb verb 制御ブロックへのポインター

vcb_size
verb 制御ブロックのバイト数

戻り

戻り値は、非同期要求が正常に行われたかどうかを示します。 関数が正常に実行された場合、実際の戻り値は handle です。

注釈

このエントリー・ポイントは、Win32** API において WaitForMultipleObjects とともに使用するためのものです。 この関数の詳細については、Win32 API についてのプログラミングの資料を参照してください。

非同期操作が完了したとき、アプリケーションは、イベントのシグナル処理によって、それを通知されます。 イベントのシグナル処理の際に、どのエラー条件の場合でも、1 次戻りコードおよび 2 次戻りコードを検査してください。

注: WinNOFCancelAsyncRequest() も参照してください。

WinNOFCancelAsyncRequest()

この関数は、未解決の **WinAsyncNOF** ベースの要求を取り消します。

構文

```
int WINAPI WinNOFCancelAsyncRequest(HANDLE handle);
```

パラメーター

説明

ハンドル (handle)

指定パラメーター。取り消したい要求の handle を指定します。

戻り

戻り値は、非同期要求が取り消されたかどうかを示します。 値がゼロである場合、要求は取り消されました。 それ以外の値は次のとおりです。

WNOFALREADY

取り消された非同期要求がすでに完了済みであること、または、ハンドルが無効であったことを示すエラー・コード。

注釈

WinAsyncNOF 関数の 1 つによってすでに発行された非同期要求は、**WinNOFCancelAsyncRequest()** 呼出しの発行によって、完了前に取り消すことができ、初期設定機能によって戻された handle を *handle* に指定します。

非同期要求を取り消すと、アプリケーションの verb 制御ブロックに対する更新をすべて停止し、(ウィンドウ・メッセージまたはイベントのいずれかの方法により)、アプリケーションに対して verb が完了したことを通知するのを停止します。 基礎要求の取り消しは行いません。 実際に基礎要求を取り消すには、アプリケーションで適切な NOF verb (すなわち、 START_LS を 取り消すための STOP_LS) を発行しなければなりません。

もし、既存の非同期 **WinAsyncNOF** ルーチンの取り消しの実行が、エラー・コード WNOFALREADY で失敗したとすると、次の 2 つの事柄のうちの 1 つが起こります。 それは、元のルーチンがすでに完了していて、アプリケーションが結果の通知を処理済みであるか、元のルーチンがすでに完了済みであるのに、アプリケーションが完了通知を処理していないかのいずれかです。

注: WinAsyncNOF() も参照してください。

WinNOFCleanup()

この関数は、ノード・オペレーター機能 API からアプリケーションを終了し、登録を解除します。

構文

```
BOOL WINAPI WinNOFCleanup(void);
```

戻り

戻り値は、登録解除が正常に実行されたかどうかを示します。値がゼロでない場合、アプリケーションの登録解除は正常に実行されました。値ゼロが戻された場合、アプリケーションは登録解除されませんでした。

注釈

WinNOFCleanup() は、ノード・オペレーター機能 API からノード・オペレーター機能のアプリケーションの登録解除を指示するために使用されます。

WinNOFCleanup は、**WinNOFGetIndication** で待機中のスレッドをどれでも非ブロック化します。 これらは、WNOFNOTREG（アプリケーションが登録されていないため、指示を受け取れません）を戻します。 **WinNOFCleanup** は、すべての指示に対してアプリケーションの登録を抹消します。 **WinNOFCleanup** は未解決の verb（同期のものも非同期のものも）をエラー AP_CANCELLED とともに戻します。ただし、verb はノード内では完了します。

WinNOFStartup および **WinNOFCleanup** は、必ずしも使用する必要はありません。 ただし、アプリケーションでこれらの呼出しを使用する際には一貫性がなければなりません。 これらの関数については、両方とも使用するか、両方とも使用しないかにすべきです。

注: WinNOFStartup() も参照してください。

WinNOFStartup()

この関数により、アプリケーションは、ノード・オペレーター機能 API の必要なバージョンを指定し、この製品がサポートする API の該当バージョンを取り出すことができます。 この関数は、登録のためのノード・オペレーター機能 API 呼出しをさらに発行する前に、アプリケーションによって呼び出すことができます。

構文

```
int WINAPI WinNOFStartup(WORD wVersionRequired,
                         LPWNOFDATA nofdata);
```

パラメーター

説明

wVersionRequired

ノード・オペレーター機能 API サポートの必要なバージョンを指定します。 高位バイトは、マイナー・バージョン（改訂版）番号を指定し、低位バイトは、メジャー・バージョン番号を指定します。

nofdata

ノード・オペレーター機能 API のバージョンおよび API 実現の記述を戻します。

戻り

戻り値は、アプリケーションが正常に登録されたかどうか、ならびに、ノード・オペレーター機能 API 実現が指定されたバージョン番号をサポートできるかどうかを示します。 この値がゼロである場合、アプリケーションは正常に登録されており、指定されたバージョンをサポートすることができます。 ゼロ以外の戻り値は、以下の値の 1 つを戻します。

WNOFSYSERROR

基礎ネットワーク・サブシステムにおけるネットワーク通信が作動不能。

WNOFVERNOTSUPPORTED

ノード・オペレーター機能 API サポートの必要なバージョンが、この特定の実現によって提供されませんでした。

WNOFBADPOINTER

nofdata パラメーターに誤りがありました。

注釈

この呼出しは、API の今後のリリースと互換性をもたせるのに役立ちます。 現行バージョンは、1.0 です。

WinNOFStartup および **WinNOFCleanup** は、必ずしも使用する必要はありません。 ただし、アプリケーションでこれらの呼出しを使用する際には一貫性がなければなりません。 これらの関数については、両方とも使用するか、両方とも使用しないかにすべきです。

注: WinNOFCleanup() も参照してください。

WinNOFRegisterIndicationSink()

これにより、アプリケーションを、非送信要求指示を受け取るように登録することができます。

構文

```
BOOL WINAPI WinNOFRegisterIndicationSink(unsigned short indication_opcode,
                                         unsigned short queue_size,
                                         unsigned short *primary_rc,
                                         unsigned long  *secondary_rc);
```

パラメーター

説明

indication_opcode
登録の指示。

queue_size
キューに受け取られていない指示の数。 ゼロは、現行値を使用するということです。(初期値は 10 に設定されています)。 アプリケーションが登録する指示のすべてに対してキューは 1 つだけしか存在しません。

primary_rc
戻り値 : 1 次戻りコード

secondary_rc
戻り値 : 2 次戻りコード

戻り

この関数は、登録が正常に実行されたかどうかを示す値を戻します。 値がゼロでない場合、アプリケーションの登録は正常に実行されました。 値がゼロである場合、アプリケーションの登録は正常に実行されませんでした。

注釈

WinNOFRegisterIndicationSink は、タイプ **indication_opcode** の非送信請求指示を受け取るように、登録するために使用します。

アプリケーションは、受け取りたい指示の各タイプごとに **WinNOFRegisterIndicationSink** を発行しなければなりません。

注: **WinNOFUnregisterIndicationSink** および **WinNOFGetIndication** も参照してください。

WinNOFUnregisterIndicationSink()

これにより、アプリケーションは、非送信要求指示の受取りを停止することができます。

構文

```
BOOL WINAPI WinNOFUnregisterIndicationSink(unsigned short indication_opcode,
                                           unsigned short *primary_rc,
                                           unsigned long  *secondary_rc);
```

パラメーター

説明

indication_opcode
登録抹消の指示。

primary_rc
戻されるもの：1 次戻りコード。

secondary_rc
戻されるもの：2 次戻りコード。

戻り

この関数は、登録の取消しが成功したかどうかを示す値を戻します。 値がゼロでない場合、登録抹消は正常に実行されました。 値がゼロである場合、登録抹消は正常に実行されませんでした。

注釈

WinNOFUnregisterIndicationSink は、タイプ **indication_opcode** の非送信請求指示の受取りを停止するために使用します。

アプリケーションは、受取りを停止したい指示の各タイプごとに **WinNOFUnregisterIndicationSink** を発行しなければなりません。

注: **WinNOFRegisterIndicationSink** および **WinNOFGetIndication** も参照してください。

WinNOFGetIndication()

これにより、アプリケーションは非送信請求指示を受け取ることができます。

構文

```
int WINAPI WinNOFGetIndication(long buffer,
                               unsigned short *buffer_size,
                               unsigned long  timeout);
```

パラメーター

説明

buffer 指示を受け取るバッファーへのポインター。

buffer_size
バッファーのサイズ。 戻されるもの : 指示のサイズ。

timeout
指示待ち時間（ミリ秒単位)。

戻り

この関数は、指示が受信されたかどうかを示す値を戻します。

0 指示が戻されました。

WNOFTIMEOUT
指示待ち時間のタイムアウト。

WNOFSYSNOTREADY
基礎ネットワーク・サブシステムにおけるネットワーク通信が作動不能。

WNOFNOTREG
アプリケーションが指示を受信するように登録されていません。

WNOFBADSIZE
バッファーが小さすぎるため、指示を受け取ることができません。 十分な大きさのバッファーを設定して、**WinNOFGetIndication** 呼出しを再発行してください。指示のサイズは、**buffer_size** パラメーターに戻されます。

WNOFBADPOINTER
buffer パラメーターもしくは **buffer_size** パラメーターのいずれかが無効です。

WNOFSYSERROR
予期しないシステム・エラーが発生しました。

注釈

これはブロッキング呼出しで、以下の状況のいずれか 1 つになります。

- 指示が戻された場合
- タイムアウトが満了した場合
- アプリケーションが WinNOFCleanup 呼出しを発行する。

WinNOFGetIndication()

- プロダクトが停止した場合
- システム・エラーが発生した場合

注: **WinNOFRegisterIndicationSink** および **WinNOFUnregisterIndicationSink** も参照してください。

第4章 ノード構成 verb

以下の verb は、ノードの構成についての情報の定義および削除に使用されます。

DEFINE_ADJACENT_NODE

DEFINE_ADJACENT_NODE は、隣接ノード上の資源に対するノード・ディレクトリー・データベースに項目を追加します。

注: CP-CP セッションを使用する隣接ノードへの活動状態のパスが存在する場合、この verb は不要なので、発行する必要はありません。

この verb は、エンド・ノードで発行することができます。その場合、ノードの制御点がディレクトリーのルートに追加されます。

ノードの制御点 LU を定義するには、以下のフィールドに値を設定します。

- **cp_name** フィールドにノードの制御点の名前を指定する。
- ADJACENT_NODE_LU 構造を追加して、**fqlu_name** フィールドに制御点の名前を指定する。

ノード上の追加の LU が、ノードの制御点の子として、ディレクトリーに追加されます。DEFINE_ADJACENT_NODE を使用して、既存のノード定義に LU 定義を追加することもできます。 DELETE_ADJACENT_NODE verb の発行により、同様に LU が除去できます。 verb が処理の途中で失敗した場合、新しいディレクトリー項目はすべて除去され、ディレクトリーは、verb の発行前の状態のまま残されます。

VCB 構造

DEFINE_ADJACENT_NODE verb には、不定数の ADJACENT_NODE_LU オーバーレーが含まれています。 ADJACENT_NODE_LU 構造が、DEFINE_ADJACENT_NODE 構造の後に連結されています。

```
typedef struct define_adjacent_node
{
        unsigned short  opcode;                 /* verb operation code       */
        unsigned char   reserv2;                /* reserved                  */
        unsigned char   format;                 /* format                    */
        unsigned short  primary_rc;             /* primary return code       */
        unsigned long   secondary_rc;           /* secondary return code     */

unsigned char   cp_name[17];            /* CP name                      */

unsigned char   description[RD_LEN];    /* resource description         */

unsigned char   reserv3[19];            /* reserved                     */

unsigned short  num_of_lus;             /* number of LUs                */

unsigned char   reserv4[2];             /* reserved                     */
} DEFINE_ADJACENT_NODE;
typedef struct adjacent_node_lu
{
        unsigned char   wildcard_lu;            /* wildcard LU name          */
```

```
                                                /* indicator                */
        unsigned char   fqlu_name[17];          /* fully qualified LU name  */
        unsigned char   reserv1[6];             /* reserved                 */
} ADJACENT_NODE_LU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_ADJACENT_NODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

cp_name

隣接エンド・ノードの制御点の完全修飾名。これは、ノードがその XID（ノードが XID をサポートしている場合）上で送信する名前、およびノードにリンクするために DEFINE_LS 上で指定した隣接制御点の名前と一致させる必要があります。 この名前は、長さが 17 バイトで、右側が EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A のEBCDIC 文字ストリングで構成されます。各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

description

資源の説明（QUERY_DIRECTORY_LU で戻されます）。 これは、ローカルで表示可能な文字セットの 16 バイト（非ゼロ）ストリングです。 16 バイトすべてが有効です。

num_of_lus

DEFINE_ADJACENT_NODE VCB の後続の隣接 LU オーバーレーの数。

adjacent_node_lu.wildcard_lu

指定された LU 名がワイルドカード名であるかどうか（AP_YES または AP_NO）を示します。

adjacent_node_lu.fqlu_name

定義する LU 名。 この名前が完全修飾名でない場合、CP 名のネットワーク ID が想定されます。 この名前は、長さが 17 バイトで、右側が EBCDIC スペースで埋められます。 この名前は、EBCDIC ピリオドで連結された 1 つまたは 2 つのタイプ A のEBCDIC 文字ストリングから構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CP_NAME

AP_INVALID_LU_NAME
AP_INVALID_WILDCARD_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_CP_NAME

AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

secondary_rc
AP_MEMORY_SHORTAGE

AP_DIRECTORY_FULL

DEFINE_CN

DEFINE_CN は、接続ネットワーク（仮想経路指定ノードまたは VRN ともいいます）を定義します。このverb は、接続ネットワークのネットワーク修飾名をその伝送グループ (TG) 特性とともに提供します。 これは、この接続ネットワークにアクセスすることができるローカル・ポート名のリストも提供します。

DEFINE_CN を使用して、既存の接続ネットワークを再定義することができます。特に、別に DEFINE_CN を発行して、接続ネットワークにアクセスするポートのリストに新規ポートを追加することができます。 （DELETE_CN verb を 発行して、同様にポートを除去することができます。）

VCB 構造

```
typedef struct define_cn
{
        unsigned short  opcode;             /* verb operation code        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* primary return code        */
        unsigned long   secondary_rc;       /* secondary return code      */
        unsigned char   fqcn_name[17];      /* name of connection network */
        CN_DEF_DATA     def_data;           /* CN defined data            */
        unsigned char   port_name[8][8];
                                            /* port names                 */
} DEFINE_CN;
typedef struct cn_def_data
{
        unsigned char   description[RD_LEN];
                                            /* resource description       */
        unsigned char   num_ports;          /* number of ports on CN      */
        unsigned char   reserv1[16];        /* reserved                   */
        TG_DEFINED_CHARS tg_chars;          /* TG characteristics         */
} CN_DEF_DATA;
typedef struct tg_defined_chars
{
        unsigned char   effect_cap;         /* effective capacity         */
        unsigned char   reserve1[5];        /* reserved                   */
        unsigned char   connect_cost;       /* connection cost            */
        unsigned char   byte_cost;          /* byte cost                  */
        unsigned char   reserve2;           /* reserved                   */
        unsigned char   security;           /* security                   */
        unsigned char   prop_delay;         /* propagation delay          */
        unsigned char   modem_class;        /* modem class                */
        unsigned char   user_def_parm_1;    /* user-defined parameter 1   */
        unsigned char   user_def_parm_2;    /* user-defined parameter 2   */
        unsigned char   user_def_parm_3;    /* user-defined parameter 3   */
} TG_DEFINED_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_CN

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

fqcn_name

定義する接続ネットワークの完全修飾名（長さ 17 バイト）。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

def_data.description

資源の説明（QUERY_CN で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.num_ports

この接続ネットワークに関連のあるポートの数。 ポートは、各 DEFINE_CN verb ごとに 8 個まで、CN ごとに全体で 239 個まで（239 も含む）もつことができます。

def_data.tg_chars.effect_cap

実効容量の実際の単位。 この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。 実効容量の各単位は 300bps です。

def_data.tg_chars.connect_cost

接続時間当たりのコスト。有効な値は 0 から 255 までの整数です。ここで、0 は接続時間当たりの最も低いコストであり、255 は接続時間当たりの最も高いコストです。

def_data.tg_chars.byte_cost

バイト当たりのコスト。有効な値は 0 から 255 までの整数です。ここで、0 はバイト当たりの最も低いコストであり、255 はバイト当たりの最も高いコストです。

def_data.tg_chars.security

機密保護値は、以下にリストするとおりです。

AP_SEC_NONSECURE

セキュリティーが存在しません。

AP_SEC_PUBLIC_SWITCHED_NETWORK

この接続ネットワーク上を伝送されたデータが公共交換網を介して流れます。

AP_SEC_UNDERGROUND_CABLE

保護された地下回線でデータが伝送されます。

AP_SEC_SECURE_CONDUIT

回線は防護されていない保護コンジットです。

AP_SEC_GUARDED_CONDUIT

コンジットが物理的に漏えいしないように保護されています。

AP_SEC_ENCRYPTED
回線を暗号化します。

AP_SEC_GUARDED_RADIATION
回線が物理的および放射的に漏えいしないように保護されています。

def_data.tg_chars.prop_delay
波及遅延。信号がリンクの長さを通るのにかかる時間をマイクロ秒で表します。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。省略時値は以下のとおりです。

AP_PROP_DELAY_MINIMUM
波及遅延はありません。

AP_PROP_DELAY_LAN
480 マイクロ秒未満の遅延です。

AP_PROP_DELAY_TELEPHONE
480 マイクロ秒から 49 512 マイクロ秒の間の遅延です。

AP_PROP_DELAY_PKT_SWITCHED_NET
49 512 マイクロ秒から 245 760 マイクロ秒の間の遅延です。

AP_PROP_DELAY_SATELLITE
245 760 マイクロ秒を越える遅延です。

AP_PROP_DELAY_MAXIMUM
最大秒の波及遅延です。

def_data.tg_chars.modem_class
予約済みこのフィールドは常にゼロに設定しなくてはなりません。

def_data.tg_chars.user_def_parm_1
ユーザー定義のパラメーターで、範囲は 0 から 255

def_data.tg_chars.user_def_parm_2
ユーザー定義のパラメーターで、範囲は 0 から 255

def_data.tg_chars.user_def_parm_3
ユーザー定義のパラメーターで、範囲は 0 から 255

port_name
接続ネットワーク上で定義された最大 8 個のポート名の配列。 名前付きポートは、それぞれ DEFINE_PORT verb によってすでに定義済みでなければなりません。 各ポート名は、ローカルで表示可能な文字セットの 8 バイト・ストリングで、関連する DEFINE_PORT verb のポート名と一致しなければなりません。 新規ポート名を指定した別の DEFINE_CN を発行することにより、接続ネットワーク上に追加ポートを定義することができます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CN_NAME

AP_INVALID_NUM_PORTS_SPECIFIED
AP_INVALID_PORT_NAME
AP_INVALID_PORT_TYPE
AP_DEF_LINK_INVALID_SECURITY
AP_EXCEEDS_MAX_ALLOWED

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PORT_ACTIVE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_COS

DEFINE_COS は、サービス・クラス定義を追加します。 DEFINE_COS verb は、あらかじめ定義済みの COS の任意のフィールドを修正する場合にも使用することができます。

この定義はノードおよび TG 『行』を提供します。これらの行はノードの範囲およびTG 特性を、経路計算に使用する重みと関連付けます。 重みが小さいほど、経路はより好ましいものになります。

VCB 構造

DEFINE_COS verb には、不定数の **cos_tg_row** オーバーレーおよび **cos_node_row** オーバーレーが含まれます。 **cos_node_row** 構造が DEFINE_COS の後に連結され（しかも、重みの昇順に配列されます）、その後に **cos_tg_row** 構造が続けられます（これも重みの昇順に配列されます)。

```
typedef struct define_cos
{
        unsigned short  opcode;               /* verb operation code          */
        unsigned char   reserv2;              /* reserved                     */
        unsigned char   format;               /* format                       */
        unsigned short  primary_rc;           /* primary return code          */
        unsigned long   secondary_rc;         /* secondary return code        */
        unsigned char   cos_name[8];          /* class-of-service name        */
        unsigned char   description[RD_LEN];
                                              /* resource description         */
        unsigned char   transmission_priority;
                                              /* transmission priority        */
        unsigned char   reserv3[9];           /* reserved                     */
        unsigned char   num_of_node_rows;     /* number of node rows          */
        unsigned char   num_of_tg_rows;       /* number of TG rows            */
} DEFINE_COS;

typedef struct cos_node_row
{
        COS_NODE_STATUS minimum;              /* minimum                      */
        COS_NODE_STATUS maximum;              /* max                          */
        unsigned char    weight;              /* weight                       */
        unsigned char   reserv1;              /* reserved                     */
} COS_NODE_ROW;

typedef struct cos_node_status
{
        unsigned char   rar;                  /* route additional resistance  */
        unsigned char   status;               /* node status.                 */
        unsigned char   reserv1[2];           /* reserved                     */
} COS_NODE_STATUS;

typedef struct cos_tg_row
{
        TG_DEFINED_CHARS minimum;             /* minimum                      */
        TG_DEFINED_CHARS maximum;             /* maximum                      */
        unsigned char    weight;              /* weight                       */
        unsigned char   reserv1;              /* reserved                     */
} COS_TG_ROW;
```

```
typedef struct tg_defined_chars
{
        unsigned char   effect_cap;          /* effective capacity          */
        unsigned char   reserve1[5];         /* reserved                    */
        unsigned char   connect_cost;        /* cost per connect time       */
        unsigned char   byte_cost;           /* cost per byte               */
        unsigned char   reserve2;            /* reserved                    */
        unsigned char   security;            /* security                    */
        unsigned char   prop_delay;          /* propagation delay           */
        unsigned char   modem_class;         /* modem class                 */
        unsigned char   user_def_parm_1;     /* user-defined parameter 1    */
        unsigned char   user_def_parm_2;     /* user-defined parameter 2    */
        unsigned char   user_def_parm_3;     /* user-defined parameter 3    */
} TG_DEFINED_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_COS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

cos_name

サービス・クラス名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description

資源の説明（QUERY_COS で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

transmission_priority

伝送優先順位。以下の値のいずれか 1 つに設定されます。

AP_LOW
AP_MEDIUM
AP_HIGH
AP_NETWORK

num_of_node_rows

DEFINE_COS VCB の後続のノード行オーバーレーの数。最大は 8 です。

num_of_tg_rows

ノード行オーバーレーの後続の TG 行オーバーレーの数。最大は 8 です。各ノード行には、一連の最小ノード特性、一連の最大ノード特性、および重みが含まれます。ノードについての重みを計算するときに、各ノード行ごとに定義された最小特性および最大特性に対して、そのノードの特性が検査されます。続いて、ノードに最初のノード行の重みが割り当てられます。これは、すべてのノードの特性を、指定された限度内に制限します。ノード特性がリストされたノード行のいずれをも満足させない場合、そのノードはこの

COS には不適当であるとみなされ、無限大の重みが割り当てられます。ノード行は重みの昇順に連結されなければならないことに注意してください。

cos_node_row.minimum.rar
経路追加抵抗の最小値。値は 0 から 255 までの範囲でなければなりません。

cos_node_row.minimum.status
ノードの混雑状況の最小値を指定します。以下の値のいずれか 1 つになります。

AP_UNCONGESTED
ノードは混雑していません。

AP_CONGESTED
ISR セッションの数は、**isr_sessions_upper_threshold** よりも大きいです。

cos_node_row.maximum.rar
経路追加抵抗の最大値。値は 0 から 255 までの範囲でなければなりません。

cos_node_row.maximum.status
ノードの混雑状況の最大値を指定します。以下の値のいずれか 1 つになります。

AP_UNCONGESTED
ノードは混雑していません。

AP_CONGESTED
ISR セッションの数は、**isr_sessions_upper_threshold** よりも大きいです。

cos_node_row.weight
このノード行に関連する重み。値は 0 から 255 までの範囲でなければなりません。各 TG 行には、一連の最小 TG 特性、一連の最大 TG 特性、および重みが含まれています。TG についての重みを計算するときに、各 TG 行ごとに定義された最小特性および最大特性に対して、この TG 特性が検査されます。続いて、TG に最初の TG 行の重みが割り当てられます。これは、すべての TG の特性を、指定された限度内に制限します。TG 特性がリストされた TG 行のいずれをも満足させない場合、その TG はこの COS には不適当であるとみなされ、無限大の重みが割り当てられます。TG 行は重みの昇順に連結されなければならないことに注意してください。

cos_tg_row.minimum.effect_cap
実効容量の実際の単位の最小限度。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。実効容量の各単位は 300bps です。

cos_tg_row.minimum.connect_cost
接続時間当たりのコストの最小限度。有効な値は 0 から 255 までの整数です。ここで、0 は接続時間当たりの最も低いコストであり、255 は接続時間当たりの最も高いコストです。

cos_tg_row.minimum.byte_cost
バイト当たりのコストの最小限度。有効な値は 0 から 255 までの整数です。ここで、0 はバイト当たりの最も低いコストであり、255 はバイト当たりの最も高いコストです。

cos_tg_row.minimum.security
以下にリストする機密保護値の最小限度。

AP_SEC_NONSECURE
セキュリティーが存在しません。

AP_SEC_PUBLIC_SWITCHED_NETWORK
この接続ネットワーク上を伝送されたデータが公共交換網を介して流れます。

AP_SEC_UNDERGROUND_CABLE
保護された地下回線でデータが伝送されます。

AP_SEC_SECURE_CONDUIT
回線は防護されていない保護コンジットです。

AP_SEC_GUARDED_CONDUIT
コンジットが物理的に漏えいしないように保護されています。

AP_SEC_ENCRYPTED
回線を暗号化します。

AP_SEC_GUARDED_RADIATION
回線が物理的および放射的に漏えいしないように保護されています。

cos_tg_row.minimum.prop_delay
リンクの全長を信号が伝わるのにかかる時間（マイクロ秒単位）を表す伝播遅延の最小限度。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。省略時値は以下のとおりです。

AP_PROP_DELAY_MINIMUM
波及遅延はありません。

AP_PROP_DELAY_LAN
480 マイクロ秒未満の遅延です。

AP_PROP_DELAY_TELEPHONE
480 マイクロ秒から 49 512 マイクロ秒の間の遅延です。

AP_PROP_DELAY_PKT_SWITCHED_NET
49 512 マイクロ秒から 245 760 マイクロ秒の間の遅延です。

AP_PROP_DELAY_SATELLITE
245 760 マイクロ秒を越える遅延です。

AP_PROP_DELAY_MAXIMUM
最大秒の波及遅延です。

cos_tg_row.minimum.modem_class
予約済みこのフィールドは常にゼロに設定しなくてはなりません。

cos_tg_row.minimum.user_def_parm_1
0 から 255 までの範囲のユーザー定義のパラメーターの最小限度。

cos_tg_row.minimum.user_def_parm_2
0 から 255 までの範囲のユーザー定義のパラメーターの最小限度。

cos_tg_row.minimum.user_def_parm_3
0 から 255 までの範囲のユーザー定義のパラメーターの最小限度。

cos_tg_row.maximum.effect_cap
実効容量の実際の単位の最大限度。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。実効容量の各単位は 300bps です。

cos_tg_row.maximum.connect_cost
接続時間当たりのコストの最大限度。有効な値は 0 から 255 までの整数です。ここで、0 は接続時間当たりの最も低いコストであり、255 は接続時間当たりの最も高いコストです。

cos_tg_row.maximum.byte_cost
バイト当たりのコストの最大限度。有効な値は 0 から 255 までの整数です。ここで、0 はバイト当たりの最も低いコストであり、255 はバイト当たりの最も高いコストです。

cos_tg_row.maximum.security
以下のリストで説明されている機密保護値の最大限度。

AP_SEC_NONSECURE
セキュリティーが存在しません。

AP_SEC_PUBLIC_SWITCHED_NETWORK
この接続ネットワーク上を伝送されたデータが公共交換網を介して流れます。

AP_SEC_UNDERGROUND_CABLE
保護された地下回線でデータが伝送されます。

AP_SEC_SECURE_CONDUIT
回線は防護されていない保護コンジットです。

AP_SEC_GUARDED_CONDUIT
物理的に漏えいしないように保護されているコンジット。

AP_SEC_ENCRYPTED
回線を暗号化します。

AP_SEC_GUARDED_RADIATION
回線が物理的および放射的に漏えいしないように保護されています。

cos_tg_row.maximum.prop_delay
リンクの全長を信号が伝わるのにかかる時間（マイクロ秒単位）を表す伝播遅延の最大限度。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。省略時値は以下のとおりです。

AP_PROP_DELAY_MINIMUM
波及遅延はありません。

AP_PROP_DELAY_LAN
480 マイクロ秒未満の遅延です。

AP_PROP_DELAY_TELEPHONE
480 マイクロ秒から 49 512 マイクロ秒の間の遅延です。

AP_PROP_DELAY_PKT_SWITCHED_NET
49 512 マイクロ秒から 245 760 マイクロ秒の間の遅延です。

AP_PROP_DELAY_SATELLITE
245 760 マイクロ秒を越える遅延です。

AP_PROP_DELAY_MAXIMUM
最大秒の波及遅延です。

cos_tg_row.maximum.modem_class
予約済み。このフィールドは常にゼロに設定しなくてはなりません。

cos_tg_row.maximum.user_def_parm_1
0 から 255 までの範囲のユーザー定義のパラメーターの最大限度。

cos_tg_row.maximum.user_def_parm_2
0 から 255 までの範囲のユーザー定義のパラメーターの最大限度。

cos_tg_row.maximum.user_def_parm_3
0 から 255 までの範囲のユーザー定義のパラメーターの最大限度。

cos_tg_row.weight
この TG 行に関連する重み。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_COS_NAME

AP_INVALID_NUMBER_OF_NODE_ROWS
AP_INVALID_NUMBER_OF_TG_ROWS
AP_NODE_ROW_WGT_LESS_THAN_LAST
AP_TG_ROW_WGT_LESS_THAN_LAST

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_COS_TABLE_FULL

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DEFAULTS

DEFINE_DEFAULTS により、ユーザーはノードの省略時のアクションを定義または再定義することができます。

VCB 構造

```
typedef struct define_defaults
{
        unsigned short  opcode;            /* verb operation code      */
        unsigned char   reserv2;           /* reserved                 */
        unsigned char   format;            /* format                   */
        unsigned short  primary_rc;        /* primary return code      */
        unsigned long   secondary_rc;      /* secondary return code    */
        DEFAULT_CHARS   default_chars;     /* default information      */
} DEFINE_DEFAULTS;
typedef struct default_chars
{
        unsigned char   description[RD_LEN];
                                           /* resource description     */
        unsigned char   mode_name[8];      /* default mode name        */
        unsigned char   reserv[248];       /* reserved                 */
} DEFAULT_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DEFAULTS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

default_chars.description

資源の説明（QUERY_DEFAULTS で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

default_chars.mode_name

省略時値として機能するモード名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

verb が無効な省略時のモード（たとえば EBCDIC A でないもの）や、有効であるが定義されていないものを指定した場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_MODE_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DEFAULT_PU

DEFINE_DEFAULT_PU により、ユーザーは省略時の PU の任意のフィールドを定義、再定義、または修正することができます。 これにより、ユーザーは空白の PU 名を指定して、省略時の PU を削除することもできます。 TRANSFER_MS_DATA verb 上で PU 名が明示的に指定されていない場合、TRANSFER_MS_DATA で運ばれた管理サービス情報は、ホスト SSCP との省略時の PU のセッションで送られます。 このことに関する詳細については、547ページの『第14章 管理サービス verb』を参照してください。

VCB 構造

```
typedef struct define_default_pu
{
        unsigned short  opcode;           /* verb operation code       */
        unsigned char   reserv2;          /* reserved                  */
        unsigned char   format;           /* format                    */
        unsigned short  primary_rc;       /* primary return code       */
        unsigned long   secondary_rc;     /* secondary return code     */
        unsigned char   pu_name[8];       /* PU name                   */
        unsigned char   description[RD_LEN];
                                          /* resource description      */
        unsigned char   reserv3[16];      /* reserved                  */
} DEFINE_DEFAULT_PU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DEFAULT_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

省略時値として機能するローカル PU 名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description

資源の説明（QUERY_DEFAULT_PU で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DLC

DEFINE_DLC は、新しく DLC を定義するか、あるいは既存の DLC を修正します。この verb は、ノードを通じて固有である DLC 名および基本構造に連結されるいくかの DLC 特定データを定義します。 このデータは DLC の初期化中に使用され、その形式はDLC タイプ（トークンリングなど）によって決まっています。DEFINE_DLC verb を使用して、verb に追加された DLC 特定データのみ修正することができます。 これを行うには、DLC がリセット状態となるように、まず STOP_DLC verb を発行しなければなりません。

DLC 、ポートおよびリンク・ステーションの間の関係については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

VCB 構造

```
typedef struct define_dlc
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   dlc_name[8];        /* name of DLC                  */
        DLC_DEF_DATA    def_data;           /* DLC defined data             */
} DEFINE_DLC;

typedef struct dlc_def_data
{
        unsigned char   description[RD_LEN];
                                            /* resource description         */
        unsigned char   dlc_type;           /* DLC type                     */
        unsigned char   neg_ls_supp;        /* negotiable LS support        */
        unsigned char   port_types;         /* allowable port types         */
        unsigned char   reserv3[11];        /* reserved                     */
        unsigned short  dlc_spec_data_len;  /* Length of DLC specific data  */
} DLC_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DLC

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dlc_name

DLC の名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。OEM 装置については、この名前はメーカー固有のものです。有効値は、LAN、SDLC、AnyNet、X25 または TWINAX（8 バイトまでスペースで埋められます）。

def_data.description

資源の説明（QUERY_DLC で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.dlc_type

DLC のタイプ。 Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

def_data.neg_ls_supp

DLC が折衝可能リンク・ステーションをサポートするかどうかを指定します (AP_YES または AP_NO) 。 **dlc_type** が AP_TWINAX である場合、AP_NO だけがサポートされます。 **dlc_type** が AP_ANYNET である場合、AP_YES だけがサポートされます。

def_data.port_types

指定された **dlc_type** に対して認められたポート・タイプを指定します。この値は、以下の値の 1 つまたは複数が一緒に OR 結合（論理和）されたものに当たります。

AP_PORT_NONSWITCHED
AP_PORT_SWITCHED
AP_PORT_SATF

対応する DLC タイプに対してフィールドを設定するには、以下の表を使用します。

表 2. DLC タイプに対応するポート・タイプ

DLC タイプ	ポート・タイプ
AP_ANYNET	AP_PORT_SATF
AP_LLC2	AP_PORT_SATF
AP_OEM_DLC	AP_PORT_SWITCHED または AP_PORT_NONSWITCHED
AP_SDLC	AP_PORT_SWITCHED または AP_PORT_NONSWITCHED
AP_TWINAX	AP_PORT_NONSWITCHED
AP_X25	AP_PORT_SWITCHED または AP_PORT_NONSWITCHED

def_data.dlc_spec_data_len

このフィールドは常にゼロに設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_DLC_NAME

AP_INVALID_DLC_TYPE
AP_INVALID_PORT_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_DLC_ACTIVE

AP_INVALID_DLC_TYPE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DLUR_DEFAULTS

DEFINE_DLUR_DEFAULTS により、ユーザーは、省略時の従属型 LU サーバー (DLUS) およびバックアップの省略時の DLUS の定義、再定義、または取消しができます。 明示的に指定された関連のある DLUR をもたない PU に対する SSCP-PU 活動化の開始時に、DLUR は省略時の DLUS 名を使用します。 DLUS 名が DEFINE_DLUR_DEFAULTS verb 上で明示的に指定されていない場合、現行の省略時値 （またはバックアップ DLUS）が取り消されます。

VCB 構造

```
typedef struct define_dlur_defaults
{
        unsigned short  opcode;             /* verb operation code   */
        unsigned char   reserv2;            /* reserved              */
        unsigned char   format;             /* format                */
        unsigned short  primary_rc;         /* primary return code   */
        unsigned long   secondary_rc;       /* secondary return code */
        unsigned char   description[RD_LEN];
                                            /* resource description  */
        unsigned char   dlus_name[17];      /* DLUS name             */
        unsigned char   bkup_dlus_name[17]; /* Backup DLUS name      */
        unsigned char   reserv3;               /* reserved           */
        unsigned short  dlus_retry_timeout; /* DLUS Retry Timeout    */
        unsigned short  dlus_retry_limit;   /* DLUS Retry Limit      */
        unsigned char   reserv4[16];        /* reserved              */
} DEFINE_DLUR_DEFAULTS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DLUR_DEFAULTS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

description

資源の記述。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

dlus_name

省略時値として機能する DLUS ノードの名前。 すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドをすべてゼロに設定すると、現行の省略時の DLUS が取り消されます。

bkup_dlus_name

バックアップの省略時値として機能する DLUS ノードの名前。 すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC

文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドをすべてゼロに設定すると、現行のバックアップの省略時の DLUS が取り消されます。

dlus_retry_timeout

DLUS への接続の 2 回目の試行とその後の試行との間の秒単位の間隔。 最初の試行と 1 回めの再試行との間の間隔は、常に 1 秒です。 ゼロを指定すると、省略時値の 5 秒が使用されます。

dlus_retry_limit

DLUS への接続が最初に失敗したあとの再試行の最大回数。ゼロを指定すると、省略時値 3 が使用されます。 X'FFFF' を指定した場合、Communications Server は無限に再試行します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_DLUS_NAME

AP_INVALID_BKUP_DLUS_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DOWNSTREAM_LU

DEFINE_DOWNSTREAM_LU verb は PU 集信に使用されます。 PU 集信を使用すると、ダウンストリーム LU はアップストリーム・ホストと通信することができます。 これを行うには、Communications Server は各ダウンストリーム LU をホスト *LU* と呼ばれる従属型ローカル LU にマップします。

DEFINE_DOWNSTREAM_LU は、新規ダウンストリーム LU を定義しますが、既存の定義の修正には使用できません。 ダウンストリーム LU は、指定されたホスト LU（DEFINE_LU_0_TO_3 verb を使用して定義します）にマップされます。 ホスト LU は、LU プールに関して指定することもできます。

VCB 構造

```
typedef struct define_downstream_lu
{
        unsigned short  opcode;            /* verb operation code        */
        unsigned char   reserv2;           /* reserved                   */
        unsigned char   format;            /* format                     */
        unsigned short  primary_rc;        /* primary return code        */
        unsigned long   secondary_rc;      /* secondary return code      */
        unsigned char   dslu_name[8];      /* Downstream LU name         */
        DOWNSTREAM_LU_DEF_DATA def_data;   /* defined data               */
} DEFINE_DOWNSTREAM_LU;

typedef struct downstream_lu_def_data
{
        unsigned char   description[RD_LEN];
                                           /* resource description       */
        unsigned char   nau_address;       /* Downstream LU NAU address  */
        unsigned char   dspu_name[8];      /* Downstream PU name         */
        unsigned char   host_lu_name[8];   /* Host LU or Pool name       */
        unsigned char   reserv2[8];        /* reserved                   */
} DOWNSTREAM_LU_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DOWNSTREAM_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dslu_name

定義するダウンストリーム LU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.description

資源の説明（QUERY_DOWNSTREAM_LU で戻されます）。 このフィールドの長さは、4 バイトの倍数でなければならず、ゼロではありません。

def_data.nau_address
DOWNSTREAM LU のネットワーク・アドレス可能単位アドレス。 これは 1 から 255 までの範囲でなければなりません。

def_data.dspu_name
DOWNSTREAM PU 名（DEFINE_LS で指定したものです）。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.host_lu_name
ダウンストリーム LU がマップされるホスト LU またはホスト LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_DNST_LU_NAME

AP_INVALID_NAU_ADDRESS

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_PU_TYPE
AP_PU_NOT_DEFINED
AP_LU_ALREADY_DEFINED
AP_LU_NAU_ADDR_ALREADY_DEFD
AP_INVALID_HOST_LU_NAME
AP_LU_NAME_POOL_NAME_CLASH

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DOWNSTREAM_LU_RANGE

DEFINE_DOWNSTREAM_LU_RANGE verb は PU 集信に使用されます。 PU 集信を使用すると、ダウンストリーム LU はアップストリーム・ホストと通信することができます。 これを行うには、Communications Server は各ダウンストリーム LU をホスト *LU* と呼ばれる従属型ローカル LU にマップします。

DEFINE_DOWNSTREAM_LU_RANGE により、指定された NAU の範囲内で複数のダウンストリーム LU を定義することができます（しかし、既存の定義の修正には使用できません)。 ノード・オペレーターは、ベース名および NAU の範囲を提供します。 ベース名と NAU アドレスを結合することにより、LU 名が生成されます。

たとえば、ベース名 LUNME に NAU の 範囲 1 から 4 までを結合すると、LU は LUNME001、LUNME002、LUNME003、および LUNME004 と定義されることになります。 埋込み文字でない 5 文字未満のベース名を使用して、埋込み文字でない 8 文字未満の LU 名ができます。 Communications Server は、このような LU 名の右側を埋めて 8 文字にします。

ダウンストリーム LU は、指定されたホスト LU（DEFINE_LU_0_TO_3 verb を使用して定義します）にマップされます。

VCB 構造

```
typedef struct define_downstream_lu_range
{
        unsigned short  opcode;            /* verb operation code        */
        unsigned char   reserv2;           /* reserved                   */
        unsigned char   format;            /* format                     */
        unsigned short  primary_rc;        /* primary return code        */
        unsigned long   secondary_rc;      /* secondary return code      */
        unsigned char   dslu_base_name[5]; /* Downstream LU base name    */
        unsigned char   reserv3;           /* reserved                   */
        unsigned char   description[RD_LEN];
                                           /* resource description       */
        unsigned char   min_nau;           /* min NAU address in range   */
        unsigned char   max_nau;           /* max NAU address in range   */
        unsigned char   dspu_name[8];      /* Downstream PU name         */
        unsigned char   host_lu_name[8];   /* Host LU or pool name       */
        unsigned char   reserv4[8];        /* reserved                   */
} DEFINE_DOWNSTREAM_LU_RANGE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DOWNSTREAM_LU_RANGE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dslu_base_name

ダウンストリーム LU 名の範囲に対するベース名。 これは、5 バイト英数

字のタイプ A のEBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。 このベース名には、3 文字のタイプ A の EBCDIC の数字（NAU アドレスの 10 進数の値を表すもの）が付加され、NAU 範囲内の各 LU となります。

description

資源の説明（QUERY_DOWNSTREAM_LU で戻されます）。 このフィールドの長さは、4 バイトの倍数でなければならず、ゼロではありません。

min_nau

範囲内の最小 NAU アドレス。これは 1 から 255 までの値を取ることができます。

max_nau

範囲内の最大 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

dspu_name

DOWNSTREAM PU 名（DEFINE_LS で指定したものです）。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

host_lu_name

範囲内にあるすべてのダウンストリーム LU がマップされるホスト LU またはホスト LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_DNST_LU_NAME

AP_INVALID_NAU_ADDRESS

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_LU_NAME_POOL_NAME_CLASH

AP_LU_ALREADY_DEFINED
AP_INVALID_HOST_LU_NAME
AP_PU_NOT_DEFINED
AP_INVALID_PU_NAME
AP_INVALID_PU_TYPE
AP_LU_NAU_ADDR_ALREADY_DEFD

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_DSPU_TEMPLATE

この verb は PU 集信に使用されます。PU 集信を使用すると、ダウンストリーム LU はアップストリーム・ホストと通信することができます。これを行うには、Communications Server は各ダウンストリーム LU をホスト LU と呼ばれる従属型ローカル LU にマップします。DEFINE_DSPU_TEMPLATE は、ダウンストリーム・ワークステーションのグループ上にある、ダウンストリーム LU のためにテンプレートを定義します。このテンプレートは、ワークステーションが暗黙のリンク (事前に定義されていないもの) を通じて Communications Server に接続した際に、ダウンストリーム LU 用の位置定義を入れるために使用されます。これらのテンプレートは、DEFINE_PORT verb 上の **implicit_dspu_template** フィールドによって参照されます。DEFINE_DSPU_TEMPLATE は、新しいテンプレートを定義するため、または既存のテンプレートを変更するために使用できます (ただし、変更されたテンプレートの既存のインスタンスは影響されません)。

VCB 構造

```
typedef struct define_dspu_template
{
        unsigned short  opcode;              /* verb operation code       */
        unsigned char   reserv2;             /* reserved                  */
        unsigned char   format;              /* format                    */
        unsigned short  primary_rc;          /* primary return code       */
        unsigned long   secondary_rc;        /* secondary return code     */
        unsigned char   template_name[8];    /* name of template          */
        unsigned char   description[RD_LEN]; /* resource description      */
        unsigned char   reserv1[12];         /* reserved                  */
        unsigned short  max_instance;        /* Max active template       */
                                             /* instances                 */
        unsigned short  num_of_dslu_templates;
                                             /* number of DSLU templates  */
} DEFINE_DSPU_TEMPLATE;

typedef struct dslu_template
{
        unsigned char   min_nau;             /* min NAU address in range  */
        unsigned char   max_nau;             /* max NAU address in range  */
        unsigned char   reserv1[10];         /* reserved                  */
        unsigned char   host_lu[8];          /* host LU or pool name      */
} DSLU_TEMPLATE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_DSPU_TEMPLATE

format

VCB の形式を示します。先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

template_name

DSPU テンプレートの名前。(この名前は、PORT_DEF_DATA 上の

implicit_dspu_template フィールドで指定されている名前と対応しています。)in the **implicit_dspu_template** field on PORT_DEF_DATA). ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

description
資源の説明（QUERY_DSPU_TEMPLATE で戻されます)。この長さは、4 バイトの倍数でなければならず、ゼロではありません。

max_instance
これは、同時にアクティブにできるテンプレートのインスタンスの最大数です。この制限に達すると、MDS コンポーネントは、待ち行列上の最も古い項目を削除します。これは、0 から 65535 までの値をとることができます。0 とは、限度がないということです。

num_of_dslu_templates
DEFINE_DSPU_TEMPLATE VCB の後続の DSLU テンプレート・オーバーレーの数。これは 1 から 255 までの値を取ることができます。

dslu_template.min_nau
範囲内の最小 NAU アドレス。これは 1 から 255 までの値を取ることができます。

dslu_template.max_nau
範囲内の最大 NAU アドレス。これは 1 から 255 までの値を取ることができます。

dslu_template.host_lu
ダウンストリーム LU がマップされるホスト LU またはホスト LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_TEMPLATE_NAME

AP_INVALID_NAU_ADDRESS
AP_INVALID_NAU_RANGE

AP_CLASHING_NAU_RANGE
AP_INVALID_NUM_DSPU_TEMPLATES

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_HOST_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_FOCAL_POINT

Communications Server は、異なるフォーカル・ポイントといくつものタイプの関係をもつことができます。 DEFINE_FOCAL_POINT verb は、Communications Server が暗黙的関係をもつフォーカル・ポイントを定義します（この関係には、1 次 またはバックアップのタイプがあります）。 これらの関係、およびそのような関係を確立することができる方法は、以下のとおりです。 あるカテゴリーに対する管理サービス・フォーカル・ポイント (FP) と管理サービス・エントリー・ポイント (EP) との関係は、管理サービス機能のメッセージの交換時に確立されます。 以下のタイプの FP-EP 関係を確立することができます。

- 明示

  この関係を確立するのは、エントリー・ポイントを自分の制御範囲に割り当てる、フォーカル・ポイントの操作員です。 フォーカル・ポイントは、管理サービス (Management Services) 機能の交換を開始します。

- 暗黙 (1 次)

  この関係が確立されるのは、エントリー・ポイントの操作員がそのエントリー・ポイントを指定されたフォーカル・ポイントに割り当てたとき（たとえば、オペレーターが DEFINE_FOCAL_POINT verb を発行したとき）です。 エントリー・ポイントが管理サービス機能の交換を開始します。

- 暗黙（バックアップ）

  この関係が確立されるのは、エントリー・ポイントが明示 1 次フォーカル・ポイントまたは暗黙 1 次フォーカル・ポイントのいずれかを失ったときです。 エントリー・ポイントが管理サービス機能の交換を開始します。 バックアップ・フォーカル・ポイントのアイデンティティーについては、定義するか（DEFINE_FOCAL_POINT verb を使用します）、管理サービス機能の交換を経由して獲得することができます。

- 省略時

  この関係が確立されるのは、FP が操作員の介入を受けないで EP を獲得したときです。 FP は MS 機能の交換を開始します。 この関係は NN である EP にのみ適用されます。

- 定義域

  この関係が確立されるのは、実行中のネットワーク・ノード (NN) がエンド・ノード・エントリー・ポイントにフォーカル・ポイントのアイデンティティーを通知したときです。 定義域の関係は、エンド・ノードでのみ有効です。

- ホスト

  この関係は、管理サービス機能の交換を伴うことはなく、エントリー・ポイント・ノードからホストへの SSCP-PU セッションの構成によって確立されます。 これは、優先順位の最も低いフォーカル・ポイントの関係です。

各 DEFINE_FOCAL_POINT verb は、暗黙フォーカル・ポイント（1 次 またはバックアップのタイプがあります）の定義にのみ使用されます。
各 DEFINE_FOCAL_POINT verb は、特定の管理サービスのカテゴリーに対して発行されます。 このカテゴリー内で、DEFINE_FOCAL_POINT verb を使用して、以下のことを行うことができます。

- フォーカル・ポイントの定義。
- フォーカル・ポイント（またはバックアップ・フォーカル・ポイント）の置換。
- 現在活動中のフォーカル・ポイントの取消し。

DEFINE_FOCAL_POINT verb 上のフィールドは、以下のように使用されます。

ms_category は必ず埋めなければなりません。**fp_fqcp_name** フィールドと **ms_appl_name** フィールドの組合せによって、指定されたカテゴリーに対するフォーカル・ポイント（または、**backup** フィールドを AP_YES に設定している場合は、バックアップ・フォーカル・ポイント）を指定します。

新しいフォーカル・ポイントを提供しないで、現在活動中のフォーカル・ポイントを取り消すために、この verb が発行される場合、**fp_fqcp_name** フィールドおよび **ms_appl_name** フィールドは、すべてゼロに設定する必要があります。フォーカル・ポイントの定義または置換を行う DEFINE_FOCAL_POINT verb を受け取ると、Communications Server は、管理サービス機能の要求を送信することにより、指定されたフォーカル・ポイントとの暗黙 1 次フォーカル・ポイント関係を確立しようとします。Communications Server は、現在活動中のフォーカル・ポイントの取消しを行う DEFINE_FOCAL_POINT verb を受け取ると、フォーカル・ポイントに対して管理サービス機能の取消しメッセージを送信します。 現在活動中のフォーカル・ポイントの取消しを行うには、DELETE_FOCAL_POINT verb（AP_ACTIVE を指定）の使用をおすすめします。

VCB 構造

```
typedef struct define_focal_point
{
        unsigned short  opcode;           /* verb operation code          */
        unsigned char   reserv2;          /* reserved                     */
        unsigned char   format;           /* format                       */
        unsigned short  primary_rc;       /* primary return code          */
        unsigned long   secondary_rc;     /* secondary return code        */
        unsigned char   reserved;         /* reserved                     */
        unsigned char   ms_category[8];   /* management services category */
        unsigned char   fp_fqcp_name[17]; /* Fully qualified focal        */
                                          /* point CP name                */
        unsigned char   ms_appl_name[8];  /* Focal point application name */
        unsigned char   description[RD_LEN];
                                          /* resource description         */
        unsigned char   backup;           /* is focal point a backup      */
        unsigned char   reserv3[16];      /* reserved                     */
} DEFINE_FOCAL_POINT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_FOCAL_POINT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

ms_category

管理サービス・カテゴリー。 これは、SNA 管理サービスで説明するような、管理サービス・カテゴリーに対して体系的に定義された 4 バイトの値（右側は EBCDIC スペースで埋められます）の 1 つか、8 バイトのタイプ 1134 の EBCDIC 導入定義名のいずれかです。

fp_fqcp_name

フォーカル・ポイントの制御点の完全修飾名。 これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) フォーカル・ポイントを取り消す場合、このフィールドをすべてゼロに設定する必要があります。

ms_appl_name

フォーカル・ポイント・アプリケーション名。 これは、SNA 管理サービスで説明するような、管理サービス・アプリケーションに対して体系的に定義された 4 バイトの値（右側は EBCDIC スペースで埋められます）の 1 つか、8 バイトのタイプ 1134 の EBCDIC 導入定義名のいずれかです。 フォーカル・ポイントを取り消す場合、このフィールドをすべてゼロに設定する必要があります。

description

資源の説明（QUERY_FOCAL_POINT で戻されます)。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

backup

バックアップ・フォーカル・ポイントが定義されているかどうか（AP_YES または AP_NO）を指定します。現在活動中のフォーカル・ポイントの取消しが行われている場合、このフィールドは予約されています。現在活動中のフォーカル・ポイントの取消しを行うには、DELETE_FOCAL_POINT verb（AP_ACTIVE を指定）の使用をおすすめします。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_FP_NAME

AP_INVALID_CATEGORY_NAME

verb が正常に実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_REPLACED

AP_UNSUCCESSFUL

secondary_rc

AP_IMPLICIT_REQUEST_REJECTED

AP_IMPLICIT_REQUEST_FAILED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因、または Communications Server のフォーカル・ポイントへの接続が正常に実行されなかったことが原因で、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_INTERNAL_PU

DEFINE_INTERNAL_PU verb は、DLUR 実行のローカル PU を定義します。この verb は、ホストに直接接続されるローカル PU を定義するためには使用されません。この目的については、76ページの『DEFINE_LS』 を参照してください。

VCB 構造

```
typedef struct define_internal_pu
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   pu_name[8];          /* internal PU name             */
        INTERNAL_PU_DEF_DATA def_data;       /* defined data                 */
} DEFINE_INTERNAL_PU;

typedef struct internal_pu_def_data
{
        unsigned char   description[RD_LEN];
                                             /* resource description         */
        unsigned char   dlus_name[17];       /* DLUS name                    */
        unsigned char   bkup_dlus_name[17];  /* backup DLUS name             */
        unsigned char   pu_id[4];            /* PU identifier                */
        unsigned char   reserv2[8];          /* reserved                     */
} INTERNAL_PU_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_INTERNAL_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

定義する内部 PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.description

資源の説明（QUERY_DLUR_PU および QUERY_PU で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.dlus_name

DLUR が SSCP-PU 活動化の開始時に使用する DLUS ノードの名前。これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールド

をすべてゼロに設定すると、グローバルな省略時の DLUS（この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が、DLUR 開始の SSCP-PU 活動化で使用されます。

def_data.bkup_dlus_name
この PU に対するバックアップ DLUS として機能する DLUS ノードの名前。これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドをすべてゼロに設定すると、グローバルな省略時のバックアップ DLUS（この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が、この PU 用のバックアップとして使用されます。

def_data.pu_id
PU 識別子。 これは 4 バイトの 16 進数ストリングです。0 ビットから 11 ビットまではブロック番号が設定され、12 ビットから 31 ビットまでは PU を固有に識別する ID 番号が設定されます。これは、ホスト上に構成される **pu_id** と一致していなければなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_PU_ID
AP_INVALID_DLUS_NAME
AP_INVALID_BKUP_DLUS_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PU_ALREADY_DEFINED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_LOCAL_LU

DEFINE_LOCAL_LU verb は、指定された特性をもつローカル LU の定義、または、既存の LU の場合は、LU の **attach_routing_data** 特性の修正を要求します。DEFINE_LOCAL_LU を使用して既存の定義を修正した場合、**attach_routing_data** フィールド以外のパラメーターはどれも無視されます。

VCB 構造

```
typedef struct define_local_lu
{
        unsigned short  opcode;           /* verb operation code        */
        unsigned char   reserv2;          /* reserved                   */
        unsigned char   format;           /* format                     */
        unsigned short  primary_rc;       /* primary return code        */
        unsigned long   secondary_rc;     /* secondary return code      */
        unsigned char   lu_name[8];       /* local LU name              */
        LOCAL_LU_DEF_DATA def_data;       /* defined data               */
} DEFINE_LOCAL_LU;

typedef struct local_lu_def_data
{
        unsigned char   description[RD_LEN];
                                          /* resource description       */
        unsigned char   lu_alias[8];      /* local LU alias             */
        unsigned char   nau_address;      /* NAU address                */
        unsigned char   syncpt_support;   /* is sync-point supported?   */
        unsigned short  lu_session_limit; /* LU session limit           */
        unsigned char   reserv1;          /* reserved                   */
        unsigned char   reserv2;          /* reserved                   */
        unsigned char   pu_name[8];       /* PU name                    */
        unsigned char   reserv3[8];       /* reserved                   */
        unsigned char   attach_routing_data[128];
                                          /* routing data for           */
                                          /* incoming attaches          */
} LOCAL_LU_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LOCAL_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

削除するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.description

資源の説明（QUERY_LOCAL_LU で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.lu_alias
定義するローカル LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

def_data.nau_address
LU のネットワーク・アドレス可能単位のアドレスで、0 から 255 までの範囲でなければなりません。ゼロ以外の値は、LU が従属 LU であることを暗黙指定します。ゼロの値は LU が独立 LU であることを暗黙指定します。

def_data.syncpt_support
予約済みこのフィールドは必ず AP_NO に設定する必要があります。

def_data.lu_session_limit
LU によってサポートされるセッションの最大数。ゼロは限界がないということです。LU が独立型である場合、これはどんな値に設定することもできます。LU が従属型である場合、これは 1 に設定しなければなりません。

def_data.pu_name
この LU が使用する PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。このフィールドは、従属型 LU によってのみ使用され、独立型 LU の場合はすべて 2 進数のゼロに設定する必要があります。

def_data.attach_routing_data
接続経路指定データのタイプ。

AP_REGISTERED_OR_DEFAULT_ATTACH_MGR
このローカル LU におけるトランザクション・プログラム (TP) に対する接続到着の結果得られた DYNAMIC_LOAD_INDICATION が、この LU に対する DLI を受け取るように登録した接続マネージャー、または、この LU に対して接続マネージャーの登録がない場合は、省略時の接続マネージャーに送信されることを指定します。

AP_REGISTERED_ATTACH_MGR_ONLY
このローカル LU におけるトランザクション・プログラム (TP) に対する接続到着の結果得られた DYNAMIC_LOAD_INDICATION が、この LU に対する DLI を受け取るように登録した接続マネージャーにだけ送信されることを指定します。 この LU に対して接続マネージャーが何も登録されていない場合、その接続は拒否されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_NAU_ADDRESS
AP_INVALID_SESSION_LIMIT

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_PU_NOT_DEFINED

AP_INVALID_LU_NAME
AP_LU_ALREADY_DEFINED
AP_ALLOCATE_NOT_PENDING
AP_LU_ALIAS_ALREADY_USED
AP_PLU_ALIAS_CANT_BE_CHANGED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

secondary_rc

AP_MEMORY_SHORTAGE

DEFINE_LS

DEFINE_LS は新しいリンク・ステーション (LS) の定義または既存のリンク・ステーションの修正に使用されます。 この verb は、ノードを通じて固有な LS 名と、この LS で使用すべきポート名を提供します。 このポートは、DEFINE_PORT verb を使用して、すでに定義済みでなければなりません。 リンク特定データは、基本構造に連結されます。 DEFINE_LS は、リンク・ステーションがリセット状態にあり（STOP_LS の発行後）、DEFINE_LS 上で指定された **port_name** に LS の前回の定義以降変更がない場合、既存のリンク・ステーションの 1 つまたは複数のフィールドを修正するためにだけ使用されます。

DLC、ポート、およびリンク・ステーションの関係に関する詳細については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

LS_DEF_DATA の数多くのフィールドの設定値は、**adj_cp_type** フィールドの値によって決まります。

adj_cp_type が取り得る値は 6 つあり（**def_data.adj_cp_type** において詳細に説明します）、そのうちの以下の 4 つは隣接のタイプ 2.1 (APPN) ノードに対するリンクに使用されます。

- AP_NETWORK_NODE
- AP_END_NODE
- AP_APPN_NODE
- AP_BACK_LEVEL_LEN_NODE

そのうちの 4 つは、PU タイプ 2.0 のトラフィックを運ぶリンクのみに使用されます。

- AP_HOST_XID3
- AP_HOST_XID0
- AP_DSPU_XID
- AP_DSPU_NOXID

APPN ノードには 4 つのタイプがあり、そのちがいは以下のとおりです。

- APPN ネットワーク・ノードは、その XID3 に Network Name Control Vector (CV) をもち、並列 TG をサポートして、XID3 内のネットワーキング機能ビットを立て、リンク上で CP-CP セッションをサポートすることができます。
- APPN エンド・ノードは、その XID3 に Network Name CV をもち、並列 TG をサポートして、XID3 内のネットワーキング機能ビットを立てず、リンク上で CP-CP セッションをサポートすることができます。
- 上位レベル・ノードは、その XID3 内に Network Name CV をもち、並列 TG をサポートすることができ、XID3 内のネットワーキング機能ビットは立てず、CP-CP セッションをサポートしません。
- バックレベル・ノードは、その XID3 にNetwork Name CV をもたず、並列 TG をサポートせず、XID3 内のネットワーキング機能ビットを立てず、CP-CP セッションをサポートしません。

以下のフィールドはすべてのリンクに対して設定しなければなりません。

port_name

adj_cp_type

dest_address

auto_act_supp

disable_remote_act

limited_resource

link_deact_timer

ls_attributes

adj_node_id

local_node_id

target_pacing_count

max_send_btu_size

link_spec_data_len

ls_role

他のフィールドは以下のとおり設定しなければなりません。

- **adj_cp_type** を AP_NETWORK_NODE、AP_END_NODE、または AP_APPN_NODE に設定する場合、以下のフィールドを設定しなければなりません。

  adj_cp_name

  tg_number

  solicit_sscp_sessions

  dspu_services

  hpr_supported

  hpr_link_lvl_error

  default_nn_server

  cp_cp_sess_support

  use_default_tg_chars

  tg_chars

- **adj_cp_type** をAP_BACK_LEVEL_LEN_NODE に設定する場合、以下のフィールドを設定しなければなりません。

  adj_cp_name

  solicit_sscp_sessions

  dspu_services

  use_default_tg_chars

  tg_chars

- ローカル PU がリンクを使用する場合（**adj_cp_type** を AP_HOST_XID3 または AP_HOST_XID0 に設定するか、**solicit_sscp_sessions** を APPN ノードに対するリンク上で AP_YES に設定します）、次のフィールドを設定しなければなりません。

  pu_name
- ダウンストリーム PU がリンクを使用し、PU 集信によって実行される場合（（**dspu_services** を AP_PU_CONCENTRATION に設定します）、次のフィールドを設定しなければなりません。

  dspu_name
- ダウンストリーム PU がリンクを使用し、DLUR によって実行される場合（**dspu_services** を AP_DLUR に設定します）、以下のフィールドを設定しなければなりません。

  dspu_name

  dlus_name

  bkup_dlus_name

VCB 構造

```
typedef struct define_ls
{
        unsigned short  opcode;             /* verb operation code            */
        unsigned char   reserv2;            /* reserved                       */
        unsigned char   format;             /* format                         */
        unsigned short  primary_rc;         /* primary return code            */
        unsigned long   secondary_rc;       /* secondary return code          */
        unsigned char   ls_name[8];         /* name of link station           */
        LS_DEF_DATA     def_data;           /* LS defined data                */
} DEFINE_LS;
typedef struct ls_def_data
{
        unsigned char   description[RD_LEN];
                                            /* resource description           */
        unsigned char   port_name[8];       /* name of associated port        */
        unsigned char   adj_cp_name[17];    /* adjacent CP name               */
        unsigned char   adj_cp_type;        /* adjacent node type             */
        LINK_ADDRESS    dest_address;       /* destination address            */
        unsigned char   auto_act_supp;      /* auto-activate supported        */
        unsigned char   tg_number;          /* Pre-assigned TG number         */
        unsigned char   limited_resource;   /* limited resource               */
        unsigned char   solicit_sscp_sessions;
                                            /* solicit SSCP sessions          */
        unsigned char   pu_name[8];         /* Local PU name (reserved if     */

/* solicit_sscp_sessions is set   */
                                            /* to AP_NO)                      */
        unsigned char   disable_remote_act; /* disable remote activation flag */
        unsigned char   dspu_services;      /* Services provided for          */
                                            /* downstream PU                  */
        unsigned char   dspu_name[8];       /* Downstream PU name (reserved   */
                                            /* if dspu_services is set to     */
```

```
                                        /* AP_NONE or AP_DLUR)            */
        unsigned char   dlus_name[17];      /* DLUS name if dspu_services     */
                                        /* set to AP_DLUR                 */
        unsigned char   bkup_dlus_name[17]; /* Backup DLUS name if            */
                                        /* dspu_services set to AP_DLUR   */
        unsigned char   hpr_supported;      /* does the link support HPR?     */
        unsigned char   hpr_link_lvl_error; /* does link use link-level       */
                                        /* error recovery for HPR frms?   */
        unsigned short  link_deact_timer;   /* HPR link deactivation timer    */
        unsigned char   reserv1;            /* reserved                       */
        unsigned char   default_nn_server;  /* Use as deflt LS to NN server   */
        unsigned char   ls_attributes[4];   /* LS attributes                  */
        unsigned char   adj_node_id[4];     /* adjacent node ID               */
        unsigned char   local_node_id[4];   /* local node ID                  */
        unsigned char   cp_cp_sess_support; /* CP-CP session support          */
        unsigned char   use_default_tg_chars;
                                        /* Use the default tg_chars       */
        TG_DEFINED_CHARS tg_chars;          /* TG characteristics             */
        unsigned short  target_pacing_count;/* target pacing count            */
        unsigned short  max_send_btu_size;  /* max send BTU size              */
        unsigned char   ls_role;            /* link station role to use       */
                                        /* on this link                   */
        unsigned char   max_ifrm_rcvd;      /* max number of I-frames rcvd    */
        unsigned char   reserv3[34];        /* reserved                       */
        unsigned short  link_spec_data_len; /* length of link specific data   */
} LS_DEF_DATA;

typedef struct tg_defined_chars
{
        unsigned char   effect_cap;         /* effective capacity             */
        unsigned char   reserve1[5];        /* reserved                       */
        unsigned char   connect_cost;       /* connection cost                */
        unsigned char   byte_cost;          /* byte cost                      */
        unsigned char   reserve2;           /* reserved                       */
        unsigned char   security;           /* security                       */
        unsigned char   prop_delay;         /* propagation delay              */
        unsigned char   modem_class;        /* modem class                    */
        unsigned char   user_def_parm_1;    /* user-defined parameter 1       */
        unsigned char   user_def_parm_2;    /* user-defined parameter 2       */
        unsigned char   user_def_parm_3;    /* user-defined parameter 3       */
} TG_DEFINED_CHARS;

typedef struct link_address
{
        unsigned short  length;             /* length                         */
        unsigned short  reserve1;           /* reserved                       */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                        /* address                        */
} LINK_ADDRESS;

typedef struct link_spec_data
{
       unsigned char  link_data[SIZEOF_LINK_SPEC_DATA];
}  LINK_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

ls_name

リンク・ステーションの名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

フィールド **ls_name** を特殊値 $ANYNET$（ASCII ストリング）に設定すると、ノード・オペレーター機能に、これが、AnyNet DLC によって経路指定される独立型 LU セッションのトラフィックの送信先のリンク・ステーションであることを通知する効果があります。 AnyNet の経路指定が必要である場合、この名前のリンク・ステーションは、AnyNet DLC 上のポートで定義しなければなりません。

def_data.description

資源の説明（QUERY_LS、QUERY_PU で戻されます）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

def_data.port_name

このリンク・ステーションと関連したポート名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。この名前付きポートは、DEFINE_PORT verb によってすでに定義済みでなければなりません。

def_data.adj_cp_name

隣接制御点の 17 バイトの完全修飾名で、右側が EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。APPN ノードに対するリンクの場合、フィールド **tg_number** が 1 から 20 までの範囲の数に設定されているか、あるいは、フィールド **adj_cp_type** が AP_BACK_LEVEL_LEN_NODE に設定されているのでない場合、このフィールドをすべてゼロに設定することができます。このフィールドがすべてゼロに設定されている場合、XID 交換中に隣接ノードから受け取った名前に対する検査は行われません。このフィールドがすべてゼロに設定されているのではない場合、**adj_cp_type** が AP_BACK_LEVEL_LEN_NODE に設定されていない限り（この場合、隣接ノードを識別するのに使用されます）、XID 交換中に隣接ノードから受け取った名前に対する検査が行われます。

def_data.adj_cp_type

隣接ノード・タイプです。

AP_NETWORK_NODE

ノードが APPN ネットワーク・ノードであることを指定します。

AP_END_NODE
ノードが APPN エンド・ノードまたは上位レベル・ノードであることを指定します。

AP_APPN_NODE
ノードが APPN ネットワーク・ノード、APPN エンド・ノードまたは上位レベル・ノードであることを指定します。 ノード・タイプは XID 交換の間に認識されます。

AP_BACK_LEVEL_LEN_NODE
ノードが back_level_len ノードであることを指定します。 すなわち、このノードは XID で制御点名を送信しません。 独立型 LU セッションをサポートする AnyNet DLC を使用するリンクの場合、AP_BACK_LEVEL_LEN_NODE を指定しなければなりません。

AP_HOST_XID3
ノードがホストであり、Communications Server が形式 3 XID をもつノードからのポーリング XID に応答することを指定します。

AP_HOST_XID0
ノードがホストであり、Communications Server が形式 0 XID をもつノードからのポーリング XID に応答することを指定します。 従属型 LU セッションをサポートする AnyNet DLC を使用するリンクの場合、AP_HOST_XID0 を指定しなければなりません。

AP_DSPU_XID
ノードがダウンストリーム PU で、Communications Server がリンク活動化における XID 交換を含むことを指定します。

AP_DSPU_NOXID
ノードがダウンストリーム PU で、Communications Server がリンク活動化における XID 交換を含まないことを指定します。

注: VRN へのリンク・ステーションは常に動的であり、したがって定義されません。

def_data.dest_address.length
隣接ノードでの宛先リンク・ステーションのアドレスの長さ

def_data.dest_address.address
隣接ノードでのリンク・ステーションの宛先アドレス AnyNet DLC を使用するリンクの場合、**dest_address** は、隣接ノード ID または隣接制御点名を指定します。 隣接ノード ID を指定する場合、その長さは 4 バイトで、しかも、そのアドレスには 4 バイト 16 進数のノード ID（そのうちの 1 バイトがブロック ID で、3 バイトが PU ID）が含まれていなければなりません。隣接制御点名を指定する場合、その長さは 17 バイトで、そのアドレスには制御点名（EBCDIC で、右側が EBCDIC スペースで埋められます）が含まれていなければなりません。

def_data.auto_act_supp
セッションで必要な時に、リンクが自動的に活動化できるかどうかを指定し

ます （AP_YES または AP_NO)。 このリンクが APPN ノードに対するリンクではない場合、このフィールドは、常に AP_YES に設定することが可能で、他のパラメーターについての要件は何もありません。このリンクが APPN ノードに対するリンクである場合、リンクが CP-CP セッションもサポートする（**cp_cp_sess_support** を AP_YES に設定します）のであれば、このフィールドを AP_YES に設定することができず、事前割当ての TG 数もこのリンクに対して定義している（**tg_number** を 1 から 20 までの間の値に設定します）のであれば、このフィールドを AP_YES にのみ設定することができます。 **adj_cp_type** を AP_BACK_LEVEL_LEN_NODE に設定すると、これらの要件は必ず満たされます。この場合、**cp_cp_sess_support** および **tg_number** が無視されるからです。

def_data.tg_number

事前割当て TG 番号。 このフィールドは、隣接 APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。 **adj_cp_type** を AP_BACK_LEVEL_LEN_NODE に設定した場合、それも無視され、1 に設定されているものと想定されます。 隣接 APPN ノードに対するリンクの場合、これには、1 から 20 までの範囲の番号を設定しなければなりません。この番号は、リンクが活動化されたときそのリンクを示すのに使用されます。Communications Server は、このリンクの活動化中に、隣接ノードからこれ以外の番号を受け取ることはありません。事前割当て TG 番号の不一致が原因でリンク活動化が失敗することを避けるため、隣接リンク・ステーション上の隣接ノードは、同じ TG 番号を定義していなければなりません（事前割当て TG 番号を使用する場合)。事前割当て TG 番号を定義する場合、**adj_cp_name** も定義して（すべてゼロに設定することはできません）、さらに、**adj_cp_type** を AP_NETWORK_NODE または AP_END_NODE に設定しなければなりません。 ゼロを入力した場合、リンクが活動化されたときに、TG 番号が事前に割り当てられず、折衝されます。

def_data.limited_resource

このリンクを使用するセッションがない場合に、リンク・ステーションを非活動化するかどうかを指定します。 以下の値のいずれか 1 つに設定されます。

AP_NO

リンクは限定資源ではなく、自動的に非活動化されません。

AP_YES または AP_NO_SESSIONS

リンクは限定資源であり、このリンクを使用する活動セッションがない場合は自動的に非活動化されます。 限定資源のリンク・ステーションは、CP-CP セッション・サポートに対して構成することができます。 （これを行うには、このフィールドを AP_YES に設定して、**cp_cp_sess_support** を AP_YES に設定します。）この場合、CP-CP セッションがこのリンク上で立ち上げられると、Communications Server はこのリンクを限定資源として扱いません（しかも、このリンクを停止しません)。

AP_INACTIVITY

リンクは限定資源であり、このリンクを使用する活動セッションがない場合、または **link_deact_timer** フィールドによって指定された時間内にこのリンクを流れたデータがない場合は、自動的に非活動化されます。 非交換ポート上のリンク・ステーションは、限定資源として構成することができないことに注意してください。

def_data.solicit_sscp_sessions

AP_YES は、隣接ノードに対して、SSCP およびローカル制御点と従属型 LU との間のセッションの開始を要求します。 （この場合、**pu_name** を設定しなければなりません。）AP_NO は、このリンクでの SSCP とのセッションを要求しません。 このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。 隣接ノードをホストと定義した場合（**adj_cp_type** を AP_HOST_XID3 または AP_HOST_XID0 に設定します）、Communications Server は必ずそのホストに対して、SSCP およびローカル制御点と従属型 LU との間のセッションの開始を要求します(さらにもう一度、**pu_name** を設定しなければなりません)。

def_data.pu_name

隣接ノードをホストと定義した場合、または、**solicit_sscp_sessions** を APPN ノードに対するリンク上で AP_YES に設定した場合、このリンクを使用するローカル PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 隣接ノードをホストとしても、AP_YES に設定した **solicit_sscp_sessions** をもつ APPN ノードとしても定義しない場合、このフィールドは無視されます。

def_data.disable_remote_act

このリンクのリモート活動化をサポートするかどうかを

def_data.dspu_services

ローカル・ノードがリンクを介してダウンストリーム LU に提供するサービスを指定します。以下のいずれか 1 つに設定されます。

AP_PU_CONCENTRATION

ローカル・ノードは、ダウンストリーム PU に対して PU 集信を提供します。

AP_DLUR

ローカル・ノードはダウンストリーム PU に対する DLUR サービスを提供します。

AP_NONE

ローカル・ノードは、このダウンストリーム PU に対するサービスを何も提供しません。

dspu_name フィールドも AP_PU_CONCENTRATION または AP_DLUR に設定しなくてはなりません

隣接ノードがダウンストリーム PU として定義されている場合、このフィールドは AP_PU_CONCENTRATION または AP_DLUR に設

定されなければなりません (つまり、**adj_cp_type** は AP_DSPU_XID または AP_DSPU_NOXID に設定されます)。**solicit_sscp_sessions** が AP_NO に設定される場合、APPN ノードへのリンクで、フィールドを AP_PU_CONCENTRATION または AP_DLUR に設定することができます。隣接ノードがホストとして定義されている場合、このフィールドは無視されます。

def_data.dspu_name
ダウンストリーム PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

このフィールドは、**dspu_services** が AP_PU_CONCENTRATION または AP_DLUR に設定されている場合は設定しなければならず、そうでない場合は無視されます。

def_data.dlus_name
ダウンストリーム・ノードへのリンクが活動化された時点から DLUR が SSCP サービスを要求する DLUS のノードの名前。すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドをすべてゼロに設定すると、リンクが活動化された時点にグローバルな省略時の DLUS （この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が要求されます。**dlus_name** がゼロに設定されており、グローバルな省略時の DLUS がない場合は、DLUR はリンクが活動化されても SSCP 接続を開始しません。**dspu_services** が AP_DLUR に設定されていない場合、このフィールドは無視されます。

def_data.bkup_dlus_name
このダウンストリーム PU に対するバックアップとして機能する DLUS ノードの名前。すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドをすべてゼロに設定すると、グローバルな省略時のバックアップ DLUS（この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が、この PU 用のバックアップとして使用されます。**dspu_services** が AP_DLUR に設定されてない場合、このフィールドは無視されます。

def_data.hpr_supported
このリンクで HPR がサポートされるかどうかを指定します (AP_YES または AP_NO) 。このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。

def_data.hpr_link_lvl_error
リンク・レベル・エラー回復を使用して、このリンク上に HPR トラフィッ

クを送信すべきかどうかを指定します（AP_YES または AP_NO）。**hpr_supported** が AP_NO に設定されている場合、このパラメーターは無視されます。

def_data.link_deact_timer
限定資源のリンク非活動化タイマー（秒数）。

limited_resource を AP_INACTIVITY に設定した場合、このタイマーの持続時間中にこのリンクを通るデータがなければ、そのリンクは自動的に非活化されます。

ゼロを指定すると、省略時値 30 が使用されます。 ゼロ以外の場合、最小値は 5 です。(これより小さく設定されると、指定された値は無視され、5 が使用されます)。**limited_resource** を AP_NO に設定した場合、このパラメーターは予約されます。

def_data.default_nn_server
リンクがエンド・ノードによって自動的に活動化され、ネットワーク・ノード・サーバーへの CP-CP セッションをサポートできるかどうかを指定します（AP_YES または AP_NO)。このフィールドが有効となるためには、CP-CP セッションをサポートするようにリンクを定義しなければならないことに注意してください。

def_data.ls_attributes
隣接ノードについての詳細な情報を指定します。

def_data.ls_attributes[0]
ホスト・タイプです。

AP_SNA
標準 SNA ホスト

AP_FNA
FNA (VTAM-F) ホスト

AP_HNA
HNA ホスト

def_data.ls_attributes[1]
バック・レベル・ノードに対するリンクについての Network Name CV 抑止オプション。(**adj_cp_type** を AP_BACK_LEVEL_LEN_NODE または AP_HOST_XID3 に設定しない限り、このフィールドは無視されます)。

AP_NO
ネットワーク名 CV を XID3 に入れます。

AP_SUPPRESS_CP_NAME
ネットワーク名 CV を XID3 に入れません。

def_data.adj_node_id
隣接ノードのノード ID。これは 4 バイトの 16 進数ストリングです。隣接モードが T2.1 ノードであることを **adj_cp_type** が示している場合、このフィールドが非ゼロで、しかも、**adj_cp_type** が AP_BACK_LEVEL_LEN_NODE に設定されているか、隣接ノードが XID3

で Network Name CV を送信しないかのいずれか、でない限り、このフィールドは無視されます。**adj_cp_type** を AP_HOST_XID3 または AP_HOST_XID0 に設定した場合、このフィールドは必ず無視されます。**adj_cp_type** を AP_DSPU_XID に設定して、しかも、このフィールドが非ゼロである場合、このフィールドはダウンストリーム PU のアイデンティティーの検査に使用されます。**adj_cp_type** を AP_DSPU_NOXID に設定した場合、このフィールドは無視されるか（**dspu_services** が AP_PU_CONCENTRATION である場合）、DLUS に対するダウンストリーム PU を示すために使用されます（**dspu_services** が AP_DLUR である場合)。

def_data.local_node_id
このリンク・ステーションで XID で送信されるノード ID。これは 4 バイトの 16 進数ストリングです。このフィールドをゼロに設定した場合、**node_id** が XID 交換で使用されます。このフィールドをゼロ以外に設定すると、この LS の XID 交換の値を置き換えます。

def_data.cp_cp_sess_support
CP-CP セッションがサポートされるかどうかを指定します (AP_YES または AP_NO) 。 このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。**adj_cp_type** を AP_BACK_LEVEL_LEN_NODE に設定した場合、このフィールドも無視され、AP_NO に設定されているものと想定されます。

def_data.use_default_tg_chars
DEFINE_PORT verb 上で指定された省略時の TG 特性を使用するべきかどうかを指定します（AP_YES または AP_NO)。 これを AP_YES に設定した場合、**tg_chars** フィールドは無視されます。 このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。

def_data.tg_chars
TG の特性 (37ページの『DEFINE_CN』 を参照) このフィールドは、APPN ノードに対するリンクの場合のみ有効で、それ以外の場合は無視されます。

def_data.target_pacing_count
1 以上 32 767 以下の数値で、この TG 上での BIND に必要なペーシング・ウィンドウ・サイズを示します。 この数値は、固定バインド・ペーシングが実行される場合のみ有効です。一般的に、Communications Server はこの値を使用しません。

def_data.max_send_btu_size
このリンク・ステーションから送信することのできる最大 BTU サイズ。この値は、2 つのリンク・ステーション間で送信可能な最大 BTU サイズの折衝に使用されます。 リンクが HPR 可能でない場合、この値を 99 以上に設定しなければなりません。 リンクが HPR 可能である場合、この値を 768 以上に設定しなければなりません。

def_data.ls_role
このリンク・ステーションが前提とするべきリンク・ステーションの役割。AP_LS_NEG、AP_LS_PRI、または AP_LS_SEC のいずれか 1 つを指定し、

役割を折衝可能、1 次、2 次のなかから選択することができます。 このフィールドを AP_USE_PORT_DEFAULTS に設定すると、DEFINE_PORT verb で構成された値を選択することもできます。 **dlc_type** が AP_TWINAX である場合、AP_LS_SEC だけがサポートされます。 **dlc_type** が AP_ANYNET である場合（さらに **ls_name** が $ANYNET$）、AP_LS_PRI はサポートされません。

def_data.max_ifrm_rcvd
肯定応答の前に XID 送信側が受信できる I フレームの最大数。DEFINE_PORT からの省略時値を使用しなくてはならない場合はゼロに設定します。

def_data.link_spec_data_len
このフィールドは常にゼロに設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_DEF_LINK_INVALID_SECURITY

AP_INVALID_CP_NAME
AP_INVALID_LIMITED_RESOURCE
AP_INVALID_LINK_NAME
AP_INVALID_LS_ROLE
AP_INVALID_NODE_TYPE
AP_INVALID_PORT_NAME
AP_INVALID_AUTO_ACT_SUPP
AP_INVALID_PU_NAME
AP_INVALID_SOLICIT_SSCP_SESS
AP_INVALID_DLUS_NAME
AP_INVALID_BKUP_DLUS_NAME
AP_INVALID_NODE_TYPE_FOR_HPR
AP_INVALID_TARGET_PACING_COUNT
AP_INVALID_BTU_SIZE
AP_HPR_NOT_SUPPORTED
AP_INVALID_TG_NUMBER
AP_MISSING_CP_NAME

AP_MISSING_CP_TYPE
AP_MISSING_TG_NUMBER
AP_PARALLEL_TGS_NOT_SUPPORTED

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_LOCAL_CP_NAME

AP_DEPENDENT_LU_SUPPORTED
AP_DUPLICATE_DEST_ADDR
AP_INVALID_NUM_LS_SPECIFIED
AP_LS_ACTIVE
AP_PU_ALREADY_DEFINED
AP_DSPU_SERVICES_NOT_SUPPORTED
AP_DUPLICATE_TG_NUMBER
AP_TG_NUMBER_IN_USE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_LU_0_TO_3

この verb はタイプ 0、1、2 または 3 の LU を定義します。これにより、LU を LU プールに追加することができます。プールがまだ存在しない場合は、プールが追加されます。 この verb は既存の定義の修正には使用できません。

Communications Server は、ACTLU によって暗黙 LU タイプ 0、1、2 または 3 の定義をサポートします。 暗黙定義を削除することはできませんが、LU が非活動状態になったときに除去されます。 暗黙定義に関する情報を得るには、QUERY_LU_0_TO_3 を使用するか、LU_0_TO_3_INDICATION の登録をしてください。 **lu_name**、**pu_name**、および **nau_address** が正しく、**pool_name** がすべてゼロであれば（このとき LU は、最初の場所の操作員がそれを構成したかのように扱われます）、DEFINE_LU_0_TO_3 を使用して、暗黙 LU 定義を再定義することができます。

VCB 構造

```
typedef struct define_lu_0_to_3
{
        unsigned short  opcode;                 /* verb operation code        */
        unsigned char   reserv2;                /* reserved                   */
        unsigned char   format;                 /* format                     */
        unsigned short  primary_rc;             /* primary return code        */
        unsigned long   secondary_rc;           /* secondary return code      */
        unsigned char   lu_name[8];             /* LU name                    */
        LU_0_TO_3_DEF_DATA def_data;            /* defined data               */
} DEFINE_LU_0_TO_3;

typedef struct lu_0_to_3_def_data
{
        unsigned char   description[RD_LEN];
                                                /* resource description       */
        unsigned char   nau_address;            /* LU NAU address             */
        unsigned char   pool_name[8];           /* LU pool name               */
        unsigned char   pu_name[8];             /* PU name                    */
        unsigned char   priority;               /* LU priority                */
        unsigned char   lu_model;               /* LU model                   */
        unsigned char   reserv2[8];             /* reserved                   */
        unsigned char   app_spec_def_data[16];  /* Application Specified Data */
} LU_0_TO_3_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LU_0_TO_3

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

削除するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.description
資源の説明（QUERY_LU_0_TO_3 で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.nau_address
LU のネットワーク・アドレス可能単位のアドレスで、1 から 255 までの範囲でなければなりません。

def_data.pool_name
この LU が属する LU プールの名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。LU がプールに属していない場合、このフィールドはすべて 2 進ゼロに設定されます。現在、プールが存在しない場合、プールが作成されます。

def_data.pu_name
この LU が使用する PU 名（DEFINE_LS verb で指定したものです）。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

def_data.priority
ホストに送信するときの LU の優先順位。以下の値のいずれか 1 つに設定されます。

AP_NETWORKAP_HIGH
AP_MEDIUMAP_LOW

def_data.lu_model
LU の型式および番号。 以下の値のいずれか 1 つに設定されます。

AP_3270_DISPLAY_MODEL_2
AP_3270_DISPLAY_MODEL_3
AP_3270_DISPLAY_MODEL_4
AP_3270_DISPLAY_MODEL_5
AP_RJE_WKSTNAP_PRINTER
AP_UNKNOWN

AP_UNKNOWN 以外の値が指定され、ホスト・システムが SDDLU（自己定義従属型 LU）をサポートする場合、ホストでローカル LU を動的に定義するために、ノードは非送信請求 PSID NMVT 応答を生成します。

def_data.app_spec_def_data
アプリケーション指定定義データ。 このフィールドは、Communications Server には解釈されませんが、格納された後、QUERY_LU_0_TO_3 verb で戻されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LU_NAME

AP_INVALID_NAU_ADDRESS
AP_INVALID_PRIORITY

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_PU_TYPE
AP_PU_NOT_DEFINED
AP_LU_NAME_POOL_NAME_CLASH
AP_LU_ALREADY_DEFINED
AP_LU_NAU_ADDR_ALREADY_DEFD

システムが従属型 LU をサポートするように構築されていないために、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_INVALID_VERB

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_LU_0_TO_3_RANGE

この verb により、指定した NAU の範囲内で複数の LU を定義することができます。 ノード・オペレーターは、ベース名および NAU の範囲を提供します。 ベース名と NAU アドレスを結合することにより、LU 名が生成されます。 この verb は既存の定義の修正には使用できません。

たとえば、ベース名 LUNME に NAU の 範囲 1 から 4 までを結合すると、LU は LUNME001、LUNME002、LUNME003、および LUNME004 と定義されることになります。 埋込み文字でない 5 文字未満のベース名を使用して、埋込み文字でない 8 文字未満の LU 名ができます。 Communications Server は、このような LU 名の右側を埋めて 8 文字にします。

VCB 構造

```
typedef struct define_lu_0_to_3_range
{
        unsigned short  opcode;                 /* verb operation code        */
        unsigned char   reserv2;                /* reserved                   */
        unsigned char   format;                 /* format                     */
        unsigned short  primary_rc;             /* primary return code        */
        unsigned long   secondary_rc;           /* secondary return code      */
        unsigned char   base_name[5];           /* base name                  */
        unsigned char   reserv3;                /* reserved                   */
        unsigned char   description[RD_LEN];
                                                /* resource description       */
        unsigned char   min_nau;                /* minimum NAU address        */
        unsigned char   max_nau;                /* maximum NAU address        */
        unsigned char   pool_name[8];           /* LU pool name               */
        unsigned char   pu_name[8];             /* PU name                    */
        unsigned char   priority;               /* LU priority                */
        unsigned char   lu_model;               /* LU model                   */
        unsigned char   reserv4[8];             /* reserved                   */
        unsigned char   app_spec_def_data[16]; /* application specified data */
} DEFINE_LU_0_TO_3_RANGE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LU_0_TO_3_RANGE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

base_name

ベース LU 名。 これは、5 バイト英数字のタイプ A のEBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。 このベース名には、3 文字のタイプ A の EBCDIC の数字（NAU アドレスの 10 進数の値を表すもの）が付加され、NAU 範囲内の各 LU となります。

description

資源の説明（QUERY_LU_0_TO_3 で戻されます）。 このフィールドの長さは、4 バイトの倍数でなければならず、ゼロではありません。

min_nau

範囲内の最小 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

max_nau

範囲内の最大 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

pool_name

この LU が属する LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 LU がプールに属していない場合、このフィールドはすべて 2 進ゼロに設定されます。

pu_name

この LU が使用する PU 名（DEFINE_LS verb で指定したものです）。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

priority

ホストに送信するときの LU の優先順位。 以下の値のいずれか 1 つに設定されます。

AP_NETWORKAP_HIGH
AP_MEDIUMAP_LOW

lu_model

LU の型式および番号。 以下の値のいずれか 1 つに設定されます。

AP_3270_DISPLAY_MODEL_2
AP_3270_DISPLAY_MODEL_3
AP_3270_DISPLAY_MODEL_4
AP_3270_DISPLAY_MODEL_5
AP_RJE_WKSTNAP_PRINTER
AP_UNKNOWN

AP_UNKNOWN 以外の値が指定され、ホスト・システムが自己定義従属型 LU (SDDLU) をサポートする場合、ホストでローカル LU を動的に定義するために、ノードは非送信請求 PSID NMVT 応答を生成します。

app_spec_def_data

アプリケーション指定定義データ。 このフィールドは Communications Server では解釈されませんが、格納された後、QUERY_LU_0_TO_3 verb で戻されます（範囲内の各 LU に同じデータが戻されます）。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LU_NAME

AP_INVALID_NAU_ADDRESS
AP_INVALID_PRIORITY

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PU_NOT_DEFINED

AP_INVALID_PU_NAME
AP_INVALID_PU_TYPE
AP_LU_NAME_POOL_NAME_CLASH
AP_LU_ALREADY_DEFINED
AP_LU_NAU_ADDR_ALREADY_DEFD
AP_IMPLICIT_LU_DEFINED

システムが従属型 LU をサポートするように構築されていないために、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_INVALID_VERB

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_LU_POOL

この verb は、LU プールを定義したり、既存のプールに LU を追加するために使用されます。 追加する LU は、DEFINE_LU_0_TO_3 verb または DEFINE_LU_0_TO_3_RANGE verb のいずれかを使用して、すでに定義済みでなければなりません。 LU は同時に 1 つの LU プールにしか属すことができません。 指定した LU がすでにあるプールに属している場合、これらの LU は既存のプールから定義したプールに移動されます。 プール内の合計 LU 数に限度はありませんが、一度にプールに追加できる LU は 10 までです。

VCB 構造

```
typedef struct define_lu_pool
{
        unsigned short  opcode;               /* verb operation code        */
        unsigned char   reserv2;              /* reserved                   */
        unsigned char   format;               /* format                     */
        unsigned short  primary_rc;           /* primary return code        */
        unsigned long   secondary_rc;         /* secondary return code      */
        unsigned char   pool_name[8];         /* LU pool name               */
        unsigned char   description[RD_LEN];  /* resource description       */
        unsigned char   reserv3[4];           /* reserved                   */
        unsigned short  num_lus;              /* number of LUs to add       */
        unsigned char   lu_names[10][8];      /* LU names                   */
} DEFINE_LU_POOL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LU_POOL

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pool_name

これらの LU が属するプールの名前。 この名前は、8 バイト・ストリングで、右側がスペースで埋められます。 これは、EBCDIC ストリングであっても、ローカルで表示可能な文字セットのストリングであっても構いません。

description

資源の説明（QUERY_LU_POOL で戻されます）。 このフィールドの長さは、4 バイトの倍数でなければならず、ゼロではありません。

num_lus

0 から 10 までの範囲の追加する LU 数。

lu_names

プールに追加する LU の名前。 各名前とも 8 バイト英数字のタイプ A の EBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LU_NAME

AP_INVALID_NUM_LUS
AP_INVALID_POOL_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_LU_NAME_POOL_NAME_CLASH

AP_INVALID_POOL_NAME

システムが従属型 LU をサポートするように構築されていないために、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_INVALID_VERB

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_MODE

DEFINE_MODE verb は、特定のモード（またはセッション群）に割り当てる一連のネットワーキング特性を定義します。 この verb は、あらかじめ定義済みのモード上の任意のフィールドを修正するために使用することもできます。 SNASVCMG モードを再定義する場合、その **mode_name** および **cos_name** を修正することはできません。 CPSVCMG モードは再定義できません。

DEFINE_MODE verb は、不明モードがマップされる省略時の COS を定義するために使用することもできます。 これを行うには、**mode_name** をすべてゼロに設定します。 省略時の COS の初期設定は #CONNECTです。

注: 使用したいモードは、ネットワーク・ノード上、および、それを使用する可能性があればパートナー・ノード上で定義しなければなりませんが、その使用したいモードをすべてローカルに定義する必要はありません。 定義されていないモードを指定する ALLOCATE が発行された場合、ノードは DEFINE_DEFAULTS verb 上で指定されたモデルの省略時のモードの特性を使用します。 このようなモデルの指定がない場合、そのモデルに対しては、ブランク・モードの特性が使用されます。

VCB 構造

```
typedef struct define_mode
{
        unsigned short  opcode;                 /* verb operation code          */
        unsigned char   reserv2;                /* reserved                     */
        unsigned char   format;                 /* format                       */
        unsigned short  primary_rc;             /* primary return code          */
        unsigned long   secondary_rc;           /* secondary return code        */
        unsigned char   mode_name[8];           /* mode name                    */
        unsigned short  reserv3;                /* reserved                     */
        MODE_CHARS      mode_chars;             /* mode characteristics         */
} DEFINE_MODE;

typedef struct mode_chars
{
        unsigned char   description[RD_LEN]
                                                /* resource description         */
        unsigned short  max_ru_size_upp;        /* max RU size upper bound      */
        unsigned char   receive_pacing_win;     /* receive pacing window        */
        unsigned char   default_ru_size;        /* default RU size to maximize  */
                                                /* performance                  */
        unsigned short  max_neg_sess_lim;       /* max negotiable session limit */
        unsigned short  plu_mode_session_limit; /* LU-mode session limit        */
        unsigned short  min_conwin_src;         /* min source contention winner */
                                                /* sessions                     */
        unsigned char   cos_name[8];            /* class-of-service name        */
        unsigned char   cryptography;           /* cryptography                 */
        unsigned char   reserv1;                /* reserved                     */
        unsigned short  auto_act;               /* initial auto-activation count*/
        unsigned char   reserv2[6];             /* reserved                     */
} MODE_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_MODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

mode_name

モードの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 これがすべてゼロに設定された場合、省略時の COS が **mode_chars.cos_name** に設定され、他のすべての **mode_chars** フィールドは無視されます。

mode_chars.description

資源の説明（QUERY_MODE_DEFINITION および QUERY_MODE で戻されます)。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

mode_chars.max_ru_size_upp

このモードでセッション上を送受信される RU の最大サイズの上限。 この値は、セッション活動化中に最大 RU サイズを折衝するときに使用されます。 値の範囲は、256 以上 16384 以下です。**default_ru_size** が AP_YES に設定されている場合、この値は無視されます。

mode_chars.receive_pacing_win

このモードのセッションについてのセッション・ペーシング・ウィンドウ。固定ペーシングの場合、この値は受信ペーシング・ウィンドウを指定します。最適ペーシングの場合、この値は初期受信ウィンドウ・サイズとして使用されます。 隣接ノードが最適ペーシングをサポートしないことを指定していない場合、Communications Server は必ず最適ペーシングを使用することに注意してください。 値の範囲は 1 以上 63 以下です。値ゼロは認められません。

mode_chars.default_ru_size

最大 RU サイズの省略時の上限を使用するかどうかを指定します。 このパラメーターに AP_YES を指定した場合、**max_ru_size_upp** は無視され、最大 RU サイズの上限は、リンク BTU サイズから TH および RH のサイズを差し引いた値に設定されます。

AP_YES
AP_NO

mode_chars.max_neg_sess_lim

任意のローカル LU とパートナー LU との間のこのモードで使用できるセッションの最大数。 値ゼロが指定された場合、暗黙 CNOS 交換はありません。 値の範囲は、0 以上 32 767 以下です。

mode_chars.plu_mode_session_limit
このモードに対する省略時のセッション限度。これは、あるローカル LU とパートナー LU との間のこのモードでのセッションの数を制限します。この値は、CNOS（セッション数変更）交換の暗黙開始時に使用されます。値ゼロが指定された場合、暗黙 CNOS 交換はありません。値の範囲は、0 以上 32 767 以下です。

mode_chars.min_conwin_src
任意のあるローカル LU がこのモードを使用して活動化可能な、競合勝者セッションの最小数。この値は、CNOS（セッション数変更）交換の暗黙開始時に使用されます。値ゼロが指定された場合、暗黙 CNOS 交換はありません。値の範囲は、0 以上 32 767 以下です。

mode_chars.cos_name
このモードでのセッション活動化時に要求するサービス・クラスの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

mode_chars.cryptography
セッション・レベルの暗号化を使用しなければならないかどうか（AP_NONE または AP_MANDATORY）を指定します。

mode_chars.auto_act
このモードに対して自動的に活動化されるセッションの数を指定します。この値は、セッション数変更 (CNOS) 交換の暗黙開始時に使用されます。

範囲は 0 から 32767 です。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_COS_NAME

AP_CPSVCMG_ALREADY_DEFD
AP_INVALID_CNOS_SLIM
AP_INVALID_COS_SNASVCMG_MODE
AP_INVALID_DEFAULT_RU_SIZE
AP_INVALID_MAX_NEGOT_SESS_LIM
AP_INVALID_MAX_RU_SIZE_UPPER

AP_INVALID_MIN_CONWINNERS
AP_INVALID_MODE_NAME
AP_INVALID_SESSION_LIMIT
AP_INVALID_RECV_PACING_WINDOW
AP_INVALID_DEFAULT_RU_SIZES
AP_INVALID_SNASVCMG_MODE_LIMIT
AP_MODE_SESS_LIM_EXCEEDS_NEG

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_PARTNER_LU

DEFINE_PARTNER_LU verb は、ローカル LU とパートナー LU との間の LU-LU セッションのために、パートナー LU のパラメーターを定義します。 または、DEFINE_PARTNER_LU を使用して、パートナー LU についてすでに定義済みのすべてのパラメーター（**fqplu_name** および **plu_alias** 以外）を修正することができます。

VCB 構造

```
typedef struct define_partner_lu
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        PLU_CHARS       plu_chars;           /* partner LU characteristics    */
} DEFINE_PARTNER_LU;

typedef struct plu_chars
{
        unsigned char   fqplu_name[17];      /* fully qualified partner       */
                                             /* LU name                       */
        unsigned char   plu_alias[8];        /* partner LU alias              */
        unsigned char   description[RD_LEN];
                                             /* resource description          */
        unsigned char   plu_un_name[8];      /* partner LU uninterpreted name */
        unsigned char   preference           /* routing preference            */
        unsigned short  max_mc_ll_send_size; /* max MC send LL size           */
        unsigned char   conv_security_ver;   /* already_verified accepted?    */
        unsigned char   parallel_sess_supp;  /* parallel sessions supported?  */
        unsigned char   reserv2[8];          /* reserved                      */
} PLU_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_PARTNER_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

plu_chars.fqplu_name

パートナー LU の完全修飾名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

plu_chars.plu_alias

パートナー LU の別名。 ローカルで表示可能な文字セットで 8 バイト・

ストリングです。 関連のある別名をもたないパートナー LU の場合、このフィールドをすべてゼロに設定することがあります。

plu_chars.description

資源の説明（QUERY_PARTNER_LU および QUERY_PARTNER_LU_DEFINITION で戻されます）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

plu_chars.plu_un_name

パートナー LU の非解釈名。これは、8 バイトのタイプ A の EBCDIC 文字ストリングです。

plu_chars.max_mc_ll_send_size

パートナー LU でマップ式会話サービスによって送受信される LL レコードの最大サイズ。値の範囲は、1 から 32 767 までです（32 767 は、このフィールドをゼロに設定することによって指定します）。

plu_chars.preference

このパートナー LU に対するセッション活動化のために優先して使用される経路指定プロトコル。 このフィールドは、以下の値を取ることができます。

AP_NATIVE

ネイティブ (APPN) の経路指定プロトコルだけを使用します。

AP_NONNATIVE

非ネイティブ (AnyNet) プロトコルのみを使用します。

AP_NATIVE_THEN_NONNATIVE

ネイティブ (APPN) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、非ネイティブ (AnyNet) プロトコルを使用して、セッション活動化を再試行してください。

AP_NONNATIVE_THEN_NATIVE

非ネイティブ (AnyNet) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、ネイティブ (APPN) プロトコルを使用して、セッション活動化を再試行してください。

AP_USE_DEFAULT_PREFERENCE

ノードが開始したときに定義した省略時推奨値を使用します。（これは、QUERY_NODE により再呼出しすることができます。）

注: 非ネイティブ経路指定が意味をもつのは、ノード・オペレーター機能にとって AnyNet DLC が使用可能で、かつ定義済みの AnyNet リンク・ステーションがある場合だけです。（Defined_LS を参照してください。）

plu_chars.conv_security_ver

ローカル LU の代わりにパートナー LU が **user_ids** の妥当性検査を行うことを認可するかどうか、すなわちパートナー LU が、接続要求においてすでに検査済みの標識を設定することができるかどうか（AP_YES または AP_NO）を指定します。

plu_chars.parallel_sess_supp
パートナー LU が並列セッションをサポートするかどうか（AP_YES または AP_NO）を指定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_ANYNET_NOT_SUPPORTED

AP_DEF_PLU_INVALID_FQ_NAME
AP_INVALID_UNINT_PLU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PLU_ALIAS_CANT_BE_CHANGED

AP_PLU_ALIAS_ALREADY_USED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_PORT

DEFINE_PORT は新規ポートの定義または既存ポートの修正を行います。 このポートは指定された DLC に属しています。この DLC は DEFINE_DLC verb を使用して、すでに定義済みでなければなりません。 DEFINE_PORT verb は、ポート特定パラメーターおよび動的リンク・ステーションに関して使用する省略時の LS 特性とともに、ノードを通じて固有なポート名を提供します。ポート特定パラメーターは、基本構造に連結されます。 省略時の LS 特性は、ポート特定パラメーターのすぐ後に連結されます。

DEFINE_PORT は、ポートがリセット状態にあり（STOP_PORT の発行後）、DEFINE_PORT 上で指定した **dlc_name** に前回のポート定義以降変更がない場合、既存ポート上の 1 つまたは複数のフィールドを修正するのに使用することができます。

DLC 、ポートおよびリンク・ステーションの間の関係については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

VCB 構造

```
typedef struct define_port
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   port_name[8];        /* name of port                  */
        PORT_DEF_DATA   def_data;            /* port defined data             */
} DEFINE_PORT;

typedef struct port_def_data
{
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   dlc_name[8];         /* DLC name associated with port */
        unsigned char   port_type;           /* port type                     */
        unsigned char   reserv3[7];          /* reserved                      */
        unsigned long   port_number;         /* port number                   */
        unsigned short  max_rcv_btu_size;    /* max receive BTU size          */
        unsigned short  tot_link_act_lim;    /* total link activation limit   */
        unsigned short  inb_link_act_lim;    /* inbound link activation limit */
        unsigned short  out_link_act_lim;    /* outbound link activation      */
                                             /* limit                         */
        unsigned char   ls_role;             /* initial link station role     */
        unsigned char   reserv1[15];         /* reserved                      */
        unsigned char   implicit_dspu_template[8];
                                             /* reserved                      */
        unsigned char   reserv2[3];
                                             /* reserved                      */
        unsigned char   implicit_dspu_services;
                                             /* implicit links support DSPUs  */
        unsigned char   implicit_deact_timer;
                                             /* Implicit link HPR link        */
                                             /* deactivation timer            */
```

```
        unsigned short  act_xid_exchange_limit;
                                        /* act.  XID exchange limit     */
        unsigned short  nonact_xid_exchange_limit;
                                        /* nonact.  XID exchange limit  */
        unsigned char   ls_xmit_rcv_cap;    /* LS transmit-receive      */
                                        /* capability                   */
        unsigned char   max_ifrm_rcvd;      /* max number of I-frames that */
                                        /* can be received              */
        unsigned short  target_pacing_count; /* Target pacing count     */
        unsigned short  max_send_btu_size;  /* Desired max send BTU size */
        LINK_ADDRESS    dlc_data;           /* DLC data                 */
        LINK_ADDRESS    hpr_dlc_data;       /* HPR DLC data             */
        unsigned char   implicit_cp_cp_sess_support;
                                        /* Implicit links allow CP-CP   */
                                        /* sessions                     */
        unsigned char   implicit_limited_resource;
                                        /* Implicit links are limited   */
                                        /* resource                     */
        unsigned char   implicit_hpr_support;
                                        /* Implicit links support HPR   */
        unsigned char   implicit_link_lvl_error;
                                        /* Implicit links support HPR   */
                                        /* link-level error recovery    */
        unsigned char   retired1;           /* reserved                 */
        TG_DEFINED_CHARS default_tg_chars;
                                        /* Default TG chars             */
        unsigned char   discovery_supported;
                                        /* Discovery function           */
                                        /* supported?                   */
        unsigned short  port_spec_data_len; /* length of port spec data */
        unsigned short  link_spec_data_len; /* length of link spec data */
} PORT_DEF_DATA;

typedef struct link_address
{
        unsigned short  length;             /* length                   */
        unsigned short  reserve1;           /* reserved                 */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                        /* address                      */
} LINK_ADDRESS;

typedef struct tg_defined_chars
{
        unsigned char   effect_cap;         /* effective capacity       */
        unsigned char   reserve1[5];        /* reserved                 */
        unsigned char   connect_cost;       /* connection cost          */
        unsigned char   byte_cost;          /* byte cost                */
        unsigned char   reserve2;           /* reserved                 */
        unsigned char   security;           /* security                 */
        unsigned char   prop_delay;         /* propagation delay        */
        unsigned char   modem_class;        /* modem class              */
        unsigned char   user_def_parm_1;    /* user-defined parameter 1 */
        unsigned char   user_def_parm_2;    /* user-defined parameter 2 */
        unsigned char   user_def_parm_3;    /* user-defined parameter 3 */
} TG_DEFINED_CHARS;
```

```
typedef struct port_spec_data
{
       unsigned char  port_data[SIZEOF_PORT_SPEC_DATA];
}  PORT_SPEC_DATA;
typedef struct link_spec_data
{
       unsigned char  link_data[SIZEOF_LINK_SPEC_DATA];
}  LINK_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

port_name

定義するポート名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

def_data.description

資源の説明（QUERY_PORT で戻されます）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.dlc_name

関連した DLC 名で、ローカルで表示可能な文字セットの 8 バイト・ストリング。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 この名前付き DLC は、DEFINE_DLC verb により、すでに定義済みでなければなりません。

def_data.port_type

ポートによって使用されている回線のタイプを指定します。 この値は、以下の回線のタイプのいずれか 1 つに当たります。

AP_PORT_NONSWITCHED
AP_PORT_SWITCHED
AP_PORT_SATF

このフィールドを AP_PORT_SATF に設定した場合、**ls_role** を AP_LS_NEG に設定しなければならないことに注意してください。

def_data.port_number

ポート番号

def_data.max_rcv_btu_size

受信できる最大 BTU サイズ ポート上で暗黙 HPR 可能リンクがサポートされない場合、この値を 99 以上に設定しなければなりません。 ポート上

で暗黙 HPR 可能リンクがサポートされる場合、この値を 768 以上に設定しなければなりません。このポートが AnyNet DLC 用の場合は、65535 を使用しなければなりません。

def_data.tot_link_act_lim
合計リンク活動化の限界 これは同時に活動状態にできるリンク・ステーションの最大数を指定します。 これは **inb_link_act_lim** フィールドと **out_link_act_lim** フィールドの合計数より大きいか、または等しくなければなりません。 **port_type** を AP_PORT_NONSWITCHED に設定し、**ls_role** を AP_LS_NEG または AP_LS_SEC に設定する場合、このフィールドは 1 に設定しなければなりません。**ls_role** を AP_LS_PRI に設定した場合、このフィールドは 1 以上 256 以下の範囲になければなりません。 このポートが AnyNet DLC 用の場合は、65535 を使用しなければなりません。

def_data.inb_link_act_lim
インバウンド・リンク活動化の限界 これは、このポート上でのインバウンド活動化用に予約されるリンク・ステーションの数を指定します。 したがって、同時に活動状態にできるアウトバウンド・リンク・ステーションの最大数は、**def_data.tot_link_act_lim** - **def_data.inb_link_act_lim** です。 **port_type** を AP_PORT_NONSWITCHED に設定し、**ls_role** を AP_LS_NEG または AP_LS_PRI に設定する場合、このフィールドはゼロに設定しなければなりません。 **port_type** を AP_PORT_NONSWITCHED に設定し、**ls_role** を AP_LS_SEC に設定する場合、このフィールドはゼロまたは 1 に設定しなければなりません。 このポートが AnyNet DLC 用の場合は、ゼロを使用しなければなりません。

def_data.out_link_act_lim
アウトバウンド・リンク活動化の限界 これは、このポート上でのアウトバウンド活動化用に予約されるリンク・ステーションの数を指定します。 したがって、同時に活動状態にできるインバウンド・リンク・ステーションの最大数は、**def_data.tot_link_act_lim** - **def_data.out_link_act_lim** です。 **port_type** を AP_PORT_NONSWITCHED に設定し、**ls_role** を AP_LS_NEG に設定する場合、このフィールドはゼロに設定しなければなりません。 **ls_role** を AP_LS_PRI に設定する場合、このフィールドは **tot_link_act_lim** と等しくなければなりません。 **port_type** を AP_PORT_NONSWITCHED に設定し、**ls_role** を AP_LS_SEC に設定する場合、このフィールドはゼロまたは 1 に設定しなければなりません。 このポートが AnyNet DLC 用の場合は、ゼロを使用しなければなりません。

def_data.ls_role
リンク・ステーションの役割。 これは、折衝可能 (AP_LS_NEG)、1 次 (AP_LS_PRI)、または 2 次 (AP_LS_SEC) のいずれでも構いかまいません。 リンク・ステーションの役割によって、上記の **tot_act_lim**、**inb_link_act_lim**、および **out_link_act_lim** の各フィールドで指定する値の間の関係が決まります。 **port_type** を AP_PORT_SATF に設定した場合、**ls_role** を AP_LS_NEG に設定しなければならないことに注意してください。

def_data.implicit_dspu_template

DEFINE_DSPU_TEMPLATE verb によって定義されている DSPU テンプレートを指定します。このテンプレートは、このポート上で活動化されている暗黙リンクに対して、ローカル・ノードが PU 集信を提供する際に、定義のために使用されます。リンクが活動化された時点に、指定されたテンプレートが存在しない場合 (または既にインスタンスが限度数に達している場合) 、活動化は失敗します。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

def_data.implicit_dspu_services を AP_PU_CONCENTRATION に設定しないと、このフィールドは予約されます。

def_data.implicit.dspu_services

このポート上で活動化された暗黙リンク全体にわたって、ローカル・ノードによってダウンストリーム PU に提供されるサービスを指定します。 以下の値のいずれか 1 つに設定されます。

AP_DLUR

ローカル・ノードは、ダウンストリーム PU に DLUR サービスを提供します（DEFINE_DLUR_DEFAULTS verb によって構成済みの省略時 DLUS を使用して）。

AP_PU_CONCENTRATION

ローカル・ノードは、ダウンストリーム PU に対して PU 集信を提供します (また、フィールド **def_data.implicit_dspu_template** で指定された DSPU テンプレートが指定したように、位置定義を入れます)。

AP_NONE

ローカル・ノードは、このダウンストリーム PU に対するサービスを何も提供しません。

def_data.implicit_deact_timer

限定資源のリンク非活動化タイマー（秒数）。**implicit_limited_resource** が AP_YES または AP_NO_SESSIONS に設定されている場合で、このタイマーの期間中にデータがリンクを通らず、またリンクを使用しているセッションがない場合は、HPR 可能な暗黙リンクは自動的に非活動化されます。

implicit_limited_resource が AP_INACTIVITY に設定されている場合で、このタイマーの期間中にデータがこのリンクを通らなかった場合には、暗黙リンクは自動的に非活動化されます。

ゼロを指定すると、省略時値 30 が使用されます。 ゼロ以外の場合、最小値は 5 です。(さらに低い値を設定した場合、指定した値は無視され、5 が使用されます)。**implicit_limited_resource** が AP_NO に設定されていない場合、このパラメーターが予約されることに注意してください。

def_data.act_xid_exchange_limit

活動化 XID 交換の限界

def_data.nonact_xid_exchange_limit
非活動化 XID 交換の限度。

def_data.ls_xmit_rcv_cap
リンク・ステーションの送受信能力を指定します。 これは、両方向同時 (AP_LS_TWS)（二重または全二重とも呼ばれます）または両方向交互 (AP_LS_TWA) （半二重とも呼ばれます）のいずれかです。

def_data.max_ifrm_rcvd
肯定応答の送信前にローカル・リンク・ステーションが受信できる I フレームの最大数。 値の範囲は 1 以上 127 以下です。

def_data.target_pacing_count
1 以上 32 767 以下の数値で、この TG 上の BIND に対する望ましいペーシング・ウィンドウ・サイズを示します。 この数値は、固定バインド・ペーシングが実行される場合のみ有効です。 Communications Server は現在この値を使用していないことに注意してください。

def_data.max_send_btu_size
このリンク・ステーションから送信することのできる最大 BTU サイズ。この値は、2 つのリンク・ステーション間で送信可能な最大 BTU サイズの折衝に使用されます。 ポート上で暗黙 HPR 可能リンクがサポートされない場合、この値を 99 以上に設定しなければなりません。 ポート上で暗黙 HPR 可能リンクがサポートされる場合、この値を 768 以上に設定しなければなりません。

def_data.dlc_data.length
ポート・アドレスの長さ

def_data.dlc_data.address
ポート・アドレス

def_data.hpr_dlc_data.length
HPR ポート・アドレスの長さ

def_data.hpr_dlc_data.address
HPR ポート・アドレス。 これは現在、HPR リンクをサポートするとき、使用されます。このフィールドは、このポートを使用するリンク・ステーション上で交換される XID 3 上の X'61' 制御ベクトルの X'80' サブフィールドで Communications Server が送信する情報を指定します。これは、Communications Server が DLC に対して発行する ACTIVATE_PORT で渡されます。 この情報を HPR リンクをサポートするポートについて埋めるように要求することができる DLC もあります。

def_data.implicit_cp_cp_sess_support
このポート上にない暗黙リンク・ステーションに対して、CP-CP セッションを許可するかどうか（AP_YES または AP_NO）を指定します。

def_data.implicit_limited_resource
リンクを使用しているセッションがないとき、このポート上にない暗黙リンク・ステーションを非活動化すべきかどうかを指定します。 以下の値のいずれか 1 つに設定されます。

AP_NO
暗黙リンクは、限定資源ではなく、自動的には非活動化されません。

AP_YES または AP_NO_SESSIONS
暗黙リンクは、限定資源であり、活動状態のセッションがそれらを使用していないとき、自動的に非活動化されます。

AP_INACTIVITY
暗黙リンクは、限定資源であり、活動状態のセッションがそれらを使用していないとき、またはリンク上で **implicit_deact_timer** フィールドで指定された期間中にデータが流れなかったとき、自動的に非活動化されます。

def_data.implicit_hpr_support
暗黙リンク上で HPR をサポートすべきかどうかを指定します（AP_YES または AP_NO）。

def_data.implicit_link_lvl_error
リンク・レベル・エラー回復を使用して、暗黙リンク上に HPR トラフィックを送信すべきかどうかを指定します（AP_YES または AP_NO）。 **implicit_hpr_support** を AP_NO に設定した場合、このパラメーターは予約されることに注意してください。

def_data.default_tg_chars
TG 特性（41ページの『DEFINE_COS』を参照してください）。 これらは、このポートから外れた暗黙リンク・ステーションおよび **use_default_tg_chars** を指定する定義済みリンク・ステーションに対して使用されます。

def_data.discovery_supported
このポート上でディスカバリー機能を実行できるようにするかどうか指定します（AP_YES または AP_NO）。

def_data.port_spec_data_len
ACTIVATE_PORT シグナル上のポートに未変更のまま渡されるデータの長さ。このデータは、基本構造に連結する必要があります。

def_data.link_spec_data_len
このフィールドは常にゼロに設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PORT_NAME

AP_INVALID_DLC_NAME
AP_INVALID_PORT_TYPE
AP_INVALID_BTU_SIZE
AP_INVALID_LS_ROLE
AP_INVALID_LINK_ACTIVE_LIMIT
AP_INVALID_MAX_IFRM_RCVD
AP_INVALID_DSPU_SERVICES
AP_HPR_NOT_SUPPORTED
AP_DLUR_NOT_SUPPORTED
AP_INVALID_DISCOVERY_SUPPORT

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PORT_ACTIVE

AP_DUPLICATE_PORT_NUMBER

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_TP

DEFINE_TP verb は、ノード・オペレーター機能 TP 接続マネージャーがパートナー LU からの着信接続を処理するときに使用するトランザクション・プログラム (TP) 情報を定義します。 この verb は、すでに定義済みのトランザクション・プログラム上の 1 つまたは複数のフィールドを修正するために使用することもできます。(ただし、Communications Server で定義したトランザクション・プログラムの修正に使用することはできません。)

VCB 構造

```
typedef struct define_tp
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   tp_name[64];        /* TP name                       */
        TP_CHARS        tp_chars;           /* TP characteristics            */
} DEFINE_TP;
typedef struct tp_chars
{
        unsigned char   description[RD_LEN]
                                            /* resource description          */
        unsigned char   conv_type;          /* conversation type             */
        unsigned char   security_rqd;       /* security support              */
        unsigned char   sync_level;         /* synchronization level support */
        unsigned char   dynamic_load;       /* dynamic load                  */
        unsigned char   enabled;            /* is the TP enabled?            */
        unsigned char   pip_allowed;        /* program initialization        */
                                            /* parameters supported          */
        unsigned char   duplex_support;     /* duplex supported              */
        unsigned char   reserv3[9];
                                            /* reserved                      */
        unsigned short  tp_instance_limit ; /* limit on currently active TP  */
                                            /* instances                     */
        unsigned short  incoming_alloc_timeout;
                                            /* incoming allocation timeout   */
        unsigned short  rcv_alloc_timeout;  /* receive allocation timeout    */
        unsigned short  tp_data_len;        /* TP data length                */
        TP_SPEC_DATA    tp_data;            /* TP data                       */
} TP_CHARS;
typedef struct tp_spec_data
{
       unsigned char   pathname[256];       /* path and TP name              */
       unsigned char parameters[64];        /* parameters for TP             */
       unsigned char   queued;              /* queued TP                     */
       unsigned char load_type;             /* type of load-DETACHED/CONSOLE */
       unsigned char   dynamic_load         /* dynamic loading of TP enabled */
       unsigned char   reserved[5];         /* reserved                      */
} TP_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_TP

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

tp_name

定義するトランザクション・プログラムの名前。 これは 64 バイトの EBCDIC ストリングで、右側がスペースで埋められます。 Communications Server はこのフィールドの文字セットを検査しないことに注意してください。

tp_chars.description

資源の説明（QUERY_TP_DEFINITION および QUERY_TP で戻されます）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

tp_chars.conv_type

このトランザクション・プログラムがサポートする会話のタイプを指定します。

AP_BASIC
AP_MAPPEDAP_EITHER

tp_chars.security_rqd

トランザクション・プログラムを開始するのに、会話機密保護情報を必要とするかどうか（AP_NONE、AP_SAME、または AP_PGM）を指定します。

tp_chars.sync_level

トランザクション・プログラムがサポートする同期レベルを指定します。

AP_NONE

トランザクション・プログラムは、同期化レベル None をサポートします。

AP_CONFIRM_SYNC_LEVEL

トランザクション・プログラムは、同期化レベル Confirm をサポートします。

AP_EITHER

トランザクション・プログラムは、同期化レベル None または Confirm をサポートします。

AP_SYNCPT_REQUIRED

トランザクション・プログラムは、同期化レベル Sync-point をサポートします。

AP_SYNCPT_NEGOTIABLE

トランザクション・プログラムは、None、Confirm または Sync-point の同期レベルをサポートします。

tp_chars.dynamic_load
トランザクション・プログラムを動的にロードできるようにするかどうか（AP_YES または AP_NO）を指定します。

tp_chars.enabled
トランザクション・プログラムを正常に生成できるようにするかどうか（AP_YES または AP_NO）を指定します。 省略時値は AP_NO です。

tp_chars.pip_allowed
トランザクション・プログラムにプログラム初期設定パラメーター (PIP) の受信をできるようにするかどうか（AP_YES または AP_NO）を指定します。

tp_chars.duplex_support
トランザクション・プログラムを全二重か、または半二重に指定します。

AP_FULL_DUPLEX
トランザクション・プログラムが全二重であることを指定します。

AP_HALF_DUPLEX
トランザクション・プログラムが半二重であることを指定します。

AP_EITHER_DUPLEX
トランザクション・プログラムが半二重と全二重のいずれかを取ることができることを指定します。

tp_chars.tp_instance_limit
並行して活動状態のトランザクション・プログラムのインスタンスの数の限界値を 0 に指定すると、制限はなくなります。

tp_chars.incoming_alloc_timeout
着信接続が RECEIVE_ALLOCATE を待って、待ち行列内で待機する秒数を指定します。 ゼロは、タイムアウトなしの暗黙指定で、無期限に保持されます。

tp_chars.rcv_alloc_timeout
Attach を待っている間に、RECEIVE_ALLOCATE を待ち行列に入れる秒数を指定します。 ゼロは、タイムアウトなしの暗黙指定で、無期限に保持されます。

tp_chars.tp_data_len
実行依存性トランザクション・プログラムのデータ長

tp_spec_data
トランザクション・プログラムの開始時に接続マネージャーが使用する情報。この使用方法の詳細については、*Communications Server クライアント/サーバー 通信プログラミング* の接続マネージャーを参照してください。

tp_chars.tp_data.pathname
パスおよびトランザクション・プログラム名を指定します。

tp_chars.tp_data.parameters
トランザクション・プログラムに対するパラメーターを指定します。

tp_chars.tp_data.queued
トランザクション・プログラムを待ち行列に入れるかどうかを指定します。

tp_chars.tp_data.load_type
トランザクション・プログラムのロードの仕方を指定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_ADJACENT_NODE

DELETE_ADJACENT_NODE は、隣接ノード上の資源と関連のあるノード・ディレクトリー・データベースの項目を除去します。

ノードの制御点をその LU とともにディレクトリーから除去するには、**num_of_lus** をゼロに設定します。 **num_of_lus** がゼロでない場合、この verb は、ディレクトリーからノード LU を除去するために使用され、制御点の定義を変更することはありません。

verb がどのような理由で失敗しても、ディレクトリー項目は全く削除されません。

VCB 構造

DELETE_ADJACENT_NODE verb には、不定数の ADJACENT_NODE_LU オーバーレーが含まれています。 ADJACENT_NODE_LU 構造は、DELETE_ADJACENT_NODE 構造の後に連結されています。

```
typedef struct delete_adjacent_node
{
        unsigned short  opcode;          /* verb operation code        */
        unsigned char   reserv2;         /* reserved                   */
        unsigned char   format;          /* format                     */
        unsigned short  primary_rc;      /* primary return code        */
        unsigned long   secondary_rc;    /* secondary return code      */
        unsigned char   cp_name[17];     /* CP name                    */
        unsigned char   reserv3[3];      /* reserved                   */
        unsigned short  num_of_lus;      /* number of LUs              */
} DELETE_ADJACENT_NODE;

typedef struct adjacent_node_lu
{
        unsigned char   wildcard_lu;     /* wildcard LU name indicator */
        unsigned char   fqlu_name[17];   /* fully qualified LU name    */
        unsigned char   reserv1[6];      /* reserved                   */
} ADJACENT_NODE_LU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_ADJACENT_NODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

cp_name

隣接 LEN エンド・ノードの制御点の完全修飾名。 この名前は、長さが 17 バイトで、右側が EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

num_of_lus
削除する LU の数。 ノード定義全体を削除する場合、これをゼロに設定します。 この数は、DELETE_ADJACENT_NODE VCB に続く隣接 LU オーバーレーの数を表します。

adjacent_node_lu.wildcard_lu
指定された LU 名がワイルドカード名であるかどうか（AP_YES または AP_NO）を示します。

adjacent_node_lu.fqlu_name
削除する LU 名。 この名前が完全修飾名でない場合、CP 名のネットワーク ID が想定されます。 この名前は、長さが 17 バイトで、右側が EBCDIC スペースで埋められます。 この名前は、EBCDIC ピリオドで連結された 1 つまたは 2 つのタイプ A のEBCDIC 文字ストリングから構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CP_NAME

AP_INVALID_LU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_CP_NAME

AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_CN

DELETE_CN は、関連のあるポートがすべてリセットされている場合、接続ネットワーク制御ブロック用の記憶域を削除して解放します。 DELETE_CN は、接続ネットワークから選択したポートを削除するために使用することもできます。 これを行うには、ユーザーは **num_ports** フィールドを非ゼロに設定し、削除するポートのポート名を指定する必要があります。

VCB 構造

```
typedef struct delete_cn
{
        unsigned short  opcode;         /* verb operation code        */
        unsigned char   reserv2;        /* reserved                   */
        unsigned char   format;         /* format                     */
        unsigned short  primary_rc;     /* primary return code        */
        unsigned long   secondary_rc;   /* secondary return code      */
        unsigned char   fqcn_name[17];
                                        /* name of connection network */
        unsigned char   reserv1;        /* reserved                   */
        unsigned short  num_ports;      /* number of ports to delete  */
        unsigned char   port_name[8][8];
                                        /* names of ports to delete   */
} DELETE_CN;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_CN

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

fqcn_name

削除する接続ネットワーク名（長さ 17 バイト）。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

num_ports

接続ネットワーク上で削除するポートの数。 接続ネットワーク全体を削除する場合、これをゼロに設定する必要があります。

port_name

num_ports がゼロでない場合、削除するポートの名前。 各ポート名は、ローカルで表示可能な文字セットの 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **num_ports** フィールドがゼロである場合、このフィールドは予約されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CN_NAME

AP_INVALID_NUM_PORTS_SPECIFIED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_COS

サービス・クラス項目が SNA によって定義された省略時のサービス・クラスの 1 つではない場合、DELETE_COS はサービス・クラス項目を削除します。

VCB 構造

```
typedef struct delete_cos
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   cos_name[8];        /* class-of-service name        */
} DELETE_COS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_COS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

cos_name

サービス・クラス名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_COS_NAME_NOT_DEFD

AP_SNA_DEFD_COS_CANT_BE_DELETE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_DLC

DELETE_DLC は、DLC がリセットされている場合、その DLC と関連のあるすべてのポート、リンク・ステーション、接続ネットワーク伝送グループ (TG) を削除します。 すべての DLC 制御ブロックが削除され、その記憶域が解放されます。ノード・オペレーター機能は、DLC が正常に削除されたかどうかを示す応答を戻します。

関連のある PU をもつリンク・ステーションが削除される場合（このリンク・ステーションがその DLC に関連したものであるため）、この PU 上で定義されている LU もすべて削除されることに注意してください。

VCB 構造

```
typedef struct delete_dlc
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   dlc_name[8];        /* name of DLC                  */

} DELETE_DLC;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_DLC

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dlc_name

削除する DLC 名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_DLC_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_DLC_ACTIVE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_DOWNSTREAM_LU

DELETE_DOWNSTREAM_LU verb は特定のダウンストリーム LU を削除するために使用されます。

VCB 構造

```
typedef struct delete_downstream_lu
{
        unsigned short  opcode;          /* verb operation code       */
        unsigned char   reserv2;         /* reserved                  */
        unsigned char   format;          /* format                    */
        unsigned short  primary_rc;      /* primary return code       */
        unsigned long   secondary_rc;    /* secondary return code     */
        unsigned char   dslu_name[8];    /* Downstream LU name        */
} DELETE_DOWNSTREAM_LU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_DOWNSTREAM_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dslu_name

削除するダウンストリーム LU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_DOWNSTREAM_LU_RANGE

DELETE_DOWNSTREAM_LU_RANGE verb はダウンストリーム LU の範囲を削除するために使用されます。 ノード・オペレーターは、ベース名および NAU の範囲を提供します。 この範囲にある LU 名は、ベース名に NAU アドレスを結合して生成されます。

たとえば、ベース名 LUNME に NAU の 範囲 1 から 4 までを結合すると、LUNME001、LUNME002、LUNME003、および LUNME004 という LU を削除します。 埋込み文字でない 5 文字未満のベース名を使用して、埋込み文字でない 8 文字未満の LU 名ができます。

この verb は範囲内の LU をすべて削除します。 この範囲内の最初に LU が存在しない場合、verb は存在する次の LU について処理を続行します。 verb が失敗するのは、指定範囲内に LU が全く存在しない場合のみです。

VCB 構造

```
typedef struct delete_downstream_lu_range
{
        unsigned short  opcode;              /* verb operation code        */
        unsigned char   reserv2;             /* reserved                   */
        unsigned char   format;              /* format                     */
        unsigned short  primary_rc;          /* primary return code        */
        unsigned long   secondary_rc;        /* secondary return code      */
        unsigned char   dslu_base_name[5];   /* Downstream LU base name    */
        unsigned char   min_nau;             /* min NAU address in range   */
        unsigned char   max_nau;             /* max NAU address in range   */
} DELETE_DOWNSTREAM_LU_RANGE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_DOWNSTREAM_LU_RANGE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dslu_base_name

ダウンストリーム LU 名の範囲に対するベース名。 これは、5 バイト英数字のタイプ A のEBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。 このベース名には、3 文字のタイプ A の EBCDIC の数字（NAU アドレスの 10 進数の値を表すもの）が付加され、NAU 範囲内の各 LU となります。

min_nau

範囲内の最小 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

max_nau
範囲内の最大 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_NAU_ADDRESS

AP_INVALID_LU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_DSPU_TEMPLATE

この verb は特定の DSPU テンプレートを削除するために使用されます。

VCB 構造

```
typedef struct delete_dspu_template
{
        unsigned short  opcode;             /* verb operation code       */
        unsigned char   reserv2;            /* reserved                  */
        unsigned char   format;             /* format                    */
        unsigned short  primary_rc;         /* primary return code       */
        unsigned long   secondary_rc;       /* secondary return code     */
        unsigned char   template_name[8];   /* name of template          */

} DELETE_DSPU_TEMPLATE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_DSPU_TEMPLATE

format

VCB の形式を示します。先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

template_name

削除する DSPU テンプレートの名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_TEMPLATE_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_FOCAL_POINT

DELETE_FOCAL_POINT verb は、指定されたタイプおよびカテゴリーのフォーカル・ポイントを削除するために使用することができます。 フォーカル・ポイントのタイプに関する詳細については、66ページの『DEFINE_FOCAL_POINT』 を参照してください。 活動状態のフォーカル・ポイントを削除した場合、このフォーカル・ポイントは取り消されます。 活動状態のフォーカル・ポイント（どのタイプであっても）を取り消すには、AP_ACTIVE のタイプを指定します。 バックアップまたは暗黙フォーカル・ポイントが現在活動状態でないときに削除された場合（AP_BACKUP または AP_IMPLICIT を指定することによって削除します）、そのフォーカル・ポイントに関して格納されたすべての情報が単に除去されます。

DEFINE_FOCAL_POINT verb は、現在活動中のフォーカル・ポイントを取り消すために使用することもできることに注意してください。 この重複機能は、過去のバージョンとの互換性を保つためにあります。

VCB 構造

```
typedef struct delete_focal_point
{
        unsigned short  opcode;           /* verb operation code          */
        unsigned char   reserv2;          /* reserved                     */
        unsigned char   format;           /* format                       */
        unsigned short  primary_rc;       /* primary return code          */
        unsigned long   secondary_rc;     /* secondary return code        */
        unsigned char   reserved;         /* reserved                     */
        unsigned char   ms_category[8];   /* management services category */
        unsigned char   type;             /* type of focal point          */
} DELETE_FOCAL_POINT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_FOCAL_POINT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

ms_category

管理サービス・カテゴリー。 これは、SNA 管理サービスで説明するような、管理サービス・カテゴリーに対して体系的に定義された 4 バイトの値（右側は EBCDIC スペースで埋められます）の 1 つか、8 バイトのタイプ 1134 の EBCDIC 導入定義名のいずれかです。

type 削除するフォーカル・ポイントのタイプを指定します。 以下のタイプを取ることができます。

AP_ACTIVE

現在活動中のフォーカル・ポイント（どのタイプもあり得ます）が取り消されます。

AP_IMPLICIT
暗黙定義が除去されます。 現在活動中のフォーカル・ポイントが暗黙フォーカル・ポイントである場合、そのフォーカル・ポイントは取り消されます。

AP_BACKUP
バックアップの定義が除去されます。 現在活動中のフォーカル・ポイントがバックアップ・フォーカル・ポイントである場合、そのフォーカル・ポイントは取り消されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_TYPE

AP_INVALID_CATEGORY_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_INTERNAL_PU

DELETE_INTERNAL_PU verb は、DLUR 実行のローカル PU の削除を要求します。この verb が正常に実行されるのは、PU が活動中の SSCP-PU セッションをもたない場合のみです。

PU に関連する LU はすべて削除されます。

VCB 構造

```
typedef struct delete_internal_pu
{
        unsigned short  opcode;             /* verb operation code   */
        unsigned char   reserv2;            /* reserved              */
        unsigned char   format;             /* format                */
        unsigned short  primary_rc;         /* primary return code   */
        unsigned long   secondary_rc;       /* secondary return code */
        unsigned char   pu_name[8];         /* internal PU name      */
} DELETE_INTERNAL_PU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_INTERNAL_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

削除する内部 PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PU_NAME

AP_INVALID_PU_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_PU_NOT_RESET

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LOCAL_LU

DELETE_LOCAL_LU verb はローカル LU 定義の削除を要求します。

VCB 構造

```
typedef struct delete_local_lu
{
        unsigned short  opcode;          /* verb operation code       */
        unsigned char   reserv2;         /* reserved                  */
        unsigned char   format;          /* format                    */
        unsigned short  primary_rc;      /* primary return code       */
        unsigned long   secondary_rc;    /* secondary return code     */
        unsigned char   lu_name[8];      /* local LU name             */
} DELETE_LOCAL_LU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode
AP_DELETE_LOCAL_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name
削除するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LU_NAME

AP_CANT_DELETE_CP_LU

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LS

DELETE_LS は、リンク・ステーションが事前に定義済みで、リセットされていることを検査します。 これは、リンク・ステーション制御ブロックを除去して、このリンク・ステーションが正常に削除されたかどうかを示すノード・オペレーター機能からの応答を戻します。 このリンク・ステーションを使用する PU 上で定義された LU もすべて削除されることに注意してください。

VCB 構造

```
typedef struct delete_ls
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   ls_name[8];         /* name of link station          */
} DELETE_LS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_LS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

ls_name

削除するリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LINK_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_LS_ACTIVE

AP_INVALID_LINK_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LU_0_TO_3

この verb は特定の LU を削除するために使用されます。

VCB 構造

```
typedef struct delete_lu_0_to_3
{
        unsigned short  opcode;         /* verb operation code          */
        unsigned char   reserv2;        /* reserved                     */
        unsigned char   format;         /* format                       */
        unsigned short  primary_rc;     /* primary return code          */
        unsigned long   secondary_rc;   /* secondary return code        */
   unsigned char   lu_name[8];          /* LU name                      */
} DELETE_LU_0_TO_3;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_LU_0_TO_3

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

削除する LU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LU_0_TO_3_RANGE

この verb は ある範囲内の LU を削除するために使用されます。ノード・オペレーターは、ベース名および NAU の範囲を提供します。 ベース名と NAU アドレスを結合することにより、LU 名が生成されます。

たとえば、ベース名 LUNME に NAU の 範囲 1 から 4 までを結合すると、LUNME001、LUNME002、LUNME003、および LUNME004 という LU を削除することになります。 埋込み文字でない 5 文字未満のベース名を使用して、埋込み文字でない 8 文字未満の LU 名ができます。

この範囲内の LU はすべて削除されます。 この範囲内の最初に LU が存在しない場合、verb は存在する次の LU について処理を続行します。 指定範囲内に LU が全く存在しない場合、verb は失敗します。

VCB 構造

```
typedef struct delete_lu_0_to_3_range
{
        unsigned short  opcode;             /* verb operation code        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* primary return code        */
        unsigned long   secondary_rc;       /* secondary return code      */
        unsigned char   base_name[5];       /* base name                  */
        unsigned char   min_nau;            /* minimum NAU address        */
        unsigned char   max_nau;            /* maximum NAU address        */
        unsigned char   reserv3;            /* reserved                   */
} DELETE_LU_0_TO_3_RANGE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_LU_0_TO_3_RANGE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

base_name

ベース LU 名。 これは、5 バイト英数字のタイプ A のEBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。 このベース名には、3 文字のタイプ A の EBCDIC の数字（NAU アドレスの 10 進数の値を表すもの）が付加され、NAU 範囲内の各 LU となります。

min_nau

範囲内の最小 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

max_nau

範囲内の最大 NAU アドレス。 これは 1 から 255 までの値を取ることができます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_NAU_ADDRESS

AP_INVALID_LU_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_LU_NAME

AP_CANT_DELETE_IMPLICIT_LU

システムが従属型 LU をサポートするように構築されていないために、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_INVALID_VERB

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LU_POOL

この verb は、LU プールを削除したり、プールから LU を除去するために使用されます。 LU 名を指定しなければ、プール全体が除去されます。 この verb は、LU プール内の指定した LU、あるいは LU プール自体がもう存在しなくなったときに、正常に完了します。 この verb が失敗するのは、指定した LU が全く存在しないか、指定したプールに LU が全く存在しない場合のみです。

VCB 構造

```
typedef struct delete_lu_pool
{
        unsigned short  opcode;             /* verb operation code        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* primary return code        */
        unsigned long   secondary_rc;       /* secondary return code      */
        unsigned char   pool_name[8];       /* LU pool name               */
        unsigned short  num_lus;            /* number of LUs to add       */
        unsigned char   lu_names[10][8];    /* LU names                   */
} DELETE_LU_POOL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_LU_POOL

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pool_name

LU プールの名前。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

num_lus

lu_names リストで指定する LU の数。

lu_names

除去する LU の名前。 各名前とも 8 バイト英数字のタイプ A の EBCDIC ストリング（文字で始めます）で、右側が EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_POOL_NAME

AP_INVALID_LU_NAME
AP_INVALID_NUM_LUS

システムが従属型 LU をサポートするように構築されていないために、verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_INVALID_VERB

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_MODE

DELETE_MODE verb はモード定義の削除を要求します。 CPSVCMG モード、SNASVCMG モード、および他の標準 SNA モードについての省略時の定義は削除されません。

VCB 構造

```
typedef struct delete_mode
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   mode_name[8];        /* mode name                    */
} DELETE_MODE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_MODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

mode_name

モードの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_CP_OR_SNA_SVCMG_UNDELETABLE

AP_MODE_UNDELETABLE
AP_DEL_MODE_DEFAULT_SPCD
AP_MODE_NAME_NOT_DEFD

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_PARTNER_LU

DELETE_PARTNER_LU は、パートナー LU の定義の削除を要求します。

VCB 構造

```
typedef struct delete_partner_lu
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   fqplu_name[17];     /* fully qualified partner      */
                                            /* LU name                      */
} DELETE_PARTNER_LU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_PARTNER_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

fqplu_name

パートナー LU の完全修飾名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PLU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_PORT

DELETE_PORT は、ポートがリセットされている場合、そのポートに関連するすべてのリンク・ステーションおよび接続ネットワーク伝送グループ (TG) を削除します。 続いて、ポートの制御ブロックを削除し、記憶域を解放して、そのポートが正常に削除されたかどうかを示す応答をノード・オペレーター機能から戻します。

関連のある PU をもつリンク・ステーションが削除される場合（このリンク・ステーションがそのポートに関連したものであるため）、この PU 上で定義されている LU もすべて削除されることに注意してください。

VCB 構造

```
typedef struct delete_port
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   port_name[8];        /* name of port                 */
} DELETE_PORT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

port_name

削除するポート名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PORT_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_PORT_ACTIVE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_TP

DELETE_TP は、トランザクション・プログラム (TP) の定義の削除を要求します。

VCB 構造

```
typedef struct delete_tp
{
        unsigned short  opcode;           /* verb operation code          */
        unsigned char   reserv2;          /* reserved                     */
        unsigned char   format;           /* format                       */
        unsigned short  primary_rc;       /* primary return code          */
        unsigned long   secondary_rc;     /* secondary return code        */
        unsigned char   tp_name[64];      /* TP name                      */
} DELETE_TP;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_TP 形式。VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

tp_name

トランザクション・プログラムの名前。Communications Server は、このフィールドの文字セットを検査しません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_TP_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

第5章 活動化 Verb と非活動化 Verb

この章では、以下のものを活動化および非活動化するのに使用される verb について説明します。

- データ・リンク制御 (DLC)
- 内部 PU
- ポート
- リンク・ステーション
- セッション
- 会話グループ

またこの章では、高性能経路指定 (HPR) をサポートする接続に対してパス・スイッチを要求するのに使用される verb についても説明します。

START_DLC

START_DLC はデータ・リンク制御 (DLC) の活動化を要求します。 この verb はその後戻されて、DLC の活動化が正常に実行されたかどうかを示します。 DLC に対してポートが定義されていなくても、DLC は開始できるという点に注意してください。 DLC、ポート、およびリンク・ステーションの関係に関する詳細については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

VCB 構造

```
typedef struct start_dlc
{
        unsigned short  opcode;          /* verb operation code       */
        unsigned char   reserv2;         /* reserved                  */
        unsigned char   format;          /* format                    */
        unsigned short  primary_rc;      /* primary return code       */
        unsigned long   secondary_rc;    /* secondary return code     */
        unsigned char   dlc_name[8];     /* name of DLC               */
} START_DLC;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_START_DLC

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

dlc_name

開始するデータ・リンク制御インスタンスの名前。 ローカルで表示可能な文字セットの 8 バイト・ストリングで、DEFINE_DLC verb によって事前に定義していなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_DLC

DLC が非活動化中であるために verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_DLC_DEACTIVATING

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

START_INTERNAL_PU

START_INTERNAL_PU verb は従属 LU リクエスター (DLUR) に、DLUR によって提供される、事前に定義されているローカル PU に対する SSCP-PU セッションの活動化を開始するように要求します。

VCB 構造

```
typedef struct start_internal_pu
{
        unsigned short  opcode;              /* verb operation code       */
        unsigned char   reserv2;             /* reserved                  */
        unsigned char   format;              /* format                    */
        unsigned short  primary_rc;          /* primary return code       */
        unsigned long   secondary_rc;        /* secondary return code     */
        unsigned char   pu_name[8];          /* internal PU name          */
        unsigned char   dlus_name[17];       /* DLUS name                 */
        unsigned char   bkup_dlus_name[17];  /* Backup DLUS name          */
} START_INTERNAL_PU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_START_INTERNAL_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

SSCP-PU セッションの活動化のフローが送信請求される宛先の内部 PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlus_name

指定の PU に対する SSCP-PU セッション活動化を送信請求するために DLUR が通信する従属 LU サーバー (DLUS) ノードの名前。 これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。） この値は、DEFINE_INTERNAL_PU verb で指定された値を上書きします。 フィールドがすべてゼロに設定されている場合、DEFINE_INTERNAL_PU verb で指定した DLUS が使用されます。 DEFINE_INTERNAL_PU verb で DLUS が指定されていない場合、（DEFINE_DLUR_DEFAULTS verb によって指定されていれば）グローバル省略時値が使用されます。

bkup_dlus_name

指定の PU に対するバックアップ DLUS として DLUS が保管する DLUS ノードの名前。 これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側

が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）この値は、DEFINE_INTERNAL_PU verb で指定された値を上書きします。 フィールドがすべてゼロに設定されている場合、DEFINE_INTERNAL_PU verb によって指定されたバックアップ DLUS 名は、この PU に対するバックアップ DLUS として保存されます。DEFINE_INTERNAL_PU verb によってバックアップ DLUS が指定されていない場合、（DEFINE_DLUR_DEFAULTS verb によって定義されていれば）この PU に対するバックアップ省略時値としてグローバル・バックアップ省略時値 DLUS が保存されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_DLUS_NAME

AP_INVALID_BKUP_DLUS_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_NO_DEFAULT_DLUS_DEFINED

AP_PU_NOT_DEFINED
AP_PU_ALREADY_ACTIVATING
AP_PU_ALREADY_ACTIVE

verb が正常に実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNSUCCESSFUL

secondary_rc
AP_DLUS_REJECTED

AP_DLUS_CAPS_MISMATCH
AP_PU_FAILED_ACTPU

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

START_LS

START_LS はリンクの活動化を要求します。 この verb は応答として戻されて、リンクが正常に活動化したかどうかを示します。

DLC 、ポートおよびリンク・ステーションの間の関係については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

VCB 構造

```
typedef struct start_ls
{
        unsigned short  opcode;           /* verb operation code         */
        unsigned char   reserv2;          /* reserved                    */
        unsigned char   format;           /* format                      */
        unsigned short  primary_rc;       /* primary return code         */
        unsigned long   secondary_rc;     /* secondary return code       */
        unsigned char   ls_name[8];       /* name of link station        */
        unsigned char   enable;           /* whether the link is enabled */
        unsigned char   reserv3[3];       /* reserved                    */
} START_LS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_START_LS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

ls_name

開始するリンク・ステーションの名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **ls_name** の値は、DEFINE_LS verb での値と一致しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LINK_NAME_SPECIFIED

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PORT_INACTIVE

AP_ACTIVATION_LIMITS_REACHED
AP_PARALLEL_TGS_NOT_SUPPORTED
AP_ALREADY_STARTING
AP_LINK_DEACT_IN_PROGRESS

リンクが活動状態になる前に後述の STOP_LS または STOP_PORT によって verb が取り消されたために verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_CANCELLED

secondary_rc
AP_LINK_DEACTIVATED

リンク・ソフトウェアによってパートナーが見つけられないために verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_LS_FAILURE

secondary_rc
AP_PARTNER_NOT_FOUND

リンクの確立中にリンク・エラーが発生したために verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_LS_FAILURE

secondary_rc
AP_ERROR

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

START_PORT

START_PORT はポートの活動化を要求します。 この verb は戻されて、ポートが正常に実行したかどうかを示します。リンク・ステーションがポートに定義されていなくてもポートは開始できますが、ポートの親 DLC が非活動状態の場合、ポートは開始しません。

DLC 、ポートおよびリンク・ステーションの間の関係については、18ページの『DLC プロセス、ポート、およびリンク・ステーション』を参照してください。

VCB 構造

```
typedef struct start_port
{
        unsigned short  opcode;              /* verb operation code       */
        unsigned char   reserv2;             /* reserved                  */
        unsigned char   format;              /* format                    */
        unsigned short  primary_rc;          /* primary return code       */
        unsigned long   secondary_rc;        /* secondary return code     */
        unsigned char   port_name[8];        /* name of port              */
} START_PORT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_START_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

port_name

開始するポートの名前。 ローカルで表示可能な文字セットの 8 バイト・ストリングで、DEFINE_PORT verb での名前と一致しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PORT_NAME

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_DLC_INACTIVE

AP_STOP_PORT_PENDING
AP_DUPLICATE_PORT

verb が取り消されたために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_CANCELLED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

STOP_DLC

STOP_DLC は DLC の停止を要求します。 この verb は戻されて DLC が正常に停止したかどうかを示します。

VCB 構造

```
typedef struct stop_dlc
{
        unsigned short  opcode;             /* verb operation code       */
        unsigned char   reserv2;            /* reserved                  */
        unsigned char   format;             /* format                    */
        unsigned short  primary_rc;         /* primary return code       */
        unsigned long   secondary_rc;       /* secondary return code     */
        unsigned char   stop_type;          /* stop type                 */
        unsigned char   dlc_name[8];        /* name of DLC               */
} STOP_DLC;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_STOP_DLC

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

stop_type

DLC を停止する方法。

AP_ORDERLY_STOP

DLC を停止する前に、ノードはクリーンアップ操作を実行します。

AP_IMMEDIATE_STOP

ノードは即時に DLC を停止します。

dlc_name

停止する DLC の名前。 ローカルで表示可能な文字セットの 8 バイト・ストリングで、DEFINE_DLC verb での名前と一致しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_DLC

AP_UNRECOGNIZED_DEACT_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_STOP_DLC_PENDING

verb が取り消されたために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_CANCELLED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

STOP_INTERNAL_PU

STOP_INTERNAL_PU verb は 事前に定義されたローカル PU に対する SSCP-PU セッションの非活動化を開始するように要求します。by DLUR.

VCB 構造

```
typedef struct stop_internal_pu{
        unsigned short  opcode;             /* verb operation code       */
        unsigned char   reserv2;            /* reserved                  */
        unsigned char   format;             /* format                    */
        unsigned short  primary_rc;         /* primary return code       */
        unsigned long   secondary_rc;       /* secondary return code     */
        unsigned char   pu_name[8];         /* internal PU name          */
        unsigned char   stop_type;          /* type of stop requested    */
} STOP_INTERNAL_PU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_STOP_INTERNAL_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

SSCP-PU セッションが非活動化される宛先の内部 PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

stop_type

PU に対して要求する停止のタイプを指定します。 正常停止では、SSCP-PU セッションを非活動化する前に、基礎となるすべての PLU-SLU セッションおよび SSCP-LU セッションを非活動化します。

AP_ORDERLY_STOP
AP_IMMEDIATE_STOP

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_STOP_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_PU_NOT_DEFINED

AP_PU_ALREADY_DEACTIVATING
AP_PU_NOT_ACTIVE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

STOP_LS

STOP_LS はリンク・ステーションの非活動化を要求します。 この verb は戻されて、リンクが正常に停止したかどうかを示します。

VCB 構造

```
typedef struct stop_ls
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   stop_type;          /* stop type                    */
        unsigned char   ls_name[8];         /* name of link station         */
        unsigned char   disable;            /* whether the link is disabled */
        unsigned char   reserved[3];        /* reserved                     */
} STOP_LS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_STOP_LS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

stop_type

リンク・ステーションを停止する方法。

AP_ORDERLY_STOP

リンク・ステーションを停止する前に、ノードはクリーンアップ操作を実行します。

AP_IMMEDIATE_STOP

ノードは即時にリンク・ステーションを停止します。

ls_name

開始するリンク・ステーションの名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。**ls_name** の値は、DEFINE_LS verb での値と一致しなくてはなりません。

disable

このリンク・ステーションに関して、リモート活動化または要求による活動化が使用不能であるかどうかを示します。 AP_NO に設定している場合、DEFINE_LS verb からの **auto_act_supp** と **disable_remote_act** の値によって与えられる状態にリンク・ステーションが戻されます。 設定していない場合、以下の値が可能(で、ともに論理和 (OR) をとることが可能)です。

AP_AUTO_ACT
このリンクは、ローカル・ノードからの要求によっては再活動化できません。

AP_REMOTE_ACT
このリンクはリモート・ノードによっては活動化できません。AP_YES に設定した **disable_remote_act** で構成したリンクでは、このビットは無視されます。(リモート・ノードによる活動化は STOP_LS により常に使用不能です。)

disable フィールドを AP_NO に設定していない場合、**disable** フィールドを設定する目的で、活動状態ではないリンクまたは非活動化の処理中であるリンクに対して STOP_LS が発行できます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_UNRECOGNIZED_DEACT_TYPE

AP_LINK_NOT_DEFD

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_LINK_DEACT_IN_PROGRESS

verb が取り消されたために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_CANCELLED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

STOP_PORT

STOP_PORT はポートの停止を要求します。 この verb は戻されて、ポートが正常に停止したかどうかを示します。

VCB 構造

```
typedef struct stop_port
{
        unsigned short  opcode;              /* verb operation code      */
        unsigned char   reserv2;             /* reserved                 */
        unsigned char   format;              /* format                   */
        unsigned short  primary_rc;          /* primary return code      */
        unsigned long   secondary_rc;        /* secondary return code    */
        unsigned char   stop_type;           /* Stop Type                */
        unsigned char   port_name[8];        /* name of port             */
} STOP_PORT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_STOP_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

stop_type

ポートを停止する方法。

AP_ORDERLY_STOP

ポートを停止する前に、ノードはクリーンアップ操作を実行します。

AP_IMMEDIATE_STOP

ノードは即時にポートを停止します。

port_name

停止するポートの名前。 ローカルで表示可能な文字セットの 8 バイト・ストリングで、DEFINE_PORT verb での名前と一致しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PORT_NAME

AP_UNRECOGNIZED_DEACT_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_STOP_PORT_PENDING

verb が取り消されたために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_CANCELLED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

ACTIVATE_SESSION

ACTIVATE_SESSION verb は、特定モードの特性を使用して、ローカル LU と指定のパートナー LU との間のセッションの活動化を要求します。

VCB 構造

```
typedef struct activate_session
{
        unsigned short  opcode;          /* verb operation code       */
        unsigned char   reserv2;         /* reserved                  */
        unsigned char   format;          /* format                    */
        unsigned short  primary_rc;      /* primary return code       */
        unsigned long   secondary_rc;    /* secondary return code     */
        unsigned char   lu_name[8];      /* local LU name             */
        unsigned char   lu_alias[8];     /* local LU alias            */
        unsigned char   plu_alias[8];    /* partner LU alias          */
        unsigned char   mode_name[8];    /* mode name                 */
        unsigned char   fqplu_name[17];  /* fully qualified partner   */
                                         /* LU name                   */
        unsigned char   reserv3;         /* reserved                  */
        unsigned char   session_id[8];   /* session identifier        */
} ACTIVATE_SESSION;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_ACTIVATE_SESSION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

セッションの活動化を要求するローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

セッションの活動化を要求するローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_alias** と **lu_name** の両方がすべてゼロに設定されている場合、verb は制御点に関連した LU（省略時値 LU）に転送されます。

plu_alias

ローカル LU に認識されているパートナー LU の別名。 この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトす

べてが有効であり、8 バイトすべてを設定しなくてはなりません。 このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。

mode_name
構成の際に定義される一連のネットワーキングの特性の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

fqplu_name
パートナー LU の完全修飾名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

secondary_rc
AP_AS_SPECIFIED

AP_AS_NEGOTIATED

session_id
活動化したセッションの 8 バイトの識別子です。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PLU_NAME

AP_LU_SESS_LIMIT_EXCEEDED
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS
AP_INVALID_MODE_NAME

verb がモードのセッション限度を越えた場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

Secondary_rc

AP_EXCEEDS_MAX_ALLOWED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

その他のエラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターのいずれかを戻します。

primary_rc

AP_ACTIVATION_FAIL_NO_RETRY

AP_ACTIVATION_FAIL_RETRY

DEACTIVATE_CONV_GROUP

DEACTIVATE_CONV_GROUP verb は、指定の会話グループに対応するセッションの非活動化を要求します。 この verb は、ノード・オペレーター機能 API の一部ですが、基本的には、Communications Server APPC API を使用するトランザクション・プログラムを作成するアプリケーション・プログラマーによって使用されることを想定しています。会話グループ識別子は、*Communications Server クライアント/サーバー 通信プログラミング*.で定義されている MC_ALLOCATE、 ALLOCATE、 MC_GET_ATTRIBUTES、 GET_ATTRIBUTES および RECEIVE_ALLOCATE によって戻されます。

VCB 構造

```
typedef struct deactivate_conv_group
{
        unsigned short  opcode;          /* verb operation code           */
        unsigned char   reserv2;         /* reserved                      */
        unsigned char   format;          /* format                        */
        unsigned short  primary_rc;      /* primary return code           */
        unsigned long   secondary_rc;    /* secondary return code         */
        unsigned char   lu_name[8];      /* local LU name                 */
        unsigned char   lu_alias[8];     /* local LU alias                */
        unsigned long   conv_group_id;   /* conversation group identifier */
        unsigned char   type;            /* deactivation type             */
        unsigned char   reserv3[3];      /* reserved                      */
        unsigned long   sense_data;      /* deactivation sense data       */
} DEACTIVATE_CONV_GROUP;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEACTIVATE_CONV_GROUP

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

会話グループの非活動化を要求するローカル LU の LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

会話グループの非活動化を要求するローカル LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** と **lu_alias** の両方がすべてゼロに設定されている場合、verb は制御点に関連した LU（省略時値 LU）に転送されます。

conv_group_id
非活動されるセッションの会話グループ識別子。

type 非活動化のタイプ。

AP_DEACT_CLEANUP
このセッションはパートナー LU からの応答を待たずに、即時に終了します。

AP_DEACT_NORMAL
セッションを使用しているすべての会話が終了した後でこのセッションを終了します。

sense_data
CLEANUP タイプの非活動化で使用されるセンス・データを指定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CLEANUP_TYPE

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEACTIVATE_SESSION

DEACTIVATE_SESSION verb は、特定のセッション、または特定モードのすべてのセッションの非活動化を要求します。

VCB 構造

```
typedef struct deactivate_session
{
        unsigned short  opcode;          /* verb operation code       */
        unsigned char   reserv2;         /* reserved                  */
        unsigned char   format;          /* format                    */
        unsigned short  primary_rc;      /* primary return code       */
        unsigned long   secondary_rc;    /* secondary return code     */
        unsigned char   lu_name[8];      /* local LU name             */
        unsigned char   lu_alias[8];     /* local LU alias            */
        unsigned char   session_id[8];   /* session identifier        */
        unsigned char   plu_alias[8];    /* partner LU alias          */
        unsigned char   mode_name[8];    /* mode name                 */
        unsigned char   type;            /* deactivation type         */
        unsigned char   reserv3[3];      /* reserved                  */
        unsigned long   sense_data;      /* deactivation sense data   */
        unsigned char   fqplu_name[17];  /* fully qualified partner   */
                                         /* LU name                   */
        unsigned char   reserv4[20];     /* reserved                  */
} DEACTIVATE_SESSION;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEACTIVATE_SESSION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

セッションの非活動化を要求するローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

セッションの非活動化を要求するローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** フィールドと **lu_alias** フィールドの両方がすべてゼロに設定されている場合、verb は制御点に関連した LU （省略時値 LU）に転送されます。

session_id

非活動化するセッションの 8 バイトの識別子です。このフィールドがすべてゼロに設定されている場合、Communications Server はパートナー LU およびモードに対するすべてのセッションを非活動化します。

plu_alias

ローカル LU に認識されているパートナー LU の別名。この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。

mode_name

構成の際に定義される一連のネットワーキングの特性の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

type 非活動化のタイプ。

AP_DEACT_CLEANUP

パートナー LU からの応答を待たずに、即時にセッションを終了します。

AP_DEACT_NORMAL

セッションを使用しているすべての会話が終了した後でセッションを終了します。

sense_data

CLEANUP タイプの非活動化に使用されるセンス・データを指定します。

fqplu_name

パートナー LU の完全修飾名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

session_id がどの既存セッションとも一致しない場合、それはセッションがすでに非活動されているために一致しない、とみなされる点に注意してください。 この場合、verb は正常に完了します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_MODE_NAME

AP_INVALID_PLU_NAME
AP_INVALID_CLEANUP_TYPE
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

PATH_SWITCH

PATH_SWITCH verb は Communications Server に、高性能経路指定 (HPR) をサポートする接続の際、経路をスイッチするように要求します。 よりよいパスが見つからない場合、接続は未変更のままです。

VCB 構造

```
typedef struct path_switch
{
        unsigned short  opcode;         /* verb operation code   */
        unsigned char   reserv2;        /* reserved              */
        unsigned char   format;         /* format                */
        unsigned short  primary_rc;     /* primary return code   */
        unsigned long   secondary_rc;   /* secondary return code */
        unsigned char   rtp_connection_name[8];
                                        /* RTP connection name   */
} PATH_SWITCH;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_PATH_SWITCH

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

rtp_connection_name

パス・スイッチへの RTP 接続を示します。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_RTP_CONNECTION

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_PATH_SWITCH_IN_PROGRESS

パス・スイッチ試行が失敗したために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNSUCCESSFUL

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

第6章 照会 verb

この章では、ノードの構成と状況についての情報の照会に使用される verb について、説明します。

SNA API クライアントでは、特定のパラメーターのみがサポートされています。

詳細は、*鳴っている電話*を参照してください。

QUERY_ADJACENT_NN

QUERY_ADJACENT_NN はネットワーク・ノードでのみ使用され、隣接するネットワーク・ノード (CP-CP セッションが活動状態であるか、活動状態であったか、または活動状態のときがあったネットワーク・ノード) についての情報を戻します。

隣接ノード情報は、形式化されたリストとして戻されます。 特定のネットワーク・ノードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**adj_nncp_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、**adj_nncp_name** で配列されます。名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます (IBM 6611 APPN MIB の配列順序と同一です) 。AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、リストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_adjacent_nn
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   adj_nncp_name[17];  /* CP name of adj network node   */
} QUERY_ADJACENT_NN;
typedef struct adj_nncp_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   adj_nncp_name[17];  /* CP name of adj. network node  */
        unsigned char   cp_cp_sess_status;  /* CP-CP session status          */
        unsigned long   out_of_seq_tdus;    /* out of sequence TDUs          */
        unsigned long   last_frsn_sent;     /* last FRSN sent                */
        unsigned long   last_frsn_rcvd;     /* last FRSN received            */
        unsigned char   reserva[20];        /* reserved                      */
} ADJ_NNCP_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_ADJACENT_NN

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **adj_nncp_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

adj_nncp_name

隣接ネットワーク・ノードの、17 バイトの完全修飾名で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

adj_nncp_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

adj_nncp_data.adj_nncp_name
隣接ネットワーク・ノードの、17 バイトの完全修飾 CP 名で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

adj_nncp_data.cp_cp_sess_status
CP-CP セッションの状況。 以下のいずれか 1 つに設定されます。

AP-ACTIVE
AP_CONWINNER_ACTIVE
AP_CONLOSER_ACTIVE
AP_INACTIVE

adj_nncp_data.out_of_seq_tdus
このノードから受信した out_of_sequence TDU の数

adj_nncp_data.last_frsn_sent
このノードに送信された、最後のフロー縮小順序番号

adj_nncp_data.last_frsn_rcvd
このノードから受信した、最後のフロー縮小順序番号

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_ADJ_NNCP_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_CN

QUERY_CN は、隣接する接続ネットワークについての情報を戻します。 この情報は、『判別済みデータ』 (実行の間に動的に収集されたデータ)、および 『定義済みデータ』 (アプリケーションにより DEFINE_CN で提供されたデータ) として構造化されます。

この情報は、形式化されたリストとして戻されます。 特定の CN についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**fqcn_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。

このリストは **fqcn_name** によって配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、リストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_cn
{
        unsigned short  opcode;             /* Verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* Primary return code           */
        unsigned long   secondary_rc;       /* Secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   fqcn_name[17];      /* Name of connection network    */
} QUERY_CN;
typedef struct cn_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   fqcn_name[17];      /* Name of connection network    */
        unsigned char   reserv1;            /* reserved                      */
        CN_DET_DATA     det_data;           /* Determined data               */
        CN_DEF_DATA     def_data;           /* Defined data                  */
} CN_DATA;
typedef struct cn_det_data
{
        unsigned short  num_act_ports;      /* number of active ports        */
        unsigned char   reserva[20];        /* reserved                      */
} CN_DET_DATA;
```

```
typedef struct cn_def_data
{
        unsigned char   description[RD_LEN];
                                            /* resource description          */
        unsigned char   num_ports;          /* number of ports on CN         */
        unsigned char   reserv1[16];        /* reserved                      */
        TG_DEFINED_CHARS tg_chars;          /* TG characteristics            */
} CN_DEF_DATA;
typedef struct tg_defined_chars
{
        unsigned char   effect_cap;         /* effective capacity            */
        unsigned char   reserve1[5];        /* reserved                      */
        unsigned char   connect_cost;       /* connection cost               */
        unsigned char   byte_cost;          /* byte cost                     */
        unsigned char   reserve2;           /* reserved                      */
        unsigned char   security;           /* security                      */
        unsigned char   prop_delay;         /* propagation delay             */
        unsigned char   modem_class;        /* modem class                   */
        unsigned char   user_def_parm_1;    /* user-defined parameter 1      */
        unsigned char   user_def_parm_2;    /* user-defined parameter 2      */
        unsigned char   user_def_parm_3;    /* user-defined parameter 3      */
} TG_DEFINED_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_CN

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **fqcn_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

fqcn_name
17 バイトの完全修飾接続ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

cn_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

cn_data.fqcn_name
17 バイトの完全修飾接続ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

cn_data.det_data.num_act_ports
接続ネットワーク上の活動状態にあるポートの数を示す、動的な値

cn_data.def_data.description
資源の記述 (DEFINE_CN で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

cn_data.def_data.num_ports
接続ネットワーク上のポートの数

cn_data.def_data.tg_chars.effect_cap
実効容量の実際の単位。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。実効容量の各単位は 300bps です。

cn_data.def_data.tg_chars.connect_cost Cost per connect time.
有効な値は 0 から 255 までの整数です。ここで、0 は接続時間当たりの最も低いコストであり、255 は接続時間当たりの最も高いコストです。

cn_data.def_data.tg_chars.byte_cost
バイト当たりのコスト。有効な値は 0 から 255 までの整数です。ここで、0 はバイト当たりの最も低いコストであり、255 はバイト当たりの最も高いコストです。

cn_data.def_data.tg_chars.security
以下のリストに示すセキュリティーについての値です。

AP_SEC_NONSECURE
セキュリティーが存在しません。

AP_SEC_PUBLIC_SWITCHED_NETWORK
この接続ネットワーク上を伝送されたデータが公共交換網を介して流れます。

AP_SEC_UNDERGROUND_CABLE
データは保護された地下回線を介して伝送されます。

AP_SEC_SECURE_CONDUIT
回線は防護されていない保護コンジットです。

AP_SEC_GUARDED_CONDUIT
コンジットが物理的に漏えいしないように保護されています。

AP_SEC_ENCRYPTED
回線を暗号化します。

AP_SEC_GUARDED_RADIATION
回線が物理的および放射的に漏えいしないように保護されています。

cn_data.def_data.tg_chars.prop_delay
波及遅延。信号がリンクの長さを通るのにかかる時間をマイクロ秒で表します。この値は 1 バイトの浮動小数点数として指定され、公式 0.1mmm * 2 eeeee によって表されます。ここでバイトのビット表現は eeeeemmm です。省略時値は以下のとおりです。

AP_PROP_DELAY_MINIMUM
波及遅延はありません。

AP_PROP_DELAY_LAN
480 マイクロ秒未満の遅延です。

AP_PROP_DELAY_TELEPHONE
480 マイクロ秒から 49 512 マイクロ秒の間の遅延です。

AP_PROP_DELAY_PKT_SWITCHED_NET
49 512 マイクロ秒から 245 760 マイクロ秒の間の遅延です。

AP_PROP_DELAY_SATELLITE
245 760 マイクロ秒を越える遅延です。

AP_PROP_DELAY_MAXIMUM
最大秒の波及遅延です。

cn_data.def_data.tg_chars.modem_class
予約済みこのフィールドは常にゼロに設定しなくてはなりません。

cn_data.def_data.tg_chars.user_def_parm_1
ユーザー定義のパラメーターで、範囲は 0 から 255

cn_data.def_data.tg_chars.user_def_parm_2
ユーザー定義のパラメーターで、範囲は 0 から 255

cn_data.def_data.tg_chars.user_def_parm_3
ユーザー定義のパラメーターで、範囲は 0 から 255

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CN_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_CN_PORT

QUERY_CN_PORT は、隣接接続ネットワークで定義されたポートについての情報を戻します。 この情報は、形式化されたリストとして戻されます。 特定のポートについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**port_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 **fqcn_name** フィールドは常に、有効な接続ネットワークの名前に設定しなくてはならない点に注意してください。

リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

VCB 構造

```
typedef struct query_cn_port
{
        unsigned short  opcode;             /* Verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* Primary return code          */
        unsigned long   secondary_rc;       /* Secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   fqcn_name[17];      /* Name of connection network   */
        unsigned char   port_name[8];       /* port name                    */
} QUERY_CN_PORT;
typedef struct cn_port_data
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   fqcn_name[17];      /* Name of connection network   */
        unsigned char   port_name[8];       /* name of port                 */
        unsigned char   tg_num;             /* transmission group number    */
        unsigned char   reserva[20];        /* reserved                     */
} CN_PORT_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_CN_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケ

ーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。指定された **fqcn_name** および **port_name** の組合せ (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

fqcn_name
17 バイトの完全修飾接続ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドは常に設定しなくてはなりません。

port_name
ローカルで表示可能な文字セットの 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

cn_port_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

cn_port_data.fqcn_name
17 バイトの完全修飾接続ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

cn_port_data.port_name
ローカルで表示可能な文字セットの、8 バイトのポート名。8 バイトすべてが有効です。

cn_port_data.tg_num
指定されたポートのための伝送グループ番号

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_CN_NAME

AP_INVALID_PORT_NAMEAP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_COS

QUERY_COS は、特定のサービス・クラスのための経路計算情報を戻します。 この情報は、形式化されたリストとして戻されます。 特定の COS（サービス・クラス）についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**cos_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。 このリストは **cos_name** によって配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です）。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_cos
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   cos_name[8];        /* COS name                      */
} QUERY_COS;
typedef struct cos_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   cos_name[8];        /* COS name                      */
        unsigned char   description[RD_LEN];
                                            /* resource description          */
        unsigned char   transmission_priority;
                                            /* transmission priority         */
        unsigned char   reserv1;            /* reserved                      */
        unsigned short  num_of_node_rows;   /* number of node rows           */
        unsigned short  num_of_tg_rows;     /* number of TG rows             */
        unsigned long   trees;              /* number of tree caches for COS */
        unsigned long   calcs;              /* number of route calculations  */
                                            /* for this COS                  */
        unsigned long   rejs;               /* number of route rejects       */
                                            /* for COS                       */
        unsigned char   reserva[20];        /* reserved                      */
} COS_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_COS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **cos_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

cos_name

サービス・クラス名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

cos_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

cos_data.cos_name
サービス・クラス名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

cos_data.description
資源の記述 (DEFINE_COS で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

cos_data.transmission_priority
伝送優先順位。以下の値のいずれか 1 つに設定されます。

AP_LOW
AP_MEDIUM
AP_HIGH
AP_NETWORK

cos_data.num_of_node_rows
この COS のノードの行数

cos_data.num_of_tg_rows
この COS の TG 行数

cos_data.trees
最後の初期設定以降この COS のために作成された経路ツリー・キャッシュの数

cos_data.calcs
このサービス・クラス (COS) を指定するセッション活動化要求（およびそれに伴う経路計算）の数

cos_data.rejs
ネットワークを介してこのノードから名前指定した宛先へ向かう経路で、指定したサービス・クラス (COS) を使用する受入れ可能な経路がないために

失敗した、セッション活動化要求の数。経路は、指定したサービス・クラスを提供できる活動状態の TG およびノードで全体を作成された場合のみ、受入れ可能です。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_COS_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DEFAULT_PU

QUERY_DEFAULT_PU を使用すると、ユーザーが、DEFINE_DEFAULT_PU verb を使用して定義された省略時の PU を照会することができます。

VCB 構造

```
typedef struct query_default_pu
{
        unsigned short  opcode;         /* verb operation code   */
        unsigned char   reserv2;        /* reserved              */
        unsigned char   format;         /* format                */
        unsigned short  primary_rc;     /* primary return code   */
        unsigned long   secondary_rc;   /* secondary return code */
        unsigned char   def_pu_name[8]; /* default PU name       */
        unsigned char   description[RD_LEN];
                                        /* resource description  */
        unsigned char   def_pu_sess[8]; /* PU name of active     */
                                        /* default session       */
        unsigned char   reserv3[16];    /* reserved              */
} QUERY_DEFAULT_PU;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DEFAULT_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

def_pu_name

一番最近の DEFINE_DEFAULT_PU verb に指定された PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。 DEFINE_DEFAULT_PU verb が発行されていない場合は、このフィールドはすべてゼロに設定されます。

description

資源の記述 (DEFINE_DEFAULT_PU で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_pu_sess

現在活動状態の省略時の PU セッションに関連した PU の名前。これは、

省略時の PU が定義されているがそれと関連したセッションが活動状態でない場合、**def_pu_name** フィールドと異なる名前になります。この場合 Communications Server は、定義された省略時の PU に関連したセッションが活動状態になるまで、直前の省略時の PU に関連したセッションを継続して使用します。活動状態の PU セッションがない場合、このフィールドはすべてゼロに設定されます。

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DEFAULTS

QUERY_DEFAULTS を使用すると、ユーザーが、DEFINE_DEFAULTS verb を使用して定義された省略時値を照会することができます。

VCB 構造

```
typedef struct query_defaults
{
        unsigned short  opcode;           /* verb operation code   */
        unsigned char   reserv2;          /* reserved              */
        unsigned char   format;           /* format                */
        unsigned short  primary_rc;       /* primary return code   */
        unsigned long   secondary_rc;     /* secondary return code */
        DEFAULT_CHARS   default_chars;    /* default information   */
} QUERY_DEFAULTS;
typedef struct default_chars
{
        unsigned char   description[RD_LEN];
                                          /* resource description  */
        unsigned char   mode_name[8];     /* default mode name     */
        unsigned char   reserv[248];      /* reserved              */
} DEFAULT_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DEFAULTS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

default_chars.description

資源の記述 (DEFINE_DEFAULTS で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

default_chars.mode_name

一番最近の DEFINE_DEFAULTS verb に指定されたモードの名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 DEFINE_DEFAULTS verb が発行されていない場合は、このフィールドはすべてゼロに設定されます。

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DIRECTORY_LU

QUERY_DIRECTORY_LU は、ディレクトリー・データベースから LU のリストを戻します。 この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**lu_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **lu_name** によって配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です)。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_directory_lu
{
        unsigned short  opcode;              /* verb operation code         */
        unsigned char   reserv2;             /* reserved                    */
        unsigned char   format;              /* format                      */
        unsigned short  primary_rc;          /* primary return code         */
        unsigned long   secondary_rc;        /* secondary return code       */
        unsigned char   *buf_ptr;            /* pointer to buffer           */
        unsigned long   buf_size;            /* buffer size                 */
        unsigned long   total_buf_size;      /* total buffer size required  */
        unsigned short  num_entries;         /* number of entries           */
        unsigned short  total_num_entries;   /* total number of entries     */
        unsigned char   list_options;        /* listing options             */
        unsigned char   reserv3;             /* reserved                    */
        unsigned char   lu_name[17];         /* network qualified LU name   */
} QUERY_DIRECTORY_LU;

typedef struct directory_lu_summary
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   lu_name[17];         /* network qualified LU name   */
        unsigned char   description[RD_LEN]; /* resource description        */
} DIRECTORY_LU_SUMMARY;

typedef struct directory_lu_detail
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   lu_name[17];         /* network qualified LU name   */
        unsigned char   description[RD_LEN]; /* resource description        */
        unsigned char   server_name[17];     /* network qualified           */
                                             /* server name                 */
        unsigned char   lu_owner_name[17];   /* network qualified           */
                                             /* LU owner name               */
        unsigned char   location;            /* Resource location           */
```

```
        unsigned char   entry_type;          /* Type of the directory entry */
        unsigned char   wild_card;           /* type of wildcard entry      */
        unsigned char   reserva[20];         /* reserved                    */
} DIRECTORY_LU_DETAIL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DIRECTORY_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **lu_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

lu_name

ネットワーク修飾 LU 名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースな

しで最大 8 バイトまで指定できます。）**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
戻されるディレクトリー項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

directory_lu_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

directory_lu_summary.lu_name
ネットワーク修飾 LU 名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

directory_lu_summary.description
資源の記述 (DEFINE_LOCAL_LU、または DEFINE_ADJACENT_NODE で指定されたものです) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

directory_lu_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

directory_lu_detail.lu_name
ネットワーク修飾 LU 名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

directory_lu_detail.description
資源の記述 (DEFINE_LOCAL_LU、または DEFINE_ADJACENT_NODE で

指定されたものです) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

directory_lu_detail.server_name
LU を提供するノードのネットワーク修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

directory_lu_detail.lu_owner_name
LU を所有するノードのネットワーク修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

directory_lu_detail.location
資源の位置を指定します。以下の値のいずれか 1 つになります。

AP_LOCAL
資源はローカル・ノードにあります。

AP_DOMAIN
資源は接続されたエンド・ノードにあります。

AP_CROSS_DOMAIN
資源はローカル・ノードのドメイン内にはありません。

directory_lu_detail.entry_type
ディレクトリー項目のタイプを指定します。以下の値のいずれか 1 つになります。

AP_HOME
ローカル資源。

AP_CACHE
キャッシュ項目。

AP_REGISTER
登録された資源 (NN のみ) 。

directory_lu_detail.wild_card
LU が一致するワイルドカードのタイプを指定します。

AP_OTHER
未認知のタイプの LU 項目。

AP_EXPLICIT
完全 **lu_name** がこの LU の位置付けに使用されます。

AP_PARTIAL_WILDCARD
lu_name のスペース以外の部分のみがこの LU の位置付けに使用されます。

AP_FULL_WILDCARD
すべての **lu_names** がこの LU に方向づけられます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DIRECTORY_STATS

QUERY_DIRECTORY_STATS は、ディレクトリー・データベースの統計を戻します。（キャッシュ情報に関連する統計は、エンド・ノードの場合は予約されています）。この verb を使用してネットワーク locate トラフィックのレベルを判断することができます。 ネットワーク・ノードの場合、この情報を使用してディレクトリー・キャッシュのサイズを調整することができます。ディレクトリー・キャッシュはノードの初期設定時に構成可能です。

VCB 構造

```
typedef struct query_directory_stats
{
        unsigned short  opcode;            /* verb operation code           */
        unsigned char   reserv2;           /* reserved                      */
        unsigned char   format;            /* format                        */
        unsigned short  primary_rc;        /* primary return code           */
        unsigned long   secondary_rc;      /* secondary return code         */
        unsigned long   max_caches;        /* max number of cache entries   */
        unsigned long   cur_caches;        /* cache entry count             */
        unsigned long   cur_home_entries;  /* home entry count              */
        unsigned long   cur_reg_entries;   /* registered entry count        */
        unsigned long   cur_directory_entries;
                                           /* current number of dir entries */
        unsigned long   cache_hits;        /* count of cache finds          */
        unsigned long   cache_misses;      /* count of resources found by   */
                                           /* broadcast search (not cache)  */
        unsigned long   in_locates;        /* locates in                    */
        unsigned long   in_bcast_locates;  /* broadcast locates in          */
        unsigned long   out_locates;       /* locates out                   */
        unsigned long   out_bcast_locates; /* broadcast locates out         */
        unsigned long   not_found_locates; /* unsuccessful locates          */
        unsigned long   not_found_bcast_locates;
                                           /* unsuccessful broadcast        */
                                           /* locates                       */
        unsigned long   locates_outstanding;
                                           /* total outstanding locates     */
        unsigned char   reserva[20];       /* reserved                      */
} QUERY_DIRECTORY_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DIRECTORY_STATS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

max_caches
予約済み

cur_caches
予約済み

cur_home_entries
ホーム項目の現在の数

cur_reg_entries
登録された項目の現在の数

cur_directory_entries
現在ディレクトリーにある項目数の合計

cache_hits
予約済み

cache_misses
予約済み

in_locates
受信された方向づけ locate の数

in_bcast_locates
受信された同報通信 locate の数

out_locates
送信された方向づけ locate の数

out_bcast_locates
送信された同報通信 locate の数

not_found_locates
not found. とともに戻された方向づけ locate の数

not_found_bcast_locates
not found. とともに戻された同報通信 locate の数

locates_outstanding
未解決 locate の現在の数で、方向づけ locate と同報通信 locate の両方を含む。

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DLC

QUERY_DLC は、ノードで定義された DLC についての情報のリストを戻します。この情報は、『判別済みデータ』
(実行の間に動的に収集されたデータ)、および 『定義済みデータ』 (アプリケーションにより DEFINE_DLC で提供されたデータ) として構造化されます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の DLC についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**dlc_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **dlc_name** によって配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_dlc
{
        unsigned short  opcode;             /* verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* primary return code         */
        unsigned long   secondary_rc;       /* secondary return code       */
        unsigned char   *buf_ptr;           /* pointer to buffer           */
        unsigned long   buf_size;           /* buffer size                 */
        unsigned long   total_buf_size;     /* total buffer size required  */
        unsigned short  num_entries;        /* number of entries           */
        unsigned short  total_num_entries;  /* total number of entries     */
        unsigned char   list_options;       /* listing options             */
        unsigned char   reserv3;            /* reserved                    */
        unsigned char   dlc_name[8];        /* name of DLC                 */
} QUERY_DLC;
typedef struct dlc_summary
{
        unsigned short  overlay_size;       /* size of this entry          */
        unsigned char   dlc_name[8];        /* name of DLC                 */
        unsigned char   description[RD_LEN];
                                            /* resource description        */
        unsigned char   state;              /* State of the DLC            */
        unsigned char   dlc_type;           /* DLC type                    */
} DLC_SUMMARY;
typedef struct dlc_detail
{
        unsigned short  overlay_size;       /* size of this entry          */
        unsigned char   dlc_name[8];        /* name of DLC                 */
```

```
        unsigned char   reserv2[2];         /* reserved                    */
        DLC_DET_DATA    det_data;           /* Determined data             */
        DLC_DEF_DATA    def_data;           /* Defined data                */
} DLC_DETAIL;
typedef struct dlc_det_data
{
        unsigned char   state;             /* State of the DLC            */
        unsigned char   reserv3[3];          /* reserved                   */
        unsigned char   reserva[20];         /* reserved                   */
} DLC_DET_DATA;
typedef struct dlc_def_data
{
        unsigned char   description[RD_LEN];
                                          /* resource description        */
        unsigned char   dlc_type;         /* DLC type                    */
        unsigned char   neg_ls_supp;      /* negotiable LS support       */
        unsigned char   port_types;       /* allowable port types        */
        unsigned char   reserv3[11];        /* reserved                  */
        unsigned short  dlc_spec_data_len; /* Length of DLC specific data */
} DLC_DEF_DATA
;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DLC

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **dlc_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

dlc_name
DLC 名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

dlc_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

dlc_summary.dlc_name
DLC 名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

dlc_summary.description
資源の記述 (DEFINE_DLC で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

dlc_summary.state
DLC の状況。このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_ACTIVE
AP_NOT_ACTIVE
AP_PENDING_INACTIVE

dlc_summary.dlc_type
DLC のタイプ。Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

dlc_detail.overlay_size
この項目のバイト数であり (dlc_spec_data を含む)、したがって次に戻される項目（ある場合）へのオフセット

dlc_detail.dlc_name
DLC 名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

dlc_detail.det_data.state
DLC の状況。このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_ACTIVE
AP_NOT_ACTIVE
AP_PENDING_INACTIVE

dlc_detail.def_data.description
資源の記述 (DEFINE_DLC で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

dlc_detail.def_data.dlc_type
DLC のタイプ。Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

dlc_detail.def_data.neg_ls_supp
DLC が折衝可能リンク・ステーションをサポートするかどうかを指定します (AP_YES または AP_NO) 。

dlc_detail.def_data.port_types
指定された **dlc_type** に対して認められたポート・タイプを指定します。この値は、以下の 1 つまたは複数の、一緒に論理和 (OR) された値になります。

AP_PORT_NONSWITCHED
AP_PORT_SWITCHED
AP_PORT_SATF

dlc_detail.def_data.dlc_spec_data_len
DLC のタイプ特定データの埋込みなしの長さで、バイトで表します。このデータは DLC_DETAIL 構造に連結されます。このデータは 4 バイトの境界まで埋められて終了します。このフィールドは常にゼロに設定しなくてはなりません。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_DLC_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DLUR_LU

QUERY_DLUR_LU は、DLUR がサポートする LU についての情報のリストを戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**lu_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。

このリストは **lu_name** によって配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

戻される LU のリストを、**pu_name** によってフィルター処理することも、LU がローカルであるかダウンストリームであるかによってフィルター処理することも、またはその両方によってフィルター処理することもできます。 PU によるフィルター処理を希望する場合は、**pu_name** フィールドを設定しなくてはなりません（それ以外の場合このフィールドはすべてゼロに設定しなくてはなりません)。 位置によるフィルター処理を希望する場合は、**filter** フィールドを AP_INTERNAL または AP_DOWNSTREAM に設定しなくてはなりません（それ以外で、フィルター処理が必要ない場合は、このフィールドを AP_NONE に設定しなくてはなりません)。

VCB 構造

```
typedef struct query_dlur_lu
{
        unsigned short  opcode;             /* verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* primary return code         */
        unsigned long   secondary_rc;       /* secondary return code       */
        unsigned char   *buf_ptr;           /* pointer to buffer           */
        unsigned long   buf_size;           /* buffer size                 */
        unsigned long   total_buf_size;     /* total buffer size required  */
        unsigned short  num_entries;        /* number of entries           */
        unsigned short  total_num_entries;  /* total number of entries     */
        unsigned char   list_options;       /* listing options             */
        unsigned char   reserv3;            /* reserved                    */
        unsigned char   lu_name[8];         /* name of LU                  */
        unsigned char   pu_name[8];         /* name of PU to filter on     */
        unsigned char   filter;             /* reserved                    */
} QUERY_DLUR_LU;
```

```
typedef struct dlur_lu_summary
{
        unsigned short  overlay_size;       /* size of this entry          */
        unsigned char   lu_name[8];         /* name of LU                  */
} DLUR_LU_SUMMARY;
typedef struct dlur_lu_detail
{
        unsigned short  overlay_size;       /* size of this entry          */
        unsigned char   lu_name[8];         /* name of LU                  */
        unsigned char   pu_name[8];         /* name of owning PU           */
        unsigned char   dlus_name[17];      /* DLUS name if SSCP-LU        */
                                            /* session active              */
        unsigned char   lu_location;        /* downstream or local LU      */
        unsigned char   nau_address;        /* NAU address of LU           */
        unsigned char   plu_name[17];       /* PLU name if PLU-SLU session */
                                            /* active                      */
        unsigned char   reserv1[27];        /* reserved                    */
        unsigned char   rscv_len;           /* length of appended RSCV     */
} DLUR_LU_DETAIL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DLUR_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **lu_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

lu_name
照会する LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

pu_name
PU 名によるフィルター。 すべてゼロに設定するか 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) に設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されると、指定した PU に関連した LU のみが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

filter 位置によるフィルター。 戻される LU を位置によってフィルター処理するかどうかを指定します (AP_INTERNAL または AP_DOWNSTREAM) 。 フィルター処理が必要ない場合は、このフィールドを AP_NONE に設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

dlur_lu_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

dlur_lu_summary.lu_name
LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlur_lu_detail.overlay_size
この項目のバイト数であり (追加の RSCV を含む)、したがって次に戻される項目（ある場合）へのオフセット

dlur_lu_detail.lu_name
LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlur_lu_detail.pu_name
LU と関連した PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlur_lu_detail.dlus_name
SSCP-LU セッションが活動状態の場合 DLUS ノードの名前。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) SSCP-LU セッションが活動状態でない場合、このフィールドはすべてゼロに設定されます。

dlur_lu_detail.lu_location
LU の位置。戻される値は以下の値のみです。

AP_INTERNAL
AP_DOWNSTREAM

dlur_lu_detail.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 1 から 255 です。

dlur_lu_detail.plu_name
LU に活動状態の PLU-SLU セッションがある場合の PLU の名前。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) PLU-SLU セッションが活動状態でない場合、このフィールドはすべてゼロに設定されます。

dlur_lu_detail.rscv_len
この値は常にゼロになります。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_FILTER_OPTION
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DLUR_PU

QUERY_DLUR_PU は、DLUR がサポートする PU についての情報のリストを戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の PU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**pu_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **pu_name** によって配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

戻される PU のリストを、**dlus_name** によってフィルター処理することも、PU がローカルであるかダウンストリームであるかによってフィルター処理することも、またはその両方によってフィルター処理することもできます。 DLUS によるフィルター処理を希望する場合は、**dlus_name** フィールドを設定しなくてはなりません（それ以外の場合このフィールドはすべてゼロに設定しなくてはなりません)。 PU の位置によるフィルター処理を希望する場合は、**filter** フィールドを AP_INTERNAL または AP_DOWNSTREAM に設定しなくてはなりません（それ以外で、フィルター処理が必要ない場合は、このフィールドを AP_NONE に設定しなくてはなりません)。

VCB 構造

```
typedef struct query_dlur_pu
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   pu_name[8];         /* name of PU                   */
        unsigned char   dlus_name[17];      /* fully qualified DLUS name    */
        unsigned char   filter;             /* local/downstream filter      */
} QUERY_DLUR_PU;
```

```
typedef struct dlur_pu_summary
{
        unsigned short  overlay_size;        /* size of this entry           */
        unsigned char   pu_name[8];          /* name of PU                   */
        unsigned char   description[RD_LEN];
                                             /* resource description         */
} DLUR_PU_SUMMARY;
typedef struct dlur_pu_detail
{
        unsigned short  overlay_size;        /* size of this entry           */
        unsigned char   pu_name[8];          /* name of PU                   */
        unsigned char   description[RD_LEN];
                                             /* resource description         */
        unsigned char   defined_dlus_name[17];
                                             /* defined DLUS name            */
        unsigned char   bkup_dlus_name[17];  /* backup DLUS name             */
        unsigned char   pu_id[4];            /* PU identifier                */
        unsigned char   pu_location;         /* downstream or local PU       */
        unsigned char   active_dlus_name[17];
                                             /* active DLUS name             */
        unsigned char   ans_support;         /* Auto-Network shutdown support */
        unsigned char   pu_status;           /* status of the PU             */
        unsigned char   dlus_session_status; /* status of the DLUS pipe      */
        unsigned char   reserv3;             /* reserved                     */
        FQPCID  fqpcid;                      /* FQPCID used on pipe          */
} DLUR_PU_DETAIL;
typedef struct fqpcid
{
        unsigned char   pcid[8];             /* proc correlator identifier   */
        unsigned char   fqcp_name[17];       /* originator's network         */
                                             /* qualified CP name            */
        unsigned char   reserve3[3];         /* reserved                     */
} FQPCID;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DLUR_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **pu_name** (後に示すパラメーターを参照) は、実際に情報が戻される開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

pu_name
照会する PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dlus_name
DLUS によるフィルター。 すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されると、指定した DLUS ノードへの SSCP-PU セッションに関連した PU のみが戻されます。このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

filter このフィールドは AP_NONE に設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

dlur_pu_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

dlur_pu_summary.pu_name
PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlur_pu_summary.description
資源の記述 (DEFINE_INTERNAL_PU で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

dlur_pu_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

dlur_pu_detail.pu_name
PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

dlur_pu_detail.description
資源の記述 (DEFINE_INTERNAL_PU で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

dlur_pu_detail.defined_dlus_name
DEFINE_INTERNAL_PU verb または DEFINE_LS verb のいずれかによって定義された DLUS ノードの名前 (**dspu_services** は AP_DLUR に設定) 。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

dlur_pu_detail.bkup_dlus_name
DEFINE_INTERNAL_PU verb または DEFINE_LS verb のいずれかによって定義されたバックアップの DLUS ノードの名前 (**dspu_services** は AP_DLUR に設定) 。EBCDIC ピリオドで連結された 2 つのタイプ A の

EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

dlur_pu_detail.pu_id
DEFINE_INTERNAL_PU verb に定義されたか、またはダウンストリーム PU から XID で入手した PU 識別子。これは 4 バイトの 16 進数ストリングです。 0 から 11 までのビットはブロック番号に、 12 から 31 までのビットは ID 番号に設定されて、固有に PU を示します。

dlur_pu_detail.pu_location
PU の位置。戻される値は以下の値のみです。

AP_INTERNALAP_DOWNSTREAM

dlur_pu_detail.active_dlus_name
PU が現在使用している DLUS ノードの名前。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) SSCP-PU セッションが活動状態でない場合、このフィールドはすべてゼロに設定されます。

dlur_pu_detail.ans_support
自動ネットワーク遮断サポートです。このフィールドは、SSCP-LU セッションが非活動状態の場合予約されます。このサポート設定は、SSCP-PU 活動化の時点で DLUS から DLUR へ送られます。PU を制御する SSCP に対して、サブエリア・ノードが自動ネットワーク遮断プロシージャーを開始する場合、リンク・レベルの連絡を継続するべきかどうかを指定します。以下の値のいずれか 1 つになります。

AP_CONT
AP_STOP

dlur_pu_detail.pu_status
PU の状況 (DLUR から見たもの) 。以下の値のいずれか 1 つに設定されます。

AP_RESET
PU はリセット状態です。

AP_PEND_ACTPU
PU はホストからの ACTPU を待っています。

AP_PEND_ACTPU_RSP
ACTPU を PU に転送し、DLUR は現在 PU がそれに応答するのを待っています。

AP_ACTIVE
PU は活動状態です。

AP_PEND_DACTPU_RSP
DACTPU を PU に転送し、DLUR は PU がそれに応答するのを待っています。

AP_PEND_INOP
DLUR は PU を非活動化する前に、完了する必要のあるすべてのイベントを待っています。

dlur_pu_detail.dlus_session_status
現在 PU によって使用されている DLUS パイプの状況。以下の値のいずれか 1 つになります。

AP_PENDING_ACTIVE
AP_ACTIVE
AP_PENDING_INACTIVE
AP_INACTIVE

dlur_pu_detail.fqpcid.pcid
パイプで使用されている処理手順相関識別子。これは 8 バイトの 16 進数ストリングです。SSCP-PU セッションが活動状態でない場合、このフィールドはゼロに設定されます。

dlur_pu_detail.fqpcid.fqcp_name
パイプで使用されている制御ポイントの完全修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）SSCP-PU セッションが活動状態でない場合、このフィールドはゼロに設定されます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_FILTER_OPTION
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DLUS

QUERY_DLUS は、DLUR によって認識される DLUS ノードについての情報を戻します。

この情報は、リストとして戻されます。 特定の DLUS ノードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**dlus_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。

このリストは **dlus_name** で配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

この verb がパイプの統計を戻すことに注意してください。

VCB 構造

```
typedef struct query_dlus
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   dlus_name[17];      /* fully qualified DLUS name     */
} QUERY_DLUS;
typedef struct dlus_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   dlus_name[17];      /* fully qualified DLUS name     */
        unsigned char   is_default;         /* is the DLUS the default       */
        unsigned char   is_backup_default;  /* is DLUS the backup default    */
        unsigned char   pipe_state;         /* state of CPSVRMGR pipe        */
        unsigned short  num_active_pus;     /* num of active PUs using pipe  */
        PIPE_STATS      pipe_stats;         /* pipe statistics               */
} DLUS_DATA;
typedef struct pipe_stats
{
        unsigned long   reqactpu_sent;      /* REQACTPUs sent to DLUS        */
```

```
        unsigned long   reqactpu_rsp_received;
                                        /* RSP(REQACTPU)s received      */
                                        /* from DLUS                    */
        unsigned long   actpu_received;     /* ACTPUs received from DLUS    */
        unsigned long   actpu_rsp_sent;     /* RSP(ACTPU)s sent to DLUS     */
        unsigned long   reqdactpu_sent;     /* REQDACTPUs sent to DLUS      */
        unsigned long   reqdactpu_rsp_received;
                                        /* RSP(REQDACTPU)s received     */
                                        /* from DLUS                    */
        unsigned long   dactpu_received;    /* DACTPUs received from DLUS   */
        unsigned long   dactpu_rsp_sent;    /* RSP(DACTPU)s sent to DLUS    */
        unsigned long   actlu_received;     /* ACTLUs received from DLUS    */
        unsigned long   actlu_rsp_sent;     /* RSP(ACTLU)s sent to DLUS     */
        unsigned long   dactlu_received;    /* DACTLUs received from DLUS   */
        unsigned long   dactlu_rsp_sent;    /* RSP(DACTLU)s sent to DLUS    */
        unsigned long   sscp_pu_mus_rcvd;   /* MUs for SSCP-PU              */
                                        /* sessions received            */
        unsigned long   sscp_pu_mus_sent;   /* MUs for SSCP-PU sessions sent */
        unsigned long   sscp_lu_mus_rcvd;   /* MUs for SSCP-LU sessions     */
                                        /* received                     */
        unsigned long   sscp_lu_mus_sent;   /* MUs for SSCP-LU sessions sent */
} PIPE_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DLUS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **dlus_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

dlus_name
照会する DLUS の名前。 すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成される 17 バイト・ストリングに設定しなくてはならず、右側は EBCDIC スペースで埋めます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

dlus_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

dlus_data.dlus_name
DLUS の名前。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC

文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

dlus_data.is_default
DLUS ノードが DEFINE_DLUR_DEFAULTS verb によって省略時値に指定されているかどうかを指定します (AP_YES または AP_NO) 。

dlus_data.is_backup_default
DLUS ノードが DEFINE_DLUR_DEFAULTS verb によってバックアップの省略時値に指定されているかどうかを指定します (AP_YES または AP_NO) 。

dlus_data.pipe_state
DLUS へのパイプの状況。以下の値のいずれか 1 つになります。

AP_ACTIVE
AP_PENDING_ACTIVE
AP_INACTIVE
AP_PENDING_INACTIVE

dlus_data.num_active_pus
DLUS へのパイプを現在使用している PU の数

dlus_data.pipe_stats.reqactpu_sent
パイプを介して DLUS へ送信する REQACTPU の数

dlus_data.pipe_stats.reqactpu_rsp_received
パイプを介して DLUS から受信する RSP(REQACTPU) の数

dlus_data.pipe_stats.actpu_received
パイプを介して DLUS から受信する ACTPU の数

dlus_data.pipe_stats.actpu_rsp_sent
パイプを介して DLUS へ送信する RSP(ACTPU) の数

dlus_data.pipe_stats.reqdactpu_sent
パイプを介して DLUS へ送信する REQDACTPU の数

dlus_data.pipe_stats.reqdactpu_rsp_received
パイプを介して DLUS から受信する RSP(REQDACTPU) の数

dlus_data.pipe_stats.dactpu_received
パイプを介して DLUS から受信する DACTPU の数

dlus_data.pipe_stats.dactpu_rsp_sent
パイプを介して DLUS へ送信する RSP(DACTPU) の数

dlus_data.pipe_stats.actlu_received
パイプを介して DLUS から受信する ACTLU の数

dlus_data.pipe_stats.actlu_rsp_sent
パイプを介して DLUS へ送信する RSP(ACTLU) の数

dlus_data.pipe_stats.dactlu_received
パイプを介して DLUS から受信する DACTLU の数

dlus_data.pipe_stats.dactlu_rsp_sent
パイプを介して DLUS へ送信する RSP(DACTLU) の数

dlus_data.pipe_stats.sscp_pu_mus_rcvd
パイプを介して DLUS から受信する SSCP-PU MU の数

dlus_data.pipe_stats.sscp_pu_mus_sent
パイプを介して DLUS へ送信する SSCP-PU MU の数

dlus_data.pipe_stats.sscp_lu_mus_rcvd
パイプを介して DLUS から受信する SSCP-LU MU の数

dlus_data.pipe_stats.sscp_lu_mus_sent
パイプを介して DLUS へ送信する SSCP-LU MU の数

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_DLUS_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DOWNSTREAM_LU

QUERY_DOWNSTREAM_LU は、DLUR または PU の集中化、またはその両方によって提供されるダウンストリーム LU についての情報を戻します。 この情報は、判別済みデータ (実行の間に動的に収集されたデータ) および定義済みデータとして構造化されます。 （定義済みデータはアプリケーションにより DEFINE_DOWNSTREAM_LU verb で提供されます。 DLUR がサポートする LU では、暗黙に定義されたデータはダウンストリーム LU が活動化するときに挿入されることに注意してください)。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。特定の ローカル LU についての情報を入手するため、またはリスト情報をいくつかのかたまりに分けて入手するためには、**dslu_name** フィールドを設定しなくてはなりません。それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。

戻される LU を、ローカル・ノードが提供するサービスのタイプによってフィルター処理することも、その LU に関連したダウンストリーム PU によってフィルター処理することも、またはその両方によってフィルター処理することもできます。 サービスのタイプによるフィルター処理を希望する場合は、**dspu_services** フィールドを AP_PU_CONCENTRATION または AP_DLUR に設定しなくてはなりません（それ以外の場合このフィールドは AP_NONE に設定しなくてはなりません)。 PU によるフィルター処理を希望する場合は、**dspu_name** フィールドを設定しなくてはなりません（それ以外の場合このフィールドはすべてゼロに設定しなくてはなりません)。

VCB 構造

```
typedef struct query_downstream_lu
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   dslu_name[8];       /* Downstream LU name            */
        unsigned char   dspu_name[8];       /* Downstream PU name filter     */
        unsigned char   dspu_services;      /* filter on DSPU services type  */
} QUERY_DOWNSTREAM_LU;

typedef struct downstream_lu_summary
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   dslu_name[8];       /* LU name                       */
        unsigned char   dspu_name[8];       /* PU name                       */
        unsigned char   description[RD_LEN];
```

```
                                        /* resource description          */
        unsigned char   dspu_services;  /* type of service provided to   */
                                        /* downstream node               */
        unsigned char   nau_address;    /* NAU address                   */
        unsigned char   lu_sscp_sess_active;
                                        /* Is LU-SSCP session active     */
        unsigned char   plu_sess_active;/* Is PLU-SLU session active     */
} DOWNSTREAM_LU_SUMMARY;

typedef struct downstream_lu_detail
{
        unsigned short  overlay_size;   /* size of this entry            */
        unsigned char   dslu_name[8];   /* LU name                       */
        unsigned char   reserv1[2];     /* reserved                      */
        DOWNSTREAM_LU_DET_DATA det_data;/* Determined data               */
        DOWNSTREAM_LU_DEF_DATA def_data;/* Defined data                  */
} DOWNSTREAM_LU_DETAIL;

typedef struct downstream_lu_det_data
{
        unsigned char   lu_sscp_sess_active;
                                        /* Is LU-SSCP session active     */
        unsigned char   plu_sess_active;/* Is PLU-SLU session active     */
        unsigned char   dspu_services;  /* type of services provided to  */
                                        /* downstream node               */
        unsigned char   reserv1;        /* reserved                      */
        SESSION_STATS   lu_sscp_stats;  /* LU-SSCP session statistics    */
        SESSION_STATS   ds_plu_stats;   /* downstream PLU-SLU session    */
                                        /* statistics                    */
        SESSION_STATS   us_plu_stats;   /* upstream PLU_SLU sess stats   */
        unsigned char   reserva[20];    /* reserved                      */
} DOWNSTREAM_LU_DET_DATA;

typedef struct downstream_lu_def_data
{
        unsigned char   description[RD_LEN];
                                        /* resource description          */
        unsigned char   nau_address;    /* NAU address                   */
        unsigned char   dspu_name[8];   /* Downstream PU name            */
        unsigned char   host_lu_name[8];/* host LU or pool name          */
        unsigned char   reserv2[8];     /* reserved                      */
} DOWNSTREAM_LU_DEF_DATA;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;    /* session receive RU size       */
        unsigned short  send_ru_size;   /* session send RU size          */
        unsigned short  max_send_btu_size; /* max send BTU size          */
        unsigned short  max_rcv_btu_size;  /* max rcv BTU size           */
        unsigned short  max_send_pac_win;  /* max send pacing win size   */
        unsigned short  cur_send_pac_win;  /* current send pacing win size */
        unsigned short  max_rcv_pac_win;   /* max receive pacing win size */
        unsigned short  cur_rcv_pac_win;   /* current receive pacing      */
                                        /* window size                   */
        unsigned long   send_data_frames;  /* number of data frames sent */
        unsigned long   send_fmd_data_frames;
                                        /* num of FMD data frames sent   */
        unsigned long   send_data_bytes;   /* number of data bytes sent  */
        unsigned long   rcv_data_frames;   /* num data frames received   */
```

```
        unsigned long   rcv_fmd_data_frames;
                                        /* num of FMD data frames recvd  */
        unsigned long   rcv_data_bytes;     /* number of data bytes received */
        unsigned char   sidh;               /* session ID high byte          */
        unsigned char   sidl;               /* session ID low byte           */
        unsigned char   odai;               /* ODAI bit set                  */
        unsigned char   ls_name[8];         /* Link station name             */
        unsigned char   reserve;            /* reserved                      */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DOWNSTREAM_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **dslu_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

dslu_name

照会するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC

ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dspu_name
PU 名によるフィルター。 すべてゼロに設定するか 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) に設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されると、指定した PU に関連した LU のみが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

dspu_services
DSPU のサービスによるフィルター。 AP_PU_CONCENTRATION に設定すると、PU の集中化によって提供されるダウンストリーム LU のみが戻されます。AP_DLUR に設定すると、DLUR がサポートする LU のみが戻されます。 それ以外で AP_NONE に設定すると、すべてのダウンストリーム LU についての情報が戻されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

downstream_lu_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

downstream_lu_summary.dslu_name
照会するローカル LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

downstream_lu_summary.dspu_name
この LU が使用しているローカル PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

downstream_lu_summary.description
資源の記述 (DEFINE_DOWNSTREAM_LU または DEFINE_DOWNSTREAM_LU_RANGE で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

downstream_lu_summary.dspu_services
ローカル・ノードがリンクを介してダウンストリーム LU に提供するサービスを指定します。 以下のいずれか 1 つに設定されます。

AP_PU_CONCENTRATION
ダウンストリーム LU のための PU の集中化を行うローカル・ノード。

AP_DLUR
ダウンストリーム LU のための DLUR サポートを行うローカル・ノード。

downstream_lu_summary.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 1 から 255。

downstream_lu_summary.lu_sscp_sess_active
LU-SSCP セッションが活動状態であるかどうかを示します (AP_YES または AP_NO) 。

downstream_lu_summary.plu_sess_active
PLU-SLU セッションが活動状態であるかどうかを示します (AP_YES または AP_NO) 。

downstream_lu_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

downstream_lu_detail.dslu_name
照会するローカル LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

downstream_lu_detail.det_data.lu_sscp_sess_active
ダウンストリーム LU への LU-SSCP セッションが活動状態であるかどうかを示します (AP_YES または AP_NO) 。

downstream_lu_detail.det_data.plu_sess_active
ダウンストリーム LU への PLU-SLU セッションが活動状態であるかどうかを示します (AP_YES または AP_NO) 。

downstream_lu_detail.det_data.dspu_services
ローカル・ノードがリンクを介してダウンストリーム LU に提供するサービスを指定します。 以下の値のいずれか 1 つに設定されます。

AP_PU_CONCENTRATION
ダウンストリーム LU のための PU の集中化を行うローカル・ノード。

AP_DLUR
ダウンストリーム LU のための DLUR サポートを行うローカル・ノード。

downstream_lu_detail.det_data.lu_sscp_stats.rcv_ru_size
受信のときの最大 RU サイズ**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.det_data.lu_sscp_stats.send_ru_size
送信のときの最大 RU サイズ**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.det_data.lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

downstream_lu_detail.det_data.lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

downstream_lu_detail.det_data.lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

downstream_lu_detail.det_data.lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

downstream_lu_detail.det_data.lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

downstream_lu_detail.det_data.lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

downstream_lu_detail.det_data.lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.lu_sscp_stats.sidh
セッション ID の高位バイト

downstream_lu_detail.det_data.lu_sscp_stats.sidl
セッション ID の低位バイト

downstream_lu_detail.det_data.lu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

downstream_lu_detail.det_data.lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

downstream_lu_detail.det_data.ds_plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

downstream_lu_detail.det_data.ds_plu_stats.send_ru_size
送信のときの最大 RU サイズ

downstream_lu_detail.det_data.ds_plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

downstream_lu_detail.det_data.ds_plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

downstream_lu_detail.det_data.ds_plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

downstream_lu_detail.det_data.ds_plu_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

downstream_lu_detail.det_data.ds_plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

downstream_lu_detail.det_data.ds_plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

downstream_lu_detail.det_data.ds_plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.ds_plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.ds_plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.ds_plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.ds_plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.ds_plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.ds_plu_stats.sidh
セッション ID の高位バイト

downstream_lu_detail.det_data.ds_plu_stats.sidl
セッション ID の低位バイト

downstream_lu_detail.det_data.ds_plu_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

downstream_lu_detail.det_data.ds_plu_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

downstream_lu_detail.det_data.us_plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

downstream_lu_detail.det_data.us_plu_stats.send_ru_size
送信のときの最大 RU サイズ

downstream_lu_detail.det_data.us_plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

downstream_lu_detail.det_data.us_plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

downstream_lu_detail.det_data.us_plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

downstream_lu_detail.det_data.us_plu_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

downstream_lu_detail.det_data.us_plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

downstream_lu_detail.det_data.us_plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

downstream_lu_detail.det_data.us_plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.us_plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.us_plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.us_plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

downstream_lu_detail.det_data.us_plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

downstream_lu_detail.det_data.us_plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

downstream_lu_detail.det_data.us_plu_stats.sidh
セッション ID の高位バイト**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.det_data.us_plu_stats.sidl
セッション ID の低位バイト**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.det_data.us_plu_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.det_data.us_plu_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_lu_detail.def_data.description
資源の記述 (DEFINE_DOWNSTREAM_LU または DEFINE_DOWNSTREAM_LU_RANGE で指定されたもの) 。

downstream_lu_detail.def_data.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 1 から 255。

downstream_lu_detail.def_data.dspu_name
LU と関連した PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

downstream_lu_detail.def_data.host_lu_name
ダウンストリーム LU がマップされるホスト LU またはホスト LU プールの名前。 LU の場合は 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。LU プールの場合は Communications Server はこのフィールドに文字セットを指定しません。 このフィールドは DLUR がサポートするダウンストリーム LU のために予約されます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DOWNSTREAM_PU

QUERY_DOWNSTREAM_PU は、ダウンストリーム PU (DEFINE_LS verb を使用して定義) についての情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の ローカル PU についての情報を入手するため、またはリスト情報をいくつかのかたまりに分けて入手するためには、**dspu_name** フィールドを設定しなくてはなりません。それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。

PU のリストを、ローカル・ノードがダウンストリーム PU に提供するサービスのタイプによってフィルター処理することができます。 このためには、**dspu_services** フィールドを AP_PU_CONCENTRATION または AP_DLUR に設定しなくてはなりません。

VCB 構造

```
typedef struct query_downstream_pu
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   dspu_name[8];       /* Downstream PU name           */
        unsigned char   dspu_services;      /* filter on DSPU services type */
} QUERY_DOWNSTREAM_PU;

typedef struct downstream_pu_data
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   dspu_name[8];       /* PU name                      */
        unsigned char   description[RD_LEN];
                                            /* resource description         */
        unsigned char   ls_name[8];         /* Link name                    */
        unsigned char   pu_sscp_sess_active;
                                            /* Is PU-SSCP session active    */
        unsigned char   dspu_services;      /* DSPU service type            */
        SESSION _STATS  pu_sscp_stats;      /* SSCP-PU session stats        */
        unsigned char   reserva[20];        /* reserved                     */
} DOWNSTREAM_PU_DATA;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;        /* session receive RU size      */
        unsigned short  send_ru_size;       /* session send RU size         */
        unsigned short  max_send_btu_size;  /* max send BTU size            */
        unsigned short  max_rcv_btu_size;   /* max rcv BTU size             */
```

```
        unsigned short  max_send_pac_win;   /* max send pacing win size      */
        unsigned short  cur_send_pac_win;   /* current send pacing win size  */
        unsigned short  max_rcv_pac_win;    /* max receive pacing win size   */
        unsigned short  cur_rcv_pac_win;    /* current receive pacing        */
                                            /* window size                   */
        unsigned long   send_data_frames;   /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                            /* num of FMD data frames sent   */
        unsigned long   send_data_bytes;    /* number of data bytes sent     */
        unsigned long   rcv_data_frames;    /* num data frames received      */
        unsigned long   rcv_fmd_data_frames;
                                            /* num of FMD data frames recvd  */
        unsigned long   rcv_data_bytes;     /* number of data bytes received */
        unsigned char   sidh;               /* session ID high byte          */
        unsigned char   sidl;               /* session ID low byte           */
        unsigned char   odai;               /* ODAI bit set                  */
        unsigned char   ls_name[8];         /* Link station name             */
        unsigned char   reserve;            /* reserved                      */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DOWNSTREAM_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **dslu_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

dspu_name
照会するダウンストリーム PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dspu_services
DSPU のサービスによるフィルター。 AP_PU_CONCENTRATION に設定すると、PU の集中化によって提供されるダウンストリーム LU のみが戻されます。AP_DLUR に設定すると、DLUR がサポートする LU のみが戻されます。 それ以外で AP_NONE に設定すると、すべてのダウンストリーム LU についての情報が戻されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

downstream_pu_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

downstream_pu_data.dspu_name
ダウンストリーム PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

downstream_pu_data.description
資源の記述 (DEFINE_LS で指定されたもの) 。

downstream_pu_data.ls_name
リンク・ステーションの名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

downstream_pu_data.pu_sscp_sess_active
ダウンストリーム PU への PU_SSCP セッションが活動状態であるかどうかを示します。 AP_YES または AP_NO のいずれかに設定します。

downstream_pu_data.dspu_services
ローカル・ノードがリンクを介してダウンストリーム PU に提供するサービスを指定します。 以下の値のいずれか 1 つに設定されます。

AP_PU_CONCENTRATION
ダウンストリーム LU のための PU の集中化を行うローカル・ノード。

AP_DLUR
ダウンストリーム LU のための DLUR サポートを行うローカル・ノード。

downstream_pu_data.pu_sscp_stats.rcv_ru_size
受信のときの最大 RU サイズ**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_pu_data.pu_sscp_stats.send_ru_size
送信のときの最大 RU サイズ**downstream_lu_detail.det_data.dspu_services** を AP_PU_CONCENTRATION に設定すると、このフィールドは予約されます。

downstream_pu_data.pu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

downstream_pu_data.pu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

downstream_pu_data.pu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

downstream_pu_data.pu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

downstream_pu_data.pu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

downstream_pu_data.pu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

downstream_pu_data.pu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

downstream_pu_data.pu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

downstream_pu_data.pu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

downstream_pu_data.pu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

downstream_pu_data.pu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

downstream_pu_data.pu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

downstream_pu_data.pu_sscp_stats.sidh
セッション ID の高位バイト

downstream_pu_data.pu_sscp_stats.sidl
セッション ID の低位バイト

downstream_pu_data.pu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

downstream_pu_data.pu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_PU_TYPE
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_DSPU_TEMPLATE

QUERY_DSPU_TEMPLATE は、暗黙リンク上の PU 集信に使用される定義済みのダウンストリーム PU テンプレートに関する情報を戻します。この情報は、リストとして戻されます。特定のダウンストリーム PU テンプレートに関する情報、または複数のかたまり内のリスト情報を得るためには、**template_name** フィールドを設定しなければなりません。それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。

VCB 構造

```
typedef struct query_dspu_template
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   template_name[8];   /* name of DSPU template         */
} QUERY_DSPU_TEMPLATE;

typedef struct dspu_template_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   template_name[8];   /* name of DSPU template         */
        unsigned char   description[RD_LEN];
                                            /* resource description          */
        unsigned char   reserv1[12];        /* reserved                      */
        unsigned short  max_instance;       /* max active template instances */
        unsigned short  active instance;    /* current active instances      */
        unsigned short  num_of_dslu_templates;
                                            /* number of DSLU templates      */
} DSPU_TEMPLATE_DATA;
```

それぞれの **dspu_template_data** には、**num_of_dslu_templates** ダウンストリーム LU テンプレートが続きます。ダウンストリーム LU テンプレートは、それぞれ以下の形式になっています。

```
typedef struct dslu_template_data
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   reserv1[2];         /* reserved                      */
        DSLU_TEMPLATE   dslu_template;      /* downstream LU template        */
} DSLU_TEMPLATE_DATA;

typedef struct dslu_template
{
        unsigned char   min_nau;            /* min NAU address in range      */
```

```
        unsigned char   max_nau;            /* max NAU address in range      */
        unsigned char   reserv1[10];        /* reserved                      */
        unsigned char   host_lu[8];         /* host LU or pool name          */
} DSLU_TEMPLATE;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_DSPU_TEMPLATE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定されていなくてはなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

指定された **template_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

template_name

DSPU テンプレートの名前。これは、ローカルで表示可能な文字セットの 8 バイト・ストリングです。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

dspu_template_data.overlay_size
この項目のバイト数（すべてのダウンストリーム LU を含む。したがって次に戻される項目（ある場合）へのオフセット) です。

dspu_template_data.template_name
DSPU テンプレートの名前。これは、ローカルで表示可能な文字セットの 8 バイト・ストリングです。

dspu_template_data.description
資源の記述 (QUERY_DSPU_TEMPLATE で指定されたもの) 。

dspu_template_data.max_instance
これは、同時にアクティブにできるテンプレートのインスタンスの最大数です。

dspu_template_data.active_instance
現在活動状態である、テンプレートのインスタンスの数です。

dspu_template_data.num_of_dslu_templates
このダウンストリーム PU テンプレート用の、ダウンストリーム LU テンプレートの数です。このフィールドの後には、フォーカル・ポイント・カテゴリーに登録された各アプリケーションごとに **num_of_dslu_templates_application_id** が続きます。

dslu_template_data.overlay_size
この項目のバイト数 (したがって次に戻される項目（ある場合）へのオフセット)です。

dslu_template_data.dslu_template.min_nau
範囲内の最小 NAU アドレス。

dslu_template_data.dslu_template.max_nau
範囲内の最大 NAU アドレス。

dslu_template_data.dslu_template.host_lu_name
範囲内にあるすべてのダウンストリーム LU がマップされる、ホスト LU

またはホスト LU プールの名前。この名前は、8 バイトの英数字で、タイプ A-EBCDIC の EBCDIC ストリング（英字で始まる）です。右側は EBCDIC スペースで埋められます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_TEMPLATE_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_FOCAL_POINT

QUERY_FOCAL_POINT は、Communications Server が認識するフォーカル・ポイントについての情報を戻します。

この情報は、リストとして戻されます。 特定のフォーカル・ポイント・カテゴリーについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**ms_category** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。

VCB 構造

```
typedef struct query_focal_point
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   ms_category[8];     /* name of MS category          */
} QUERY_FOCAL_POINT;
typedef struct fp_data
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   ms_appl_name[8];    /* focal point application name */
        unsigned char   ms_category[8];     /* focal point category         */
         unsigned char   description[RD_LEN];
                                            /* resource description         */
        unsigned char   fp_fqcp_name[17];   /* focal pt fully qual CP name  */
        unsigned char   bkup_appl_name[8];  /* backup focal pt appl name    */
        unsigned char   bkup_fp_fqcp_name[17];
                                            /* backup FP fully qualified    */
                                            /* CP name                      */
        unsigned char   implicit_appl_name[8];
                                            /* implicit FP appl name        */
        unsigned char   implicit_fp_fqcp_name[17];
                                            /* implicit FP fully            */
                                            /* qualified CP name            */
        unsigned char   fp_type;            /* focal point type             */
        unsigned char   fp_status;          /* focal point status           */
        unsigned char   fp_routing;         /* type of MDS routing to use   */
        unsigned char   reserva[20];        /* reserved                     */
        unsigned short  number_of_appls;    /* number of applications       */
} FP_DATA;
```

各 **fp_data** の後に **number_of_appls** アプリケーション名が続きます。 各アプリケーション名のフォーマットは以下のとおりです。

```
typedef struct application_id
{
        unsigned char    appl_name[8];     /* application name              */
} APPLICATION_ID;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_FOCAL_POINT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **ms_category** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

ms_category

管理サービス・カテゴリー。 SNA Management Services に記載されているように管理サービス・カテゴリーに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

fp_data.overlay_size
この項目のバイト数であり (任意のアプリケーション名を含む)、したがって次に戻される項目（ある場合）へのオフセット

fp_data.ms_appl_name
現在活動中のフォーカル・ポイント・アプリケーションの名前。 SNA Management Services に記載されているように管理サービス・アプリケーションに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。

fp_data.ms_category
管理サービス・カテゴリー。 SNA Management Services に記載されているように管理サービス・カテゴリーに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。

fp_data.description
資源の記述 (DEFINE_FOCAL_POINT で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

fp_data.fp_fqcp_name
現在活動中のフォーカル・ポイントの完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

fp_data.bkup_appl_name

バックアップ・フォーカル・ポイント・アプリケーションの名前。 SNA Management Services に記載されているように管理サービス・アプリケーションに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。

fp_data.bkup_fp_fqcp_name

バックアップ・フォーカル・ポイントの完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

fp_data.implicit_appl_name

暗黙フォーカル・ポイント・アプリケーションの名前 (DEFINE_FOCAL_POINTverb を使用して指定) 。 SNA Management Services に記載されているように管理サービス・アプリケーションに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。暗黙フォーカル・ポイントが現在活動中のフォーカル・ポイントである場合、このフィールドは **ms_appl_name** と同じになります。

fp_data.bkup_fp_fqcp_name

暗黙フォーカル・ポイントの完全修飾制御点名 (DEFINE_FOCAL_POINTverb を使用して指定) 。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）暗黙フォーカル・ポイントが現在活動中のフォーカル・ポイントである場合、このフィールドは **fp_fqcp_name** と同じになります。

fp_data.fp_type

フォーカル・ポイントのタイプ。 詳細は *SNA Management Services* を参照してください。 以下の値のいずれか 1 つになります。

AP_EXPLICIT_PRIMARY_FP
AP_BACKUP_FP
AP_DEFAULT_PRIMARY_FP
AP_IMPLICIT_PRIMARY_FP
AP_DOMAIN_FP
AP_HOST_FP
AP_NO_FP

fp_data.fp_status

フォーカル・ポイントの状況。 以下の値のいずれか 1 つになります。

AP_NOT_ACTIVE
フォーカル・ポイントが現在活動していません。

AP_ACTIVE
フォーカル・ポイントが現在活動中です。

AP_PENDING
フォーカル・ポイントが活動保留状態です。 暗黙要求をフォーカル・ポイントに送信し、その応答を受信する前にこうなります。

AP_NEVER_ACTIVE
カテゴリーのアプリケーション登録が受け入れられましたが、指定されたカテゴリーについて使用可能なフォーカル・ポイント情報がありません。

fp_data.fp_routing
MDS トランスポートを使用してデータをフォーカル・ポイントに送信するときにアプリケーションが指定するべき経路指定のタイプ。

AP_DEFAULT
省略時の経路指定を使用して MDS_MU をフォーカル・ポイントに送達します。

AP_DIRECT
MDS_MU が直接フォーカル・ポイントに向かうセッションに経路指定されます。

fp_data.number_of_appls
このフォーカル・ポイント・カテゴリーに登録されるアプリケーションの数。このフィールドの後に、フォーカル・ポイント・カテゴリーに登録された各アプリケーションごとに **number_of_appls application_id entries** が続きます。

appl_name
フォーカル・ポイント・カテゴリーに登録されたアプリケーションの名前。SNA Management Services に記載されているように管理サービス・アプリケーションに対して体系的に定義された 4 バイトの値 (右側を EBCDIC スペースで埋める) か、8 バイトのタイプ 1134 の EBCDIC 導入時定義名のいずれか 1 つになります。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_MS_CATEGORY

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_ISR_SESSION

QUERY_ISR_SESSION はネットワーク・ノードでのみ使用され、そのネットワーク・ノードが中間セッション経路指定を提供しているセッションについてのリスト情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のセッションについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**fqpcid** 構造内のフィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、この構造内のフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストはまず **fqpcid.pcid** で配列され、次に **fqpcid.fqcp_name** の EBCDIC 辞書配列順序に従って配列されます。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

fqpcid 構造の形式は、8 バイトの処理手順相関識別子 (PCID) およびセッション開始元のネットワーク修飾 CP 名です。

VCB 構造

```
typedef struct query_isr_session
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        FQPCID          fqpcid;             /* fully qualified procedure     */
                                            /* correlator ID                 */
} QUERY_ISR_SESSION;
typedef struct isr_session_summary
{
        unsigned short  overlay_size;       /* size of this entry            */
        FQPCID          fqpcid;             /* fully qualified procedure     */
                                            /* correlator ID                 */
} ISR_SESSION_SUMMARY;
typedef struct isr_session_detail
{
        unsigned short  overlay_size;       /* size of this entry            */
        FQPCID          fqpcid;             /* fully qualified procedure     */
                                            /* correlator ID                 */
        unsigned char   trans_pri;          /* Transmission priority:        */
```

```
        unsigned char   cos_name[8];        /* Class-of-service name        */
        unsigned char   ltd_res;            /* Session spans a limited      */
                                            /* resource                     */
        SESSION_STATS   pri_sess_stats;     /* primary hop session stats    */
        SESSION_STATS   sec_sess_stats;     /* secondary hop session        */
                                            /* statistics                   */
        unsigned char   reserv3[3];         /* reserved                     */
        unsigned char   reserva[20];        /* reserved                     */
        unsigned char   rscv_len;           /* Length of following RSCV     */
} ISR_SESSION_DETAIL;

typedef struct fqpcid
{
        unsigned char   pcid[8];            /* pro correlator identifier    */
        unsigned char   fqcp_name[17];      /* orig's network qualified     */
                                            /* CP name                      */
        unsigned char   reserve3[3];        /* reserved                     */
} FQPCID;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;        /* session receive RU size     */
        unsigned short  send_ru_size;       /* session send RU size        */
        unsigned short  max_send_btu_size;  /* Maximum send BTU size       */
        unsigned short  max_rcv_btu_size;   /* Maximum rcv BTU size        */
        unsigned short  max_send_pac_win;   /* Max send pacing window size */
        unsigned short  cur_send_pac_win;   /* Curr send pacing window size*/
        unsigned short  max_rcv_pac_win;    /* Max receive pacing win size */
        unsigned short  cur_rcv_pac_win;    /* Curr rec pacing window size */
        unsigned long   send_data_frames;   /* Number of data frames sent  */
        unsigned long   send_fmd_data_frames;
                                            /* num of FMD data frames sent */
        unsigned long   send_data_bytes;    /* Number of data bytes sent   */
        unsigned long   rcv_data_frames;    /* Num data frames received    */
        unsigned long   rcv_fmd_data_frames;
                                            /* num of FMD data frames recvd*/
        unsigned long   rcv_data_bytes;     /* Num data bytes received     */
        unsigned char   sidh;               /* Session ID high byte        */
        unsigned char   sidl;               /* Session ID low byte         */
        unsigned char   odai;               /* ODAI bit set                */
        unsigned char   ls_name[8];         /* Link station name           */
        unsigned char   reserve;            /* reserved                    */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_ISR_SESSION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケ

ーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **fqpcid** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

fqpcid.pcid
処理手順相関識別子。 これは 8 バイトの 16 進数ストリングです。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

fqpcid.pcid_name
完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

isr_session_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

isr_session_summary.fqpcid.pcid
処理手順相関識別子。

isr_session_summary.fqpcid.fqcp_name
完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

isr_session_detail.overlay_size
この項目のバイト数（追加されたすべての RSCV を含む）であり 、したがって次に戻される項目（ある場合）へのオフセット。

isr_session_detail.fqpcid.pcid
処理手順相関識別子。

isr_session_detail.fqpcid.fqcp_name
完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

session_detail.trans_pri
伝送優先順位。以下の値のいずれか 1 つに設定されます。

AP_LOWAP_MEDIUM
AP_HIGHAP_NETWORK

session_detail.cos_name
サービス・クラス名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

session_detail.ltd_res
セッションが限定資源リンクを使用するかどうか（AP_YES または AP_NO）を指定します。

isr_session_detail.pri_sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

isr_session_detail.pri_sess_stats.send_ru_size
送信のときの最大 RU サイズ

isr_session_detail.pri_sess_stats.max_send_btu_size
1 次セッション・ホップで送信できる最大 BTU サイズ

isr_session_detail.pri_sess_stats.max_rcv_btu_size
1 次セッション・ホップで受信できる最大 BTU サイズ

isr_session_detail.pri_sess_stats.max_send_pac_win
1 次セッション・ホップでの送信ペーシング・ウィンドウの最大サイズ

isr_session_detail.pri_sess_stats.cur_send_pac_win
1 次セッション・ホップでの送信ペーシング・ウィンドウの現行サイズ

isr_session_detail.pri_sess_stats.max_rcv_pac_win
1 次セッション・ホップでの受信ペーシング・ウィンドウの最大サイズ

isr_session_detail.pri_sess_stats.cur_rcv_pac_win
1 次セッション・ホップでの受信ペーシング・ウィンドウの現行サイズ

isr_session_detail.pri_sess_stats.send_data_frames
1 次セッション・ホップで送信する通常フローのデータ・フレームの数

isr_session_detail.pri_sess_stats.send_fmd_data_frames
1 次セッション・ホップで送信する通常フローの FMD データ・フレームの数

isr_session_detail.pri_sess_stats.send_data_bytes
1 次セッション・ホップで送信する通常フローのデータ・バイト数

isr_session_detail.pri_sess_stats.rcv_data_frames
1 次セッション・ホップで受信する通常フローのデータ・フレームの数

isr_session_detail.pri_sess_stats.rcv_fmd_data_frames
1 次セッション・ホップで受信する通常フローの FMD データ・フレームの数

isr_session_detail.pri_sess_stats.rcv_data_bytes
1 次セッション・ホップで受信する通常フローのデータ・バイト数

isr_session_detail.pri_sess_stats.sidh
セッション ID の高位バイト

isr_session_detail.pri_sess_stats.sidl
セッション ID の低位バイト

isr_session_detail.pri_sess_stats.odai
起点宛先アドレス標識。セッションを立ち上げるとき、ローカル・ノードが 1

次リンク・ステーションを含む場合、BIND の送信側は、このフィールドをゼロに設定します。BIND の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールド 1 に設定します。

isr_session_detail.pri_sess_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドは、セッション・データが流れるリンクとセッション統計を関係づけるために使用することができます。

isr_session_detail.sec_sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

isr_session_detail.sec_sess_stats.send_ru_size
送信のときの最大 RU サイズ

isr_session_detail.sec_sess_stats.max_send_btu_size
2 次セッション・ホップで送信できる最大 BTU サイズ

isr_session_detail.sec_sess_stats.max_rcv_btu_size
2 次セッション・ホップで受信できる最大 BTU サイズ

isr_session_detail.sec_sess_stats.max_send_pac_win
2 次セッション・ホップでの送信ペーシング・ウィンドウの最大サイズ

isr_session_detail.sec_sess_stats.cur_send_pac_win
2 次セッション・ホップでの送信ペーシング・ウィンドウの現行サイズ

isr_session_detail.sec_sess_stats.max_rcv_pac_win
2 次セッション・ホップでの受信ペーシング・ウィンドウの最大サイズ

isr_session_detail.sec_sess_stats.cur_rcv_pac_win
2 次セッション・ホップでの受信ペーシング・ウィンドウの現行サイズ

isr_session_detail.sec_sess_stats.send_data_frames
2 次セッション・ホップで送信する通常フローのデータ・フレームの数

isr_session_detail.sec_sess_stats.send_fmd_data_frames
2 次セッション・ホップで送信する通常フローの FMD データ・フレームの数

isr_session_detail.sec_sess_stats.send_data_bytes
2 次セッション・ホップで送信する通常フローのデータ・バイト数

isr_session_detail.sec_sess_stats.rcv_data_frames
2 次セッション・ホップで受信する通常フローのデータ・フレームの数

isr_session_detail.sec_sess_stats.rcv_fmd_data_frames
2 次セッション・ホップで受信する通常フローの FMD データ・フレームの数

isr_session_detail.sec_sess_stats.rcv_data_bytes
2 次セッション・ホップで受信する通常フローのデータ・バイト数

isr_session_detail.sec_sess_stats.sidh
セッション ID の高位バイト

isr_session_detail.sec_sess_stats.sidl
セッション ID の低位バイト (LFSID から)

isr_session_detail.sec_sess_stats.odai
起点宛先アドレス標識。 セッションを立ち上げるとき、ローカル・ノードが 1 次リンク・ステーションを含む場合、BIND の送信側は、このフィールドをゼロに設定します。 BIND の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールド 1 に設定します。

isr_session_detail.sec_sess_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 このフィールドを使用して、中間セッション統計と特定のリンク・ステーションとを関連付けることができます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_FQPCID

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_LOCAL_LU

QUERY_LOCAL_LU はローカル LU についての情報を戻します。 QUERY_LOCAL_LU を発行して、Communications Server 制御点 LU についての情報を取り出すことができます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のローカル LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**lu_name** フィールドまたは **lu_alias** フィールドを設定しなくてはなりません。 **lu_name** フィールドをゼロ以外に設定すると、このフィールドが索引を判別するのに使用されます。 **lu_name** フィールドをすべてゼロに設定すると、**lu_alias** が索引を判別するのに使用されます。 **lu_name** と **lu_alias** の両方のフィールドをすべてゼロに設定すると、制御点（省略時の LU）に関連した LU が使用されます。 **list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合には、これら両方のフィールドは無視されます。 （このとき戻されるリストは、AP_LIST_BY_ALIAS が **list_options** に設定されている場合は LU の別名で配列され、それ以外の場合は LU 名で配列されます。）リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、指定されるオプションに従って **lu_alias** または **lu_name** のいずれかで配列されます。フィールドは EBCDIC 辞書配列順序に従って配列されます。

戻されるローカル LU のリストを、それらのローカル LU が関連した PU の名前によってフィルター処理することができます。 この場合は **pu_name** フィールドを設定しなくてはなりません（それ以外の場合、このフィールドはすべてゼロに設定しなくてはなりません ）。

VCB 構造

```
typedef struct query_local_lu
{
        unsigned short  opcode;            /* verb operation code           */
        unsigned char   reserv2;           /* reserved                      */
        unsigned char   format;            /* format                        */
        unsigned short  primary_rc;        /* primary return code           */
        unsigned long   secondary_rc;      /* secondary return code         */
        unsigned char   *buf_ptr;          /* pointer to buffer             */
        unsigned long   buf_size;          /* buffer size                   */
        unsigned long   total_buf_size;    /* total buffer size required    */
        unsigned short  num_entries;       /* number of entries             */
        unsigned short  total_num_entries; /* total number of entries       */
        unsigned char   list_options;      /* listing options               */
        unsigned char   reserv3;           /* reserved                      */
        unsigned char   lu_name[8];        /* LU name                       */
        unsigned char   lu_alias[8];       /* LU alias                      */
        unsigned char   pu_name[8];        /* PU name filter                */
} QUERY_LOCAL_LU;
typedef struct local_lu_summary
{
        unsigned short  overlay_size;      /* size of this entry            */
        unsigned char   lu_name[8];        /* LU name                       */
```

```
        unsigned char   lu_alias[8];       /* LU alias                     */
        unsigned char   description[RD_LEN];
                                           /* resource description         */
} LOCAL_LU_SUMMARY;
typedef struct local_lu_detail
{
        unsigned short  overlay_size;      /* size of this entry           */
        unsigned char   lu_name[8];        /* LU name                      */
        LOCAL_LU_DEF_DATA def_data;        /* defined data                 */
        LOCAL_LU_DEF_DATA det_data;        /* determined data              */
} LOCAL_LU_DETAIL;
typedef struct local_lu_def_data
{
        unsigned char   description[RD_LEN];
                                           /* resource description         */
        unsigned char   lu_alias[8];       /* local LU alias               */
        unsigned char   nau_address;       /* NAU address                  */
        unsigned char   syncpt_support;    /* Reserved                     */
        unsigned short  lu_session_limit;  /* LU session limit             */
        unsigned char   reserv1;           /* reserved                     */
        unsigned char   reserv2;           /* reserved                     */
        unsigned char   pu_name[8];        /* PU name                      */
        unsigned char   reserv3[8];        /* reserved                     */
        unsigned char   attach_routing_data[128];
                                           /* routing data for             */
                                           /* incoming attaches            */
} LOCAL_LU_DEF_DATA;
typedef struct local_lu_det_data
{
        unsigned char   lu_sscp_sess_active;
                                           /* Is LU-SSCP session active    */
        unsigned char   appl_conn_active;  /* Is LU-SSCP session active    */
        unsigned char   reserv1[2];        /* reserved                     */
        SESSION_STATS   lu_sscp_stats;     /* LU-SSCP session statistics   */
} LOCAL_LU_DET_DATA;
typedef struct session_stats
{
        unsigned short  rcv_ru_size;       /* session receive RU size      */
        unsigned short  send_ru_size;      /* session send RU size         */
        unsigned short  max_send_btu_size; /* max send BTU size            */
        unsigned short  max_rcv_btu_size;  /* max rcv BTU size             */
        unsigned short  max_send_pac_win;  /* max send pacing win size     */
        unsigned short  cur_send_pac_win;  /* current send pacing win size */
        unsigned short  max_rcv_pac_win;   /* max receive pacing win size  */
        unsigned short  cur_rcv_pac_win;   /* current receive pacing       */
                                           /* window size                  */
        unsigned long   send_data_frames;  /* number of data frames sent   */
        unsigned long   send_fmd_data_frames;
                                           /* num of FMD data frames sent  */
        unsigned long   send_data_bytes;   /* number of data bytes sent    */
        unsigned long   rcv_data_frames;   /* num data frames received     */
        unsigned long   rcv_fmd_data_frames;
                                           /* num of FMD data frames recvd */
        unsigned long   rcv_data_bytes;    /* number of data bytes received */
        unsigned char   sidh;              /* session ID high byte         */
```

```
        unsigned char   sidl;           /* session ID low byte          */
        unsigned char   odai;           /* ODAI bit set                 */
        unsigned char   ls_name[8];     /* Link station name            */
        unsigned char   reserve;        /* reserved                     */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_LOCAL_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **lu_name** (または、**lu_name** がすべてゼロに設定されている場合は **lu_alias**) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

AP_LIST_BY_ALIAS

戻されるリストは、**lu_alias** で配列されます。このオプションは、

AP_FIRST_IN_LIST が指定されている場合にだけ有効です。AP_LIST_FROM_NEXT または AP_LIST_INCLUSIVE を指定する場合、リストの配列順序は、**lu_name** または **lu_alias** のどちらが開始点として指定されているかによって異なります。

lu_name

LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。このフィールドをすべてゼロに設定すると、**lu_alias** フィールドが索引の判別に使用されます。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

lu_alias

ローカルで定義されている LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **lu_name** と **lu_alias** の両方のフィールドをすべてゼロに設定すると、制御点（省略時の LU）に関連した LU が使用されます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

pu_name

PU 名によるフィルター。 すべてゼロに設定するか 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) に設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されると、この PU に関連したローカル LU のみが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries

実際に戻される項目の数

total_num_entries

戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

local_lu_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

local_lu_summary.lu_name
LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。

local_lu_summary.lu_alias
ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

local_lu_summary.description
資源の記述 (DEFINE_LOCAL_LU で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

local_lu_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

local_lu_detail.lu_name
LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。

local_lu_detail.def_data.description
資源の記述 (DEFINE_LOCAL_LU で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

local_lu_detail.def_data.lu_alias
ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

local_lu_detail.def_data.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 0 から 255。ゼロ以外の値は、LU が従属 LU であることを暗黙指定します。ゼロの値は LU が独立 LU であることを暗黙指定します。

local_lu_detail.def_data.syncpt_support
予約済み

local_lu_detail.def_data.lu_session_limit
ローカル LU のセッションの最大数。ゼロの値は制限がないことを示します。

local_lu_detail.def_data.default_pool
LU が従属 LU 6.2 省略時プールのメンバーである場合、AP_YES 。独立 LU の場合は常に AP_NO です。

local_lu_detail.def_data.pu_name
この LU が使用する PU の名前。8 バイトの英数字のタイプ A の EBCDIC

ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。このフィールドは従属 LU のみが使用し、独立 LU の場合はすべて 2 進ゼロに設定されます。

local_lu_detail.def_data.attach_routing_data
予約済み

local_lu_detail.det_data.lu_sscp_session_active
LU-SSCP セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。 **def_data.nau_address** がゼロの場合、このフィールドは予約されます。

local_lu_detail.det_data.appl_conn_active
アプリケーションが LU を使用中かどうかを指定します (AP_YES または AP_NO) 。 **def_data.nau_address** がゼロの場合、このフィールドは予約されます。

local_lu_detail.det_data.lu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

local_lu_detail.det_data.lu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

local_lu_detail.det_data.lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

local_lu_detail.det_data.lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

local_lu_detail.det_data.lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

local_lu_detail.det_data.lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

local_lu_detail.det_data.lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

local_lu_detail.det_data.lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

local_lu_detail.det_data.lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

local_lu_detail.det_data.lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

local_lu_detail.det_data.lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

local_lu_detail.det_data.lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

local_lu_detail.det_data.lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

local_lu_detail.det_data.lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

local_lu_detail.det_data.lu_sscp_stats.sidh
セッション ID の高位バイト

local_lu_detail.det_data.lu_sscp_stats.sidl
セッション ID の低位バイト

local_lu_detail.det_data.lu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、ACTLU の送信側がこのフィールドをゼロに設定し、ACTLU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

local_lu_detail.det_data.lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

注: LU-SSCP 統計 (**local_lu_detail.det_data.lu_sscp_stats**) は、**nau_address** がゼロ以外の場合のみ有効です。 ゼロの場合、このフィールドは予約されます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LU_ALIAS

AP_INVALID_LU_NAME
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_LOCAL_TOPOLOGY

すべての APPN ノードは、すべての隣接ノードへの伝送グループ (TG) についての情報が入っているローカル・トポロジー・データベースを保守します。

QUERY_LOCAL_TOPOLOGY を使用して、これらの TG についての情報を戻すことができます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のローカル TG についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**dest** フィールド、**dest_type** フィールド、および **tg_num** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、これらのフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。 このリストはまず **dest**、次に **dest_type**、最後に **tg_num** で配列されます。 **dest** の名前はまず名前の長さによって配列され、長さが同じ名前の場合は辞書配列順序に従って配列されます。 **dest_type** フィールドの配列は AP_LEN_NODE、 AP_NETWORK_NODE、 AP_END_NODE、 AP_VRN の順になっています。**tg_num** の配列は数字順です。

AP_LIST_INCLUSIVE が選択されている場合、戻されるリストは、その名前の最初の有効なレコードから開始します。

AP_LIST_FROM_NEXT を選択すると、リストは、指定された名前に続く名前をもつ最初の有効なレコードから開始します。

VCB 構造

```
typedef struct query_local_topology
{
        unsigned short  opcode;             /* verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* primary return code         */
        unsigned long   secondary_rc;       /* secondary return code       */
        unsigned char   *buf_ptr;           /* pointer to buffer           */
        unsigned long   buf_size;           /* buffer size                 */
        unsigned long   total_buf_size;     /* total buffer size required  */
        unsigned short  num_entries;        /* number of entries           */
        unsigned short  total_num_entries;  /* total number of entries     */
        unsigned char   list_options;       /* listing options             */
        unsigned char   reserv3;            /* reserved                    */
        unsigned char   dest[17];           /* TG destination node         */
        unsigned char   dest_type;          /* TG destination node type    */
        unsigned char   tg_num;             /* TG number                   */
} QUERY_LOCAL_TOPOLOGY;
typedef struct local_topology_summary
{
        unsigned short  overlay_size;       /* size of this entry          */
        unsigned char   dest[17];           /* TG destination node         */
        unsigned char   dest_type;          /* TG destination node type    */
        unsigned char   tg_num;             /* TG number                   */
} LOCAL_TOPOLOGY_SUMMARY;
```

```
typedef struct local_topology_detail
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   dest[17];            /* TG destination node         */
        unsigned char   dest_type;           /* TG destination node type    */
        unsigned char   tg_num;              /* TG number                   */
        unsigned char   reserv1;             /* reserved                    */
        LINK_ADDRESS    dlc_data;            /* DLC signalling data         */
        unsigned long   rsn;                 /* resource sequence number    */
        unsigned char   status;              /* TG status                   */
        TG_DEFINED_CHARS tg_chars;           /* TG characteristics          */
        unsigned char   reserva[16];         /* reserved                    */
} LOCAL_TOPOLOGY_DETAIL;

typedef struct link_address
{
        unsigned short  length;              /* length                      */
        unsigned short  reserve1;            /* reserved                    */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                             /* address                     */
} LINK_ADDRESS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_LOCAL_TOPOLOGY

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **dest**、**dest_type**、および **tg_num** の組合せ（後に示すパラメーターを参照）は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

dest TG に対する完全修飾の宛先ノード名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dest_type
この TG のための宛先ノードのノード・タイプ。以下の値のいずれか 1 つになります。

AP_NETWORK_NODE
AP_VRNAP_END_NODE
AP_LEARN_NODE

dest_type が不明の場合、 AP_LEARN_NODE を指定しなければなりません。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

tg_num
TG と関連した数。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

local_topology_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

local_topology_summary.dest
TG に対する完全修飾の宛先ノード名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

local_topology_summary.dest_type
この TG に対する宛先ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRNAP_END_NODE

dest_type を AP_END_NODE に設定すると、TG の宛先が LEN ノードまたはエンド・ノードのいずれかに指定されることに注意してください。

local_topology_summary.tg_num
TG と関連した数。

local_topology_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

local_topology_detail.dest
TG に対する完全修飾の宛先ノード名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

local_topology_detail.dest_type
この TG に対する宛先ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN
AP_END_NODE

dest_type を AP_END_NODE に設定すると、TG の宛先が LEN ノードまたはエンド・ノードのいずれかに指定されることに注意してください。

local_topology_detail.tg_num
TG と関連した数。

local_topology_detail.dlc_data.length
VRN への接続の DLC アドレスの長さ（**dest_type** が AP_VRN でない場合、ゼロに設定します)。

local_topology_detail.dlc_data.address
VRN への接続の DLC アドレス。

local_topology_detail.rsn
資源順序番号。この資源を所有しているネットワーク・ノードによって割り当てられます。

local_topology_detail.status
TG の状況を指定します。これは、以下の値の 1 つ以上を一緒に論理和 (OR) 演算したものです。

AP_TG_OPERATIVE
AP_TG_CP_CP_SESSIONS
AP_TG_QUIESCING
AP_TG_HPRAP_TG_RTP
AP_NONE

local_topology_detail.tg_chars
TG の特性 (37ページの『DEFINE_CN』 を参照)

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_TG

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_LS

QUERY_LS は、ノードで定義されたリンク・ステーションについての情報のリストを戻します。 この情報は、『判別済みデータ』 (実行の間に動的に収集されたデータ)、および 『定義済みデータ』 (アプリケーションにより DEFINE_LS で提供されたデータ) として構造化されます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の LS についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**ls_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **ls_name** で配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます (IBM 6611 APPN MIB の配列順序と同一です) 。AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

戻されるリンク・ステーションのリストを、それらのリンク・ステーションが属するポートの名前によってフィルター処理することができます。 この場合は **port_name** フィールドを設定しなくてはなりません（それ以外の場合、このフィールドはすべてゼロに設定しなくてはなりません ）。

VCB 構造

```
typedef struct query_ls
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* Primary return code           */
        unsigned long   secondary_rc;       /* Secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   ls_name[8];         /* name of link station          */
        unsigned char   port_name[8];       /* name of link station          */
} QUERY_LS;
typedef struct ls_summary
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   ls_name[8];         /* link station name             */
        unsigned char   description[RD_LEN];
                                            /* resource description          */
```

```
        unsigned char   dlc_type;           /* DLC type                      */
        unsigned char   state;              /* link station state            */
        unsigned short  act_sess_count;     /* currently active sess count   */
        unsigned char   det_adj_cp_name[17];
                                            /* determined adj CP name        */
        unsigned char   det_adj_cp_type;    /* determined adj node type      */
        unsigned char   port_name[8];       /* port name                     */
        unsigned char   adj_cp_name[17];    /* adjacent CP name              */
        unsigned char   adj_cp_type;        /* adjacent node type            */
} LS_SUMMARY;

typedef struct ls_detail
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   ls_name[8];         /* link stations name            */
        LS_DET_DATA     det_data;           /* determined data               */
        LS_DEF_DATA     def_data;           /* defined data                  */
} LS_DETAIL;

typedef struct ls_det_data
{
        unsigned short  act_sess_count;     /* curr active sessions count    */
        unsigned char   dlc_type;           /* DLC type                      */
        unsigned char   state;              /* link station state            */
        unsigned char   sub_state;          /* link station sub state        */
        unsigned char   det_adj_cp_name[17];
                                            /* adjacent CP name              */
        unsigned char   det_adj_cp_type;    /* adjacent node type            */
        unsigned char   dlc_name[8];        /* name of DLC                   */
        unsigned char   dynamic;            /* is LS is dynamic ?            */
        unsigned char   migration;          /* supports migration partners   */
        unsigned char   tg_num;             /* TG number                     */
        LS_STATS        ls_stats;           /* link station statistics       */
        unsigned long   start_time;         /* time LS started               */
        unsigned long   stop_time;          /* time LS stopped               */
        unsigned long   up_time;            /* total time LS active          */
        unsigned long   current_state_time; /* time in current state         */
        unsigned char   deact_cause;        /* deactivation cause            */
        unsigned char   hpr_support;        /* TG HPR support                */
        unsigned char   anr_label[2];       /* local ANR label               */
        unsigned char   hpr_link_lvl_error; /* HPR link-level error          */
        unsigned char   auto_act;           /* auto activate                 */
        unsigned char   ls_role;            /* link station role             */
        unsigned char   reserva;            /* reserved                      */
        unsigned char   node_id[4];         /* determined node id            */
        unsigned char   reservb[32];        /* reserved                      */
} LS_DET_DATA;

typedef struct ls_def_data
{
        unsigned char   description[RD_LEN];
                                            /* resource description          */
        unsigned char   port_name[8];       /* name of associated port       */
        unsigned char   adj_cp_name[17];    /* adjacent CP name              */
        unsigned char   adj_cp_type;        /* adjacent node type            */
        LINK_ADDRESS    dest_address;       /* destination address           */
        unsigned char   auto_act_supp;      /* auto-activate supported       */
        unsigned char   tg_number;          /* Pre-assigned TG number        */
```

```
        unsigned char   limited_resource;   /* limited resource              */
        unsigned char   solicit_sscp_sessions;
                                            /* solicit SSCP sessions         */
        unsigned char   pu_name[8];         /* Local PU name (reserved if    */
                                            /* solicit_sscp_sessions is set  */
                                            /* to AP_NO)                     */
        unsigned char   disable_remote_act; /* disable remote activation flag */
        unsigned char   dspu_services;      /* Services provided for         */
                                            /* downstream PU                 */
        unsigned char   dspu_name[8];       /* Downstream PU name (reserved  */
                                            /* if dspu_services is set to    */
                                            /* AP_NONE or AP_DLUR)           */
        unsigned char   dlus_name[17];      /* DLUS name if dspu_services    */
                                            /* is set to AP_DLUR             */
        unsigned char   bkup_dlus_name[17]; /* Backup DLUS name if           */
                                            /* dspu_services is set          */
                                            /* to AP_DLUR                    */
        unsigned char   hpr_supported;      /* does the link support HPR?    */
        unsigned char   hpr_link_lvl_error; /* does the link support HPR     */
                                            /* link-level error recovery?    */
        unsigned short  link_deact_timer;   /* HPR link deactivation timer   */
        unsigned char   reserv1;            /* reserved                      */
        unsigned char   default_nn_server;  /* Use as default LS to NN server */
        unsigned char   ls_attributes[4];   /* LS attributes                 */
        unsigned char   adj_node_id[4];     /* adjacent node ID              */
        unsigned char   local_node_id[4];   /* local node ID                 */
        unsigned char   cp_cp_sess_support; /* CP-CP session support         */
        unsigned char   use_default_tg_chars;
                                            /* Use default tg_chars          */
        TG_DEFINED_CHARS tg_chars;          /* TG characteristics            */
        unsigned short  target_pacing_count;
                                            /* target pacing count           */
        unsigned short  max_send_btu_size;  /* max send BTU size             */
        unsigned char   ls_role;            /* link station role to use      */
                                            /* on this link                  */
        unsigned char   max_ifrm_rcvd;      /* max number of I-frames rcvd   */
        unsigned char   reserv3[34];        /* reserved                      */
        unsigned short  link_spec_data_len; /* length of link specific data  */
} LS_DEF_DATA;

typedef struct link_address
{
        unsigned short  length;             /* length                        */
        unsigned short  reserve1;           /* reserved                      */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                            /* address                       */
} LINK_ADDRESS;

typedef struct link_spec_data
{
       unsigned char  link_data[SIZEOF_LINK_SPEC_DATA];
}  LINK_SPEC_DATA;

typedef struct tg_defined_chars
{
        unsigned char   effect_cap;         /* Effective capacity            */
        unsigned char   reserve1[5];        /* Reserved                      */
        unsigned char   connect_cost;       /* Connection Cost               */
```

```
        unsigned char   byte_cost;          /* Byte cost                    */
        unsigned char   reserve2;           /* Reserved                     */
        unsigned char   security;           /* Security                     */
        unsigned char   prop_delay;         /* Propagation delay            */
        unsigned char   modem_class;        /* Modem class                  */
        unsigned char   user_def_parm_1;    /* User-defined parameter 1     */
        unsigned char   user_def_parm_2;    /* User-defined parameter 2     */
        unsigned char   user_def_parm_3;    /* User-defined parameter 3     */
} TG_DEFINED_CHARS;

typedef struct ls_stats
{
        unsigned long   in_xid_bytes;       /* number of XID bytes received */
        unsigned long   in_msg_bytes;       /* num message bytes received   */
        unsigned long   in_xid_frames;      /* num XID frames received      */
        unsigned long   in_msg_frames;      /* num message frames received  */
        unsigned long   out_xid_bytes;      /* num XID bytes sent           */
        unsigned long   out_msg_bytes;      /* num message bytes sent       */
        unsigned long   out_xid_frames;     /* num XID frames sent          */
        unsigned long   out_msg_frames;     /* num message frames sent      */
        unsigned long   in_invalid_sna_frames;
                                            /* num invalid frames received  */
        unsigned long   in_session_control_frames;
                                            /* num control frames received  */
        unsigned long   out_session_control_frames;
                                            /* num control frames sent      */
        unsigned long   echo_rsps;          /* response from adj LS count   */
        unsigned long   current_delay;      /* time taken for last test sig */
        unsigned long   max_delay;          /* max delay by test signal     */
        unsigned long   min_delay;          /* min delay by test signal     */
        unsigned long   max_delay_time;     /* time since longest delay     */
        unsigned long   good_xids;          /* successful XID on LS count   */
        unsigned long   bad_xids;           /* unsuccessful XID on LS count */
} LS_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_LS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **ls_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

ls_name

リンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

port_name

ポート名によるフィルター。 すべてゼロに設定するか 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) に設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されると、このポートに属するリンク・ステーションのみが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

ls_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

ls_summary.ls_name
リンク・ステーションの名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

ls_summary.description
資源の記述 (DEFINE_LS で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

ls_summary.dlc_type
DLC のタイプ。 Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

ls_summary.state
このリンク・ステーションの状況。 このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_NOT_ACTIVE
AP_PENDING_ACTIVE
AP_ACTIVE
AP_PENDING_INACTIVE

ls_summary.act_sess_count
リンクを使用する活動中セッション（端点と中間の両方）の数の合計

ls_summary.det_adj_cp_name
リンク活動化の間に判別される 17 バイトの完全修飾隣接 CP 名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）LS が非活動状態の場合は空になります。

ls_summary.adj_cp_type が AP_NETWORK_NODE、AP_END_NODE、AP_APPN_NODE、または AP_BACK_LEVEL_LEN_NODE のいずれでもない場合、このフィールドは予約されます。

ls_summary.det_adj_cp_type
リンク活動化の間に判別される隣接ノードのタイプ。 以下の値のいずれか 1 つになります。

AP_END_NODE
AP_NETWORK_NODE
AP_LEARN_NODE
AP_VRN

LS が非活動状態の場合は AP_LEARN_NODE になります。

ls_summary.adj_cp_type が AP_NETWORK_NODE、AP_END_NODE、AP_APPN_NODE、または AP_BACK_LEVEL_LEN_NODE のいずれでもない場合、このフィールドは予約されます。

ls_summary.port_name
このリンク・ステーションと関連したポート名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

ls_summary.adj_cp_name
17 バイトの完全修飾隣接制御点名で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) 暗黙リンクの場合は空になります。

ls_summary.adj_cp_type
隣接ノードのタイプ。以下の値のいずれか 1 つになります。

AP_END_NODE
AP_NETWORK_NODE
AP_APPN_NODE
AP_BACK_LEVEL_LEN__NODE
AP_HOST_XID3
AP_HOST_XID0
AP_DSPU_XID
AP_DSPU_NOXID

ls_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

ls_detail.ls_name
リンク・ステーションの名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

ls_detail.det_data.act_sess_count
リンクを使用する活動中セッション（端点と中間の両方）の数の合計

ls_detail.det_data.dlc_type

DLC のタイプ。Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

DEFINE_DLC verb に新規のタイプを指定することにより、追加の DLC タイプを定義することができます。詳細は、52ページの『DEFINE_DLC』 を参照してください。

ls_detail.det_data.state

このリンク・ステーションの状況。このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_NOT_ACTIVE
AP_PENDING_ACTIVE
AP_ACTIVE
AP_PENDING_INACTIVE

ls_detail.det_data.sub_state

このフィールドでは、このリンク・ステーションの状況についてのより詳細な情報を得ることができます。このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_SENT_CONNECT_OUT
AP_PENDING_XID_EXCHANGE
AP_SENT_ACTIVATE_AS
AP_SENT_SET_MODE
AP_ACTIVE
AP_SENT_DEACTIVATE_AS_ORDERLY
AP_SENT_DISCONNECT
AP_WAITING_STATS
AP_RESET

ls_detail.det_data.det_adj_cp_name

リンク活動化の間に判別される 17 バイトの完全修飾隣接制御点名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

ls_summary.adj_cp_type が AP_NETWORK_NODE、AP_END_NODE、AP_APPN_NODE、または AP_BACK_LEVEL_LEN_NODE のいずれでもない場合、このフィールドは予約されます。

ls_detail.det_data.det_adj_cp_type
リンク活動化の間に判別される隣接ノードのタイプ。以下の値のいずれか 1 つになります。

AP_END_NODE
AP_NETWORK_NODE
AP_LEARN_NODE
AP_VRN

ls_summary.adj_cp_type が AP_NETWORK_NODE、AP_END_NODE、AP_APPN_NODE、または AP_BACK_LEVEL_LEN_NODE のいずれでもない場合、このフィールドは予約されます。

ls_detail.det_data.dlc_name
DLC の名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

ls_detail.det_data.dynamic
リンクが (DEFINE_LS コマンドによって) 明示的に定義されたか、または（隣接ノードからの接続要求に応じて、あるいは接続ネットワークを介して別のノードへ動的に接続するために）暗示的にまたは動的に定義されたかを指定します。AP_YES または AP_NO になります。

ls_detail.det_data.migration
隣接ノードが移行レベル・ノード (ロー・エントリー・ネットワーキング (LEN) ノードなど) であるか、または完全 APPN ネットワーク・ノードあるいはエンド・ノードであるかを指定します (AP_YES、AP_NO、または AP_UNKNOWN) 。

ls_detail.det_data.tg_num
TG と関連した数。

ls_detail.det_data.ls_stats.in_xid_bytes
このリンク・ステーションで受信する XID (交換識別) バイト数の合計

ls_detail.det_data.ls_stats.in_msg_bytes
このリンク・ステーションで受信するデータ・バイト数の合計

ls_detail.det_data.ls_stats.in_xid_frames
このリンク・ステーションで受信する XID (交換識別) フレーム数の合計

ls_detail.det_data.ls_stats.in_msg_frames
このリンク・ステーションで受信するデータ・フレーム数の合計

ls_detail.det_data.ls_stats.out_xid_bytes
このリンク・ステーションで送信する XID (交換識別) バイト数の合計

ls_detail.det_data.ls_stats.out_msg_bytes
このリンク・ステーションで送信するデータ・バイト数の合計

ls_detail.det_data.ls_stats.out_xid_frames
このリンク・ステーションで送信する XID (交換識別) フレーム数の合計

ls_detail.det_data.ls_stats.out_msg_frames
このリンク・ステーションで送信するデータ・フレーム数の合計

ls_detail.det_data.ls_stats.in_invalid_sna_frames
このリンク・ステーションで受信する SNA の誤ったフレーム数の合計

ls_detail.det_data.ls_stats.in_session_control_frames
このリンク・ステーションで受信するセッション制御フレーム数の合計

ls_detail.det_data.ls_stats.out_session_control_frames
このリンク・ステーションで送信するセッション制御フレーム数の合計

ls_detail.det_data.ls_stats.echo_rsps
隣接ノードから受信するエコー応答の数。エコー要求が定期的に送信され、隣接ノードへの波及遅延を判別します。

ls_detail.det_data.ls_stats.current_delay
このリンク・ステーションから隣接リンク・ステーションへ最後に送信したテスト信号の送信および戻りにかかった時間 (ミリ秒単位)

ls_detail.det_data.ls_stats.max_delay
このリンク・ステーションから隣接リンク・ステーションへ送信したテスト信号の送信および戻りにかかった最長時間 (ミリ秒単位)

ls_detail.det_data.ls_stats.min_delay
このリンク・ステーションから隣接リンク・ステーションへ送信したテスト信号の送信および戻りにかかった最短時間 (ミリ秒単位)

ls_detail.det_data.ls_stats.max_delay_time
最長の遅延が起こったときに、システム始動以降経過した時間 (1/100 秒単位)

ls_detail.det_data.ls_stats.good_xids
開始以降このリンク・ステーションで起こった正常な XID 交換数の合計

ls_detail.det_data.ls_stats.bad_xids
開始以降このリンク・ステーションで起こった正常でない XID 交換数の合計

ls_detail.det_data.start_time
リンク・ステーションが最後に活動化されたときに（つまり、モード設定コマンドが完了したときに）、システム始動以降経過した時間 (1/100 秒単位)

ls_detail.det_data.stop_time
リンク・ステーションが最後に非活動化されたときに、システム始動以降経過した時間 (1/100 秒単位)

ls_detail.det_data.up_time
このリンク・ステーションがシステム始動以降活動化されていた合計時間 (1/100 秒単位)

ls_detail.det_data.current_state_time
このリンク・ステーションの現行状態での合計時間 (1/100 秒単位)

ls_detail.det_data.deact_cause

リンク・ステーションが最後に非活動化された原因。このフィールドは、以下の値のいずれか 1 つに設定されます。

AP_NONE

リンク・ステーションが非活動化されたことはありません。

AP_DEACT_OPER_ORDERLY

リンク・ステーションは、オペレーターからの正常停止コマンドの結果非活動化されました。

AP_DEACT_OPER_IMMEDIATE

リンク・ステーションは、オペレーターからの即時停止コマンドの結果非活動化されました。

AP_DEACT_AUTOMATIC

リンク・ステーションは、このリンク・ステーションを使用するセッションがもうないなどの理由で自動的に非活動化されました。

AP_DEACT_FAILURE

リンク・ステーションは、障害が原因で非活動化されました。

ls_detail.det_data.hpr_support

この TG でサポートされる高性能経路指定 (HPR) のレベル (AP_NONE、AP_BASE または AP_RTP) 。ローカル・ノードおよび隣接ノードの機能を考慮します。

ls_detail.det_data.anr_label

ローカル・リンクに割り振られた HPR 自動ネットワーク経路指定 (ANR) ラベル

ls_detail.det_data.hpr_link_lvl_error

このリンクの HPR トラフィックに対してリンク・レベルのエラー回復を行うかどうかを指定します。

ls_detail.det_data.ls_role

このリンク・ステーションが前提としたリンク・ステーションの役割を、パートナー・リンク・ステーションとの折衝後に指定します。

ls_detail.det_data.node_id

XID 交換の間に隣接ノードから受信したノード ID。これは 4 バイトの 16 進数ストリングです。

ls_detail.def_data.description

資源の記述 (DEFINE_LS で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

ls_detail.def_data.port_name

このリンク・ステーションと関連したポート名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 リンクが VRN へのリンクである場合、このフィールドには VRN への接続に実際に使用されるポートの名前を指定します (DEFINE_CN verb で指定されたもの) 。

ls_detail.def_data.adj_cp_name
17 バイトの完全修飾隣接制御点名で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）これは、**back_lvl_len_end_node** が AP_NO に設定されない場合、またはリンク・ステーションに関連したポートがスイッチするように定義された場合に定義されます。

ls_detail.def_data.adj_cp_type
隣接ノード・タイプです。

AP_NETWORK_NODE
ノードが APPN ネットワーク・ノードであることを指定します。

AP_END_NODE
ノードが APPN エンド・ノードまたは上位レベル LEN ノードであることを指定します。

AP_APPN_NODE
ノードが APPN ネットワーク・ノード、APPN エンド・ノード、または上位レベル LEN ノードであることを指定します。 ノード・タイプは XID 交換の間に認識されます。

AP_BACK_LEVEL_LEN_NODE
ノードがバック・レベル LEN ノードであることを指定します。

AP_HOST_XID3
ノードがホストであること、およびノード・オペレーター機能がノードからのポーリング XID に形式 3 の XID で応答することを指定します。

AP_HOST_XID0
ノードがホストであること、およびノード・オペレーター機能がノードからのポーリング XID に形式 0 の XID で応答することを指定します。

AP_DSPU_XID
ノードがダウンストリーム PU であること、およびノード・オペレーター機能がリンク活動化中に XID 交換を行うことを指定します。

AP_DSPU_NOXID
ノードがダウンストリーム PU であること、およびノード・オペレーター機能がリンク活動化中に XID 交換を行わないことを指定します。

注: VRN へのリンク・ステーションは常に動的であり、したがって定義されません。

ls_detail.def_data.dest_address.length
隣接ノードでの宛先リンク・ステーションのアドレスの長さ

ls_detail.def_data.dest_address.address
隣接ノードでのリンク・ステーションの宛先アドレス

ls_detail.def_data.auto_act_supp

リンクが START_LS verb によって開始した後に自動的に活動化され、STOP_LS によって停止するように指定します（AP_YES または AP_NO）。

ls_detail.def_data.tg_number

事前に割り当てられた TG 番号 (範囲は 1 から 20) 。この番号は、リンクが活動化されたときそのリンクを示すのに使用されます。ゼロは、TG 番号が事前に割り当てられていないがリンクが活動化されたときに折衝されることを示します。

ls_detail.def_data.limited_resource

このリンクを使用するセッションがない場合に、リンク・ステーションを非活動化するかどうかを指定します。以下の値のいずれか 1 つに設定されます。

AP_NO

リンクは限定資源ではなく、自動的に非活動化されません。

AP_YES または AP_NO_SESSIONS

リンクは限定資源であり、このリンクを使用する活動セッションがない場合は自動的に非活動化されます。

AP_INACTIVITY

リンクは限定資源であり、このリンクを使用する活動セッションがない場合、または **link_deact_timer** フィールドによって指定された時間内にこのリンクを流れたデータがない場合は、自動的に非活動化されます。

ls_detail.def_data.solicit_sscp_sessions

AP_YES は、SSCP と ローカル制御点、および従属型 LU との間のセッションを開始するようにホストに要求します。AP_NO は、このリンクでの SSCP とのセッションを要求しません。

ls_detail.def_data.pu_name

solicit_sscp_sessions が AP_YES に設定される場合、このリンクを使用するローカル PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。**solicit_sscp_sessions** を AP_NO に設定すると、このフィールドは予約されます。

ls_detail.def_data.disable_remote.act

このリンクのリモート活動化をサポートするかどうかを

ls_detail.def_data.dspu_services

solicit_sscp_sessions が AP_NO に設定されている場合に、このリンク上で、ローカル・ノードがダウンストリーム PU に提供するサービスを指定します。以下のいずれか 1 つに設定されます。

AP_PU_CONCENTRATION

ダウンストリーム LU のための PU の集中化を行うローカル・ノード。

AP_DLUR
ローカル・ノードは、ダウンストリーム PU に DLUR サービスを提供します。

AP_NONE
ローカル・ノードは、このダウンストリーム PU に対するサービスを何も提供しません。

solicit_sscp_sessions を AP_YES に設定すると、このフィールドは予約されます。

ls_detail.def_data.dspu_name
ダウンストリーム PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。この名前は、**solicit_sscp_sessions** が AP_NO に設定されているときのみ有効です。

ls_detail.def_data.dlus_name
ダウンストリーム・ノードへのリンクが活動化された時点から DLUR が SSCP サービスを要求する DLUS のノードの名前。これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドをすべてゼロに設定すると、リンクが活動化された時点にグローバルな省略時の DLUS （この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が要求されます。**dlus_name** がゼロに設定されており、グローバルな省略時の DLUS がない場合は、DLUR はリンクが活動化されても SSCP 接続を開始しません。**dspu_services** を AP_DLUR に設定していない場合、このフィールドは予約済みです。

ls_detail.def_data.bkup_dlus_name
このダウンストリーム PU に対するバックアップとして機能する DLUS ノードの名前。これは、すべてゼロに設定するか、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングから構成され、右側が EBCDIC スペースで埋められた 17 バイトのストリングを設定する必要があります。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドをすべてゼロに設定すると、グローバルな省略時のバックアップ DLUS （この DLUS が DEFINE_DLUR_DEFAULTS verb によって定義されている場合）が、この PU 用のバックアップとして使用されます。**dspu_services** を AP_DLUR に設定していない場合、このフィールドは予約済みです。

ls_detail.def_data.hpr_supported
このリンクで HPR がサポートされるかどうかを指定します (AP_YES または AP_NO) 。

ls_detail.def_data.hpr_link_lvl_error
このリンクで HPR リンク・レベルのエラー回復タワーがサポートされるか

どうかを指定します (AP_YES または AP_NO) 。 **hpr_supported** を AP_NO に設定すると、パラメーターが予約されることに注意してください。

ls_detail.def_data.link_deact_timer
限定資源のリンク非活動化タイマー（秒数）。

limited_resource を AP_INACTIVITY に設定すると、このタイマーに定められた時間内にリンクを流れるデータがない場合、そのリンクは自動的に非活動化されます。

ls_detail.def_data.default_nn_server
リンクがエンド・ノードによって自動的に活動化され、ネットワーク・ノード・サーバーへの CP-CP セッションをサポートできるかどうかを指定します（AP_YES または AP_NO）。このフィールドが作用するためには、CP-CP セッションをサポートするようにリンクを定義しなくてはなりません。

ls_detail.def_data.ls_attributes
隣接ノードについての詳細な情報を指定します。

ls_detail.def_data.ls_attributes[0]
ホスト・タイプです。

AP_SNA
標準 SNA ホスト

AP_FNA
FNA (VTAM-F) ホスト

AP_HNA
HNA ホスト

ls_detail.def_data.ls_attributes[1]
バック・レベル LEN ノードへのリンクのためのネットワーク名 CV 抑止オプション （このフィールドは、**adj_cp_type** が AP_BACK_LEVEL_LEN_NODE または AP_HOST_XID3 に設定されない限り無視されます。）

AP_NO
ネットワーク名 CV を XID3 に入れます。

AP_SUPPRESS_CP_NAME
ネットワーク名 CV を XID3 に入れません。

ls_detail.def_data.adj_node_id
隣接ノードの定義済みノード ID

ls_detail.def_data.local_node_id
このリンク・ステーションで XID で送信されるノード ID。4 バイトの 16 進数ストリングです。このフィールドをゼロに設定すると、**node_id** が XID 交換に使用されます。このフィールドをゼロ以外に設定すると、この LS の XID 交換の値を置き換えます。

ls_detail.def_data.cp_cp_sess_support
CP-CP セッションがサポートされるかどうかを指定します (AP_YES または AP_NO) 。

ls_detail_def_data.use_default_tg_chars
DEFINE_LS に指定された TG 特性が、DEFINE_PORT に指定された省略時の特性を優先するために廃棄されたかどうかを指定します (AP_YES または AP_NO) 。このフィールドは、暗黙リンクには適用されません。

ls_detail.def_data.tg_chars
TG の特性 (37ページの『DEFINE_CN』 を参照)

ls_detail.def_data.target_pacing_count
1 以上 32 767 以下の数値で、この TG 上の BIND に対する望ましいペーシング・ウィンドウ・サイズを示します。この数値は、固定バインド・ペーシングが実行される場合のみ有効です。Communications Server は現在この値を使用していないことに注意してください。

ls_detail.def_data.max_send_btu_size
送信できる最大 BTU サイズ

ls_detail.def_data.ls_role
このリンク・ステーションが前提とするべきリンク・ステーションの役割。AP_LS_NEG、AP_LS_PRI、または AP_LS_SEC のいずれか 1 つを指定し、役割を折衝可能、1 次、2 次のなかから選択することができます。このフィールドを AP_USE_PORT_DEFAULTS に設定すると、DEFINE_PORT verb で構成された値を選択することもできます。

ls_detail.def_data.max_ifrm_rcvd
肯定応答のまえに、XID 送信側によって受信できる I フレームの最大数。DEFINE_PORT からの省略時値を使用しなくてはならない場合はゼロに設定します。

ls_detail.def_data.link_spec_data_len
初期設定の間にリンク・ステーションの構成要素へ未変更で渡されるデータの埋込みなしでのバイト長。このデータは LS_DETAIL 構造に連結されます。このデータは 4 バイトの境界まで埋められて終了します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LINK_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_LU_0_TO_3

QUERY_LU_0_TO_3 は、タイプ 0、1、2、および 3 のローカル LU についての情報を戻します。 この情報は、『判別済みデータ』 (実行の間に動的に収集されたデータ)、および 『定義済みデータ』 (アプリケーションにより DEFINE_LU_0_TO_3 で提供されたデータ) として構造化されます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のローカル LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**lu_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。

SNA API クライアントでは、特定のパラメーターのみがサポートされています。

詳細は、*鳴っている電話*を参照してください。

VCB 構造

```
typedef struct query_lu_0_to_3
{
        unsigned short  opcode;            /* verb operation code           */
        unsigned char   reserv2;           /* reserved                      */
        unsigned char   format;            /* format                        */
        unsigned short  primary_rc;        /* primary return code           */
        unsigned long   secondary_rc;      /* secondary return code         */
        unsigned char   *buf_ptr;          /* pointer to buffer             */
        unsigned long   buf_size;          /* buffer size                   */
        unsigned long   total_buf_size;    /* total buffer size required    */
        unsigned short  num_entries;       /* number of entries             */
        unsigned short  total_num_entries; /* total number of entries       */
        unsigned char   list_options;      /* listing options               */
        unsigned char   reserv3;           /* reserved                      */
        unsigned char   pu_name[8];        /* PU name filter                */
        unsigned char   lu_name[8];        /* LU name                       */
        unsigned char   host_attachment;   /* Host attachment filter        */
} QUERY_LU_0_TO_3;
typedef struct lu_0_to_3_summary
{
        unsigned short  overlay_size;      /* size of this entry            */
        unsigned char   pu_name[8];        /* PU name                       */
        unsigned char   lu_name[8];        /* LU name                       */
        unsigned char   description[RD_LEN];
                                           /* resource description          */
        unsigned char   nau_address;       /* NAU address                   */
        unsigned char   lu_sscp_sess_active;
                                           /* Is LU-SSCP session active     */
        unsigned char   appl_conn_active;  /* Is connection to appl active? */
        unsigned char   plu_sess_active;   /* Is PLU-SLU session active     */
        unsigned char   host_attachment;   /* LU's host attachment          */
} LU_0_TO_3_SUMMARY;
```

```
typedef struct lu_0_to_3_detail
{
        unsigned short  overlay_size;      /* size of this entry           */
        unsigned char   lu_name[8];        /* LU name                      */
        unsigned char   reserv1[2];        /* reserved                     */
        LU_0_TO_3_DET_DATA det_data;       /* Determined data              */
        LU_0_TO_3_DEF_DATA def_data;       /* Defined data                 */
} LU_0_TO_3_DETAIL;

typedef struct lu_0_to_3_det_data
{
        unsigned char   lu_sscp_sess_active;
                                           /* Is LU-SSCP session active    */
        unsigned char   appl_conn_active;  /* Application is using LU      */
        unsigned char   plu_sess_active;   /* Is PLU-SLU session active    */
        unsigned char   host_attachment;   /* Host attachment              */
        SESSION_STATS   lu_sscp_stats;     /* LU-SSCP session statistics   */
        SESSION_STATS   plu_stats;         /* PLU-SLU session statistics   */
        unsigned char   plu_name[8];       /* PLU name                     */
        unsigned char   session_id[8];     /* Internal ID of PLU-SLU sess  */

unsigned char   app_spec_det_data[256];    /* Application Specified Data   */
        unsigned char   app_type;          /* Application type             */
        unsigned char   reserva[19];       /* reserved                     */
} LU_0_TO_3_DET_DATA;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;       /* session receive RU size      */
        unsigned short  send_ru_size;      /* session send RU size         */
        unsigned short  max_send_btu_size; /* max send BTU size            */
        unsigned short  max_rcv_btu_size;  /* max rcv BTU size             */
        unsigned short  max_send_pac_win;  /* max send pacing win size     */
        unsigned short  cur_send_pac_win;  /* current send pacing win size */
        unsigned short  max_rcv_pac_win;   /* max receive pacing win size  */
        unsigned short  cur_rcv_pac_win;   /* current receive pacing       */
                                           /* window size                  */
        unsigned long   send_data_frames;  /* number of data frames sent   */
        unsigned long   send_fmd_data_frames;
                                           /* num of FMD data frames sent  */
        unsigned long   send_data_bytes;   /* number of data bytes sent    */
        unsigned long   rcv_data_frames;   /* num data frames received     */
        unsigned long   rcv_fmd_data_frames;
                                           /* num of FMD data frames recvd */
        unsigned long   rcv_data_bytes;    /* number of data bytes received */
        unsigned char   sidh;              /* session ID high byte         */
        unsigned char   sidl;              /* session ID low byte          */
        unsigned char   odai;              /* ODAI bit set                 */
        unsigned char   ls_name[8];        /* Link station name            */
        unsigned char   reserve;           /* reserved                     */
} SESSION_STATS;

typedef struct lu_0_to_3_def_data
{
        unsigned char   description[RD_LEN];
                                           /* resource description         */
        unsigned char   nau_address;       /* LU NAU address               */
        unsigned char   pool_name[8];      /* LU Pool name                 */
```

```
        unsigned char   pu_name[8];         /* PU name                        */
        unsigned char   priority;           /* LU priority                    */
        unsigned char   lu_model;           /* LU model                       */
        unsigned char   reserv2[8];         /* reserved                       */
        unsigned char   app_spec_def_data[16];
                                            /* Application Specified Data     */
} LU_0_TO_3_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_LU_0_TO_3

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_SUMMARY の値も、SNA API クライアントについてサポートされています。

AP_DETAIL

詳細情報を戻します。

指定された **lu_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_FIRST_IN_LIST の値も、SNA API クライアントについてサポートされています。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

lu_name
照会するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

list_options の値は SNA API クライアントでは無視されます。

.

pu_name
PU 名。 このPU を使用する LU のみが戻されます。 すべての LU のリストが必要な場合は、このフィールドをすべて 2 進ゼロに設定しなくてはなりません。

pu_name の値は SNA API クライアントでは無視されます。

host_attachment
ホスト接続のためのフィルター

AP_NONE
すべての LU についての情報を戻します。

AP_NONE は SNA API クライアントに対してサポートされる唯一の値です。

AP_DLUR_ATTACHED
DLUR によってサポートされるすべての LU についての情報を戻します。

AP_DIRECT_ATTACHED
ホスト・システムに直接接続された LU についてのみ情報を戻します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

lu_0_to_3_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

lu_0_to_3_summary.pu_name
この LU が使用しているローカル PU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

lu_0_to_3_summary.pu_name の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.lu_name
照会するローカル LU の名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

lu_0_to_3_summary.description
資源の記述 (DEFINE_LU_0_TO_3 で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

lu_0_to_3_summary.description の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 1 から 255。

lu_0_to_3_summary.nau_address の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.lu_sscp_sess_active
LU-SSCP セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_summary.lu_sscp_sess_active の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.appl_conn_active
アプリケーションが LU を使用中かどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_summary.appl_conn_active の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.plu_sess_active
PLU-SLU セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_summary.plu_sess_active の値は SNA API クライアントでは戻されません。

lu_0_to_3_summary.host_attachment
LU ホスト接続のタイプ。以下のとおりです。

AP_DLUR_ATTACHED
LU は DLUR を使用してホスト・システムに接続されています。

AP_DIRECT_ATTACHED
LU は直接ホスト・システムに接続されています。

lu_0_to_3_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

lu_0_to_3_detail.lu_name
照会するローカル LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

lu_0_to_3_detail.det_data.lu_sscp_sess_active
LU-SSCP セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_detail.det_data.appl_conn_active
この LU がアプリケーションによって現在使用されているかどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_detail.det_data.plu_sess_active
PLU-SLU セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

lu_0_to_3_detail.det_data.host_attachment
LU ホスト接続のタイプ。以下のとおりです。

AP_DLUR_ATTACHED
LU は DLUR を使用してホスト・システムに接続されています。

AP_DIRECT_ATTACHED
LU は直接ホスト・システムに接続されています。

lu_0_to_3_detail.det_data.lu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

lu_0_to_3_detail.det_data.lu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

lu_0_to_3_detail.det_data.lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

lu_0_to_3_detail.det_data.lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

lu_0_to_3_detail.det_data.lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_0_to_3_detail.det_data.lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_0_to_3_detail.det_data.lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_0_to_3_detail.det_data.lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_0_to_3_detail.det_data.lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

lu_0_to_3_detail.det_data.lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

lu_0_to_3_detail.det_data.lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

lu _0_to_3_detail.det_data.lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

lu_0_to_3_detail.det_data.lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

lu_0_to_3_detail.det_data.lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

lu_0_to_3_detail.det_data.lu_sscp_stats.sidh
セッション ID の高位バイト

lu_0_to_3_detail.det_data.lu_sscp_stats.sidl
セッション ID の低位バイト

lu_0_to_3_detail.det_data.lu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、ACTLU の送信側がこのフィールドをゼロに設定し、ACTLU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

lu_0_to_3_detail.det_data.lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

lu_0_to_3_detail.det_data.plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

lu_0_to_3_detail.det_data.plu_stats.send_ru_size
送信のときの最大 RU サイズ

lu_0_to_3_detail.det_data.plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

lu_0_to_3_detail.det_data.plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

lu_0_to_3_detail.det_data.plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

lu_0_to_3_detail.det_data.plu_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

lu_0_to_3_detail.det_data.plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

lu_0_to_3_detail.det_data.plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

lu_0_to_3_detail.det_data.plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

lu_0_to_3_detail.det_data.plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

lu_0_to_3_detail.det_data.plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

lu_0_to_3_detail.det_data.plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

lu_0_to_3_detail.det_data.plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

lu_0_to_3_detail.det_data.plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

lu_0_to_3_detail.det_data.plu_stats.sidh
セッション ID の高位バイト

lu_0_to_3_detail.det_data.plu_stats.sidl
セッション ID の低位バイト

lu_0_to_3_detail.det_data.plu_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

lu_0_to_3_detail.det_data.plu_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

lu_0_to_3_detail.det_data.plu_name
1次 LU 名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。(PLU-SLU セッションが非活動状態の場合、このフィールドは予約されます。)

lu_0_to_3_detail.det_data.session_id
PLU-SLU セッションの 8 バイトの内部識別子

lu_0_to_3_detail.det_data.app_spec_det_data
予約済み

lu_0_to_3_detail.det_data.app_type
予約済み

lu_0_to_3_detail.def_data.description
資源の記述 (DEFINE_LU_0_TO_3 で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

lu_0_to_3_detail.def_data.nau_address
LU のネットワーク・アドレス可能単位アドレス。範囲は 1 から 255。

lu_0_to_3_detail.def_data.pool_name
この LU が属するプールの名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。LU がプールに属していない場合、このフィールドはすべて 2 進ゼロに設定されます。

lu_0_to_3_detail.def_data.pu_name
この LU が使用する PU 名（DEFINE_LS verb で指定したものです)。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

lu_0_to_3_detail.def_data.priority
ホストに送信するときの LU の優先順位。以下の値のいずれか 1 つに設定されます。

AP_NETWORK
AP_HIGHAP_MEDIUM
AP_LOW

lu_0_to_3_detail.def_data.lu_model
LU の型式および番号。以下の値のいずれか 1 つに設定されます。

AP_3270_DISPLAY_MODEL_2
AP_3270_DISPLAY_MODEL_3
AP_3270_DISPLAY_MODEL_4
AP_3270_DISPLAY_MODEL_5
AP_RJE_WKSTN
AP_PRINTER
AP_UNKNOWN

lu_0_to_3_detail.def_data.app_spec_def_data
DEFINE_LU_0_TO_3 からのアプリケーション指定のデータ。Communications Server はこのフィールドを解釈せず、単に格納されて QUERY_LU_0_TO_3 verb に戻されます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_LU_POOL

QUERY_LU_POOL は、プールおよびそれらのプールに属する LU のリストを戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の LU プールについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**pool_name** フィールドおよび **lu_name** フィールドを設定しなくてはなりません。 **lu_name** フィールドをすべてゼロに設定すると、戻される情報は指定されたプールの最初の LU から開始します。 **list_options** フィールドを AP_FIRST_IN_LIST に設定すると、これらのフィールドは両方とも無視されます。

VCB 構造

```
typedef struct query_lu_pool
{
        unsigned short  opcode;              /* verb operation code         */
        unsigned char   reserv2;             /* reserved                    */
        unsigned char   format;              /* format                      */
        unsigned short  primary_rc;          /* primary return code         */
        unsigned long   secondary_rc;        /* secondary return code       */
        unsigned char   *buf_ptr;            /* pointer to buffer           */
        unsigned long   buf_size;            /* buffer size                 */
        unsigned long   total_buf_size;      /* total buffer size required  */
        unsigned short  num_entries;         /* number of entries           */
        unsigned short  total_num_entries;   /* total number of entries     */
        unsigned char   list_options;        /* listing options             */
        unsigned char   reserv3;             /* reserved                    */
        unsigned char   pool_name[8];        /* pool name                   */
        unsigned char   lu_name[8];          /* LU name                     */
} QUERY_LU_POOL;
typedef struct lu_pool_summary
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   pool_name[8];        /* pool name                   */
        unsigned char   description[RD_LEN]; /* resource description        */
        unsigned short  num_active_lus;      /* num of currently active LUs */
} LU_POOL_SUMMARY;
typedef struct lu_pool_detail
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   pool_name[8];        /* pool name                   */
        unsigned char   description[RD_LEN]; /* resource description        */
        unsigned char   lu_name[8];          /* LU name                     */
        unsigned char   lu_sscp_sess_active; /* Is LU-SSCP session active   */
        unsigned char   appl_conn_active;    /* Is SSCP connection open     */
        unsigned char   plu_sess_active;     /* Is PLU-SLU session active   */
} LU_POOL_DETAIL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_LU_POOL

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **pool_name** および **lu_name** の組合せ (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

pool_name

LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

lu_name

LU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。 すべて 2 進ゼロに設定すると、指定されたプールに属する LU がプールの先頭からリストされます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
戻されるディレクトリー項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

lu_pool_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

lu_pool_summary.pool_name
指定された LU が属する LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 （要求時にこのフィールドを指定し、かつ **lu_name** フィールドをすべて 2 進ゼロに設定すると、そのプールの LU のみが戻されることに注意してください。）

lu_pool_summary.description
LU プールの記述 (DEFINE_LU_POOL で指定されたもの)

lu_pool_detail.num_active_lus
指定されたプール内で、活動状態の LU-SSCP セッションがある LU の数

lu_pool_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

lu_pool_detail.pool_name
指定された LU が属する LU プールの名前。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 （要求時にこのフィールドを指定し、かつ **lu_name** フィールドをすべて 2 進ゼロに設定すると、そのプールの LU のみが戻されることに注意してください。）

lu_pool_detail.description
LU の記述 (DEFINE_LU_0_TO_3 で指定されたもの)

lu_pool_detail.lu_name
プールに属する LU の LU 名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。この名前をすべてゼロに設定すると、指定されたプールが空であることを示します。

lu_pool_detail.lu_sscp_sess_active
LU-SSCP セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

lu_pool_detail.appl_conn_active
LU セッションがアプリケーションによって現在使用されているかどうかを指定します (AP_YES または AP_NO) 。

lu_pool_detail.plu_sess_active
PLU-SLU セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LIST_OPTION

AP_INVALID_POOL_NAME
AP_INVALID_LU_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_MDS_APPLICATION

QUERY_MDS_APPLICATION は、MDS レベル・メッセージ用に登録されているアプリケーションのリストを戻します。

REGISTER_MS_APPLICATION verb の発行によるアプリケーションの登録は、547 ページの『第14章 管理サービス verb』 に記載してあります。

特定のアプリケーションについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**application** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

VCB 構造

```
typedef struct query_mds_application
{
        unsigned short  opcode;             /* verb operation code        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* primary return code        */
        unsigned long   secondary_rc;       /* secondary return code      */
        unsigned char   *buf_ptr;           /* pointer to buffer          */
        unsigned long   buf_size;           /* buffer size                */
        unsigned long   total_buf_size;     /* total buffer size required */
        unsigned short  num_entries;        /* number of entries          */
        unsigned short  total_num_entries;  /* total number of entries    */
        unsigned char   list_options;       /* listing options            */
        unsigned char   reserv3;            /* reserved                   */
        unsigned char   application[8];     /* application                */
} QUERY_MDS_APPLICATION;
typedef struct mds_application_data
{
        unsigned short  overlay_size;       /* size of this entry         */
        unsigned char   application[8];     /* application name           */
        unsigned short  max_rcv_size;       /* max data size application  */
                                            /* can receive                */
        unsigned char   reserva[20];        /* reserved                   */
} MDS_APPLICATION_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_MDS_APPLICATION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケ

ーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。指定された **application** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

application
アプリケーション名。 8 バイトの英数字のタイプ A の EBCDIC 文字ストリングです。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

mds_application_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mds_application_data.application
登録されたアプリケーションの名前。 8 バイトの英数字のタイプ A の EBCDIC 文字ストリングです。

mds_application_data.max_rcv_size
アプリケーションが 1 つの「かたまり」として受信できる最大バイト数（アプリケーションが MDS で登録するときに指定されます)。 MDS レベルのアプリケーション登録の詳細は、547ページの『第14章 管理サービス verb』を参照してください。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_APPLICATION_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_MDS_STATISTICS

QUERY_MDS_STATISTICS は管理サービスの統計を戻します。 この verb を使用して MDS 経路指定トラフィックのレベルを判断することができます。

VCB 構造

```
typedef struct query_mds_statistics
{
        unsigned short  opcode;             /* verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* primary return code         */
        unsigned long   secondary_rc;       /* secondary return code       */
        unsigned long   alerts_sent;        /* number of alert sends       */
        unsigned long   alert_errors_rcvd;  /* error messages received     */
                                            /* for alert sends             */
        unsigned long   uncorrelated_alert_errors;
                                            /* uncorrelated alert          */
                                            /* errors received             */
        unsigned long   mds_mus_rcvd_local; /* number of MDS_MUs received  */
                                            /* from local applications     */
        unsigned long   mds_mus_rcvd_remote;
                                            /* number of MDS_MUs received  */
                                            /* from remote applications    */
        unsigned long   mds_mus_delivered_local;
                                            /* num of MDS_MUs delivered    */
                                            /* to local applications       */
        unsigned long   mds_mus_delivered_remote;
                                            /* num of MDS_MUs              */
                                            /* delivered to remote appls   */
        unsigned long   parse_errors;       /* number of MDS_MUs received  */
                                            /* with parse errors           */
        unsigned long   failed_deliveries;  /* number of MDS_MUs where     */
                                            /* delivery failed             */
        unsigned long   ds_searches_performed;
                                            /* number of DS searches done  */
        unsigned long   unverified_errors;  /* number of unverified errors */
        unsigned char   reserva[20];        /* reserved                    */
} QUERY_MDS_STATISTICS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_MDS_STATISTICS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

alerts_sent
MDS トランスポート・システムを使用して送信される、ローカルで発信されたアラートの数

alert_errors_rcvd
MDS によって受信される、アラートを含むメッセージの送達障害を示すエラー・メッセージの数。

uncorrelated_errors_rcvd
MDS によって受信される、アラートを含むメッセージの送達障害を示すエラー・メッセージの数。 MDS 送信アラート待ち行列でエラー・メッセージとアラートとを関連付けることができなかったときに、送達障害が起こります。 MDS は固定サイズの待ち行列を保守し、問題判別フォーカル・ポイントに送信されたアラートをこの待ち行列に入れます。待ち行列が最大サイズに達すると、最も古いアラートが廃棄され、新しいアラートに置き換えられます。送達エラー・メッセージを受け取ると、MDS は、問題判別フォーカル・ポイントが復元されるまで、待ち行列に入れたアラートを保管できるように、エラー・メッセージとアラートを関連付けようとします。

注: **alert_errors_rcvd** および **uncorrelated_errors_rcvd** の 2 つの数値は常に、送信アラート待ち行列のサイズに適合できる数値になります。 **uncorrelated_errors_rcvd** が時間を越えて増大する場合は、送信アラート待ち行列が小さ過ぎることを示します。

mds_mus_rcvd_local
ローカル・アプリケーションから受信した MDS_MU の数

mds_mus_rcvd_remote
MDS_RECEIVE および MSU_HANDLER トランザクション・プログラムを使用してリモート・ノードから受信した MDS_MU の数

mds_mus_delivered_local
ローカル・アプリケーションに正常に送達された MDS_MU の数

mds_mus_delivered_remote
MDS_SEND トランザクション・プログラムを使用してリモート・ノードに正常に送達された MDS_MU の数

parse_errors
受信したがヘッダー形式にエラーがあった MDS_MU の数

failed_deliveries
このノードが送達に失敗した MDS_MU の数

ds_searches_performed
予約済み

unverified_errors
予約済み

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_MODE

QUERY_MODE は、ローカル LU によって特定のパートナー LU とともに使用されたモードについての情報を戻します。この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のモードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**mode_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 **lu_name** (または **lu_alias**) フィールドおよび **plu_alias** (または **fqplu_name**) フィールドを常に設定しなくてはならないことに注意してください。 **lu_name** が非ゼロの場合、**lu_name** が **lu_alias** に優先して使用されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

リストには、**lu_name** (または **lu_alias**) によって指定されたローカル LU についての情報のみが含まれます。 このリストはまず **fqplu_name** で、次に **mode_name** で配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

plu_alias をすべてゼロに設定すると **fqplu_name** の値が使用され、それ以外の場合は **plu_alias** が常に使用されて **fqplu_name** は無視されます。

戻されるモードのリストを、それらのモードに現在活動中のセッションがあるかどうかによってフィルター処理することができます。 フィルター処理を希望する場合は、**active_sessions** フィールドを AP_YES に設定しなくてはなりません（それ以外の場合このフィールドは AP_NO に設定しなくてはなりません)。 この verb は、一度モードが開始してローカル LU によってパートナー LU とともに使用されると、判別される情報を戻します。 QUERY_MODE_DEFINITION verb は、定義情報のみを戻します。

VCB 構造

```
typedef struct query_mode
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   *buf_ptr;            /* pointer to buffer            */
        unsigned long   buf_size;            /* buffer size                  */
        unsigned long   total_buf_size;      /* total buffer size required   */
        unsigned short  num_entries;         /* number of entries            */
        unsigned short  total_num_entries;   /* total number of entries      */
        unsigned char   list_options;        /* listing options              */
        unsigned char   reserv3;             /* reserved                     */
        unsigned char   lu_name[8];          /* LU name                      */
        unsigned char   lu_alias[8];         /* LU alias                     */
        unsigned char   plu_alias[8];        /* partner LU alias             */
        unsigned char   fqplu_name[17];      /* fully qualified partner      */
```

```
                                        /* LU name                      */
        unsigned char   mode_name[8];   /* mode name                    */
        unsigned char   active_sessions;/* active sessions only filter  */
} QUERY_MODE;
typedef struct mode_summary
{
        unsigned short  overlay_size;   /* size of this entry           */
        unsigned char   mode_name[8];   /* mode name                    */
        unsigned char   description[RD_LEN];
                                        /* resource description         */
        unsigned short  sess_limit;     /* current session limit        */
        unsigned short  act_sess_count; /* curr active sessions count   */
        unsigned char   fqplu_name[17]; /* partner LU name              */
        unsigned char   reserv1[3];     /* reserved                     */
} MODE_SUMMARY;
typedef struct mode_detail
{
        unsigned short  overlay_size;   /* size of this entry           */
        unsigned char   mode_name[8];   /* mode name                    */
        unsigned char   description[RD_LEN];
                                        /* resource description         */
        unsigned short  sess_limit;     /* session limit                */
        unsigned short  act_sess_count; /* currently active sess count  */
        unsigned char   fqplu_name[17]; /* partner LU name              */
        unsigned char   reserv1[3];     /* reserved                     */
        unsigned short  min_conwinners_source;
                                        /* min conwinner sess limit     */
        unsigned short  min_conwinners_target;
                                        /* min conloser limit           */
        unsigned char   drain_source;   /* drain source?                */
        unsigned char   drain_partner;  /* drain partner?               */
        unsigned short  auto_act;       /* auto activated conwinner     */
                                        /* session limit                */
        unsigned short  act_cw_count;   /* active conwinner sess count  */
        unsigned short  act_cl_count;   /* active conloser sess count   */
        unsigned char   sync_level;     /* synchronization level        */
        unsigned char   default_ru_size;/* default RU size to maximize  */
                                        /* performance                  */
        unsigned short  max_neg_sess_limit; /* max negotiated session limit */
        unsigned short  max_rcv_ru_size;/* max receive RU size          */
        unsigned short  pending_session_count;
                                        /* pending sess count for mode  */
        unsigned short  termination_count;  /* termination count for mode   */
        unsigned char   implicit;       /* implicit or explicit entry   */
        unsigned char   reserva[15];    /* reserved                     */
} MODE_DETAIL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_MODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr
リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **lu_name** (**lu_name** をすべてゼロに設定する場合は **lu_alias**)、**plu_alias** (**plu_alias** をすべてゼロに設定する場合は **fqplu_name**)、および **mode_name** の組合せ (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。パートナー LU の索引を指定する場合、他のパートナー LU についての情報は、可能であればこのリストに含まれます。

AP_FIRST_IN_LIST
plu_alias および **fqplu_name** をすべてゼロに設定すると、戻されるリストはそのリストの最初のパートナー LU から開始し、**mode_name** の索引は無視されます。**plu_alias** または **fqplu_name** のいずれかを指定すると、リストはこの索引から開始しますが、**mode_name** の索引値は無視され、戻されるリストはそのリストの最初のモード項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

lu_name
LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。このフィールドをすべてゼロに設定すると、**lu_alias** フィールドが索引の判別に使用されます。

lu_alias

ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** と **lu_alias** の両方がすべてゼロに設定されている場合には、制御点と関連した LU（省略時 LU）が使用されます。

plu_alias

パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロの設定されている場合、索引を判別するため、**fqplu_name** フィールドが使用されます。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

mode_name

モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

active_sessions

活動状態のセッションでのフィルター。 戻されるモードを、それらのモードに現在活動中のセッションがあるかどうかによってフィルター処理するかどうかを指定します (AP_YES または AP_NO) 。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries

実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

mode_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mode_summary.mode_name
モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

mode_summary.description
資源の記述 (DEFINE_MODE で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

mode_summary.sess_limit
現行セッションの限度

mode_summary.act_sess_count
このモードを使用する活動中セッション数の合計。**active_sessions** フィルターが AP_YES に設定されている場合には、このフィールドは常にゼロより大きくなります。

mode_summary.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

mode_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mode_detail.mode_name
モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

mode_detail.description
資源の記述 (DEFINE_MODE で指定されたもの) 。

mode_detail.sess_limit
現行セッションの限度

mode_detail.act_sess_count
このモードを使用する活動中セッション数の合計。**active_sessions** フィルターが AP_YES に設定されている場合には、このフィールドは常にゼロより大きくなります。

mode_detail.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオ

ドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

mode_detail.min_conwinners_source
ローカル LU が回線争奪勝者（または ファースト・スピーカー）であるセッションの最小数を指定します。

mode_detail.min_conwinners_target
ローカル LU が回線争奪敗者（または 送信権要求者）であるセッションの最小数を指定します。

mode_detail.drain_source
セッション限度が変更またはリセットされたとき、セッションを非活動化する前に待ちセッションの要求をローカル LU が満たすかどうかを指定します (AP_NO または AP_YES) 。

mode_detail.drain_partner
セッション限度が変更またはリセットされたとき、セッションを非活動化する前に待ちセッションの要求をパートナー LU が満たすかどうかを指定します (AP_NO または AP_YES) 。

mode_detail.auto_act
セッション数変更のパートナー LU との交換に続いて自動的に活動化される回線争奪勝者セッションの数

mode_detail.act_cw_count
このモードを使用する、活動状態の回線争奪勝者 (または ファースト・スピーカー) セッションの数。(ローカル LU は、これらのセッションの 1 つを使用する前に送信権を要求する必要はありません。)

mode_detail.act_cl_count
このモードを使用する、活動状態の回線争奪敗者 (または 送信権要求者) セッションの数。(ローカル LU は、これらのセッションの 1 つを使用する前に送信権を要求しなくてはなりません。)

mode_detail.sync_level
モードによってサポートされる同期化レベルを指定します (AP_NONE、AP_CONFIRM、または AP_SYNCPT) 。

mode_detail.default_ru_size
最大 RU サイズの省略時の上限を使用するかどうかを指定します。このパラメーターの値が AP_YES の場合、**define_mode** で指定された **mode_chars.max_ru_size_upp** フィールドが無視され、最大 RU サイズの上限は、リンク BTU サイズから TH および RH のサイズを引いたものに設定されます。

AP_YES
AP_NO

mode_detail.max_neg_sess_limit
折衝可能な最大セッション限度。ローカル LU が、CNOS 処理中に宛先 LU として使用できるモード名についてのセッションの最大限度を指定します。

mode_detail.max_rcv_ru_size
受信のときの最大 RU サイズ

mode_detail.pending_session_count
保留のセッションの数を指定します（完了するためにセッション活動化を待機中のもの）。

mode_detail.termination_count
前の CNOS verb によってモード・セッション限度がゼロにリセットされた場合は、それらのセッションを使用中の会話があったか、またはそれらのセッションの使用を待っている会話があった可能性があります。このフィールドの数値は、このようなセッションでまだ非活動化されていなかったものがいくつあるかを示します。

mode_detail.implicit
暗黙 (AP_YES) 定義または明示 (AP_NO) 定義のために項目が置かれたかどうかを指定します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_MODE_NAME

AP_INVALID_PLU_NAME
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_MODE_DEFINITION

QUERY_MODE_DEFINITION は、直前に DEFINE_MODE verb で渡された情報と、SNA 定義の省略時モードについての情報の両方を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のモードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**mode_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **mode_name** で配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。

AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

この verb は、定義情報のみを戻します。 QUERY_MODE verb は、一度モードが開始してローカル LU によってパートナー LU とともに使用されると、判別される情報を戻します。

VCB 構造

```
{
typedef struct query_mode_definition
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   *buf_ptr;           /* pointer to buffer             */
        unsigned long   buf_size;           /* buffer size                   */
        unsigned long   total_buf_size;     /* total buffer size required    */
        unsigned short  num_entries;        /* number of entries             */
        unsigned short  total_num_entries;  /* total number of entries       */
        unsigned char   list_options;       /* listing options               */
        unsigned char   reserv3;            /* reserved                      */
        unsigned char   mode_name[8];       /* mode name                     */
} QUERY_MODE_DEFINITION;

typedef struct mode_def_summary
{
        unsigned short  overlay_size;       /* size of this entry            */
        unsigned char   mode_name[8];       /* mode name                     */
        unsigned char   description[RD_LEN];
                                            /* resource description          */
} MODE_DEF_SUMMARY;
```

```
typedef struct mode_def_detail
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   mode_name[8];       /* mode name                    */
        MODE_CHARS      mode_chars;         /* mode characteristics         */
} MODE_DEF_DETAIL;
typedef struct mode_chars
{
        unsigned char   description[RD_LEN];
                                            /* resource description         */
        unsigned short  max_ru_size_upp;    /* max RU size upper bound      */
        unsigned char   receive_pacing_win; /* receive pacing window        */
        unsigned char   default_ru_size;    /* default RU size to maximize  */
                                            /* performance                  */
        unsigned short  max_neg_sess_lim;   /* max negotiable session limit */
        unsigned short  plu_mode_session_limit;
                                            /* LU-mode session limit        */
        unsigned short  min_conwin_src;     /* min source contention winner */
                                            /* sessions                     */
        unsigned char   cos_name[8];        /* class-of-service name        */
        unsigned char   cryptography;       /* cryptography                 */
        unsigned char   reserv1;            /* reserved                     */
        unsigned short  auto_act;           /* initial auto-activation count*/
        unsigned char   reserv2[6];         /* reserved                     */
} MODE_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_MODE_DEFINITION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **mode_name** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

mode_name
モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

mode_def_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mode_def_summary.mode_name
8 バイトのモード名。セッションのグループのネットワーク特性を指定します。

mode_def_summary.description
資源の記述 (DEFINE_MODE で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

mode_def_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mode_def_detail.mode_name
モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

mode_def_detail.mode_chars.description
資源の記述 (DEFINE_MODE で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

mode_def_detail.mode_chars.max_ru_size_upp
セッションでこのモード名で使用される最大 RU サイズの上限。

mode_def_detail.mode_chars.receive_pacing_win
固定ペーシングが使用されているときのセッションのセッション・ペーシング・ウィンドウを指定します。

mode_def_detail.mode_chars.default_ru_size
最大 RU サイズの省略時の上限を使用するかどうかを指定します。 このパラメーターを AP_YES に指定すると、**max_ru_size_upp** は無視されます。

AP_YES
AP_NO

mode_def_detail.mode_chars.max_neg_sess_lim
折衝可能な最大セッション限度。 指定したモード名のローカル LU とパートナー LU との間で許可できるセッションの最大数を折衝するのに使用される値です。

mode_def_detail.mode_chars.plu_mode_session_limit
このモードで最初に折衝するためのセッション限度。 この値は優先されるセッション限度を示し、暗黙 CNOS に使用されます。

範囲は 0 から 32 767 です。

mode_def_detail.mode_chars.min_conwin_src
このモードを使用するローカル LU によって活動化可能な回線争奪勝者セッションの最小数。

範囲は 0 から 32 767 です。

mode_def_detail.mode_chars.cos_name
サービス・クラス名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

mode_def_detail.mode_chars.cryptography
予約済み

mode_def_detail.mode_chars.auto_act
このモードで自動活動化されるセッションの数を指定します。この値は暗黙 CNOS のため に使用されます。 範囲は 0 から 32767 です。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_MODE_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_MODE_TO_COS_MAPPING

QUERY_MODE_TO_COS_MAPPING は、COS マッピングへのモードについての情報を戻します。

この情報は、形式化されたリストとして戻されます。 特定のモードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**mode_name** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **mode_name** で配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です)。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

DEFINE_MODE を使用して、不明モードをマップしている省略時 COSを上書き変更した場合、QUERY_MODE_TO_COS_MAPPING は、やはり空の **mode_name**（すべてゼロ）および省略時 COS で項目を戻します。 この項目は、配列の中で先頭に来ます。

VCB 構造

```
typedef struct query_mode_to_cos_mapping
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   mode_name[8];       /* mode name                    */
} QUERY_MODE_TO_COS_MAPPING;

typedef struct mode_to_cos_mapping_data
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   mode_name[8];       /* mode name                    */
        unsigned char   cos_name[8];        /* COS name                     */
        unsigned char   reserva[20];        /* reserved                     */
} MODE_TO_COS_MAPPING_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_MODE_TO_COS_MAPPING

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **mode_name** (下記のパラメーターを参照してください) は、戻される実情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

mode_name

モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。 このフィールドをすべてゼロに設定すると、省略時 COS のために項目を指定することができます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

mode_to_cos_mapping_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

mode_to_cos_mapping_data.mode_name
8 バイトのモード名。セッションのグループのネットワーク特性を指定します。このモード名がすべてゼロに設定されている場合、省略時 COS としての項目を示します。

mode_to_cos_mapping_data.cos_name
モード名と関連したサービス・クラスの名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_MODE_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_NMVT_APPLICATION

QUERY_NMVT_APPLICATION は、あらかじめ REGISTER_NMVT_APPLICATION verb を発行することによってネットワーク管理ベクトル・トランスポート (NMVT) レベルのメッセージのために登録したアプリケーションのリストを戻します（詳細は、547ページの『第14章 管理サービス verb』 を参照してください)。

この情報は、リストとして戻されます。 特定のアプリケーションについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**application** フィールドを設定しなくてはなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

VCB 構造

```
typedef struct query_nmvt_application
{
        unsigned short  opcode;                 /* verb operation code           */
        unsigned char   reserv2;                /* reserved                      */
        unsigned char   format;                 /* format                        */
        unsigned short  primary_rc;             /* primary return code           */
        unsigned long   secondary_rc;           /* secondary return code         */
        unsigned char   *buf_ptr;               /* pointer to buffer             */
        unsigned long   buf_size;               /* buffer size                   */
        unsigned long   total_buf_size;         /* total buffer size required    */
        unsigned short  num_entries;            /* number of entries             */
        unsigned short  total_num_entries;      /* total number of entries       */
        unsigned char   list_options;           /* listing options               */
        unsigned char   reserv3;                /* reserved                      */
        unsigned char   application[8];         /* application                   */
} QUERY_NMVT_APPLICATION;
typedef struct nmvt_application_data
{
        unsigned short  overlay_size;           /* size of this entry            */
        unsigned char   application[8];         /* application name              */
        unsigned short  ms_vector_key_type;     /* MS vector key accepted        */
                                                /* by appl                       */
        unsigned char   conversion_required;    /* conversion to MDS_MU required */
        unsigned char   reserv[5];              /* reserved                      */
        unsigned char   reserva[20];            /* reserved                      */
} NMVT_APPLICATION_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_NMVT_APPLICATION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。指定された **application** (後に示すパラメーターを参照) は、実際に戻される情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

application

アプリケーション名。 この名前は、8 バイトの英数字のタイプ A の EBCDIC 文字ストリング、またはすべて EBCDIC ゼロです。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries

実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 これは、**num_entries** より大きくなることがあります。

nmvt_application_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

nmvt_application_data.application
登録されたアプリケーションの名前。 8 バイトの英数字のタイプ A の EBCDIC 文字ストリングです。

nmvt_application_data.ms_vector_key_type
アプリケーションによって受け入れられた管理サービス・ベクトル・キー 。アプリケーションは、NMVT メッセージのために登録されると、どの管理サービス・ベクトル・キーを受け入れるかを指定します。 NMVT アプリケーションの登録についての詳細は、547ページの『第14章 管理サービス verb』を参照してください。

nmvt_application_data.conversion_required
登録されたアプリケーションが、メッセージについて NMVT から MDS_MU 形式（AP_YES または AP_NO）への変換を必要とするかどうかを指定します。 アプリケーションは、NMVT メッセージのために登録されると、この変換が必要かどうかを指定します。NMVT アプリケーションの登録についての詳細は、547ページの『第14章 管理サービス verb』を参照してください。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_APPLICATION_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_NN_TOPOLOGY_NODE

各ネットワーク・ノードは、ネットワーク・トポロジー・データベースを保守します。このデータベースは、ネットワーク内のネットワーク・ノード、VRN、およびネットワーク・ノード対ネットワーク・ノード TG についての情報を保持しています。

QUERY_NN_TOPOLOGY_NODE は、このデータベース内のネットワーク・ノードおよび VRN 項目についての情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。特定のノードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**node_name**、**node_type**、および **frsn** フィールドを設定しなくてはなりません。 それ以外の場合（**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合）、これらのフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、**node_name**、**node_type**、および **frsn** 順になっています。**node_name** は、まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIBの配列順序と同一です）。**node_type** フィールドは、AP_NETWORK_NODE, AP_VRN の配列に従います。 **frsn** は数値順に配列されます。

AP_LIST_INCLUSIVE が選択されている場合、戻されるリストは、その名前の最初の有効なレコードから開始します。

AP_LIST_FROM_NEXT を選択すると、リストは、指定された名前に続く名前をもつ最初の有効なレコードから開始します。

frsn フィールド（フロー縮小の順序番号）が非ゼロ値に設定されている場合、この値よりも高い FRSN をもつデータベース項目だけが戻されます。 これを利用して、まずノードの現行 FRSN を入手することによって、整合性のあるトポロジー・データベースをいくつかの かたまり に分けて戻すことができます。 以下のように作業します。

1. QUERY_NODE を発行します。これがノードの現行 FRSN を戻します。
2. すべてのデータベース項目を かたまり に分けて入手するため、必要なだけの QUERY_NN_TOPOLOGY_NODE（FRSN をゼロに設定して）を発行します。
3. 再度、QUERY_NODE を発行し、ステップ 1 で戻された FRSN と新規の FRSN とを比較します。
4. 2 つの FRSN が異なっていれば、データベースは変更されています。そのため、ステップ 1 で指定した FRSN よりも 1 つ大きく設定した FRSN で QUERY_NN_TOPOLOGY_NODE を発行します。

VCB 構造

```
typedef struct query_nn_topology_node
{
        unsigned short  opcode;             /* verb operation code          */
```

```
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   node_name[17];      /* network qualified node name  */
        unsigned char   node_type;          /* node type                    */
        unsigned long   frsn;               /* flow reduction sequence num  */
} QUERY_NN_TOPOLOGY_NODE;
```

注: frsn フィールドが非ゼロ値に設定されている場合、指定した FRSN よりも大きい FRSN をもったノード項目だけが戻されます。 **frsn** フィールドがゼロに設定されている場合は、すべてのノード項目が戻されます。

```
typedef struct nn_topology_node_summary
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   node_name[17];      /* network qualified node name  */
        unsigned char   node_type;          /* node type                    */
} NN_TOPOLOGY_NODE_SUMMARY;

typedef struct nn_topology_node_detail
{
        unsigned short  overlay_size;       /* size of this entry           */
        unsigned char   node_name[17];      /* network qualified node name  */
        unsigned char   node_type;          /* node type                    */
        unsigned short  days_left;          /* days left until entry purged */
        unsigned char   reserv1[2];         /* reserved                     */
        unsigned long   frsn;               /* flow reduction sequence num  */
        unsigned long   rsn;                /* resource sequence number     */
        unsigned char   rar;                /* route additional resistance  */
        unsigned char   status;             /* node status                  */
        unsigned char   function_support;   /* function support             */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   reserva[20];        /* reserved                     */
} NN_TOPOLOGY_NODE_DETAIL;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_NN_TOPOLOGY_NODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケ

ーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **node_name**、**node_type**、および **frsn**（下記のパラメーターを参照してください）の組合せは、戻される実情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

node_name
ネットワーク・トポロジー・データベースからのネットワーク修飾ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

node_type
ノードのタイプ。 以下の値のいずれか 1 つになります。

AP_NETWORK_NODE
AP_VRN

node_type が不明の場合、AP_LEARN_NODE を指定しなければなりません。

frsn フロー縮小順序番号。 これが非ゼロの場合、この値より大きいか等しい FRSN をもったノードだけが戻されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

nn_topology_node_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

nn_topology_node_summary.node_name
ネットワーク・トポロジー・データベースからのネットワーク修飾ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

nn_topology_node_summary.node_type
ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

nn_topology_node_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

nn_topology_node_detail.node_name
ネットワーク・トポロジー・データベースからのネットワーク修飾ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

nn_topology_node_detail.node_type
ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

nn_topology_node_detail.days_left
トポロジー・データベースからこのノード項目を削除するまでの日数。 ローカル・ノード項目に対しては、これをゼロに設定します（この項目は決して削除されません）。

nn_topology_node_detail.frsn
フロー縮小順序番号。 ローカル・ノードで最後にこの資源を更新した時刻を示します。

nn_topology_node_detail.rsn
資源順序番号。 この資源を所有しているネットワーク・ノードによって割り当てられます。

nn_topology_node_detail.rar
ノード経路の追加抵抗

nn_topology_node_detail.status
ノードの状況を指定します。これは、AP_UNCONGESTED かまたは、以下の値の 1 つ以上を一緒に論理OR演算したものであることがあります。

AP_CONGESTED
ISR セッションの数は、**isr_sessions_upper_threshold** よりも大きいです。

AP_ERR_DEPLETED
エンドポイント・セッションの数が指定された最大値に達しました。

AP_IRR_DEPLETED
ISR セッションの数が最大に達しました。

AP_QUIESCING
STOP_NODE 、またはタイプ AP_QUIESCE か AP_QUIESCE_ISR が発行されました。

nn_topology_node_detail.function_support
どの機能をサポートするかを指定します。 以下の値のうちの 1 つまたは複数になります。

AP_BORDER_NODE
ボーダー・ノード機能をサポートします。

AP_CDS
ノードは、中央ディレクトリー・サーバー機能をサポートします。

AP_GATEWAY
ノードは、ゲートウェイ・ノードです。(この機能は、まだアーキテクチャー定義が済んでいません)。

AP_ISR
ノードは、中間セッションの経路指定をサポートします。

AP_HPR
ノードは、高性能経路指定の基本機能をサポートします。

AP_RTP_TOWER
ノードは、HPR の RTP タワーをサポートします。

AP_CONTROL_OVER_RTP_TOWER
ノードは、RTP タワー上の制御フローをサポートします。

注: AP_CONTROL_OVER_RTP_TOWER は、 AP_HPR と AP_RTP_TOWER の両方の設定に対応します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_NODE

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_NN_TOPOLOGY_STATS

QUERY_NN_TOPOLOGY_STATS は、トポロジー・データベースについての統計情報を戻します。これは、ネットワーク・ノードにおいてのみ発行されます。

VCB 構造

```
typedef struct query_nn_topology_stats
{
         unsigned short  opcode;               /* verb operation code          */
         unsigned char   reserv2;              /* reserved                     */
         unsigned char   format;               /* format                       */
         unsigned short  primary_rc;           /* primary return code          */
         unsigned long   secondary_rc;         /* secondary return code        */
         unsigned long   max_nodes;            /* max num of nodes in database */
         unsigned long   cur_num_nodes;        /* current number of nodes in   */
                                               /* database                     */
         unsigned long   node_in_tdus;         /* number of TDUs received      */
         unsigned long   node_out_tdus;        /* number of TDUs sent          */
         unsigned long   node_low_rsns;        /* node updates received with   */
                                               /* low RSNs                     */
         unsigned long   node_equal_rsns;      /* node updates in with         */
                                               /* equal RSNs                   */
         unsigned long   node_good_high_rsns;  /* node updates in with         */
                                               /* high RSNs                    */
         unsigned long   node_bad_high_rsns;   /* node updates in with         */
                                               /* high and odd RSNs            */
         unsigned long   node_state_updates;   /* number of node updates sent  */
         unsigned long   node_errors;          /* number of node entry         */
                                               /* errors found                 */
         unsigned long   node_timer_updates;   /* number of node records built */
                                               /* due to timer updates         */
         unsigned long   node_purges;          /* num node records purged      */
         unsigned long   tg_low_rsns;          /* TG updates received with     */
                                               /* low RSNs                     */
         unsigned long   tg_equal_rsns;        /* TG updates in with equal RSNs */
         unsigned long   tg_good_high_rsns;    /* TG updates in with high RSNs */
         unsigned long   tg_bad_high_rsns;     /* TG updates in with high      */
                                               /* and odd RSNs                 */
         unsigned long   tg_state_updates;     /* number of TG updates sent    */
         unsigned long   tg_errors;            /* number of TG entry errors    */
                                               /* found                        */
         unsigned long   tg_timer_updates;     /* number of node records       */
                                               /* built due to timer updates   */
         unsigned long   tg_purges;            /* num node records purged      */
         unsigned long   total_route_calcs;    /* num routes calculated for COS */
         unsigned long   total_route_rejs;     /* num failed route calculations */
         unsigned long   total_tree_cache_hits;
                                               /* total num of tree cache hits */
         unsigned long   total_tree_cache_misses;
                                               /* total num of tree cache      */
                                               /* misses                       */
         unsigned char   reserva[20];
                                               /* reserved                     */
} QUERY_NN_TOPOLOGY_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode
AP_QUERY_NN_TOPOLOGY_STATS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

max_nodes
トポロジー・データベース内のノード・レコードの最大数（ゼロは、制限がないことを意味します）

cur_num_nodes
このノードのトポロジー・データベース内のノードの現在数。 この値が、許可されているノードの最大数を越える場合、アラートが発行されます。

node_in_tdus
このノードによって受信されるトポロジー・データベース更新 (TDU) の合計数

node_out_tdus
前回の初期化以降、すべての隣接ネットワーク・ノードに送信するため、このノードによって作成されたトポロジー・データベース更新 (TDU) の合計数

node_low_rsns
現行 RSN より小さい RSN で、このノードによって受信されたトポロジー・ノード更新の合計数。 偶数と奇数の両方の RSN がカウントされています。（これらの TDU は、エラーではありませんが、すべての隣接ネットワーク・ノードに対して TDU を同報通信したときに生じます。 このノードのトポロジー・データベースに対する更新は生じません。しかし、このノードは、低い RSN を送信した隣接ノードに対して、より高い RSN で TDU を送信します。）

node_equal_rsns
現行 RSN と等しい RSN で、このノードが受信したトポロジー・ノード更新の合計数。 偶数と奇数の両方の RSN がカウントされています。 （これらの TDU は、エラーではありませんが、すべての隣接ネットワーク・ノードに対して TDU を同報通信したときに生じます。 このノードのトポロジー・データベースに対する更新は生じません。）

node_good_high_rsns

現行 RSN より大きい RSN で、このノードによって受信されたトポロジー・ノード更新の合計数。ノードは、そのトポロジーを更新し、すべての隣接ネットワーク・ノードに対して TDU を同報通信します。 この更新の送信側に TDU を送信する必要はありません。その理由は、そのノードがすでにその更新をもっているからです。

node_bad_high_rsns

現行 RSN より大きい奇数 RSN で、このノードによって受信されたトポロジー・ノード更新の合計数。これらの更新は、APPN ネットワーク・ノードの 1 つによって検出されたトポロジー矛盾を表します。 ノードは、そのトポロジーを更新し、すべての隣接ネットワーク・ノードに対して TDU を同報通信します。

node_state_updates

APPN トポロジーおよび経路指定に影響するノード状態変更を内部的に検出した結果、作成されたトポロジー・ノード更新の合計数。 更新は、TDU によってすべての隣接ネットワーク・ノードに送信されます。

node_errors

このノードによって検出されたトポロジー・ノード更新の矛盾の合計数。これは、このノードがそのトポロジー・データベースの更新を試みて、データの矛盾を検出したときに生じます。 このノードは、現行 RSN の値を次に続く奇数に増分して TDU を作成し、すべての隣接ネットワーク・ノードにそれを同報通信します。

node_timer_updates

タイマーによる更新期限が到来しているこのノードの資源に対して作成されたトポロジー・ノード更新の合計数。更新は、TDU によってすべての隣接ネットワーク・ノードに送信されます。これらの更新によって、他のネットワーク・ノードがそれらのトポロジー・データベースからこのノードの資源を絶対に削除しなくなります。

node_purges

このノードのトポロジー・データベースから除去されたトポロジー・ノード・レコードの合計数。 これは、ノード・レコードが指定された時間内に更新されなかった場合に生じます。 所有ノードは、ネットワーク・トポロジー内に保持しておきたい資源の更新を同報通信する責任があります。

tg_low_rsns

現行 RSN より小さい RSN で、このノードによって受信されたトポロジー TG 更新の合計数。 偶数と奇数の両方の RSN がカウントされています。（これらの TDU は、エラーではありませんが、すべての隣接ネットワーク・ノードに対して TDU を同報通信したときに生じます。 このノードのトポロジー・データベースに対する更新は生じません。しかし、このノードは、低い RSN を送信した隣接ノードに対して、より高い RSN で TDU を送信します。）

tg_equal_rsns

現行 RSN と等しい RSN で、このノードによって受信されたトポロジー TG

更新の合計数。 偶数と奇数の両方の RSN がカウントされています。 （これらの TDU は、エラーではありませんが、すべての隣接ネットワーク・ノードに対して TDU を同報通信したときに生じます。 このノードのトポロジー・データベースに対する更新は生じません。）

tg_good_high_rsns
現行 RSN より大きい RSN で、このノードによって受信されたトポロジー TG 更新の合計数。ノードは、そのトポロジーを更新し、すべての隣接ネットワーク・ノードに対して TDU を同報通信します。

tg_bad_high_rsns
現行 RSN より大きい奇数 RSN で、このノードによって受信されたトポロジー TG 更新の合計数。これらの更新は、APPN ネットワーク・ノードの 1 つによって検出されたトポロジー矛盾を表します。 ノードは、そのトポロジーを更新し、すべての隣接ネットワーク・ノードに対して TDU を同報通信します。

tg_state_updates
APPN トポロジーおよび経路指定に影響するノード状態変更を内部的に検出した結果、作成されたトポロジー TG 更新の合計数。 更新は、TDU によってすべての隣接ネットワーク・ノードに送信されます。

tg_errors
このノードによって検出されたトポロジー TG 更新の矛盾の合計数。 これは、このノードがそのトポロジー・データベースの更新を試みて、データの矛盾を検出したときに生じます。 このノードは、現行 RSN の値を次に続く奇数に増分して TDU を作成し、すべての隣接ネットワーク・ノードにそれを同報通信します。

tg_timer_updates
タイマーによる更新期限が到来しているこのノードの資源に対して作成されたトポロジー TG 更新の合計数。更新は、TDU によってすべての隣接ネットワーク・ノードに送信されます。これらの更新によって、他のネットワーク・ノードがそれらのトポロジー・データベースからこのノードの資源を絶対に削除しなくなります。

tg_purges
このノードのトポロジー・データベースから除去されたトポロジー TG レコードの合計数。 これは、ノード・レコードが指定された時間内に更新されなかった場合に生じます。 所有ノードは、ネットワーク・トポロジー内に保持しておきたい資源の更新を同報通信する責任があります。

total_route_calcs
すべてのサービス・クラスに対して、前回以降、計算された経路の数

total_route_rejs
前回の初期化以降、すべてのサービス・クラスに対して計算できなかった経路指定要求の数

total_tree_cache_hits
キャッシュ経路ツリーによって条件を満足させた経路計算の数。 この数は、

計算された経路の合計より大きくなることがある点に注意してください。その理由は、各経路が複数のツリーの検査を必要とすることがあるからです。

total_tree_cache_misses
キャッシュ経路ツリーによって条件を満足させなかったので、新規の経路ツリーを作成しなければならなかった経路計算の数

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_NN_TOPOLOGY_TG

各ネットワーク・ノードは、ネットワーク・トポロジー・データベースを保守します。このデータベースは、ネットワーク内のネットワーク・ノード、VRN、およびネットワーク・ノード対ネットワーク・ノード TG についての情報を保持しています。 QUERY_NN_TOPOLOGY_TG は、このデータベース内の TG 項目についての情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のノードについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**owner**、**owner_type**、**dest**、**dest_type**、**tg_num**、および **frsn** フィールドを設定しなければなりません。それ以外の場合（**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合）、これらのフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、**owner**、**owner_type**、**dest**、**dest_type**、**tg_num**、および **frsn** 順になっています。**owner** 名および **dest** 名は、まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です）。**owner_type** および **dest_type** は、AP_NETWORK_NODE、 AP_VRN の配列に従います。 **tg_num** および **frsn** は、数値順に配列されます。

AP_LIST_INCLUSIVE が選択されている場合、戻されるリストは、その名前の最初の有効なレコードから開始します。

AP_LIST_FROM_NEXT を選択すると、リストは、指定された名前に続く名前をもつ最初の有効なレコードから開始します。

frsn フィールド（フロー縮小の順序番号）が非ゼロ値に設定されている場合、この値よりも高い FRSN をもつデータベース項目だけが戻されます。 これを利用して、まずノードの現行 FRSN を入手することによって、整合性のあるトポロジー・データベースをいくつかの かたまり に分けて戻すことができます。 以下のように作業します。

1. QUERY_NODE を発行します。これがノードの現行 FRSN を戻します。
2. すべてのデータベース項目を かたまり に分けて入手するため、必要なだけの QUERY_NN_TOPOLOGY_TG（FRSN をゼロに設定して）を発行します。
3. 再度、QUERY_NODE を発行し、ステップ 1 で戻された FRSN と新規の FRSN とを比較します。
4. 2 つの FRSN が異なっていれば、データベースは変更されています。そのため、ステップ 1 で指定した FRSN よりも 1 つ大きく設定した FRSN で QUERY_NN_TOPOLOGY_TG を発行します。

VCB 構造

```
typedef struct query_nn_topology_tg
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
```

```
        unsigned char   format;             /* format                          */
        unsigned short  primary_rc;         /* primary return code             */
        unsigned long   secondary_rc;       /* secondary return code           */
        unsigned char   *buf_ptr;           /* pointer to buffer               */
        unsigned long   buf_size;           /* buffer size                     */
        unsigned long   total_buf_size;     /* total buffer size required      */
        unsigned short  num_entries;        /* number of entries               */
        unsigned short  total_num_entries;  /* total number of entries         */
        unsigned char   list_options;       /* listing options                 */
        unsigned char   reserv3;            /* reserved                        */
        unsigned char   owner[17];          /* node that owns the TG           */
        unsigned char   owner_type;         /* type of node that owns the TG   */
        unsigned char   dest[17];           /* TG destination node             */
        unsigned char   dest_type;          /* TG destination node type        */
        unsigned char   tg_num;             /* TG number                       */
        unsigned char   reserv1;            /* reserved                        */
        unsigned long   frsn;               /* flow reduction sequence num     */
} QUERY_NN_TOPOLOGY_TG;

typedef struct topology_tg_summary
{
        unsigned short  overlay_size;       /* size of this entry              */
        unsigned char   owner[17];          /* node that owns the TG           */
        unsigned char   owner_type;         /* type of node that owns the TG   */
        unsigned char   dest[17];           /* TG destination node             */
        unsigned char   dest_type;          /* TG destination node type        */
        unsigned char   tg_num;             /* TG number                       */
        unsigned char   reserv3[1];         /* reserved                        */
        unsigned long   frsn;               /* flow reduction sequence num     */
} TOPOLOGY_TG_SUMMARY;

typedef struct topology_tg_detail
{
        unsigned short  overlay_size;       /* size of this entry              */
        unsigned char   owner[17];          /* node that owns the TG           */
        unsigned char   owner_type;         /* type of node that owns the TG   */
        unsigned char   dest[17];           /* TG destination node             */
        unsigned char   dest_type;          /* TG destination node type        */
        unsigned char   tg_num;             /* TG number                       */
        unsigned char   reserv3[1];         /* reserved                        */
        unsigned long   frsn;               /* flow reduction sequence num     */
        unsigned short  days_left;          /* days left until entry purged    */
        LINK_ADDRESS    dlc_data            /* DLC signalling data             */
        unsigned long   rsn;                /* resource sequence number        */
        unsigned char   status;             /* node status                     */
        TG_DEFINED_CHARS tg_chars;          /* TG characteristics              */
        unsigned char   reserva[20];        /* reserved                        */
} TOPOLOGY_TG_DETAIL;

typedef struct link_address
{
        unsigned short  length;             /* length                          */
        unsigned short  reserve1;           /* reserved                        */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                            /* address                         */
} LINK_ADDRESS;
```

注: **frsn** フィールドが非ゼロ値に設定されている場合、FRSN をもったノード項目だけが戻されます。**frsn** フィールドがゼロに設定されている場合は、すべてのノード項目が戻されます。

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode
AP_QUERY_NN_TOPOLOGY_TG

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr
リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **owner**、**owner_type**、**dest**、**dest_type**、**tg_num**、および **frsn**（下記のパラメーターを参照してください）の組合せは、戻される実情報の開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

owner TG の起点ノードの名前。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右

側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

owner_type
TG を所有するノードのタイプ。以下の値のいずれか 1 つになります。

AP_NETWORK_NODE
AP_VRN

owner_type が不明の場合、AP_LEARN_NODE を指定しなければなりません。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dest TG に対する完全修飾の宛先ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dest_type
この TG に対する宛先ノードのタイプ。以下の値のいずれか 1 つになります。

AP_NETWORK_NODE
AP_VRN

dest_type が不明の場合、AP_LEARN_NODE を指定しなければなりません。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

tg_num
TG と関連した数。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

frsn フロー縮小順序番号。 これが非ゼロの場合、この値より大きいか等しい FRSN をもったノードだけが戻されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

topology_tg_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

topology_tg_summary.owner
TG の起点ノードの名前。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

topology_tg_summary.owner_type
TG を所有するノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

topology_tg_summary.dest
TG に対する完全修飾の宛先ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

topology_tg_summary.dest_type
この TG に対する宛先ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

topology_tg_summary.tg_num
TG と関連した数。

topology_tg_summary.frsn
フロー縮小順序番号。 ローカル・ノードで最後にこの資源を更新した時刻を示します。

topology_tg_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

topology_tg_detail.owner

TG の起点ノードの名前。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

topology_tg_detail.owner_type

TG を所有するノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

topology_tg_detail.dest

TG に対する完全修飾の宛先ノード名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

topology_tg_detail.dest_type

この TG に対する宛先ノードのタイプ。以下の値のいずれか 1 つに設定されます。

AP_NETWORK_NODE
AP_VRN

topology_tg_detail.tg_num

TG と関連した数。

topology_tg_detail.frsn

フロー縮小順序番号。 ローカル・ノードで最後にこの資源を更新した時刻を示します。

topology_node_detail.days_left

トポロジー・データベースからこのノード項目を削除するまでの日数。

topology_tg_detail.dlc_data.length

VRN への接続の DLC アドレスの長さ (**dest_type** が AP_VRN でない場合、ゼロに設定します)。 .

topology_tg_detail.dlc_data.address

VRN への接続の DLC アドレス。 **dest_type** が AP_VRN でない場合、ゼロに設定されます。 .

topology_tg_detail.rsn

資源順序番号。 この資源を所有しているネットワーク・ノードによって割り当てられます。

topology_tg_detail.status

TG の状況を指定します。これは、以下の値の 1 つ以上を一緒に論理和 (OR) 演算したものです。

AP_TG_OPERATIVE
AP_TG_QUIESCING

AP_TG_CP_CP_SESSIONS
AP_TG_HPR
AP_TG_RTP
AP_TG_NONE

topology_tg_detail.tg_chars
TG 特性

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_TG

AP_INVALID_ORIGIN_NODE
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_NODE

QUERY_NODE は、ノード固有の情報と統計を戻します。 QUERY_NODE は、実行中に動的に判別された情報を戻すことに加えて、ノードの初期化中に設定されたパラメーターも戻します。

VCB 構造

```
typedef struct query_node
{
        unsigned short  opcode;               /* verb operation code          */
        unsigned char   reserv2;              /* reserved                     */
        unsigned char   format;               /* format                       */
        unsigned short  primary_rc;           /* primary return code          */
        unsigned long   secondary_rc;         /* secondary return code        */
        CP_CREATE_PARMS cp_create_parms;      /* create parameters            */
        unsigned long   up_time;              /* time since node started      */
        unsigned long   mem_size;             /* size of memory available     */
        unsigned long   mem_used;             /* size of memory used          */
        unsigned long   mem_warning_threshold;
                                              /* memory constrained           */
                                              /* threshold                    */
        unsigned long   mem_critical_threshold;
                                              /* memory critical threshold    */
        unsigned char   nn_functions_supported;
                                              /* NN functions supported       */
        unsigned char   functions_supported;
                                              /* functions supported          */
        unsigned char   en_functions_supported;
                                              /* EN functions supported       */
        unsigned char   nn_status;            /* node status.  One or more of */
        unsigned long   nn_frsn;              /* NN flow reduction            */
                                              /* sequence number              */
        unsigned long   nn_rsn;               /* Resource sequence number     */
        unsigned short  def_ls_good_xids;     /* Good XIDs for defined        */
                                              /* link stations                */
        unsigned short  def_ls_bad_xids;      /* Bad XIDs for defined         */
                                              /* link stations                */
        unsigned short  dyn_ls_good_xids;     /* Good XIDs for dynamic        */
                                              /* link stations                */
        unsigned short  dyn_ls_bad_xids;      /* Bad XIDs for dynamic         */
                                              /* link stations                */
        unsigned char   dlur_release_level;   /* Current DLUR release level   */
        unsigned char   reserva[19];
                                              /* reserved                     */
} QUERY_NODE;
typedef struct cp_create_parms
{
        unsigned short  crt_parms_len;        /* length of CP_CREATE_PARMS    */
        unsigned char   description[RD_LEN];
                                              /* resource description         */
        unsigned char   node_type;            /* node type                    */
        unsigned char   fqcp_name[17];
                                              /* fully qualified CP name      */
        unsigned char   cp_alias[8];
```

```
                                        /* CP alias                     */
        unsigned char   mode_to_cos_map_supp;
                                        /* mode to COS mapping support  */
        unsigned char   mds_supported;  /* MDS and MS capabilities      */
        unsigned char   node_id[4];
                                        /* node ID                      */
        unsigned short  max_locates;    /* max locates node can process */
        unsigned short  dir_cache_size; /* directory cache size         */
                                        /* (reserved) if not NN)        */
        unsigned short  max_dir_entries;/* max directory entries        */
        unsigned short  locate_timeout; /* locate timeout in seconds    */
        unsigned char   reg_with_nn;    /* register resources with NNS  */
        unsigned char   reg_with_cds;   /* resource registration with   */
                                        /* CDS                          */
        unsigned short  mds_send_alert_q_size;
                                        /* size of MDS send alert queue */
        unsigned short  cos_cache_size; /* number of COS definitions    */
        unsigned short  tree_cache_size;/* Topology Database routing    */
                                        /* tree cache size              */
        unsigned short  tree_cache_use_limit;
                                        /* num times tree can be used   */
        unsigned short  max_tdm_nodes;  /* max num nodes that can be    */
                                        /* stored in Topology Database  */
        unsigned short  max_tdm_tgs;    /* max num TGs that can be      */
                                        /* stored in Topology Database  */
        unsigned long   max_isr_sessions;/* max ISR sessions            */
        unsigned long   isr_sessions_upper_threshold;
                                        /* upper threshold for ISR sess */
        unsigned long   isr_sessions_lower_threshold;
                                        /* lower threshold for ISR sess */
        unsigned short  isr_max_ru_size;/* max RU size for ISR          */
        unsigned short  isr_rcv_pac_window;/* ISR rcv pacing window size*/
        unsigned char   store_endpt_rscvs;/* endpoint RSCV storage      */
        unsigned char   store_isr_rscvs;/* ISR RSCV storage             */
        unsigned char   store_dlur_rscvs;/* DLUR RSCV storage           */
        unsigned char   dlur_support;   /* is DLUR supported?           */
        unsigned char   pu_conc_support;/* is PU conc supported?        */
        unsigned char   nn_rar;         /* Route additional resistance  */
        unsigned char   hpr_support;    /* level of HPR support         */
        unsigned char   mobile;         /* HPR path-switch controller?  */
        unsigned char   discovery_support;/* Discovery function utilized*/
        unsigned char   discovery_group_name[8];
                                        /* Group name for Discovery     */
        unsigned char   implicit_lu_0_to_3;/* Implicit LU 0 to 3 support*/
        unsigned char   default_preference;/* Default routing preference*/
        unsigned char   anynet_supported;/* Support for non-native      */
                                        /* (AnyNet) routing             */
        unsigned char   reserv2[4];
                                        /* reserved                     */
        unsigned char   node_spec_data_len;/* length of node specific data */
        unsigned char   ptf[64];
                                        /* program temporary fix array  */
} CP_CREATE_PARMS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_NODE

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

cp_create_parms.crt_parms_len

create パラメーター構造の長さ。

cp_create_parms.description

資源の記述。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

cp_create_parms.node_type

常に次のとおりになります。

AP_END_NODE

cp_create_parms.fqcp_name

ノードの 17 バイトの完全修飾された制御点名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定します。)

cp_create_parms.cp_alias

ローカルで使用される制御点の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

cp_create_parms.mode_to_cos_map_supp

COS マッピングに対するモードがノードによってサポートされているかどうか（AP_YES または AP_NO）を指定します。これが AP_YES に設定されている場合、DEFINE_MODE verb で指定されている COS は、SNA 定義の COS であるかまたは、DEFINE_COS verb を発行することによって定義されたもののいずれかでなければなりません。

cp_create_parms.mds_supported

管理サービスが複数ドメイン・サポート、および管理サービス機能をサポートするかどうか（AP_YES または AP_NO）を指定します。

cp_create_parms.node_id
XID 交換で使用されるノード識別子。これは 4 バイトの 16 進数ストリングです。

cp_create_parms.max_locates
ノードが処理できる locate の最大数

cp_create_parms.dir_cache_size
ネットワーク・ノードのみ。ディレクトリー・キャッシュのサイズ。

cp_create_parms.max_dir_entries
ディレクトリー項目の最大数。このフィールドがゼロに設定されている場合、無制限となります。

cp_create_parms.locate_timeout
ネットワーク検索がタイムアウトするまでの時間を秒数で指定します。ゼロ値は、検索がタイムアウトをもたないことを示します。

cp_create_parms.reg_with_nn
資源をネットワーク・ノード・サーバーで登録するかどうか（AP_YES または AP_NO）を指定します。このフィールドが AP_YES に設定されている場合、エンド・ノードのネットワーク・ノード・サーバーは、指示された locate だけを、エンド・ノードに転送します。このフィールドが AP_YES に設定されていない場合、ネットワーク・ノード・サーバーは、エンド・ノードにすべての同報通信検索を転送します。登録の失敗は、ノードの初期化の正常終了に影響を及ぼしません。詳細については、484ページの『REGISTRATION_FAILURE』を参照してください。

cp_create_parms.reg_with_cds
ネットワーク・ノード・サーバーによる、エンド・ノード資源の中央ディレクトリーサーバーでの登録を許可するかどうか（AP_YES または AP_NO）を指定します(**reg_with_nn** が AP_NO に設定されている場合、このフィールドは無視されます。）ネットワーク・ノードは、任意選択で中央ディレクトリー・サーバーでローカルまたはドメイン資源を登録できるかどうか（AP_YES または AP_NO）を指定します。登録の失敗は、ノードの初期化の正常終了に影響を及ぼしません。

cp_create_parms.mds_send_alert_q_size
MDS 送信アラート待ち行列のサイズ。この制限に達すると、MDS コンポーネントは、待ち行列上の最も古い項目を削除します。

cp_create_parms.cos_cache_size
COS データベースの重みキャッシュのサイズ

cp_create_parms.tree_cache_size
トポロジー・データベースの経路ツリー・キャッシュのサイズ

cp_create_parms.tree_cache_use_limit
キャッシュ・ツリーの使用の最大数。この数を一度超過すると、ツリーは廃棄され、再計算されます。これによって、ノードは、等しい重みの経路間でセッションの平衡を保つことができます。この値が低いと、活動化の待ち時間は増加しますが、よりよい負荷平衡が得られます。

cp_create_parms.max_tdm_nodes
トポロジー・データベース内に格納することができるノードの最大数（ゼロは、制限がないことを意味します）。

cp_create_parms.max_tdm_tgs
トポロジー・データベース内に格納することができる TG の最大数（ゼロは、制限がないことを意味します）。

cp_create_parms.max_isr_sessions
ノードが同時に参加することができる ISR セッションの最大数

cp_create_parms.isr_sessions_upper_threshold
cp_create_parms.isr_sessions_lower_threshold を参照してください。

cp_create_parms.isr_sessions_lower_threshold
限界値の上限と下限がノードの混雑状況を制御します。ISR セッションの数が限界値の上限を越えると、ノードの状況は非混雑から混雑に変わります。ISR セッションの数が、限界値の下限を下回ると、ノードの状況は非混雑に戻ります。

cp_create_parms.isr_max_ru_size
中間セッションに対してサポートされている最大 RU サイズ

cp_create_parms.isr_rcv_pac_window
中間セッションに対する推奨受信ペーシング・ウィンドウ・サイズ。この値は、隣接ノードが適応ペーシングをサポートしない場合に、中間セッションの 2 次ホップにおいてのみ使用されます。

cp_create_parms.store_endpt_rscvs
診断目的のために RSCV を格納するかどうか（AP_YES または AP_NO）を指定します。

cp_create_parms.store_isr_rscvs
診断目的のために RSCV を格納するかどうか（AP_YES または AP_NO）を指定します。

cp_create_parms.store_dlur_rscvs
ノードが診断目的のために RSCV を格納するかどうか（AP_YES または AP_NO）を指定します。このフィールドが AP_YES に設定されている場合には、RSCV は QUERY_DLUR_LU verb で戻されます。

cp_create_parms.dlur_support
DLUR をサポートするかどうか（AP_YES または AP_NO）を指定します。

cp_create_parms.pu_conc_support
PU 集信をサポートするかどうか（常に AP_NO）を指定します。

cp_create_parms.nn_rar
ネットワーク・ノード経路の追加抵抗

cp_create_parms.hpr_support
ノードによって提供される HPR に対するサポートのレベル（AP_NONE、 AP_BASE、または AP_RTP）を指定します。

cp_create_parms.mobile
ノードが HPR パス・スイッチ制御装置であるかどうか（AP_YES または AP_NO）を指定します。**cp_create_parms.hpr_support** フィールドが AP_RTP に設定されていない場合、このフィールドは予約されます。

cp_create_parms.discovery_support
このノードによって、ディスカバリー機能が使用されるかどうかを指定します。

AP_DISCOVERY_CLIENT
クライアント・ディスカバリー機能が、このノードによって使用されます。

AP_DISCOVERY_SERVER
サーバー・ディスカバリー機能が、このノードによって使用されます。

cp_create_parms.discovery_group_name
ノードが用いるディスカバリー機能で使用されるグループ名を指定します。このフィールドがすべてゼロに設定されている場合、省略時グループ名が使用されます。

cp_create_parms.implicit_lu_0_to_3
ノードが ACTLU のタイプ 0 から 3 の LU 暗黙定義をサポートするかどうか（AP_YES または AP_NO）を指定します。

cp_create_parms.default_preference
このノードからセッションを開始するときに推奨する経路指定方法を指定します。

注: これは、DEFINE_PARTNER_LU verb を使用して、LU ごとに上書き変更することができます。このフィールドは、以下の値を取ることができます。

AP_NATIVE
ネイティブ (APPN) の経路指定プロトコルだけを使用します。

AP_NONNATIVE
非ネイティブ (AnyNet) の経路指定プロトコルだけを使用します。

AP_NATIVE_THEN_NONNATIVE
ネイティブ (APPN) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、非ネイティブ (AnyNet) プロトコルを使用して、セッション活動化を再試行してください。

AP_NONNATIVE_THEN_NATIVE
非ネイティブ (AnyNet) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、ネイティブ (APPN) プロトコルを使用して、セッション活動化を再試行してください。

注: 終わりの 3 つの値に意味があるのは、AnyNet DLC がノード・オペレーター機能にとって使用可能で、かつ AnyNet リンク・ステーションが定義されている場合だけです。

cp_create_parms.anynet_supported

AnyNet (TCP/IP) 経路指定に対するサポートを指定します。

cp_create_parms.node_spec_data_len

このフィールドは常にゼロに設定しなくてはなりません。

cp_create_parms.ptf

将来のプログラム一時修正 (PTF) 操作の構成と制御のための配列です。

cp_create_parms.ptf[0]

REQDISCONT サポート。Communications Server は、セッション・トラフィックがもう必要としなくなった限定資源ホスト・リンクを非活動化するため、通常、REQDISCONT を使用します。このバイトは、Communications Server による REQDISCONT の使用を抑制するため、または Communications Server によって送信された REQDISCONT 要求で使用された設定値を修正するために使用することができます。

AP_SUPPRESS_REQDISCONT

このビットが設定されると、Communications Server は REQDISCONT を使用しません（このバイト内の他のすべてのビットが無視されます)。

AP_OVERRIDE_REQDISCONT

このビットが設定されると、Communications Server は、以下の 2 つのビットにもとづき、REQDISCONT 上の通常の設定値を上書き変更します。

AP_REQDISCONT_TYPE

このビットが設定されると、Communications Server は、REQDISCONT に 即時に のタイプを指定します。それ以外の場合、Communications Server は、通常 のタイプを指定します
（AP_OVERRIDE_REQDISCONT が設定されないと、このビットは無視されます)。

AP_REQDISCONT_RECONTACT

このビットが設定されると、Communications Server は、REQDISCONT に 即時再交信 を指定します。それ以外の場合、Communications Server は、非即時再交信 を指定します（AP_OVERRIDE_REQDISCONT が設定されないと、このビットは無視されます)。

cp_create_parms.ptf[1]

ERP サポート。

Communications Server は、通常、ACTPU(ERP) を ERP として処理します（ACTPU(ERP) は、PU-SSCP セッションのリセットを要求します。しかし、ACTPU（コールド）とは違って、補助的な LU-SSCP と PLU-SLU のセッ

ションの暗黙の非活動化を要求しません)。SNA では、まるで ACTPU（コールド）であるかのように ACTPU (ERP) を正しく処理することができます。

AP_OVERRIDE_ERP
このビットが設定されると、Communications Server は、すべての ACTPU 要求を ACTPU（コールド）として処理します。

cp_create_parms.ptf[2]
BIS サポート。

Communications Server は、限定資源 LU 6.2 セッションを非活動化する前に、通常、BIS プロトコルを使用します。このバイトによって、BIS の使用を上書き変更することができます。

AP_SUPPRESS_BIS
このビットが設定されると、Communications Server は、BIS プロトコルを使用しません。限定資源 LU 6.2 セッションは、UNBIND（クリーンアップ）を使用して即時に非活動化されます。

up_time
ノードを開始（または再開始）した以降の時間（100 分の 1 秒単位で）

mem_size
記憶管理が基礎オペレーティング・システムから入手した使用可能記憶域のサイズ。

mem_used
現在、プロセスに割り振られている記憶域のバイト数

mem_warning_threshold
記憶管理が記憶域資源を制約するために使用する割当て限界値

mem_critical_threshold
記憶管理が記憶域資源を決定的に制約するために使用する割当て限界値

nn_functions_supported
予約済み

functions_supported
どの機能をサポートするかを指定します。以下の値のうちの 1 つまたは複数になります。

AP_NEGOTIABLE_LS
AP_SEGMENT_REASSEMBLY
AP_BIND_REASSEMBLY
AP_PARALLEL_TGS
AP_CALL_INAP_ADAPTIVE_PACING

en_functions_supported
サポートされるエンド・ノード機能を指定します。

AP_SEGMENT_GENERATION
ノードは、セグメント生成をサポートします。

AP_MODE_TO_COS_MAP
ノードは、COS 名マッピングに対するモード名をサポートします。

AP_LOCATE_CDINIT
ノードは、locate の生成、およびリモート LU の位置を調べるための定義域間開始 GDS 変数の生成をサポートします。

AP_REG_WITH_NN
ノードは、その LU を、隣接して働くネットワーク・ノードで登録します。

AP_REG_CHARS_WITH_NN
ノードは、send 登録特性をサポートします（send 登録名もサポートされているときにだけサポートすることができます）。

nn_status
予約済み

nn_frsn
予約済み

nn_rsn
予約済み

def_ls_good_xids
ノードが最後に開始して以降、すべての定義済みリンク・ステーション上で発生した正常な XID 交換の合計数

def_ls_bad_xids
ノードが最後に開始して以降、すべての定義済みリンク・ステーション上で発生した不成功の XID 交換の合計数

dyn_ls_good_xids
ノードが最後に開始して以降、すべての動的リンク・ステーション上で発生した正常な XID 交換の合計数

dyn_ls_bad_xids
ノードが最後に開始して以降、すべての動的リンク・ステーション上で発生した不成功の XID 交換の合計数

dlur_release_level
現行 DLUR のリリース・レベルを指定します。

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_PARTNER_LU

QUERY_PARTNER_LU は、ローカル LU によって使用されていたパートナー LU についての情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。特定のパートナー LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**plu_alias** フィールド（あるいは、**plu_alias** がすべてゼロに設定されている場合は、**fqplu_name**）を設定しなければなりません。 **list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合、これらのフィールドの両方ともが無視されます。 **lu_name** または **lu_alias** フィールドは、常に設定しておかなければなりません。 **lu_name** が非ゼロの場合、**lu_name** が **lu_alias** に優先して使用されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **fqplu_name** によって配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

plu_alias がすべてゼロに設定されている場合、**fqplu_name** 値が使用されます。それ以外の場合、**plu_alias** が必ず使用され、**fqplu_name** は無視されます。

戻されたパートナー LU のリストは、それらが現在アクティブ・セッションをもっているかどうかによって、フィルター処理することができます。 フィルター処理を希望する場合は、**active_sessions** フィールドを AP_YES に設定しなくてはなりません（それ以外の場合このフィールドは AP_NO に設定しなくてはなりません）。

この verb は、少なくとも 1 つのセッションがパートナー LU との間に確立された際に判別された情報を戻します。

QUERY_PARTNER_LU_DEFINITION verb は、定義情報だけを戻します。

VCB 構造

```
typedef struct query_partner_lu
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   *buf_ptr;           /* pointer to buffer            */
        unsigned long   buf_size;           /* buffer size                  */
        unsigned long   total_buf_size;     /* total buffer size required   */
        unsigned short  num_entries;        /* number of entries            */
        unsigned short  total_num_entries;  /* total number of entries      */
        unsigned char   list_options;       /* listing options              */
        unsigned char   reserv3;            /* reserved                     */
        unsigned char   lu_name[8];         /* LU name                      */
        unsigned char   lu_alias[8];        /* LU alias                     */
```

```
        unsigned char   plu_alias[8];        /* partner LU alias              */
        unsigned char   fqplu_name[17];      /* fully qualified partner       */
                                             /* LU name                       */
        unsigned char   active_sessions;     /* active sessions only filter   */
} QUERY_PARTNER_LU;

typedef struct plu_summary
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   plu_alias[8];        /* partner LU alias              */
        unsigned char   fqplu_name[17];      /* fully qualified partner       */
                                             /* LU name                       */
        unsigned char   reserv1;             /* reserved                      */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned short  act_sess_count;      /* curr active sessions count    */
        unsigned char   partner_cp_name[17]; /* partner LU CP name            */
        unsigned char   partner_lu_located;  /* CP name resolved?             */
} PLU_SUMMARY;

typedef struct plu_detail
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   plu_alias[8];        /* partner LU alias              */
        unsigned char   fqplu_name[17];      /* fully qualified partner       */
                                             /* LU name                       */
        unsigned char   reserv1;             /* reserved                      */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned short  act_sess_count;      /* curr active sessions count    */
        unsigned char   partner_cp_name[17]; /* partner LU CP name            */
        unsigned char   partner_lu_located;  /* CP name resolved?             */
        unsigned char   plu_un_name[8];      /* partner LU uninterpreted name */
        unsigned char   parallel_sess_supp;  /* parallel sessions supported?  */
        unsigned char   conv_security;       /* conversation security         */
        unsigned short  max_mc_ll_send_size; /* max send LL size for mapped   */
                                             /* conversations                 */
        unsigned char   implicit;            /* implicit or explicit entry    */
        unsigned char   security_details;    /* conversation security detail  */
        unsigned char   duplex_support;      /* full-duplex support           */
        unsigned char   preference;          /* routing preference            */
        unsigned char   reserva[16];         /* reserved                      */
} PLU_DETAIL;
```

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_PARTNER_LU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

lu_name（または、**lu_name** がすべてゼロに設定されている場合は、**lu_alias**）と **plu_alias**（または、**plu_alias** がすべてゼロに設定されている場合は、**fqplu_name**）の組合せを指定（下記のパラメーターを参照してください）すると、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST
plu_alias フィールドおよび **fqplu_name** フィールドは無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

lu_name
LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。このフィールドをすべてゼロに設定すると、**lu_alias** フィールドが索引の判別に使用されます。

lu_alias
ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** と **lu_alias** の両方がすべてゼロに設定されている場合には、制御点と関連した LU（省略時 LU）が使用されます。

plu_alias
パートナー LU 別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが索引値として使用されます。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。 この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

active_sessions

活動状態のセッションでのフィルター。 現在活動状態のセッションをもっているかどうかに従って、戻されるパートナー LU をフィルター処理すべきかどうか（AP_YES または AP_NO）を指定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries

実際に戻される項目の数

total_num_entries

戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

plu_summary.overlay_size

この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

plu_summary.plu_alias

パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

plu_summary.fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。この名前は、17 バイト長で、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋め込まれています。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

plu_summary.description

資源の記述（DEFINE_PARTNER_LU で指定されたものです）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

plu_summary.act_sess_count
ローカル LU と パートナー LU 間の活動状態のセッションの合計数。**active_sessions** フィルターが AP_YES に設定されている場合には、このフィールドは常にゼロより大きくなります。

plu_summary.partner_cp_name
パートナー LU の制御点に対する 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

plu_summary.partner_lu_located
パートナー LU に対する制御点名が解決されたかどうか (AP_YES または AP_NO) を指定します。

plu_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目 (ある場合) へのオフセット。

plu_detail.plu_alias
パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

plu_detail.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

plu_detail.description
資源の記述 (DEFINE_PARTNER_LU で指定されたものです)。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

plu_detail.act_sess_count
ローカル LU と パートナー LU 間の活動状態のセッションの合計数。**active_sessions** フィルターが AP_YES に設定されている場合には、このフィールドは常にゼロより大きくなります。

plu_detail.partner_cp_name
パートナー LU の制御点に対する 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

plu_detail.partner_lu_located
パートナー LU に対する制御点名が解決されたかどうか (AP_YES または AP_NO) を指定します。

plu_detail.plu_un_name
パートナー LU の非解釈名。これは、8 バイトのタイプ A の EBCDIC 文字ストリングです。

plu_detail.parallel_sess_supp

並列セッションがサポートされるかどうか（AP_YES または AP_NO）を指定します。

plu_detail.conv_security

このパートナー LU に会話機密保護の情報を送信できるようにするかどうか（ AP_YES または AP_NO）を指定します。これが AP_NO に設定されている場合には、トランザクション・プログラムによって提供されたどのような安全保護情報も、パートナー LU に送信されません。

plu_detail.max_mc_ll_send_size

パートナー LU に送信することができる論理長 (LL) レコードの最大サイズ。これより大きいデータ・レコードは、パートナー LU に送信される前にいくつかの LL レコードに分割されます。**max_mc_ll_send_size** が取ることができる最大値は、32 767 です。

plu_detail.implict

結果としての項目が暗黙定義 (AP_YES) によるものか、または明示定義 (AP_NO) によるものかを指定します。

plu_detail.security_details

BIND で折衝されたとおりに会話機密保護サポートを戻します。以下の値のうちの 1 つまたは複数になります。

AP_CONVERSATION_LEVEL_SECURITY

会話機密保護の情報は、会話を割り振るパートナー LU への要求、またはそのパートナー LU からの要求で受け入れられます。 会話機密保護のサポートの特定のタイプが以下の値で記述されます。

AP_ALREADY_VERIFIED

ローカル LU と パートナー LU の両方は、すでに検査済みの要求による会話の割振りを受け入れることに同意します。すでに検査済みの要求は、ユーザー ID だけをもっていればよく、パスワードをもつ必要はありません。

AP_PERSISTENT_VERIFICATION

ローカル LU とパートナー LU 間のセッションでは、持続検査がサポートされています。 このことは、会話に対する最初の要求（ユーザー ID をもち、一般的にはパスワードももつ）が一度検査済みとなると、それ以降の会話に対する要求では、ユーザー ID だけをもっていればよいということを意味します。

AP_PASSWORD_SUBSTITUTION

ローカル LU とパートナー LU は、パスワード置換の会話機密保護をサポートします。 会話を割り振る要求が発行されると、その要求は暗号化された形式のパスワードをもちます。 パスワード置換がサポートされていない場合、明確なテキスト（非暗号化）形式でパスワードをもちます。

注: セッションがパスワード置換をサポートしない場合には、AP_PGM_STRONG のセキュリティー・タイプをもった ALLOCATE や SEND_CONVERSATION は失敗します。

plu_detail.duplex_support
BIND で折衝されたとおりに会話二重サポートを戻します。これは、以下の値の 1 つになります。

AP_HALF_DUPLEX
半二重会話だけがサポートされます。

AP_FULL_DUPLEX
半二重会話と同様に全二重会話がサポートされます。

AP_UNKNOWN
会話二重サポートは不明です。その理由は、パートナー LU に対する活動状態のセッションが存在しないからです。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PLU_NAME

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_PARTNER_LU_DEFINITION

QUERY_PARTNER_LU_DEFINITION は、あらかじめ DEFINE_PARTNER_LU verb で渡されていた情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のパートナー LU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**plu_alias** フィールド（または、**plu_alias** がすべてゼロに設定されている場合は、**fqplu_name**）を設定しなければなりません。**plu_alias** フィールドが非ゼロの場合、このフィールドは索引を判別するために使用され、**fqplu_name** は無視されます。 **plu_alias** フィールドがすべてゼロに設定されている場合、**fqplu_name** が索引を判別するために使用されます。 **list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合には、これら両方のフィールドは無視されます。 （この場合に戻されるリストは、AP_LIST_BY_ALIAS が **list_options** に設定されている場合は、**plu_alias** によって配列されます。それ以外の場合に戻されるリストは、**fqplu_name** によって配列されます。）リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、指定されたオプションに従って、**plu_alias** または **fqplu_name** のいずれかで配列されます。 名前の長さによって配列され、長さが同じ名前の場合は ASCII の辞書配列順序に従って配列されます (通常の MIB の配列順序と同一です) 。AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

この verb は、定義情報だけを戻すことに注意してください。 QUERY_PARTNER_LU verb は、少なくとも 1 つのセッションがパートナー LU との間に確立された際に判別した情報を戻します。

VCB 構造

```
typedef struct query_partner_lu_definition
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   *buf_ptr;            /* pointer to buffer            */
        unsigned long   buf_size;            /* buffer size                  */
        unsigned long   total_buf_size;      /* total buffer size required   */
        unsigned short  num_entries;         /* number of entries            */
        unsigned short  total_num_entries;   /* total number of entries      */
        unsigned char   list_options;        /* listing options              */
        unsigned char   reserv3;             /* reserved                     */
        unsigned char   plu_alias[8];        /* partner LU alias             */
        unsigned char   fqplu_name[17];      /* fully qualified partner      */
                                             /* LU name                      */
} QUERY_PARTNER_LU_DEFINITION;

typedef struct partner_lu_def_summary
{
        unsigned short  overlay_size;        /* size of this entry           */
```

```
        unsigned char   plu_alias[8];        /* partner LU alias             */
        unsigned char   fqplu_name[17];      /* fully qualified partner      */

/* LU name                      */
        unsigned char   description[RD_LEN]; /* resource description         */
} PARTNER_LU_DEF_SUMMARY;
typedef struct partner_lu_def_detail
{
        unsigned short  overlay_size;        /* size of this entry           */
        unsigned char   plu_alias[8];        /* partner LU alias             */
        unsigned char   fqplu_name[17];      /* fully qualified partner      */
                                             /* LU name                      */
        unsigned char   reserv1;             /* reserved                     */
        PLU_CHARS       plu_chars;           /* partner LU characteristics   */
} PARTNER_LU_DEF_DETAIL;
typedef struct plu_chars
{
        unsigned char   fqplu_name[17];      /* fully qualified partner      */
                                             /* LU name                      */
        unsigned char   plu_alias[8];        /* partner LU alias             */
        unsigned char   description[RD_LEN]; /* resource description         */
        unsigned char   plu_un_name[8];      /* partner LU uninterpreted name */
        unsigned char   preference;          /* routing preference           */
        unsigned short  max_mc_ll_send_size; /* max MC send LL size          */
        unsigned char   conv_security_ver;   /* already_verified accepted    */
        unsigned char   parallel_sess_supp;  /* parallel sessions supported? */
        unsigned char   reserv2[8];          /* reserved                     */
} PLU_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_PARTNER_LU_DEFINITION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **plu_alias**（または、**plu_alias** がすべてゼロに設定されている場合は、**fqplu_name**）（下記のパラメーターを参照してください）は、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

AP_LIST_BY_ALIAS
戻されるリストは、**plu_alias** によって配列されます。このオプションは、AP_FIRST_IN_LIST が指定されている場合にだけ有効です。AP_LIST_FROM_NEXT または AP_LIST_INCLUSIVE が指定されている場合、リスト配列は、**plu_alias** または **fqplu_name** のどちらが開始点として指定されているかによって異なります。

plu_alias
パートナー LU 別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

partner_lu_def_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

partner_lu_def_summary.plu_alias
パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

partner_lu_def_summary.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

partner_lu_def_summary.description
資源の記述（DEFINE_PARTNER_LU で指定されたものです)。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

partner_lu_def_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

partner_lu_def_detail.plu_alias
パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

partner_lu_def_detail.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

partner_lu_def_detail.plu_chars.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオ

ドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

partner_lu_def_detail.plu_chars.plu_alias
パートナー LU 別名。

partner_lu_def_detail.plu_chars.description
資源の記述 (DEFINE_PARTNER_LU で指定されたものです)。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

partner_lu_def_detail.plu_chars.plu_un_name
パートナー LU の非解釈名。これは、8 バイトのタイプ A の EBCDIC 文字ストリングです。

plu_chars.preference
このパートナー LU に対するセッション活動化のために優先される経路指定プロトコルのセット。 このフィールドは、以下の値を取ることができます。

AP_NATIVE
ネイティブ (APPN) の経路指定プロトコルだけを使用します。

AP_NONNATIVE
非ネイティブ (AnyNet) の経路指定プロトコルだけを使用します。

AP_NATIVE_THEN_NONNATIVE
ネイティブ (APPN) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、非ネイティブ (AnyNet) プロトコルを使用して、セッション活動化を再試行してください。

AP_NONNATIVE_THEN_NATIVE
非ネイティブ (AnyNet) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、ネイティブ (APPN) プロトコルを使用して、セッション活動化を再試行してください。

AP_USE_DEFAULT_PREFERENCE
ノードが開始したときに定義した省略時推奨値を使用します。

注: 非ネイティブ経路指定が意味をもつのは、ノード・オペレーター機能にとって AnyNet DLC が使用可能で、かつ定義済みの AnyNet リンク・ステーションがある場合だけです。

partner_lu_def_detail.plu_chars.max_mc_ll_send_size
パートナー LU に送信することができる論理長 (LL) レコードの最大サイズ。これより大きいデータ・レコードは、パートナー LU に送信される前にいくつかの LL レコードに分割されます。**max_mc_ll_send_size** が取ることができる最大値は、32 767 です。

partner_lu_def_detail.plu_chars.conv_security_ver
ローカル LU に代わって、パートナー LU が **user_ids** の妥当性検査をす

ることを許可されているかどうか、つまり、Attach 要求ですでに検査済みの標識を、パートナー LU が設定できるかどうかということを指定します。

AP_YES
AP_NO

partner_lu_def_detail.plu_chars.parallel_sess_supp
並列セッションがサポートされるかどうか（AP_YES または AP_NO）を指定します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PLU_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_PORT

QUERY_PORT は、ノードのポートについての情報のリストを戻します。 この情報は、『判別済みデータ』 (実行の間に動的に収集されたデータ)、および 『定義済みデータ』 (アプリケーションにより DEFINE_PORT で提供されたデータ) として構造化されます。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のポートについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**port_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **port_name** によって配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です)。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

戻されたポートのリストは、ポートが属する DLC の名前によってフィルター処理することができます。 この場合、**dlc_name** フィールドを設定しなければなりません（それ以外の場合、このフィールドは、すべてゼロに設定しなければなりません)。

VCB 構造

```
typedef struct query_port
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   *buf_ptr;            /* pointer to buffer             */
        unsigned long   buf_size;            /* buffer size                   */
        unsigned long   total_buf_size;      /* total buffer size required    */
        unsigned short  num_entries;         /* number of entries             */
        unsigned short  total_num_entries;   /* total number of entries       */
        unsigned char   list_options;        /* listing options               */
        unsigned char   reserv3;             /* reserved                      */
        unsigned char   port_name[8];        /* port name                     */
        unsigned char   dlc_name[8];         /* DLC name filter               */
} QUERY_PORT;
typedef struct port_summary
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   port_name[8];        /* port name                     */
        unsigned char   description[RD_LEN]; /* resource description          */
```

```
        unsigned char   port_state;          /* port state                    */
        unsigned char   reserv1[1];          /* reserved                      */
        unsigned char   dlc_name[8];         /* name of DLC                   */
} PORT_SUMMARY;
typedef struct port_detail
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   port_name[8];        /* port name                     */
        unsigned char   reserv1[2];          /* reserved                      */
        PORT_DET_DATA   det_data;            /* determined data               */
        PORT_DEF_DATA   def_data;            /* defined data                  */
} PORT_DETAIL;
typedef struct port_det_data
{
        unsigned char   port_state;          /* port state                    */
        unsigned char   dlc_type;            /* DLC type                      */
        unsigned char   port_sim_rim;        /* port initialization options   */
        unsigned char   reserv1;             /* reserved                      */
        unsigned short  def_ls_good_xids;    /* number of successful XIDs     */
        unsigned short  def_ls_bad_xids;     /* number of unsuccessful XIDs   */
        unsigned short  dyn_ls_good_xids;    /* successful XIDs on dynamic    */
                                             /* LS count                      */
        unsigned short  dyn_ls_bad_xids;     /* failed XIDs on dynamic        */
                                             /* LS count                      */
        unsigned char   reserva[20];         /* reserved                      */
} PORT_DET_DATA;
typedef struct port_def_data
{
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   dlc_name[8];         /* DLC name associated with port */
        unsigned char   port_type;           /* port type                     */
        unsigned char   reserv3[7];          /* unsigned char                 */
        unsigned long   port_number;         /* port number                   */
        unsigned short  max_rcv_btu_size;    /* max receive BTU size          */
        unsigned short  tot_link_act_lim;    /* total link activation limit   */
        unsigned short  inb_link_act_lim;    /* inbound link activation limit */
        unsigned short  out_link_act_lim;    /* outbound link activation limit*/
        unsigned char   ls_role;             /* initial link station role     */
        unsigned char   reserv1[15];         /* reserved                      */
        unsigned char   implicit_dspu_template[8];
                                             /* implicit DSPU template        */
        unsigned char   reserv2[3];          /* reserved                      */
        unsigned char   implicit_dspu_services;
                                             /* implicit links support DSPUs  */
        unsigned short  implicit_deact_timer;
                                             /* Implicit link HPR link        */
                                             /* deactivation timer            */
        unsigned short  act_xid_exchange_limit;
                                             /* activation XID exchange limit */
        unsigned short  nonact_xid_exchange_limit;
                                             /* non-act. XID exchange limit   */
        unsigned char   ls_xmit_rcv_cap;     /* LS transmit-rcv capability    */
        unsigned char   max_ifrm_rcvd;       /* max number of I-frames that   */
                                             /* can be received               */
        unsigned short  target_pacing_count; /* target pacing count           */
```

```
        unsigned short  max_send_btu_size;   /* max send BTU size             */
        LINK_ADDRESS    dlc_data;            /* DLC data                      */
        LINK_ADDRESS    hpr_dlc_data;        /* HPR DLC data                  */
        unsigned char   implicit_cp_cp_sess_support;
                                             /* Implicit links allow CP-CP    */
                                             /* sessions                      */
        unsigned char   implicit_limited_resource;
                                             /* Implicit links are            */
                                             /* limited resource              */
        unsigned char   implicit_hpr_support;
                                             /* Implicit links support HPR    */
        unsigned char   implicit_link_lvl_error;
                                             /* Implicit links support        */
                                             /* HPR link-level error recovery */
        unsigned char   retired1;            /* reserved                      */
        TG_DEFINED_CHARS default_tg_chars;   /* Default TG chars              */
        unsigned char   discovery_supported; /* Discovery function supported? */
        unsigned short  port_spec_data_len;  /* length of port spec data      */
        unsigned short  link_spec_data_len;  /* length of link spec data      */
} PORT_DEF_DATA;

typedef struct link_address
{
        unsigned short  length;              /* length                        */
        unsigned short  reserve1;            /* reserved                      */
        unsigned char   address[MAX_LINK_ADDR_LEN];
                                             /* address                       */
} LINK_ADDRESS;

typedef struct tg_defined_chars
{
         unsigned char   effect_cap;         /* effective capacity            */
         unsigned char   reserve1[5];        /* reserved                      */
         unsigned char   connect_cost;       /* connection cost               */
         unsigned char   byte_cost;          /* byte cost                     */
         unsigned char   reserve2;           /* reserved                      */
         unsigned char   security;           /* security                      */
         unsigned char   prop_delay;         /* propagation delay             */
         unsigned char   modem_class;        /* modem class                   */
         unsigned char   user_def_parm_1;    /* user_defined parameter 1      */
         unsigned char   user_def_parm_2;    /* user_defined parameter 2      */
         unsigned char   user_def_parm_3;    /* user_defined parameter 3      */
} TG_DEFINED_CHARS;

typedef struct port_spec_data
{
   unsigned char  port_data[SIZEOF_PORT_SPEC_DATA];
}  PORT_SPEC_DATA;

typedef struct link_spec_data
{
   unsigned char  link_data[SIZEOF_LINK_SPEC_DATA];
}  LINK_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_PORT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

指定された **port_name**（下記のパラメーターを参照してください）は、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

port_name

照会中のポート名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

dlc_name

DLC 名のフィルター。 これは、すべてゼロかまたはローカルで表示可能な文字セットで 8 バイト・ストリングに設定しなければなりません。 このフ

ィールドが設定された場合には、この DLC に属しているポートだけが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

port_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

port_summary.port_name
このリンク・ステーションと関連したポート名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

port_summary.description
資源の記述（DEFINE_PORT で指定されたものです）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

port_summary.port_state
ポートの現行状態を指定します。

AP_NOT_ACTIVE
AP_PENDING_ACTIVE
AP_ACTIVE
AP_PENDING_INACTIVE

port_summary.dlc_name
DLC の名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

port_detail.overlay_size
この項目のバイト数（すべての **link_spec_data** を含む）であり、したがって次に戻される項目（ある場合）へのオフセット。

port_detail.port_name
このリンク・ステーションと関連したポート名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

port_detail.det_data.port_state
ポートの現行状態を指定します。

AP_NOT_ACTIVE
AP_PENDING_ACTIVE
AP_ACTIVE
AP_PENDING_INACTIVE

port_detail.det_data.dlc_type
DLC のタイプ。Communications Server は以下のタイプをサポートします。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

port_detail.det_data.port_sim_rim
初期設定モードの設定 (SIM) および 初期設定モードの受信 (RIM) をサポートするかどうか（AP_YES または AP_NO）を指定します。

port_detail.det_data.def_ls_good_xids
このポートが最後に開始して以降、このポート上のすべての定義済みリンク・ステーションで発生した正常な XID 交換の合計数

port_detail.det_data.def_ls_bad_xids
このポートが最後に開始して以降、このポート上のすべての定義済みリンク・ステーションで発生した不成功の XID 交換の合計数

port_detail.det_data.dyn_ls_good_xids
このポートが最後に開始して以降、このポート上のすべての動的リンク・ステーションで発生した正常な XID 交換の合計数

port_detail.det_data.dyn_ls_bad_xids
このポートが最後に開始して以降、このポート上のすべての動的リンク・ステーションで発生した不成功の XID 交換の合計数

port_detail.def_data.description
資源の記述（DEFINE_PORT で指定されたものです）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

port_detail.def_data.dlc_name
関連した DLC の名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

port_detail.def_data.port_type
ポートによって使用されている回線のタイプを指定します。この値は、以下の値の 1 つと一致します。

AP_PORT_NONSWITCHED
AP_PORT_SWITCHED
AP_PORT_SATF

port_detail.def_data.port_number
ポート番号

port_detail.def_data.max_rcv_btu_size
受信できる最大 BTU サイズ

port_detail.def_data.tot_link_act_lim
合計リンク活動化の限界

port_detail.def_data.inb_link_act_lim
インバウンド・リンク活動化の限界

port_detail.def_data.out_link_act_lim
アウトバウンド・リンク活動化の限界

port_detail.def_data.ls_role
リンク・ステーションの役割。これは、折衝可能 (AP_LS_NEG)、1 次 (AP_LS_PRI)、または 2 次 (AP_LS_SEC) のいずれでも構いかまいません。**implicit_hpr_support** が AP_NO に設定されている場合は、予約済みです。

def_data.implicit_dspu_template
DEFINE_DSPU_TEMPLATE verb によって定義されている DSPU テンプレートを指定します。このテンプレートは、このポート上で活動化されている暗黙リンクに対して、ローカル・ノードが PU 集信を提供する際に、定義のために使用されます。リンクが活動化された時点に、指定されたテンプレートが存在しない場合 (または既にインスタンスが限度数に達している場合) 、活動化は失敗します。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

def_data.implicit_dspu_services フィールドがAP_PU_CONCENTRATION に設定されていない場合、このフィールドは予約されます。

def_data.implicit.dspu_services
このポート上で活動化された暗黙リンク全体にわたって、ローカル・ノードによってダウンストリーム PU に提供されるサービスを指定します。 以下の値のいずれか 1 つに設定されます。

AP_DLUR
ローカル・ノードは、ダウンストリーム PU に DLUR サービスを提供します（DEFINE_DLUR_DEFAULTS verb によって構成済みの省略時 DLUS を使用して）。

AP_PU_CONCENTRATION
ローカル・ノードは、ダウンストリーム PU に対して PU 集信を提供します (また、フィールド **def_data.implicit_dspu_template** で指定された DSPU テンプレートが指定したように、位置定義を入れます)。

AP_NONE
ローカル・ノードは、このダウンストリーム PU に対するサービスを何も提供しません。

def_data.implicit_deact_timer
限定資源のリンク非活動化タイマー（秒数）。**implicit_limited_resource** が AP_YES または AP_NO_SESSIONS に設定されている場合で、このタイマーの期間中にデータがリンクを通らず、またリンクを使用しているセッションがない場合は、HPR 可能な暗黙リンクは自動的に非活動化されます。

implicit_limited_resource が AP_INACTIVITY に設定されている場合で、このタイマーの期間中にデータがこのリンクを通らなかった場合には、暗黙リンクは自動的に非活動化されます。

ゼロが指定されていると、省略時値 30 が使用されます。それ以外の場合、最小値は 5 です（これより小さく設定されると、指定された値は無視され、5 が使用されます）。**implicit_limited_resource** が AP_NO に設定されていない場合、このパラメーターが予約されることに注意してください。

port_detail.def_data.act_xid_exchange_limit
活動化 XID 交換の限界

port_detail.def_data.nonact_xid_exchange_limit
非活動化 XID 交換の限界

port_detail.def_data.ls_xmit_rcv_cap
リンク・ステーションの送受信能力を指定します。これは、両方向同時 (AP_LS_TWS) かまたは両方向交互 (AP_LS_TWA) のいずれかです。

port_detail.def_data.max_ifrm_rcvd
肯定応答が送信される前に、ローカル・リンク・ステーションが受信することができる I フレームの最大数。 範囲は 1 から 127 までです。

port_detail.def_data.target_pacing_count
1 以上 32 767 以下の数値で、この TG 上の BIND に対する望ましいペーシング・ウィンドウ・サイズを示します。この数値は、固定バインド・ペーシングが実行される場合のみ有効です。Communications Server は現在この値を使用していないことに注意してください。

port_detail.def_data.max_send_btu_size
送信できる最大 BTU サイズ

port_detail.def_data.dlc_data.length
ポート・アドレスの長さ

port_detail.def_data.dlc_data.address
ポート・アドレス

port_detail.def_data.hpr_dlc_data.length
HPR ポート・アドレスの長さ

port_detail.def_data.hpr_dlc_data.address
HPR ポート・アドレス。これは現在、HPR リンクをサポートするとき、使用されます。このフィールドは、このポートを使用しているリンク・ステーション上で交換された XID3 上の X'61' 制御ベクトルの X'80' サブフィールドで、Communications Server によって送信される情報を指定します。

port_detail.def_data.implicit_cp_cp_sess_support
このポート上にない暗黙リンク・ステーションに対して、CP-CP セッションを許可するかどうか（AP_YES または AP_NO）を指定します。

port_detail.def_data.implicit_limited_resource
リンクを使用しているセッションがないとき、このポート上にない暗黙リンク・ステーションを非活動化すべきかどうかを指定します。 以下の値のいずれか 1 つに設定されます。

AP_NO
暗黙リンクは、限定資源ではなく、自動的には非活動化されません。

AP_YES または AP_NO_SESSIONS
暗黙リンクは、限定資源であり、活動状態のセッションがそれらを使用していないとき、自動的に非活動化されます。

AP_INACTIVITY
暗黙リンクは、限定資源であり、活動状態のセッションがそれらを使用していないとき、またはリンク上で **implicit_deact_timer** フィールドで指定された期間中にデータが流れなかったとき、自動的に非活動化されます。

port_detail.def_data.implicit_hpr_support
暗黙リンク上で HPR をサポートするかどうか（AP_YES または AP_NO）を指定します。

port_detail.def_data.implicit_link_lvl_error
リンク・レベルのエラー回復を使用中の暗黙リンクに HPR トラフィックを送信するかどうか（AP_YES または AP_NO）を指定します。

port_detail.def_data.default_tg_chars
TG 特性（41ページの『DEFINE_COS』を参照してください）。これらは、このポート上にない暗黙リンク・ステーションや、**use_default_tg_chars** を指定した定義済みリンク・ステーションに対して使用されます。

port_detail.def_data.discovery_supported
このポート上で検索ディスカバリー機能を実行するかどうか（AP_YES または AP_NO）を指定します。

port_detail.def_data.port_spec_data_len

ACTIVATE_PORT シグナルで、未変更のままポートに渡されたデータの埋込みなしの長さをバイトで表したもの。このデータは、PORT_DETAIL 構造に連結されています。

port_detail.def_data.link_spec_data_len

初期化中に、リンク・ステーション・コンポーネントに未変更のまま渡されたデータ。このデータは、ポート固有のデータのすぐ後ろに続く PORT_DETAIL 構造に連結しています。ポート固有データおよびリンク固有データはともに、4 バイト境界の終りまで埋め込まれています。ポート固有データとリンク固有データの間には、明示の埋込みはありません。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PORT_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_PU

QUERY_PU は、ローカル PU のリストおよびそれらと関連したリンクを戻します。

この情報は、リストとして戻されます。 特定の PU についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**pu_name** フィールドを設定しなくてはなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

この verb は、ローカル PU を、ホスト・システムに直接に接続するか、または DLUR を通じて接続するかを指定します。 **host_attachment** フィールドは、指定された接続タイプの情報だけが戻されるように、フィルターとして使用することができます。

VCB 構造

```
typedef struct query_pu
{
        unsigned short  opcode;                 /* verb operation code          */
        unsigned char   reserv2;                /* reserved                     */
        unsigned char   format;                 /* format                       */
        unsigned short  primary_rc;             /* primary return code          */
        unsigned long   secondary_rc;           /* secondary return code        */
        unsigned char   *buf_ptr;               /* pointer to buffer            */
        unsigned long   buf_size;               /* buffer size                  */
        unsigned long   total_buf_size;         /* total buffer size required   */
        unsigned short  num_entries;            /* number of entries            */
        unsigned short  total_num_entries;      /* total number of entries      */
        unsigned char   list_options;           /* listing options              */
        unsigned char   reserv3;                /* reserved                     */
        unsigned char   pu_name[8];             /* PU name                      */
        unsigned char   host_attachment;        /* Host Attachment              */
} QUERY_PU;

typedef struct pu_data
{
        unsigned short  overlay_size;           /* size of this entry           */
        unsigned char   pu_name[8];             /* PU name                      */
        unsigned char   description[RD_LEN];    /* resource description         */
        unsigned char   ls_name[8];             /* LS name                      */
        unsigned char   pu_sscp_sess_active;    /* Is PU-SSCP session active    */
        unsigned char   host_attachment;        /* Host attachment              */
        SESSION_STATS   pu_sscp_stats;          /* PU-SSCP session statistics   */
        unsigned char   reserva[20];            /* reserved                     */
} PU_DATA;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;            /* session receive RU size      */
        unsigned short  send_ru_size;           /* session send RU size         */
        unsigned short  max_send_btu_size;      /* max send BTU size            */
        unsigned short  max_rcv_btu_size;       /* max rcv BTU size             */
        unsigned short  max_send_pac_win;       /* max send pacing window size  */
```

```
        unsigned short  cur_send_pac_win;      /* curr send pacing window size  */
        unsigned short  max_rcv_pac_win;       /* max recv pacing window size   */
        unsigned short  cur_rcv_pac_win;       /* current receive pacing        */
                                               /* window size                   */
        unsigned long   send_data_frames;      /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                               /* num of FMD data frames sent   */
        unsigned long   send_data_bytes;       /* number of data bytes sent     */
        unsigned long   rcv_data_frames;       /* num data frames received      */
        unsigned long   rcv_fmd_data_frames;
                                               /* num of FMD data frames rcvd   */
        unsigned long   rcv_data_bytes;        /* number of data bytes received */
        unsigned char   sidh;                  /* session ID high byte          */
                                               /* (from LFSID)                  */
        unsigned char   sidl;                  /* session ID low byte           */
                                               /* (from LFSID)                  */
        unsigned char   odai;                  /* ODAI bit set                  */
        unsigned char   ls_name[8];            /* Link station name             */
        unsigned char   reserve;               /* reserved                      */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_PU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

指定された **pu_name** (後に示すパラメーターを参照) は、実際に情報が戻される開始点を指定するのに使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

pu_name
リストされる最初の PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。**list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

host_attachment
ホスト接続のためのフィルター。

AP_NONE
すべてのローカル PU についての情報を戻します。

AP_DLUR_ATTACHED
DLUR によってサポートされているすべてのローカル PU についての情報を戻します。

AP_DIRECT_ATTACHED
ホスト・システムに直接接続している PU だけについての情報を戻します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

pu_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

pu_data.pu_name
PU 名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

pu_data.description
資源の記述（DEFINE_LS または DEFINE_INTERNAL_PU で指定されたものです）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

pu_data.ls_name
この PU と関連したリンク・ステーションの名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

pu_data.pu_sscp_sess_active
PU-SSCP セッションが活動状態かどうか（AP_YES または AP_NO）を指定します。

pu_data.host_attachment
ローカル PU ホスト接続タイプ。

AP_DLUR_ATTACHED
PU は、DLUR を使用してホスト・システムに接続しています。

AP_DIRECT_ATTACHED
PU は、直接ホスト・システムに接続しています。

pu_data.pu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

pu_data.pu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

pu_data.pu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

pu_data.pu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

pu_data.pu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_data.pu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_data.pu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_data.pu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_data.pu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

pu_data.pu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

pu_data.pu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

pu_data.pu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

pu_data.pu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

pu_data.pu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

pu_data.pu_sscp_stats.sidh
セッション ID の高位バイト

pu_data.pu_sscp_stats.sidl
セッション ID の低位バイト

pu_data.pu_sscp_stats.odai
起点宛先アドレス標識。セッションを立ち上げるとき、ACTPU の送信側は、ローカル・ノードが 1 次リンク・ステーションを含む場合、このフィールドをゼロに設定し、ACTPU の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールドを 1 に設定します。

pu_data.pu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_PU_TYPE
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_RTP_CONNECTION

QUERY_RTP_CONNECTION は、ネットワーク・ノードまたはエンド・ノードで使用され、ノードがエンドポイントである Rapid Transport Protocol (RTP) 接続についてのリスト情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定の RTP 接続についての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**rtp_name** フィールドを設定しておかなければなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは **rtp_name** によって配列されます。 まず名前の長さによって配列され、長さが同じ名前の場合は ASCII 辞書配列順序に従って配列されます（IBM 6611 APPN MIB の配列順序と同一です)。 AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

VCB 構造

```
typedef struct query_rtp_connection
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* Primary return code           */
        unsigned long   secondary_rc;        /* Secondary return code         */
        unsigned char   *buf_ptr;            /* pointer to buffer             */
        unsigned long   buf_size;            /* buffer size                   */
        unsigned long   total_buf_size;      /* total buffer size required    */
        unsigned short  num_entries;         /* number of entries             */
        unsigned short  total_num_entries;   /* total number of entries       */
        unsigned char   list_options;        /* listing options               */
        unsigned char   reserv3;             /* reserved                      */
        unsigned char   rtp_name[8];         /* name of RTP connection      */
} QUERY_RTP_CONNECTION;

typedef struct rtp_connection_summary
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   rtp_name[8];         /* RTP connection name         */
        unsigned char   first_hop_ls_name[8];
                                             /* LS name of first hop          */
        unsigned char   dest_node_name[17];  /* fully qualified name of     */
                                             /* destination node              */
        unsigned char   reserv1;             /* reserved                      */
        unsigned char   cos_name[8];         /* class-of-service name       */
        unsigned short  num_sess_active;     /* number of active sessions     */
} RTP_CONNECTION_SUMMARY;

typedef struct rtp_connection_detail
{
        unsigned short  overlay_size;        /* size of this entry            */
```

```
        unsigned char   rtp_name[8];         /* RTP connection name          */
        unsigned char   first_hop_ls_name[;8];
                                             /* LS name of first hop           */
        unsigned char   dest_node_name[17];  /* fully qualified name of      */
                                             /* destination node               */
        unsigned char   reserv1[3];          /* reserved                     */
        unsigned char   cos_name[8];         /* class-of-service name        */
        unsigned short  max_btu_size;        /* max BTU size                   */
        unsigned long   liveness_timer;      /* liveness timer                 */
        unsigned char   local_tcid[8];       /* local TCID                   */
        unsigned char   remote_tcid[8];      /* remote TCID                  */
        RTP_STATISTICS  rtp_stats;           /* RTP statistics                 */
        unsigned short  num_sess_active;     /* number of active sessions      */
        unsigned char   reserv2[16];         /* reserved                     */
        unsigned short  rscv_len;            /* length of appended RSCV        */
} RTP_CONNECTION_DETAIL;

typede  struct rtp_statistics
{
        unsigned long   bytes_sent;          /* total number of bytes sent     */
        unsigned long   bytes_received;      /* total number of bytes received */
        unsigned long   bytes_resent;        /* total number of bytes resent   */
        unsigned long   bytes_discarded;     /* total number bytes discarded   */
        unsigned long   packets_sent;        /* total number of packets sent   */
        unsigned long   packets_received;    /* total number packets received  */
        unsigned long   packets_resent;      /* total number of packets resent */
        unsigned long   packets_discarded;   /* total number packets discarded */
        unsigned long   gaps_detected;       /* gaps detected                  */
        unsigned long   send_rate;           /* current send rate              */
        unsigned long   max_send_rate;       /* maximum send rate              */
        unsigned long   min_send_rate;       /* minimum send rate              */
        unsigned long   receive_rate;        /* current receive rate           */
        unsigned long   max_receive_rate;    /* maximum receive rate           */
        unsigned long   min_receive_rate;    /* minimum receive rate           */
        unsigned long   burst_size;          /* current burst size             */
        unsigned long   up_time;             /* total uptime of connection     */
        unsigned long   smooth_rtt;          /* smoothed round-trip time       */
        unsigned long   last_rtt;            /* last round-trip time           */
        unsigned long   short_req_timer;     /* SHORT_REQ timer duration       */
        unsigned long   short_req_timeouts;  /* number of SHORT_REQ timeouts   */
        unsigned long   liveness_timeouts;   /* number of liveness timeouts    */
        unsigned long   in_invalid_sna_frames;
                                             /* number of invalid SNA frames   */
                                             /* received                       */
        unsigned long   in_sc_frames;        /* number of SC frames received   */
        unsigned long   out_sc_frames;       /* number of SC frames sent       */
        unsigned char   reserve[40];         /* reserved                       */
} RTP_STATISTICS;
```

注: SNA によって定義されているように、**rtp_connection_detail** オーバーレーの後ろには、Route Selection Control Vector (RSCV) が続いています。 RTP 接続セットアップのあと、およびすべてのパス・スイッチの前に、 RTP接続のための RSCV が以下のように各ノードごとに格納され、表示されます。

- RSCV は、ローカル・ノードからパートナー RTP ノードへのすべてのホップ回数を含んでいます。
- パートナー RTP ノードが、RTP 接続を活動化させるセッションのエンドポイントでない場合、RSCV はまた、パートナー RTP ノードから先行する 1 つの 境界機能ホップ も格納します。
- RSCV は、たとえ、ローカル・ノードがセッションのエンドポイントを含まない場合でも、ローカル・ノードに導く境界機能ホップを含むことは決してありません。

パス・スイッチが起こったあと、格納と表示が行われた RSCV には、ローカル・ノードからパートナー RTP ノードへのホップ回数だけが含まれます （境界機能ホップは、決して含まれません)。

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode
AP_QUERY_RTP_CONNECTION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr
リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size
提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries
戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options
リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

rtp_name は、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST
rtp_name は無視され、戻されるリストが、リストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

rtp_name
RTP 結合名。 この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

rtp_connection_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

rtp_connection_summary.rtp_name
RTP 結合名。この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

rtp_connection_summary.first_hop_ls_name
RTP 接続の先頭ホップのリンク・ステーション名。この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

rtp_connection_summary.dest_node_name
RTP 接続の宛先ノードの完全修飾された 17 バイトの名前で、EBCDIC ピリオドによって連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

rtp_connection_summary.cos_name
RTP 接続に対するサービス・クラス名。8 バイトの英数字のタイプ A の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

rtp_connection_summary.num_sess_active
RTP 接続上で現在活動状態のセッションの数

rtp_connection_detail.overlay_size
この項目のバイト数（追加されたすべての RSCV を含む）であり 、したがって次に戻される項目（ある場合）へのオフセット 。

rtp_connection_detail.rtp_name
RTP 結合名。この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

rtp_connection_detail.first_hop_ls_name
RTP 接続の先頭ホップのリンク・ステーション名。この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

rtp_connection_detail.dest_node_name
RTP 接続の宛先ノードの完全修飾された 17 バイトの名前で、EBCDIC ピリオドによって連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

rtp_connection_detail.cos_name
RTP 接続に対するサービス・クラス名。8 バイトの英数字のタイプ A の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

rtp_connection_detail.max_btu_size
RTP 接続に対する最大 btu サイズで、バイト数です。

rtp_connection_detail.liveness_timer
RTP 接続に対するライブネス・タイマーで、秒数です。

rtp_connection_detail.local_tcid
RTP 接続に対するローカル TCID

rtp_connection_detail.remote_tcid
RTP 接続に対するリモート TCID

rtp_connection_detail.rtp_stats.bytes_sent
ローカル・ノードがこの RTP 接続上で送信したバイトの合計数

rtp_connection_detail.rtp_stats.bytes_received
ローカル・ノードがこの RTP 接続上で受信したバイトの合計数

rtp_connection_detail.rtp_stats.bytes_resent
転送中のロスのため、ローカル・ノードが再送したバイトの合計数

rtp_connection_detail.rtp_stats.bytes_discarded
RTP 接続の他の終端から送信され、すでに受信済みのデータの重複として廃棄されたバイトの合計数

rtp_connection_detail.rtp_stats.packets_sent
ローカル・ノードがこの RTP 接続上で送信したパケットの合計数

rtp_connection_detail.rtp_stats.packets_received
ローカル・ノードがこの RTP 接続上で受信したパケットの合計数

rtp_connection_detail.rtp_stats.packets_resent
転送中のロスのため、ローカル・ノードが再送したパケットの合計数

rtp_connection_detail.rtp_stats.packets_discarded
RTP 接続の他の終端から送信され、すでに受信済みのデータの重複として廃棄されたパケットの合計数

rtp_connection_detail.rtp_stats.gaps_detected
ローカル・ノードによって検出されたギャップの合計数。各ギャップは、1 つ以上の失われたフレームに対応します。

rtp_connection_detail.rtp_stats.send_rate
この RTP 接続上の現行の送信レート（秒当たりのキロビット数です）。これは、ARB アルゴリズムで計算される最大許容の送信レートです。

rtp_connection_detail.rtp_stats.max_send_rate
この RTP 接続上の最大送信レート（秒当たりのキロビット数です）

rtp_connection_detail.rtp_stats.min_send_rate
この RTP 接続上の最小送信レート（秒当たりのキロビット数です）

rtp_connection_detail.rtp_stats.receive_rate
この RTP 接続上の現行の受信レート（秒当たりのキロビット数です）これは、最後の測定間隔において計算された実際の受信レートです。

rtp_connection_detail.rtp_stats.max_receive_rate
この RTP 接続上の最大受信レート（秒当たりのキロビット数です）

rtp_connection_detail.rtp_stats.min_receive_rate
この RTP 接続上の最小受信レート（秒当たりのキロビット数です）

rtp_connection_detail.rtp_stats.burst_size
RTP 接続上の現行の切離しサイズで、バイト数です。

rtp_connection_detail.rtp_stats.up_time
RTP 接続が活動状態であった秒数の合計

rtp_connection_detail.rtp_stats.smooth_rtt
ローカル・ノードとパートナー RTP ノード間の、平滑化された往復時間（ミリ秒です）

rtp_connection_detail.rtp_stats.last_rtt
ローカル・ノードとパートナー RTP ノード間の、最後に測定された往復時間（ミリ秒です）

rtp_connection_detail.rtp_stats.short_req_timer
SHORT_REQ タイマーに対して使用される現行期間（ミリ秒です）

rtp_connection_detail.rtp_stats.short_req_timeouts
この RTP 接続に対して SHORT_REQ タイマーが期限切れした合計回数

rtp_connection_detail.rtp_stats.liveness_timeouts
この RTP 接続に対してライブネス・タイマーが期限切れした合計回数ライブネス・タイマーが期限切れするのは、接続が **rtp_connection_detail.liveness_timer** に指定した期間にわたってアイドルであったときです。

rtp_connection_detail.rtp_stats.in_invalid_sna_frames
この RTP 接続上で受信され、無効として廃棄された SNA フレームの合計数

rtp_connection_detail.rtp_stats.in_sc_frames
この RTP 接続上で受信されたセッション制御フレームの合計数

rtp_connection_detail.rtp_stats.out_sc_frames
この RTP 接続上で送信されたセッション制御フレームの合計数

rtp_connection_detail.num_sess_active
RTP 接続上で現在活動状態のセッションの数

rtp_connection_detail.rscv_len
RTP 接続に対する、追加 Route Selection Control Vector の長さ（なにも追加されていない場合には、長さはゼロです）。RSCV は、次の詳細項目の位置合せが確実に正しくなるように、4 バイト境界の終りまで埋め込まれます。しかし、**rscv_len** には、この埋込みがありません。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_RTP_CONNECTION

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_SESSION

QUERY_SESSION は、ノードがエンドポイントであるセッションについてのリスト情報を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。特定のセッションについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**session_id** フィールドを設定しておかなければなりません。 それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 **lu_name** (または **lu_alias**) フィールドおよび **plu_alias** (または **fqplu_name**) フィールドを常に設定しなくてはならないことに注意してください。 **lu_name** が非ゼロの場合、**lu_name** が **lu_alias** に優先して使用されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

plu_alias をすべてゼロに設定すると **fqplu_name** の値が使用され、それ以外の場合は **plu_alias** が常に使用されて **fqplu_name** は無視されます。

戻されたセッションのリストは、セッションが関連しているモードの名前でフィルター処理することができます。 この場合、**mode_name** フィールドを設定しておかなければなりません（それ以外の場合、このフィールドはすべてゼロに設定しておかなければなりません)。

VCB 構造

```
typedef struct query_session
{
        unsigned short  opcode;                 /* verb operation code         */
        unsigned char   reserv2;                /* reserved                    */
        unsigned char   format;                 /* format                      */
        unsigned short  primary_rc;             /* primary return code         */
        unsigned long   secondary_rc;           /* secondary return code       */
        unsigned char   *buf_ptr;               /* pointer to buffer           */
        unsigned long   buf_size;               /* buffer size                 */
        unsigned long   total_buf_size;         /* total buffer size required  */
        unsigned short  num_entries;            /* number of entries           */
        unsigned short  total_num_entries;      /* total number of entries     */
        unsigned char   list_options;           /* listing options             */
        unsigned char   reserv3;                /* reserved                    */
        unsigned char   lu_name[8];             /* LU name                     */
        unsigned char   lu_alias[8];            /* LU alias                    */
        unsigned char   plu_alias[8];           /* partner LU alias            */
        unsigned char   fqplu_name[17];         /* fully qualified partner     */
                                                /* LU name                     */
        unsigned char   mode_name[8];           /* mode name                   */
        unsigned char   session_id[8];          /* session ID                  */
} QUERY_SESSION;
typedef struct session_summary
{
        unsigned short  overlay_size;           /* size of this entry          */
        unsigned char   plu_alias[8];           /* partner LU alias            */
        unsigned char   fqplu_name[17];         /* fully qualified partner     */
                                                /* LU name                     */
```

```
        unsigned char   reserv3[1];             /* reserved                    */
        unsigned char   mode_name[8];           /* mode name                   */
        unsigned char   session_id[8];          /* session ID                  */
        FQPCID          fqpcid;                 /* fully qualified procedure   */
                                                /* correlator ID               */
} SESSION_SUMMARY;

typedef struct session_detail
{
        unsigned short  overlay_size;           /* size of this entry          */
        unsigned char   plu_alias[8];           /* partner LU alias            */
        unsigned char   fqplu_name[17];         /* fully qualified partner     */
                                                /* LU name                     */
        unsigned char   reserv3[1];             /* reserved                    */
        unsigned char   mode_name[8];           /* mode name                   */
        unsigned char   session_id[8];          /* session ID                  */
        FQPCID          fqpcid;                 /* fully qualified procedure   */
                                                /* correlator ID               */
        unsigned char   cos_name[8];            /* Class-of-service name       */
        unsigned char   trans_pri;              /* Transmission priority:      */
        unsigned char   ltd_res;                /* Session spans a limited     */
                                                /* resource                    */
        unsigned char   polarity;               /* Session polarity            */
        unsigned char   contention;             /* Session contention          */
        SESSION_STATS   sess_stats;             /* Session statistics          */
        unsigned char   duplex_support;         /* full-duplex support         */
        unsigned char   reserv3a[2];            /* reserved                    */
        unsigned char   reserva[20];            /* reserved                    */
        unsigned char   rscv_len;               /* Length of following RSCV    */
} SESSION_DETAIL;

typedef struct fqpcid
{
        unsigned char   pcid[8];                /* pro correlator identifier   */
        unsigned char   fqcp_name[17];          /* orig's network qualified    */
                                                /* CP name                     */
        unsigned char   reserve3[3];            /* reserved                    */
} FQPCID;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;            /* session receive RU size     */
        unsigned short  send_ru_size;           /* session send RU size        */
        unsigned short  max_send_btu_size;      /* Maximum send BTU size       */
        unsigned short  max_rcv_btu_size;       /* Maximum rcv BTU size        */
        unsigned short  max_send_pac_win;       /* Max send pacing window size */
        unsigned short  cur_send_pac_win;       /* Curr send pacing window size*/
        unsigned short  max_rcv_pac_win;        /* Max receive pacing win size */
        unsigned short  cur_rcv_pac_win;        /* Curr rec pacing window size */
        unsigned long   send_data_frames;       /* Number of data frames sent  */
        unsigned long   send_fmd_data_frames;   /* num of FMD data frames sent */
        unsigned long   send_data_bytes;        /* Number of data bytes sent   */
        unsigned long   rcv_data_frames;        /* Num data frames received    */
        unsigned long   rcv_fmd_data_frames;    /* num of FMD data frames recvd*/
        unsigned long   rcv_data_bytes;         /* Num data bytes received     */
        unsigned char   sidh;                   /* Session ID high byte        */
        unsigned char   sidl;                   /* Session ID low byte         */
```

```
        unsigned char   odai;                   /* ODAI bit set                 */
        unsigned char   ls_name[8];             /* Link station name            */
        unsigned char   reserve;                /* reserved                     */
} SESSION_STATS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_SESSION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY

要約情報のみを戻します。

AP_DETAIL

詳細情報を戻します。

lu_name（または、**lu_name** がすべてゼロに設定されている場合は、**lu_alias**）、**plu_alias**（または、**plu_alias** がすべてゼロに設定されている場合は、**fqplu_name**）、**mode_name**、および **session_id** の組合せを指定（下記のパラメーターを参照してください）すると、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST

session_id は無視され、戻されるリストがリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

lu_name

LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。このフィールドをすべてゼロに設定すると、**lu_alias** フィールドが索引の判別に使用されます。

lu_alias

ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** と **lu_alias** の両フィールドがすべてゼロに設定されている場合、制御点と関連した LU（省略時 LU）が使用されます。

plu_alias

パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロの設定されている場合、索引を判別するため、**fqplu_name** フィールドが使用されます。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

mode_name

モード名フィルター。 すべてゼロに設定するか 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) に設定しなくてはならず、右側は EBCDIC スペースで埋めます。 このフィールドが設定されている場合には、このモードと関連したセッションだけが戻されます。 このフィールドがすべてゼロに設定されると、このフィールドは無視されます。

session_id

セッションの 8 バイト識別子。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

session_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

session_summary.plu_alias
パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

session_summary.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。

session_summary.mode_name
モード名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

session_summary.session_id
セッションの 8 バイト識別子。

session_summary.fqpcid.pcid
プロシージャー相関 ID。これは 8 バイトの 16 進数ストリングです。

session_summary.fqpcid.fqcp_name
完全修飾制御点名。この 17 バイトの名前は、EBCDIC ピリオドによって連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側が EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

session_detail.overlay_size
この項目のバイト数（追加されたすべての RSCV を含む）であり 、したがって次に戻される項目（ある場合）へのオフセット。

session_detail.plu_alias
パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

session_detail.fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。

session_detail.mode_name
モード名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

session_detail.session_id
セッションの 8 バイト識別子。

session_detail.fqpcid.pcid
プロシージャー相関 ID。これは 8 バイトの 16 進数ストリングです。

session_detail.fqpcid.fqcp_name
完全修飾制御点名。この 17 バイトの名前は、EBCDIC ピリオドによって連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側が EBCDIC スペースで埋め込まれています。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

session_detail.cos_name
サービス・クラス名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

session_detail.trans_pri
伝送優先順位。以下の値のいずれか 1 つに設定されます。

AP_LOW
AP_MEDIUM
AP_HIGH
AP_NETWORK

session_detail.ltd_res
セッションが限定資源リンクを使用するかどうか（AP_YES または AP_NO）を指定します。

session_detail.polarity
セッションの極性を指定（AP_PRIMARY または AP_SECONDARY）します。

session_detail.contention
セッションの競合極性を指定します。これは、ローカル LU がこのセッションの使用に対する '優先的選択権' をもつかどうか (AP_CONWINNER)、またはローカル LU がセッションを使用する前に送信権要求をしなければならないかどうか (AP_CONLOSER) を指定します。

session_detail.sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

session_detail.sess_stats.send_ru_size
送信のときの最大 RU サイズ

session_detail.sess_stats.max_send_btu_size
送信できる最大 BTU サイズ

session_detail.sess_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

session_detail.sess_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

session_detail.sess_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

session_detail.sess_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

session_detail.sess_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

session_detail.sess_stats.send_data_frames
送信される通常フロー・データ・フレームの数

session_detail.sess_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

session_detail.sess_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

session_detail.sess_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

session_detail.sess_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

session_detail.sess_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

session_detail.sess_stats.sidh
セッション ID の高位バイト

session_detail.sess_stats.sidl
セッション ID の低位バイト

session_detail.sess_stats.odai
起点宛先アドレス標識。セッションを立ち上げるとき、ローカル・ノードが 1 次リンク・ステーションを含む場合、BIND の送信側は、このフィールドをゼロに設定します。BIND の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールド 1 に設定します。

session_detail.sess_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドは、セッション・データが流れるリンクとセッション統計を関係づけるために使用することができます。

session_detail.duplex_support
BIND で折衝されたとおりに会話二重サポートを戻します。これは、以下の値の 1 つになります。

AP_HALF_DUPLEX
半二重会話だけがサポートされます。

AP_FULL_DUPLEX
半二重会話と同様に全二重会話がサポートされます。優先データもまたサポートされます。

session_detail.rscv_len

session_detail 構造に追加される RSCV の長さ（なにも追加されていない場合には、長さはゼロです）。RSCV は、4 バイト境界の終りまで埋め込まれます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_SESSION_ID

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_STATISTICS

QUERY_STATISTICS は、リンク・ステーションおよびポート統計を照会します。 Communications Server は、この照会を直接に DLC に渡します。統計の形式は、DLC の実行の仕方によって異なります。

VCB 構造

```
typedef struct query_statistics
{
        unsigned short  opcode;           /* verb operation code        */
        unsigned char   reserv2;          /* reserved                   */
        unsigned char   format;           /* format                     */
        unsigned short  primary_rc;       /* primary return code        */
        unsigned long   secondary_rc;     /* secondary return code      */
        unsigned char   name[8];          /* LS or port name            */
        unsigned char   stats_type;       /* LS or port statistics?     */
        unsigned char   table_type;       /* statistics table requested */
        unsigned char   reset_stats;      /* reset the statistics?      */
        unsigned char   dlc_type;         /* type of DLC                */
        unsigned char   statistics[256];  /* current statistics         */
        unsigned char   reserva[20];      /* reserved                   */
} QUERY_STATISTICS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_STATISTICS

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

name リンク・ステーションまたはポートに対して定義された名前（**stats_type**パラメーターの設定値に応じて異なります)。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。 Communications Server は、正しいリンク・ステーションまたはポートに対して応答を関係づけるためにこの名前を使用します。

stats_type

統計が要求されている資源のタイプ。 以下の値のいずれか 1 つに設定しなければなりません。

AP_LS
AP_PORT

table_type

要求された統計表のタイプ。これは、以下の情報カテゴリーの 1 つに設定しなければなりません。

AP_STATS_TBL
統計情報を戻すことを指定します。

AP_ADMIN_TBL
管理情報を戻すことを指定します。

AP_OPER_TBL
操作情報を戻すことを指定します。 各カテゴリーに対して戻される情報の形式は、DLC 実行に特有のものです。

reset_stats
統計をリセットするかどうか（AP_YES または AP_NO）を指定します。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

dlc_type
DLC のタイプ。 このフィールドの値は、DLC 実行に特有のものです。値は、以下のとおりです。

AP_ANYNET
AP_LLC2
AP_OEM_DLC
AP_SDLC
AP_TWINAX
AP_X25

statistics
リンク・ステーションまたはポートの現行の統計

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_LINK_NAME

AP_INVALID_PORT_NAME
AP_INVALID_STATS_TYPE
AP_INVALID_TABLE_TYPE

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc

AP_LINK_DEACTIVATED

AP_PORT_DEACTIVATED

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_TP

QUERY_TP は、ローカル LU によって現在使用されているトランザクション・プログラムについての情報を戻します。

この情報は、リストとして戻されます。 特定のトランザクション・プログラムについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**tp_name** フィールドを設定しておかなければなりません。 **list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合には、このフィールドは無視されます。 **lu_name** または **lu_alias** フィールドを常に設定しておかなければならない点に注意してください。 **lu_name** フィールドは、非ゼロの場合、**lu_alias** フィールドに優先して使用されます。 リスト形式の使用方法の背景については、14ページの『ノードの照会』 を参照してください。

このリストは、同じ長さの名前に対しては、EBCDIC 辞書配列順序を使用して **tp_name** によって配列されます。 この verb は、ローカル LU による TP の使用開始以降に判別された情報を戻します。 QUERY_TP_DEFINITION verb は、定義情報だけを戻します。

VCB 構造

```
typedef struct query_tp
{
        unsigned short  opcode;              /* Verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* Primary return code          */
        unsigned long   secondary_rc;        /* Secondary return code        */
        unsigned char   *buf_ptr;            /* pointer to buffer            */
        unsigned long   buf_size;            /* buffer size                  */
        unsigned long   total_buf_size;      /* total buffer size required   */
        unsigned short  num_entries;         /* number of entries            */
        unsigned short  total_num_entries;   /* total number of entries      */
        unsigned char   list_options;        /* listing options              */
        unsigned char   reserv3;             /* reserved                     */
        unsigned char   lu_name[8];          /* LU name                      */
        unsigned char   lu_alias[8];         /* LU alias                     */
        unsigned char   tp_name[64];         /* TP name                      */
} QUERY_TP;
typedef struct tp_data
{
        unsigned short  overlay_size;        /* size of this entry           */
        unsigned char   tp_name[64];         /* TP name                      */
        unsigned char   description[RD_LEN]; /* resource description         */
        unsigned short  instance_limit;      /* max instance count           */
        unsigned short  instance_count;      /* current instance count       */
        unsigned short  locally_started_count;
                                             /* locally started instance     */
                                             /* count                        */
        unsigned short  remotely_started_count;
```

```
                                        /* remotely started instance     */
                                        /* count                         */
        unsigned char   reserva[20];    /* reserved                      */
} TP_DATA;
typedef struct tp_spec_data
{
   unsigned char pathname[256];         /* path and TP name              */
   unsigned char parameters[64];        /* parameters for TP             */
   unsigned char queued;                /* queued TP (AP_YES)            */
   unsigned char load_type;             /* type of load-DETACHED/CONSOLE */
   unsigned char dynamic_load;          /* dynamic loading of TP enabled */
   unsigned char reserved[5];           /* max size is 120 bytes         */
} TP_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_TP

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。項目数がこの値より大きくなることはありません。値を 0 に指定すると、制限はなくなります。

list_options

これは、リスト情報になにを戻すべきかを示します。指定された**lu_name**（または、**lu_name** がすべてゼロに設定されている場合、**lu_alias**）と **tp_name**（以下のパラメーターを参照してください）の組合せは、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

lu_name

LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロに設定すると、**lu_alias** フィールドが索引の判別に使用されます。

lu_alias

ローカルで定義されている LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。 **lu_name** と **lu_alias** の両方がすべてゼロに設定されている場合、制御点と関連した LU（省略時 LU）が使用されます。

tp_name

トランザクション・プログラム名。 これは、64 バイト・ストリングで、右側がスペースで埋め込まれています。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

buf_size

バッファーに戻される情報の長さ

total_buf_size

戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries

実際に戻される項目の数

total_num_entries

戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

tp_data.overlay_size

この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

tp_data.tp_name

トランザクション・プログラム名。これは、64 バイト・ストリングで、右側がスペースで埋め込まれています。

tp_data.instance.description

資源の記述 (DEFINE_TP で指定されたものです) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

tp_data.instance_limit
指定されたトランザクション・プログラムの並行活動状態インスタンスの最大数

tp_data.instance_count
現在活動状態の、指定されたトランザクション・プログラムのインスタンスの数

tp_data.locally_started_count
ローカルで開始された（TP_STARTED verb を発行しているトランザクション・プログラムによって）、指定されたトランザクション・プログラムのインスタンスの数

tp_data.remotely_started_count
リモートで開始された（受信された Attach 要求によって）、指定されたトランザクション・プログラムのインスタンスの数

tp_chars.tp_data.pathname
パスおよびトランザクション・プログラム名を指定します。

tp_chars.tp_data.parameters
トランザクション・プログラムに対するパラメーターを指定します。

tp_chars.tp_data.queued
トランザクション・プログラムを待ち行列に入れるかどうかを指定します。

tp_chars.tp_data.load_type
トランザクション・プログラムのロードの仕方を指定します。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_TP_NAME

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS
AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

QUERY_TP_DEFINITION

QUERY_TP_DEFINITION は、以前に DEFINE_TP verb で渡されていた情報と、Communications Server によって定義されたトランザクション・プログラムに関する情報の両方を戻します。

この情報は、要約情報または詳細情報という 2 つのうちいずれか 1 つの形式のリストで戻されます。 特定のトランザクション・プログラムについての情報を入手するため、またはリスト情報をいくつかの 『かたまり』 に分けて入手するためには、**tp_name** フィールドを設定しておかなければなりません。

それ以外の場合 (**list_options** フィールドが AP_FIRST_IN_LIST に設定されている場合)、このフィールドは無視されます。 リスト形式の使用方法の背景については、14 ページの『ノードの照会』 を参照してください。

このリストは、EBCDIC 辞書配列順序を使用して、**tp_name** によって配列されます。AP_LIST_FROM_NEXT を選択すると、指定された項目が存在するかどうかにかかわらず、戻されるリストは定義された配列順序での次の項目から開始します。

この verb は、定義情報のみを戻します。 QUERY_TP verb は、ローカル LU で使用するためにトランザクション・プログラムを開始して以降、判別された情報を戻します。

VCB 構造

```
typedef struct query_tp_definition
{
        unsigned short  opcode;              /* Verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* Primary return code           */
        unsigned long   secondary_rc;        /* Secondary return code         */
        unsigned char   *buf_ptr;            /* pointer to buffer             */
        unsigned long   buf_size;            /* buffer size                   */
        unsigned long   total_buf_size;      /* total buffer size required    */
        unsigned short  num_entries;         /* number of entries             */
        unsigned short  total_num_entries;   /* total number of entries       */
        unsigned char   list_options;        /* listing options               */
        unsigned char   reserv3;             /* reserved                      */
        unsigned char   tp_name[64];         /* TP name                       */
} QUERY_TP_DEFINITION;

typedef struct tp_def_summary
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   tp_name[64];         /* TP name                       */
        unsigned char   description[RD_LEN]; /* resource description          */
} TP_DEF_SUMMARY;

typedef struct tp_def_detail
{
        unsigned short  overlay_size;        /* size of this entry            */
        unsigned char   tp_name[64];         /* TP name                       */
        TP_CHARS        tp_chars;            /* TP characteristics            */
} TP_DEF_DETAIL;
```

```
typedef struct tp_chars
{
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   conv_type;           /* conversation type            */
        unsigned char   security_rqd;        /* security support             */
        unsigned char   sync_level;          /* synchronization level support */
        unsigned char   dynamic_load;        /* dynamic load                 */
        unsigned char   enabled;             /* is the TP enabled?           */
        unsigned char   pip_allowed;         /* program initialization       */
                                             /* parameters supported         */
        unsigned char   duplex_support;      /* duplex supported             */
        unsigned char   reserv3[9];          /* reserved                     */
        unsigned short  tp__instance_limit;  /* limit on currently active TP  */
                                             /* instances                    */
        unsigned short  incoming_alloc_timeout;
                                             /* incoming allocation timeout   */
        unsigned short  rcv_alloc_timeout;   /* receive allocation timeout    */
        unsigned short  tp_data_len;         /* TP data length               */
        TP_SPEC_DATA    tp_data;             /* TP data                      */
} TP_CHARS;

typedef struct tp_spec_data
{
        unsigned char pathname[256];         /* path and TP name             */
        unsigned char parameters[64];        /* parameters for TP            */
        unsigned char queued;                /* queued TP (AP_YES)           */
        unsigned char load_type;             /* type of load-DETACHED/CONSOLE */
        unsigned char dynamic_load;          /* dynamic loading of TP enabled */
        unsigned char reserved[5];           /* max size is 120 bytes        */
} TP_SPEC_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_TP_DEFINITION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。 アプリケーションは、VCB の最後にデータを追加することができます。その場合、**buf_ptr** は NULL に設定しておかなければなりません。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。

AP_SUMMARY
要約情報のみを戻します。

AP_DETAIL
詳細情報を戻します。

指定された **tp_name**（下記のパラメーターを参照してください）は、戻される実情報の開始点を指定するために使用される索引値を表します。

AP_FIRST_IN_LIST
索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT
戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE
戻されるリストは、索引値によって指定された項目から開始します。

tp_name
定義済みトランザクション・プログラムの名前。 これは、64 バイト・ストリングで、右側がスペースで埋め込まれています。 **list_options** が AP_FIRST_IN_LIST に設定される場合、このフィールドは無視されます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 これは、**buf_size** より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。これは、**num_entries** より大きくなることがあります。

tp_def_summary.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

tp_def_summary.tp_name
定義済みトランザクション・プログラム名。これは、64 バイト・ストリングで、右側がスペースで埋め込まれています。

tp_def_summary.description
資源の記述 (DEFINE_TP で指定されたものです) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

tp_def_detail.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

tp_def_detail.tp_name
定義済みトランザクション・プログラム名。これは、64 バイト・ストリングで、右側がスペースで埋め込まれています。

tp_def_detail.tp_chars.description
資源の記述 (DEFINE_TP で指定されたものです) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

tp_def_detail.tp_chars.conv_type
トランザクション・プログラムによってサポートされている会話のタイプを指定します。

AP_BASIC
AP_MAPPED
AP_EITHER

tp_def_detail.tp_chars.security_rqd
トランザクション・プログラムを開始するのに、会話機密保護情報を必要とするかどうか（AP_NONE、 AP_SAME、または AP_PGM）を指定します。

tp_def_detail.tp_chars.sync_level
トランザクション・プログラムによってサポートされている同期化レベルを指定します。

AP_NONE
トランザクション・プログラムは、同期化レベル None をサポートします。

AP_CONFIRM_SYNC_LEVEL
トランザクション・プログラムは、同期化レベル Confirm をサポートします。

AP_EITHER
トランザクション・プログラムは、同期化レベル None または Confirm をサポートします。

AP_SYNCPT_REQUIRED
トランザクション・プログラムは、同期化レベル Sync-point をサポートします。

AP_SYNCPT_NEGOTIABLE
トランザクション・プログラムは、同期化レベル None、Confirm、または Sync-point をサポートします。

tp_def_detail.tp_chars.dynamic_load
トランザクション・プログラムを動的にロードできるようにするかどうか（AP_YES または AP_NO）を指定します。

tp_def_detail.tp_chars.enabled
トランザクション・プログラムを正常に生成できるようにするかどうか（AP_YES または AP_NO）を指定します。 省略時値は AP_NO です。

tp_def_detail.tp_chars.pip_allowed
トランザクション・プログラムにプログラム初期設定パラメーター (PIP) の受信をできるようにするかどうか（AP_YES または AP_NO）を指定します。

tp_def_detail.tp_chars.duplex_support
トランザクション・プログラムを全二重か、または半二重に指定します。

AP_FULL_DUPLEX
トランザクション・プログラムを全二重に指定します。

AP_HALF_DUPLEX
トランザクション・プログラムを半二重に指定します。

AP_EITHER_DUPLEX
トランザクション・プログラムを、半二重、または全二重のいずれでもよいように指定します。

tp_def_detail.tp_chars.tp_instance_limit
並行して活動状態のトランザクション・プログラムのインスタンスの数の限界

tp_def_detail.tp_chars.incoming_alloc_timeout
RECEIVE_ALLOCATE を待つために、着信 Attach を待ち行列に入れる秒数を指定します。ゼロは、タイムアウトなしの暗黙指定で、無期限に保持されます。

tp_def_detail.tp_chars.rcv_alloc_timeout
Attach を待っている間に、RECEIVE_ALLOCATE を待ち行列に入れる秒数を指定します。ゼロは、タイムアウトなしの暗黙指定で、無期限に保持されます。

tp_def_detail.tp_chars.tp_data_len
実行依存性トランザクション・プログラムのデータ長

tp_def_detail.tp_chars.tp_data
DYNAMIC_LOAD_INDICATION 上で、未変更のまま渡された実行依存性トランザクション・プログラムのデータ

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_TP_NAME

AP_INVALID_LIST_OPTION

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

第7章 セッション限度 Verb

この章では、セッション限度の初期設定、変更、またはリセットに使用される verb を説明します。

CHANGE_SESSION_LIMIT

CHANGE_SESSION_LIMIT verb は、特定のモード（またはセッション・グループ）のセッション限度を変更するように要求します。 セッションは、この verb の処理の結果として、活動化または非活動することができます。

VCB 構造

```
typedef struct change_session_limit
{
        unsigned short  opcode;              /* verb operation code        */
        unsigned char   reserv2;             /* reserved                   */
        unsigned char   format;              /* format                     */
        unsigned short  primary_rc;          /* primary return code        */
        unsigned long   secondary_rc;        /* secondary return code      */
        unsigned char   lu_name[8];          /* local LU name              */
        unsigned char   lu_alias[8];         /* local LU alias             */
        unsigned char   plu_alias[8];        /* partner LU alias           */
        unsigned char   fqplu_name[17];      /* fully qualified partner    */
                                             /* LU name                    */
        unsigned char   reserv3;             /* reserved                   */
        unsigned char   mode_name[8];        /* mode name                  */
        unsigned char   reserv3a;            /* reserved                   */
        unsigned char   set_negotiable;      /* set max negotiable limit?  */
        unsigned short  plu_mode_session_limit;
                                             /* session limit              */
        unsigned short  min_conwinners_source;
                                             /* min source contention      */
                                             /* winner sessions            */
        unsigned short  min_conwinners_target;
                                             /* min target contention      */
                                             /* winner sessions            */
        unsigned short  auto_act;            /* auto activation limit      */
        unsigned char   responsible;         /* responsible indicator      */
        unsigned char   reserv4[3];          /* reserved                   */
        unsigned long   sense_data;          /* sense data                 */
} CHANGE_SESSION_LIMIT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_CHANGE_SESSION_LIMIT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

セッション限度の変更を要求するローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

セッション限度の変更を要求するローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** フィールドと **lu_alias** フィールドの両方がすべてゼロに設定されている場合、verb は制御点に関連した LU （省略時値 LU）に転送されます。

plu_alias

ローカル LU に認識されているパートナー LU の別名。この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。

fqplu_name

パートナー LU の完全修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

mode_name

構成の際に定義される一連のネットワーキングの特性の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

set_negotiable

plu_mode_session_limit になるように、このモードに対する最大交渉可能セッション限度を修正すべきかどうかを指定します。

AP_YES
AP_NO

plu_mode_session_limit

このモードに対して要求された合計セッション限度。(パートナー LU と交渉することのできる) 実際のセッション限度は、このモード上でローカル LU とパートナー LU との間でサポートされる合意最大セッション数です。

min_conwinners_source

このモードにおいてローカル LU が競合勝者である、セッションの最小数。

min_conwinners_target

このモードにおいてパートナー LU が競合勝者である、セッションの最小数。

auto_act

セッション限度が変更された後で自動的に活動化するセッション数。 自動的に活動化されるセッションの実際の数は、この値の最小数であり、ローカ

ル LU に対する競合勝者のセッションの交渉済みの最小数です。 この限度以下で、セッションが正常に非活動化されている（AP_DEACT_NORMALを示している）とき、新規セッションがこの限度まで活動化されます。

responsible
セッション限度が変更された後で、ソース（ローカル）またはターゲット（パートナー）LU がセッションの非活動状態に対して責任をもつかどうかを示します (AP_SOURCE または AP_TARGET)。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

secondary_rc
AP_AS_SPECIFIED

AP_AS_NEGOTIATED

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_EXCEEDS_MAX_ALLOWED

AP_INVALID_MODE_NAME
AP_INVALID_PLU_NAME
AP_INVALID_RESPONSIBLE
AP_INVALID_SET_NEGOTIABLE
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_MODE_RESET

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

割振りエラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します

primary_rc
AP_ALLOCATION_ERROR

secondary_rc
AP_ALLOCATION_FAILURE_NO_RETRY

sense_data
割振りエラーに関連したセンス・データ。

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

エラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_CONV_FAILURE_NO_RETRY

AP_CNOS_PARTNER_LU_REJECT

secondary_rc
AP_CNOS_COMMAND_RACE_REJECT

AP_CNOS_MODE_NAME_REJECT

INITIALIZE_SESSION_LIMIT

INITIALIZE_SESSION_LIMIT verb はモード・セッション限度を初期設定します。

VCB 構造

```
typedef struct initialize_session_limit
{
        unsigned short  opcode;             /* verb operation code        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* primary return code        */
        unsigned long   secondary_rc;       /* secondary return code      */
        unsigned char   lu_name[8];         /* local LU name              */
        unsigned char   lu_alias[8];        /* local LU alias             */
        unsigned char   plu_alias[8];       /* partner                    */
        unsigned char   fqplu_name[17];     /* fully qualified partner    */
                                            /* LU name                    */
        unsigned char   reserv3;            /* reserved                   */
        unsigned char   mode_name[8];       /* mode name                  */
        unsigned char   reserv3a;           /* reserved                   */
        unsigned char   set_negotiable;     /* set max negotiable limit?  */
        unsigned short  plu_mode_session_limit;
                                            /* session limit              */
        unsigned short  min_conwinners_source;
                                            /* min source contention      */
                                            /* winner sessions            */
        unsigned short  min_conwinners_target;
                                            /* min target contention      */
                                            /* winner sessions            */
        unsigned short  auto_act;           /* auto activation limit      */
        unsigned char    reserv4[4];        /* reserved                   */
        unsigned long   sense_data;         /* sense data                 */
} INITIALIZE_SESSION_LIMIT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_INITIALIZE_SESSION_LIMIT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

セッション限度の初期設定を要求するローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

セッション限度の初期設定を要求するローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、

lu_name フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。**lu_name** と **lu_alias** の両方がすべてゼロに設定されている場合、verb は制御点に関連した LU（省略時値 LU）に転送されます。

plu_alias

ローカル LU に認識されているパートナー LU の別名。この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。

fqplu_name

パートナー LU の完全修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

mode_name

構成の際に定義される一連のネットワーキングの特性の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

set_negotiable

plu_mode_session_limit になるように、このモードに対する最大交渉可能セッション限度を修正すべきかどうかを指定します。

AP_YES
AP_NO

plu_mode_session_limit

このモードに対して要求された合計セッション限度。(パートナー LU と交渉することのできる) 実際のセッション限度は、このモード上でローカル LU とパートナー LU との間でサポートされる合意最大セッション数です。この値が設定される範囲は、1 から 32 767 です。

min_conwinners_source

このモードにおいてローカル LU が競合勝者である、セッションの最小数。この値が設定される範囲は、0 から 32 767 です。

min_conwinners_target

このモードにおいてパートナー LU が競合勝者である、セッションの最小数。 この値が設定される範囲は、0 から 32 767 です。

auto_act

セッション限度が変更された後で自動的に活動化するセッション数。 自動的に活動化されるセッションの実際の数は、この値の最小数であり、ローカル LU に対する競合勝者のセッションの交渉済みの最小数です。 この限度以下で、セッションが正常に非活動化されている（AP_DEACT_NORMALを

示している）とき、新規セッションがこの限度まで活動化されます。 この値が設定される範囲は、0 から 32 767 です。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

secondary_rc
AP_AS_SPECIFIED

AP_AS_NEGOTIATED

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_EXCEEDS_MAX_ALLOWED

AP_INVALID_SET_NEGOTIABLE
AP_INVALID_PLU_NAME
AP_INVALID_MODE_NAME
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_MODE_NOT_RESET

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

割振りエラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します

primary_rc

AP_ALLOCATION_ERROR

secondary_rc

AP_ALLOCATION_FAILURE_NO_RETRY

sense_data

割振りエラーに関連したセンス・データ。

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

エラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_CONV_FAILURE_NO_RETRY

AP_CNOS_PARTNER_LU_REJECT

secondary_rc

AP_CNOS_COMMAND_RACE_REJECT

AP_CNOS_MODE_NAME_REJECT

RESET_SESSION_LIMIT

RESET_SESSION_LIMIT verb はモード・セッション限度をリセットするように要求します。

VCB 構造

```
typedef struct reset_session_limit
{
        unsigned short  opcode;           /* verb operation code        */
        unsigned char   reserv2;          /* reserved                   */
        unsigned char   format;           /* format                     */
        unsigned short  primary_rc;       /* primary return code        */
        unsigned long   secondary_rc;     /* secondary return code      */
        unsigned char   lu_name[8];       /* local LU name              */
        unsigned char   lu_alias[8];      /* local LU alias             */
        unsigned char   plu_alias[8];     /* partner LU alias           */
        unsigned char   fqplu_name[17];   /* fully qual partner LU name */
        unsigned char   reserv3;          /* reserved                   */
        unsigned char   mode_name[8];     /* mode name                  */
        unsigned char   mode_name_select; /* select mode name           */
        unsigned char   set_negotiable;   /* set max negotiable limit?  */
        unsigned char   reserv4[8];       /* reserved                   */
        unsigned char   responsible;      /* responsible                */
        unsigned char   drain_source;     /* drain source               */
        unsigned char   drain_target;     /* drain target               */
        unsigned char   force;            /* force                      */
        unsigned long   sense_data;       /* sense data                 */
} RESET_SESSION_LIMIT;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_RESET_SESSION_LIMIT

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

セッション限度のリセットを要求するローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

セッション限度のリセットを要求するローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。この別名がすべてゼロに設定されている場合、verb は制御点に関連した LU に転送されます（省略時 LU ）。

plu_alias

ローカル LU に認識されているパートナー LU の別名。この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。このフィールドがすべてゼロに設定されている場合、**fqplu_name** フィールドが、必要なパートナー LU を指定するために使用されます。

fqplu_name

パートナー LU の完全修飾名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

mode_name

構成の際に定義される一連のネットワーキングの特性の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

mode_name_select

セッション限度を単一指定モードでリセットするか、あるいはローカル LU とパートナー LU 間のすべてのモードでリセットするかを選択します。

AP_ONEAP_ALL

set_negotiable

このモードに対する最大交渉可能セッション限度を修正すべきかどうかを指定します。

AP_YES
AP_NO

responsible

セッション限度がリセットされた後で、ソース (ローカル) またはターゲット (パートナー) LU がセッションの非活動状態に対して責任をもつかどうかを示します (AP_SOURCE または AP_TARGET)。

drain_source

セッション限度が変更またはリセットされたときに、セッションを非活動化する前に、ソース LU が待機中のセッション要求を満たすかどうかを指定します (AP_NO または AP_YES)。

drain_target

セッション限度が変更またはリセットされたときに、セッションを非活動化する前に、ターゲット LU が待機中のセッション要求を満たすかどうかを指定します (AP_NO または AP_YES)。

force CNOS 交渉が失敗した場合でも、セッション限度をゼロに設定するかどうかを指定します (AP_YES または AP_NO)。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_OK

secondary_rc

AP_FORCED

AP_AS_SPECIFIED
AP_AS_NEGOTIATED

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_EXCEEDS_MAX_ALLOWED

AP_INVALID_PLU_NAME
AP_INVALID_MODE_NAME
AP_INVALID_MODE_NAME_SELECT
AP_INVALID_RESPONSIBLE
AP_INVALID_DRAIN_SOURCE
AP_INVALID_DRAIN_TARGET
AP_INVALID_FORCE
AP_INVALID_SET_NEGOTIABLE
AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_MODE_RESET

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

割振りエラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します

primary_rc

AP_ALLOCATION_ERROR

secondary_rc

AP_ALLOCATION_FAILURE_NO_RETRY

sense_data

割振りエラーに関連したセンス・データ。

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

エラーが原因で verb が実行されない場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_CONV_FAILURE_NO_RETRY

AP_CNOS_PARTNER_LU_REJECT

secondary_rc

AP_CNOS_COMMAND_RACE_REJECT

AP_CNOS_MODE_NAME_REJECT

第8章 ノード・オペレーター機能 API の指示

ノード・オペレーター機能 API は、ノードにおける変更をノード・オペレーターに通知するための指示 verb を生成します。 指示 verb は、以下の一般構造を使用します。

```
typedef struct indication_hdr
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   data_lost;           /* previous indication lost     */
} INDICATION_HDR;
```

DLC_INDICATION

この指示は、DLC が活動状態から非活動状態、あるいは非活動状態から活動状態に切り替わったときに生成されます。

VCB 構造

```
typedef struct dlc_indication
{
        unsigned short  opcode;               /* verb operation code           */
        unsigned char   reserv2;              /* reserved                      */
        unsigned char   format;               /* format                        */
        unsigned short  primary_rc;           /* primary return code           */
        unsigned long   secondary_rc;         /* secondary return code         */
        unsigned char   data_lost;            /* previous indication lost      */
        unsigned char   deactivated;          /* has session been deactivated? */
        unsigned char   dlc_name[8];          /* link station name             */
        unsigned char   description[RD_LEN];  /* resource description          */
        unsigned char   reserva[20];          /* reserved                      */
} DLC_INDICATION;
```

パラメーター

opcode

AP_DLC_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated

DLC が非活動状態になったときに AP_YES に設定します。 DLC が活動状態になったときに AP_NO に設定します。

dlc_name

DLC 名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

description

資源の記述 (DEFINE_DLC で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

DLUR_LU_INDICATION

この指示は、DLUR LU を活動化または非活動化したときに生成されます。この指示により、登録済みアプリケーションは現在活動状態にある DLUR LU のリストを保守することができます。

VCB 構造

```
typedef struct dlur_lu_indication
{
        unsigned short  opcode;         /* verb operation code        */
        unsigned char   reserv2;        /* reserved                   */
        unsigned char   format;         /* format                     */
        unsigned short  primary_rc;     /* primary return code        */
        unsigned long   secondary_rc;   /* secondary return code      */
        unsigned char   data_lost;      /* previous indication lost   */
        unsigned char   reason;         /* reason for this indication */
        unsigned char   lu_name[8];     /* LU name                    */
        unsigned char   pu_name[8];     /* PU name                    */
        unsigned char   nau_address;    /* NAU address                */
        unsigned char   reserv5[7];     /* reserved                   */
} DLUR_LU_INDICATION;
```

パラメーター

opcode

AP_DLUR_LU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO)。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

reason

DLUR LU が DLUS によって活動化された直後に AP_ADDED に設定します。DLUR LU が、DLUS により明示的に非活動化されたか、リンク障害または PU の非活動化により暗黙的に非活動化された場合、AP_REMOVED に設定します。

lu_name

LU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

DLUR_LU_INDICATION

pu_name

この LU が使用する PU 名。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

nau_address

LU のネットワーク・アドレス可能単位のアドレスで、1 から 255 までの範囲でなければなりません。

DLUS_INDICATION

この指示は、DLUS ノードに対するパイプが非活動状態から活動状態（またはその逆）に切り替わったときに生成されます。 パイプが非活動状態になったときに、パイプに関する統計が提供されます。

VCB 構造

```
typedef struct dlus_indication
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   data_lost;           /* previous indication lost     */
        unsigned char   deactivated;         /* has session been deactivated? */
        unsigned char   dlus_name[17];       /* DLUS name                    */
        unsigned char   reserv1;             /* reserved                     */
        PIPE_STATS      pipe_stats;          /* pipe statistics              */
        unsigned char   reserva[20];         /* reserved                     */
} DLUS_INDICATION;
typedef struct pipe_stats
{
        unsigned long   reqactpu_sent;       /* REQACTPUs sent to DLUS       */
        unsigned long   reqactpu_rsp_received;
                                             /* RSP(REQACTPU)s received      */
                                             /* from DLUS                    */
        unsigned long   actpu_received;      /* ACTPUs received from DLUS    */
        unsigned long   actpu_rsp_sent;      /* RSP(ACTPU)s sent to DLUS     */
        unsigned long   reqdactpu_sent;      /* REQDACTPUs sent to DLUS      */
        unsigned long   reqdactpu_rsp_received;
                                             /* RSP(REQDACTPU)s received     */
                                             /* from DLUS                    */
        unsigned long   dactpu_received;     /* DACTPUs received from DLUS   */
        unsigned long   dactpu_rsp_sent;     /* RSP(DACTPU)s sent to DLUS    */
        unsigned long   actlu_received;      /* ACTLUs received from DLUS    */
        unsigned long   actlu_rsp_sent;      /* RSP(ACTLU)s sent to DLUS     */
        unsigned long   dactlu_received;     /* DACTLUs received from DLUS   */
        unsigned long   dactlu_rsp_sent;     /* RSP(DACTLU)s sent to DLUS    */
        unsigned long   sscp_pu_mus_rcvd;    /* MUs for SSCP-PU sess received */
        unsigned long   sscp_pu_mus_sent;    /* MUs for SSCP-PU sessions sent */
        unsigned long   sscp_lu_mus_rcvd;    /* MUs for SSCP-LU sess received */
        unsigned long   sscp_lu_mus_sent;    /* MUs for SSCP-LU sessions sent */
} PIPE_STATS;
```

パラメーター

opcode

AP_DLUS_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc
AP_OK

secondary_rc
ゼロと同等。

data_lost
データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated
パイプが非活動状態になったときに AP_YES に設定します。パイプが活動状態になったときに AP_NO に設定します。

dlus_name
DLUS の名前。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されている 17 バイト・ストリングで、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

pipe_stats.reqactpu_sent
パイプを介して DLUS へ送信する REQACTPU の数

pipe_stats.reqactpu_rsp_received
パイプを介して DLUS から受信する RSP(REQACTPU) の数

pipe_stats.actpu_received
パイプを介して DLUS から受信する ACTPU の数

pipe_stats.actpu_rsp_sent
パイプを介して DLUS へ送信する RSP(ACTPU) の数

pipe_stats.reqdactpu_sent
パイプを介して DLUS へ送信する REQDACTPU の数

pipe_stats.reqdactpu_rsp_received
パイプを介して DLUS から受信する RSP(REQDACTPU) の数

pipe_stats.dactpu_received
パイプを介して DLUS から受信する DACTPU の数

pipe_stats.dactpu_rsp_sent
パイプを介して DLUS へ送信する RSP(DACTPU) の数

pipe_stats.actlu_received
パイプを介して DLUS から受信する ACTLU の数

pipe_stats.actlu_rsp_sent
パイプを介して DLUS へ送信する RSP(ACTLU) の数

pipe_stats.dactlu_received
パイプを介して DLUS から受信する DACTLU の数

pipe_stats.dactlu_rsp_sent
パイプを介して DLUS へ送信する RSP(DACTLU) の数

pipe_stats.sscp_pu_mus_rcvd
パイプを介して DLUS から受信する SSCP-PU MU の数

pipe_stats.sscp_pu_mus_sent
パイプを介して DLUS へ送信する SSCP-PU MU の数

pipe_stats.sscp_lu_mus_rcvd
パイプを介して DLUS から受信する SSCP-LU MU の数

pipe_stats.sscp_lu_mus_sent
パイプを介して DLUS へ送信する SSCP-LU MU の数

DOWNSTREAM_LU_INDICATION

この指示は、ダウンストリーム LU とホスト間の LU-SSCP セッションが非活動状態から活動状態（またはその逆）、あるいは、PLU-SLU セッションが非活動状態から活動状態（またはその逆）に切り替わったときに生成されます。LU-SSCP セッションが非活動化すると LU-SSCP 統計が提供され、PLU-SLU セッションが非活動化すると PLU-SLU 統計が提供されます。

VCB 構造

```
typedef struct downstream_lu_indication
{
        unsigned short  opcode;                 /* verb operation code          */
        unsigned char   reserv2;                /* reserved                     */
        unsigned char   format;                 /* format                       */
        unsigned short  primary_rc;             /* primary return code          */
        unsigned long   secondary_rc;           /* secondary return code        */
        unsigned char   data_lost;              /* previous indication lost     */
        unsigned char   dspu_name[8];           /* PU Name                      */
        unsigned char   ls_name[8];             /* Link station name            */
        unsigned char   dslu_name[8];           /* LU Name                      */
        unsigned char   description[RD_LEN];    /* resource description         */
        unsigned char   nau_address;            /* NAU address                  */
        unsigned char   lu_sscp_sess_active;
                                                /* Is SSCP session active?      */
        unsigned char   plu_sess_active;        /* Is PLU-SLU session active?   */
        unsigned char   dspu_services;          /* DSPU services                */
        unsigned char   reserv1;                /* reserved                     */
        SESSION_STATS   lu_sscp_stats;          /* LU-SSCP session statistics   */
        SESSION_STATS   ds_plu_stats;           /* Downstream PLU-SLU sess stats */
        SESSION_STATS   us_plu_stats;           /* Upstream PLU-SLU sess stats  */
} DOWNSTREAM_LU_INDICATION;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;            /* session receive RU size      */
        unsigned short  send_ru_size;           /* session send RU size         */
        unsigned short  max_send_btu_size;      /* max send BTU size            */
        unsigned short  max_rcv_btu_size;       /* max rcv BTU size             */
        unsigned short  max_send_pac_win;       /* max send pacing window size  */
        unsigned short  cur_send_pac_win;       /* curr send pacing window size */
        unsigned short  max_rcv_pac_win;        /* max rcv pacing window size   */
        unsigned short  cur_rcv_pac_win;        /* curr receive pacing win size */
        unsigned long   send_data_frames;       /* number of data frames sent   */
        unsigned long   send_fmd_data_frames;
                                                /* num FMD data frames sent     */
        unsigned long   send_data_bytes;        /* number of data bytes sent    */
        unsigned long   rcv_data_frames;        /* num of data frames received  */
        unsigned long   rcv_fmd_data_frames;
                                                /* num FMD data frames received */
        unsigned long   rcv_data_bytes;         /* num data bytes received      */
        unsigned char   sidh;                   /* session ID high byte         */
        unsigned char   sidl;                   /* session ID low byte          */
```

```
        unsigned char   odai;                   /* ODAI bit set                 */
        unsigned char   ls_name[8];             /* Link station name            */
        unsigned char   reserve;                /* reserved                     */
} SESSION_STATS;
```

パラメーター

opcode

AP_DOWNSTREAM_LU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

dspu_name

ダウンストリーム LU に関連したダウンストリーム PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

ls_name

リンク・ステーションの名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。

dslu_name

ダウンストリーム LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description

資源記述（DEFINE_DOWNSTREAM_LU で指定されているものと同じ）。

nau_address

LU のネットワーク・アドレス可能単位のアドレスであり、1 から 255 の範囲内でなければなりません。

lu_sscp_sess_active

ダウンストリーム LU との LU-SSCP セッションが活動状態であるかどうかを示します。 AP_YES または AP_NO のいずれかに設定します。

plu_sess_active

ダウンストリーム LU との PLU-SLU セッションが活動状態であるかどうかを示します。 AP_YES または AP_NO のいずれかに設定します。

DOWNSTREAM_LU_INDICATION

dspu_services
ローカル・ノードがリンクを介してダウンストリーム LU に提供するサービスを指定します。 このパラメーターには、次のいずれかを設定します。

AP_PU_CONCENTRATION
ローカル・ノードは、ダウンストリーム PU に対して PU 集信を提供します。

AP_DLUR
ローカル・ノードは、ダウンストリーム PU に対して DLUR サポートを提供します。

lu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

lu_sscp_stats.sidh
セッション ID の高位バイト

lu_sscp_stats.sidl
セッション ID の低位バイト

lu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

ds_plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

ds_plu_stats.send_ru_size
送信のときの最大 RU サイズ

ds_plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

ds_plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

ds_plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

ds_plu_stats.cur_send_pac_win
このセッションにおける送信ペーシング・ウィンドウの現行サイズ。

ds_plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

ds_plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

ds_plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

ds_plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

ds_plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

ds_plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

ds_plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

ds_plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

DOWNSTREAM_LU_INDICATION

ds_plu_stats.sidh
セッション ID の高位バイト

ds_plu_stats.sidl
セッション ID の低位バイト

ds_plu_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

ds_plu_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

us_plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

us_plu_stats.send_ru_size
送信のときの最大 RU サイズ

us_plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

us_plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

us_plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

us_plu_stats.cur_send_pac_win
このセッションにおける送信ペーシング・ウィンドウの現行サイズ。

us_plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

us_plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

us_plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

us_plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

us_plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

us_plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

us_plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

us_plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

us_plu_stats.sidh
セッション ID の高位バイト**dspu_services** を AP_PU_CONCENTRATION に設定した場合、このフィールドは予約済みです。

us_plu_stats.sidl
セッション ID の低位バイト **dspu_services** を AP_PU_CONCENTRATION に設定した場合、このフィールドは予約済みです。

us_plu_stats.odai
起点宛先アドレス標識。 セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。 **dspu_services** を AP_PU_CONCENTRATION に設定した場合、このフィールドは予約済みです。

us_plu_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 **dspu_services** を AP_PU_CONCENTRATION に設定した場合、このフィールドは予約済みです。

DOWNSTREAM_PU_INDICATION

この指示は、ダウンストリーム PU とホスト間の PU-SSCP セッションが非活動状態から活動状態（またはその逆）に切り替わったときに生成されます。 PU-SSCP セッションが非活動化すると、PU-SSCP 統計が提供されます。

VCB 構造

```
typedef struct downstream_pu_indication
{
        unsigned short  opcode;               /* verb operation code          */
        unsigned char   reserv2;              /* reserved                     */
        unsigned char   format;               /* format                       */
        unsigned short  primary_rc;           /* primary return code          */
        unsigned long   secondary_rc;         /* secondary return code        */
        unsigned char   data_lost;            /* previous indication lost     */
        unsigned char   dspu_name[8];         /* PU Name                      */
        unsigned char   description[RD_LEN];  /* resource description         */
        unsigned char   ls_name[8];           /* Link Station name            */
        unsigned char   pu_sscp_sess_active;
                                              /* Is PU-SSCP session active?   */
        unsigned char   dspu_services;        /* DSPU services                */
        unsigned char   reserv1[2];           /* reserved                     */
        SESSION_STATS   pu_sscp_stats;        /* PU-SSCP session statistics   */
} DOWNSTREAM_PU_INDICATION;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;          /* session receive RU size      */
        unsigned short  send_ru_size;         /* session send RU size         */
        unsigned short  max_send_btu_size;    /* max send BTU size            */
        unsigned short  max_rcv_btu_size;     /* max rcv BTU size             */
        unsigned short  max_send_pac_win;     /* max send pacing window size  */
        unsigned short  cur_send_pac_win;     /* curr send pacing window size */
        unsigned short  max_rcv_pac_win;      /* max rcv pacing window size   */
        unsigned short  cur_rcv_pac_win;      /* curr receive pacing win size */
        unsigned long   send_data_frames;     /* number of data frames sent   */
        unsigned long   send_fmd_data_frames;
                                              /* num FMD data frames sent     */
        unsigned long   send_data_bytes;      /* number of data bytes sent    */
        unsigned long   rcv_data_frames;      /* num of data frames received  */
        unsigned long   rcv_fmd_data_frames;
                                              /* num FMD data frames received */
        unsigned long   rcv_data_bytes;       /* num data bytes received      */
        unsigned char   sidh;                 /* session ID high byte         */
        unsigned char   sidl;                 /* session ID low byte          */
        unsigned char   odai;                 /* ODAI bit set                 */
        unsigned char   ls_name[8];           /* Link station name           */
        unsigned char   reserve;              /* reserved                     */
} SESSION_STATS;
```

パラメーター

opcode

AP_DOWNSTREAM_PU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc
AP_OK

secondary_rc
ゼロと同等。

data_lost
データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

dspu_name
ダウンストリーム PU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description
資源記述（DEFINE_LS で指定されているものと同じ）。

ls_name
リンク・ステーションの名前。これは、ローカルで表示可能な文字セットの 8 バイト・ストリングです。8 バイトすべてが有効です。

pu_sscp_sess_active
ダウンストリーム PU との PU-SSCP セッションが活動状態であるかどうかを示します。 AP_YES または AP_NO のいずれかに設定します。

dspu_services
ローカル・ノードがリンクを通じてダウンストリーム PU に提供するサービスを指定します。 このパラメーターには、次のいずれかを設定します。

AP_PU_CONCENTRATION
ローカル・ノードは、ダウンストリーム PU に対して PU 集信を提供します。

AP_DLUR
ローカル・ノードは、ダウンストリーム PU に対して DLUR サポートを提供します。

pu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

pu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

pu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

pu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

DOWNSTREAM_PU_INDICATION

pu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

pu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

pu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

pu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

pu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

pu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

pu_sscp_stats.sidh
セッション ID の高位バイト

pu_sscp_stats.sidl
セッション ID の低位バイト

pu_sscp_stats.odai
起点宛先アドレス標識。 セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

pu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

FOCAL_POINT_INDICATION

この指示は、フォーカル・ポイントが獲得、変更あるいは取り消されたときに生成されます。

VCB 構造

```
typedef struct focal_point_indication
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   data_lost;          /* previous indication lost     */
        unsigned char   ms_category[8];     /* Focal point category         */
        unsigned char   fp_fqcp_name[17];   /* Fully qualified focal        */
                                            /* point CP name                */
        unsigned char   ms_appl_name[8];    /* Focal point application name */
        unsigned char   fp_type;            /* type of current focal point  */
        unsigned char   fp_status;          /* status of focal point        */
        unsigned char   fp_routing;         /* type of MDS routing to       */
                                            /* reach FP                     */
        unsigned char   reserva[20];        /* reserved                     */
} FOCAL_POINT_INDICATION;
```

パラメーター

opcode

AP_FOCAL_POINT_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

ms_category

フォーカル・ポイントが獲得、変更あるいは取り消された、フォーカル・ポイントのカテゴリー。これは、*SNA Management Services* で説明している、管理サービス・カテゴリーの体系的に定義された 4 バイトの値（右側は EBCDIC スペースで埋められる）、あるいは 8 バイトのタイプ 1134 の EBCDIC 導入定義名のいずれかを指定することができます。

FOCAL_POINT_INDICATION

fp_fqcp_name

現行のフォーカル・ポイントの完全修飾制御点名です。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）フォーカル・ポイントが取り消され、その後、置き換えられない（その結果、現行では活動状態のフォーカル・ポイントが存在しない）場合は、この名前はすべてゼロで指定します。

ms_appl_name

現行のフォーカル・ポイント・アプリケーションの名前。これは、*SNA Management Services* で説明している、管理サービス・アプリケーションの体系的に定義された 4 バイトの値（右側は EBCDIC スペースで埋められる）、あるいは 8 バイトのタイプ 1134 の EBCDIC 導入定義名のいずれかを指定することができます。フォーカル・ポイントが取り消され、その後、置き換えられない（その結果、現行では活動状態のフォーカル・ポイントが存在しない）場合は、この名前はすべてゼロで指定します。

fp_type

フォーカル・ポイントのタイプ。詳細については、*SNA Management Services* を参照してください。

AP_EXPLICIT_PRIMARY_FP
AP_BACKUP_FP
AP_DEFAULT_PRIMARY_FP
AP_DOMAIN_FP
AP_HOST_FP
AP_NO_FP

fp_status

フォーカル・ポイントの状況。

AP_NOT_ACTIVE

フォーカル・ポイントが活動状態から非活動状態に切り替わった。

AP_ACTIVE

フォーカル・ポイントが非活動状態または活動保留状態から活動状態に切り替わった。

fp_routing

フォーカル・ポイントにデータを送信するために MDS トランスポートを使用する際に、アプリケーションが指定する必要がある経路指定のタイプ（フォーカル・ポイントの状況が AP_ACTIVE である場合のみ有効です）。

AP_DEFAULT

省略時の経路指定を使用して MDS_MU をフォーカル・ポイントに送達します。

AP_DIRECT

MDS_MU が直接フォーカル・ポイントに向かうセッションに経路指定されます。

ISR_INDICATION

この指示は、ISR セッションを活動化または非活動化するときに生成されます。このセッションを非活動化する場合、最終セッション統計が戻されます。このセッションを活動化する場合、**pri_sess_stats** フィールドおよび **sec_sess_stats** フィールドは予約済みです。

VCB 構造

```
typedef struct isr_indication
{
        unsigned short  opcode;            /* verb operation code          */
        unsigned char   reserv2;           /* reserved                     */
        unsigned char   format;            /* format                       */
        unsigned short  primary_rc;        /* primary return code          */
        unsigned long   secondary_rc;      /* secondary return code        */
        unsigned char   data_lost;         /* previous indication lost     */
        unsigned char   deactivated;       /* has ISR session been         */
                                           /* deactivated?                 */
        FQPCID          fqpcid;            /* fully qualified procedure    */
                                           /* correlator ID                */
        unsigned char   fqplu_name[17];    /* fully qualified primary      */
                                           /* LU name                      */
        unsigned char   fqslu_name[17];    /* fully qualified secondary    */
                                           /* LU name                      */
        unsigned char   mode_name[8];      /* mode name                    */
        unsigned char   cos_name[8];       /* COS name                     */
        unsigned char   transmission_priority;
                                           /* transmission priority        */
        unsigned long   sense_data;        /* sense data                   */
        unsigned char   reserv2a[2];       /* reserved                     */
        SESSION_STATS   pri_sess_stats;    /* primary hop session stats    */
        SESSION_STATS   sec_sess_stats;    /* secondary hop session        */
                                           /* statistics                   */
        unsigned char   reserva[20];       /* reserved                     */
} ISR_INDICATION;

typedef struct fqpcid
{
        unsigned char   pcid[8];           /* pro correlator identifier    */
        unsigned char   fqcp_name[17];     /* orig's network qualified     */
                                           /* CP name                      */
        unsigned char   reserve3[3];       /* reserved                     */
} FQPCID;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;       /* session receive RU size      */
        unsigned short  send_ru_size;      /* session send RU size         */
        unsigned short  max_send_btu_size; /* Maximum send BTU size        */
        unsigned short  max_rcv_btu_size;  /* Maximum rcv BTU size         */
        unsigned short  max_send_pac_win;  /* Max send pacing window size  */
        unsigned short  cur_send_pac_win;  /* Curr send pacing window size */
        unsigned short  max_rcv_pac_win;   /* Max receive pacing win size  */
        unsigned short  cur_rcv_pac_win;   /* Curr rec pacing window size  */
        unsigned long   send_data_frames;  /* Number of data frames sent   */
```

```
        unsigned long   send_fmd_data_frames;
                                        /* num of FMD data frames sent  */
        unsigned long   send_data_bytes;   /* Number of data bytes sent    */
        unsigned long   rcv_data_frames;   /* Num data frames received     */
        unsigned long   rcv_fmd_data_frames;
                                        /* num of FMD data frames recvd */
        unsigned long   rcv_data_bytes;    /* Num data bytes received      */
        unsigned char   sidh;              /* Session ID high byte         */
        unsigned char   sidl;              /* Session ID low byte          */
        unsigned char   odai;              /* ODAI bit set                 */
        unsigned char   ls_name[8];        /* Link station name            */
        unsigned char   reserve;           /* reserved                     */
} SESSION_STATS;
```

パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_ISR_INDICATION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定した場合、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY verb を発行して、脱落した情報を更新する必要があります。

deactivate

ISR セッションを非活動化するときに、AP_YES に設定します。ISR セッションを活動化するときに、AP_NO に設定します。

fqpcid.pcid

処理手順相関識別子。 これは 8 バイトの 16 進数ストリングです。

fqpcid.pcid_name

完全修飾制御点名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

fqplu_name

完全修飾 1 次 LU 名（BIND 要求で指定されているものと同じ)。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成

されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) **deactivated** に AP_YES が指定されている場合、この名前はすべてゼロになります。

fqslu_name
完全修飾 2次 LU 名 (BIND 要求で指定されているものと同じ)。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。) **deactivated** に AP_YES が指定されている場合、この名前はすべてゼロになります。

cos_name
サービス・クラス名。 8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。**deactivated** に AP_YES が指定されている場合、この名前はすべてゼロになります。

transmission_priority
セッションに関連した伝送優先順位。**deactivated** に AP_YES が指定されている場合、このフィールドは予約済みです。

sense_data
UNBIND 要求で送信または受信されるセンス・データ。**deactivated** に AP_YES が指定されている場合、このフィールドは予約済みです。

pri_sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

pri_sess_stats.send_ru_size
送信のときの最大 RU サイズ

pri_sess_stats.max_send_btu_size
送信できる最大 BTU サイズ

pri_sess_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

pri_sess_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

pri_sess_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

pri_sess_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

pri_sess_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

pri_sess_stats.send_data_frames
送信される通常フロー・データ・フレームの数

pri_sess_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

pri_sess_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

pri_sess_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

pri_sess_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

pri_sess_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

pri_sess_stats.sidh
セッション ID の高位バイト

pri_sess_stats.sidl
セッション ID の低位バイト

pri_sess_stats.odai
起点宛先アドレス標識。セッションを立ち上げるとき、ローカル・ノードが 1 次リンク・ステーションを含む場合、BIND の送信側は、このフィールドをゼロに設定します。BIND の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールド 1 に設定します。

pri_sess_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドを使用して、セッション統計とセッション・トラフィックが流れるリンクを相関させることができます。

sec_sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

sec_sess_stats.send_ru_size
送信のときの最大 RU サイズ

sec_sess_stats.max_send_btu_size
送信できる最大 BTU サイズ

sec_sess_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

sec_sess_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

sec_sess_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

sec_sess_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

sec_sess_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

sec_sess_stats.send_data_frames
送信される通常フロー・データ・フレームの数

sec_sess_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

sec_sess_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

sec_sess_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

sec_sess_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

sec_sess_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

sec_sess_stats.sidh
セッション ID の高位バイト

sec_sess_stats.sidl
セッション ID の低位バイト

sec_sess_stats.odai
起点宛先アドレス標識。 セッションを立ち上げるとき、ローカル・ノードが 1 次リンク・ステーションを含む場合、BIND の送信側は、このフィールドをゼロに設定します。 BIND の送信側が 2 次リンク・ステーションを含むノードである場合、このフィールド 1 に設定します。

sec_sess_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 このフィールドを使用して、セッション統計とセッション・トラフィックが流れるリンクを相関させることができます。

LOCAL_LU_INDICATION

この指示は、LOCAL LU が定義または削除されるときに常に生成されます。 この指示により、登録済みアプリケーションは現在定義されているすべての LU のリストを保守することができます。

VCB 構造

```
typedef struct local_lu_indication
{
        unsigned short  opcode;              /* verb operation code          */
        unsigned char   reserv2;             /* reserved                     */
        unsigned char   format;              /* format                       */
        unsigned short  primary_rc;          /* primary return code          */
        unsigned long   secondary_rc;        /* secondary return code        */
        unsigned char   data_lost;           /* previous indication lost     */
        unsigned char   reason;              /* reason for this indication   */
        unsigned char   lu_name[8];          /* LU name                      */
        unsigned char   description[RD_LEN];
                                             /* resource description         */
        unsigned char   lu_alias[8];         /* LU alias                     */
        unsigned char   nau_address;         /* NAU address                  */
        unsigned char   reserv4;             /* reserved                     */
        unsigned char   pu_name[8];          /* PU name                      */
        unsigned char   lu_sscp_active;      /* Is LU-SSCP session active    */
        unsigned char   reserv5;             /* reserved                     */
        SESSION_STATS   lu_sscp_stats;       /* LU-SSCP session statistics   */
} LOCAL_LU_INDICATION;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;         /* session receive RU size       */
        unsigned short  send_ru_size;        /* session send RU size          */
        unsigned short  max_send_btu_size;   /* max send BTU size             */
        unsigned short  max_rcv_btu_size;    /* max rcv BTU size              */
        unsigned short  max_send_pac_win;    /* max send pacing window size   */
        unsigned short  cur_send_pac_win;    /* current send pacing win size  */
        unsigned short  max_rcv_pac_win;     /* max receive pacing win size   */
        unsigned short  cur_rcv_pac_win;     /* curr receive pacing win size  */
        unsigned long   send_data_frames;    /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                             /* num of FMD data frames sent   */
        unsigned long   send_data_bytes;     /* number of data bytes sent     */
        unsigned long   rcv_data_frames;     /* num of data frames received   */
        unsigned long   rcv_fmd_data_frames;
                                             /* num FMD data frames received  */
        unsigned long   rcv_data_bytes;      /* number of data bytes received */
        unsigned char   sidh;                /* session ID high byte          */
        unsigned char   sidl;                /* session ID low byte           */
        unsigned char   odai;                /* ODAI bit set                  */
        unsigned char   ls_name[8];          /* Link station name             */
        unsigned char   reserve;             /* reserved                      */
} SESSION_STATS;
```

注: LU-SSCP 統計は、**nau_address** が非ゼロであり、LU-SSCP セッションが活動状態から非活動状態に切り替わった場合のみ有効です。それ以外の場合、フィールドは予約済みです。

パラメーター

opcode

AP_LOCAL_LU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO)。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定した場合、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

reason

指示が発行される理由。

AP_ADDED

LU が定義された。

AP_REMOVED

LU が、DELETE_LOCAL_LU を使用して明示的に削除されたか、あるいは DELETE_LS、DELETE_PORT、または DELETE_DLC を使用して暗黙的に削除された。

AP_SSCP_ACTIVE

ノードが正常に ACTLU を処理した後で、LU-SSCP セッションが活動状態になった。

AP_SSCP_INACTIVE

正常 DACTLU あるいはリンク障害の後で、LU-SSCP セッションが非活動状態になった。

lu_name

LU 名。 状態が変更されたローカル LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング (先頭は文字) で、右側は EBCDIC スペースで埋められます。

description

資源の記述 (DEFINE_LOCAL_LU で指定されたもの) 。

LOCAL_LU_INDICATION

lu_alias
ローカルで定義されている LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

nau_address
LU のネットワーク・アドレス可能単位のアドレスで、0 から 255 までの範囲でなければなりません。 非ゼロの値は、LU が従属型 LU であることを示します。ゼロの値は LU が独立 LU であることを暗黙指定します。

pu_name
この LU が使用する PU 名。これは、8 バイトの英数字で、タイプ A の EBCDIC ストリングです。このフィールドは LU が従属型 LU である場合のみ有効です（つまり、**nau_address** が非ゼロである場合)。独立型 LU である場合は、フィールドはすべて 2 進ゼロで設定します。

lu_sscp_sess_active
LU-SSCP セッションが活動状態であるかどうかを指定します (AP_YES または AP_NO) 。 **nau_address** がゼロである場合、このフィールドは予約済みです。

lu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

lu_sscp_stats.sidh
セッション ID の高位バイト

lu_sscp_stats.sidl
セッション ID の低位バイト

lu_sscp_stats.odai
起点宛先アドレス標識。 セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、ACTLU の送信側がこのフィールドをゼロに設定します。ACTLU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドは ACTLU の送信側により 1 に設定されます。

lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

LOCAL_TOPOLOGY_INDICATION

この指示は、ノードのローカル・トポロジー・データベース内の TG エントリーが活動状態から非活動状態に変更されたとき、あるいは非活動状態から活動状態に変更されたときに生成されます。

VCB 構造

```
typedef struct local_topology_indication
{
        unsigned short  opcode;             /* verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* primary return code          */
        unsigned long   secondary_rc;       /* secondary return code        */
        unsigned char   data_lost;          /* previous indication lost     */
        unsigned char   status;             /* TG status                    */
        unsigned char   dest[17];           /* name of TG destination node  */
        unsigned char   dest_type;          /* TG destination node type     */
        unsigned char   tg_num;             /* TG number                    */
        unsigned char   reserva[20];        /* reserved                     */
} LOCAL_TOPOLOGY_INDICATION;
```

パラメーター

opcode

AP_LOCAL_TOPOLOGY_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

状況（status）

TG の状況を指定します。これは、以下の値の 1 つ以上を一緒に論理和 (OR) 演算したものです。

AP_TG_OPERATIVE
AP_TG_CP_CP_SESSIONS
AP_TG_QUIESCING
AP_NONE

dest TG に対する完全修飾の宛先ノード名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

dest_type
ノードのタイプ。 以下の値のいずれか 1 つになります。

AP_END_NODE
AP_NETWORK_NODE
AP_VRN

tg_num
TG と関連した数。

LS_INDICATION

この指示は、リンクを使用する活動セッションの数、またはリンク・ステーションの外部状態が変更されたときに生成されます。 リンク・ステーションが非活動状態になったときに、リンク・ステーション統計が提供されます。

VCB 構造

```
typedef struct ls_indication
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   data_lost;           /* previous indication lost      */
        unsigned char   deactivated;         /* has session been deactivated? */
        unsigned char   ls_name[8];          /* link station name             */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   adj_cp_name[17];     /* network qualified Adj CP name */
        unsigned char   adj_node_type;       /* adjacent node type            */
        unsigned short  act_sess_count;      /* active session count on link  */
        unsigned char   indication_cause;    /* cause of indication           */
        LS_STATS        ls_stats;            /* link station statistics       */
        unsigned char   tg_num;              /* TG number                     */
        unsigned long   sense_data;          /* sense data                    */
        unsigned char   reserva[19];         /* reserved                      */
} LS_INDICATION;
typedef struct ls_stats
{
        unsigned long   in_xid_bytes;        /* num of XID bytes received     */
        unsigned long   in_msg_bytes;        /* num message bytes received    */
        unsigned long   in_xid_frames;       /* num XID frames received       */
        unsigned long   in_msg_frames;       /* num message frames received   */
        unsigned long   out_xid_bytes;       /* num XID bytes sent            */
        unsigned long   out_msg_bytes;       /* num message bytes sent        */
        unsigned long   out_xid_frames;      /* number of XID frames sent     */
        unsigned long   out_msg_frames;      /* num message frames sent       */
        unsigned long   in_invalid_sna_frames;
                                             /* num invalid frames recvd      */
        unsigned long   in_session_control_frames;
                                             /* number of control             */
                                             /* frames recvd                  */
        unsigned long   out_session_control_frames;
                                             /* number of control             */
                                             /* frames sent                   */
        unsigned long   echo_rsps;           /* response from adj LS count    */
        unsigned long   current_delay;       /* time taken for last           */
                                             /* test signal                   */
        unsigned long   max_delay;           /* max delay by test signal      */
        unsigned long   min_delay;           /* min delay by test signal      */
        unsigned long   max_delay_time;      /* time since longest delay      */
        unsigned long   good_xids;           /* successful XID on LS count    */
        unsigned long   bad_xids;            /* unsuccessful XID on LS count  */
} LS_STATS;
```

パラメーター

opcode

AP_LS_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated

LS が非活動状態になったときに AP_YES に設定します。LS が活動状態になったときに AP_NO に設定します。

ls_name

リンク・ステーションの名前。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

description

資源の記述 (DEFINE_LS で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

adj_cp_name

完全修飾で、17 バイトの長さの隣接制御点名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

adj_node_type

ノードのタイプ。以下の値のいずれか 1 つになります。

AP_END_NODE
AP_NETWORK_NODE
AP_LEN_NODE
AP_VRN

act_sess_count

リンクを使用する活動中セッション（端点と中間の両方）の数の合計

indication_cause

指示の原因。以下の値のいずれか 1 つになります。

AP_ACTIVATION_STARTED
リンクを活動化中。

AP_ACTIVATING
リンクが活動状態になった。

AP_DEACTIVATION_STARTED
リンクを非活動化中。

AP_DEACTIVATING
リンクが非活動状態になった。

AP_SESS_COUNT_CHANGING
リンクを使用する活動セッションの数が変更された。

AP_CP_NAME_CHANGING
隣接ノードの制御点名が変更された。

AP_FAILED
リンクが失敗した。

AP_ACTIVATION_FAILED
リンクの活動化に失敗した。

AP_DATA_LOST
直前の指示が脱落した。リンク・ステーション統計は、リンク・ステーションが活動状態から非活動状態に切り替わったときのみ提供される（つまり、非活動化を AP_YES に設定し、**indication_cause** を AP_DEACTIVATING に設定しているとき)ことに注意してください。それ以外の場合、フィールドは予約済みです。

ls_stats.in_xid_bytes
このリンク・ステーションで受信する XID (交換識別) バイト数の合計

ls_stats.in_msg_bytes
このリンク・ステーションで受信するデータ・バイト数の合計

ls_stats.in_xid_frames
このリンク・ステーションで受信する XID (交換識別) フレーム数の合計

ls_stats.in_msg_frames
このリンク・ステーションで受信するデータ・フレーム数の合計

ls_stats.out_xid_bytes
このリンク・ステーションで送信する XID (交換識別) バイト数の合計

ls_stats.out_msg_bytes
このリンク・ステーションで送信するデータ・バイト数の合計

ls_stats.out_xid_frames
このリンク・ステーションで送信する XID (交換識別) フレーム数の合計

ls_stats.out_msg_frames
このリンク・ステーションで送信するデータ・フレーム数の合計

ls_stats.in_invalid_sna_frames
このリンク・ステーションで受信する SNA の誤ったフレーム数の合計

ls_stats.in_session_control_frames
このリンク・ステーションで受信するセッション制御フレーム数の合計

ls_stats.out_session_control_frames
このリンク・ステーションで送信するセッション制御フレーム数の合計

ls_stats.echo_rsps
隣接ノードから受信するエコー応答の数。エコー要求が定期的に送信され、隣接ノードへの波及遅延を判別します。

ls_stats.current_delay
このリンク・ステーションから隣接リンク・ステーションへ最後に送信したテスト信号の送信および戻りにかかった時間 (ミリ秒単位)

ls_stats.max_delay
このリンク・ステーションから隣接リンク・ステーションへ送信したテスト信号の送信および戻りにかかった最長時間 (ミリ秒単位)

ls_stats.min_delay
このリンク・ステーションから隣接リンク・ステーションへ送信したテスト信号の送信および戻りにかかった最短時間 (ミリ秒単位)

ls_stats.max_delay_time
最長の遅延が起こったときに、システム始動以降経過した時間 (1/100 秒単位)

ls_stats.good_xids
開始以降このリンク・ステーションで起こった正常な XID 交換数の合計

ls_stats.bad_xids
開始以降このリンク・ステーションで起こった正常でない XID 交換数の合計

tg_num
TG と関連した数。

sense_data
このセンス・データは、Communications Server が XID プロトコル・エラーを検出したときに設定します。 **indication_cause** に AP_FAILED を設定しない限り、このフィールドは予約済みです。

LU_0_TO_3_INDICATION

この指示は、ローカル LU（タイプ 0-3）の状態が変更されたときに生成されます。

VCB 構造

```
typedef struct lu_0_to_3_indication
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   data_lost;           /* previous indication lost      */
        unsigned char   pu_name[8];          /* PU Name                       */
        unsigned char   lu_name[8];          /* LU Name                       */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   nau_address;         /* NAU address                   */
        unsigned char   lu_sscp_sess_active;
                                             /* Is SSCP session active?       */
        unsigned char   appl_conn_active;    /* Is application using LU?      */
        unsigned char   plu_sess_active;     /* Is PLU-SLU session active?    */
        unsigned char   host_attachment;     /* Host attachment               */
        SESSION_STATS   lu_sscp_stats;       /* LU-SSCP session statistics    */
        SESSION_STATS   plu_stats;           /* PLU-SLU session statistics    */
} LU_0_TO_3_INDICATION;
typedef struct session_stats
{
        unsigned short  rcv_ru_size;         /* session receive RU siz e      */
        unsigned short  send_ru_size;        /* session send RU size          */
        unsigned short  max_send_btu_size;   /* max send BTU size             */
        unsigned short  max_rcv_btu_size;    /* max rcv BTU size              */
        unsigned short  max_send_pac_win;    /* max send pacing window size   */
        unsigned short  cur_send_pac_win;    /* current send pacing win size  */
        unsigned short  max_rcv_pac_win;     /* max receive pacing win size   */
        unsigned short  cur_rcv_pac_win;     /* curr receive pacing win size  */
        unsigned long   send_data_frames;    /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                             /* num of FMD data frames sent   */
        unsigned long   send_data_bytes;     /* number of data bytes sent     */
        unsigned long   rcv_data_frames;     /* num of data frames received   */
        unsigned long   rcv_fmd_data_frames;
                                             /* num FMD data frames received  */
        unsigned long   rcv_data_bytes;      /* number of data bytes received */
        unsigned char   sidh;                /* session ID high byte          */
        unsigned char   sidl;                /* session ID low byte           */
        unsigned char   odai;                /* ODAI bit set                  */
        unsigned char   ls_name[8];          /* Link station name             */
        unsigned char   reserve;             /* reserved                      */
} SESSION_STATS;
```

パラメーター

opcode

AP_LU_0_TO_3_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc
AP_OK

secondary_rc
ゼロと同等。

data_lost
データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

pu_name
ローカル PU 名。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

lu_name
状態が変更されたローカル LU の名前。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description
資源の記述 (DEFINE_LU_0_TO_3 で指定されたもの) 。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

nau_address
LU のネットワーク・アドレス可能単位のアドレス（10 から 2554 の範囲内でなければならない）。

lu_sscp_sess_active
ACTLU が正常に処理されたかどうかを指定します（AP_YES または AP_NO）。

appl_conn_active
アプリケーションがこの LU を使用している場合に設定します（AP_YES または AP_NO）。

plu_sess_active
PLU-SLU セッションが活動化されたかどうかを指定します（AP_YES または AP_NO)。

host_attachment
LU のホスト接続のタイプを指定します。

AP_DLUR_ATTACHED
LU は DLUR を使用してホスト・システムに接続されています。

AP_DIRECT_ATTACHED
LU は直接ホスト・システムに接続されています。LU-SSCP 統計お

よび PLU-SLU 統計は、セッションが活動状態から非活動状態に切り替わったときのみ有効であることに注意してください。それ以外の場合、フィールドは予約済みです。

lu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

lu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

lu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

lu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

lu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

lu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

lu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

lu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

lu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

lu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

lu_sscp_stats.sidh
セッション ID の高位バイト

lu_sscp_stats.sidl
セッション ID の低位バイト

lu_sscp_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、ACTLU の送信側がこのフィー

ルドをゼロに設定します。ACTLU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドは ACTLU の送信側により 1 に設定されます。

lu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

plu_stats.rcv_ru_size
受信のときの最大 RU サイズ

plu_stats.send_ru_size
送信のときの最大 RU サイズ

plu_stats.max_send_btu_size
送信できる最大 BTU サイズ

plu_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

plu_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

plu_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

plu_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

plu_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

plu_stats.send_data_frames
送信される通常フロー・データ・フレームの数

plu_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

plu_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

plu_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

plu_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

plu_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

plu_stats.sidh
セッション ID の高位バイト

plu_stats.sidl

セッション ID の低位バイト

plu_stats.odai

起点宛先アドレス標識。 セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、ACTLU の送信側がこのフィールドをゼロに設定します。ACTLU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドは ACTLU の送信側により 1 に設定されます。

plu_stats.ls_name

統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

MODE_INDICATION

この指示は、ローカル LU とパートナー LU の組合せが特定モードの使用を開始するとき、および現行セッションがローカル LU、パートナー LU、およびモードの組合せにおける変更をカウントするときに送信されます。

VCB 構造

```
typedef struct mode_indication
{
        unsigned short  opcode;               /* verb operation code       */
        unsigned char   reserv2;              /* reserved                  */
        unsigned char   format;               /* format                    */
        unsigned short  primary_rc;           /* primary return code       */
        unsigned long   secondary_rc;         /* secondary return code     */
        unsigned char   data_lost;            /* previous indication lost  */
        unsigned char   removed;              /* is entry being removed?   */
        unsigned char   lu_alias[8];          /* LU alias                  */
        unsigned char   plu_alias[8];         /* partner LU alias          */
        unsigned char   fqplu_name[17];       /* fully qualified partner   */
                                              /* LU name                   */
        unsigned char   mode_name[8];         /* mode name                 */
        unsigned char   description[RD_LEN];  /* resource description      */
        unsigned short  curr_sess_count;      /* current session count     */
        unsigned char   reserva[20];          /* reserved                  */
} MODE_INDICATION;
```

パラメーター

opcode

AP_MODE_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

removed

項目を除去するかどうかを指定します（AP_YES または AP_NO）。項目を追加せずに除去するときに設定します。

MODE_INDICATION

lu_alias

ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

plu_alias

パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

mode_name

モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description

資源の記述 (DEFINE_MODE で指定されたもの) 。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

curr_sess_count

このローカル LU、パートナー LU、およびモードの組合せに対するセッションの現行カウントです。

NN_TOPOLOGY_NODE_INDICATION

この指示は、ネットワーク・ノードのトポロジー・データベース内のノード項目が活動状態から非活動状態に変更されたとき、あるいは非活動状態から活動状態に変更されたときに生成されます。

VCB 構造

```
typedef struct nn_topology_node_indication
{
   unsigned short  opcode;         /* verb operation code           */
   unsigned char   reserv2;        /* reserved                      */
   unsigned char   format;         /* format                        */
   unsigned short  primary_rc;     /* primary return code           */
   unsigned long   secondary_rc;   /* secondary return code         */
   unsigned char   data_lost;      /* previous indication lost      */
   unsigned char   deactivated;    /* has the node become inactive? */
   unsigned char   node_name[17];  /* node name                     */
   unsigned char   node_type;      /* node type                     */
   unsigned char   reserva[20];    /* reserved                      */
} NN_TOPOLOGY_NODE_INDICATION;
```

パラメーター

opcode

AP_NN_TOPOLOGY_TG_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated

ノードが非活動状態になったときに AP_YES に設定します。ノードが活動状態になったときに AP_NO に設定します。

node_name

ネットワーク・トポロジー・データベースからのネットワーク修飾ノード名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

node_type

ノードのタイプ。 タイプは以下のいずれか 1 つです。

NN_TOPOLOGY_NODE_INDICATION

AP_NETWORK_NODE
AP_VRN

NN_TOPOLOGY_TG_INDICATION

この指示は、ネットワーク・ノードのトポロジー・データベース内の TG 項目が活動状態から非活動状態に変更されたとき、あるいは非活動状態から活動状態に変更されたときに生成されます。

VCB 構造

```
typedef struct nn_topology_tg_indication
{
        unsigned short  opcode;             /* verb operation code           */
        unsigned char   reserv2;            /* reserved                      */
        unsigned char   format;             /* format                        */
        unsigned short  primary_rc;         /* primary return code           */
        unsigned long   secondary_rc;       /* secondary return code         */
        unsigned char   data_lost;          /* previous indication lost      */
        unsigned char   status;             /* TG status                     */
        unsigned char   owner[17];          /* name of TG owner node         */
        unsigned char   dest[17];           /* name of TG destination node   */
        unsigned char   tg_num;             /* TG number                     */
        unsigned char   owner_type;         /* Type of node that owns the TG */
        unsigned char   dest_type;          /* TG destination node type      */
        unsigned char   reserva[18];        /* reserved                      */
} NN_TOPOLOGY_TG_INDICATION;
```

パラメーター

opcode

AP_NN_TOPOLOGY_TG_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

状況（status）

TG の状況を指定します。これは、以下の値の 1 つ以上を一緒に論理和 (OR) 演算したものです。

AP_TG_OPERATIVE
AP_TG_QUIESCING
AP_TG_CP_CP_SESSIONS
AP_NONE

owner TG の起点ノード名（常にローカル・ノード名に設定する）。この名前の長

さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

dest TG に対する完全修飾の宛先ノード名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

tg_num
TG と関連した数。

owner_type
TG を所有するノードのタイプ。

AP_NETWORK_NODE
AP_VRN

dest_type
ノードのタイプ。

AP_NETWORK_NODE
AP_VRN

PLU_INDICATION

この指示は、ローカル LU が最初にパートナー LU に接続するときに生成されます。この指示は、この PLU に対する最初の ALLOCATE が処理されるとき、あるいはこの PLU から最初の BIND が受信されるときに生成されます。 この指示はまた、パートナーの制御点名が変更された場合も生成されます。

VCB 構造

```
typedef struct plu_indication
{
        unsigned short  opcode;              /* verb operation code        */
        unsigned char   reserv2;             /* reserved                   */
        unsigned char   format;              /* format                     */
        unsigned short  primary_rc;          /* primary return code        */
        unsigned long   secondary_rc;        /* secondary return code      */
        unsigned char   data_lost;           /* has previous indication    */
                                             /* been lost?                 */
        unsigned char   removed;             /* is entry being removed?    */
        unsigned char   lu_alias[8];         /* LU alias                   */
        unsigned char   plu_alias[8];        /* partner LU alias           */
        unsigned char   fqplu_name[17];      /* fully qualified partner    */
                                             /* LU name                    */
        unsigned char   description[RD_LEN]; /* resource description       */
        unsigned char   partner_cp_name[17]; /* partner CP name            */
        unsigned char   partner_lu_located;  /* partner CP name resolved?  */
        unsigned char   reserva[20];         /* reserved                   */
} PLU_INDICATION;
```

パラメーター

opcode

AP_PLU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

1 つまたは複数の指示が脱落したかどうかを指定します（AP_YES または AP_NO）。 内部構成要素が直前の指示を送信することができなかったときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

removed

項目を除去するかどうかを指定します（AP_YES または AP_NO）。 項目を追加せずに除去するときに設定します。

PLU_INDICATION

lu_alias

ローカルで定義されている LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

plu_alias

パートナー LU 別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

description

資源の記述 (DEFINE_PARTNER_LU で指定されたものです)。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

partner_cp_name

パートナー LU の制御点に対する 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 (各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

partner_lu_located

パートナーの制御点名が解決されたかどうか (AP_YES または AP_NO)、つまり、**partner_cp_name** フィールドに制御点名が含まれるかどうかを指定します。

PORT_INDICATION

この指示は、ポートが活動状態から非活動状態（またはその逆）に切り替わったときに生成されます。

VCB 構造

```
typedef struct port_indication
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   data_lost;           /* previous indication lost      */
        unsigned char   deactivated;         /* has session been deactivated? */
        unsigned char   port_name[8];        /* link station name             */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   reserva[20];         /* reserved                      */
} PORT_INDICATION;
```

パラメーター

opcode

AP_PORT_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated

ポートが非活動状態になったときに AP_YES に設定します。 ポートが活動状態になったときに AP_NO に設定します。

port_name

ポート名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。

PORT_INDICATION

description

資源の記述（DEFINE_PORT で指定されたものです）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

PU_INDICATION

この指示は、ローカル PU の状態が変更されたときに生成されます。

VCB 構造

```
typedef struct pu_indication
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   data_lost;           /* previous indication lost      */
        unsigned char   pu_name[8];          /* PU Name                       */
        unsigned char   description[RD_LEN]; /* resource description          */
        unsigned char   pu_sscp_sess_active;
                                             /* Is SSCP session active?       */
        unsigned char   host_attachment;     /* Host attachment               */
        unsigned char   reserv1[2];          /* reserved                      */
        SESSION_STATS   pu_sscp_stats;       /* PU-SSCP session statistics    */
} PU_INDICATION;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;         /* session receive RU size       */
        unsigned short  send_ru_size;        /* session send RU size          */
        unsigned short  max_send_btu_size;   /* max send BTU size             */
        unsigned short  max_rcv_btu_size;    /* max rcv BTU size              */
        unsigned short  max_send_pac_win;    /* max send pacing window size   */
        unsigned short  cur_send_pac_win;    /* curr send pacing window size  */
        unsigned short  max_rcv_pac_win;     /* max rcv pacing window size    */
        unsigned short  cur_rcv_pac_win;     /* curr receive pacing win size  */
        unsigned long   send_data_frames;    /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                             /* num FMD data frames sent      */
        unsigned long   send_data_bytes;     /* number of data bytes sent     */
        unsigned long   rcv_data_frames;     /* num of data frames received   */
        unsigned long   rcv_fmd_data_frames;
                                             /* num FMD data frames received  */
        unsigned long   rcv_data_bytes;      /* num data bytes received       */
        unsigned char   sidh;                /* session ID high byte          */
        unsigned char   sidl;                /* session ID low byte           */
        unsigned char   odai;                /* ODAI bit set                  */
        unsigned char   ls_name[8];          /* Link station name             */
        unsigned char   reserve;             /* reserved                      */
} SESSION_STATS;
```

パラメーター

opcode

AP_PU_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc
AP_OK

secondary_rc
ゼロと同等。

data_lost
データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

pu_name
PU 名（DEFINE_LS verb で構成されたもの）。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

description
資源の記述（DEFINE_LS または DEFINE_INTERNAL_PU で指定されたものです）。ローカルで表示可能な文字セットで、16 バイト・ストリングです。16 バイトすべてが有効です。

pu_sscp_sess_active
ACTPU が正常に処理されたかどうかを指定します（AP_YES または AP_NO）。

host_attachment
PU のホスト接続のタイプで、以下のとおりです。

AP_DLUR_ATTACHED
PU は、DLUR を使用してホスト・システムに接続しています。

AP_DIRECT_ATTACHED
PU は、直接ホスト・システムに接続しています。

注: PU-SSCP 統計は、セッション状態が活動状態から非活動状態に切り替わったときのみ有効です。それ以外の場合、以下のフィールドは予約済みです。

pu_sscp_stats.rcv_ru_size
このフィールドは、常に予約されています。

pu_sscp_stats.send_ru_size
このフィールドは、常に予約されています。

pu_sscp_stats.max_send_btu_size
送信できる最大 BTU サイズ

pu_sscp_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

pu_sscp_stats.max_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.cur_send_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.max_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.cur_rcv_pac_win
このフィールドは、常にゼロに設定されます。

pu_sscp_stats.send_data_frames
送信される通常フロー・データ・フレームの数

pu_sscp_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

pu_sscp_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

pu_sscp_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

pu_sscp_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

pu_sscp_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

pu_sscp_stats.sidh
セッション ID の高位バイト

pu_sscp_stats.sidl
セッション ID の低位バイト

pu_sscp_stats.odai
起点宛先アドレス標識。 セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、 ACTPU の送信側がこのフィールドをゼロに設定します。 ACTPU の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドは ACTPU の送信側により 1 に設定されます。

pu_sscp_stats.ls_name
統計と関連したリンク・ステーション名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効です。 このフィールドを使用して、このセッションとセッションが流れるリンクとを関連付けることができます。

REGISTRATION_FAILURE

REGISTRATION_FAILURE は、ネットワーク・ノード・サーバーによる資源の登録が失敗したことを指示します。

VCB 構造

```
typedef struct registration_failure
{
        unsigned short  opcode;               /* verb operation code      */
        unsigned char   reserv2;              /* reserved                 */
        unsigned char   format;               /* format                   */
        unsigned short  primary_rc;           /* primary return code      */
        unsigned long   secondary_rc;         /* secondary return code    */
        unsigned char   data_lost;            /* previous indication lost */
        unsigned char   resource_name[17];    /* network qualified        */
                                              /* resource name            */
        unsigned short  resource_type;        /* resource type            */
        unsigned char   description[RD_LEN];  /* resource description     */
        unsigned char   reserv2b[2];          /* reserved                 */
        unsigned long   sense_data;           /* sense data               */
        unsigned char   reserva[20];          /* reserved                 */
} REGISTRATION_FAILURE;
```

パラメーター

opcode

AP_REGISTRATION_FAILURE

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

resource_name

登録に失敗した資源名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。この名前は、EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

resource_type

資源タイプ。値は以下のいずれかの 1 つです。

AP_NNCP_RESOURCE
AP_ENCP_RESOURCE
AP_LU_RESOURCE

description

資源の記述 (DEFINE_LOCAL_LU、または DEFINE_ADJACENT_NODE で指定されたものです) 。

sense_data

センス・データ（*SNA Formats* で指定されているもの)。

RTP_INDICATION

この指示が生成されるのは、以下の場合です。

- RTP 接続が接続または切断されたとき
- 活動セッションのカウントが変更されたとき
- 接続がパス・スイッチを実行するとき

接続が切断される場合、最終 RTP 統計が戻されます。 接続が切断されていない場合、**rtp_stats** フィールドは予約済みです。

VCB 構造

```
typedef struct rtp_indication
{
        unsigned short  opcode;              /* verb operation code           */
        unsigned char   reserv2;             /* reserved                      */
        unsigned char   format;              /* format                        */
        unsigned short  primary_rc;          /* primary return code           */
        unsigned long   secondary_rc;        /* secondary return code         */
        unsigned char   data_lost;           /* previous indication(s) lost   */
        unsigned char   connection_state;    /* the current state of the RTP  */
                                             /* connection                    */
        unsigned char   rtp_name[8];         /* name of the RTP connection    */
        unsigned short  num_sess_active;     /* number of active sessions     */
        unsigned char   indication_cause;    /* reason for this indication    */
        unsigned char   reserv3[3];          /* reserved                      */
        RTP_STATISTICS  rtp_stats;           /* RTP statistics                */
} RTP_INDICATION;

typedef struct rtp_statistics
{
        unsigned long   bytes_sent;         /* total num of bytes sent       */
        unsigned long   bytes_received;     /* total num bytes received      */
        unsigned long   bytes_resent;       /* total num of bytes resent     */
        unsigned long   bytes_discarded;    /* total num bytes discarded     */
        unsigned long   packets_sent;       /* total num of packets sent     */
        unsigned long   packets_received;   /* total num packets received    */
        unsigned long   packets_resent;     /* total num of packets resent   */
        unsigned long   packets_discarded;  /* total num packets discarded   */
        unsigned long   gaps_detected;      /* gaps detected                 */
        unsigned long   send_rate;          /* current send rate             */
        unsigned long   max_send_rate;      /* maximum send rate             */
        unsigned long   min_send_rate;      /* minimum send rate             */
        unsigned long   receive_rate;       /* current receive rate          */
        unsigned long   max_receive_rate;   /* maximum receive rate          */
        unsigned long   min_receive_rate;   /* minimum receive rate          */
        unsigned long   burst_size;         /* current burst size            */
        unsigned long   up_time;            /* total uptime of connection    */
        unsigned long   smooth_rtt;         /* smoothed round-trip time      */
        unsigned long   last_rtt;           /* last round-trip time          */
        unsigned long   short_req_timer;    /* SHORT_REQ timer duration      */
        unsigned long   short_req_timeouts; /* number of SHORT_REQ timeouts  */
        unsigned long   liveness_timeouts;  /* number of liveness timeouts   */
        unsigned long   in_invalid_sna_frames;
                                            /* number of invalid SNA frames  */
```

```
                                        /* received                        */
        unsigned long   in_sc_frames;   /* number of SC frames received    */
        unsigned long   out_sc_frames;  /* number of SC frames sent        */
        unsigned char   reserve[40];    /* reserved                        */
} RTP_STATISTICS;
```

パラメーター

opcode

AP_RTP_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。**data_lost** フラグを AP_YES に設定した場合、直前の指示が受信されてから、データが 2 回以上変更された可能性があります。

connection_state

RTP 接続の現行の状態。以下の値のいずれか 1 つになります。

AP_CONNECTING

接続のセットアップは開始されたが、まだ完了していない。

AP_CONNECTED

接続は完全に活動状態にある。

AP_DISCONNECTED

接続はもはや活動状態にない。

rtp_name

RTP 結合名。この名前は、ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

num_sess_active

現在活動状態にある接続上のセッション数。

indication_cause

指示の原因。以下の値のいずれか 1 つになります。

AP_ACTIVATED

接続が活動状態になった。

AP_DEACTIVATED

接続が非活動状態になった。

AP_PATH_SWITCHED

接続が正常にパス・スイッチを完了した。

AP_SESS_COUNT_CHANGING
接続を使用する活動セッションの数が変更された。

AP_SETUP_FAILED
接続が、完全に活動状態になる前に失敗した。RTP 接続統計は、接続が活動状態から非活動状態に切り替わったときのみ提供される、つまり、**indication_cause** を AP_DEACTIVATED または AP_SETUP_FAILED に設定しているときのみ提供されることに注意してください。それ以外の場合、フィールドは予約済みです。

rtp_stats.bytes_sent
ローカル・ノードがこの RTP 接続上で送信したバイトの合計数

rtp_stats.bytes_received
ローカル・ノードがこの RTP 接続上で受信したバイトの合計数

rtp_stats.bytes_resent
転送中のロスのため、ローカル・ノードが再送したバイトの合計数

rtp_stats.bytes_discarded
RTP 接続のもう一方の起点が送信し、受信済みのデータと重複するとして廃棄されたバイト数の合計。

rtp_stats.packets_sent
ローカル・ノードがこの RTP 接続上で送信したパケットの合計数

rtp_stats.packets_received
ローカル・ノードがこの RTP 接続上で受信したパケットの合計数

rtp_stats.packets_resent
転送中のロスのため、ローカル・ノードが再送したパケットの合計数

rtp_stats.packets_discarded
RTP 接続の他の終端から送信され、すでに受信済みのデータの重複として廃棄されたパケットの合計数

rtp_stats.gaps_detected
ローカル・ノードによって検出されたギャップの合計数。各ギャップは、1 つ以上の失われたフレームに対応します。

rtp_stats.send_rate
この RTP 接続上の現行の送信レート（秒当たりのキロビット数です）。これは、ARB アルゴリズムで計算される最大許容の送信レートです。

rtp_stats.max_send_rate
この RTP 接続上の最大送信レート（秒当たりのキロビット数です）

rtp_stats.min_send_rate
この RTP 接続上の最小送信レート（秒当たりのキロビット数です）

rtp_stats.receive_rate
この RTP 接続上の現行の受信レート（秒当たりのキロビット数です）これは、最後の測定間隔において計算された実際の受信レートです。

rtp_stats.max_receive_rate
この RTP 接続上の最大受信レート（秒当たりのキロビット数です）

rtp_stats.min_receive_rate
この RTP 接続上の最小受信レート（秒当たりのキロビット数です）

rtp_stats.burst_size
この RTP 接続のバイト単位の現行バースト・サイズ。

rtp_stats.up_time
RTP 接続が活動状態であった秒数の合計

rtp_stats.smooth_rtt
ローカル・ノードとパートナー RTP ノード間の、平滑化された往復時間（ミリ秒です）

rtp_stats.last_rtt
ローカル・ノードとパートナー RTP ノード間の、最後に測定された往復時間（ミリ秒です）

rtp_stats.short_req_timer
SHORT_REQ タイマーに対して使用される現行期間（ミリ秒です）

rtp_stats.short_req_timeouts
この RTP 接続に対して SHORT_REQ タイマーが期限切れした合計回数

rtp_stats.liveness_timeouts
この RTP 接続に対してライブネス・タイマーが期限切れした合計回数ライブネス・タイマーが期限切れするのは、接続が **rtp_connection_detail.liveness_timer** に指定した期間にわたってアイドルであったときです。

rtp_stats.in_invalid_sna_frames
この RTP 接続上で受信され、無効として廃棄された SNA フレームの合計数

rtp_stats.in_sc_frames
この RTP 接続上で受信されたセッション制御フレームの合計数

rtp_stats.out_sc_frames
この RTP 接続上で送信されたセッション制御フレームの合計数

SESSION_INDICATION

この指示は、セッションを活動化または非活動化するときに生成されます。 セッションを非活動化する場合、最終セッション統計が戻されます。 セッションを活動化する場合、**sess_stats** フィールドは予約済みです。

VCB 構造

```
typedef struct session_indication
{
        unsigned short  opcode;            /* verb operation code           */
        unsigned char   reserv2;           /* reserved                      */
        unsigned char   format;            /* format                        */
        unsigned short  primary_rc;        /* primary return code           */
        unsigned long   secondary_rc;      /* secondary return code         */
        unsigned char   data_lost;         /* previous indication lost      */
        unsigned char   deactivated;       /* has session been deactivated? */
        unsigned char   lu_name[8];        /* LU name                       */
        unsigned char   lu_alias[8];       /* LU alias                      */
        unsigned char   plu_alias[8];      /* partner LU alias              */
        unsigned char   fqplu_name[17];    /* fully qualified partner       */
                                           /* LU name                       */
        unsigned char   mode_name[8];      /* mode name                     */
        unsigned char   session_id[8];     /* session ID                    */
        FQPCID          fqpcid;            /* fully qualified procedure     */
        unsigned long   sense_data;        /* sense_data                    */
        unsigned char   duplex_support;    /* full-duplex support           */
        SESSION_STATS   sess_stats;        /* session statistics            */
                                           /* correlator ID                 */
       unsigned char   reserva[20];        /* reserved                      */
} SESSION_INDICATION;

typedef struct fqpcid
{
        unsigned char   pcid[8];           /* procedure correlator         */
                                           /* identifier                   */
        unsigned char   fqcp_name[17];     /* originator's network         */
                                           /* qualified CP name            */
        unsigned char   reserve3[3];       /* reserved                     */
} FQPCID;

typedef struct session_stats
{
        unsigned short  rcv_ru_size;       /* session receive RU size       */
        unsigned short  send_ru_size;      /* session send RU size          */
        unsigned short  max_send_btu_size; /* max send BTU size             */
        unsigned short  max_rcv_btu_size;  /* max rcv BTU size              */
        unsigned short  max_send_pac_win;  /* max send pacing window size   */
        unsigned short  cur_send_pac_win;  /* curr send pacing window size  */
        unsigned short  max_rcv_pac_win;   /* max receive pacing win size   */
        unsigned short  cur_rcv_pac_win;   /* curr receive pacing win size  */
        unsigned long   send_data_frames;  /* number of data frames sent    */
        unsigned long   send_fmd_data_frames;
                                           /* num FMD data frames sent      */
        unsigned long   send_data_bytes;   /* number of data bytes sent     */
        unsigned long   rcv_data_frames;   /* num data frames received      */
        unsigned long   rcv_fmd_data_frames;
```

```
                                        /* num FMD data frames received  */
        unsigned long   rcv_data_bytes;    /* num data bytes received       */
        unsigned char   sidh;              /* session ID high byte          */
        unsigned char   sidl;              /* session ID low byte           */
        unsigned char   odai;              /* ODAI bit set                  */
        unsigned char   ls_name[8];        /* Link station name             */
        unsigned char   reserve;           /* reserved                      */
} SESSION_STATS;
```

パラメーター

opcode

AP_SESSION_INDICATION

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

primary_rc

AP_OK

secondary_rc

ゼロと同等。

data_lost

データが脱落したかどうかを指定します（AP_YES または AP_NO）。 直前の指示が脱落する原因となった障害を内部構成要素が検出したときに設定します。 **data_lost** フラグを AP_YES に設定すると、後続のデータ・フィールドをヌルに設定することができます。 アプリケーションは QUERY 動詞を発行して、脱落した情報を更新する必要があります。

deactivated

セッションを活動化するときに、AP_NO に設定します。セッションを非活動化するときに、AP_YES に設定します。

lu_name

LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。

lu_alias

ローカルで定義されている LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。

plu_alias

パートナー LU 別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。

fqplu_name

パートナー LU の 17 バイトの完全修飾ネットワーク名。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

SESSION_INDICATION

mode_name
モード名。セッションのグループのネットワーク特性を指定します。8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

session_id
セッションの 8 バイト識別子。

fqpcid.pcid
プロシージャー相関 ID。これは 8 バイトの 16 進数ストリングです。

fqpcid.fqcp_name
完全修飾制御点名。この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。(各名前は組込みスペースなしで最大 8 バイトまで指定できます。)

sense_data
UNBIND 要求で送信または受信されるセンス・データ。**deactivated** に AP_NO が指定されている場合、このフィールドは予約済みです。

duplex_support
BIND で折衝されたとおりに会話二重サポートを戻します。これは、以下の値の 1 つになります。

AP_HALF_DUPLEX
半二重会話だけがサポートされます。

AP_FULL_DUPLEX
半二重会話と同様に全二重会話がサポートされます。

AP_UNKNOWN
会話二重サポートは不明です。その理由は、パートナー LU に対する活動状態のセッションが存在しないからです。

sess_stats.rcv_ru_size
受信のときの最大 RU サイズ

sess_stats.send_ru_size
送信のときの最大 RU サイズ

sess_stats.max_send_btu_size
送信できる最大 BTU サイズ

sess_stats.max_rcv_btu_size
受信できる最大 BTU サイズ

sess_stats.max_send_pac_win
このセッションでの送信ペーシング・ウィンドウの最大サイズ

sess_stats.cur_send_pac_win
このセッションでの送信ペーシング・ウィンドウの現行サイズ

sess_stats.max_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの最大サイズ

sess_stats.cur_rcv_pac_win
このセッションでの受信ペーシング・ウィンドウの現行サイズ

sess_stats.send_data_frames
送信される通常フロー・データ・フレームの数

sess_stats.send_fmd_data_frames
送信される通常フロー FMD データ・フレームの数

sess_stats.send_data_bytes
送信される通常フロー・データ・バイトの数

sess_stats.rcv_data_frames
受信される通常フロー・データ・フレームの数

sess_stats.rcv_fmd_data_frames
受信される通常フロー FMD データ・フレームの数

sess_stats.rcv_data_bytes
受信される通常フロー・データ・バイトの数

sess_stats.sidh
セッション ID の高位バイト

sess_stats.sidl
セッション ID の低位バイト

sess_stats.odai
起点宛先アドレス標識。セッションが起動されたときに、ローカル・ノードに 1 次リンク・ステーションがある場合は、BIND の送信側がこのフィールドをゼロに設定し、BIND の送信側が 2 次リンク・ステーションが入っているノードである場合は、このフィールドを 1 に設定します。

sess_stats_stats.ls_name
統計と関連したリンク・ステーション名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。8 バイトすべてが有効です。このフィールドを使用して、セッション統計とセッション・トラフィックが流れるリンクを相関させることができます。

SESSION_INDICATION

第9章 機密保護 verb

この章は、セキュリティー・パスワードを定義したり削除するために使用する verb について説明します。

DEFINE_LU_LU_PASSWORD

DEFINE_LU_LU_PASSWORD は、ローカル LU とパートナー LU 間のセッション・レベルの検証に使用するパスワードを提供します。

VCB 構造

```
typedef struct define_lu_lu_password
{
        unsigned short  opcode;           /* verb operation code        */
        unsigned char   reserv2;          /* reserved                   */
        unsigned char   format;           /* format                     */
        unsigned short  primary_rc;       /* primary return code        */
        unsigned long   secondary_rc;     /* secondary return code      */
        unsigned char   lu_name[8];       /* local LU name              */
        unsigned char   lu_alias[8];      /* local LU alias             */
        unsigned char   fqplu_name[17];   /* fully qualified partner    */
                                          /* LU name                    */
        unsigned char   verification_protocol
                                          /* LULU verification protocol */
        unsigned char   description[RD_LEN];
                                          /* resource description       */
        unsigned char   reserv3[8];       /* reserved                   */
        unsigned char   password[8];      /* password                   */
} DEFINE_LU_LU_PASSWORD;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_LU_LU_PASSWORD

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

ローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

ローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。 **lu_alias** および **lu_name** の両方をすべてゼロで設定した場合、この verb は制御点に関連した LU（省略時 LU）に送られます。

fqplu_name

完全修飾パートナー LU 名。 この名前は 17 バイトの長さで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つ

のタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

verification_protocol

このパートナー LU に関して使用するLU-LU 検証プロトコル。

AP_BASIC_PROTOCOL

このパートナー LU に関しては、基本プロトコルだけを使用します。

AP_ENHANCED_PROTOCOL

このパートナー LU に関しては、拡張プロトコルだけを使用します。

AP_EITHER_PROTOCOL

このパートナー LU に関しては、基本プロトコルあるいは拡張プロトコルのいずれかを使用することができます。どちらを使用する場合も、以下の詳細に従ってください。

- このフィールドの省略時設定値は、AP_EITHER_PROTOCOL です。
- 値 AP_EITHER_PROTOCOL は、拡張プロトコルの使用への移行を容易にするために用意されています。 ローカル LU は、パートナー LU が拡張プロトコルの実行に同意するまで基本プロトコルを受け入れます。 パートナー LU が拡張プロトコルの実行に同意した場合、それ以後は、基本プロトコルを許可する後続の DEFINE_LU_LU_PASSWORD が発行されない限り、基本プロトコルは受け入れられません。

description

資源の記述。

パスワード (password)

パスワード。 これは 8 バイトの 16 進数ストリングです。 パスワードの各バイトの最低位ビットはセッション・レベルの検証には使用されないことに注意してください。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_PLU_NAME

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_USERID_PASSWORD

DEFINE_USERID_PASSWORD は、ユーザー ID に関連したパスワードを定義します。

VCB 構造

```
define_userid_password
{
        unsigned short  opcode;              /* verb operation code        */
        unsigned char   reserv2;             /* reserved                   */
        unsigned char   format;              /* format                     */
        unsigned short  primary_rc;          /* primary return code        */
        unsigned long   secondary_rc;        /* secondary return code      */
        unsigned short  define_type;         /* what the define type is    */
        unsigned char   user_id[10];         /* user id                    */
        unsigned char   reserv3[8];          /* reserved                   */
        USERID_PASSWORD_CHARS password_chars;
                                             /* password characteristics   */
} DEFINE_USERID_PASSWORD;

typedef struct userid_password_chars
{
        unsigned char   description[RD_LEN];
                                             /* resource description       */
        unsigned short  profile_count;       /* number of profiles         */
        unsigned short  reserv1;             /* reserved                   */
        unsigned char   password[10];        /* password                   */
        unsigned char   profiles[10][10];    /* profiles                   */
} USERID_PASSWORD_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_USERID_PASSWORD

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

define_type

定義済みのユーザー・パスワードのタイプを指定します。

AP_ADD_USER

新規ユーザーを指定したり、または既存のユーザーのパスワードの変更を指定します。

AP_ADD_PROFILES

既存のユーザーのプロファイルへの追加を指定します。

user_id

ユーザー識別子。これは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

password_chars.description
資源の記述。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

password_chars.profile_count
プロファイルの数。

password_chars.password
ユーザー・パスワード。 これは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

password_chars.profiles
ユーザーに関連したプロファイル。 各プロファイルは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_NO_PROFILES

AP_UNKNOWN_USER
AP_INVALID_UPDATE_TYPE
AP_TOO_MANY_PROFILES
AP_INVALID_USERID
AP_INVALID_PROFILE
AP_INVALID_PASSWORD

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_LU_LU_PASSWORD

DELETE_LU_LU_PASSWORD は LU-LU パスワードを削除します。

VCB 構造

```
typedef struct delete_lu_lu_password
{
        unsigned short  opcode;             /* verb operation code       */
        unsigned char   reserv2;            /* reserved                  */
        unsigned char   format;             /* format                    */
        unsigned short  primary_rc;         /* primary return code       */
        unsigned long   secondary_rc;       /* secondary return code     */
        unsigned char   lu_name[8];         /* LU name                   */
        unsigned char   lu_alias[8];        /* local LU alias            */
        unsigned char   fqplu_name[17];     /* fully qualified partner   */
                                            /* LU name                   */
        unsigned char   reserv3;            /* reserved                  */
} DELETE_LU_LU_PASSWORD;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_LU_LU_PASSWORD

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

ローカル LU の LU 名。 この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。 このフィールドをすべてゼロで設定した場合、**lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

ローカル LU の別名。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、**lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。 **lu_alias** および **lu_name** の両方をすべてゼロで設定した場合、この verb は制御点に関連した LU（省略時 LU）に送られます。

fqplu_name

完全修飾パートナー LU 名。 この名前の長さは 17 バイトで、右側は EBCDIC スペースで埋められます。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成されます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PLU_NAME

AP_INVALID_LU_NAME
AP_INVALID_LU_ALIAS

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_USERID_PASSWORD

DELETE_USERID_PASSWORD は、ユーザー ID に関連したパスワードを削除します。

VCB 構造

```
typedef struct delete_userid_password
{
        unsigned short  opcode;               /* verb operation code        */
        unsigned char   reserv2;              /* reserved                   */
        unsigned char   format;               /* format                     */
        unsigned short  primary_rc;           /* primary return code        */
        unsigned long   secondary_rc;         /* secondary return code      */
        unsigned short  delete_type;          /* type of delete             */
        unsigned char   user_id[10];          /* user id                    */
        USERID_PASSWORD_CHARS password_chars;
                                              /* password characteristics   */
} DELETE_USERID_PASSWORD;
typedef struct userid_password_chars
{
        unsigned char   description[RD_LEN];  /* resource description       */
        unsigned short  profile_count;        /* number of profiles         */
        unsigned short  reserv1;              /* reserved                   */
        unsigned char   password[10];         /* password                   */
        unsigned char   profiles[10][10];     /* profiles                   */
} USERID_PASSWORD_CHARS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_USERID_PASSWORD

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

delete_type

削除のタイプを指定します。

AP_REMOVE_USER

ユーザー・パスワードおよび関連するすべてのプロファイルを削除します。

AP_REMOVE_PROFILES

指定したプロファイルを削除します。

user_id

ユーザー識別子。 これは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

password_chars.description

このフィールドは、この verb の処理では無視されます。

password_chars.profile_count
プロファイルの数。

password_chars.password
このフィールドは、この verb の処理では無視されます。

password_chars.profiles
ユーザーに関連したプロファイル。 各プロファイルは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_NO_PROFILES

AP_UNKNOWN_USER
AP_INVALID_UPDATE_TYPE

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

DELETE_USERID_PASSWORD

第10章 APING および CPI-C verb

この章では、他のノードに対して 『ping』 を行う verb、および CPI-C サイド情報を定義、削除、照会するために使用する verb を説明します。

APING

APING は、管理アプリケーションがネットワーク内のリモート LU に対して 『ping』を実行することを許可します。 **partner_ver_len** フィールドにゼロより大きい値を設定した場合、検証データ・ストリング（指定された長さ）を VCB の最後に追加し、戻すことができます。

Communications ServerAPING は、内部『サービス・トランザクション・プログラム』として組み込まれており、Communications Server APPC API (*Communications Server クライアント/サーバー 通信プログラミング* に説明があります)を使用します。

VCB 構造

```
typedef struct aping
{
        unsigned short  opcode;           /* verb operation code          */
        unsigned char   reserv2;          /* reserved                     */
        unsigned char   format;           /* format                       */
        unsigned short  primary_rc;       /* primary return code          */
        unsigned long   secondary_rc;     /* secondary return code        */
        unsigned char   lu_name[8];       /* local LU name                */
        unsigned char   lu_alias[8];      /* local LU alias               */
        unsigned long   sense_data;       /* sense data                   */
        unsigned char   plu_alias[8];     /* partner LU alias             */
        unsigned char   mode_name[8];     /* mode name                    */
        unsigned char   tp_name[64];      /* destination TP name          */
        unsigned char   security;         /* security level               */
        unsigned char   reserv3a[3];      /* reserved                     */
        unsigned char   pwd[10];          /* password                     */
        unsigned char   user_id[10];      /* user ID                      */
        unsigned short  dlen;             /* length of data to send       */
        unsigned short  consec;           /* number of consecutive sends  */
        unsigned char   fqplu_name[17];   /* fully qualified partner      */
                                          /* LU name                      */
        unsigned char   echo;             /* data echo flag               */
        unsigned short  iterations;       /* number of iterations         */
        unsigned long   alloc_time;       /* time taken for ALLOCATE      */
        unsigned long   min_time;         /* min send/receive time        */
        unsigned long   avg_time;         /* average send/receive time    */
        unsigned long   max_time;         /* max send/receive time        */
        unsigned short  partner_ver_len;  /* size of string to receive    */
} APING;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_APING

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

lu_name

APING verb の送信元となるローカル LU の LU 名。この名前は、8 バイトのタイプ A の EBCDIC 文字ストリングです。このフィールドをすべてゼロで設定した場合、 **lu_alias** フィールドがローカル LU 名の判別に使用されます。

lu_alias

APING verb の送信元となるローカル LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。このフィールドは、 **lu_name** フィールドがすべてゼロに設定されている場合にだけ有効で、この場合、すべての 8 バイトが有効であり、8 バイトすべてを設定しなければなりません。 **lu_name** および **lu_alias** の両方を 2 進ゼロに設定した場合、省略時（制御点）LU が使用されます。

plu_alias

ローカル・トランザクション・プログラムに認識されるパートナー LU の別名。ローカルで表示可能な文字セットで 8 バイト・ストリングです。 8 バイトすべてが有効であり、8 バイトすべてを設定しなくてはなりません。この名前は、構成中に設定されたパートナー LU の名前と一致している必要があります。 このパラメーターに 2 進ゼロを設定した場合、**fqplu_name** パラメーターが代りに使用されます。

mode_name

使用するモード名。 8 バイトの英数字のタイプ A の EBCDIC ストリング（先頭は文字）で、右側は EBCDIC スペースで埋められます。

tp_name

呼び出されるトランザクション・プログラムの名前。 これは 64 バイトのストリングです。 ノード・オペレーター機能はこのストリングの文字セットを検査しません。 **tp_name** の値は、リモート LU で構成された値と一致している必要があります。 このストリングは、通常 EBCDIC の『APINGD』 に設定され、右側は EBCDIC スペースで埋められます。

security

呼び出されたトランザクション・プログラムに対するアクセスの妥当性検査を行うために、パートナー LU が必要とする情報を指定します。

AP_NONE
AP_PGM
AP_SAME
AP_PGM_STRONG

pwd **user_id** に関連したパスワード。 これは、10 バイトのタイプ AE の EBCDIC 文字ストリングで、右側は EBCDIC スペースで埋められます。 **security** に AP_PGM または AP_PGM_STRONG を設定した場合のみ必要です。

user_id

パートナー・トランザクション・プログラムにアクセスするために必要なユーザー ID。 これは、10 バイトのタイプ AE の EBCDIC 文字ストリング

で、右側は EBCDIC スペースで埋められます。 **security** に AP_PGM、AP_PGM_STRONG、または AP_SAME を設定した場合のみ必要です。

dlen APING トランザクション・プログラムにより送信されるデータの長さ。APING は、**dlen** の長さで、ゼロで構成されるストリングを送信します。

consec
各反復の実行中に送信される連続送信の回数。 APING は、それぞれが **dlen** バイトのデータで構成される、MC_SEND_DATA verbs をこの数だけ発行します。 **echo** パラメーターに AP_YES を設定した場合、APING は最後の MC_SEND_DATA を AP_SEND_DATA_P_TO_R_FLUSH（フラッシュ受信の準備）としてマークし、パートナー APINGD トランザクション・プログラムからのデータを含む応答を待ちます（MC_RECEIVE_AND_WAIT を発行することにより）。 **echo** パラメーターに AP_NO を設定した場合、APING はデータをフラッシュし、確認を待ちます（最後の MC_SEND_DATA を AP_SEND_DATA_CONFIRM としてマークすることにより）。 どちらの場合も、上記の順序列は SNA 連鎖と対応しています。

fqplu_name
パートナー LU の 17 バイトの完全修飾ネットワーク名。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）このフィールドが有効になるのは、**plu_alias** フィールドがすべてゼロに設定されている場合だけです。

エコー (echo)
APING トランザクション・プログラムが、要求されたデータ量の送信を完了したときに応答を必要とするかどうかを指定します。

AP_YES
AP_NO

iterations
APING が発行する連続順序列（**consec** パラメーターにより定義する）の反復の回数。 SNA としては、このパラメーターは送信される連鎖の数を定義します。

partner_ver_len
管理アプリケーションが受信することができる、パートナー・トランザクション・プログラムの検証データ・ストリングの最大長。

戻りパラメーター

verb が正常に実行されると、APING は以下のパラメーターを戻します。

primary_rc
AP_OK

sense_data
これは、verb が正常に戻された場合ゼロです。

alloc_time

リモート・トランザクション・プログラムに対する MC_ALLOCATE が完了するまでに必要な時間（ミリ秒単位）。

min_time

データ送信の反復に必要な最小時間（ミリ秒単位）。このパラメーターにはパートナーが応答（**echo** パラメーターの設定により異なりますが、データの送信または確認の発行により）するために必要な時間が含まれます。

avg_time

データ送信の反復に必要な平均時間（秒単位）。このパラメーターにはパートナーが応答（**echo** パラメーターの設定により異なりますが、データの送信または確認の発行により）するために必要な時間が含まれます。

max_time

データ送信の反復に必要な最長時間（ミリ秒単位）。このパラメーターにはパートナーが応答（**echo** パラメーターの設定により異なりますが、データの送信または確認の発行により）するために必要な時間が含まれます。

partner_ver_len

パートナー・トランザクション・プログラムにより戻される検証ストリングの長さ。このストリングがそのまま VCB の最後に追加されます。

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_LU_NAME

AP_INVALID_LU_ALIAS

APING は、Communications Server APPC API が提供する MC_ALLOCATE、 MC_SEND_DATA、 MC_RECEIVE_AND_WAIT、 MC_CONFIRM、および MC_DEALLOCATE verbs を使用します。これらの verb が正常に実行されなかった場合に戻されるパラメーターは、*Communications Server クライアント/サーバー 通信プログラミング.* に記載されています。

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DEFINE_CPIC_SIDE_INFO

この verb は、記憶域内のサイド情報項目を追加または置換します。 CPI-C サイド情報項目は、一連の会話特性を記号宛先名と関連付けます。 この verb が指定する記号宛先名と同じ記号宛先名をもつサイド情報項目が記憶域に既に存在する場合、そのサイド情報項目はこの呼出しに指定されているデータにより上書きされます。 Communications Server が提供する CPI-C サポートの詳細については、*CPI-C Reference* を参照してください。

VCB 構造

```
typedef struct define_cpic_side_info
{
        unsigned short  opcode;               /* verb operation code        */
        unsigned char   reserv2;              /* reserved                   */
        unsigned char   format;               /* format                     */
        unsigned short  primary_rc;           /* primary return code        */
        unsigned long   secondary_rc;         /* secondary return code      */
        unsigned char   reserv2a[8];          /* reserved                   */
        unsigned char   sym_dest_name[8];     /* Symbolic destination name  */
        CPIC_SIDE_INFO_DEF_DATA def_data;     /* defined data               */
} DEFINE_CPIC_SIDE_INFO;

typedef struct cpic_side_info_def_data
{
        unsigned char   description[RD_LEN];
                                              /* resource description       */
        CPIC_SIDE_INFO  side_info;            /* CPIC side info             */
        unsigned char   user_data[32];        /* User defined data          */
} CPIC_SIDE_INFO_DEF_DATA;

typedef struct cpic_side_info
{
        unsigned char   partner_lu_name[17];
                                              /* Fully qualified partner    */
                                              /* LU name                    */
        unsigned char   reserved[3];          /* Reserved                   */
        unsigned long   tp_name_type;         /* TP name type               */
        unsigned char   tp_name[64];          /* TP name                    */
        unsigned char   mode_name[8];         /* Mode name                  */
        unsigned long   conversation_security_type;
                                              /* Conversation security type */
        unsigned char   security_user_id[CPIC_SECURITY_INFO_LEN];
                                              /* User ID                    */
        unsigned char   security_password[CPIC_SECURITY_INFO_LEN];
                                              /* Password                   */
} CPIC_SIDE_INFO;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DEFINE_CPIC_SIDE_INFO

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

sym_dest_name
サイド情報項目を示す記号宛先名。これは、最大 8 バイトの長さで、スペースで埋められ、ローカルで表示可能な文字セットにより指定されます。指定できる文字は、英字の大文字（A から Z）および数字の 0 から 9 までです。

def_data.description
資源記述（QUERY_CPIC_SIDE_INFO に対して戻されるもの）。 ローカルで表示可能な文字セットで、16 バイト・ストリングです。 16 バイトすべてが有効です。

def_data.side_info.partner_lu_name
パートナー LU の完全修飾名。 この名前は 17 バイトの長さで、右側はスペースで埋められ、ローカルで表示可能な文字セットにより構成されます。この名前は、ピリオドで連結された 2 つの文字ストリングで構成されます。（各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

def_data.side_info.tp_name_type
トランザクション・プログラム名のタイプ。 このフィールドは、以下の値のいずれか 1 つに設定されます。

XC_APPLICATION_TP
指定されているトランザクション・プログラム名がサービス・トランザクション・プログラム名ではないことを指定します。 トランザクション・プログラム名に指定するすべての文字は、ローカルで表示可能な文字セットの中の有効なものである必要があります。

XC_SNA_SERVICE_TP
指定されているトランザクション・プログラム名がサービス・トランザクション・プログラム名であることを指定します。 トランザクション・プログラム名に指定する先頭文字以外のすべての文字は、ローカルで表示可能な文字セットの中の有効なものである必要があります。先頭文字は、X'01' からX'3F'までの 16 進数(X'0E' から X'0F' を除く）にする必要があります。

def_data.side_info.tp_name
トランザクション・プログラム名。ローカルで表示可能な文字セットの、64 バイトの文字ストリングで、右側はスペースで埋められます。

def_data.side_info.mode_name
モード名。ローカルで表示可能な文字セットの、8 バイトの文字ストリングで、右側はスペースで埋められます。

def_data.side_info.conversation_security_type
会話機密保護のタイプ。 このフィールドは、以下の値のいずれか 1 つに設定されます。

XC_SECURITY_NONE
XC_SECURITY_SAME
XC_SECURITY_PROGRAM
XC_SECURITY_PROGRAM_STRONG

def_data.side_info.security_user_id
ユーザー ID。 Communications Server は、会話レベル・セキュリティーを実施するためにこのフィールドを使用します。

def_data.side_info.security_password
パスワード。 Communications Server は、会話レベル・セキュリティーを実施するためにこのフィールドを使用します。

def_data.user_data
ユーザー・データ。 このデータは QUERY_CPIC_SIDE_INFO に対して戻されますが、Communications Server はこのデータを使用または解釈しません。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_SYM_DEST_NAME

AP_INVALID_LENGTH

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

DEFINE_CPIC_SIDE_INFO

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

DELETE_CPIC_SIDE_INFO

この verb は CPI-C サイド情報項目を削除します。 Communications Server が提供する CPI-C サポートの詳細については、*CPI-C Reference* を参照してください。

VCB 構造

```
typedef struct delete_cpic_side_info
{
   unsigned short  opcode;              /* verb operation code        */
   unsigned char   reserv2;             /* reserved                   */
   unsigned char   format;              /* format                     */
   unsigned short  primary_rc;          /* primary return code        */
   unsigned long   secondary_rc;        /* secondary return code      */
   unsigned char   reserv2a[8];         /* reserved                   */
   unsigned char   sym_dest_name[8];    /* Symbolic destination name  */
} DELETE_CPIC_SIDE_INFO;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DELETE_CPIC_SIDE_INFO

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

sym_dest_name

サイド情報項目を示す記号宛先名。これは、最大 8 バイトの長さで、スペースで埋められ、ローカルで表示可能な文字セットにより指定されます。指定できる文字は、英字の大文字（A から Z）および数字の 0 から 9 までです。

戻りパラメーター

verb が正常に実行された場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_STATE_CHECK

secondary_rc

AP_INVALID_SYM_DEST_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_CPIC_SIDE_INFO

この verb は、指定した記号宛先名に対するサイド情報項目を戻します。 この情報は、リストとして戻されます。 特定サイド情報項目または特定範囲の項目を入手するためには、**sym_dest_name** フィールドを設定する必要があります。 特定サイド情報項目または特定範囲の項目を入手する必要がない場合は、このフィールドはすべてゼロで設定します。

VCB 構造

```
typedef struct query_cpic_side_info
{
        unsigned short  opcode;              /* verb operation code         */
        unsigned char   reserv2;             /* reserved                    */
        unsigned char   format;              /* format                      */
        unsigned short  primary_rc;          /* primary return code         */
        unsigned long   secondary_rc;        /* secondary return code       */
        unsigned char   *buf_ptr;            /* pointer to buffer           */
        unsigned long   buf_size;            /* buffer size                 */
        unsigned long   total_buf_size;      /* total buffer size required  */
        unsigned short  num_entries;         /* number of entries           */
        unsigned short  total_num_entries;   /* total number of entries     */
        unsigned char   list_options;        /* listing options             */
        unsigned char   reserv3;             /* reserved                    */
        unsigned char   sym_dest_name[8];    /* Symbolic destination name   */
} QUERY_CPIC_SIDE_INFO;

typedef struct cpic_side_info_data
{
        unsigned short  overlay_size;        /* size of this entry          */
        unsigned char   sym_dest_name[8];    /* Symbolic destination name   */
        unsigned char   reserv1[2];          /* reserved                    */
        CPIC_SIDE_INFO_DEF_DATA  def_data;
} CPIC_SIDE_INFO_DATA;

typedef struct cpic_side_info
{
        unsigned char   partner_lu_name[17];
                                             /* Fully qualified partner     */
                                             /* LU name                     */
        unsigned char   reserved[3];         /* Reserved                    */
        unsigned long   tp_name_type;        /* TP name type                */
        unsigned char   tp_name[64];         /* TP name                     */
        unsigned char   mode_name[8];        /* Mode name                   */
        unsigned long   conversation_security_type;
                                             /* Conversation security type  */
        unsigned char   security_user_id[CPIC_SECURITY_INFO_LEN];
                                             /* User ID                     */
        unsigned char   security_password[CPIC_SECURITY_INFO_LEN];
                                             /* Password                    */
} CPIC_SIDE_INFO;

typedef struct cpic_side_info_def_data
{
        unsigned char   description[RD_LEN];
```

```
                                                /* resource description       */
        CPIC_SIDE_INFO  side_info;              /* CPIC side info             */
        unsigned char   user_data[32];          /* User defined data          */
} CPIC_SIDE_INFO_DEF_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_QUERY_CPIC_SIDE_INFO

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

buf_ptr

リスト情報を書き込むことができるバッファーへのポインター。

buf_size

提供されるバッファーのサイズ。 戻されるデータがこのサイズより大きくなることはありません。

num_entries

戻す項目の最大数。 項目数がこの値より大きくなることはありません。 値を 0 に指定すると、制限はなくなります。

list_options

リスト情報に何を戻すべきかを示します。 **sym_dest_name** での指定内容（下記を参照）は、戻される実際の情報の開始点を指定するために使用される索引値を示します。

AP_FIRST_IN_LIST

索引値は無視され、戻されるリストはリストの最初の項目から開始します。

AP_LIST_FROM_NEXT

戻されるリストは、提供された索引値によって指定されたリスト項目の次の項目から開始します。

AP_LIST_INCLUSIVE

戻されるリストは、索引値によって指定された項目から開始します。

sym_dest_name

サイド情報項目を示す記号宛先名。これは、最大 8 バイトの長さで、スペースで埋められ、ローカルで表示可能な文字セットにより指定されます。指定できる文字は、英字の大文字（A から Z）および数字の 0 から 9 までです。

戻りパラメーター

verb が正常に実行された場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_OK

buf_size
バッファーに戻される情報の長さ

total_buf_size
戻される値で、要求したすべてのリスト情報を戻すのに必要となる、バッファーのサイズを示します。 この値は、**buf_size** の値より大きくなることがあります。

num_entries
実際に戻される項目の数

total_num_entries
戻された可能性のある項目の数の合計。 この値は、**num_entries** の値より大きくなることがあります。

cpic_side_info_data.overlay_size
この項目のバイト数であり、したがって次に戻される項目（ある場合）へのオフセット。

cpic_side_info_data.sym_dest_name
戻されるサイド情報項目の記号宛先名。

cpic_side_info_data.def_data
DEFINE_CPIC_SIDE_INFO verb で指定されているものと同じ定義済み CPI-C サイド情報。

注: CPIC 呼出しは、DEFINE_CPIC_SIDE_INFO が Communications Server により処理された後で、この verb で戻されるサイド情報を変更する場合があります。.

状態エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_INVALID_SYM_DEST_NAME

ノードがまだ開始していないために verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_NOT_STARTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc
AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

第11章 接続マネージャー verb

Communications Server の接続マネージャーは、APPC プログラムまたは CPI-C プログラムの開始を管理するために使用します。 接続マネージャー機能についての説明は、*Communications Server クライアント/サーバー 通信プログラミング*に記載されています。

Communications Server ノード・オペレーター機能は、接続マネージャーの制御のために 3 つの verb をサポートしています。 Communications Server ノード・オペレーター機能を使用するアプリケーション・プログラムはすべてこれらの verb を使用することができます。

DISABLE_ATTACH_MANAGER

Communications Server の接続マネージャーは、ノードが開始されるときに省略時解釈により使用可能になります。 ユーザーはこの verb を発行してすべての動的ローディングを使用不能にすることができます。この verb は、接続マネージャーがトランザクション・プログラムを開始する前に検査する大域フラグをリセットします。

VCB 構造

```
typedef struct disable_am
{
        unsigned short  opcode;             /* Verb operation code     */
        unsigned char   reserv2;            /* reserved                */
        unsigned char   format;             /* format                  */
        unsigned short  primary_rc;         /* Primary return code     */
        unsigned long   secondary_rc;       /* Secondary return code   */
} DISABLE_AM;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_DISABLE_ATTACH_MGR

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行されると、接続マネージャーは以下のパラメーターを戻します。

primary_rc

AP_OK

ノードがまだ開始されていないために verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_NODE_NOT STARTED

システム・エラーが原因で verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

ENABLE_ATTACH_MANAGER

接続マネージャーが使用不能になっている場合、Communications Server ノード・オペレーター機能の verb である ENABLE_AM を発行することにより、接続マネージャーを再び使用可能にすることができます。 この verb は、接続マネージャーがトランザクション・プログラムを開始する前に検査する大域フラグを設定します。

VCB 構造

```
typedef struct enable_am
{
        unsigned short  opcode;          /* Verb operation code     */
        unsigned char   reserv2;         /* reserved                */
        unsigned char   format;          /* format                  */
        unsigned short  primary_rc;      /* Primary return code     */
        unsigned long   secondary_rc;    /* Secondary return code   */
} ENABLE_AM
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_ENABLE_ATTACH_MGR

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行されると、接続マネージャーは以下のパラメーターを戻します。

primary_rc

AP_OK

ノードがまだ開始されていないために verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_NODE_NOT_STARTED

システム・エラーが原因で verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

QUERY_ATTACH_MANAGER

QUERY_ATTACH_MANAGER verb を使用して、接続マネージャー構成要素の状況を発見することができます。接続マネージャー構成要素は、ENABLE_ATTACH_MANAGER コマンドおよび DISABLE_ATTACH_MANAGER コマンドを使用して開始あるいは停止されます。

VCB 構造

```
typedef struct query_am
{
        unsigned short  opcode;          /* Verb operation code          */
        unsigned char   reserv2;         /* reserved                     */
        unsigned char   format;          /* format                       */
        unsigned short  primary_rc;      /* primary return code          */
        unsigned long   secondary_rc;    /* secondary return code        */
        unsigned short   active;         /* status of the Attach Manager */
} QUERY_AM;
```

提供パラメーター

opcode

AP_QUERY_ATTACH_MGR

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

戻りパラメーター

verb が正常に実行されると、以下のパラメーターが戻されます。

primary_rc

AP_OK

active このフィールドは、接続マネージャー構成要素の状態を報告します。

AP_YES

接続マネージャーは活動状態にある。

AP_NO

接続マネージャーは活動状態にない。

パラメーター・エラーが原因で verb が実行されない場合、次のパラメーターが戻されます。

primary_rc

AP_PARAMETER_CHECK

ノードがまだ開始されていないために verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_NODE_NOT STARTED

システム・エラーが原因で verb が実行されない場合、接続マネージャーは次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

第2部 Communications Server 管理サービス API

第12章 管理サービス API について

この部では、Communications Server によって提供される管理サービス API について説明しています。

管理サービス verb

Communications Server は以下の管理サービス (MS) verb をサポートし、潜在的な問題を SNA ネットワークで使用可能な管理サービス・フォーカル・ポイントに報告するメソッドをアプリケーション・プログラムに提供します。

- ALERT_INDICATION
- FP_INDICATION
- MDS_MU_RECEIVED
- NMVT_RECEIVED
- SEND_MDS_MU
- TRANSFER_MS_DATA

エントリー・ポイント

Communications Server は、管理サービス verb を扱うライブラリー・ファイルを提供します。

管理サービス verb は、分かりやすい言語インターフェースをもっています。 プログラムは、まず *verb 制御ブロック*と呼ばれる記憶域ブロックのフィールドを埋めます。それから、エントリー・ポイントを呼び出し、verb 制御ブロックへのポインターを渡します。 この操作が完了すると、管理サービス (MS) API は、verb 制御ブロックのフィールドを使用し、修正してから、戻ります。 そこで、プログラムは、戻りパラメーターを verb 制御ブロックから参照することができます。 以下は管理サービス verb のエントリー・ポイントのリストです。

- WinMS()
- WinAsyncMS()
- WinAsyncMSEx()
- WinMSCancelAsyncRequest()
- WinMSCleanup()
- WinMSStartup()

エントリー・ポイントの詳細については、535ページの『第13章 管理サービスのエントリー・ポイント』を参照してください。

verb 制御ブロック (VCB)

プログラミング上の注意事項: 基本オペレーティング・システムは、呼出し側アプリケーションのアドレス空間でいくつかのサブシステムを実行することで、パフォーマンスを最適化します。 これは、完全にデバッグされていないかまたは正しくデバッグされていないアプリケーション・プログラムによりローカル記述子テーブル (LDT) セレクターが誤って使用されると、誤操作または場合によってはシステム障害の原因となることを意味します。したがって、アプリケーション・プログラムでは、ポインターの LDT セレクター・フィールドの変更を伴うようなポインターの算術演算を実行しないようにする必要があります。

verb 制御ブロック (VCB) に使用するセグメントは、読取り/書込みデータ・セグメントでなければなりません。 ユーザー・プログラムは、VCB をそのプログラム内の変数として宣言すること、VCB を割り振ること、またはより大きいセグメントから VCB を 2 次割振りすることができます。 このセグメントには、プログラムが発行する verb のすべてのフィールドを入れる十分な大きさが必要です。

アプリケーション・プログラムは、verb が発行されてから完了するまでは、verb 制御ブロック のどの部分も変更しません。 管理サービスは、あるverb の実行を完了すると、修正済みの完全な VCB を元のブロックに複写して戻します。 したがって、プログラムが変数として verb 制御ブロックを宣言する場合は、内部プロシージャーのスタック上ではなく、静的記憶域で宣言することを考慮してください。

各 VCB の中の予約済みで未使用のフィールドは、すべてゼロ (X'00') で埋めてください。実際には、プログラムが値をパラメーターに割り当てる前に、verb 制御ブロック全体をゼロに設定する方が時間効率がよい場合があります。 予約済みフィールドをゼロに設定しておくことは、特に重要です。

注: VCB が読取り/書込みでない場合、またはそれが 10 バイト (つまり管理サービス の 1 次戻りコードおよび 2 次戻りコードを収容するための十分な大きさ)より短い場合は、管理サービスはその VCB にアクセスできないため、基本オペレーティング・システムはプロセスを異常終了させます。 このような終了は、*一般保護障害*、プロセッサー例外トラップ D として認識されます。

管理サービスは、VCB が短すぎる場合、または間違ったタイプのセグメントが使用された場合、1 次戻りコード INVALID_VERB_SEGMENT を戻します。

管理サービス (MS) プログラムの作成

Communications Server は管理サービス verb を扱う動的リンク・ライブラリー (DLL)・ファイルを提供します。

この DLL は再入可能で、複数のアプリケーション・プロセスおよびスレッドが、DLL を同時に呼び出すことができます。

管理サービス verb は、分かりやすい言語インターフェースをもっています。 ユーザー・プログラムが、*verb 制御ブロック* (VCB) と呼ばれる記憶域にあるブロック内のフィールドを埋めます。 それから、WINMS DLL を呼び出し、verb 制御ブロックへのポインターを渡します。 この操作が完了すると、管理サービスは、VCB

制御ブロックのフィールドを使用し、修正してから、戻ります。 そこで、プログラムは、戻りパラメーターを verb 制御ブロックから参照することができます。

533ページの表 3 は、管理サービス・プログラムをコンパイルおよびリンクする際に必要な指定ヘッダー・ファイルおよびライブラリーのソース・モジュールの使用法を示しています。 ヘッダー・ファイルのいくつかは、その他の必須ヘッダー・ファイルを含んでいる場合があります。

表 3. 管理サービスのヘッダー・ファイルおよびライブラリー

オペレーティング・システム	ヘッダー・ファイル	ライブラリー	DLL 名
WINNT & WIN95	WINMS.H	WINMS32.LIB	WINMS32.DLL
WIN3.1	WINCSV.H	WINCSV.LIB	WINCSV.DLL
OS/2	ACSSVCC.H	ACSSVC.LIB	ACSSVC.DLL

SNA API クライアントによるサポート

SNA API クライアントは、すべての管理サービス verb のサブセットだけをサポートします。 特に、**WINMS** は、Windows クライアント (95, NT, 3.1) でサポートされている唯一の API です。 以下は、SNA API クライアントでサポートされている管理サービス verb のリストです。

- TRANSFER_MS_DATA
- SEND_MDS_MU

第13章 管理サービスのエントリー・ポイント

この章は、管理サービス verb のエントリー・ポイントについて説明します。

WinMS()

これは、以下の管理サービス API verb を発行するための同期エントリー・ポイントを提供します。

- SEND_MDS_MU
- TRANSFER_MS_DATA

構文

```
void WINAPI WinMS(long vcb, unsigned short vcb_size);
```

パラメーター

説明

vcb verb 制御ブロックに対するポインター

vcb_size
verb 制御ブロックに含まれるバイト数

戻り

戻り値なし。 verb 制御ブロック内の **primary_rc** フィールドおよび **secondary_rc** フィールドはエラーを示します。

注釈

これは管理サービス API の主要な同期エントリー・ポイントです。この呼出しは、verb が完了するまでブロック化されます。

WinMSCleanup()

この関数は、アプリケーションを終了させ、管理サービス API からアプリケーションの登録を取り消します。

構文

```
BOOL WINAPI WinMSCleanup(void);
```

戻り

戻り値は、登録解除が正常に実行されたかどうかを示します。値がゼロでない場合、アプリケーションの登録解除は正常に実行されました。値ゼロが戻された場合、アプリケーションは登録解除されませんでした。

注釈

管理サービス・アプリケーションの管理サービス API からの登録の取消しを指示するためには、**WinMSCleanup()** を使用します。

WinMSCleanup は、**WinMSGetIndication** に含まれる待機中のすべてのスレッドを非ブロック化します。 その結果、WMSNOTREG （アプリケーションは指示を受信するために登録されていない）が戻されます。 **WinMSCleanup** は、すべての指示に対するアプリケーションの登録を取り消します。 **WinMSCleanup** は、すべての未解決の verb（同期または非同期）に対して AP_CANCELLED エラーを戻します。ただし、verb はノード内では完了します。

WinMSStartup および **WinMSCleanup** を必ず使用する必要はありません。 ただし、アプリケーションでこれらの呼出しを使用する際には一貫性がなければなりません。 これらの関数については、両方とも使用するか、両方とも使用しないかにすべきです。

注: WinMSStartup() も合わせて参照してください。

WinMSStartup()

この関数は、アプリケーションが必要な管理サービス API のバージョンを指定したり、その製品がサポートしている API のバージョンを検索することができるようにします。 アプリケーション自身の登録のための管理サービス API 呼出しを発行する前であれば、アプリケーションはこの関数を呼び出すことができます。

構文

```
int WINAPI WinMSStartup(WORD wVersionRequired,
                        LPWMSDATA msdata);
```

パラメーター

説明

wVersionRequired
必要な管理サービス API サポートのバージョンを指定します。高位バイトは、マイナー・バージョン（改訂版）番号を指定し、低位バイトは、メジャー・バージョン番号を指定します。

msdata
管理サービス API のバージョンおよび実施管理サービスの説明を戻します。

戻り

戻り値は、アプリケーションが正常に登録されたかどうか、また、実施管理サービス API が指定するバージョン番号をサポートしているかどうかを戻します。 この値がゼロである場合、アプリケーションは正常に登録されており、指定されたバージョンをサポートすることができます。 ゼロ以外の戻り値は、以下の値の 1 つを戻します。

WMSSYSERROR
基礎ネットワーク・サブシステムにおけるネットワーク通信が作動不能。

WMSVERNOTSUPPORTED
要求された管理サービス API サポートのバージョンは、この特定の管理サービス API では提供されていません。

WMSBADPOINTER
msdata パラメーターの誤り。

注釈

WinMSStartup は、API の今後のバージョンと互換性を保つためのものです。サポートされている現行バージョンは 1.0 です。

WinMSStartup および **WinMSCleanup** を必ず使用する必要はありません。 ただし、アプリケーションでこれらの呼出しを使用する際には一貫性がなければなりません。 これらの関数については、両方とも使用するか、両方とも使用しないかにすべきです。

注: WinMSCleanup() も合わせて参照してください。

WinMSRegisterApplication()

この関数は、アプリケーションを NMVT レベルのアプリケーション、MDS レベルのアプリケーション、またはアラート・ハンドラーとして登録します。 この登録は、アプリケーションが受信する非送信請求指示を判別します。

- NMVT レベルのアプリケーションは NMVT_RECEIVED 指示を受信します。
- MDS レベルのアプリケーションは MDS_MU_RECEIVED 指示、および、フォーカル・ポイントの状況が変更されたときは FP_NOTIFICATION 指示も受信します。
- アラート・ハンドラーは ALERT_INDICATION 指示を受信します。

注: MDS MU に対する会話で NMVT 指示を受信することができるように登録することも可能です。

これらの指示を処理しないアプリケーションは、**WinMSRegisterApplication** を呼び出してはなりません。

構文

```
BOOL WINAPI WinMSRegisterApplication(unsigned short reg_type,
                                     unsigned char  *ms_appl_name,
                                     unsigned short vector_key,
                                     unsigned char  mds_conv_reqd,
                                     unsigned char  *ms_category,
                                     unsigned short max_rcv_size,
                                     unsigned char  alert_dest,
                                     unsigned short *primary_rc,
                                     unsigned long  *secondary_rc);
```

パラメーター

説明

reg_type

登録のタイプ

WMSNMVTAPP	NMVT レベルのアプリケーション（または NMVT を受信するための MDS レベルの登録アプリケーション）
WMSMDSAPP	MDS レベルのアプリケーション
WMSALERTHANDLER	アラート・ハンドラー

ms_appl_name

管理サービス・アプリケーション名。有効な名前は、8 バイトの英数字で、タイプ 1134 の EBCDIC 文字（必要であれば、後書きスペース (X'40') 文字で埋められる）、あるいは、 付録 D の *SNA 管理サービスの参照事項*で指定している管理サービス規律指定のアプリケーション・プログラムの 1 つ（後書きスペース (X'40') 文字で埋められる）です。

この名前は、**reg_type** が WMSNMVTAPP または WMSMDSAPP である場合に使用されます。この名前は、**reg_type** が WMSALERTHANDLER である場合は使用されません。

vector_key

アプリケーションが受信する管理サービス主ベクトル・キーです。許可される値は、以下のとおりです。

X'YYYY'	特定の主ベクトル・キー
AP_SPCF_KEYS	X'8061' から X'8064'までの主ベクトル・キー
AP_ALL_KEYS	すべての主ベクトル・キー

このキーは、**reg_type** が WMSNMVTAPP である場合に使用されます。このキーは、**reg_type** が WMSMDSAPP または WMSALERTHANDLER である場合は使用されません。

mds_conv_reqd

登録アプリケーションが MDS レベルのアプリケーションであるかどうか、また、その登録アプリケーションに送信する NMVT を MDS MU に変換する必要があるかどうかを指定します。

（AP_YES または AP_NO）

このパラメーターは、**reg_type** が WMSNMVTAPP である場合に使用されます。このパラメーターは、**reg_type** が WMSMDSAPP または WMSALERTHANDLER である場合は使用されません。

ms_category

アプリケーションが管理サービス・カテゴリー用のフォーカル・ポイントに関する情報を必要とする場合に、その管理サービス・カテゴリーを指定します。管理サービス・カテゴリーは、付録 D の*SNA 管理サービスの参照事項*で記載している管理サービス規律指定のアプリケーション・プログラム・テーブルで指定されているカテゴリー・コードの 1 つ（後書きスペース (X'40') 文字で埋められる）、あるいは、ユーザー定義のカテゴリーのいずれかです。ユーザー定義のカテゴリー名は、8 バイトの英数字で、タイプ 1134 の EBCDIC 文字（必要であれば、後書きスペース (X'40') 文字で埋められる）でなければなりません。

このパラメーターは、**reg_type** が WMSMDSAPP である場合に使用されます。このパラメーターは、**reg_type** が WMSNMVTAPP または WMSALERTHANDLER である場合は使用されません。

max_rcv_size

アプリケーションが一度に受信できる最大バイト数。このサイズより大きい MDS MU はセグメントに分けられ、各セグメントは別々の MDS_MU_RECEIVED 指示で送達されます。

このパラメーターは、**reg_type** が WMSMDSAPP である場合に使用されます。このパラメーターは、**reg_type** が WMSNMVTAPP または WMSALERTHANDLER である場合は使用されません。

alert_dest

すべてのアラートの宛先をこのアプリケーションだけにするかどうかを指定します。 これを AP_YES に設定した場合、すべてのアラートはこのアプリケーションに経路指定され、このアプリケーション以外には経路指定されま

せん。AP_NO に設定した場合、アラートは通常の方法で SNA ネットワークを介してアプリケーションに経路指定されます。

このパラメーターは、**reg_type** が WMSALERTHANDLER である場合に使用されます。このパラメーターは、**reg_type** が WMSNMVTAPP または WMSMDSAPP である場合は使用されません。

primary_rc
戻り値 : 1 次戻りコード

secondary_rc
戻り値 : 2 次戻りコード

戻り

この関数は、登録が正常に実行されたかどうかを示す値を戻します。 値がゼロでない場合、アプリケーションの登録は正常に実行されました。 値がゼロである場合、アプリケーションの登録は正常に実行されませんでした。

注釈

アプリケーションは、1 つ以上の指示のクラスを登録するために複数の呼出しを実行することができます。

WinMSRegisterApplication を呼び出すアプリケーションは、それらのアプリケーション用に待ち行列に入れられた指示を受信するために、**WinMSGetIndication** を呼び出す必要があります。

注: **WinMSUnregisterApplication** および **WinMSGetIndication** も合わせて参照してください。

WinMSUnregisterApplication()

この関数は、以前の **WinMSRegisterApplication** 呼出しの効果を元に戻し、そのアプリケーションに対する後続の指示が待機することを停止させることにより、アプリケーションの登録を取り消します。

構文

```
BOOL WINAPI WinMSUnregisterApplication(unsigned short reg_type,
                                       unsigned char  *ms_appl_name,
                                       unsigned short *primary_rc,
                                       unsigned long  *secondary_rc);
```

パラメーター

説明

reg_type

登録のタイプ以下の値のいずれか 1 つになります。

WMSNMVTAPP

NMVT レベルのアプリケーション

WMSMDSAPP

MDS レベルのアプリケーション

WMSALERTHANDLER

アラート・ハンドラー

ms_appl_name

MS アプリケーション名。 有効な名前は、8 バイトの英数字で、タイプ 1134 の EBCDIC 文字（必要であれば、後書きスペース (X'40') 文字で埋められる）、あるいは、 付録 D の *SNA 管理サービスの参照事項*で指定している管理サービス規律指定のアプリケーション・プログラムの 1 つ（後書きスペース (X'40') 文字で埋められる）です。

このパラメーターは、**reg_type** が WMSNMVTAPP または WMSMDSAPPである場合に使用されます。 このパラメーターは、**reg_type** が WMSALERTHANDLER である場合は使用されません。

primary_rc

戻り値 : 1 次戻りコード

secondary_rc

戻り値 : 2 次戻りコード

戻り

この関数は、登録の取消しが成功したかどうかを示す値を戻します。 値がゼロでない場合、登録抹消は正常に実行されました。 値がゼロである場合、登録抹消は正常に実行されませんでした。

注釈

WinMSUnregisterApplication の各呼出しは、それより前の **WinMSRegisterApplication** の呼出しにより実行れた登録を終了させます。 **WinMSRegisterApplication** の複数の呼出しを実行したアプリケーションは、自己の登録を終了させるために **WinMSUnregisterApplication** の複数の呼出しを実行する必要があります。

WinMSUnregisterApplication と **WinMSCleanup** は、以下の点で異なります。

- **WinMSUnregisterApplication** は、指示を受信するための以前の登録を終了させますが、アプリケーションは別の管理サービス API 呼出しを実行することができます（たとえば、WinMS など）。
- **WinMSCleanup** は、管理サービス API の使用を終了させます。

アプリケーションが **WinMSUnregisterApplication** を呼び出したとき、そのアプリケーションに対して既に待ち行列化されている指示が存在する場合があります。 そのような指示は、待ち行列に入ったままなので、それらの指示を受信して処理するためには、アプリケーションは **WinMSGetIndication** を呼び出す必要があります。 アプリケーションの登録の取消しが完了すると、新しい指示がそのアプリケーションに対して待ち行列化することはそれ以後ありません。

注: **WinMSRegisterApplication** および **WinMSGetIndication** も合わせて参照してください。

WinMSGetIndication()

これにより、アプリケーションは非送信請求指示を受け取ることができます。

構文

```
int WINAPI WinMSGetIndication(long buffer,
                              unsigned short *buffer_size,
                              unsigned long  timeout);
```

パラメーター

説明

buffer 指示を受信するバッファーに対するポインター。

buffer_size
バッファーのサイズ。 戻されるもの : 指示のサイズ。

timeout
指示待ち時間（ミリ秒単位)。

戻り

この関数は、指示が受信されたかどうかを示す値を戻します。

0 指示が戻されました。

WMSTIMEOUT
指示待ち時間のタイムアウト。

WMSSYSNOTREADY
基礎ネットワーク・サブシステムにおけるネットワーク通信が作動不能。

WMSNOTREG
アプリケーションが指示を受信するように登録されていません。

WMSBADSIZE
バッファーが小さすぎるため、指示を受け取ることができません。十分な大きさのバッファーを用意して、**WinMSGetIndication** 呼出しを再発行してください。指示のサイズは、**buffer_size** パラメーターに戻されます。

WMSBADPOINTER
buffer パラメーターもしくは **buffer_size** パラメーターのいずれかが無効です。

WMSSYSERROR
予期しないシステム・エラーが発生しました。

注釈

これはブロッキング呼出しで、以下の状況のいずれか 1 つになります。

- 指示が戻された場合
- タイムアウトが満了した場合
- アプリケーションが **WinMSCleanup** 呼出しを発行した場合

- プロダクトが停止した場合
- システム・エラーが発生した場合

注: **WinMSRegisterApplication** および **WinMSUnregisterApplication** も合わせて参照してください。

第14章 管理サービス verb

Communications Server によって提供される管理サービス API は、アプリケーションがアラートおよび MDS MU を送信することを可能にし、ノードが MDS または NMVT データを受信したとき、あるいはアラートを発行したときに指示を受信することを可能にします。

TRANSFER_MS_DATA

この verb は、NMVT レベルのアプリケーションが非送信請求アラートを送信し、以前に受信した NMVT 要求に応答するために使用されます。

TRANSFER_MS_DATA はまた、MDS レベルのアプリケーションが非送信請求アラートを送信するためにも使用されます。この verb は、WinMS 呼出しを使用するアプリケーションで使用することができます。

VCB 構造

```
typedef struct ms_transfer_ms_data
{
        unsigned short  opcode;             /* Verb operation code         */
        unsigned char    data_type;         /* Data type supplied by app   */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* Primary return code         */
        unsigned long   secondary_rc;       /* Secondary return code       */
        unsigned char    options;           /* Verb options                */
        unsigned char   reserv3;            /* reserved                    */
        unsigned char    originator_id[8];  /* Originator ID               */
        unsigned char    pu_name[8];        /* Physical unit name          */
        unsigned char    reserv4[4];        /* reserved                    */
        unsigned short   dlen;              /* Length of data              */
        unsigned char    *dptr;             /* Data                        */
} MS_TRANSFER_MS_DATA;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

SV_TRANSFER_MS_DATA

data_type

格納するデータのタイプを指定します。 管理サービスは、以下に記載するとおりにデータを処理します。 許可される値は以下のとおりです。

SV_NMVT

このデータは、完全な NMVT 要求単位を含みます。 このデータがアラートを含み、そのアラートが MDS レベルあるいは移行レベルのフォーカル・ポイントに送信するものである場合、管理サービスはこのデータを MDS_MU 形式または CP_MSU 形式に変換します。これは、アプリケーションが NMVT_RECEIVED シグナルに応答するときに指定する必要があるタイプです。

SV_ALERT_SUBVECTORS

このデータは、アラート主ベクトルのための SNA 定義形式で、管理サービス・サブベクトルを含みます。 管理サービスは、NMVT ヘッダーおよびアラート主ベクトル・ヘッダーを追加します。 続いて、アラートが MDS レベルあるいは移行レベルのフォーカル・ポ

イントに送信するものである場合、管理サービスはこのデータを MDS_MU 形式または CP_MSU 形式に変換します。

SV_USER_DEFINED
このデータは、完全な NMVT 要求単位を含みます。 管理サービスは常にこのデータをログしますが、送信しません。

SV_PDSTATS_SUBVECTORS
このデータは問題判別統計を含みます。 管理サービスは常にこのデータをログし、アラート・ハンドラーが登録されている場合、ALERT_INDICATION に含まれるデータを送信します。

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

options
この verb に指定されているデータに対する任意選択の処理を指定します。指定された **data_type** とオプションの間に矛盾がある場合、管理サービスは**タイプ**を優先してデータを処理します。 このパラメーターは 1 バイトの値で、個々のビットの設定値は選択したオプションを示します。すべてのオプションを指定する場合は、このバイトをゼロに設定します。

ビット 0 が最大有効ビットで、ビット 7 が最小有効ビットです。

（**data_type** に SV_USER_DEFINED を指定した場合、ビット 1 から 3 は無視されます。）

ビット 0: ゼロに設定した場合、データに日付/時刻 (X'01') サブベクトルを追加します。

ビット 1: ゼロに設定した場合、データに製品セット ID (X'10') サブベクトルを追加します。 アプリケーションが既に製品セット ID サブベクトルを含むデータを指定した場合、管理サービスは、既存の製品セット ID サブベクトルの直前に Communications Server の製品セット ID サブベクトルを追加します。

ビット 2: ゼロに設定した場合、SNA セッションを介してデータを送信します。 データがアラートを含まない場合、管理サービスは省略時の SSCP-PU セッションを介してデータを送信します。 データがアラートを含む場合、アラートをアラート・フォーカル・ポイントへ送信するのに Communications Server がどのタイプのセッションを使用するかによって、管理サービスは SSCP-PU セッション、CP-CP セッション、あるいは LU-LU セッションのいずれかを介してデータを送信します。

ビット 3: ゼロに設定した場合、Communications Server の問題判別機能によりデータをログします。

注: 以下の定数は管理サービスのヘッダー・ファイルで提供されており、それぞれ上記で指定した個々のビットを参照します。

SV_TIME_STAMP_SUBVECTOR
SV_PRODUCT_SET_ID_SUBVECTOR
SV_SEND_ON_SESSION
SV_LOCAL_LOGGING

ビット 4 から 7 は予約済みです。

originator_id
verb を発行した構成要素の名前。 ローカルで表示可能な文字セットで 8 バイト・ストリングです。 このフィールドは、管理サービスが TRANSFER_MS_DATA をロギングするときのみ使用します。

pu_name
データを送信する宛先の物理装置の名前を指定します。 これは、8 バイトの英数字で、タイプ A の EBCDIC ストリング（右側は EBCDIC スペースで埋められる）で設定するか、あるいはどの **pu_name** も指定しない場合はすべて 2 進ゼロで設定する必要があります。 NMVT_RECEIVED シグナルに応答するために TRANSFER_MS_DATA を使用するアプリケーションは、NMVT_RECEIVED シグナルで受信する **pu_name** を指定する必要があります。 **pu_name** を指定しないタイプ SV_NMVT の TRANSFER_MS_DATA シグナルに含まれるデータは、省略時値の PU セッション（使用可能である場合）を介して送信されます。 アプリケーションがアラート・データを特定の PU に送信することを明示的に指示していない限り、アラートを含む TRANSFER_MS_DATA シグナルが **pu_name** を指定してはなりません。 これは、管理サービスの正常なアラート経路指定アルゴリズムを迂回します。

dlen データの長さ。

dptr データに対するポインター。 これを NULL に設定した場合、管理サービスはデータを VCB に連続するもの（VCB の直後に続く）と見なします。

戻りパラメーター

verb が正常に実行されると、管理サービスは以下のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されない場合、管理サービスは以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
SV_INVALID_DATA_TYPE

SV_DATA_EXCEEDS_RU_SIZE
AP_INVALID_PU_NAME

状態エラーが原因で verb が実行されない場合、管理サービスは以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc

SV_SSCP_PU_SESSION_NOT_ACTIVE

システム・エラーが原因で verb が実行されない場合、Communications Server APPN は以下のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

MDS_MU_RECEIVED

この指示 verb は、以下の場合に管理サービスにより登録済み MDS レベルのアプリケーションに送信されます。

- 対等 MDS レベルのアプリケーションからの MDS_MU を受信した場合
- NMVT を受信し、
  - 適切な NMVT レベルのアプリケーションがまだ登録されていない場合。
  - MDS レベルのアプリケーションが、着信 NMVT（管理サービスは NMVT から MDS_MU への会話を実行します）の管理サービス主ベクトル・キーに含まれる名前と対応する名前で登録されている場合。

VCB 構造

```
typedef struct ms_mds_mu_received
{
        unsigned short  opcode;            /* Verb operation code           */
        unsigned char   reserv2;           /* reserved                      */
        unsigned char   format;            /* format                        */
        unsigned short  primary_rc;        /* Primary return code           */
        unsigned long   secondary_rc;      /* Secondary return code         */
        unsigned char    first_message;    /* First message for curr MDS_MU */
        unsigned char    last_message;     /* Last message for curr MDS_MU  */
        unsigned char    pu_name[8];       /* Physical unit name            */
        unsigned char   reserv3[8];        /* reserved                      */
        unsigned short   mds_mu_length;    /* Length of incoming MDS_MU     */
        unsigned char    *mds_mu;          /* MDS_MU data                   */
} MS_MDS_MU_RECEIVED;
```

提供パラメーター

opcode

AP_MDS_MU_RECEIVED

形式 VCB の形式を示します。 上記に記載した VCB のバージョンを指定するには、このフィールドをゼロに設定します。

first_message

MDS_MU に対する最初のメッセージであるかどうかを示すフラグ（AP_YES または AP_NO）。 **WinMSRegisterApplication** 呼出しで指定された **max_rcv_size** が、送達された MDS_MU の長さよりも小さい場合、MDS_MU はまとめてアプリケーションに送信されます。

last_message

MDS_MU に対する最後のメッセージであるかどうかを示すフラグ（AP_YES または AP_NO）。

pu_name

NMVT（MDS_MU に変換されたもの）の送信元である物理装置の名前。アプリケーションは、着信 NMVT に応答する必要があります。 アプリケーションは、応答を送信するために SEND_MDS_MU を使用します。アプリケーションが応答を送信する場合、SEND_MDS_MU の **pu_name** フィ

ールドを MDS_MU_RECEIVED シグナルで指定されている **pu_name** と同じに設定する必要があります。 MDS_MU が MDS レベルのトランスポート・メカニズムから受信された場合、**pu_name** はすべて 2 進ゼロで設定されます。

mds_mu_length
シグナルに含まれる MDS_MU 部分の長さ。

mds_mu
MDS_MU データ。 データ・ポインターは NULL に設定され、データは VCB に連続します（VCB の直後に続く）。

SEND_MDS_MU

この verb は、MDS レベルのアプリケーションが、**WinMS** エントリー・ポイントを使用してアラート以外のネットワーク管理データを送信するために使用します。宛先アプリケーションに MDS_MU を送信しているときにエラーが起こった場合、そのエラーはいずれかの方法を使用して起点アプリケーションに報告されます。 ローカル・ノードでエラーが検出された場合、アプリケーションは SEND_MDS_MU 応答の戻りコードにより通知されます。 リモート・ノードでエラーが検出された場合、エラーは MDS_MU_RECEIVED VCB に格納して移送されるエラー MDS_MU を通じて報告されます。 発信 MDS_MU が SSCP-PU セッションを介して宛先ノードに到達する場合、管理サービスは発信 MDS_MU を NMVT に変換することができます。 アプリケーションは、ローカル・ノードの識別を認識している必要はありません。 MDS 経路指定情報 GDS 変数の起点ロケーション名サブベクトルの **netid** サブフィールドまたは **nau** サブフィールド、あるいは両方のサブフィールドに、アプリケーションが 8 つの EBCDIC ブランクを指定した場合、Communications Server は適切な値を指定します。アプリケーションが **netid** および **nau** のいずれのサブフィールドも指定せず、8 より少ないブランクを指定した場合、Communications Server は AP_INVALID_MDS_MU_FORMAT の 2 次戻りコードを戻します。

VCB 構造

```
typedef struct ms_send_mds_mu
{
        unsigned short  opcode;             /* Verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* Primary return code         */
        unsigned long   secondary_rc;       /* Secondary return code       */
        unsigned char   options;            /* Verb options                */
        unsigned char   reserv3;            /* reserved                    */
        unsigned char   originator_id[8];   /* Originator ID               */
        unsigned char   pu_name[8];         /* Physical unit name          */
        unsigned char   reserv4[4];         /* reserved                    */
        unsigned short  dlen;               /* Length of data              */
        unsigned char   *dptr;              /* Data                        */
} MS_SEND_MDS_MU;
```

提供パラメーター

opcode

AP_SEND_MDS_MU

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

options

この verb に指定されているデータに対する任意選択の処理を指定します。このパラメーターは 1 バイトの値で、個々のビットの設定値は選択したオプションを示します。 すべてのオプションを指定する場合は、このバイトをゼロに設定します。

ビット 0 が最大有効ビットで、ビット 7 が最小有効ビットです。

ビット 0: ゼロに設定した場合、データに日付/時刻 (X'01') サブベクトルを追加します。

ビット 1: ゼロに設定した場合、データに製品セット ID (X'10') サブベクトルを追加します。 アプリケーションが既に製品セット ID サブベクトルを含むデータを指定した場合、管理サービスは、既存の製品セット ID サブベクトルの直前に Communications Server の製品セット ID サブベクトルを追加します。

ビット 2 は予約済みです。

ビット 3: ゼロに設定した場合、Communications Server の問題判別機能によりデータをログします。

注: 上記で指定したビット 0、1、および 3 を参照する以下の定数は、管理サービスのヘッダー・ファイルで提供されています。

SV_TIME_STAMP_SUBVECTOR
SV_PRODUCT_SET_ID_SUBVECTOR
SV_LOCAL_LOGGING

ビット 4: 管理サービスが、管理サービス・データを宛先アプリケーションに送信するために省略時の経路指定を使用するか、直接経路指定を使用するかを指定します（AP_DEFAULT または AP_DIRECT）。

注: ビット 4 を設定するには、適切に桁送りした AP_DEFAULT または AP_DIRECT を使用します（たとえば、AP_DIRECT<<3）。

ビット 5 から 7 は予約済みです。

originator_id
verb を発行した構成要素の名前。このフィールドは、管理サービスが SEND_MDS_MU をロギングするときのみ使用します。

pu_name
データを送信する宛先の物理装置の名前を指定します。 これは、8 バイトの英数字で、タイプ A の EBCDIC ストリング（右側は EBCDIC スペースで埋められる）で設定するか、あるいはどの **pu_name** も指定しない場合はすべて 2 進ゼロで設定する必要があります。 着信 NMVT から変換された MDS_MU_RECEIVED 指示に応答するために SEND_MDS_MU を使用するアプリケーションは、MDS_MU_RECEIVED シグナルで受信した **pu_name** を指定する必要があります。 MDS トランスポート機能を使用して移送する MDS_MU は、**pu_name** をすべて 2 進ゼロに設定する必要があります。

dlen データの長さ。

dptr データに対するポインター。 これを NULL に設定した場合、管理サービスはデータを VCB に連続するもの（VCB の直後に続く）と見なします。

戻りパラメーター

verb が正常に実行されると、Communications Server の管理サービスは以下のパラメーターを戻します。

primary_rc
AP_OK

パラメーター・エラーが原因で verb が実行されない場合、Communications Server の管理サービスは以下のパラメーターを戻します。

primary_rc
AP_PARAMETER_CHECK

secondary_rc
AP_INVALID_PU_NAME

AP_INVALID_MDS_MU_FORMAT
SV_INVALID_DATA_SIZE

状態エラーが原因で verb が実行されない場合、Communications Server の管理サービスは以下のパラメーターを戻します。

primary_rc
AP_STATE_CHECK

secondary_rc
AP_SSCP_PU_SESSION_NOT_ACTIVE

システム・エラーが原因で verb が実行されない場合、Communications Server APPN は以下のパラメーターを戻します。

primary_rc
AP_UNEXPECTED_SYSTEM_ERROR

ALERT_INDICATION

この指示 verb は、アラート主ベクトルを処理する、登録済みアラート・ハンドラーまたは登録済み保留アラート・ハンドラーに、管理サービスがアラート主ベクトルを送信するために使用します。

VCB 構造

```
typedef struct ms_alert_indication
{
        unsigned short  opcode;             /* AP_AlERT_INDICATION        */
        unsigned char   reserv2;            /* reserved                   */
        unsigned char   format;             /* format                     */
        unsigned short  primary_rc;         /* Primary return code        */
        unsigned long   secondary_rc;       /* Secondary return code      */
        unsigned short  alert_length;       /* Length of alert            */
        unsigned char   reserv3[6];         /* reserved                   */
        unsigned char   *alert;             /* Alert data                 */
} MS_ALERT_INDICATION;
```

提供パラメーター

opcode

AP_ALERT_INDICATION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

alert_length

アラート・データの長さ。

alert アラート・データに対するポインター。 データ・ポインターは NULL に設定され、データは VCB に連続します（VCB の直後に続く）。

FP_NOTIFICATION

MDS レベルのアプリケーションが特定の管理サービス・カテゴリーで登録されており、そのカテゴリーのフォーカル・ポイントの状況が変更された場合、管理サービスはこの verb シグナルをアプリケーションに送信します。

VCB 構造

```
typedef struct ms_fp_notification
{
        unsigned short  opcode;             /* Verb operation code          */
        unsigned char   reserv2;            /* reserved                     */
        unsigned char   format;             /* format                       */
        unsigned short  primary_rc;         /* Primary return code          */
        unsigned long   secondary_rc;       /* Secondary return code        */
        unsigned char   fp_routing;         /* Type of routing to focal pt  */
        unsigned char   reserv1;            /* reserved                     */
        unsigned short  fp_data_length;     /* Length of incoming focal     */
                                            /* point data                   */
        unsigned char   *fp_data;           /* focal point data             */
} MS_FP_NOTIFICATION;
```

提供パラメーター

opcode

AP_FP_NOTIFICATION

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

fp_routing

フォーカル・ポイントにメッセージを送信するときに SEND_MDS_MU で指定する必要がある経路指定のタイプ（AP_DEFAULT または AP_DIRECT）。

fp_data_length

フォーカル・ポイント・データの長さ。

fp_data

フォーカル・ポイント通知 (X'E1') サブベクトルおよびフォーカル・ポイント識別 (X'21') サブベクトルを含むフォーカル・ポイント・データ。 このデータ・ポインターは NULL に設定され、データは VCB に連続します（VCB の直後に続く）。

NMVT_RECEIVED

この verb シグナルは、リモート・ノードから NMVT を受信したときに、管理サービスにより NMVT レベルのアプリケーションに送信されます。

管理サービスは、着信 NMVT の経路指定に以下の規則を適用します。

1. 着信 NMVT で運ばれる主ベクトル・キーで登録されている NMVT レベルのアプリケーションに経路指定することを試行する、さもなければ、
2. 主ベクトル・キーが X'8061' から X'8064' のいずれかである場合、登録済み NMVT レベルの AP_SPCF_KEYS プリケーションに経路指定することを試行する、さもなければ、
3. NMVT レベルの登録済み AP_ALL_KEYS アプリケーションに経路指定することを試行する、さもなければ、
4. NMVT （MDS_MU に変換後）を、着信 NMVT で運ばれる主ベクトル・キーで登録されている MDS レベルのアプリケーションに経路指定することを試行する、さもなければ、
5. 主ベクトル・キーが X'8061' から X'8064' のいずれかである場合、NMVT（MDS_MU に変換後）を登録済み MDS レベルのアプリケーションに経路指定することを試行する、さもなければ、
6. （MDS_MU に変換後）登録済み AP_ALL_KEYS の MDS レベルのアプリケーションに経路指定することを試行する、さもなければ、
7. NMVT に対して否定応答する。

VCB 構造

```
typedef struct ms_nmvt_received
{
        unsigned short  opcode;             /* Verb operation code         */
        unsigned char   reserv2;            /* reserved                    */
        unsigned char   format;             /* format                      */
        unsigned short  primary_rc;         /* Primary return code         */
        unsigned long   secondary_rc;       /* Secondary return code       */
        unsigned char   pu_name[8];         /* Physical unit name          */
        unsigned char   reserv3[6];         /* reserved                    */
        unsigned short  nmvt_length;        /* Length of incoming NMVT     */
        unsigned char   *nmvt;              /* NMVT data                   */
} MS_NMVT_RECEIVED;
```

提供パラメーター

opcode

AP_NMVT_RECEIVED

形式 VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

pu_name

NMVT の送信元の物理装置の名前。 アプリケーションは、着信 NMVT に応答する必要があります。 アプリケーションは、応答を送信するために

TRANSFER_MS_DATA を使用します。 アプリケーションが応答を送信する場合、TRANSFER_MS_DATA の **pu_name**フィールドを NMVT_RECEIVED シグナルで指定されている **pu_name** と同じに設定する必要があります。

nmvt_length
NMVT データの長さ。

nmvt REGISTER_NMVT_APPLICATION で指定されているタイプの管理サービス主ベクトルを含む完全 NMVT。このデータ・ポインターは NULL に設定され、データは VCB に連続します（VCB の直後に続く）。

第3部 Communications Server ASCII 構成

第15章 ASCII 構成について

この章では、Communications Server によって提供される ASCII 構成について説明します。 ASCII 構成は、構成情報の作成、保管、およびアクセスの新しい方法を提供します。この方法では、構成レコードを保管するのに、バイナリー・ファイルではなく ASCII ファイルを使用します。この方法では、ユーザーは SNA ノード構成ユーザー・インターフェースを使用せずに、構成ファイルの作成および修正をします。

Keywords

Communications Server は、特定のノードを構成するために、すべての構成データを含む ASCII (ACG) ファイルを提供します。 構成データは、複合キーワードと単純キーワードとして示されます。 単純キーワードは keyword=value 形式の構成パラメーターです。複合キーワードは単純キーワードをグループ化したものです。 以下は、Communications Server によってサポートされているキーワードのリストです。

- ADJACENT_NODE
- ANYNET_COMMON_PARAMETERS
- ANYNET_SOCKETS_OVER_SNA
- CONNECTION_NETWORK
- CPIC_SIDE_INFO
- DLUR_DEFAULTS
- DOWNSTREAM_LU
- DOWNSTREAM_PU_TEMPLATE
- FOCAL_POINT
- INTERNAL_PU
- LINK_STATION
- LOCAL_LU
- LU_0_TO_3
- LU_LU_PASSWORD
- MODE
- NODE
- PARTNER_LU
- PORT
- SPLIT_STACK
- TN3270E_DEF
- TP
- USERID_PASSWORD
- VERIFY

各 verb の詳しい説明については、33ページの『第4章 ノード構成 verb』 を参照してください。

ASCII 構成の検査ユーティリティー

ASCII 構成の検査ユーティリティーは、ユーザーの構成ファイルに全くエラーがないようにするために、ユーザーの構成ファイルを検査をします。 エラーがある場合、Communications Server 構成 GUI を介さずにファイルを編集しなくてはなりません。

構成ファイルの検査

Communications Server は構成ファイルを検査する 2 つのユーティリティーを提供します。

- コンソール検査（コマンド行）ユーティリティー
- Windows SDI 検査ユーティリティー

コンソール検査

コンソール検査方法は、Windows DOS コンソール・アプリケーションとして実行されます。 この方法を開始するには、以下のコマンド行構文を発行します。

```
vacgcon <filename>
```

この <filename> は **.ACG** ファイルです。

検査が実行され、検査が正常に実行されたかどうかを示すメッセージが生成されます。 メッセージとエラーは DOS コンソール画面に書き込まれます。 コンソールにエラー・メッセージが書き込まれていると、スクロールすることができません。コマンド行ユーティリティーからの出力はファイルに宛先変更することができます。

Windows SDI 検査

Windows SDI 検証ユーティリティーは、Windows SDI アプリケーションとして実行されます。 Communications Server フォルダー内の Communications Server の Verification アイコンを選択することによって、あるいは次のコマンド行構文を発行することによって、このユーティリティーを開始することができます。

```
vacgwin <filename>
```

この <filename> は **.ACG** ファイルです。

コマンド・オプションを使用すると、ファイルは自動的にオープンされ、検査されます。アイコンを選択する場合は、ウィンドウ・メニューまたはツール・バー機能を使用してファイルを検査します。 以下のとおりです。

1. 構成ファイルを選択し、オープンする。
2. ファイルを検査する。
3. エラーとメッセージが表示される。

構成ファイルの編集

検査ユーティリティー（コンソール検査でも Windows SDI 検査でも）がエラーを生成した場合、任意の ASCII テキスト・エディターを使用して **.ACG** 構成ファイルを編集します。 以下のとおり、構成ファイルを編集します。

- Windows メニューからの場合
  1. **File** を選択する。
  2. **Edit a file** を選択する。
  3. 選択された構成ファイル名を指定して ASCII エディターを開始する。
  4. 必要に応じてファイルを編集する。
  5. ファイルを **Save** する。
  6. ファイルを **Re-verify** する。
- ツール・バーからの場合
  1. **Edit** アイコン（鉛筆）を選択する。
  2. 選択された構成ファイル名を指定して ASCII エディターを開始する。
  3. 必要に応じてファイルを編集する。
  4. ファイルを **Save** する。
  5. ファイルを **Re-verify** する。

Windows SDI アプリケーションのメニューとツール・バーでの選択の使用方法に関する詳細については、オンライン・ヘルプを参照してください。

第16章 ASCII 構成キーワード

この章には、以下のことが示してあります。

- キーワードの種類とタイプの説明
- キーワードの形式
- キーワードの例

キーワードの種類とタイプの説明

ASCII 構成ファイルのデータの読取り方および解釈の仕方を理解する手助けとして、以下でキーワードの種類とタイプを説明します。

キーワードの種類

キーワードは 2 種類あります。

単純キーワード

他のキーワードを含まないキーワード、つまり、組込みキーワードのないキーワードです。 keywordname = value 形式で、この値は左かっこではありません。 次の例で FQ_CP_NAME と NODE_TYPE は単純キーワードですが、NODE は単純キーワードではありません。

```
NODE=(
     FQ_CP_NAME=USIBMNM.NT265
     NODE_TYPE=END_NODE
)
```

複合キーワード

組込み単純キーワードまたは組込み複合キーワードを含んでいます。 次の例で PORT と PORT_LAN_SPECIFIC_DATA は複合キーワードです。

```
PORT=(
     PORT_NAME=LAN1_04
     DLC_NAME=LAN
     PORT_LAN_SPECIFIC_DATA=(
          ADAPTER_ID=LAN1
          ADAPTER_NAME=0001
     )
)
```

キーワードのタイプ

単純キーワードは 6 つのタイプがあって、そのタイプは単純キーワードがもつことのできる値を示します。

BOOLEANKEYWORD

ブール値 (0 または 1) だけをもつことのできるキーワード。

ENUMKEYWORD

値を列挙したキーワード。 有効値は ENUM_LIST にリストされています。

HEXNUMBERKEYWORD
16 進数の値をもつキーワード。 16 進数の有効範囲は RANGE によって与えられます。

HEXSTRINGKEYWORD
その値として16 進数の数字のストリングをもつキーワード。 16 進数ストリングの有効な長さは FIELD_LENGTH によって与えられます。

STRINGKEYWORD
ストリング値をもつキーワード。 ストリングの有効な長さは FIELD_LENGTH によって与えられます。

UNSIGNEDNUMBERKEYWORD
無符号の数値をもつキーワード。 数値の有効範囲は RANGE によって与えられます。

その他のキーワード・フィールドとその意味

キーワードの多くは、以下の値をサポートします。

@REQUIRED
そのキーワードが必須かどうかを判別します。1 は必須であることを意味します。0 は必須ではないことを意味します。ただし、デフォルト値が指定されている場合、それは自動的に追加されます。

@DEFAULT
そのキーワードに対する省略時値を指定します。 ASCII ファイルにそのキーワードがない場合、この省略時値が ASCII ファイルに充てんされ、追加されます。

@COMPLETE_SYNTAX
STRINGKEYWORD に対する有効文字値を指定します。 たとえば、SNA_TYPE_A です。

キーワードの形式

このセクションではキーワードの説明をします。値を埋めたサンプル・キーワードと一緒に、キーワードの定義を含むテンプレートが示してあります。

NODE

キーワードの定義

NODE キーワードは START_NODE verb に関連します。 各フィールドの記述については、619ページの『START_NODE』 を参照してください。

```
NODE = (
   ANYNET_SUPPORT = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         ANYNET_SUPPORTED = 0
         ACCESS_NODE = 1
         GATEWAY = 2
      )
      @REQUIRED = 1
      @DEFAULT = ANYNET_SUPPORTED
    )
   CP_ALIAS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
    )
   DEFAULT_PREFERENCE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         NATIVE = 0
         NONNATIVE = 1
         NATIVE_THEN_NONNATIVE = 2
         NONNATIVE_THEN_NATIVE = 3
         USE_DEFAULT_PREFERENCE = 255
      )
      @REQUIRED = 1
      @DEFAULT = NATIVE
    )
   DISCOVERY_GROUP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
    )
   DISCOVERY_SUPPORT = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         NO = 0
         DISCOVERY_CLIENT = 1
         DISCOVERY_SERVER = 2
      )
      @REQUIRED = 1
      @DEFAULT = DISCOVERY_CLIENT
    )
   FQ_CP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
    )
   HPR_SUPPORT = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         NONE = 0
         BASE = 1
         RTP = 2
    )
```

```
   NODE_ID = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @DEFAULT = 05D00000
    )
   NODE_TYPE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         NETWORK_NODE = 2
         END_NODE = 3
      )
      @REQUIRED = 1
      @DEFAULT = END_NODE
    )
   REGISTER_WITH_CDS = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
    )
   REGISTER_WITH_NN = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
    )
)

*The end of Complex Keyword NODE
```

NODE のサンプル: NODE キーワードのサンプルは、以下のとおりです。

```
NODE=(
     ANYNET_SUPPORT=ACCESS_NODE
     CP_ALIAS=NT265
     DEFAULT_PREFERENCE=NATIVE
     DISCOVERY_GROUP_NAME=<NONE>
     DISCOVERY_SUPPORT=DISCOVERY_CLIENT
     FQ_CP_NAME=USIBMNM.NT265
     NODE_ID=05D00000
     NODE_TYPE=END_NODE
     REGISTER_WITH_CDS=1
     REGISTER_WITH_NN=1
)
```

PORT

キーワードの定義

PORT キーワードは DEFINE_PORT verb に関連します。 各フィールドの記述については、105ページの『DEFINE_PORT』 を参照してください。

PORT キーワードは Port_*_Specific_Data_ キーワードの 1 つを含まなくてはなりません。 ここで使用される Port_*_Specific_Data キーワードは DLC_NAME の値によって決まります。 たとえば、DLC_NAME=LAN を伴うPORT キーワードには PORT_LAN_SPECIFIC_DATA キーワードが組み込まれます。

OEM ポートの特定データは ASCII 構成を介して構成することはできません。

```
PORT = (
DLC_DATA = (
   @TYPE = HEXSTRINGKEYWORD
   @FIELD_LENGTH = 1,32
 )
DLC_NAME = (
   @TYPE = STRINGKEYWORD
   @FIELD_LENGTH = 1,8
   @REQUIRED = 1
 )
IMPLICIT_DEACT_TIMER = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
 )
IMPLICIT_DSPU_SERVICES = (
   @TYPE = ENUMKEYWORD @ENUM_LIST = (
      NONE = 0
      PU_CONCENTRATION = 1
      DLUR = 2
   )
   @REQUIRED = 1
   @DEFAULT = NONE
 )
IMPLICIT_DSPU_TEMPLATE = (
   @TYPE = STRINGKEYWORD
   @FIELD_LENGTH = 1,8
 )
IMPLICIT_LIMITED_RESOURCE = (
   @TYPE = ENUMKEYWORD @ENUM_LIST = (
      NO = 0
      YES = 1
      INACTIVITY = 2
   )
IMPLICIT_HPR_SUPPORT = (
   @TYPE = BOOLEANKEYWORD
 )
LINK_STATION_ROLE = (
   @TYPE = ENUMKEYWORD @ENUM_LIST = (
      NEGOTIABLE = 0
      PRIMARY = 1
      SECONDARY = 2
      MSEC = 4
```

```
            USE_PORT_DEFAULTS = 255
         )

      LS_XMIT_RCV_CAP = (
         @TYPE = ENUMKEYWORD @ENUM_LIST = (
            TWS = 1
            TWA = 2
         )
      MAX_IFRM_RCVD = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 0,127
         )
      MAX_RCV_BTU_SIZE = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         )
      MAX_SEND_BTU_SIZE = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         )
      PORT_NAME = (
         @TYPE = STRINGKEYWORD
         @FIELD_LENGTH = 1,8
         @REQUIRED = 1
         )
      PORT_TYPE = (
         @TYPE = ENUMKEYWORD @ENUM_LIST = (
            NONSWITCHED = 1
            SWITCHED = 2
            SATF = 4
         )


   PORT_LAN_SPECIFIC_DATA = (
      ACK_DELAY = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 30,1000
         @REQUIRED = 1
         @DEFAULT = 100
         )

      ACK_TIMEOUT = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 500,10000
         @REQUIRED = 1
         @DEFAULT = 3000
         )
      ADAPTER_NUMBER = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 0,7
         @DEFAULT = 0
         )
      BUSY_STATE_TIMEOUT = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 10,60
         @REQUIRED = 1
         @DEFAULT = 15
         )
```

```
IDLE_STATE_TIMEOUT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 10,120
   @REQUIRED = 1
   @DEFAULT = 30
 )
LOCAL_SAP = (
   @TYPE = HEXNUMBERKEYWORD
   @RANGE = 04,FC
   @DEFAULT = 04
 )
OUTSTANDING_TRANSMITS = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 2,64
   @REQUIRED = 1
   @DEFAULT = 16
 )
POLL_TIMEOUT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 500,10000
   @REQUIRED = 1
   @DEFAULT = 3000
 )
POOL_SIZE = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 2,64
   @REQUIRED = 1
   @DEFAULT = 32
 )
REJECT_RESPONSE_TIMEOUT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 5,30
   @REQUIRED = 1
   @DEFAULT = 10
 )
TEST_RETRY_INTERVAL = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 5,30
   @REQUIRED = 1
   @DEFAULT = 8
 )
TEST_RETRY_LIMIT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 3,30
   @REQUIRED = 1
   @DEFAULT = 5
 )
XID_RETRY_INTERVAL = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 5,60
   @REQUIRED = 1
   @DEFAULT = 8
 )
XID_RETRY_LIMIT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 3,30
```

```
      @REQUIRED = 1
      @DEFAULT = 5
    )
)

*The end of Complex Keyword PORT_LAN_SPECIFIC_DATA

PORT_SDLC_SPECIFIC_DATA = (
   ACCEPT_INCOMING_CALLS = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
    )
   CONNECT_RETRY_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,127
      @REQUIRED = 1
      @DEFAULT = 10
    )
   CONNECT_TIMER = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,30
      @REQUIRED = 1
      @DEFAULT = 2
    )
   FRAMING_STANDARD = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         SNA_OVER_ASYNC = 0
         ADVANTIS = 1
         HAYES_AUTOSYNC = 2
      )
      @REQUIRED = 1
      @DEFAULT = SNA_OVER_ASYNC
    )
   FULL_DUPLEX_SUPPORT = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
    )
   INACTIVITY_TIMER = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 40,160
      @REQUIRED = 1
      @DEFAULT = 80
    )
   IRQ_LEVEL = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,15
      @REQUIRED = 1
      @DEFAULT = 3
    )
   MODEM_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,256
    )
   MULTIDROP_PRIMARY_SERVER = (
```

```
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
    )
   PORT_SPEED = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 2400,115200
      @REQUIRED = 1
      @DEFAULT = 57600
    )
   RESPONSE_RETRY_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,127
      @REQUIRED = 1
      @DEFAULT = 10
    )
   RESPONSE_TIMER = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 2,20
      @REQUIRED = 1
      @DEFAULT = 4
    )
   SHARED_RAM_ADDRESS = (
      @TYPE = HEXNUMBERKEYWORD
      @RANGE = C0000,FC000
    )
   STATION_POLL_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,10
      @REQUIRED = 1
      @DEFAULT = 0
    )
   TRANSMISSION_FLAGS = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,10
      @REQUIRED = 1
      @DEFAULT = 1
    )
   USE_CONSTANT_RTS = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
    )
   USE_NRZI_ENCODING = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
    )
)
*The end of Complex Keyword PORT_SDLC_SPECIFIC_DATA
PORT_TWINAX_SPECIFIC_DATA = (
   ADAPTER_TYPE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         OTHER_TWINAX_ADAPTER = 0
         5250E_DISPLAY_STATION_EMULATION_ADAPTER = 1
         5250_AT_COMMUNICATION_ADAPTER = 2
```

```
            5250_EMULATION_PCMCIA_ADAPTER = 3
            5250_PCMCIA_ADAPTER_CARD = 4
            SYSTEM_36_WORKSTATION_EMULATION_ADAPTER_A = 5
            5250_EMULATION_ADAPTER_A = 6
            5205_EMULATION_PCI_ADAPTER = 7
            NONE = -1
         )
         @DEFAULT = NONE
      )
      IO_ADDRESS = (
         @TYPE = HEXNUMBERKEYWORD
         @RANGE = 240A,27FA
         @REQUIRED = 1
         @DEFAULT = 271A
      )
      IRQ_LEVEL = (
         @TYPE = UNSIGNEDNUMBERKEYWORD
         @RANGE = 3,7
         @REQUIRED = 1
         @DEFAULT = 5
      )

      MEMORY_ADDRESS = (
         @TYPE = HEXNUMBERKEYWORD
         @RANGE = C0000,DC000
         @REQUIRED = 1
         @DEFAULT = DC000
      )
)

*The end of Complex Keyword PORT_TWINAX_SPECIFIC_DATA

PORT_X25_SPECIFIC_DATA = (
   ACCEPT_INCOMING_CALLS = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = NO
   )
   ALTERNATE_REMOTE_PHONE_NUMBER = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,64
   )
   COMPLIANCE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         1980_COMPLIANCE = 1980
         1984_COMPLIANCE = 1984
         1988_COMPLIANCE = 1988
      )
      @DEFAULT = 1984_COMPLIANCE
   )
   DEFAULT_WINDOW_SIZE = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,7
      @DEFAULT = 2
   )
   DIAL_TYPE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
```

```
      PULSE = 80
      TONE = 84
   )
   @DEFAULT = TONE
   )
FRAME_INACTIVITY_TIMEOUT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 0,255
   @DEFAULT = 30
 )
FRAME_RETRANSMISSION_TIMEOUT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 1,60
   @DEFAULT = 3
 )
FRAME_SEQUENCE = (
   @TYPE = ENUMKEYWORD @ENUM_LIST = (
      MODULO_8 = 8
      MODULO_128 = 128
   )
   @DEFAULT = MODULO_8
   )
FRAME_TRANSMISSION_RETRY_COUNT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 1,255
   @DEFAULT = 20
 )
FRAME_WINDOW_SIZE = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 1,7
   @DEFAULT = 7
 )

INSERT_CALLING_ADDRESS = (
   @TYPE = BOOLEANKEYWORD
 )
IN_ONLY_SVC_COUNT = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 0,60000
   @DEFAULT = 0
 )
IN_ONLY_SVC_START = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 0,60000
   @DEFAULT = 0
 )
LOCAL_DTE_ADDRESS = (
   @TYPE = STRINGKEYWORD
   @FIELD_LENGTH = 1,16
 )
MAX_PIU_SIZE = (
   @TYPE = UNSIGNEDNUMBERKEYWORD
   @RANGE = 265,4115
   @DEFAULT = 2048
 )
MODEM_NAME = (
```

```
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,256
   )
   NETWORK_CONNECTION_TYPE = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         LEASED = 0
         SWITCHED = 1
      )
      @DEFAULT = SWITCHED
   )
   OUT_ONLY_SVC_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
   )
   OUT_ONLY_SVC_START = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
   )
   PACKET_SIZE = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 16,4096
      @DEFAULT = 128
   )
   PORT_SPEED = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 2400,115200
      @DEFAULT = 57600
   )
   PVC_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
   )
   PVC_START = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
   )
   REMOTE_PHONE_NUMBER = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,64
   )
   SEQUENCING = (
      @TYPE = ENUMKEYWORD @ENUM_LIST = (
         MODULO_8 = 8
         MODULO_128 = 128
      )
      @DEFAULT = MODULO_8
)
   SHARED_RAM_ADDRESS = (
      @TYPE = HEXNUMBERKEYWORD
      @RANGE = C0000,FC000
   )
   TRANSMISSION_FLAGS = (
```

```
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,10
      @DEFAULT = 1
    )
   TWO_WAY_SVC_COUNT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
    )
   TWO_WAY_SVC_START = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,60000
      @DEFAULT = 0
    )
   USE_CONSTANT_RTS = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 1
    )
   USE_NRZI_ENCODING = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 0
    )
   USE_X32_PROTOCOL = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 0
    )
   X32_IDENTITY = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,34
    )
   X32_SIGNATURE = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,34
    )
INCOMING_CALL_FILTER = (
      @MERGE_SIMPLE_KEYWORDS = 0

   ACCEPT_CHARGES = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 0
    )
   DTE_ADDRESS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 0,16
    )
   DTE_ADDRESS_EXTENSION = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 0,8
    )

)
*The end of Complex Keyword INCOMING_CALL_FILTER
)
```

```
*The end of Complex Keyword PORT_X25_SPECIFIC_DATA
)

*The end of Complex Keyword PORT
```

PORT のサンプル: PORT キーワードのサンプルは、以下のとおりです。

```
PORT=(
     PORT_NAME=ANYNET
     DLC_NAME=ANYNET
     IMPLICIT_DEACT_TIMER=0
     IMPLICIT_DSPU_SERVICES=NONE
     IMPLICIT_HPR_SUPPORT=0
     IMPLICIT_LIMITED_RESOURCE=NO
     MAX_IFRM_RCVD=127
     MAX_RCV_BTU_SIZE=9216
     MAX_SEND_BTU_SIZE=9216

     PORT_TYPE=SATF
)
PORT=(
     PORT_NAME=LAN0_04
     DLC_DATA=00000000000004
     DLC_NAME=LAN
     IMPLICIT_DEACT_TIMER=0
     IMPLICIT_DSPU_SERVICES=NONE
     IMPLICIT_HPR_SUPPORT=1
     IMPLICIT_LIMITED_RESOURCE=NO
     MAX_IFRM_RCVD=8
     MAX_RCV_BTU_SIZE=65535
     MAX_SEND_BTU_SIZE=65535

     PORT_TYPE=SATF
     PORT_LAN_SPECIFIC_DATA=(
          ACK_DELAY=100
          ACK_TIMEOUT=1000
          ADAPTER_ID=LAN0
          ADAPTER_NAME=0000
          BUSY_STATE_TIMEOUT=15
          IDLE_STATE_TIMEOUT=30
          OUTSTANDING_TRANSMITS=16
          POLL_TIMEOUT=3000
          REJECT_RESPONSE_TIMEOUT=10
          TEST_RETRY_INTERVAL=8
          TEST_RETRY_LIMIT=5
          XID_RETRY_INTERVAL=8
          XID_RETRY_LIMIT=5
     )
)
```

LINK_STATION

キーワードの定義

LINK_STATION キーワードは DEFINE_LS verb に関連します。 各フィールドの記述については、76ページの『DEFINE_LS』 を参照してください。

LINK キーワードは Link_Station_*_Specific_Data キーワードの 1 つを含みます。ここで使用される Link_Station_*_Specific_Data キーワードは、PORT_NAME の値によって決まります。 たとえば、PORT_NAME の値が LAN ポートを参照する場合、LINK_STATION_LAN_SPECIFIC_DATA キーワードが組み込まれます。

```
LINK_STATION = (
  ACTIVATE_AT_STARTUP = (
     @TYPE = BOOLEANKEYWORD
     @REQUIRED = 1
     @DEFAULT = 1
  )
  ADJACENT_NODE_ID = (
     @TYPE = HEXSTRINGKEYWORD
     @FIELD_LENGTH = 1,8
  )
  ADJACENT_NODE_TYPE = (
     @TYPE = ENUMKEYWORD
     @ENUM_LIST = (
        APPN_NODE = 0
        NETWORK_NODE = 2
        END_NODE = 3
        BACK_LEVEL_LEN_NODE = 5
        HOST_XID3 = 6
        HOST_XID0 = 7
        DSPU_XID = 8
        DSPU_NO_XID = 9
     )
     @REQUIRED = 1
     @DEFAULT = APPN_NODE
  )
  AUTO_ACTIVATE_SUPPORT = (
     @TYPE = BOOLEANKEYWORD
  )
  BKUP_DLUS_NAME = (
     @TYPE = STRINGKEYWORD
     @FIELD_LENGTH = 1,17
     @COMPLETE_SYNTAX = %FULLY_QUALIFIED%
  )
  CP_CP_SESS_SUPPORT = (
     @TYPE = BOOLEANKEYWORD
  )
  DEFAULT_NN_SERVER = (
     @TYPE = BOOLEANKEYWORD
  )
  DEST_ADDRESS = (
     @TYPE = HEXSTRINGKEYWORD
     @FIELD_LENGTH = 0,32
     @REQUIRED = 1
```

```
)
DISABLE_REMOTE_ACT = (
   @TYPE = BOOLEANKEYWORD
   @REQUIRED = 1
   @DEFAULT = 0
)
DLUS_NAME = (
   @TYPE = STRINGKEYWORD
   @FIELD_LENGTH = 1,17
   @COMPLETE_SYNTAX = %FULLY_QUALIFIED%
)
DSPU_NAME = (
    @TYPE = STRINGKEYWORD
    @FIELD_LENGTH = 1,8
)
DSPU_SERVICES = (
    @TYPE = ENUMKEYWORD
    @ENUM_LIST = (
       NONE = 0
       PU_CONCENTRATION = 1
       DLUR = 2
    )
    @REQUIRED = 1
    @DEFAULT = NONE
    )
FQ_ADJACENT_CP_NAME = (
    @TYPE = STRINGKEYWORD
    @FIELD_LENGTH = 1,17
    @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
    )
HOST_TYPE = (
    @TYPE = ENUMKEYWORD
    @ENUM_LIST = (
       SNA = 0
       HNA = 64
       FNA = 128
       )
HPR_LINK_LVL_ERROR = (
    @TYPE = BOOLEANKEYWORD
)
HPR_SUPPORT = (
    @TYPE = BOOLEANKEYWORD
    @DEFAULT = 0
      )
LIMITED_RESOURCE = (
    @TYPE = ENUMKEYWORD
    @ENUM_LIST = (
       NO = 0
       YES = 1
       INACTIVITY = 2
    )
    @REQUIRED = 1
)
LINK_DEACT_TIMER = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
)
```

```
LINK_SPEC_DATA_LEN = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
    @DEFAULT = 0
)
LINK_STATION_ROLE = (
    @TYPE = ENUMKEYWORD
    @ENUM_LIST = (
       NEGOTIABLE = 0
       PRIMARY = 1
       SECONDARY = 2
       USE_ADAPTER_DEFAULTS = 255
    )
LS_NAME = (
    @TYPE = STRINGKEYWORD
    @FIELD_LENGTH = 1,8
    @REQUIRED = 1
)
MAX_IFRM_RCVD = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
    @RANGE = 0,127
)
MAX_SEND_BTU_SIZE = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
)
NODE_ID = (
    @TYPE = HEXSTRINGKEYWORD
    @FIELD_LENGTH = 1,8
)
PORT_NAME = (
    @TYPE = STRINGKEYWORD
    @FIELD_LENGTH = 1,8
    @REQUIRED = 1
)
PU_NAME = (
    @TYPE = STRINGKEYWORD
    @FIELD_LENGTH = 1,8
    @COMPLETE_SYNTAX = %SNA_TYPE_A%
)
SOLICIT_SSCP_SESSION = (
    @TYPE = BOOLEANKEYWORD
)
SUPPRESS_CP_NAME = (
    @TYPE = ENUMKEYWORD
    @ENUM_LIST = (
       NO = 0
       YES = 128
    )
TARGET_PACING_COUNT = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
    @RANGE = 1,32767
)
TG_NUMBER = (
    @TYPE = UNSIGNEDNUMBERKEYWORD
    @RANGE = 0,20
    @DEFAULT = 0
)
```

```
   USE_DEFAULT_TG_CHARS = (
       @TYPE = BOOLEANKEYWORD
   )

LINK_STATION_ANYNET_SPECIFIC_DATA = (
   PARTNER_ADDRESS_TYPE = (
       @TYPE = ENUMKEYWORD
       @ENUM_LIST = (
          USE_CP_NAME = 0
          USE_BLOCK_AND_PU_ID = 1
       )
       @DEFAULT = USE_CP_NAME
)

*The end of Complex Keyword LINK_STATION_ANYNET_SPECIFIC_DATA
LINK_STATION_LAN_SPECIFIC_DATA = (
   TEST_RETRY_INTERVAL = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 5,30
       @DEFAULT = 8
   )
   TEST_RETRY_LIMIT = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 3,30
       @DEFAULT = 5
   )
   XID_RETRY_INTERVAL = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 2,20
       @REQUIRED = 1
       @DEFAULT = 4
   )
   XID_RETRY_LIMIT = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 3,30
       @DEFAULT = 5
   )

*The end of Complex Keyword LINK_STATION_LAN_SPECIFIC_DATA
LINK_STATION_SDLC_SPECIFIC_DATA = (
   AUTO_REACTIVATE_SUPPORT = (
       @TYPE = BOOLEANKEYWORD
       @DEFAULT = 0
   )
   BACKUP_PHONE_NUMBER = (
       @TYPE = STRINGKEYWORD
       @FIELD_LENGTH = 1,62
   )
   CONNECT_RETRY_COUNT = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 0,127
       @DEFAULT = 10
   )
   CONNECT_TIMER = (
       @TYPE = UNSIGNEDNUMBERKEYWORD
       @RANGE = 1,30
```

```
        @DEFAULT = 2
    )
    FRAMING_STANDARD = (
        @TYPE = ENUMKEYWORD
        @ENUM_LIST = (
           SNA_OVER_ASYNC = 0
           ADVANTIS = 1
           HAYES_AUTOSYNC = 2
        )
        @DEFAULT = SNA_OVER_ASYNC
        )
    INACTIVITY_TIMER = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 40,160
        @DEFAULT = 80
    )
    PORT_SPEED = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 2400,115200
        @DEFAULT = 57600
    )
    PRIMARY_PHONE_NUMBER = (
        @TYPE = STRINGKEYWORD
        @FIELD_LENGTH = 1,62
    )
    RESPONSE_RETRY_COUNT = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 1,127
        @DEFAULT = 10
    )
    RESPONSE_TIMER = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 2,20
        @DEFAULT = 4
    )
    USE_NRZI_ENCODING = (
        @TYPE = BOOLEANKEYWORD
        @DEFAULT = 0
    )
)

*The end of Complex Keyword LINK_STATION_SDLC_SPECIFIC_DATA
LINK_STATION_X25_SPECIFIC_DATA = (
    ADDITIONAL_FACILITIES = (
        @TYPE = STRINGKEYWORD
        @FIELD_LENGTH = 1,110
    )
    CALL_USER_GROUP_FORMAT = (
        @TYPE = ENUMKEYWORD
        @ENUM_LIST = (
           NONE = 0
           BASIC = 1
           EXTENDED = 2
        )
    CALL_USER_GROUP_INDEX = (
        @TYPE = STRINGKEYWORD
```

```
            @FIELD_LENGTH = 1,6
        )
        CONNECTION_ID = (
            @TYPE = HEXSTRINGKEYWORD
            @TYPE = STRINGKEYWORD
            @FIELD_LENGTH = 1,16
        )
        CONNECTION_TYPE = (
            @TYPE = ENUMKEYWORD
            @ENUM_LIST = (
               PVC = 0
               SVC = 1
            )
            @DEFAULT = PVC
        )
        LOGICAL_CHANNEL_NUMBER = (
            @TYPE = UNSIGNEDNUMBERKEYWORD
            @RANGE = 5,9
        )
        NETWORK_USER_ID = (
            @TYPE = STRINGKEYWORD
            @FIELD_LENGTH = 1,42
        )
        PACKET_SIZE = (
            @TYPE = UNSIGNEDNUMBERKEYWORD
            @RANGE = 16,4096
            @DEFAULT = 128
        )
        REMOTE_CONFORMANCE = (
            @TYPE = ENUMKEYWORD
            @ENUM_LIST = (
               1980_COMPLIANCE = 1980
               1984_COMPLIANCE = 1984
               1988_COMPLIANCE = 1988
            )
            @DEFAULT = 1984_COMPLIANCE
        )
        REQUEST_REVERSE_CHARGING = (
            @TYPE = BOOLEANKEYWORD
            @DEFAULT = 0
        )
        WINDOW_SIZE = (
            @TYPE = UNSIGNEDNUMBERKEYWORD
            @RANGE = 1,7
            @DEFAULT = 2
        )

    X25_DESTINATION_ADDRESS = (
        DTE_ADDRESS = (
           @TYPE = STRINGKEYWORD
           @FIELD_LENGTH = 1,16
        )
        DTE_ADDRESS_EXTENSION = (
           @TYPE = STRINGKEYWORD
           @FIELD_LENGTH = 1,42
        )
```

```
)

*The end of Complex Keyword X25_DESTINATION_ADDRESS
)

*The end of Complex Keyword LINK_STATION_X25_SPECIFIC_DATA

*The end of Complex Keyword LINK_STATION
```

LINK_STATION のサンプル: LINK_STATION キーワードのサンプルは、以下のとおりです。

```
LINK_STATION = (
   LS_NAME=LINK0000
   ACTIVATE_AT_STARTUP=0
   ADJACENT_NODE_TYPE=APPN_NODE
   AUTO_ACTIVATE_SUPPORT=1
   CP_CP_SESS_SUPPORT=1
   DEFAULT_NN_SERVER=0
   DEST_ADDRESS=40000000000004
   DISABLE_REMOTE_ACT=0
   DSPU_SERVICES=NONE
   HPR_LINK_LVL_ERROR=0
   HPR_SUPPORT=0
   LIMITED_RESOURCE=NO
   LINK_DEACT_TIMER=0
   LINK_STATION_ROLE=USE_ADAPTER_DEFAULTS
   MAX_IFRM_RCVD=0
   MAX_SEND_BTU_SIZE=65535
   NODE_ID=05D00000
   PORT_NAME=LAN0_04
   SOLICIT_SSCP_SESSION=0
   SUPPRESS_CP_NAME=NO
   TARGET_PACING_COUNT=1
   TG_NUMBER=0
   USE_DEFAULT_TG_CHARS=1
   LINK_STATION_LAN_SPECIFIC_DATA = (
      TEST_RETRY_INTERVAL=8
      TEST_RETRY_LIMIT=5
      XID_RETRY_INTERVAL=8
      XID_RETRY_LIMIT=5
   )
)
LINK_STATION = (
   LS_NAME=LINK0001
   ACTIVATE_AT_STARTUP=0
   ADJACENT_NODE_TYPE=DSPU_XID
   AUTO_ACTIVATE_SUPPORT=0
   CP_CP_SESS_SUPPORT=1
   DEFAULT_NN_SERVER=0
   DEST_ADDRESS=40000000000104
   DISABLE_REMOTE_ACT=0
   DSPU_NAME=LINK0001
   DSPU_SERVICES=PU_CONCENTRATION
   HPR_LINK_LVL_ERROR=0
   HPR_SUPPORT=0
   LIMITED_RESOURCE=NO
```

```
      LINK_DEACT_TIMER=0
      LINK_STATION_ROLE=USE_ADAPTER_DEFAULTS
      MAX_IFRM_RCVD=0
      MAX_SEND_BTU_SIZE=65535
      NODE_ID=05D00000
      PORT_NAME=LAN0_04
      SOLICIT_SSCP_SESSION=0
      STARTUP=1
      SUPPRESS_CP_NAME=NO
      TARGET_PACING_COUNT=1
      TG_NUMBER=0
      USE_DEFAULT_TG_CHARS=1
      LINK_STATION_LAN_SPECIFIC_DATA = (
         TEST_RETRY_INTERVAL=8
         TEST_RETRY_LIMIT=5
         XID_RETRY_INTERVAL=8
         XID_RETRY_LIMIT=5
      )
   )
```

INTERNAL_PU

キーワードの定義

INTERNAL_PU キーワードは DEFINE_INTERNAL_PU verb に関連します。 各フィールドの記述については、70ページの『DEFINE_INTERNAL_PU』 を参照してください。

```
INTERNAL_PU = (
   BKUP_DLUS_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   FQ_DLUS_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   NODE_ID = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
   PU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   STARTUP = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
   )
)

*The end of Complex Keyword INTERNAL_PU
```

INTERNAL_PU のサンプル: INTERNAL_PU キーワードのサンプルは、以下のとおりです。INTERNAL_PU キーワードのサンプルは、以下のとおりです。

```
INTERNAL_PU=(
   PU_NAME=NT265
   NODE_ID=05D00000
   STARTUP=1
)
```

DLUR_DEFAULTS

キーワードの定義

DLUR_DEFAULTS キーワードは DEFINE_DLUR_DEFAULTS verb に関連します。各フィールドの記述については、55ページの『DEFINE_DLUR_DEFAULTS』 を参照してください。

```
DLUR_DEFAULTS = (
   BKUP_DLUS_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   DEFAULT_PU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   DLUS_RETRY_LIMIT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
   )
   DLUS_RETRY_TIMEOUT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
   )
   FQ_DLUS_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
)

*The end of Complex Keyword DLUR_DEFAULTS
```

DLUR_DEFAULTS のサンプル: DLUR_DEFAULTS キーワードのサンプルは、以下のとおりです。

```
DLUR_DEFAULTS=(
   BKUP_DLUS_NAME=USIBMNR.DLURBACK
   DEFAULT_PU_NAME=NT265
   DLUS_RETRY_LIMIT=3
   DLUS_RETRY_TIMEOUT=5
   FQ_DLUS_NAME=USIBMNM.DLURSRV
)
```

SPLIT_STACK

キーワードの定義

SPLIT_STACK キーワードのフィールド記述については、Communications Server のオンライン・ヘルプを参照してください。

```
SPLIT_STACK = (
   POOL_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   STARTUP = (
      @TYPE = BOOLEANKEYWORD
         @REQUIRED = 1
           @DEFAULT = 1
            )
)

*The end of Complex Keyword SPLIT_STACK
```

SPLIT_STACK のサンプル: SPLIT_STACK キーワードのサンプルは、以下のとおりです。

```
SPLIT_STACK=(
     STARTUP=1
)
```

TN3270E_DEF

キーワードの定義

TN3270E_DEF キーワードのフィールド記述については、Communications Server のオンライン・ヘルプを参照してください。

```
TN3270E_DEF = (
   AUTO_LOGOFF = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
   )
   DEFAULT_POOL_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   FREQUENCY = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
      @REQUIRED = 1
      @DEFAULT = 60
   )
   KEEPALIVE_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         TN_NONE = 0
         TN_NOP = 1
         TN_TIMING_MARK = 2
      )
      @REQUIRED = 1
      @DEFAULT = TN_NONE
   )
   LOGOFF = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
      @REQUIRED = 1
      @DEFAULT = 30
   )
   PORT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
      @REQUIRED = 1
      @DEFAULT = 23
   )
   TIMER = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,65535
      @REQUIRED = 1
      @DEFAULT = 10
   )
)

*The end of Complex Keyword TN3270E_DEF
```

TN3270E_DEF のサンプル: TN3270E_DEF キーワードのサンプルは、以下のとおりです。

```
TN3270E_DEF=(
   AUTO_LOGOFF=1
   DEFAULT_POOL_NAME=POOL1
   FREQUENCY=60
   KEEPALIVE_TYPE=TN_NOP
   LOGOFF=30
   PORT=23
   TIMER=10
)
```

ADJACENT_NODE

キーワードの定義

ADJACENT_NODE キーワードは、DEFINE_ADJACENT_NODE verb に関連します。各フィールドの記述については、34ページの『DEFINE_ADJACENT_NODE』 を参照してください。

```
ADJACENT_NODE = (
   FQ_CP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   FQ_LU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @MERGE_SIMPLE_KEYWORDS = 0
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
)

*The end of Complex Keyword ADJACENT_NODE
```

ADJACENT_NODE のサンプル: ADJACENT_NODE キーワードのサンプルは、以下のとおりです。

```
ADJACENT_NODE=(
   FQ_CP_NAME=USIBMNM.PARTNER
   FQ_LU_NAME=USIBMNM.PARTLU
   FQ_LU_NAME=USIBMNM.PARTLU1
   FQ_LU_NAME=USIBMNM.PARTLU2
)
```

CONNECTION_NETWORK

キーワードの定義

CONNECTION_NETWORK キーワードは、以下のとおりです。

```
CONNECTION_NETWORK = (
   FQCN_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   PORT_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @MERGE_SIMPLE_KEYWORDS = 0
   )
)

*The end of Complex Keyword CONNECTION_NETWORK
```

CONNECTION_NETWORK のサンプル: CONNECTION_NETWORK キーワードのサンプルは、以下のとおりです。

```
CONNECTION_NETWORK=(
   FQCN_NAME=USIBMNR.CONNET
   PORT_NAME=LAN0_04
)
```

DSPU_TEMPLATE

キーワードの定義

DSPU_TEMPLATE キーワードは DEFINE_DSPU_TEMPLATE verb に関連します。各フィールドの記述については 63ページの『DEFINE_DSPU_TEMPLATE』 を参照してください。

```
DSPU_TEMPLATE = (
   MAX_INSTANCE = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
   )
   NUMBER_OF_DSLU_TEMPLATES = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
   )
   TEMPLATE_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )

DSLU_TEMPLATE = (
   HOST_LU = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   MAX_NAU = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
   )
   MIN_NAU = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
   )
)

*The end of Complex Keyword DSLU_TEMPLATE

*The end of Complex Keyword DSPU_TEMPLATE
```

DSPU_TEMPLATE のサンプル: DSPU_TEMPLATE キーワードのサンプルは、以下のとおりです。

```
DSPU_TEMPLATE=(
   TEMPLATE_NAME=DOWN
   MAX_INSTANCE=0
   NUMBER_OF_DSLU_TEMPLATES=1

DSLU_TEMPLATE=(
      HOST_LU=PUBLIC
      MAX_NAU=5
      MIN_NAU=1
)
```

DOWNSTREAM_LU

キーワードの定義

DOWNSTREAM_LU キーワードは DEFINE_DOWNSTREAM_LU verb に関連します。 各フィールドの記述については、57ページの『DEFINE_DOWNSTREAM_LU』を参照してください。

```
DOWNSTREAM_LU = (
   DSLU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   DSPU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   HOST_LU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   NAU_ADDRESS = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,255
   )
)

*The end of Complex Keyword DOWNSTREAM_LU
```

DOWNSTREAM_LU のサンプル: DOWNSTREAM_LU キーワードのサンプルは、以下のとおりです。

```
DOWNSTREAM_LU=(
   DSLU_NAME=GR08005
   DSPU_NAME=GR08
   HOST_LU_NAME=PUBLIC
   NAU_ADDRESS=5
)
```

FOCAL_POINT

キーワードの定義

FOCAL_POINT キーワードは DEFINE_FOCAL_POINT verb に関連します。 各フィールドの記述については、66ページの『DEFINE_FOCAL_POINT』 を参照してください。

```
FOCAL_POINT = (
   BKUP_FP_FQCP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   BKUP_MS_APPL_NAME = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED%
   )
   FP_FQCP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
      @REQUIRED = 1
   )
   MS_APPL_NAME = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
   MS_CATEGORY = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
)

*The end of Complex Keyword FOCAL_POINT
```

FOCAL_POINT のサンプル: FOCAL_POINT キーワードのサンプルは、以下のとおりです。

```
FOCAL_POINT=(
   MS_CATEGORY=23F0F1F7
   BKUP_FP_FQCP_NAME=USIBMNR.BACKUP
   BKUP_MS_APPL_NAME=23F0F1F6
   FP_FQCP_NAME=USIBMNR.FOCAL
   MS_APPL_NAME=23F0F1F6
)
```

LOCAL_LU

キーワードの定義

LOCAL_LU キーワードは DEFINE_LOCAL_LU verb に関連します。 各フィールドの記述については、73ページの『DEFINE_LOCAL_LU』 を参照してください。

```
LOCAL_LU = (
   LU_ALIAS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
   LU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   LU_SESSION_LIMIT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,65535
      @DEFAULT = 0
   )
   NAU_ADDRESS = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,255
   )
   PU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   ROUTE_TO_CLIENT = (
      @TYPE = BOOLEANKEYWORD
   )
)

*The end of Complex Keyword LOCAL_LU
```

LOCAL_LU のサンプル: LOCAL_LU キーワードのサンプルは、以下のとおりです。

```
LOCAL_LU=(
   LU_NAME=LOCLU62
   LU_ALIAS=LOCALIAS
   LU_SESSION_LIMIT=0
   NAU_ADDRESS=0
   ROUTE_TO_CLIENT=0
)
```

LU_0_TO_3

キーワードの定義

LU_0_TO_3 キーワードは DEFINE_LU_0_TO_3 に関連します。 各フィールドの記述については、89ページの『DEFINE_LU_0_TO_3』 を参照してください。

```
LU_0_TO_3 = (
   APPLICATION_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         UNASSIGNED = 0
         TN3270E = 257
      )
   ASSOC_PRINTER = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   CLASS_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         TN_UNASSIGNED = 0
         TN_IMPLICIT_WORKSTATION = 1
         TN_EXPLICIT_WORKSTATION = 2
         TN_IMPLICIT_PRINTER = 3
         TN_EXPLICIT_PRINTER = 4
         TN_ASSOC_PRINTER = 5
      )
   LU_MODEL = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         3270_DISPLAY_MODEL_2 = 2
         3270_DISPLAY_MODEL_3 = 3
         3270_DISPLAY_MODEL_4 = 4
         3270_DISPLAY_MODEL_5 = 5
         RJE_WKSTN = 32
         PRINTER = 128
         UNKNOWN = 0
      )
      @REQUIRED = 1
      @DEFAULT = 3270_DISPLAY_MODEL_2
           )
   LU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   NAU_ADDRESS = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,255
      @REQUIRED = 1
   )
   POOL_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
```

```
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   PRIORITY = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         NETWORK = 3
         HIGH = 2
         MEDIUM = 1
         LOW = 0
      )
      @REQUIRED = 1
      @DEFAULT = MEDIUM
      )
   PU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
)

*The end of Complex Keyword LU_0_TO_3
```

LU_0_TO_3 のサンプル: LU_0_TO_3 キーワードのサンプルは、以下のとおりです。

```
LU_0_TO_3=(
   LU_NAME=LUA2
   LU_MODEL=3270_DISPLAY_MODEL_2
   NAU_ADDRESS=2
   PRIORITY=MEDIUM
   PU_NAME=NT265
)
```

MODE

キーワードの定義

MODE キーワードは DEFINE_MODE verb に関連します。 各フィールドの記述については、98ページの『DEFINE_MODE』 を参照してください。

```
MODE = (
   AUTO_ACT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,1
      @REQUIRED = 1
      @DEFAULT = 0
   )
   COS_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
   CRYPTOGRAPHY = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         NONE = 0
         MANDATORY = 1
      )
      @REQUIRED = 1
      @DEFAULT = NONE
   )
   DEFAULT_RU_SIZE = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
   )
   MAX_NEGOTIABLE_SESSION_LIMIT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,32767
      @REQUIRED = 1
      @DEFAULT = 128
   )
   MAX_RU_SIZE_UPPER_BOUND = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 256,16384
      @REQUIRED = 1
   )
   MIN_CONWINNERS_SOURCE = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,32767
      @REQUIRED = 1
      @DEFAULT = 16
   )
   MODE_NAME = (
      @TYPE = STRINGKEYWORD
       @FIELD_LENGTH = 1,8
       @REQUIRED = 1
       @COMPLETE_SYNTAX = %SNA_TYPE_A%
```

```
        )
    PLU_MODE_SESSION_LIMIT = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 0,32767
        @REQUIRED = 1
        @DEFAULT = 32
    )
    RECEIVE_PACING_WINDOW = (
        @TYPE = UNSIGNEDNUMBERKEYWORD
        @RANGE = 1,63
        @REQUIRED = 1
        @DEFAULT = 1
    )
)

*The end of Complex Keyword MODE
```

MODE のサンプル: MODE キーワードのサンプルは、以下のとおりです。

```
MODE=(
   MODE_NAME=BLANK
   AUTO_ACT=0
   COS_NAME=#CONNECT
   CRYPTOGRAPHY=NONE
   DEFAULT_RU_SIZE=1
   MAX_NEGOTIABLE_SESSION_LIMIT=8
   MAX_RU_SIZE_UPPER_BOUND=1024
   MIN_CONWINNERS_SOURCE=4
   PLU_MODE_SESSION_LIMIT=8
   RECEIVE_PACING_WINDOW=3
)
MODE=(
   MODE_NAME=#INTER
   AUTO_ACT=0
   COS_NAME=#INTER
   CRYPTOGRAPHY=NONE
   DEFAULT_RU_SIZE=1
   MAX_NEGOTIABLE_SESSION_LIMIT=8
   MAX_RU_SIZE_UPPER_BOUND=4096
   MIN_CONWINNERS_SOURCE=4
   PLU_MODE_SESSION_LIMIT=8
   RECEIVE_PACING_WINDOW=20
)
```

PARTNER_LU

キーワードの定義

PARTNER_LU キーワードは DEFINE_PARTNER_LU verb に関連します。 各フィールドの記述については、102ページの『DEFINE_PARTNER_LU』 を参照してください。

```
PARTNER_LU = (
   ADJACENT_CP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   CONV_SECURITY_VERIFICATION = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 1
   )
   FQ_PLU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   MAX_MC_LL_SEND_SIZE = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 1,32767
      @DEFAULT = 32767
   )
   PARALLEL_SESSION_SUPPORT = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 1
   )
   PARTNER_LU_ALIAS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      )
   PREFERENCE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         NATIVE = 0
         NONNATIVE = 1
         NATIVE_THEN_NONNATIVE = 2
         NONNATIVE_THEN_NATIVE = 3
         USE_DEFAULT_PREFERENCE = 255
      )
      @REQUIRED = 1
      @DEFAULT = USE_DEFAULT_PREFERENCE
   )
)

*The end of Complex Keyword PARTNER_LU
```

PARTNER_LU のサンプル: PARTNER_LU キーワードのサンプルは、以下のとおりです。

```
PARTNER_LU=(
   FQ_PLU_NAME=USIBMNM.DLURSRV
   CONV_SECURITY_VERIFICATION=1
   MAX_MC_LL_SEND_SIZE=32767
   PARALLEL_SESSION_SUPPORT=1
   PARTNER_LU_ALIAS=DLURSRV
   PREFERENCE=USE_DEFAULT_PREFERENCE
)
```

@@TP

キーワードの定義

TP キーワードは DEFINE_TP verb に関連します。 各フィールドの記述については、113ページの『DEFINE_TP』 を参照してください。

```
TP = (
   API_CLIENT_USE (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
   )
   CONVERSATION_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         BASIC = 0
         MAPPED = 1
         EITHER = 2
      )
      @REQUIRED = 1
      @DEFAULT = EITHER
   )
   DUPLEX_SUPPORT = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         HALF_DUPLEX = 0
         FULL_DUPLEX = 1
         EITHER_DUPLEX = 2
      )
      @REQUIRED = 1
      @DEFAULT = EITHER_DUPLEX
   )
   DYNAMIC_LOAD = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
   )
   INCOMING_ALLOCATE_TIMEOUT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,65535
      @REQUIRED = 1
      @DEFAULT = 30
   )
   LOAD_TYPE = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
   )
   PARAMETERS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,32
   )
   PATHNAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,80
```

```
   )
   PIP_ALLOWED = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
   )
   QUEUED = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
   )
   RECEIVE_ALLOCATE_TIMEOUT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,65535
      @REQUIRED = 1
      @DEFAULT = 3600
   )
   SECURITY_RQD = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 1
   )
   SYNC_LEVEL = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         NONE = 0
         CONFIRM_SYNC_LEVEL = 1
         EITHER = 2
         SYNCPT_REQUIRED = 3
         SYNCPT_NEGOTIABLE = 4
      )
      @REQUIRED = 1
      @DEFAULT = EITHER
   )
   TP_INSTANCE_LIMIT = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,65535
      @REQUIRED = 1
      @DEFAULT = 0
   )
   TP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,64
      @REQUIRED = 1
   )
   TP_NAME_FORMAT = (
      @TYPE = BOOLEANKEYWORD
      @REQUIRED = 1
      @DEFAULT = 0
   )
)

*The end of Complex Keyword TP
```

TP のサンプル: TP キーワードのサンプルは、以下のとおりです。

@@TP

```
TP=(
   TP_NAME=MYTP
   CONVERSATION_TYPE=EITHER
   DUPLEX_SUPPORT=EITHER_DUPLEX
   DYNAMIC_LOAD=1
   INCOMING_ALLOCATE_TIMEOUT=30
   LOAD_TYPE=0
   PATHNAME=d:\tps\mytp.exe
   PIP_ALLOWED=1
   QUEUED=0
   RECEIVE_ALLOCATE_TIMEOUT=3600
   SECURITY_RQD=1
   SYNC_LEVEL=EITHER
   TP_INSTANCE_LIMIT=0
)
```

CPIC_SIDE_INFO

キーワードの定義

CPIC_SIDE_INFO キーワードは DEFINE_CPIC_SIDE_INFO に関連します。 各フィールドの記述については、513ページの『DEFINE_CPIC_SIDE_INFO』 を参照してください。

```
CPIC_SIDE_INFO = (
   CONVERSATION_SECURITY_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         NONE = 0
         SAME = 1
         PROGRAM = 2
         STRONG = 5
      )
      @DEFAULT = NONE
   )
   MODE_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
   )
   PARTNER_LU_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,17
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %FULLY_QUALIFIED_SNA_TYPE_A%
   )
   SECURITY_PASSWORD = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,10
   )
   SECURITY_USER_ID = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,10
   )
   SYM_DEST_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
   TP_NAME = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,64
   )
   TP_NAME_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         APPLICATION_TP = 0
         SNA_SERVICE = 1
      )
      @REQUIRED = 1
      @DEFAULT = APPLICATION_TP
   )
   USER_DATA = (
```

```
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,32
   )
)

*The end of Complex Keyword CPIC_SIDE_INFO
```

CPIC_SIDE_INFO のサンプル: CPIC_SIDE_INFO キーワードのサンプルは、以下のとおりです。

```
CPIC_SIDE_INFO=(
   SYM_DEST_NAME=APINGD
   CONVERSATION_SECURITY_TYPE=NONE
   MODE_NAME=#INTER
   PARTNER_LU_NAME=USIBMNM.PARTNER1
   TP_NAME=APINGD
   TP_NAME_TYPE=APPLICATION_TP
)
```

LU_LU_PASSWORD

キーワードの定義

LU_LU_PASSWORD キーワードは DEFINE_LU_LU_PASSWORD verb に関連します。 各フィールドの記述については、496ページの『DEFINE_LU_LU_PASSWORD』を参照してください。

LU_PAIR キーワードの値はローカル LU 名で構成された、コンマで区切られた完全修飾のパートナー LU 名です。

```
LU_LU_PASSWORD = (
   LU_PAIR = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,24
      @REQUIRED = 1
      @COMPLETE_SYNTAX = @SEG(1, (,), %SNA_TYPE_A% && @LENGTH<=8)
         && @SEG(2, (,), %FULLY_QUALIFIED_SNA_TYPE_A%)
    )
   PASSWORD = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,8
      @REQUIRED = 1
   )
)

*The end of Complex Keyword LU_LU_PASSWORD
```

LU_LU_PASSWORD のサンプル: LU_LU_PASSWORD キーワードのサンプルは、以下のとおりです。

```
LU_LU_PASSWORD=(
   LU_PAIR=NT265,USIBMNM.PARTLU
   PASSWORD=460C7761C854E0E6
)
```

USERID_PASSWORD

キーワードの定義

USERID_PASSWORD キーワードは DEFINE_USERID_PASSWORD verb に関連します。各フィールドの記述については、
499ページの『DEFINE_USERID_PASSWORD』 を参照してください。

```
USERID_PASSWORD = (
   PASSWORD = (
      @TYPE = HEXSTRINGKEYWORD
      @FIELD_LENGTH = 1,10
      @REQUIRED = 1
   )
   USER_ID = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,10
      @REQUIRED = 1
      @COMPLETE_SYNTAX = %SNA_TYPE_A%
   )
)

*The end of Complex Keyword USERID_PASSWORD
```

USERID_PASSWORD のサンプル: USERID_PASSWORD キーワードのサンプルは、以下のとおりです。

```
USERID_PASSWORD=(
   USER_ID=MYUSER
   PASSWORD=A098C824DC22B856748B
)
```

ANYNET_COMMON_PARAMETERS

キーワードの定義

ANYNET_COMMON_PARAMETERS キーワードのフィールド記述については、Communications Server のオンライン・ヘルプを参照してください。

```
ANYNET_COMMON_PARAMETERS = (
   CONNWAIT_SECS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 30
   )
   CONN_RETRY_SECS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 300
   )
   DG_IDLE_TIMEOUT = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 90
   )
   INACTIVITY_TIMER_SECS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 30
   )
   SNASUFFIX = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = SNA.IBM.COM
   )
   SNA_IP_NODE_TYPE = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 1
   )
   UNACKED_DG_RETRY_SECS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 10
   )
   UNSENT_DG_RETRY_SECS = (
      @TYPE = STRINGKEYWORD
      @FIELD_LENGTH = 1,257
      @DEFAULT = 3
   )
)

*The end of Complex Keyword ANYNET_COMMON_PARAMETERS
```

ANYNET_COMMON_PARAMETERS のサンプル:

ANYNET_COMMON_PARAMETERS キーワードのサンプルは、以下のとおりです。

ANYNET_COMMON_PARAMETERS

```
ANYNET_COMMON_PARAMETERS = (
   CONNWAIT_SECS=30
   CONN_RETRY_SECS=300
   DG_IDLE_TIMEOUT=90
   INACTIVITY_TIMER_SECS=30
   SNASUFFIX=SNA.IBM.COM
   SNA_IP_NODE_TYPE=1
   UNACKED_DG_RETRY_SECS=10
   UNSENT_DG_RETRY_SECS=3
   )
```

ANYNET_SOCKETS_OVER_SNA

キーワードの定義

ANYNET_SOCKETS_OVER_SNA キーワードのフィールド記述については、Communications Server のオンライン・ヘルプを参照してください。

```
ANYNET_SOCKETS_OVER_SNA = (
   CLASSA_ADDRESS = (
      @TYPE = STRINGKEYWORD
   )
   DEFAULT_MODE = (
      @TYPE = STRINGKEYWORD
   )
   DOMAIN_NAME = (
      @TYPE = STRINGKEYWORD
   )
   DOMAIN_NAME_SERVER_ADDRESS = (
      @TYPE = STRINGKEYWORD
      @MERGE_SIMPLE_KEYWORDS = 0
   )
   GW_ADAPTER_CONFIG_REQUIRED = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = NO
   )
   HOST_NAME = (
      @TYPE = STRINGKEYWORD
   )

INTERFACE = (
   INTERFACE_NAME = (
      @TYPE = STRINGKEYWORD
   )
   IP_ADDRESS = (
      @TYPE = STRINGKEYWORD
   )
   SUBNET_MASK = (
      @TYPE = STRINGKEYWORD
   )
)

*The end of Complex Keyword INTERFACE
IP_TO_LU_MAPPING = (
   IP_ADDRESS = (
      @TYPE = STRINGKEYWORD
   )
   LU_NAME = (
      @TYPE = STRINGKEYWORD
   )
   MAPPING_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         GENERATED = 0
         EXPLICITE = 1
      )
      @DEFAULT = GENERATED
```

```
   )

   NETID = (
      @TYPE = STRINGKEYWORD
   )
   SUBNET_MASK = (
      @TYPE = STRINGKEYWORD
   )
)

*The end of Complex Keyword IP_TO_LU_MAPPING
PORT_TO_MODE_MAPPING = (
   MODE_NAME = (
      @TYPE = STRINGKEYWORD
   )
   PORT_NUMBER = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
   )
)
*The end of Complex Keyword PORT_TO_MODE_MAPPING
ROUTE_ENTRY = (
   DESTINATION_ADDRESS = (
      @TYPE = STRINGKEYWORD
   )
   DESTINATION_MASK = (
      @TYPE = STRINGKEYWORD
   )
   DIRECT_CONNECTION = (
      @TYPE = BOOLEANKEYWORD
      @DEFAULT = 0
   )
   ROUTER_ADDRESS = (
      @TYPE = STRINGKEYWORD
   )
   ROUTE_TYPE = (
      @TYPE = ENUMKEYWORD
      @ENUM_LIST = (
         DEFAULT = 0
         HOST = 1
         NETWORK = 2
      )
    )
)

*The end of Complex Keyword ROUTE_ENTRY
)

*The end of Complex Keyword ANYNET_SOCKETS_OVER_SNA
```

ANYNET_SOCKETS_OVER_SNA のサンプル: ANYNET_SOCKETS_OVER_SNA キーワードのサンプルは、以下のとおりです。

```
ANYNET_SOCKETS_OVER_SNA = (
   CLASSA_ADDRESS=125.0.0.0
   DEFAULT_MODE=BLANK
   GW_ADAPTER_CONFIG_REQUIRED=0
   INTERFACE=(
```

```
      INTERFACE_NAME=sna0
      IP_ADDRESS=9.37.54.3
      SUBNET_MASK=255.0.0.0
   )
   IP_TO_LU_MAPPING=(
      IP_ADDRESS=9.37.54.3
      LU_NAME=ANY
      MAPPING_TYPE=GENERATED
      NETID=USIBMNM
      SUBNET_MASK=255.0.0.0
   )
   PORT_TO_MODE_MAPPING = (
      MODE_NAME=#BATCH
      PORT_NUMBER=5
   )
   ROUTE_ENTRY=(
      DESTINATION_ADDRESS=0.0.0.0
      DESTINATION_MASK=0.0.0.0
      DIRECT_CONNECTION=0
      ROUTER_ADDRESS=9.67.10.3
      ROUTE_TYPE=DEFAULT
   )
)
```

VERIFY

キーワードの定義

VERIFY キーワードは製品の構成に必要です。ユーザーはこのキーワードを修正したり、削除したりしてはいけません。

```
VERIFY = (
   CFG_MODIFICATION_LEVEL = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,100
   )
   CFG_VERSION_LEVEL = (
      @TYPE = UNSIGNEDNUMBERKEYWORD
      @RANGE = 0,10
   )
)

*The end of Complex Keyword VERIFY
```

VERIFY のサンプル: VERIFY キーワードのサンプルは、以下のとおりです。

```
VERIFY=(
   CFG_MODIFICATION_LEVEL=12
   CFG_VERSION_LEVEL=1
)
```

サポートされているキーワードに対するその他の Verb 構造: 以下の verb は、これまで本書に記載されていませんが、本章では参照されます。

START_NODE

START_NODE verb は、指定された CP 作成パラメーターを使用してノードを開始します。

注: この記述に含まれない、他の予約済みフィールドもあります。

VCB 構造

```
typedef struct start_node
{
        unsigned short  opcode;             /* verb operation code            */
        unsigned char   reserv2;            /* reserved                       */
        unsigned char   format;             /* format                         */
        unsigned short   primary_rc;        /* primary return code            */
        unsigned long    secondary_rc;      /* secondary return code          */
        CP_CREATE_PARMS cp_create_parms;    /* mode name                      */
} START_NODE;
typedef struct cp_create_parms{
        unsigned char   node_type;          /* node type                      */
        unsigned char   fqcp_name[17];      /* fully qualified CP name        */
        unsigned char   cp_alias[8];        /* CP alias                       */
        unsigned char   node_id[4];         /* node ID                        */
        unsigned char   reg_with_nn;        /* register resources with NNS    */
        unsigned char   reg_with_cds;       /* resource registration with CDS */
        unsigned char   hpr_support;        /* level of HPR support           */
        unsigned char   discovery_support;  /* is Discovery supported?        */

unsigned char   discovery_group_name[8];    /* Group name for Discovery       */
        unsigned char   default_preference; /* default routing preference     */
        unsigned char   anynet_supported;   /* level of AnyNet support        */
} CP_CREATE_PARMS;
```

提供パラメーター

アプリケーションは以下のパラメーターを指定します。

opcode

AP_START_NODE

format

VCB の形式を示します。 先にリストした VCB のバージョンを指定するには、このフィールドをゼロに設定します。

cp_create_parms.node_type

以下のノード・タイプのいずれか 1 つ。

AP_END_NODEAP_NETWORK_NODEAP_LEN_NODE

cp_create_parms.fqcp_name

ノードの完全修飾の制御点名（長さ 17 バイト）。 EBCDIC ピリオドで連結された 2 つのタイプ A の EBCDIC 文字ストリングで構成され、右側は EBCDIC スペースで埋められます。 （各名前は組込みスペースなしで最大 8 バイトまで指定できます。）

cp_create_parms.cp_alias
ローカルで使用される CP の別名。 これは、ローカルで表示可能な文字セットの 8 バイト・ストリングです。 8 バイトすべてが有効で、8 バイトすべてを設定しなくてはなりません。

cp_create_parms.reg_with_nn
エンド・ノードのみ。 資源をネットワーク・ノード・サーバーで登録するかどうかを指定します (AP_YES または AP_NO)。 このフィールドが AP_YES に設定されている場合、エンド・ノードのネットワーク・ノード・サーバーはエンド・ノードに方向づけられた lacate のみを転送します。 設定されていない場合、ネットワーク・ノード・サーバーはエンド・ノードにすべての同報通信の検索を転送します。 登録の失敗は START_NODE verb の正常終了に影響しません。

cp_create_parms.reg_with_cds
エンド・ノード。 ネットワーク・ノード・サーバーがエンド・ノード資源を中央ディレクトリー・サーバーで登録してもよいかどうかを指定します (AP_YES または AP_NO)。 (**reg_with_nn** が AP_NO に設定されている場合、このフィールドは無視されます。)

ネットワーク・ノード。 ローカル資源または定義域資源を中央ディレクトリー・サーバーで任意に登録することができるかどうかを指定します (AP_YES または AP_NO)。 登録の失敗は START_NODE verb の正常終了に影響しません。

cp_create_parms.hpr_support
ノードによって提供される HPR に対するサポートのレベルを指定します (AP_NODE、AP_BASE または AP_RTP)。

cp_create_parms.discovery_support
ディスカバリー機能がこのノードによって使用されるかどうかを指定します。

AP_NO
このノードによるディスカバリー機能の使用はありません。

AP_YES
このノードによってディスカバリー機能が使用されます。 ディスカバリーをサポートするポートだけが検索の送信および受信に使用されます。 (DEFINE_PORT 参照)

node_type が AP_END_NODE である場合、必要なときにネットワーク・ノード・サーバーへのリンクを動的に構成、および活動化するためにディスカバリー・クライアント機能が使用されます。

node_type が AP_NETWORK_NODE である場合、クライアントからの検索に対して応答するためにディスカバリー・サーバー機能が使用されます。

cp_create_parms.discovery_group_name
ノードによって使われるディスカバリー機能で使用されるグループ名を指定します。 このフィールドがすべてゼロに設定されている場合、省略時グループ名が使用されます。

cp_create_parms.default_preference

AP_USE_DEFAULT_PREFERENCE が指定されているパートナー LU へのセッションを開始するときの、望ましい経路指定の方法を指定します。（DEFINE_PARTNER_LU 参照）このフィールドは、以下の値を取ることができます。

AP_NATIVE

ネイティブ (APPN) の経路指定プロトコルだけを使用します。

AP_NONNATIVE

非ネイティブ (AnyNet) プロトコルのみを使用します。

AP_NATIVE_THEN_NONNATIVE

ネイティブ (APPN) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、非ネイティブ (AnyNet) プロトコルを使用して、セッション活動化を再試行してください。

AP_NONNATIVE_THEN_NATIVE

非ネイティブ (AnyNet) プロトコルを試行してみて、パートナー LU の位置を調べることができなかった場合には、ネイティブ (APPN) プロトコルを使用して、セッション活動化を再試行してください。

注: 最後の 3 つの値は、AnyNet DLC がノード・オペレーター機能に対して使用可能で、AnyNet リンク・ステーションが定義されている場合のみ、意味があります（Defined_LS を参照してください。）

cp_create_parms.anynet_supported

Anynet DLC に対するサポートのレベルを指定します。 このフィールドは、以下の値を取ることができます。

AP_NONE

ANYNET 機能はサポートされていません。 **default_preference** フィールドは AP_NATIVE という値をとらなくてはなりません。

AP_ACCESS_NODE

このノードは ANYNET アクセス・ノード機能をサポートします。

AP_GATEWAY

このノードは ANYNET ゲートウェイ機能をサポートします。 この値は **node_type** が AP_NETWORK_NODE の場合のみ有効です。

戻りパラメーター

verb が正常に実行されると、Communications Server は次のパラメーターを戻します。

primary_rc

AP_OK

パラメーター・エラーが原因で verb が実行されなかった場合、Communications Server は以下のパラメーターを戻します。

primary_rc

AP_PARAMETER_CHECK

secondary_rc

AP_INVALID_ISR_THRESHOLDS

AP_INVALID_CP_NAME
AP_INVALID_NODE_TYPE
AP_INVALID_PU_CONC_NOT_SUPPORTED
AP_INVALID_DLUR_NOT_SUPPORTED
AP_INVALID_HPR_NOT_SUPPORTED
AP_INVALID_ANYNET_NOT_SUPPORTED

ノードが停止中であるために verb が実行されない場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_NODE_STOPPING

システム・エラーが原因で verb が実行されなかった場合、Communications Server は次のパラメーターを戻します。

primary_rc

AP_UNEXPECTED_SYSTEM_ERROR

付録A. IBM APPN MIB テーブル

以下のテーブルは、IBM APPN 管理情報ブロック (MIB) からのテーブルの使用に関する詳細を示しています。これは RFC1593 によって定義されているとおりです。テーブルは、以下のこと定義しています。

- 各 MIB テーブルを使用するのに使われるノード・オペレーター機能の QUERY verb
- 入力パラメーターの設定値
- 要求された任意のフィルター操作

戻りパラメーターと MIB テーブル変数との間のマッピングは、ノード・オペレーター機能の QUERY verb の定義から導くことができます。 Communications Server は、現在、ibmappnNodePortDlcTraceTable テーブルと ibmappnLsStatusTable MIB テーブルをサポートしていません。

IBM MIB テーブル	**ノード・オペレーター機能 Verb と MIB テーブル変数**	**入力パラメーターの設定値**
ibmappnNodePortTable	QUERY_PORT	**port_name** ibmappnNodePortName
ibmappnNodePortIpTable	（注 1)	
ibmappnNodePortDlsTable	QUERY_PORT （AP_SDLC の **dlc_type** で項目を選択する）	**port_name** ibmappnNodePortDlsName
ibmappnNodePortTrTable	QUERY_PORT	**port_name** ibmappnNodePortTrName
ibmappnNodeLsTable	QUERY_LS	**ls_name** ibmappnNodeLsName
ibmappnNodeLsIpTable	（注 1)	
ibmappnNodeLsDlsTable	QUERY_LS （AP_SDLC の **dlc_type** で項目を選択する）	**ls_name** ibmappnNodeLsDlsName
ibmappnNodeLsTrTable	QUERY_LS	**ls_name** ibmappnNodeLsTrName
ibmappnNnTopoRouteTable	QUERY_COS	**cos_name** ibmappnNnTopoRouteCos
ibmappnNnAdjNodeTable	QUERY_ADJACENT_NN	**adj_nncp_name** ibmappnNnAdjNodeAdjName
ibmappnNnTopologyTable	QUERY_NN_TOPOLOGY_NODE	**node_name** ibmappnNnNodeName **node_type** AP_LEARN_NODE **frsn** 0
ibmappnNnTgTopologyTable	QUERY_NN_TOPOLOGY_TG	**owner** ibmappnNnTgOwner **owner_type** AP_LEARN_NODE **dest** ibmappnNnTgDest **dest_type** AP_LEARN_NODE **tg_num** ibmappnNnTgNum **frsn** 0

IBM MIB テーブル	ノード・オペレーター機能 Verb と MIB テーブル変数	入力パラメーターの設定値
ibmappnNnTopologyFRTable	QUERY_NN_TOPOLOGY_NODE	**node_name** ibmappnNnFRNode **node_type** AP_LEARN_NODE **frsn** ibmappnNnFRFrsn
ibmappnNnTgTopologyFRTable	QUERY_NN_TOPOLOGY_TG	**owner** ibmappnNnTgFROwner **owner_type** AP_LEARN_NODE **dest** ibmappnNnTgFRDest **dest_type** AP_LEARN_NODE **tg_num** ibmappnNnTgFRNum **frsn** ibmappnNnTgFRFrsn
ibmappnLocalTgTable	QUERY_LOCAL_TOPOLOGY	**dest** ibmappnLocalTGDest **dest_type** AP_LEARN_NODE **tg_num** ibmappnLocalTgNum
ibmappnLocalEnTable	QUERY_LOCAL_TOPOLOGY（固有な **dest** を使用して項目を選択する）（注 2）	**dest** ibmappnLocalEnName **dest_type** AP_END_NODE **dest_type** AP_LEARN_NODE
ibmappnLocalEnTgTable	QUERY_LOCAL_TOPOLOGY（注 3）	**dest** ibmappnLocalEnTgOrigin **dest_type** AP_LEARN_NODE **tg_num** ibmappnLocalEnTgNum
ibmappnDirTable	QUERY_DIRECTORY_LU	**lu_name** ibmappnDirLuName
ibmappnCosModeTable	QUERY_MODE_TO_COS_MAPPING	**mode_name** ibmappnCosModeName
ibmappnCosNameTable	QUERY_COS	**cos_name** ibmappnCosName

注:

1. Communications Server は DLC タイプとして IP をサポートしません。
2. 同じ **dest** を使用した項目は、 QUERY_LOCAL_TOPOLOGY の順に配列されます。
3. ibmappnLocalEnTgTable は、接続されたエンド・ノードの見地から TG を（つまり、エンド・ノードからの TG として）表示します。しかし、APPN アーキテクチャーの現行レベルに応じるネットワーク・ノードは、 TG と直接接続されたエンド・ノードとの間の TG に対する、エンド・ノード TG 情報の記憶だけをします。 このことにより、このテーブル内のすべての項目は ibmappnLocalEnTgDest をローカル・ノードの名前 (ibmappnNodeCpName) に設定します。

用語集

この用語集には、下記の資料から引用した用語および定義が含まれています。

- *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI)。この資料は、米国規格協会(American National Standards Institute, 11 West 42nd Street, New York, New York 10036) から購入できます。この資料から転載した定義には、(A) という記号をつけて区別してあります。
- ANSI/EIA Standard--440-A, *Fiber Optic Terminology* Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006.この資料から転載した定義には、(E) という記号をつけて区別してあります。
- *Information Technology Vocabulary*（国際標準化機構および国際電気標準会議の第 1 合同専門委員会の第 1 分科委員会 (ISO/IEC JTC1/SC1) 作成)。このうちの既刊部分で定義されている用語については、(I) という記号をつけています。ISO/IEC JTC1/SC1 で発行している国際標準化草案、委員会草案、および作業書類から採用した定義には、(T) という記号をつけています。これは、SC1 の参加各国の間で最終的な合意が得られていないことを示します。
- *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994。
- Internet Request for Comments: 1208, *Glossary of Networking Terms*
- Internet Request for Comments: 1392, *Internet Users' Glossary*
- *Object-Oriented Interface Design: IBM Common User Access Guidelines* , Carmel, Indiana: Que, 1992。

この用語集では、次の参照指示語を使用しています。

～と対比
反対の意味または実質的に異なる意味をもつ用語を示します。

～の同義語
参照している用語と同義ではあるが、参照語を使用する方が望ましいことを示します。示されている優先同義語の定義もこの用語集の該当個所に示されています。

～と同義
そこで定義された用語と同じ意味をもつ他のすべての用語を示します。

～を参照
複数語から成る用語で、最後の語が同じ語の用語を示します。

～も参照
同義語ではないが、類似した意味をもつ用語を参照する場合に使用します。

～の不適語
その用語を使用すべきでないことを示します。 適切な用語の定義は、その用語の正式名の項を参照してください。

A

受入れ、受け入れる (accept). (1) VTAM アプリケーション・プログラムにおいて、システム・サービス制御点 (SSCP) からの CINIT 要求に応えて、論理装置 (LU) とのセッションを確立すること。セッション開始要求は、端末ユーザーがログオンしたとき、VTAM アプリケーション・プログラムがマクロ命令を発行したとき、または VTAM オペレーターがコマンドを発行したときに始まる。*獲得 (する)(acquire)* も参照。(2) 配布されたコードおよび MVS タイプ・プログラムを配布ライブラリーに移すための SMP プロセス。

ACCESS. 簡易ネットワーク管理プロトコル (SNMP) の管理情報ベース (MIB) モジュール内の文節。管理ノードが 1 つのオブジェクトについて提供する最低レベルのサポートを定義する。

ACF. 拡張通信機能 (Advanced Communications Function)。

ACF/VTAM. VTAM 拡張通信機能アクセス方式 (Advanced Communications Function for the Virtual Telecommunications AccessMethod)。 *VTAM* の同義語。

肯定応答 (acknowledgment). (1) 送信側への肯定的な応答として、受信側が肯定応答文字を伝送すること。(T) (2) 送信された項目が受け取られたことを示すこと。

獲得 (する) (acquire). (1) VTAM において、前に他の定義域内でアクセス方式により制御されていた資源を引き継ぐこと、または、他の定義域内で制御されていたが現在は解放されている資源を再び制御すること。*解放 (する)(release)* と対比。*資源引継ぎ (resource takeover)* も参照。(2) VTAM アプリケーション・プログラムにおいて、他の論理装置 (LU) とのセッションを開始し確立すること。獲得プロセスは、アプリケーション・プログラムがマクロ命令を発行したときに始まる。*受入れ、受け入れる (accept)* も参照。

活動化 (する) (activate). 資源の機能が実行されように資源を準備すること。 *非活動化 (する)(deactivate)* と対比。

active. (1) 操作可能であること。(2) ノードまたは装置が、他のノードまたは装置に接続されているか、または接続できる状態にあること。(3) 資源が活動化され作動可能になっているときの状態。

活動アプリケーション (active application). 現在、端末ユーザーとの拡張回復機能 (XRF) セッション中であるアプリケーション・サブシステム。*代替アプリケーション (alternate application)* を参照。

ACTLU. 論理装置活動化 (Activate logical unit)。SNA において、物理装置でのセッションを開始するために使用するコマンド。

ACTPU. 物理装置活動化。 SNA において、物理装置でのセッションを開始するために使用するコマンド。

アダプター（adapter）. (1) SDLC、LAN、非同期通信接続機構、DFT、または他の通信接続機構に接続するために（おそらくモデムを介して）、パーソナル・コンピューターに導入しなければならないハードウェア構成要素。(2) 装置をコンピューターまたは他の装置に電気的または物理的に接続する部品。

適応ペーシング (adaptive pacing). *最適セッション・レベル・ペーシング (adaptive session-levelpacing)* の同義語。

最適セッション・レベル・ペーシング (adaptive session-level pacing). セッション・レベル・ペーシングの形式の１つ。この形式では、セッションの途上で、各種構成要素がそれぞれサイズの異なるペーシング・ウィンドウを交換し合う。 その結果、ネットワーク内の伝送を、セッション単位で動的にバッファーの可用度と需要の変化に適応させることができる。 セッション・パスに沿った個々の段階で、中間ノードおよび端点ノードにおけるローカルの輻輳（ふくそう）に応じて、セッション・レベル・ペーシングが行われる。 *適応ペーシング (adaptive pacing)* および*最適セッション・ペーシング (adaptivesession pacing)* と同義。 *固定セッション・レベル・ペーシング(fixed session-level pacing)* と対比。

最適セッション・ペーシング (adaptive session pacing). *最適セッション・レベル・ペーシング (adaptive session-levelpacing)* の同義語。

アドレス (address). データ通信では、ネットワークに接続された各装置、ワークステーション、またはユーザーに割り当てられた固有のコード。

アドレス空間 (address space). (1) ノードが接続している各ネットワークごとに、ノードのネットワーク・アクセス可能単位、セッション、隣接リンク・ステーション、およびリンクを固有に識別するために使用される一連のアドレス。APPN ノードには、イントラノード経路指定用のアドレス空間 1 つと、メッセージ単位を送信できる各伝送グループ用のアドレス空間 1 つがある。(2) MPTN アーキテクチャーにおいて、与えられたアドレス・タイプの規則に従って生成される、すべての正当なトランスポート・アドレスの集まり。 これらの規則には、アドレスの最大文字数および許可される文字の定義を含む。各トランスポート・プロトコルには、独自の規則がある。 また、あるプロトコルにおけるアドレスは別のプロトコルでも正当であり得るので、MTPN はすべてのトランスポート・アドレスをアドレス・タイプで修飾する。

隣接リンク・ステーション (adjacent link station (ALS)). (1) SNA において、ネットワーク・トラフィックを運ぶことができる、リンク接続により、与えられたノードに直接接続されるリンク・ステーション。

注: リンク接続を共用する２次側リンク・ステーションは、相互にデータを交換しないため、隣接しているとはみなされない。(2) 特定のノードに関して、隣接ノードにおけるリンク・ステーション・パートナーのこと。

隣接ノード (adjacent nodes). 他のノードと接続されていない、少なくとも１つのパスで互いに接続された２つのノード。 (T)

管理定義域 (Administrative Domain). 1 つの管理権限により管理される、ホストおよびルーターの集合、および内部接続ネットワーク。

拡張通信機能 (Advanced Communications Function (ACF)). 機能の分散および資源の共用を含む、システム・ネットワーク体系 (SNA)の概念を使用した IBM ライセンス・プログラムのグループ (主に、VTAM、TCAM、NCP、および SSP)。

拡張対等通信ネットワーキング (Advanced Peer-to-Peer Networking (APPN)). SNA に対する拡張機能の 1 つで、次のような特色を備えている。(a) 分散ネットワーク制御能力が強化された結果、重大な階層依存関係がなくなるため、単一個所の障害を切り離して影響をその個所だけに限定できる。(b) ネットワーク・トポロジー情報が動的に交換され、接続、再構成、および最適経路選択が容易になる。(c) ネットワーク資源が動的に定義できる。(d) リソース登録とディレクトリー・ルックアップが自動的に行われる。 APPN は、エンドユーザー・サービス用の LU 6.2 対等オリエンテーションをネットワーク制御にまで拡張し、LU 2、LU 3、LU 6.2 など複数の LU タイプをサポートする。

拡張対等通信ネットワーキング (APPN) エンド・ノード (Advanced Peer-to-Peer Networking (APPN) end node). 幅広いエンドユーザー・サービスを提供し、自身のローカル制御点 (CP) と隣接ネットワーク・ノード内の CP との間のセッションをサポートする。 このエンド・ノードは、自身の資源を隣接 CP (エンド・ノードのネットワーク・ノード・サーバー) に動的に登録し、ディレクトリー検索要求を送受信し、そして管理サービスを取得する。 APPN エンド・ノードは他のエンド・ノードに接続されることもある。

拡張対等通信ネットワーキング (APPN) ネットワーク (Advanced Peer-to-Peer Networking (APPN) network). 相互に接続されたいくつかのネットワーク・ノードとそれぞれのクライアント・エンド・ノードの集合。

拡張対等通信ネットワーキング (APPN) ネットワーク・ノード (Advanced Peer-to-Peer Networking (APPN) network node). 広範なエンドユーザー・サービスを提供し、次の機能を備えているノード。

- 分散ディレクトリー・サービス。たとえば、中心となるディレクトリー・サーバーへの定義域リソースの登録など。
- 他の APPN ネットワーク・ノードとのトポロジー・データベースの交換。これによって、ネットワーク全域に散在するネットワーク・ノードが、要求されたサービス・クラスに応じて、LU-LU セッション用の最適な経路を選択できるようになる。
- ローカル LU およびクライアント・エンド・ノード用のセッション・サービス。
- APPN ネットワーク内での中間経路指定サービス。

拡張対等通信ネットワーキング (APPN) ノード (Advanced Peer-to-Peer Networking (APPN) node). APPN ネットワーク・ノードまたは APPN エンド・ノード。

拡張プログラム間通信 (advanced program-to-program communication (APPC)). (1) (2) 相互接続されたシステムがプログラミング・タスクを共用できるようにする、SNA の LU 6.2 論理装置プロトコルの実施形態。LU 6.2 アーキテクチャー、および各種製品での同アーキテクチャーの実装を特徴づけている一般機能。(3) LU 6.2 アーキテクチャーとそれを実装する製品を包括的に表すこともあり、また、APPC アプリケーション・プログラミング・インターフェースなどのように、LU 6.2 製品の特定の機能を限定的に表すこともある。

AIX. 拡張対話式エグゼクティブ (Advanced Interactive Executive)。

alert. (1) 問題が起こったこと、または起こりそうな状態にあることを示すために、ネットワーク内の管理サービス・フォーカル・ポイントに送られるメッセージ。(2) SNA 管理サービス (SNA/MS) において、即時アテンションの正当な理由となる高優先順位の事象。

アラート・フォーカル・ポイント (alert focal point). アラートを受け取り、処理する（ログ、表示、および任意で転送する）ネットワーク内のシステム。 アラート・フォーカル・ポイントは、問題管理フォーカル・ポイントのサブセットである。

割振り、割り振る (allocate). (1) セッションを会話に割り当てて、そのセッションをその会話に使用させるための LU 6.2 アプリケーション・プログラミング・インターフェース (API) verb。(2) *割振り解除 (deallocate)* と対比。

全点アドレス可能 (all points addressable (APA)). コンピューター・グラフィックスにおいて、各画素(ペル)を表示画面上でアドレスし、その画素を表示するか表示しないかを決める能力に関する用語。

検査済みインディケーター (Already-Verified indicator). 2 つの論理装置間のセッションで会話要求が、ユーザー ID と一緒に送信されるが、ID パスワード確認がすでに検査されているためパスワードは送信されないことを示す、接続機能管理ヘッダー内の指示。

ALS. 隣接リンク・ステーション (Adjacent link station)。

代替アプリケーション (alternate application). アプリケーションが失敗した場合に特定の活動アプリケーションの拡張回復機能 (XRF) が行っている端末ユーザーとのセッションを引き継ぐために用意されているサブシステム。*活動アプリケーション (active application)* を参照。

AND 演算 (AND operation). *論理積 (conjunction)* の同義語。

ANR. 自動ネットワーク経路指定 (Automatic network routing)。

不特定モード (any-mode). VTAM において、以下のことを指す。

- 任意の（不特定の）セッションからの入力を得るための RECEIVE 要求の形式。
- 任意の（不特定の）待機 CINIT を受け入れることによって、セッションの確立を完了する ACCEPT 要求の形式。

特定モード (specific-mode) と対比。*継続不特定モード (continue-any mode)* を参照。

AP. 代替プリンター (Alternate printer)。

APA. 全点アドレス可能 (all points addressable)。

API. アプリケーション・プログラミング・インターフェース (applicationprogramming interface)。

APPC. 拡張プログラム間通信 (advanced program-to-program communication).

APPL. アプリケーション・プログラム (Application program)。

application. ユーザー指向の特殊な作業をコンピューター上で実行するために使用されるソフトウェアの集合。

アプリケーション・プログラム (application program). (1) 在庫管理や給与計算を行うプログラムのように、ユーザー用に作成されるか、ユーザーが作成するプログラムでそのユーザーの作業に適用されるもの。(2) ネットワーク内のステーションを接続しそれと通信するために使用されるプログラムであって、ユーザーにアプリケーション指向の活動を実行させるもの。

アプリケーション・プログラミング・インターフェース (application programming interface (API)). (1) (2) IBM システム制御プログラムまたは IBM ライセンス・プログラムとプログラム・ユーザーとの間の、定義されたプログラミング言語インターフェース。ベースとなっているオペレーティング・システムまたはサービス・プログラムが提供する特定の機能およびサービスを利用するために、アプリケーション・プログラムの中でコーディングできるプログラム言語の構成およびステートメントのセット。(3) VTAM において、アプリケーション・プログラムが制御ブロックを参照し、また VTAM に対してそれ自体を識別させるように制御ブロックの中で使用される言語構造。

アプリケーション・トランザクション・プログラム (application transaction program). ユーザーのアプリケーションを処理するために、ユーザーのために、またはユーザーによって書かれたプログラム。SNA ネットワークにおいて、タイプ 6.2 論理装置のユーザー。サービス・トランザクション・プログラム *(servicetransaction program)* と対比。

適用 (Apply). ウィンドウ内で選択した選択項目を、そのウィンドウをクローズせずに実行する押しボタン。

APPN. 拡張対等通信ネットワーキング (Advanced Peer-to-Peer Networking)。

APPN エンド・ノード (APPN end node). *拡張対等通信ネットワーキング (APPN) エンド・ノード (Advanced Peer-to-Peer Networking (APPN) end node)*を参照。

APPN ネットワーク (APPN network). *拡張対等通信ネットワーキング (APPN) ネットワーク(Advanced Peer-to-Peer Networking (APPN) network)*を参照。

APPN ネットワーク・ノード (APPN network node). *拡張対等通信ネットワーキング (APPN) ネットワーク・ノード (Advanced Peer-to-Peer Networking (APPN) network node)*を参照。

APPN ノード. *拡張対等通信ネットワーキング・ノード (Advanced Peer-to-PeerNetworking (APPN) node)* を参照。

引数 (argument). 呼出しプログラムと呼び出されるプログラムとの間で渡されるパラメーター。

ASCII (情報交換用米国標準コード) (American National Standard Code for Information Interchange). 7 ビット・コード化文字 (パリティー検査を含めると 8 ビット) によって構成されるコード化文字セットを使用した標準コード。このコードはデータ処理システム、データ通信システム、および関連した機器の間の情報交換に使用される。 ASCII セットは、制御文字とグラフィック文字から成る。 (A)

非同期 (asynchronous (ASYNC)). (1) 共通タイミング信号のように、特定のイベントの発生に依存しない複数のプロセスに関連した用語。 (T)(2) 規則的な時間関係が存在しないこと。プログラム命令の実行に関して予測不能または予期不能であること。

非同期操作 (asynchronous operation). ソフトウェアまたはハードウェアの同時操作。ソフトウェアにおいて、操作の実行時にアプリケーション・プログラムが実行を続けることができる、セッション確立やデータ転送の要求などの操作。アクセス方式は、操作が完了した後にアプリケーション・プログラムに通知する。*同期操作 (synchronous operation)* と対比。

非同期要求 (asynchronous request). VTAM において、非同期操作の要求。*同期要求 (synchronous request)* と対比。

非同期転送モード (asynchronous transfer mode (ATM)). 情報がセルに編成される転送モード。それは、個々のユーザーからの情報を含むセルの回帰参照が必ずしも定期的でないという意味で非同期である。

ATM. 非同期転送モード (Asynchronous transfer mode)。

タスク生成する、接続する、付加する (attach). (1) プログラミングにおいて、メインライン・コードの実行とは非同期的に実行できるタスクを作成すること。(2) 装置をリング・ネットワークへ論理的に接続すること。

接続機構(attachment). Communications Serverが使用する通信リンクで、アダプターと制御ソフトウェアから構成され、PC とホスト・システムを接続する。

自動ネットワーク経路指定 (automatic network routing (ANR)). 高性能経路指定機能 (HPR) における効率の高い経路指定プロトコルであり、経路上の中間ノードを通してネットワーク層パケットを経路指定するためのサイクル要件と記憶要件を最小限に抑えることができる。

自動タスク (autotask). (1) 端末またはログオンしているユーザーを必要としない不在時 NetView 操作員ステーション・タスク。自動タスクは VTAM から独立して実行でき、一般に自動コンソール操作用に使用される。(2) *ログオン操作員 (logged-on operator)* と対比。

使用可能 (available). VTAM において、活動化され、接続され、使用可能にされていて、しかもセッション限度に達していない論理装置を表す。

使用可能時間 (available time). ユーザーの視点から、機能単位が使用できる時間のこと。 (I) (A)

B

バック・レベル (back-level). 特定の現行機能をサポートしていない可能性のある、IBM 製品の以前のリリースに関する用語。

バックアップ・フォーカル・ポイント (backup focal point). 1 次フォーカル・ポイントでの通信障害の際に、特定のカテゴリーの管理サービス・サポートをノードに提供するフォーカル・ポイント。割り当てられたフォーカル・ポイント（明示と暗黙）および省略時フォーカル・ポイントは両方ともバックアップ・フォーカル・ポイントを持つことができる。*1 次フォーカル・ポイント (primary focal point)* と対比。

基本会話 (basic conversation). 割振りトランザクション・プログラムにより指定される LU 6.2 会話タイプ。基本会話を使用するトランザクション・プログラムはより広い LU 6.2機能を使用できるようになるが、自己のエラー回復に責任があり、会話で使用されるデータ・ストリームの細部を管理しなければならない。 マップ式*会話 (mapped conversation)* と対比。

基本伝送単位 (basic transmission unit (BTU)). (1) SNA において、パス制御構成要素間でやり取りされるデータと制御情報の単位。BTU は、1 つまたは複数のパス情報単位 (PIU) からなる。(2) *PIU のブロック化 (blocking of PIUs)* も参照。

送信権要求 (bid). 競合形式の送信勧誘または選択において、データを送信するために回線の制御権を得るようコンピューターまたはステーションが行う試行。

送信権要求元 (bidder). *送信権要求元セッション (bidder session)* を参照。

送信権要求元セッション (bidder session). ブラケットを開始するために他のハーフセッションからの許可を要求し受け取らなければならない、セッション活動化時に定義されたハーフセッション。ファースト・スピーカー・

セッション *(first-speaker session)* と対比。 *競合セッション (contention-loser session)* の同義語。

2 値、2 進法 (binary). 基数 2 の数体系に関する用語。2 進数字は 0 と 1 である。実行可能ファイルは、通常、文字ストリング形式ではなく 2 進形式である。テキスト・ファイルは文字ストリング形式である。

BIND. SNA において、2 つの論理装置 (LU) の間でセッションを活動化するための要求。*セッション活動化要求 (session activation request)* も参照。*UNBIND* と対比。

BIND ペーシング (BIND pacing). あるノードのアドレス空間マネージャー (ASM) が別のノードの送信 ASM の BIND 要求の伝送の速度を制御する手法。 BIND ペーシングは、2 つのノードのそれぞれが他のノードを介して開始しようとし、他のノードから受け取ったすべての BIND を拒否する、セッションの資源のほとんどを予約する、BIND スタンドバイを妨げるために使用することができる。

bis. ドイツ（連邦共和国）基本標準規格。

ビット (bit). 2 進数体系で使用される数字 0 または 1。 (T)

BIU セグメント (BIU segment). SNA において、パス情報単位 (PIU) の中に含まれる基本情報単位 (BIU) 部分。 (a) 要求応答ヘッダー (RH) の後に要求応答単位 (RU) の全体または一部を続けたものか、または、(b) RU の一部から成っている。セグメント *(segment)* と同義。

ブロック (block). 1 単位として記録または転送されるデータ要素のストリング。 要素は、文字、語、または物理レコードのいずれでもよい。 (T)

PIU のブロック化 (blocking of PIUs). SNA において、複数のパス情報単位 (PIU) を単一の基本伝送単位 (BTU) に結合する、パス制御の任意の機能。

注: ブロック化が行われない場合、BTU は 1 つの PIU からなる。

境界機能 (boundary function (BF)). (1) SNA において、接続された周辺ノードにプロトコル・サポートを提供するためのサブエリア・ノードの機能。たとえば、(a) サブエリア・パス制御と周辺パス制御要素の連結、(b)低機能周辺ノードのセッション順序番号付けの実行、(c)セッション・レベル・ペーシング・サポートの提供。(2) SNA において、これらの機能を提供する構成要素。

ブラケット・プロトコル (bracket protocol). SNA において、ブラケットを使用して 2 つのセッション・パートナー間の交換を行うデータ・フロー制御プロトコル。セッション活動化の時点で一方のパートナーがファースト・スピーカーとして指定され、もう一方のパートナーが送信権要求者として指定される。 ブラケット・プロトコルは、ブラケット開始および終了規則に従って働く。

同報通信 (broadcast). (1) すべてのあて先に同じデータを送信すること。 (T)(2) 2 つ以上のあて先にデータを同時に送信すること。(3) マルチキャスト *(multicast)* と対比。

同報通信 Locate 検索 (broadcast Locate search). *同報通信検索 (broadcast search)* の同義語。

同報通信検索 (broadcast search). APPN ネットワークのすべてのネットワーク・ノードに検索要求を同時に伝搬すること。 このタイプの検索は、資源の位置がリクエスターに知られていない場合に使用されることがある。 *方向づけ検索 (directed search)* と対比。*同報通信 Locate 検索 (broadcast Locate search)* と同義。

BTU. 基本伝送単位 (Basic transmission unit)。

buffer. (1) 1 つの装置から他の装置にデータを転送するときに、データ・フロー速度の差異、またはイベント発生時間の差異を補償するために使用されるルーチンまたは記憶域。 (A) (2) 入力または出力データを一時的に保管するために使用される記憶域の部分。

バス (bus). (1) 2 つのエンド・ポイント間に位置するいくつかの装置間でデータを転送するための機能。 1 時点で 1 つの装置だけが伝送を行うことができる。 (T)(2) プロセッサーが直列的に相互接続されているコンピューター構成。

バイパス (bypass). ステーションまたはアクセス単位を避けて通るパスをデータが流れるようにすることによって、環状ネットワークからステーションまたはアクセス単位を削除すること。

バイト（byte）. (1) いくつかのビットからなるストリングであって、1 単位として取り扱われて 1 つの文字を表すもの。 (T) (2) 1 単位として取り扱われ、通常、1 つのコンピューター・ワードよりも短い 2 進文字。 (A) (3) 1 つの EBCDIC 文字を表す、8 個の隣接した 2 進数より成るグループ。

C

キャッシュ (cache). (1) 主記憶装置から入手され、プロセッサーが次に必要とするであろう命令とデータのコピーを保管するために使用される、主記憶装置より小さくて高速な、特定目的のバッファー記憶装置。 (T) (2) 頻繁にアクセスされる命令およびデータを入れるバッファー記憶装置。アクセス時間を減らすために使用される。(3) ディレクトリーの検索を速めるため、頻繁に使用されるディレクトリー情報が保管される、ネットワーク・ノード内のディレクトリー・データベースの任意の部分。(4) キャッシュ内に置いたり、隠したり、保管したりすること。

コール (call). (1) コンピューター・プログラム、ルーチン、またはサブルーチンを実効化するアクション。この実効化は、通常、入口条件を指定して入口点にジャンプすることによって行われる。 (I) (A) (2) データ通信において、交換回線上の 2 つのステーションの間で接続を行うために必要なアクション。(3) 通信において、2 人のユーザーの間の会話。(4) プロシージャー、プログラム、ルーチン、またはサブルーチンへ制御を移すこと。(5) ユーザーと連絡をとるための試み。その試みの成功または不成功を問わない。

呼出し、呼出し側（calling）. (1) データ・ステーション間の接続を確立するために選択信号を伝送するプロセス。 (I) (A) (2) X.25 通信において、呼出しを行うロケーションまたはユーザーを表す。

取消 (Cancel). ウィンドウに何の変更も加えないで、そのウィンドウを削除する押しボタン。

カード (card). (1) システム装置のスロットに差し込む、電子回路ボード。(2) プラグイン回路アセンブリー。(3) アダプター *(adapter)* も参照。(4) NetView for AIX では、*事象カード (event card)* を参照。

大文字・小文字の区別 (case-sensitive). 大文字と小文字の判別を行う機能のことを指す。

CDS. (1) 制御データ・セット (Control data set)。(2) 構成データ・セット。(3) 中央ディレクトリー・サーバー (Central directory server)。

中央ディレクトリー (central directory). ネットワーク・ノードにより中央に登録されるかネットワーク検索の結果がキャッシュされる、資源所在情報を保管するためのリポジトリー。

中央ディレクトリー・サーバー (central directory server (CDS)). ネットワーク資源の位置に関する情報のためのリポジトリーを提供するネットワーク・ノード。また、照会および同報通信検索のためのフォーカル・ポイントを提供し、後で同じ情報を同報通信するのを避けるためにネットワーク検索の結果をキャッシングすることによって、ネットワーク検索の数を減らす。

連鎖 (chain). (1) LU 6.2 によって処理される、論理的にリンクされたユーザー・データ・レコードのグループ。(2) 開始連鎖および終了連鎖によって区切られた要求単位のグループ。 応答は常に単一単位連鎖である。*RU 連鎖 (RU Chain)* を参照。

チャネル（channel）. (1) シグナルを送信するパス。データ・チャネル、出力チャネルなどがある。 (A) (2)データ通信における、片方向伝送の手段。(3) プロセッサーによって制御される機能単位であって、主記憶装置とローカル周辺装置との間でデータ転送を処理するもの。

チャネル接続 (channel-attached). (1) 装置を直接に入出力チャネルによってホスト・プロセッサーに接続することに関する用語。(2) 通信回線ではなくケーブルによって装置を制御装置に接続することに関する用語。(3)*リンク接続 (link-attached)* と対比。(4) *ローカル (local)* と同義語。

文字セット (character set). キーボードまたは出力装置のために定義された文字の、有限のグループ。

子 (child). 親資源のユーザー・リストを使用する、ファイルやライブラリーなどの保護された資源に関する用語。子資源の親資源は 1 つだけである。 子は親プロセスにより開始されるプロセスで、親プロセスの資源を共用する。*親 (parent)* と対比。

CICS. 顧客情報管理システム (Customer Information Control System)。

回線（circuit）. (1) 電流を流すことができる 1 つまたは複数の導体。 *物理回線 (physical circuit)* および*仮想回線 (virtual circuit)* を参照。(2) 1 つの論理装置。

C 言語 (C language). 最小の変更で異なるタイプのコンピューターで実行できる、簡潔で効果的なコードでアプリケーション・ソフトウェアを開発するために使用される言語。

クラス（class）. (1) オブジェクト指向の設計またはプログラミングにおいて、共通の定義を共用し、したがって共通の特性、操作、および性質を共用するオブジェク

トのグループ。このグループのメンバーをクラスのインスタンスと呼ぶ。(2) AIX オペレーティング・システムにおいて、装置の入出力特性を表す。システム装置は、ブロック装置またはキャラクター型装置のいずれかに分類される。

サービス・クラス (class of service (COS)). セッション・パートナー間に経路を構成するために使用する(経路セキュリティー、伝送優先順位、帯域幅などの)特性のセット。サービス・クラスは、セッションの起動側が指定する名前から派生する。

終結処置 (cleanup). SNA プロダクトにおいて、相手側の LU やシステム・サービス制御点 (SSCP) の参加を必要とせずに、その LU との LU-LU セッションを夕達に終了させる、SSCP が論理装置 (LU) に送るネットワーク・サービス要求。

クライアント (client). (1) サーバーから共用サービスを受ける機能単位。 (T) (2) ユーザー (user)。(3) AIX 分散ファイル・システム環境において、サーバーに従属して、サーバーにプログラムを提供したりプログラムへのアクセスを提供するシステム。(4) *リクエスター (requester)* と同義。

クライアント/サーバー (client/server). 通信において、１つのサイトに存在するプログラムが他のサイトに存在するプログラムに要求を送って応答を待つような、分散データ処理の対話モデル。 要求元のプログラムをクライアントと呼び、応答するプログラムをサーバーと呼ぶ。

CNOS. セッション数変更 (change number of sessions)。

コマンド (command). (1) 端末から出される要求であって、特定のプログラムの動作または実行が行われるように要求するもの。(2) SNA において、伝送ヘッダー (TH)、要求ヘッダー (RH)、および場合によっては要求単位 (RU) の一部で設定されるフィールドであって、アクションまたはプロトコルを開始するもの。例として、(a) LU-LU セッションを活動化するコマンドである Bind Session (セッション制御要求単位)、(b) 連鎖の最後の RU の RH の中にある命令変更標識、(c) FID4 伝送ヘッダーの中にある仮想経路リセット・ウィンドウ標識がある。

コマンド・プロンプト (command prompt). 処理したいコマンドをユーザーが入力できることを示すために表示される、文字または文字ストリング。

共通操作サービス (common operations services (COS)). SNA 管理サービスの一部で、限定リモート操作制御のための主ベクトルに関係する。

共通プログラミング・インターフェース・コミュニケーション (Common Programming Interface for Communications (CPI-C)). 多様なアプリケーション環境からの増大する需要を満たし、通信プログラミング用の業界標準としてのオープン性を達成するための機能を含む、発展的なアプリケーション・プログラミング・インターフェース (API)。CPI-C を使用すると、(a) データの送受信、(b) プログラム間の処理の同期化、および (c) パートナーへの通信エラーの通知などのプログラム間サービスにアクセスできる。

共通ユーザー・アクセス (CUA) 体系 (Common User Access (CUA) architecture). 人とワークステーションあるいは端末との間の対話についての指針。

コミュニケーション管理構成ホスト・ノード (communication management configuration host node). データ・ホストにチャネル接続する装置の制御以外のすべてのネットワーク制御機能を使用する、コミュニケーション管理構成のタイプ 5 ホスト・プロセッサー。*通信管理ホスト (communication management host)* と同義。*データ・ホスト・ノード (data host node)* と対比。

通信管理ホスト (communication management host). *コミュニケーション管理構成ホスト・ノード (communicationmanagement configuration host node)* の同義語。*データ・ホスト (data host)* と対比。

コミュニケーション・マネージャー/2 (Communications Manager/2). *Communications Server* および *Communications Server プロダクト・ファミリー (Communications Server product family)* を参照。コミュニケーション・マネージャー/2 プロダクトの機能は、CommunicationsServer プロダクトおよび Communications Server プロダクト・ファミリーに取り込まれた。

Communications Server. 接続されているシステムまたはワークステーションを2 つ以上含むアプリケーション・プログラムの開発および使用をサポートする IBM ライセンス・プログラム。 CommunicationsServer は、アプリケーション・プログラムを他のシステムまたはワークステーションに接続するためにさまざまなプロトコルを使用して、複数の並行接続を提供する。 同時に呼ばれることもあり、クライアント/サーバーおよび分散アプリケーション・プログラム用に設計されているアプリケーション・プログラミング・インターフェース (API) をサポートする。 Communications Server には、ネットワーク管理のための重要なインターフェースが含まれる。

コンパイルする (compile). (1) 高水準言語で表現されたプログラムのすべてまたは一部を、中間言語、アセンブリー言語、または機械語で表現されたコンピューター・プログラムに変換すること。 (T) (2) 機械語以外のプログラム言語で書かれたコンピューター・プログラムから機械語プログラムを作成すること。その作成作業では、プログラムの全体的な論理構造を使用するか、各記号ステートメントについて複数のコンピューター命令を生成するか、またはこれらの両方の作業を行い、さらにアセンブラーの機能を実行する。 (A) (3) ソース・プログラムを実行可能プログラム (オブジェクト・プログラム) に変換すること。(4) 高水準プログラム言語で書かれたプログラムを機械語プログラムに変換すること。

構成要素（component）. 機能単位の部分であるハードウェアまたはソフトウェア。

コンピューター (computer). さまざまな算術演算や論理演算など大量の計算を、実行中に人間の介在なしで行える機能単位。情報処理においては、コンピューターという用語は一般にディジタル・コンピューターを意味する。コンピューターは、1 つの独立した単位から成ることもあり、いくつかの相互接続された単位から成ることもある。 (T)

構成. (1) 情報処理システムのハードウェアおよびソフトウェアが編成され、相互接続される様式。 (T) (2)システム、サブシステム、またはネットワークを構成する装置またはプログラム。(3) Communications Server において、1 つまたは複数の接続タイプにより 1 つまたは複数のホスト・システムに接続されるパーソナル・コンピューターの配置。例として、 SDLC、 LAN、 ASYNCH、 X.25、 または DFT がある。

構成する (configure). システムに導入される装置、任意選択機能、およびプログラムを、そのシステムに対して記述すること。

輻輳（ふくそう）(congestion). *ネットワーク輻輳 (network congestion)*を参照。

論理積 (conjunction). 各オペランドのブール値がそれぞれ 1 である場合に限り、演算結果がブール値 1 になるようなブール演算。 (I) (A) *AND 演算 (AND operation)* と同義。

接続状態 (connected). VTAM において、ある物理装置 (PU) または論理装置 (LU) が、それぞれを制御するシステム・サービス制御点 (SSCP) を含むホスト・プロセッサーへの活動物理パスを持っている状態。

接続 (connection). (1) データ通信において、情報を伝達するために複数の機能単位の間で確立される関連付け。 (I) (A) (2) SNA において、異なったノードに存在する 2 つの論理装置 (LU) を相互にリンクして、それらの論理装置に通信を確立させるネットワーク・パス。(3) TCP/IP において、 2 つのプロトコル・アプリケーションの間で信頼性のあるデータ・ストリーム送達サービスを提供するパス。 インターネットでは、 1 つのシステムの TCP アプリケーションから他のシステムの TCP アプリケーションへと、接続が拡張される。

接続ネットワーク (connection network). (1) 共通の仮想経路指定ノードにより SATF への接続を識別するノードが、相互の接続を個々に定義することなしに通信できる、トークン・リングなどの共用アクセス転送機能 (SATF) の APPN ネットワークにおける表現。(2) *リンク接続ネットワーク (link connectionnetwork)* と同義。

接続性(connectivity). (1) システムまたは装置を、他のシステムまたは装置に、無修正のまま接続できる能力。 (T)(2) さまざまな機能単位を無修正で接続できる能力。

競合 (contention). セッションにおいて、2 つの NAU が同時に同じアクションを開始しようとする状態。たとえば、両方が半二重プロトコルでデータを送信しようとする（半二重競合）場合や、あるいはブラケットを開始しようとする（ブラケット競合）場合に起こる。 セッション開始時に、1 つの NAU が競合勝者と定義される。競合が起きたとき競合勝者のアクションが優先される。競合敗者がアクションを始めるためには、競合勝者から明示または暗黙の許可を入手しなければならない。

競合敗者セッション (contention-loser session). (1) NAU に対して、セッション開始時に競合敗者であると定義されたセッション。(2) *送信権要求元セッション (biddersession)* と同義。

競合極性 (contention polarity). セッションの使用のために競合が起きた場合の、各 LU の役割。 一方の LU が競合勝者であり、他の LU が競合 敗者である。

競合勝者セッション (contention-winner session). (1) NAU に対して、セッション開始時に競合勝者であると定義されたセッション。(2) ファースト・スピーカー・セッション *(first-speaker session)*と同義語。

継続不特定モード (continue-any mode). VTAM において、入力が不特定モードで発行された RECEIVE 要求を満たすようにするセッションまたは会話の状態。この状態の間、セッションまたは会話での入力は、特定モードで発行される RECEIVE 要求も満たすことができる。

会話の場合、継続不特定モードは、バッファー継続不特定または論理レコード継続不特定として修飾される。これは、VTAM が論理レコードまたはバッファーに関してデータを受け取ることを指定する。*継続特定モード (continue-specific mode)* と対比。

継続特定モード (continue-specific mode). VTAM において、入力が特定モードで発行された RECEIVE 要求のみを満たすようにするセッションまたは会話の状態。*継続不特定モード (continue-any mode)* と対比。

制御ブロック (control block). (1) 制御情報を保管するためにコンピューター・プログラムにより使用される記憶域。 (I) (2) IBM トークンリング・ネットワークにおいて、操作を要求するためにアプリケーション・プログラムからアダプター・サポート・インターフェースに提供される、明確に形式設定された情報のブロック。

制御データ・セット (control data set (CDS)). NPM において、NPM 導入プロセスで使用される SMP データ・セット。

制御装置 (controller). 1 つまたは複数のワークステーションなどの入出力装置を調整および制御し、それらの装置の動作とシステムの動作を全体として同期するための装置。

制御点 (control point (CP)). (1) APPN ノードまたは LEN ノードの構成要素の 1 つで、そのノードの資源を管理する。 APPN ノードでは、CP は他の APPN ノードとの CP-CP セッションに従事することができる。APPN ネットワーク・ノードでは、CP は APPN ネットワーク内の隣接エンド・ノードへのサービスも提供する。(2) ノードの構成要素の 1 つで、そのノードの資源を管理するほか、オプションとしてネットワーク内の他のノードへのサービスも提供する。 例としては、タイプ 5 のサブエリア・ノードにおけるシステム・サービス制御点 (SSCP)、APPN ネットワーク・ノードにおけるネットワーク・ノード制御点 (NNCP)、および、APPN または LEN エンド・ノードにおけるエンド・ノード制御点 (ENCP) などがある。 SSCP および NNCP は他のノードにサービスを提供できる。

制御点管理サービス単位 (control point management services unit (CP-MSU)). 管理サービスデータを含み、一連の管理サービス機能の間を流れるメッセージ単位。このメッセージ単位は、一般に汎用データ・ストリーム (GDS) 形式である。*管理サービス単位 (management services unit (MSU))* およびネットワーク管理ベクトル・トランスポート *(network managementvector transport (NMVT))* も参照。

制御プログラム (Control Program (CP)). VM/ESA において、単一のコンピューターの資源を管理して、あたかも複数のコンピューティング・システムが存在するかのような働きをさせるための構成要素。このような見せかけのシステム、つまり仮想計算機は、IBM システム/370、370-XA、または ESA などのコンピューターと機能的に同等のものとなる。

制御ベクトル (control vector). 可変長で、同じ囲み構造内に運ばれ、識別子として使用される 1 バイトのキーがある、RU 構造の一般クラスの 1 つ。

会話（conversation）. LU 6.2 セッションを使用する 2 つのトランザクション・プログラム間の論理接続。会話は、セッションの排他使用を獲得するためにブラケットで区切られる。

会話レベル・セキュリティー (conversation-level security). *セッション・レベル・セキュリティー (session-level security)*を参照。*エンド・ユーザー検証 (end-user verification)* も参照。

世界協定時 (coordinated universal time (UTC)). 国際無線通信諮問委員会 (CCIR) が定義および勧告し、国際度量衡局 (BIPM) が原子時計により維持している国際単位系 (SI) 秒を基準とする時間目盛。

注: 国際単位系は、メートル (meter)、キログラム (kilogram)、秒 (second) の 3 つの基本計測単位を基準とするもので、これらの単位の頭文字をとって『MKS 単位系』とも呼ばれる。現実的なほとんどの目的に関しては、世界協定時 (UTC) は、イギリスのグリニッジを通る本初子午線（経度 0）における平均太陽時、つまり*グリニッジ標準時 (Greenwich mean time (GMT))* に等しいものとみなされる。*Z 時 (Z time)* および*ズールー時 (Zulu time)* と同義。

複写 (Copy). 選択されたオブジェクトのコピーをクリップボードに置く選択機能。

相関子 (correlator). 物と物の間の関係を識別する情報。たとえば、対応する要求を識別する応答の変数フィールド。

COS. サービス・クラス (class of service)。

CP. (1) 制御点 (Control point)。(2) VM において、制御プログラム (Control Program)。

CP-CP セッション (CP-CP sessions). LU 6.2 プロトコルおよびモード名 CPSVCMG を使用する、2 つの制御点の間の並列セッションで、ネットワーク・サービス要求および応答が交換される。与えられたペアの各 CP は、他と間に 1 つの競合勝者セッションと 1 つの競合敗者セッションがある。

CPI-C. 共通プログラミング・インターフェース・コミュニケーション (Common Programming Interface for Communications)。

CP-MSU. 制御点管理サービス単位 (Control point management services unit)。

CP 名 (CP name). 制御点 (CP) のネットワーク修飾名。CP のノードが属するネットワーク (または名前空間) を識別するネットワーク ID 修飾子と、CP を識別するネットワーク ID の有効範囲内の固有の名前から構成される。個々の APPN ノードまたは LEN ノードには、システム定義時に CP 名が 1 つずつ割り当てられる。

定義域間 (cross-domain). SNA において、複数の定義域を含む制御または資源に関する用語。

CUA. 共通ユーザー・アクセス (Common User Access)。

顧客情報管理システム (Customer Information Control System (CICS)). 複数のリモート端末で入力された複数のトランザクションを、ユーザー作成のアプリケーション・プログラムで同時に処理できるようにした IBM ライセンス・プログラム。 データベースの構築、使用、および保守を行う機能が含まれている。

D

DACTLU. 非活動論理装置 (Deactivate logical unit)。

DACTPU. 非活動論理装置 (Deactivate physical unit)。

データ (data). (1) 情報の再解釈可能な表現であって、その表現は通信、解釈、または処理に適した形式になっている。 データに対しては、人間または自動的手段により操作を行うことができる。 (T) (2) 意味が割り当てられているか、あるいは意味を割り当てることのできる、文字やアナログ量などの表示。 (A) (3) 人間または自動装置による通信、解釈、または処理に適した形式で事実または説明を表現したもの。 データには、定数、変数、配列、文字ストリングなどがある。

注: プログラマーは、命令とその操作対象のデータを区別して考えるが、一般的な用語の意味としては、データにはプログラムおよびプログラム命令も含まれる。

データベース (database). (1) 要求にもとづいて複数のユーザー用にデータを受け入れ、格納し、提供するための、特定の構造をもつデータの集合。 (T)(2) 1 つまたは複数のアプリケーションを取り扱うためのデータベース・スキーマに従って編成された、相互に関連するデータの集合。 (T)(3) システムにとって基本的なデータの集まり。 (A)(4) 企業にとって基本的なデータの集まり。 (A)

データ回線 (data circuit). (1) 両方向のデータ通信の手段を提供する、関連付けられた一対の送信チャネルと受信チャネル。 (I)(2) SNA において、*リンク接続 (linkconnection)* の同義語。(3) *物理回線 (physical circuit)* および*仮想回線 (virtual circuit)*も参照。

注:

1. データ交換装置間のデータ回線には、データ交換装置で使用されているインターフェースのタイプに応じて、データ回線終端装置 (DCE) も含まれることがある。
2. データ・ステーションと、データ交換装置またはデータ集中装置との間のデータ回線には、データ・ステーション側のデータ回線終端装置が含まれるほか、データ交換装置またはデータ集中装置のロケーションにある DCE に類似の装置も含まれることがある。

データ・フロー制御 (data flow control (DFC)). SNA において、1 つのハーフセッションにおけるデータ・フロー制御層とセッション・パートナーにおけるデータ・フロー制御層との間で交換される要求と応答に使用される、要求/応答単位 (RU) のカテゴリー。

データ・ホスト (data host). *データ・ホスト・ノード (data host node)* の同義語。*コミュニケーション管理構成ホスト (communication managementconfiguration host)* と対比。

データ・ホスト・ノード (data host node). コミュニケーション管理構成において、アプリケーションの処理に専念する、チャネル接続またはアダプター接続装置以外のタイプ 5 ホスト・ノード。データ・ホスト *(data host)* と同義。*コミュニケーション管理構成ホスト・ノード (communication management configuration hostnode)* と対比。

データ・リンク (data link). SNA においては、*リンク (link)* と同義。

データ・リンク制御 (data link control (DLC)). 秩序立った情報交換を行うために（SDLC リンクやトークン・リングなどの）データ・リンク上でノードが使用する一連の規則。

データ・リンク制御（DLC）層（data link control (DLC) layer）. SNA では、2 つのノード間でのリンクを介したデータ転送をスケジュールし、そのリンクのエラー制御を行うリンク・ステーションからなる層。データ・リンク制御の例としては、ビット順次リンク接続用の SDLC やシステム/370 チャネル用のデータ・リンク制御などがある。

注: DLC 層は、通常は物理的なトランスポート・メカニズムから独立していて、上位の層に送られるデータの整合性を確保できるようになっている。

データ・リンク・レベル (data link level). (1) データ端末の階層構造において、高位論理とデータ・リンクの制御を保守するデータ・リンクの間の制御の概念的レベルまたは処理ロジック。 データ・リンク・レベルは、送信ビットの挿入および受信ビットの削除、アドレスおよび制御フィールドの解釈、コマンドおよび応答の生成、送信、および解釈、ならびにフレーム検査順序列の計算および解釈を実行する。 *パケット・レベル (packet level)* および *物理レベル (physical level)* も参照。(2) X.25 通信において、*フレーム・レベル (framelevel)* の同義語。

データ・セット (data set). (1) *ファイル (file)* の同義語。(2) *モデム (modem)* の不適語。

データ・タイプ(data types). NetView プログラムにおいて、パネルの編成の記述。データ・タイプには、アラート、イベント、および統計がある。データ・タイプと資源タイプおよび表示タイプを組み合わせたものが、NetView プログラムの表示編成の記述となる。*表示タイプ (display types)* および *資源タイプ (resource types)* も参照。

DBCS. 2 バイト文字セット (double-byte character set)。

deactivate. ノードの資源がサービスされないようにし、資源を動作不能にし、または資源の実行設計機能をその資源が実行できない状態にすること。*活動化（する） (activate)* と対比。

割振り解除 (deallocate). LU 6.2 アプリケーション・プログラミング・インターフェース (API) の動詞であって、会話を終わらせてセッションを将来の会話のために解放するもの。 *割振り (allocate)* と対比。

省略時値 (default). 明示的に指定されていないときに想定される属性、条件、値、またはオプションに関する用語。 (I)

確定応答 (definite response (DR)). SNA において、要求の受信側に、要求連鎖に対する応答を肯定か否定かに関係なく無条件に戻すように指示するために、要求ヘッダーの応答形式要求フィールドで要求するプロトコル。*例外応答 (exception response)* および *無応答 (no response)* と対比。

定義ステートメント. (1) VTAM において、ネットワークの要素を記述するステートメント。(2) NCP において、NCP に対して資源を定義する命令のタイプ。(3) *マクロ命令 (macroinstruction)* も参照。(4) 636ページの図 1、636ページの図 2、および637ページの図 3を参照。

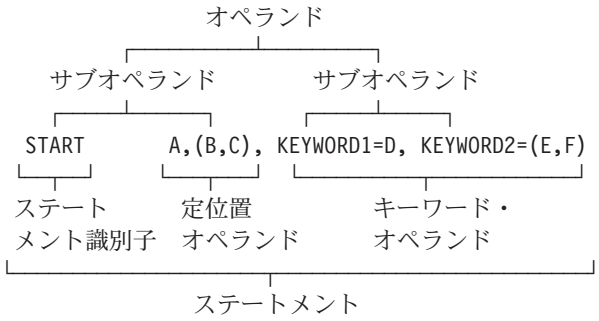


図 1. 言語ステートメントの例

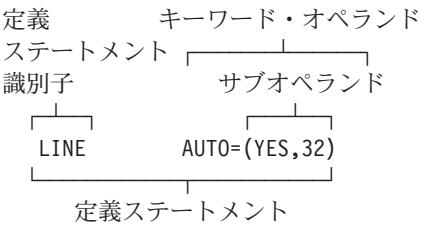


図 2. NCP 定義ステートメントの例

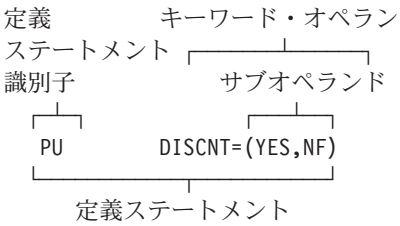


図 3. VTAM 定義ステートメントの例

DEL. 削除文字。 (A)

削除 (Delete). 選択したオブジェクトを除去するという選択。選択したオブジェクトがあった場所は、通常、ウィンドウ内に残っている 1 つまたは複数のオブジェクトで埋められる。

従属 LU (dependent LU). *SSCP 従属 LU (SSCP-dependent LU)* を参照。

従属 LU リクエスター (dependent LU requester (DLUR)). 従属 LU を所有しているが、それらの従属 LU に関する SSCP サービスを従属 LU サーバーが提供することを要求する、APPN エンド・ノードまたは APPN ネットワーク・ノード。

従属 LU サーバー (dependent LU server (DLUS)). 所属する APPN ネットワークまたは別の APPN ネットワーク内の従属 LU に SSCP サービスを提供する、APPN ネットワーク・ノード。 *従属 LU リクエスター (dependent LU requester)* と対比。

宛先 (destination). (1) ノード、ステーション、または特定の端末のように、情報が送られるポイントまたはロケーション。(2) メッセージまたは他のデータが方向付けられる外部の論理装置 (LU) またはアプリケーション・プログラム。

宛先アドレス (destination address). 情報が送られるロケーションを識別するコード。

宛先ノード (destination node). 要求またはデータが送信されるノード。

DET. 装置入力テーブル (Device entry table)。

方向づけ Locate 検索 (directed Locate search). *方向づけ検索 (directed search)* の同義語。

方向づけ検索 (directed search). 宛先ノードで資源の継続存在を検査し、経路計算についてのノードの接続情報入手するために、論理装置などの資源を含めることがわかっている特定の宛先ノードへ送信される検索要求。 *同報通信検索 (broadcast search)*と対比。 *方向づけ Locate 検索 (directed Locate search)* と同義。

ディレクトリー (directory). (1) 対応するデータ項目を識別および参照する表。 (I) (A) (2) ファイル・システムにおいて、複数のファイルを階層的にまとめて名前付きグループにしたもの。(3) APPN ノード内にあって、資源（主として論理装置）の名前をリストし、各資源が位置しているノードの CP 名を記録するデータベース。 *分散ディレクトリー・データベース (distributed directory database)* および*ローカル・ディレクトリー・データベース (local directory database)* を参照。

ディレクトリー・サービス (directory services (DS)). ネットワーク資源の位置に関する情報を保持する、APPN ノードの制御点構成要素。

直接経路指定 (direct routing). インターネット通信において、宛先およびソースが同じ IP ネットワークまたは IP サブネットにある場合の、インターネット・プロトコル (IP) データグラムの伝送。

disable. 機能しないようにすること。

使用不能 (disabled). ある種の割込みを許すような、処理装置の状態に関する用語。

ディスカバリー (discovery). データ通信において、ネットワーク・トポロジーの変更（たとえば、新規および削除されたノード、新規および削除されたインターフェースなど）の自動発見。

ディスプレイ、表示する（display）. データを視覚的に表示すること。 (I) (A)

表示レベル (display levels). *表示タイプ (display types)* の同義語。

表示タイプ (display types). NetView プログラムにおいて、パネルの編成を記述する概念。表示タイプは、合計、最新、ユーザー処置、および詳細として定義される。表示タイプと資源タイプおよびデータ・タイプを組み合わせたものが、NetView のパネル編成の記述となる。 *データ・タイプ (data types)* および*資源タイプ (resource types)* も参照。*表示レベル (display levels)* と同義。

分散ディレクトリー・データベース (distributed directory database). APPN ネットワーク内に散在する個々のディレクトリー内に維持されている、そのネットワーク内の全資源の完全リスト。 各ノードはそれぞれ

完全ディレクトリーの一部ずつを分かち持っているが、１つのノードがリストの全体を保有している必要はない。このリストの項目は、システム定義、操作員の処置、自動登録、実行中のネットワーク探索手順により、作成、変更、および削除される。 *分散ネットワーク・ディレクトリー (distributed directory directory)* およびネットワーク・ディレクトリー・データベース *(network directory database)* と同義。

分散ネットワーク・ディレクトリー (distributed network directory). *分散ディレクトリー・データベース (distributed directory database)* の同義語。

DLC. データリンク制御 (Data link control)。

DLL. 動的リンク・ライブラリー (Dynamic link library)。

DLUR. 従属 LU リクエスター (Dependent LU requester)。

DLUS. 従属 LU サーバー (Dependent LU server)。

定義域. (1) データ処理資源を共通制御下に置くコンピューター・ネットワークの一部分。 (T)(2) 単一論理システムの共用ネットワーク資源を割り振る一連のサーバー。(3) SNA については、*エンド・ノード定義域 (endnode domain)*、*ネットワーク・ノード定義域 (networknode domain)*、およびシステム・サービス制御点 *(SSCP) 定義域 (system services control point (SSCP)domain)* を参照。(4) 開放型システム間相互接続 (OSI) において、分散システムの一部分、または共通の方針が適用される一連の管理オブジェクト。(5) *管理定義域 (Administrative Domain)* および*定義域名 (domain name)* を参照。

定義域名 (domain name). インターネット・プロトコルの組において、ホスト・システムの名前。定義域名は、区切り文字で区切られた一連の副次名から構成される。たとえば、ホスト・システムの完全修飾定義域名 (FQDN) が `ralvm7.vnet.ibm.com` である場合、以下に挙げるものそれぞれが定義域名である。

- `ralvm7.vnet.ibm.com`
- `vnet.ibm.com`
- `ibm.com`

定義域操作員 (domain operator). 複数定義域ネットワークにおいて、１つのシステム・サービス制御点 (SSCP) が制御する資源の働きを制御する人またはプログラム。*ネットワーク操作員 (network operator)* も参照。

DOS セッション (DOS session). パーソナル・コンピューターが、ディスク・オペレーティング・システム (DOS) のもとで実行され、独立型コンピューターとして動作するセッション。ホスト・セッション *(host session)* を参照。

2 バイト文字セット (double-byte character set (DBCS)). 複数の文字の集合であって、各文字が 2 バイトで表されるもの。 日本語、中国語、朝鮮語など、256 個のコード・ポイントで表せる記号より多くの記号を含む言語では、2 バイト文字セットが必要である。 各文字にそれぞれ 2 バイトが必要なので、DBCS 文字を入力、表示、印刷するには DBCS をサポートするハードウェアおよびプログラムが必要になる。 *1 バイト文字セット (single-byte character set (SBCS))* と対比。

ダウンストリーム（downstream）. ホストからユーザーへ流れるデータ・フローの方向。 *アップストリーム (upstream)* と対比。

待ち行列処置 (drain). パートナー論理装置とのセッションを非活動化する以前に保留割振り要求を受けること。これは、LU 6.2 にのみ適用される。

DS. ディレクトリー・サービス (Directory services)。

二重 (duplex). データの送信と受信を同時に行うことができる通信に関する用語。 *全二重 (full-duplex)* と同義。*半二重 (half-duplex)* と対比。

動的 (dynamic). (1) プログラミング言語において、プログラムの実行中にのみ設定することができる特性に関する用語。たとえば、可変長データ・オブジェクトの長さは動的である。 (I)(2) あらかじめ定まった時間または固定した時間ではなく、必要な時間に生じる動作に関する用語。(3) *静的 (static)* と対比。

ダイナミック・リンク・ライブラリー (dynamic link library (DLL)). リンク中ではなく、ロード時間または実行時間中にプログラムにバインドされる実行可能コードおよびデータを含むファイル。 ダイナミック・リンク・ライブラリー内のコードおよびデータは、いくつかのアプリケーションで同時に共用できる。

E

EBCDIC. 拡張 2 進化 10 進コード。 256 個の 8 ビット文字で構成されるコード化文字セット。

エコー (echo). (1) コンピューター・グラフィックスにおいて、入力装置によってディスプレイ・コンソールの操作員に提供される現行値の即時通知。 (I) (A) (2) ワード処理において、各文字または行を入力されたように

印刷または表示すること。(3) データ通信において、通信チャネル上で反射される信号。たとえば、通信端末装置において、各信号は、ローカル端末で入力されたときと、通信リンクを介して戻されたときの 2 回表示される。これにより、信号を正確にチェックすることができる。

エミュレーション・プログラム (Emulation Program (EP)). (1) チャネル接続 IBM 通信コントローラーが IBM 2701 データ・アダプター装置や IBM 2702 伝送制御装置をエミュレートできるようにするための IBM 制御プログラム。(2) *ネットワーク制御プログラム (network control program)* も参照。

エミュレーター (emulator). 1 つの装置を、異なるタイプの装置のように動作させるプログラム。たとえば、Communications Server は、サポートされるパーソナル・コンピューターおよびプリンターを、それらが 3270 シリーズのワークステーションであるかのように動作させる。

EN. エンド・ノード (end node)。

使用可能 (enabled). (1) ある種の割込みを許すような、処理装置の状態に関する用語。(2) 伝送制御装置または音声応答装置が回線上の到着呼出しを受け入れることができる状態に関する用語。

ENCP. エンド・ノード制御点 (end-node control point)。

暗号化 (encryption). コンピューターのセキュリティーにおいて、暗号解読処理を行わない限り元の意味を得られないような方法で、データを判別不可能なフォームに変換する処理のこと。

エンド・ノード (end node (EN)). (1) *拡張対等通信ネットワーキング (APPN) エンド・ノード (Advanced Peer-to-Peer Networking (APPN)end node)* およびローエントリー・ネットワーキング *(LEN) エンド・ノード (low-entrynetworking (LEN) end node)* を参照。(2) 通信において、単一のデータ・リンクに頻繁に接続され、中間経路指定機能として働くことのできないノード。

エンド・ノード・ドメイン (end node domain). エンド・ノード制御点、その接続リンク、およびそのローカル LU。

エンド・ユーザー検証 (end-user verification). 論理装置 (LU) 6.2 において、接続機能管理ヘッダー (FMH) の ID およびパスワードを使用して、ユーザーの ID を調べること。 *パートナー LU 検証 (partner-LU verification)* を参照。 *会話レベル・セキュリティー (conversation-level security)* も参照。

入口点 (entry point (EP)). (1) コンピューター・プログラム、ルーチン, またはサブルーチンに入ったときに最初に実行する命令のアドレスまたはラベル。 コンピューター・プログラム、ルーチン、またはサブルーチンは、それぞれが異なる機能や目的に対応する、複数の入口点を持つことがある。 (I) (A) (2) SNA において、分散ネットワーク管理サポートを提供するタイプ 2.0、タイプ 2.1、タイプ 4、またはタイプ 5 ノード。 ここからは、このノードに関するネットワーク管理データとここで制御する資源が集中処理を行うためフォーカル・ポイントに送られ、その資源を管理および制御するためフォーカル・ポイントで開始されたコマンドが戻され実行される。

EP. 入口点 (entry point)。

ER. 明示経路 (explicit route)。

ERP. エラー回復手順(ERP) (error recovery procedures)。

エラー (error). 計算、観察、または測定された値または条件が、真の、または指定された、または理論上正しい値または条件と矛盾している状態。 (I) (A)

エラー回復手順 (error recovery procedures (ERP)). (1) 装置で発生したエラーを分離し、可能であれば回復するために役立つよう設計された手順。この手順は、しばしば、機械誤動作に関する情報を記録するプログラムと一緒に使用される。(2) 伝送エラーから回復しようとするルーチンのセット。

イーサネット (Ethernet). 複数のステーションが事前の調整を必要とせずに自由に伝送媒体にアクセスできる 10 Mbps ベースバンド・ローカル・エリア・ネットワークで、搬送波検知および送信延期を使用して競合を回避し、衝突検出および遅延再送を使用して競合を解決する。イーサネットは、搬送波検知多重アクセス／衝突検出 (CSMA/CD) を使用する。

事象 (event). あるタスクにとって何か意味のあることが発生すること。たとえば、SNMP トラップの発生、ウィンドウまたはサブマップのオープン、または、非同期操作の完了など。

事象カード (event card). NetView for AIX において、エージェントの管理ノードの状況が変わったことを反映

してエージェントからマネージャーに送られる事象に含まれている情報の、カードに似たグラフィカル表現。

例外 (exception). 異常な状態。たとえば、データ・セットまたはファイルを処理しているときに起こる I/O エラー。

例外応答 (exception response (ER)). SNA において、要求が受信時に受諾不能かまたは処理できない場合に限り応答を戻すこと、つまり、否定応答は必要だが肯定応答は不要であることを受信側に指示するために、要求ヘッダーの応答形式要求フィールドで要求するプロトコル。*確定応答 (definite response)* および*無応答 (no response)* と対比。

交換識別 (exchange identification (XID)). 隣接ノードの間でノードおよびリンクの特性を伝えるために使用する、特定のタイプの基本リンク単位 (BLU)。 XID は、リンク活動化の前、もしくはその最中に、リンクとノード特性を確立および折衝するためにリンク・ステーションの間で交換され、リンク活動化の後でこれらの特性の変更を知らせるために交換される。

実行する (execute). プログラムの全体または一部によって指定されたアクションを実行すること。 (T)

急送フロー (expedited flow). SNA において、ネットワーク制御、セッション制御、およびさまざまなデータ・フロー制御の要求/応答単位 (RU) を運ぶために使用される、伝送ヘッダー (TH) により示されるデータ・フロー。急送フローは、通常のフロー (基本的にエンド・ユーザー・データを運ぶ) とは異なり、通常のフローに影響を与えるコマンドに使用することができる。*通常フロー (normal flow)* と対比。

注: 通常フローと急送フローは、1 次から 2 次の方向へも 2 次から 1 次の方向へも移動する。与えられたフローにおける要求および応答は、通常は、通常フローであれ急送フローであれ、パス内で順次に処理される。しかし急送フローのトラフィックは、ハーフ・セッション内の待合せポイントで、境界機能のハーフ・セッション・サポートのために、通常フローのトラフィックよりも先にパス内を流される。

明示フォーカル・ポイント(FP) (explicit focal point). その制御範囲に含まれるノードのセットに割り振られたフォーカル・ポイントは、ローカルに定義される。 明示フォーカル・ポイントは、ノードをその制御範囲に運ぶ管理サービス機能の交換を開始する。 *暗黙フォーカル・ポイント (implicit focal point)* と対比。

明示経路 (explicit route (ER)). SNA において、2 つのサブエリア・ノードを接続する 1 つまたは複数の伝送グループ。 明示経路は、発信元サブエリア・アドレス、宛先サブエリア・アドレス、明示経路番号、および反転明示経路番号により識別される。 *仮想経路 (virtual route(VR))* と対比。

拡張 2 進化 10 進コード (extended binary-coded decimal interchange code (EBCDIC)). Communications Server が パーソナル・コンピューターとホスト・システム間の情報交換に使用する標準コードで、8 ビットのコード化された文字で構成されるコード化文字セットを使う。*情報交換用米国標準コード (American National Standard Code forInformation Interchange)*も参照。

E1. *T1* を参照。

F

障害 (fault). ある機能装置の本来の機能が実行できなくなるような不慮の条件。 (I) (A)

フィールド (field). (1) データを入れるために使用されるレコードまたはパネルの領域。(2) IBM 3270 データ・ストリームにおいて、同じ特性を有する表示空間の連続した位置より成るグループ。フィールドの開始部分にあるフィールド属性は表示空間の特性を定義する。(3) ウィンドウ内の識別可能領域。 フィールドの例としては、テキストを入力または置くための入力フィールド、1 つの選択項目を選択するためのラジオ・ボタン選択項目フィールドがある。

ファイル (file). 1 単位として記憶され処理される一組のレコードに名前を付けたもの。 (T) データ・セット *(data set)* と同義。

filter. 指定の基準に従ってデータ、シグナル、または資料を分離する装置またはプログラム。 (A)

ファースト・スピーカー・セッション (first speaker). *ファースト・スピーカー・セッション (first-speaker session)* を参照。

ファースト・スピーカー・セッション (first-speaker session). セッションの活動化時に定義されるハーフ・セッション。(a) もう一方のハーフ・セッションの許可を得なくてもブラケットを開始することができるように定義する場合と、(b) 両方のハーフ・セッションが同時にブラケットを開始しようとした場合に競合勝者となるように定義する場合がある。*競合勝者セッション*

(contention-winner session) の同義語。 *送信権要求者セッション (bidder session)* と対比。

固定ペーシング (fixed pacing). *固定セッション・レベル・ペーシング (fixed session-level pacing)* の同義語。

固定セッション・レベル・ペーシング (fixed session-level pacing). セッション・レベル・ペーシングの方式の 1 つ。この方式では、セッション活動化の時点で初期化された固定ペーシング・サイズを使用して、データ転送速度が制御される。 *固定ペーシング (fixed pacing)* と同義。 *最適セッション・レベル・ペーシング (adaptive session-level pacing)* と対比。

フラグ (flag). (1) それ以降の処理のために選択する情報項目にマーク付けすること。 (T) (2) ワードの終りなど、特定の条件が起きたことを知らせる文字。 (A) (3) ワードの終りやデータ伝送ブロックの始まりや終りのような、オカレンスや境界にマーク付けする文字またはビット・シーケンス。

フロー (flow). NetDA/2 において、1 つのノード、接続、または経路を一定の時間内に両方向に通過できるトラフィックの量。

フロー制御 (flow control). (1) SNA において、データ・トラフィックがネットワークの複数の構成要素の間を通過する速度を管理する処理。 フロー制御の目的は、ネットワーク内の輻輳（ふくそう）を最小限に抑えながら、メッセージ単位のフロー速度を最大限にすることにある。つまり、受信側および中間経路指定ノードのどのバッファーにもオーバーフローが生じることなく、また受信側に後続メッセージの待機状態を続けさせることもないような状態を確保することである。(2) *ペーシング (pacing)* も参照。

FMD. 機能管理データ (function management data)。

FNA. フリー・ネットワーク・アドレス (free network address)。

フォーカル・ポイント (focal point (FP)). *管理サービス・フォーカル・ポイント(management services focal point (MSFP))* を参照。

外部ホスト (foreign host). *リモート・ホスト (remote host)* の同義語。

FP. フォーカル・ポイント (focal point)。

FQDN. 完全修飾ドメイン名 (fully qualified domain name)。

FQPCID. 完全修飾プロシージャー相関 ID (fully qualifiedprocedure correlator identifier)。

フレーム（frame）. (1) 開放型システム間相互接続アーキテクチャーにおける構造の 1 つ。特定分野の知識を表現するために使用され、特定の属性の値を受け入れることのできるスロットで構成され、フレームに適切なプロシージャーを付加することにより推論を導き出すことができる。 (T)(2) IBM トークンリング・ネットワークなど、ある種のローカル・エリア・ネットワークにおける伝送単位。 区切り文字、制御文字、情報、および検査文字から成る。(3) SDLC において、すべてのコマンド、すべての応答、および SDLC 操作手順を使用して伝送されるすべての情報の伝達手段。(4) あるタイプの通信プロトコルのフィールド仕様を満たすフィールドから成るデータ構造（データ・フレーム）。フレームは、データ・リンクを介したデータ転送を制御するために使用される。(5) SDLC において、開始フラグおよび終了フラグによって限定される、ビットの順序列。 X.25 パケット交換データ・ネットワークにおいて、フレームは開始フラグおよび終了フラグによって限定された 8 ビット・バイトの順序列から構成される。X.25 におけるフレームは各種の機能、データ転送、および伝送チェックを制御する。

フレーム・レベル (frame level). *データ・リンク・レベル (data link level)* と同義。

全二重 (full-duplex (FDX)). *二重 (duplex)* の同義語。

完全修飾ドメイン名 (fully qualified domain name (FQDN)). インターネット・プロトコル群において、ドメイン・ネームのすべてのサブネームを含むホスト・システム。 完全修飾ドメイン名の例：`ralvm7.vnet.ibm.com` 。*ホスト名 (host name)* も参照。

完全修飾名 (fully qualified name). (1) SNA において、*ネットワーク修飾名 (network-qualified name)* の同義語。(2) インターネット・プロトコル群において、*完全修飾ドメイン名 (fully qualified domain name (FQDN))* を参照。

完全修飾プロシージャー相関 ID (fully qualified procedure correlator identifier (FQPCID)). 以下の目的で使用するネットワーク固有の ID。

- ノード間で送信されるメッセージを相関させる。たとえば、Locate 検索要求をその応答と相関させるなど。
- 問題の判別と解決を行うセッションを特定する。
- 会計、監査、およびパフォーマンスの監視を目的としたセッションを特定する。

この ID は、通常、プロシージャーやセッションが開始されるLU を含んだノードで割り当てられる。ただし、ノードがエンド・ノードである場合は例外で、その場合はそのネットワーク・サーバーが割り当てることができる。 固定長の相互関係子数と、それを生成した制御点のネットワーク修飾名との連絡から成る。

関数呼出し (function call). 実行の経路を現在の関数から指定した関数へ移動し、呼び出された機能より提供される戻り値を求めるための式。 関数呼出しには、制御の移動先である関数の名前と、括弧で囲まれた値のリストが含まれる。

機能管理データ (function management data (FMD)). 論理装置 (LU) 間で交換されるエンド・ユーザー・データ用として、また、LU、PU、および制御点のネットワーク・サービス構成要素間で交換される要求および応答用として使用される RU カテゴリー。

G

GDS. 汎用データ・ストリーム (general data stream)。

汎用データ・ストリーム (general data stream (GDS)). LU 6.2 セッションに使用するデータ・ストリーム。

汎用データ・ストリーム変数 (general data stream (GDS) variable). 接頭部に ID と長さフィールドが付き、アプリケーション・データ、ユーザー制御データ、または、SNA 定義制御データのいずれかを含む、RU サブストラクチャーのタイプ。

生成 (generation). ネットワーク内の必要なプログラムすべてに資源が識別されるようにするために、定義ステートメントをアセンブルし、連係編集するプロセス。

汎用 UNBIND (generic unbind). *セッション非活動化要求 (session deactivation request)* の同義語。

GMT. グリニッジ標準時 (Greenwich mean time)を参照。

グリニッチ標準時 (greenwich mean time (GMT)). イギリスのグリニッチを通る本初子午線（経度 0 ）における平均太陽時。グリニッチ標準時は、*Z 時 (Z time)* またはズールー時 *(Zulu time)* とも呼ばれる。

注: *グリニッチ標準時 (Greenwich mean time (GMT))* と *世界協定時 (coordinated universaltime (UTC))* はしばしば混用されるが、この 2 つは同義ではない。グリニッチ標準時は近似値である。現在は、秒は天文現象の観点から定義されてはいないので、この時間目盛を表す名称としては、*世界協定時 (coordinated universal time (UTC))* の方が適切である。

グループ ID (group ID (GID)). (1) RACF において、グループを識別する 1 から 8 文字までのストリング。最初の文字は、A から Z まで、あるいは #、$、または @ のいずれかでなければならない。それ以外の文字は、A から Z、#、$、@、または 0 から 9 のいずれかでなければならない。(2) AIX オペレーティング・システムにおいて、特定のグループ名に対応する数字。グループ ID は、しばしば、グループ名を値としてとるコマンドで代用される。

H

半二重 (half-duplex (HD、HDX)). データ通信において、一度に一方向のみに行われる伝送に関する用語。 *二重 (duplex)* と対比。

ハンドル (handle). 拡張 DOS および OS/2 オペレーティング・システムによって作成された 2 進値であって、ドライブ、ディレクトリー、およびファイルを識別し、そのファイルを見つけてオープンできるようにするもの。

ヘッダー (header). (1) ユーザー・データの前に置かれる、システム定義の制御情報。(2) メッセージ用の制御情報を含むメッセージ部分。制御情報としては、1 つまたは複数の宛先フィールド、発信ステーションの名前、入力順序番号、メッセージのタイプを示す文字ストリング、およびメッセージの優先順位などがある。

ヘッダー・ファイル (header file). *インクルード・ファイル (include file)* の同義語。

ヘルプ (Help). ユーザーが、オブジェクト、選択項目、タスク、および製品に関する有用な情報にアクセスするために使用できる選択項目。ヘルプ選択項目は、メニュー・バーに表示されることも、押しボタンの 1 つとして表示されることもある。

16 進数 (hexadecimal). (1) 16 個の異なった値または状態をもつことができる選択または条件に関する用語。 (I)(2) 基数 16 を使用する固定基数の数体系に関する用語。 (I)(3) 基数 16 の数体系に関する用語。16 進数は 0 から 9 および A から F までの範囲にあり、A は 10 を表し F は 15 を表す。

高水準言語 (high-level language (HLL)). 特定のコンピューターやオペレーティング・システムの構造を反映しないプログラミング言語。

強調表示 (highlighting). 表示する要素またはセグメントを、視覚的な属性を変更することにより強調すること。 (I) (A)

高性能経路指定 (High-Performance Routing (HPR)). データ経路指定のパフォーマンスとセッションの信頼性を強化する、APPN の追加機能。

高性能経路指定 (HPR) ノード(High-Performance Routing (HPR) node). 高性能経路指定をサポートしているAPPN エンド・ノードまたはネットワーク・ノード。

ホップ (hop). (1) APPN において、中間ノードを持たない経路の一部分。 隣接ノードを接続する 1 つの伝送グループのみから構成されている。(2) 経路指定の階層において、1 つのネットワーク内の 2 つのノードの間の論理的な距離のこと。

ホスト (host). (1) インターネットのプロトコルの組において、終端システム。 終端システムは任意のワークステーションでよい。つまり、メインフレームでなくてもかまわない。(2) *ホスト・プロセッサー (host processor)* を参照。

ホスト ID (host ID). インターネットのプロトコルの組において、IP アドレスのうちで、ネットワークのホスト・システムを定義する部分。 ホスト ID の長さは、ネットワークのタイプまたはネットワーク・クラス(A、B、またはC) によって異なる。

ホスト名 (host name). インターネットのプロトコルの組において、マシンに与えられた名前のこと。『ホスト名』 は、*完全修飾ドメイン名 (FQDN)* を指すことも、完全修飾ドメイン名のうち最も特殊なサブ名を意味する場合に使用することもある。 たとえば、`ralvm7.vnet.ibm.com` が完全修飾ドメイン名であるとすると、次のいずれもホスト名とみなすことができる。

- `ralvm7.vnet.ibm.com`
- `ralvm7`

ホスト・プロセッサー (host processor). (1) ユーザー・アプリケーション・ネットワークの全部または一部を制御するプロセッサー。 (T)(2) ネットワークにおいて、データ通信アクセス方式が存在している処理装置。

ホスト・セッション (host session). パーソナル・コンピューターとホスト・システムとの通信を可能にする論理接続。セッションは、LU アドレス、LT 番号、またはセッション ID で識別できる。*DOS セッション (DOS session)*を参照。*論理端末 (logical terminal)*も参照。

ホスト・システム (host system). Communications Server において、SDLC、LAN、ASYNCH、X.25、または DFT 接続により、1 つまたは複数のパーソナル・コンピューターにリンクされているコンピューター。

HPR. 高性能経路指定 (High-Performance Routing)。

I

ICP. インターネット制御プロトコル(Internet Control Protocol)。

ID. (1) 識別子 (identifier)。(2) 識別 (identification)。

暗黙フォーカル・ポイント (implicit focal point)。. その制御範囲に含まれるべきノードが、SOC ノードで定義される割り振られたフォーカル・ポイント。 ノードを暗黙フォーカル・ポイントの制御範囲に運ぶ管理サービスの機能交換は、SOC ノードで開始される。 *明示フォーカル・ポイント (explicit focalpoint)* と対比。

非活動、非活動状態 (inactive). (1) 操作不可能であること。(2) 1 つのノードまたは装置が他のノードまたは装置に接続されていないか、または接続の準備ができていないこと。(3) *活動、活動状態 (active)* と対比。

インバウンド (inbound). 通信において、ネットワークから受信するデータ。

インクルード・ファイル (include file). 機能、プログラム、またはユーザーのグループが使用する宣言を含んでいるファイル。 *ヘッダー・ファイル (header file)* と同義。

情報 (I) 形式 (information (I) format). 情報転送のために使用されるフォーマット。

情報 (I) フレーム (information (I) frame). 番号付けされた情報の転送に使用される I フォーマットのフレーム。

初期プログラム・ロード (initial program load (IPL)). (1) オペレーティング・システムに操作を開始させる初期化の手順。(2) 構成イメージが、作業日の最初か、システム誤動作の後で、記憶域にロードされる処理。(3)システム・プログラムのロードと、システムがジョブを実行する準備を行う処理。

INITIATE. 論理装置 (LU) からシステム・サービス制御点 (SSCP) に送られる、LU-LU セッションの確立を要求をネットワーク・サービス要求。

インストール、導入システム (installation). (1) Communications Server において、Communications Server ディスケットからマイクロコードをロードする処理。(2) システム開発において、ある機能単位の準備を整え、使用のために適正な位置に配置すること。 (T) (3) 特定のコンピューター・システム。この概念には、そのコンピューター・システムが行う作業と、そのコンピューター・システムを管理し、操作し、問題に適用し、保守を行い、生成される結果を利用する人間が含まれる。

インスタンス (instance). AIX オペレーティング・システムにおいて、抽象的なオブジェクト・クラスを実体のあるものとして認識したもの。ウィジェットまたはガジェットのインスタンスは、特定のグラフィカル・オブジェクトを実行時に画面上で生成するために使用する外観および性質の詳細情報が入った、特定のデータ構造となる。

INT. 内部トレース・テーブル (Internal trace table)。

対話式 (interactive). ユーザーとコンピューターとの間の情報交換に関する用語。

インターフェース (interface). (1) 場合に応じて機能特性、信号特性、または他の特性によって定義されるような 2 つの機能装置の間の共用境界。 この概念には、異なる機能を持つ 2 つの装置の接続を指定するという目的が含まれる。 (T) (2) システム、プログラム、または装置のリンクを行うハードウェアまたはソフトウェア、あるいはその両方。

中間セッション経路指定 (intermediate session routing (ISR)). APPN ネットワーク・ノード内の経路指定機能のタイプ。そのノードを通るが、エンド・ノードはほかの場所にあるすべてのセッションについて、セッション・レベル・フロー制御と故障率の報告を提供する。

インターネット制御プロトコル (Internet Control Protocol (ICP)). 例外通知、メトリック通知、および PING サポートを提供するVIrtual NEtworking System (VINES**) プロトコル。*RouTing update Protocol (RTP)* も参照。

インターネット・プロトコル (Internet Protocol (IP)). 単一または相互接続ネットワークを通してデータを伝送する無接続プロトコル。IP は、上位プロトコル層と物理ネットワークとの中間に位置する。ただし、このプロトコルでエラー回復とフローの制御はなく、物理ネットワークの信頼性を保証するものではない。

IP. インターネット・プロトコル（Internet Protocol）。

IPL. 初期プログラム・ロード (initial program load)。

ISR. 中間セッション経路指定 (Intermediate session routing)。

L

LAN. ローカル・エリア・ネットワーク (local area network)。

待ち時間 (latency). 命令制御装置がデータの呼出しを開始する時間と、データ転送が実際に開始する時間との間の時間間隔。

LDT. ローカル記述子テーブル (local descriptor table)。

LEN. ローエントリー・ネットワーキング (low-entry networking)。

LEN エンド・ノード (LEN end node). *ローエントリー・ネットワーキング (LEN) エンド・ノード (low-entrynetworking (LEN) end node)* を参照。

LEN ノード (LEN node). 独立 LU プロトコルをサポートするが、CP-CP セッションはサポートしないノード。サブエリア・ネットワークの境界ノードに接続された周辺ノード、APPN ネットワーク内の APPN ネットワーク・ノードに接続されたエンド・ノード、または、ほかの LEN ノードや APPN エンド・ノードに直接接続された対等接続ノードのいずれか。*ローエントリー・ネットワーキング (LEN) エンド・ノード (low-entrynetworking (LEN) end node)* も参照。

LFSID. ローカル・フォーム・セッション ID (local-form session identifier)。

限定資源 (limited resource). 指定の時間間隔の間にセッション活動が検出されない場合に、そこを通過するセッションを終了する接続機能。 *限定資源セッション (limited-resourcesession)* も参照。

限定資源リンク (limited-resource link). ノード操作員が、限定資源になる (つまり、使用時にのみ活動状態になる) ように定義するリンク。 限定資源リンクは、指定の時間間隔の間にセッション活動が検出されない場合には、非活動化される。 *限定資源セッション (limited-resourcesession)* も参照。

限定資源セッション (limited-resource session). 限定資源リンクを通過するセッション。 このセッションは、指定の時間間隔の間にセッション活動が検出されない場合には、非活動化される。

回線 (line）. (1) データ回線終端装置 (DCE) の外部のデータ回線にあって、DCE をデータ交換装置 (DSE) に接続するか、DCE を他の 1 つまたは複数の DCE に接続するか、または DSE を他の DSE に接続する部分。(I)(2) *チャネル (channel)* および*回線 (circuit)* と同義。

回線制御規則 (line control discipline). *リンク・プロトコル (link protocol)* および *プロトコル (protocol)* の同義語。

伝送制御手順 (line discipline). *リンク・プロトコル (link protocol)* および *プロトコル (protocol)* の同義語。

リンク、連係 (link). (1) リンク接続機構（伝送媒体）と、その両端にある 2 つのリンク・ステーション。 分岐構成またはトークン・リング構成では、複数のリンクが 1 つのリンク接続機構を共用することができる。(2) データ項目や、1 つまたは複数のコンピューター・プログラムの部分を相互に接続すること。たとえば、リンケージ・エディターによるオブジェクト・プログラムのリンク、またはポインターによるデータ項目のリンク。(T) (3) SNA において、*データ・リンク (data link)*と同義。

リンク接続 (link-attached). (1) データ・リンクによって制御装置に接続された装置に関する用語。(2) *チャネル接続 (channel-attached)* と対比。(3) *リモート (remote)* と同義語。

リンク接続機構 (link connection). (1) 1 つのリンク・ステーションと他のリンク・ステーションとの間で両方向の通信を提供する物理装置。たとえば、通信回線やデータ回線終端装置 (DCE) がある。(2) SNA において、*データ回線 (data circuit)* と同義。

リンク接続ネットワーク (link connection network). *接続ネットワーク (connection network)* の同義。

リンク接続セグメント (link connection segment). 構成の一部分で、この部分は、サービス・ポイント・コマンド・サービス(SPCS) 照会リンク構成要求リストの中で連続的にリストされている 2 つの資源の間に置かれている。

リンク・レベル (link level). (1) X.25 勧告の一部であり、加入者のマシンをネットワーク・ノードに接続する全二重リンクを介してネットワークの中にデータを取り込んだり、ネットワークの中からデータを取り出したりするために使用するリンク・プロトコルを定義する。LAP および LAPB は、CCITT が推奨するリンク・アクセス・プロトコルである。(2) *データ・リンク・レベル (data link level)*を参照。

リンク・プロトコル (link protocol). (1) リンク・レベルでデータを送受信するための規則。(2) *回線制御規則 (line control discipline)* および*伝送制御手順 (line discipline)* と同義。

リンク・ステーション（link station）. (1) 特定のリンクを介した隣接ノードへの接続を表す。ノード内のハードウェアおよびソフトウェア構成要素で。たとえば、ノードＡが３つの隣接ノードに接続する分岐回線の１次側である場合、ノードＡは、これらの隣接ノードへの接続を表すリンク・ステーションを３つもっている。(2) VTAM において、APPN またはサブエリア・リンクによって接続されている別の APPN またはサブエリア・ノードへの接続を表す、APPN またはサブエリア・ノード内の名前付き資源。サブエリア・ネットワーク内の資源階層においては、リンク・ステーションはサブエリア・リンクに従属する。(3) *隣接リンク・ステーション (adjacent link station (ALS))* も参照。

リンク・ステーションの役割 (link station role). SNA において、指定されたリンクに想定するローカル・ノードの役割。 考えられる役割は、1 次 (制御)、2 次、または折衝可能。

リンク状況 (link status (LS)). ローカル・モデムおよびリモート・モデムが維持する情報。

ロード (load). (1) コンピューター・プログラムの全体または一部を補助記憶装置からメモリーに入れ、コンピューターがそのプログラムを実行できるようにすること。(2) ディスケットをディスケット・ドライブに入れること。

ローカル (local). (1) 通信回線を使用しないで直接アクセスされる装置に関する用語。(2) *リモート (remote)* と対比。(3) *チャネル接続 (channel-attached)* の同義語。

ローカル・エリア・ネットワーク (local area network (LAN)). (1) 限られた地域内のユーザーの構内にあるコンピューター・ネットワーク。ローカル・エリア・ネットワーク内での通信は、外部の規則により規制されない。しかし、LAN の境界を越えて通信を行う場合は、なんらかの規制を受ける場合がある。 (T) (2) 一組の装置が相互に通信するように接続されているネットワークで、それよりも大きなネットワークへ接続できるもの。(3)イーサネット *(Ethernet)* およびトークンリング *(token ring)*

も参照。(4) *大都市圏ネットワーク (metropolitan area network (MAN))* および *広域ネットワーク (wide area network (WAN))* と対比。

ローカル記述子テーブル (local descriptor table (LDT)). OS/2 オペレーティング・システムにおいて、 処理がアドレス可能度をもつコードおよびデータ・セグメントに関するアクセス情報を含んだテーブル。

ローカル・ディレクトリー・データベース (local directory database). ネットワーク内において特定のノードにあるものとして知られている一組の資源 (LU)。この資源には、ノード定義域にあるすべての資源のほか、キャッシュ・エントリーも含まれる。

ローカル・フォーム・セッション ID (local-form session identifier (LFSID)). 指定された伝送グループ (TG) を使用する特定のセッションのトラフィックを識別するためにタイプ 2.1 ノードで使用する、動的に割り当てられる値。 LFSID は、TG を介して交換されるセッション・メッセージにともなう伝送ヘッダーの、ODAI、OAF′、および DAF′ フィールドにコード化される。

ローカル LU (local LU). LAN 上に分布せずに、ゲートウェイ・パーソナル・コンピューターによって制御される論理装置。通常、これはワークステーション、プリンター、または端末のような物理装置である。

ローカル・トポロジー・データベース (local topology database). 各エンドポイントに少なくとも 1 つのエンド・ノードを持つ伝送グループ (TG) それぞれについて 1 つの項目を含んでいる、APPN または LEN ノードのデータベース。 エンド・ノードでは、データベースが、ノードに接続される各 TG ごとに 1 つの項目を持っている。 ネットワーク・ノードでは、データベースが、ネットワーク・ノードをエンド・ノードに接続する各 TG ごとに 1 つの項目を持っている。 各項目は、それが表す TG の現在の特性について記述する。ネットワーク・ノードは、ローカル・トポロジー・データベースとネットワーク・トポロジー・データベースの両方を持っている。一方、エンド・ノードはローカル・トポロジー・データベースのみを持っている。

Locate. *Locate/CD-Initiate* の同義語。

Locate/CD-Initiate. (1) APPN ノード間で交換されるメッセージ用の略語。以下に示す汎用データ・ストリーム (GDS) 変数のセットのいずれかを含む。

- ネットワーク探索要求に使用されるLocate、Find Resource、および Cross-Domain Initiate GDS 変数
- ネットワーク資源が見つかったときの探索応答に使用されるLocate、Found Resource、および Cross-Domain Initiate GDS 変数

これらのメッセージ構造は、分散ネットワーク・ディレクトリーの探索を実行し、セッションを確立する CP 構成要素に対応する。Locate GDS 変数は、ネットワーク内の探索メッセージの送達を制御するために使用される情報を含んでいる。Find および Found GDS 変数は、ディレクトリー内で使用される情報を含んでいる。これには、それぞれ、発信元キャッシュ・データ (制御点情報)、探索引数 (宛先 LU 名)、および探し出された資源情報がある。Cross-Domain Initiate GDS 変数は、セッションの経路を選択する際に使用される終端 TG ベクトル情報を含んでいる。Locate/CD-Initiate メッセージの長さは、1024 バイトまでに制限されている。(2) *Locate* および *Locate search message* と同義。

Locate search message. *Locate/CD-Initiate* と同義。

ログオン操作員 (logged-on operator). (1) 端末およびログオン・ユーザーを必要とする NetView 操作員ステーション・タスク。(2) *自動タスク (autotask)* と対比。

論理端末 (logical terminal). (1) 1 つまたは複数の物理端末に関連付けられた名前の付いた宛先。(2) 特定の 3270 または 5250 エミュレーション・セッションの定義。

論理装置 (logical unit (LU)). ネットワーク・アクセス可能単位のタイプの 1 つ。ユーザーはこれを使用して、ネットワーク資源にアクセスしたり互いに通信したりする。

ローエントリー・ネットワーキング (low-entry networking (LEN)). 基本の対等通信プロトコルを使用して直接相互に接続するノードの機能。論理装置間での複数の並列セッションをサポートする。

ローエントリー・ネットワーキング (LEN) エンド・ノード (low-entry networking (LEN) end node). 隣接 APPN ネットワーク・ノードからネットワーク・サービスを受け取る LEN ノード。

ローエントリー・ネットワーキング (LEN) エンド・ノード (low-entry networking (LEN) end node). エンド・ユーザー・サービスの範囲を提供し、対等プロトコルを使用して別のノードに直接接続し、隣接 APPN ネットワーク・ノードから暗黙的にネットワーク・サービスを派生する (つまり、直接 CP-CP セッションを使用しない)。

LS. リンク状況 (link status)。

LU. 論理装置 (logical unit)。

LU-LU セッション (LU-LU session). SNA ネットワーク内の 2 つの論理装置 (LU) 間の論理接続で、一般に 2 人のユーザー間の通信を提供する。

LUS. 論理装置サービス (logical unit services)。

LU タイプ. 特定のセッションについて各 LU がサポートする SNA プロトコルおよびオプションのサブセットを基準とした場合の LU の種別。基準には次のものがある。

- セッション活動化要求で使用できる必須値および任意選択値
- データ・ストリーム制御、機能管理ヘッダー (FMH)、要求単位パラメーター、およびセンス・データ値の用途
- 表示サービス・プロトコル（FMH の用途に関連したものなど）

LU タイプ 0、1、2、3、4、6.1、6.2、および 7 が定義されている。

LU 6.2. (1) 論理装置のタイプの 1 つで、分散処理環境でのプログラム間の一般的な通信をサポートする。 LU 6.2 には以下の特徴がある。(a) セッション・パートナー間の対等関係。(b) 1 つのセッションを複数トランザクション用に効率使用。(c) 広範な終端間エラー処理。(d) 製品の実装に組み込まれている構造化された動詞から成る汎用アプリケーション・プログラミング・インターフェース (API)。(2) 分散データ処理環境において、SNA の一般データ・ストリーム（構造化フィールド・データ・ストリーム）またはユーザー定義のデータ・ストリームを使って 2 つのアプリケーションの間のセッションをサポートするタイプの LU。

LU 6.2 verb. LU 6.2 アプリケーション・プログラミング・インターフェースにおいて 1 つの命令を表す構文単位。

M

MAC. 媒体アクセス制御 (medium access control)。

マクロ命令 (macroinstruction). (1) ソース言語の中にあって、同じソース言語内の定義済みの命令順序で置き換えられる命令。この命令では、置換後のパラメーターの値も指定できる。 (T)(2) アセンブラー・プログラミングにおいて、マクロ定義と呼ばれる事前定義のステートメント・セットをアセンブラーに処理させるアセンブラー言語ステートメント。プログラム内のマクロ命令は、通常マクロ定義から作られるステートメントで置き換えられる。(3) 定義ステートメント *(definition statement)* も参照。

保守分析手順 (maintenance analysis procedure (MAP)). IBM 技術員に対して、システムをトレースして障害の原因を突き止めるためのステップごとの手順を示す、保守用の文書。

MAN. 大都市圏ネットワーク (metropolitan area network)。

管理情報ベース (Management Information Base (MIB)). (1) ネットワーク管理プロトコルを使用してアクセスすることのできるオブジェクトの集合。(2) ホストやゲートウェイから使用可能な情報と、許可される操作を指定する管理情報の定義。(3) OSI において、開放型システム内の管理情報の概念的リポジトリー。

管理サービス (management services (MS)). (1) 制御点 (CP) および物理装置 (PU) の中のネットワーク・サービスのタイプの 1 つ。 管理サービスは、SNA ネットワークの管理を支援する目的で提供されているサービスで、問題管理、パフォーマンスおよび会計管理、構成管理、変更管理などがある。(2) システムおよびネットワークの管理を支援するサービス。対象分野には、問題管理、パフォーマンス管理、ビジネス管理、運用管理、構成管理、変更管理などがある。

管理サービス・フォーカル・ポイント (management services focal point (MSFP)). ある管理サービス規則 (問題判別や応答時間モニターなど) について、特定の制御のためのネットワーク管理データ・タイプを対象とする制御点。データの収集、格納、表示の他に、これらすべてを担当する場合がある。(たとえば、問題判別フォーカル・ポイントは問題判別データを収集する制御点であり、問題判別データの格納や表示を行う場合もある)。

管理サービス単位 (management services unit (MSU)). データの移送手段に関係なく主ベクトルでコード化された、管理サービス・データの総称。つまり、管理サービス単位には、ネットワーク管理ベクトル移送 (NMVT) 内で移送される主ベクトル、制御点管理サービス単位 (CP-MSU)、または複数定義域サポート・メッセージ単位(MDS-MU) が含まれる。

マネージャー（manager）. (1) システム管理において、特定の対話に関してマネージャーの役割を果たすものとみなされるユーザー。(2) (a) オブジェクトに関する通知を受け取り、(b) オブジェクトを修正または照会するための管理操作を要求することによって、1 つまたは複数

の管理オブジェクトをモニターまたは制御するエンティティー。(3) マネージャーの役割を果たすものとみなされるシステム。

マップ (map). NetView for AIX において、ネットワークおよびそのシステムのグラフィカル表示および階層表示を提供する関連サブマップのセットによって表されるデータベース。

MAP. 保守分析手順 (maintenance analysis procedure)。

マップ式会話 (mapped conversation). 割振りトランザクション・プログラムにより指定される LU 6.2 会話タイプ。マップ式会話を使用するトランザクション・プログラムは、基盤となるデータ・ストリームに関係なく、散在フォーマットのメッセージを交換することができる。システムに定義された、またはユーザーに定義されたマッパーは、トランザクション・プログラムのデータ形式変更を実行することができる。 *会話（conversation）* も参照。 *基本会話 (basic conversation)* も参照。

マッピング（mapping）. 送信側がある形式で伝送したデータを、受信側が受け入れることのできるデータ形式に変換するプロセス。

最大化 (Maximize). あるウィンドウを可能な最大サイズに拡大するための選択項目。

MB. メガバイト (megabyte)。

MDS. 複数定義域サポート (multiple-domain support)。

MDS-MU. 複数定義域サポート・メッセージ単位 (multiple-domain support message unit)。

媒体 (medium). (1) 電気エネルギーの物理的な搬送装置。(2) 内部または表面でデータを表示できる物理的な素材。

媒体アクセス制御 (medium access control (MAC)). LAN において、媒体に依存する機能をサポートし、物理層のサービスを使用して論理リンク制御 (LLC) 副層にサービスを提供する、データ制御層の副層のこと。MAC 副層には、装置がいつ伝送媒体にアクセスしたかを判別するメソッドが組み込まれている。

媒体アクセス制御 (MAC) プロトコル (medium access control (MAC)protocol). ローカル・エリア・ネットワークにおいて、ネットワークのトポロジーの一側面を考慮しながら、データ装置間でデータを交換できるようにするために行う、伝送媒体へのアクセスを管理するプロトコル。

媒体アクセス制御 (MAC) 副層 (medium access control (MAC) sublayer). ローカル・エリア・ネットワークにおいて、媒体アクセス方式を適用するデータ・リンク層の部分。 MAC 副層は、トポロジーに依存する機能をサポートし、物理層のサービスを使用して論理リンク制御副層にサービスを提供する。 (T)

メガバイト (megabyte (MB)). (1) 主記憶装置、実記憶装置および仮想記憶装置、およびチャネル・ボリュームでは、220 または 1 048 576 バイト。(2) ディスク記憶装置の容量およびコミュニケーション・ボリュームでは、1 000 000バイト。

メモリー (memory). 処理装置またはその他の内部記憶装置内にあって、命令の実行に使用されるすべてのアドレス可能記憶域スペース。 (T)

メニュー（menu）. (1) データ処理システムがユーザーに対して表示するオプションのリスト。このリストから、ユーザーは開始するアクションを選択できる。 (T) (2) テキスト処理において、テキスト・プロセッサーがユーザーに対して表示する選択項目のリスト。このリストから、ユーザーは開始するアクションを選択できる。 (T) (3) オブジェクトに適用できる選択項目のリスト。メニューには、特定のコンテキストでは選択できない選択項目が含まれていることもある。 このような選択項目は弱いコントラストで表示される。

メッセージ (message). (1) 発信元から 1 つまたは複数の受信側に 1 つのエンティティーとして転送される、いくつかの文字と場合によっては制御コードの集合。メッセージはエンベロープと内容の 2 つの部分から成っている。 (T) (2) ある人またはプログラムから別の人またはプログラムに送られる連絡。

メッセージ単位 (message unit (MU)). SNA において、任意の層で処理されるデータの単位。たとえば、基本情報単位 (BIU)、パス情報単位 (PIU)、または要求/応答単位 (RU)。

メトリック (metric). インターネット通信において、同じ自律システムへの複数の出口または入口点を区別するために使用する、経路に関連付けられた値。メトリックが最低の経路が優先される。

大都市圏ネットワーク (metropolitan area network (MAN)). 複数のネットワークを相互接続して形成されるネットワークで、このネットワークは、構成単位の各ネットワークよりも高速で動作し、行政地区境界を越え、複数のアクセス方式を使用することができる。 (T) ロー

カル・エリア・ネットワーク *(local area network(LAN))* および*広域エリア・ネットワーク (wide area network (WAN))* と対比。

MIB. (1) MIB モジュール (MIB module)。(2) 管理情報ベース (Management Information Base)。

移行 (migration). プログラムの新しいバージョンまたはリリースを、旧バージョンまたはリリースと取り替える形で導入すること。

混合媒体多重リンク伝送グループ (mixed-media multilink transmission group (MMMLTG)). *伝送グループ (transmission group (TG))* を参照。

MLTG. 多重リンク伝送グループ (multilink transmission group)。

MMMLTG. 混合媒体多重リンク伝送グループ (mixed-mediamultilink transmission group)。

モード (mode). *モード名 (mode name)* を参照。

モデム（変復調装置）(modem (modulator/demodulator)). (1) 信号を変調および復調する機能装置。 モデムの主要機能の 1 つは、アナログ伝送機能を介してディジタル・データを伝送できるようにすることである。 (T) (A)(2) コンピューターからのディジタル・データを、通信回線上に伝送できるアナログ信号へ変換し、受け取られたアナログ信号をコンピューター用のデータへ変換する装置。

モード名 (mode name). セッションに必要な特性（トランスポート・ネットワーク内でのトラフィック・ペーシング値、メッセージ長の上限、同期点と暗号のオプション、およびサービス・クラスなど）を指定するために、セッションの起動側が使用する名前。

モジュール (module). コンパイル、他の単位との結合、およびロードという点から見て、離散的で識別可能なプログラム単位。たとえば、アセンブラー、コンパイラー、リンケージ・エディター、またはエグゼクティブ・ルーチンへの入力、またはそれからの出力など。 (A)

MS. 管理サービス (management services)。

MSFP. 管理サービス・フォーカル・ポイント (management services focal point)。

MSG. 操作卓メッセージ (console messages)。

MSU. 管理サービス単位 (Management services unit (MSU))

MU. メッセージ単位 (message unit (MU))。

マルチキャスト (multicast). (1) 選択された宛先グループに同じデータを伝送すること。 (T)(2) パケットのコピーがすべての可能な宛先のうちのサブセットにのみ送られる同報通信の特別な形。(3) *同報通信 (broadcast)* と対比。

多重リンク伝送グループ (multilink transmission group). *伝送グループ (transmission group (TG))* を参照。

複数定義域サポート (multiple-domain support (MDS)). LU-LU および CP-CP セッションを通じて管理サービス機能セット間で管理サービス・データを伝送する技術。*複数定義域サポート・メッセージ単位 (multiple-domain supportmessage unit (MDS-MU))* も参照。

複数定義域サポート・メッセージ単位 (multiple-domain support message unit (MDS-MU)). 管理サービス・データを含み、複数定義域サポートで使用される LU-LU および CP-CP セッションを通じて管理サービス機能セット間を流れる、メッセージ単位。このメッセージ単位は、これに含まれる実際の管理サービス・データと同じく、汎用データ・ストリーム (GDS) フォーマットである。*制御点管理サービス単位 (control point management services unit(CP-MSU))*、*管理サービス単位 (management services unit (MSU))*、およびネットワーク管理ベクトル・トランスポート *(NMVT)* も参照。

多重仮想記憶 (Multiple Virtual Storage (MVS)). *MVS* を参照。

MVS. 多重仮想記憶域 (Multiple Virtual Storage)。MVS/390、MVS/XA、および MVS/ESA を指す。

N

ネイティブ (native). MPTN アーキテクチャーにおいて、同じトランスポート・プロトコルにもとづいているトランスポート利用者とトランスポート提供者との間の関係に関連した用語。

NAU. (1) ネットワーク・アクセス可能単位 (network accessible unit)。(2) ネットワーク・アドレス可能単位 (network addressable unit)。

NC. ネットワーク制御 (Network control) を参照。

否定応答 (negative response (NR)). SNA において、要求が正常に到着しなかったこと、または受信側により正常に処理されなかったことを示す応答。 *肯定応答 (positive response)* と対比。

折衝 (negotiation). ネットワークと 3710 ネットワーク制御装置との間で伝送されるパケット・サイズを決定するプロセス。

NETID. *ネットワーク ID (network identifier)* を参照。

NetView 間タスク (NetView-NetView task (NNT)). 定義域間 NetView 操作員セッションが実行されるタスク。*操作員ステーション・タスク (operator station task)* を参照。

ネットワーク (network). (1) ノードおよび接続分岐を配置したもの。 (T) (2) 情報交換のために接続されたデータ処理装置とソフトウェアとの構成。(3) ノードのグループおよびそれらのノードを相互に接続するリンク。

ネットワーク・アクセス可能単位 (network accessible unit (NAU)). 論理装置 (LU)、物理装置 (PU)、制御点 (CP)、またはシステム・サービス制御点 (SSCP)。パス制御ネットワークにより伝送される情報の発信元もしくは宛先である。 *ネットワーク・アドレス可能単位 (network addressable unit)* と同義。

ネットワーク・アドレス (network address). (1) ISO 7498-3 によって OSI 環境内で明確に定義された、ネットワーク・サービス・アクセス点のセットを識別する名前。(2) リンク、リンク・ステーション、またはネットワーク・アドレス可能単位を識別する、サブエリアおよび要素フィールドからなるアドレス。サブエリア・ノードはネットワーク・アドレスを、周辺ノードはローカル・アドレスを使用する。(3) SNA において、リンク、ゲートウェイ、またはネットワーク・アドレス可能単位 (NAU) を識別する、サブエリアおよび要素フィールドからなるアドレス。(4) サブエリア・ネットワークにおいて、リンク、リンク・ステーション、物理装置、論理装置、またはシステム・サービス制御点を識別する、サブエリアおよび要素フィールドからなるアドレス。 サブエリア・ノードはネットワーク・アドレスを、周辺ノードはローカル・アドレスかローカル・フォーム・セッション ID (LFSID) を使用する。 周辺ノードが接続されるサブエリア・ノードの境界機能は、ローカル・アドレスまたは LFSID をネットワーク・アドレスに変換したり、その逆を行ったりする。*ネットワーク名 (network name)* と対比。

ネットワーク・アドレス可能単位 (network addressable unit (NAU)). *ネットワーク・アクセス可能単位 (network accessible unit)* の同義語。

ネットワーク・アーキテクチャー (network architecture). コンピューター・ネットワークの論理構造および動作原理。 (T)

注: ネットワークの動作原理には、サービス、機能、およびプロトコルの動作原理も含まれる。

ネットワーク輻輳 (network congestion). ネットワークの処理能力を超えたトラフィックによって引き起こされる、望ましくない過負荷状態。

ネットワーク制御 (network control (NC)). SNA において、明示経路および仮想経路の活動化と非活動化、そして周辺ノードの調整のためのロード・モジュールの送信などを目的として、物理装置 (PU) 間で交換される要求および応答に使用する要求応答単位 (RU) カテゴリー。*データ・フロー制御 (data flow control)* 、*機能管理データ (function management data)* 、および*セッション制御 (session control)* も参照。

ネットワーク制御ブロック (network control block (NCB)). ネットワーク制御プログラムの一部分。LAN接続における通信ネットワークで使用される資源を制御する。

ネットワーク制御プログラム (network control program). IBM 提供のモジュールのライブラリーからユーザーが作成するプログラムで、通信制御装置の操作を制御する。

ネットワーク制御プログラム (Network Control Program (NCP)). 単一定義域、複数定義域、および相互接続ネットワークに通信制御装置サポートを提供する、IBM ライセンス・プログラム。

ネットワーク・ディレクトリー・データベース (network directory database). *分散ディレクトリー・データベース (distributed directory database)* の同義語。

ネットワーク ID (network identifier). (1) TCP/IP において、ネットワークを定義する IP アドレスの一部分。ネットワーク ID の長さは、ネットワーク・クラスのタイプに従って決まる (A、B、または C)。(2) 1 から 8 バイトまでの、ユーザーが選択する名前、もしくは、8 バイトの IBM が登録した名前で、特定のサブネットワークを一意に識別する。(3) MPTN アーキテクチャーに

おける、トランスポート提供者アドレスのアドレス修飾子。常駐するネットワークに従って、ノードのグループを識別する。

ネットワーク管理（network management）. 通信向けのデータ処理または情報システムを計画、編成、および制御するプロセス。

ネットワーク管理ベクトル移送 (network management vector transport (NMVT)). 物理装置管理サービスと制御点管理サービスの間の活動状態のセッション (SSCP-PU セッション) を流れる管理サービス要求／応答単位 (RU)。

ネットワーク名 (network name). ユーザーが、1 つのサブネットワーク内のネットワーク・アクセス可能装置、リンク、またはリンク・ステーションを参照するための記号識別子。 APPN ネットワークでは、ネットワーク名は経路指定の目的でも使用される。 *ネットワーク・アドレス (network address)* と対比。

ネットワーク・ノード (network node (NN)). *拡張対等通信ネットワーキング (APPN) ネットワーク・ノード (Advanced Peer-to-Peer Networking (APPN) network node)* を参照。

ネットワーク・ノード定義域 (network node domain). APPN ネットワーク制御点、そこに接続するリンク、検索要求に対して直接応答するためのネットワーク資源 (つまり、ローカル LU とそれに隣接する LEN エンド・ノード)、ディレクトリー検索要求と応答を交換するために使用する隣接する APPN エンド・ノード、および、管理サービスの提供を受けるための独自のノードあるいは隣接するエンド・ノードに関連付けられているその他の資源 (ローカル記憶装置など)。

ネットワーク・ノード・サーバー (network node server). ローカル LU およびクライアント・エンド・ノードにネットワーク・サービスを提供する APPN ネットワーク・ノード。

ネットワーク操作員 (network operator). (1) ネットワークの全体または一部の働きを制御する人。(2) 複数定義域ネットワークにおいて、すべての定義域の制御に責任を負う人またはプログラム。(3) *定義域操作員 (domain operator)* も参照。

ネットワーク修飾名 (network-qualified name). SNA において、特定のネットワーク内の特定の資源 (LU または CP など) を一意に識別する名前。 それぞれが 1 から 8 バイトまでの記号ストリングであるネットワーク ID と資源名から構成される。*完全修飾名 (fully qualified name)* と同義。

ネットワーク・トポロジー・データベース (network topology database). APPN ネットワーク内のネットワーク・ノード間の現在の接続性を表わす。 (a) すべてのネットワーク・ノードとそれに相互接続する伝送グループ、(b) ネットワーク・ノードを接続する仮想経路指定ノードへの入口を含む。

NMVT. ネットワーク管理ベクトル移送 (network management vector transport)。

NN. ネットワーク・ノード (Network node)。

NNCP. ネットワーク・ノード制御点 (network node control point)。

ノード (node). (1) ネットワークにおいて、1 つまたは複数の機能単位がチャネルまたはデータ回線を接続するポイント。 (I) (2) ネットワークに接続されていて、データの送受信を行う任意の装置。(3) 1 つのリンクの端点、またはネットワーク内の複数のリンクに共通する接合点。 ノードには、プロセッサー、通信制御装置、クラスター制御装置、端末などがある。 ノードによって、経路指定などの機能が異なる場合がある。

ノード名 (node name). VTAM において、ネットワーク定義時に特定のメジャー・ノードまたはマイナー・ノードに割り当てられる記号名。

ノード・タイプ (node type). サポートするプロトコルや、ネットワーク内における役割に従って行われるノードの指定。 ノード・タイプは、元々は数値で (1、2.0、2.1、4、および 5 として) 指定されていたが、タイプ 2.1 ノードとタイプ 5 ノードが複数のプロトコル・タイプと役割をサポートするため、プロトコル・タイプ別に (APPN ネットワーク・ノード、LEN ノード、サブエリア・ノード、および交換ノード) より明確に指定されるようになった。

非固有 (nonnative). MPTN アーキテクチャーにおいて、異なるトランスポート・プロトコルにもとづいているトランスポート利用者とトランスポート提供者との間の関係に関連した用語。

無応答 (no response). SNA において、要求の受信側に、要求が正常に受信され処理されたかどうかに関係なく応答を一切戻さないように指示するために、要求ヘッ

ダーの応答形式要求フィールドで要求するプロトコル。*確定応答 (definite response)* および*例外応答 (exception response)* と対比。

通常フロー (normal flow). SNA において、基本的にエンド・ユーザーのデータを運ぶために使用される、伝送ヘッダーで示されるデータ・フロー。要求が通常フローを流れる比率は、セッション・レベル・ペーシングにより調整される。通常フローと急送フローは、1 次から 2 次の方向へも 2 次から 1 次の方向へも移動する。*急送フロー (expedited flow)* と対比。

通知 (notification). 発生した事象について、スケジュールなしで任意に生成されるレポート。

NOTIFY. システム・サービス制御点 (SSCP) から論理装置 (LU) 宛に送られるネットワーク・サービス要求。LU に、LU が要求したプロシージャーの状況を知らせる。

O

ODAI. 発信元-宛先割当て標識 (Origin-Destination Assignor indicator)。アドレス空間を分割するために使用する FID2 伝送ヘッダー内の 1 ビット。1 つのノードのアドレス空間管理者 (ASM) は、ODAI に 1 つの設定をして OAF′と DAF′ のすべての可能な組合せを使用でき、隣接ノードの ASM は ODAI に逆の設定をして OAF′ と DAF′ のすべての可能な組合せを使用できるようにする。

Off. 最新表示選択項目からのカスケード・メニューに現れる選択項目の 1 つ。これは最新表示機能をオフに設定する。

オフセット (offset). レコード、区域、または制御ブロックの中の任意の開始点から、ある他の点までを測定した場合の測定単位数。

OIA. 操作員情報域 (operator information area)。

OK. ウィンドウ内の情報を受け入れてそのウィンドウをクローズするための押しボタン。ウィンドウ内に変更された情報が含まれている場合は、ウィンドウをクローズするまでそれらの変更が適用される。

オン (On). 最新表示選択項目からのカスケード・メニューに現れる選択項目の 1 つ。ウィンドウ内のビューがただちに最新表示される。

オープン（する）(open). (1) 電気回路における遮断箇所。(2) アダプターを使用可能状態にすること。

オープン (open). ユーザーがオープンしたいオブジェクトを選択するためのウィンドウを表示する選択項目。

動作可能時間 (operable time). 機能単位が操作された場合に正しい結果をもたらす時間。 (I) (A)アップタイム *(uptime)* と同義。

オペレーティング・システム (operating system (OS)). プログラムの実行を制御するソフトウェアであって、資源割振り、スケジューリング、入出力制御、およびデータ管理などのサービスを提供することもある。オペレーティング・システムは原則としてソフトウェアであるが、部分的にハードウェア実装も可能である。 (T)

動作時間 (operating time). 機能単位が操作される、操作可能時間の一部分。 (A)

操作（operation）. オブジェクト指向の設計またはプログラミングにおいて、オブジェクトの境界で要求できるサービス。操作には、オブジェクトの変更や、オブジェクトに関する情報の表示などがある。

演算子、操作員（operator）. (1) 言語ステートメントにおいて、オペランドに対して行うアクションを示す字句エンティティー。*定義ステートメント (definition statement)* も参照。(2) MVS、NetView プログラム、IMS など、特定のソフトウェアが制御する活動を管理する責任を負う人またはプログラム。(3) 装置を操作する人。(4) システムの実行を維持する人。(5) *自動タスク (autotask)*、*ログオン操作員 (logged-on operator)*、*ネットワーク操作員 (network operator)*、および*操作員ステーション・タスク (operator station task)* も参照。

操作員情報域 (operator information area (OIA)). 表示域の下部にある区域であって、その区域に、端末またはシステムの状況情報が表示される。

操作員ステーション・タスク (operator station task (OST)). ネットワーク操作員とのオンライン・セッションを確立し維持するNetView タスク。NetView プログラムにログオンする各ネットワーク操作員について、操作員ステーション・タスクが 1 つずつある。*NetView 間タスク (NetView-NetView task)* を参照。

発信元 (origin). メッセージまたはその他のデータを発行する外部論理装置 (LU) またはアプリケーション・プログラム。*宛先 (destination)* も参照。

OS. オペレーティング・システム (operating system)。

アウトバウンド (outbound). 通信において、ネットワークに伝送されるデータ。

オーバーレー (overlay). 線、陰影付け、テキスト、ボックス、またはロゴなど、事前定義されたデータの集合。印刷時に、ページ上で可変データと組み合わせることができる。

P

PAC. 特権属性の保証 (Privilege Attribute Certificate)。

ペーシング. (1) 受信側の構成要素が、オーバーランや輻輳（ふくそう）を防ぐために送信側構成要素の伝送速度を制御する方法。(2) *受信ペーシング (receive pacing)*、*送信ペーシング (send pacing)*、*セッションレベル・ペーシング (session-level pacing)*、および*仮想経路 (VR) ペーシング (virtual route(VR) pacing)* を参照。(3) *フロー制御 (flow control)* も参照。

ペーシング・グループ (pacing group). *ペーシング・ウィンドウ (pacing window)* の同義語。

ペーシング応答 (pacing response). SNA において、受信側の構成要素が、他のペーシング・グループを受け入れることが可能な状態にあることを意味する標識。この標識は、セッション・レベル・ペーシングのために応答ヘッダー (RH) で送られ、仮想経路ペーシングのために伝送ヘッダー (TH) で送られる。

ペーシング・ウィンドウ (pacing window). (1) 仮想経路ペーシング応答を受信する前に仮想経路上を伝送できるパス情報単位 (PIU)。仮想経路の受信側が、経路上の後続の PIU を受信できる状態にあることを示す。(2) セッション・レベル・ペーシング応答を受信する前に、セッションの通常フローで一方向に伝送できる要求。受信側が、次の要求グループを受け入れることのできる状態にあることを示す。(3) *ペーシング・グループ (pacing group)* の同義語。

パケット・インターネット・グローパー (packet internet groper (PING)). (1) インターネット通信において、宛先にインターネット制御メッセージ・プロトコル (ICMP) のエコー要求を送り応答を待つことによって、宛先に到達する能力をテストするために TCP/IP ネットワークで使用されるプログラム。(2) 通信に置いて、到達可能度のテスト。

パケット・レベル (packet level). (1) 制御情報とユーザー・データを含むパケットを、データ端末装置 (DTE) とデータ回線終端装置 (DCE) との間で交換する、パケット・フォーマットと制御手順。 *データ・リンク・レベル (data link level)* および *物理レベル (physical level)* も参照。(2) X.25 勧告の一部分で、論理接続を 2 つの DTE で確立し、データをこれらの接続上を転送するためのプロトコルを定義する。

page. (1) 仮想記憶装置において、仮想アドレスを有し実記憶装置と補助記憶装置との間で 1 つの単位として転送される固定長ブロック。 (I) (A) (2) 表示装置の画面上で同時に表示される情報。(3) 画面上で表示された情報を、同じファイルの先行する情報または後続の情報と置換すること。

並行、並列 (parallel). (1) すべてのイベントが同じ時間間隔の中で起こり、各イベントが、別々ではあるが類似している機能単位によって処理されるプロセスに関する用語。例として、コンピューター・ワードのビットが内蔵バスの線に沿って転送される並列転送がある。 (T) (2) 複数の装置の同時的な動作または単一の装置における複数の活動の同時的実行に関する用語。 (A) (3) 複数の装置またはチャネルにおける複数の関連した活動の同時的発生に関する用語。 (A) (4) 複数のプロセスの同時性に関する用語。 (A) (5) 全体の中の個々の部分、たとえば文字の中の各ビット、ワードの中の各文字を、各部分のために別個の手段を使用して同時に処理することに関する用語。 (A) (6) *逐次*、*直列 (serial)* と対比。

並列セッション (parallel sessions). 使用しているネットワーク・アドレスまたはローカル形式セッション識別子ペアが異なる 2 つの同一ネットワーク・アクセス可能単位 (NAU) の間で、同時に活動状態にある複数のセッション。 各セッションがそれぞれ独自のセッション・パラメーターを持つことができる。

並列伝送グループ (parallel transmission groups). 各グループが個別の伝送グループ番号をもつ、隣接ノード間の複数伝送グループ。

パラメーター (parameter). (1) 指定されたアプリケーションに定数値を与える変数。また、そのアプリケーションを表示するもの。 (I) (A) (2) 基本 CUA アーキテクチャーにおいて、コマンドと共に使用されてコマンドの結果に影響を与える変数。(3) メニュー中の項目であって、ユーザーがその項目に対して値を指定するか、メニューが解釈されるときにシステムがその項目に対して値を提供するもの。(4) ユーザーまたはプログラムによって他のプログラムまたはプロシージャーへ渡されるデータ。すなわち、言語ステートメント中のオペランド、メニュー中の項目、または共用のデータ構造体。

親 (parent). フォーク・プリミティブを使用して、子プロセスを作成したプロセス。 *子 (child)* と対比。

構文解析 (parse). コマンドで入力されたオペランドを分析し、その情報からコマンド処理プログラムに関するパラメーター・リストを作成すること。

パートナー LU 検査 (partner-LU verification). 論理装置 (LU) 6.2 の場合、それぞれが LU-LU パスワードとデータ暗号化規格 (DES) アルゴリズムを使用する 2 つの LU の間で行われる 3 フローの交換。 3 フロー交換は、LU-LU 検査である。*エンド・ユーザー検査 (end-user verification)* を参照。

パスワード (password). (1) 認証のために使用される値、または特定の特権をもっている人々の集合の中でメンバーシップを確立するために使用される値。(2) コンピューター・システムおよびユーザーに知られている文字の独特のストリング。ユーザーは、システムおよびその中の情報へのアクセスを獲得するためには、その文字ストリングを指定する必要がある。(3) コンピューター・セキュリティーにおいて、コンピューター・システムとユーザーに知られた文字ストリング。ユーザーは、システムおよびその中に記憶されたデータへの無制限アクセスまたは制限アクセスを行うために、その文字ストリングを指定する必要がある。

パス. (1) ネットワークにおいて、2 つのノードの間の経路。 パスには複数のブランチが含まれることもある。 (T) (2) 2つのネットワーク・アクセス可能単位の間で交換する情報が通過する一連のトランスポート・ネットワーク構成要素（パス制御およびデータ・リンク制御）。*明示経路 (explicit route (ER))*、*経路拡張機能 (route extension (REX))*、および*仮想経路 (virtual route (VR))* も参照。 .*** 9.***1'.

パス制御 (path control (PC)). ネットワーク内のネットワーク・アクセス可能単位間でメッセージを経路指定し、それらの単位間のパスを提供する機能。 この機能は、伝送制御からの基本情報単位 (BIU) を、多くの場合セグメント化によってパス情報単位 (PIU) に変換し、1 つまたは複数の PIU を含む基本伝送単位をデータ・リンク制御との間で交換する。 パス制御はノード・タイプによって異なる。すなわち、あるノード(たとえば、APPN ノード) は経路指定のためにローカルで生成されたセッション識別子を使用し、他のノード (サブエリア・ノード) は経路指定のためにネットワーク・アドレスを使用する。

パス情報単位 (path information unit (PIU)). 伝送ヘッダー (TH) のみ、またはTH とそれに続く基本情報単位 (BIU) または BIU セグメントから構成されるメッセージ単位。

PC. (1) パーソナル・コンピューター (personal computer)。(2) パス制御 (path control)。(3) Communications Server.

PCID. プロシージャー相関 ID (procedure-correlation identifier)。

対等、対等機能 (peer). ネットワーク・アーキテクチャーにおいて、他のエンティティーと同じ層にある機能単位。 (T)

周辺 PU (peripheral PU). SNA において、周辺ノード中の物理装置。 *サブエリア PU (subarea PU)* と対比。

持続検査 (persistent verification). VTAM において、2 つの論理装置が、セッションの最初の会話でそれぞれの ID を検査し、そのセッション期間はそれ以降の会話が検証済みであるとみなすことができるようにするセキュリティー機能。

Communications Server プロダクト・ファミリー (Communications Server product family). 3270 端末と 5250 端末をエミュレートし、複数のオペレーティング・システム (OS/2、DOS、および Windows など) で稼動する、IBM ライセンス・プログラムのグループ。

パーソナル・コンピューター (personal computer (PC)). (1) 主として個人が独立して使用するためのマイクロコンピューター。 (T)(2) 通常、システム装置、表示モニター、キーボード、ディスケット・ドライブ、内蔵固定ディスク記憶装置、およびオプションのプリンターから構成されるデスクトップ型、床置き式、または携帯用マイクロコンピューター。 PC は、主として単独使用を目的として設計されているが、メインフレームまたはネットワークにも接続できる。

物理回線 (physical circuit). 多重化なしに確立される回線。 *データ回線 (data circuit)* も参照。 *仮想回線 (virtual circuit)* と対比。

物理レベル (physical level). X.25 において、データ端末装置 (DTE) とデータ回線終端装置 (DCE) との間で物理リンクを活動化、維持、非活動化するために使用する、機械的、電気的、機能的、および手続き的手段。 *データ・リンク・レベル (data link level)* および *パケット・レベル (packet level)* を参照。

物理装置 (physical unit (PU)). (1) SSCP-PU セッションを介した SSCP からの要求に応じて、ノードに関連した資源（接続しているリンクおよび隣接リンク・ステーションなど）を管理しモニターする構成要素。 SSCP は、接続しているリンクなどのノード資源を、PU を介して間接的に管理するために、物理装置とのセッションを活動化する。 この用語は、タイプ 2.0、タイプ 4、およびタイプ 5 のノードのみに適用される。(2) *周辺 PU (peripheral PU)* およびサブエリア *PU (subarea PU)* も参照。

物理装置 (PU) サービス (physical unit (PU) services). SNA において、SSCP-PU セッション用の構成サービスおよび保守サービスを提供する、物理装置 (PU) 内の構成要素。

PING. パケット・インターネット・グローパー (packet internet groper)。

PIP. プログラム初期設定パラメーター (program initialization parameters)。

パイプ (pipe). 1 つの処理からの出力が別の処理への入力になるようにデータを導くこと。

PLU. 1 次論理装置 (primary logical unit)。

ポインター (pointer). (1) 別のデータ要素の位置を示すデータ要素。 (T) (2) データの項目の位置を示す識別子 (A)

2 地点間 (point-to-point). ディスプレイ装置やコンピューターを仲介させずに 2 地点間でデータ伝送を行うこと。

ポーリング (polling). (1) 分岐接続または 2 地点間接続において、データ装置に対して一度に 1 回ずつ送信するように促す処理。 (I) (2) 競合を避ける、動作状況を調べる、またはデータの送受信が可能であるかどうかを調べるための、装置に対する問合せ。 (A)

ポップする （pop). プッシュダウン・リストの最上部から項目を除去すること。 *プッシュする (push)* と対比。

POP. ポスト・オフィス・プロトコル (Post Office Protocol) を算用。

ポート（port）. (1) データの入口または出口となるアクセス・ポイント。(2) 表示端末やプリンターなどの別の装置のケーブルに接続される、装置のコネクター。(3) リンク・ハードウェアに対する物理接続を表す。ポートはアダプターと呼ばれることもあるが、アダプター上には複数のポートが存在し得る。単一の DLC プロセスによって制御されるポートが複数ある場合もある。(4) インターネットのプロトコルの組において、TCP またはユーザー・データグラム・プロトコル (UDP) と上位のプロトコルまたはアプリケーションとの間の通信のために使用される16 ビット数。 ファイル転送プロトコル (FTP) や簡易メール転送プロトコル (SMTP) などの一部のプロトコルは、TCP/IP ではすべて同じ事前割当てポート番号を使用する。(5) ホスト・マシン内の複数の宛先を区別するためにトランスポート・プロトコルが使用する抽象概念。(6) *ソケット (socket)* と同義。

ポート番号 (port number). インターネット通信において、伝送サービスに対するアプリケーション・エンティティーの識別。

肯定応答 (positive response). SNA において、要求が受け取られ処理されたことを示す応答。 *否定応答 (negative response)* と対比。

ポスト・オフィス・プロトコル (Post Office Protocol (POP)). ネットワーク・メールの交換およびメールボックスへのアクセスに使用するプロトコル。

Prepare. コミット処理の一環として送られ、パートナーが 2 フェーズ・コミット・プロセスの第 1 フェーズを開始したことを示す表示サービス・ヘッダー。

表示空間 ID (presentation space ID (PSID)). コミュニケーション・マネージャー/2 において、*短縮名 (shortname)* の同義語。

1 次フォーカル・ポイント (primary focal point). 特定のカテゴリーをサポートする管理サービスの、優先的に使用されるソースとみなされるフォーカル・ポイント。 *バックアップ・フォーカル・ポイント (backup focal point)* と対比。

1 次論理装置 (primary logical unit (PLU)). SNA において、BIND を送信してそのパートナー LU でのセッションを活動化する論理装置 (LU)。 *2 次論理装置 (secondarylogical unit (SLU))* と対比。

特権属性の保証 (Privilege Attribute Certificate (PAC)). 分散コンピューティング機環境 (DCE) において、ユーザーまたは管理担当者がオブジェクトに対するアクセス権限を確立するために示すことのできる、アクセス特権の保証されたセット。

問題判別 (problem determination). 問題の原因を決定するプロセス。たとえば、原因がプログラム構成要素で

あるか、マシン障害であるか、電気通信設備の故障であるか、ユーザーまたは契約会社が導入したプログラムまたは機器であるか、電源切断のような環境の障害であるか、またはユーザー・エラーであるかなどを決定する。

プロシージャー、手順 (procedure). (1) プログラム言語において、プロシージャー呼出しにより呼び出して実行する、形式パラメーター付きまたは形式パラメーターなしのブロック。 (I) (2) 問題の解決のために行う一連のアクションの記述。 (A)

プロシージャー相関 ID (procedure-correlation identifier (PCID)). SNA において、指定のプロシージャーに関連するすべての要求と応答を相関するために使用する値のこと。

処理する、プロセス（process）. (1) プロセスにおいて、データの操作を実行すること。 (I) (A) (2) データ処理において、プログラムの全体または一部分の実行中に起こる複数のイベントの推移。 (T) (3) 複数のイベントの推移であって、各イベントがその推移の目的または効果によって定義され, また所与の条件の下で達成されるもの。(4) データに対する 1 つの操作、または操作の組合せ。(5) 実行されている機能、または実行を待機している機能。

プロセッサー (processor). コンピューターにおいて、命令を解釈し実行する機能単位。プロセッサーは、少なくとも 1 つの命令制御装置および算術論理装置から成っている。 (T)

プロダクトセット識別 (product-set identification (PSID)). (1) SNA において、ネットワーク構成要素を実施しているハードウェア製品およびソフトウェア製品を識別する技法。(2) 定義内に記述された情報をトランスポートする管理サービス共通サブベクトル (1)。

プロファイル (profile). 単独のユーザー、ユーザーのグループ、または単独または複数のコンピューター資源の顕著な特性を記述するデータ。

プログラム（program）. (1) コンピューターの処理に適した命令のシーケンス。 処理には、プログラムを実行することのほかに、アセンブラー、コンパイラー、インタープリター、または変換プログラムを使用して、プログラムを実行できるようにするための準備も含まれる。 (I) (2) プログラミング言語において、相互に関連付けられた 1 つまたは複数のモジュールの論理集合。 (I) (3) コンピューター・プログラムを設計、作成、およびテストすること。 (I) (A)

プログラム初期設定パラメーター (program initialization parameters (PIP)). ターゲット・プログラムに入力として渡されるか、処理環境をセットアップするために使用される、初期パラメーター値。

プログラム式操作員機能 (programmable operator facility (PROP)). 仮想マシンのリモート制御を、そのマシンに宛先指定されたメッセージを代行受信し、事前にプログラムされた処置を行うことにより可能にする VM 機能。

プログラム一時修正 (program temporary fix (PTF)). 現在未変更のプログラムのリリースで IBM が診断した問題を、一時的に解決または迂回すること。

PROP. プログラム式操作員機能 (programmable operator facility)。

プロトコル (protocol). (1) 通信を達成するうえで機能単位の行動を決定する意味的および構文的規則の集合。 (I) (2) 開放型システム間相互接続アーキテクチャーにおいて、通信機能を達成する場合に同じ層のエンティティーの行動を決定する意味的および構文的規則の集合。 (T) (3) SNA において、ネットワークの管理、データの伝送、およびネットワーク構成要素の状態の同期化に使用される要求と応答の意味と順序付け規則。*回線制御規則 (line control discipline)* および*伝送制御手順 (line discipline)* と同義。 *ブラケット・プロトコル (bracket protocol)* および*リンク・プロトコル (link protocol)* を参照。

PSID. 表示空間 ID (presentation space ID)。

PTF. プログラム一時修正 (program temporary fix)。

PU. 物理装置 (physical unit).

プッシュする (push). プッシュダウン・リストの最上部に項目を付け加えること。 *ポップする (pop)* と対比。

プッシュダウン・リスト (pushdown list). (1) 検索されるべき次のデータ要素が最初に記憶されるように構成され保守されるリスト。 (T) (2) *スタック (stack)* と同義。

PUT. プログラム更新テープ (program update tape)。

PU タイプ (PU type). (1) *ノード・タイプ (node type)* の不適語。(2) あるノードの物理装置のタイプ。

Q

待ち行列、待ち行列化 (queue). (1) 検索されるべき次のデータ要素が最初に記憶されるように構成され保守されるリスト。 (T) (2) 処理を待機している項目の行またはリスト。たとえば、実行されるべき作業、表示されるべきメッセージ。(3) 待ち行列として配列すること、または、待ち行列を形成すること。

R

高速トランスポート・プロトコル (RTP). 高性能経路指定 (HPR) 経路を介してセッション・トラフィックを搬送するための、接続指向の全二重トランスポート・プロトコル。 *自動ネットワーク経路指定 (automatic network routing (ANR))* および*Rapid Transport Protocol (RTP) 接続 (Rapid Transport Protocol (RTP)connection*) も参照。

Rapid Transport Protocol (RTP) 接続 (Rapid Transport Protocol (RTP) connection). 2 つの高性能経路指定 (HPR) ノード間の接続。1 つまたは複数の中間 HPR ノードおよびリンクが介在していることもある。 接続端点とは、エラー回復および接続通信データの最適レートによるフロー制御を行い、また、経路に障害が発生した場合には通信を中断せずに物理パスの切替えを行う。中間 HPR ノードは、自動ネットワーク経路指定 (ANR) プロトコルを使用して経路指定のオーバーヘッドを最小限に抑える。ANR プロトコルは、ヘッダー情報にもとづいて、効率的な資源の経路指定とRTP 接続を通した優先順位をつけた伝送を行う。

RAR. 経路追加抵抗機能 (route addition resistance)。

組立て (reassembly). 通信において、セグメント化したパケットを、受信後にひとつに戻す処理。

受信ペーシング (receive pacing). SNA において、構成要素が受け取っているメッセージ単位のペーシング。*送信ペーシング (send pacing)* と対比。

レコード (record). 1 つの単位とみなされるデータのセット。 (T)

解放（する）、リリース (release). (1) 新製品の配布、または既存製品に関する新機能および APAR 修正の配布。通常は、新リリースの発行後一定期間が経過したあとは、旧リリースに関するプログラミング・サポートは廃止される。製品の最初のバージョンは、リリース 1 修正レベル 0 として発表される。(2) VTAM において、資源（通信制御装置または物理装置）の制御権を放棄すること。*資源引継ぎ (resource takeover)* も参照。*獲得（する）(acquire)* と対比。

リモート (remote). (1) 通信回線によってアクセスされるシステム、プログラム、または装置に関する用語。(2) *リンク接続 (link-attached)* の同義語。(3) *ローカル (local)* と対比。

リモート・ホスト (remote host). ネットワーク上で、特定の操作員が作業しているホストを除くすべてのホスト。*外部ホスト (foreign host)* と同義。

要求 (request). 特定のアクションまたはプロトコルの開始を知らせるメッセージ。たとえば、Initiate-Self は LU-LU セッションの活動化の要求。

リクエスター. サーバーを介して共有ネットワーク資源にアクセスするコンピューター。*クライアント (client)* の同義語。

要求ヘッダー (request header (RH)). 要求単位 (RU) の前に付く制御情報。 *要求応答ヘッダー (request/response header (RH))* も参照。

要求応答ヘッダー (request/response header (RH)). 特定の RU に関連付けられる制御情報。 RH は、要求応答単位 (RU) の前に付き、RU (要求単位または応答単位) のタイプを指定する。

要求応答単位 (request/response unit (RU)). 要求単位または応答単位を総称する用語。 *要求単位 (request unit (RU))* および *応答単位 (response unit (RU))* を参照。

要求単位 (request unit (RU)). 制御情報またはエンド・ユーザー・データ、あるいはその両方を含むメッセージ単位。

リセットする（reset）. 仮想回線では、データ・フロー制御の再初期設定。 リセットすると、転送中のすべてのデータが除去される。

資源 (resource). ジョブまたはタスクのために必要なコンピューティング・システムまたはオペレーティング・システムの機能であって、主記憶装置、入出力装置、処理装置、データ・セット、および制御または処理プログラムを含む。

資源登録 (resource registration). ネットワーク・ノード・サーバーまたは中央ディレクトリー・サーバーに対してLU などの資源の名前を識別するプロセス。

資源順序番号 (resource sequence number (RSN)). ネットワーク・トポロジー・データベースにおいて、資源の更新を識別する値。

資源引継ぎ (resource takeover). VTAM において、接続が切れたり、接続上の既存の LU-LU セッションが中断されたりすることなく定義域から他の定義域へと資源の制御を転送するために、ネットワーク操作員が開始するアクション。*獲得（する）(acquire)* および*解放（する）(release)* も参照。

資源タイプ (resource types). NetView プログラムにおいて、パネルの編成を記述する概念。資源タイプは、中央処理装置、チャネル、制御装置、および入出力装置が 1 つのカテゴリーとして定義され、通信制御装置、アダプター、リンク、クラスター制御装置、および端末が別のカテゴリーとして定義される。資源タイプとデータ・タイプおよび表示タイプとの組合せにより、表示編成が記述される。*データ・タイプ (data types)* および*表示タイプ (display types)* も参照。

応答（response）. (1) データ通信において、応答フレームの制御フィールドで表現された応答。 1 次ステーションまたは複合ステーションに、2 次ステーションまたは他の複合ステーションが 1 つまたは複数のコマンドに対して行ったアクションを知らせる。(2) コマンド *(command)*も参照。

応答ヘッダー (response header (RH)). (1) 応答が肯定されたか否定されたかを示し、ペーシング応答を含むこともあるヘッダー。任意で応答単位 (RU) が後に続く。(2) *否定応答 (negative response)*、*ペーシング応答 (pacing response)*、および*肯定応答 (positive response)* も参照。

応答単位 (response unit (RU)). 要求単位を認識するメッセージ単位。 要求単位で受信した情報をプレフィックとして含む場合がある。 肯定であれば、応答単位に追加の情報 (BIND SESSION に応答するセッション・パラメーターなど) を含むことがある。 否定であれば、応答単位には例外条件を定義したセンス・データが含まれる。

戻りコード (return code). (1) 後続の命令に影響を与えるために使用されるコード。 (A) (2) プログラムによって要求された操作の結果を示すため、そのプログラムに戻される値。

REX. 経路の拡張 (Route extension) を参照。

RH. 要求応答ヘッダー (request/response header)。

リング（ring）. *リング・ネットワーク（ring network）* を参照。

リング・ネットワーク（ring network）. ネットワーク構成において、装置が単一方向の伝送リンクによって接続され、閉鎖経路を形成しているネットワーク構成。

経路 (route). (1) 発信元ノードから宛先ノードへのパスを形成し、両ノード間で交換されるトラフィックが通過する、一連のノードおよび伝送グループ (TG) の並び。(2) ネットワーク・トラフィックが起点から宛先に達するため使用するパス。

経路追加抵抗機能 (route addition resistance (RAR)). 中間セッション経路指定を実行するネットワーク・ノードの機能を示す値。

ルート d (routed). 「ルートディー」と発音する。 *経路指定デーモン (route daemon)* を参照

経路指定デーモン (route daemon). 4BSD UNIX** のもとで実行し、ローカル・エリア・ネットワークのマシンの間で情報の経路指定を伝搬するプログラム。 *ルート d (routed)* (「route-d」 と発音する) ともいう。

経路拡張機能 (route extension (REX)). SNA におけるパス制御ネットワーク構成要素。この構成要素は周辺リンクを含み、またサブエリア・ノードと、隣接した周辺ノードの中のネットワーク・アドレス可能単位 (NAU) との間のパス部分を形成している。 *明示経路 (explicit route (ER))*、*パス (path)*、および*仮想経路 (virtual route (VR))* も参照。

ルート選択制御ベクトル (Route Selection control vector (RSCV)). APPN ネットワーク内の経路を説明する制御ベクトル。 RSCV は、発信元ノードから宛先ノードへのパスを構成する TG とノードを識別する制御ベクトルのオーダー順序から構成されている。

ルーチン (routine). 汎用または使用頻度の高いプログラムまたはプログラムの一部。 (T)

経路指定 (routing). (1) ネットワークを介してメッセージを伝送するために使用されるパスを決定するプロセス。 (T) (2) メッセージが宛先に到達するために使用するパスを割り当てること。(3) SNA において、メッセージ単位の中にあるパラメーターに従って、ネットワークを通る特定のパスに沿ってそのメッセージ単位を渡すこと。パラメーターの例としては、伝送ヘッダーの中にある宛先ネットワーク・アドレスがある。

RouTing update Protocol (RTP). 経路指定データベースを保持し、VINES ノード間の経路指定情報の交換を可能にするVIrtual NEtworking System (VINES) プロトコル。*Internet Control Protocol (ICP)* も参照。

RQD. 要求未着信 (request discontact)。

RSCV. ルート選択制御ベクトル (Route Selection control vector)。

RSN. 資源順序番号 (resource sequence number)。

RTP. 高速トランスポート・プロトコル (Rapid Transport Protocol)。

RTP 接続 (RTP connection). *高速トランスポート・プロトコル (Rapid Transport Protocol)* を参照。

RU. 要求/応答単位 (request/response unit)。

RU 連鎖 (RU chain). SNA において、特定の通常または急送データ・フローの上を連続的に伝送される、関連した要求/応答単位 (RU) の集合。 要求 RU 連鎖は回復単位である。すなわち、連鎖中のRU の 1 つが処理不可能であれば、全体の連鎖が破棄される。各 RU はそれぞれ 1 つの連鎖だけに所属し、RU の始めと終りは、RU 連鎖内の要求応答ヘッダーの制御ビットにより指示される。各 RU は、先頭連鎖 (FIC)、最終連鎖 (LIC)、中間連鎖 (MIC)、または単独連鎖 (OIC) のいずれかとして指定できる。 応答単位および急送フロー要求単位は、常に「単独連鎖」として送信される。

S

SAP. (1) サービス・アクセス・ポイント (service access point)。(2) Service Advertising Protocol。

SBCS. 1 バイト文字セット (single-byte character set)。

SC. Session control.

SDLC. 同期データ・リンク制御 (Synchronous Data Link Control)。

2 次論理装置 (secondary logical unit (SLU)). SNA において、特定の LU-LU セッションの 2 次側ハーフ・セッションを含んでいる論理装置 (LU) のこと。 1 つの LU に、別の活動状態の LU-LU セッションの2 次側ハーフ・セッションと 1 次側ハーフ・セッションを含むことができる。 *1 次論理装置 (primarylogical unit (PLU))* と対比。

2 次論理装置 (SLU) キー (secondary logical unit (SLU) key). セッション暗号キーを、2 次側ハーフ・セッションへ伝送する間、保護するために使用するキー暗号化キー。

セグメント (segment). (1) 複数の構成要素または装置の間のケーブル部分。 セグメントは、単一のパッチ・ケーブル、相互に接続されたいくつかのパッチ・ケーブル、または、相互に接続された構築ケーブルおよびパッチ・ケーブルの組合せから成る。(2) インターネット通信において、異なるマシンの TCP 機能間での転送の単位。 各セグメントには、制御フィールドとデータ・フィールドが含まれている。受信したデータの妥当性検査のために、現行バイト・ストリーム位置および実データ・バイトが、チェックサムとともに識別される。(3) *BIU セグメント (BIU segment)* の同義語。(4) *リンク接続セグメント (link connection segment)* も参照。

選択する (select). あとで選択動作を適用するために 1 つまたは複数のオブジェクトを明確に識別すること。

選択 (selection). 後続の選択項目を適用する 1 つまたは複数のオブジェクトを明示的に識別するプロセス。

送信ペーシング (send pacing). SNA において、構成要素が送信しているメッセージ単位のペーシング。 *受信ペーシング (receive pacing)* と対比。

順序番号 (sequence number). (1) 通信において、伝送フローとデータの受信を制御するために特定のフレームまたはパケットに割り当てられる数字。(2) VTAM から、2 つのノード間で交換される各メッセージに割り当てられる数値。この値 (アプリケーション・プログラムから論理装置に送られるメッセージの値と、論理装置からアプリケーション・プログラムに送られるメッセージの値) は、アプリケーション・プログラムが STSN (set and test sequence number) 標識によりリセットしない限り、メッセージが伝送されるたびに 1 つずつ増加する。

逐次、直列 (serial). (1) すべてのイベントが順次に起こるようなプロセスに関する用語。たとえば、V24 CCITT プロトコルに従って文字ビットを順次に伝送する。 (T) (2) 1 つの装置またはチャネルで、関連した複数の活動が連続して順次に起こることに関する用語。 (A) (3) 全体の中の個々の部分 (たとえば、文字を構成するビットや語を構成する文字など) を、連続する各部分について同じ機能を使用して、順次に処理することを指す用語。(4) *並行、並列 (parallel)* と対比。

サーバー. (1) ネットワークを介してワークステーションに共通サービスを提供する機能単位。たとえば、ファ

イル・サーバー、プリント・サーバー、メール・サーバーなどがある。 (T) (2) ネットワークにおいて、他のステーションに機構または機能を提供するデータ・ステーション。たとえば、ファイル・サーバー、プリント・サーバー、メール・サーバーなどがある。 (A) (3) AIX オペレーティング・システムにおいて、通常、バックグラウンドで実行され、システム・プログラム制御手段によって制御されるアプリケーション・プログラム。(4)拡張 X-Windows** ツールキットにおいて、基本ウィンドウ・メカニズムを提供するプログラム。クライアントからのプロセス間通信 (IPC) 接続を操作し、画面上でグラフィックス要求の多重化を解除し、そして入力を再度多重化してクライアントに戻す。

サービス・アクセス・ポイント (service access point (SAP)). (1) 開放型システム間相互接続 (OSI) アーキテクチャーにおいて、ある層のサービスがその層のエンティティーによって次の高位層のエンティティーに対して提供されるポイント。 (T) (2) アダプターによって利用可能となる、情報の送受信を行うことのできる論理ポイント。 1 つのサービス・アクセス点は、そこで終了する多数のリンクをもつことができる。(3) 制御装置のゲートウェイ・アドレス。SAP は、制御装置をホスト・システムにリンクするための接続点となる。

Service Advertising Protocol (SAP). インターネットワーク・パケット交換機能 (Internetwork Packet Exchange** (IPX**)) において、以下のメカニズムを提供するプロトコル。

- インターネット上の IPX サーバーがそのサービスを名前とタイプによって公示できるようにするメカニズム。 このプロトコルを使用するサーバーは、その名前、サービス・タイプ、および IP アドレスを、NetWare** を実行しているすべてのファイル・サーバーに記録する。
- ワークステーションが、あらゆるタイプのすべてのサーバーの ID、特定のタイプのすべてのサーバーの ID、あるいは特定のタイプの最も近いサーバーの IDを入手するために、照会を同報通信できるようにするメカニズム。
- ワークステーションが、特定のタイプのすべてのサーバーの名前とアドレスを入手するために、NetWare を実行しているすべてのファイル・サーバーに照会できるようにするメカニズム。

サービス・ポイント・コマンド機能 (service point command facility (SPCF)). ネットワーク操作員、リンク接続構成要素マネージャー (LCCM)、およびリンク接続サブシステム・マネージャー (LCSM) の間でデータや制御を交換するプログラムまたは機能。

サービス・トランザクション・プログラム (service transaction program). ネットワーク・アクセス可能単位 (NAU) で実行する、IBM 提供のトランザクション・プログラム。 *アプリケーション・トランザクション・プログラム (application transaction program)* と対比。

セッション. (1) ネットワーク・アーキテクチャーでは、複数の機能単位の間でデータ通信を行うためのすべての活動であって、接続の確立、維持、および解放の間に起こるもの。 (T) (2) 2 つのネットワーク・アクセス可能単位（NAU）の間の論理接続。セッションは、活動化し、各種プロトコルを提供するように調整し、要求にもとづいて非活動化することができる。 各セッションは、セッション中に交換される各伝送に付随する伝送ヘッダー (TH) の中で、固有のものとして識別される。(3) サーバーとリクエスターとの間の接続であって、共有資源への要求が成功したときに開始されるもの。ホスト・セッション *(host session)*および*DOS セッション (DOS session)*も参照。

セッション活動化要求 (session activation request). SNA において、2 つのネットワーク・アクセス可能単位(NAU) の間でセッションを活動化し、セッション活動の間に種々のプロトコルを制御するセッション・パラメーターを指定する要求。たとえば、BIND および ACTPU がある。 *セッション非活動化要求 (session deactivation request)* と対比。

セッション制御 (session control (SC)). SNA において次のいずれか 1 つ。

- 伝送制御の構成要素の 1 つ。セッション制御は、回復不能エラーが発生した後でセッション内を流れるデータを除去したり、その種のエラーの後でデータ・フローを再同期したり、また暗号検査を行ったりするために使用される。
- セッションのセッション制御構成要素間で交換する要求および応答用として、また、セッションの活動化および非活動化の要求および応答用として使用される要求単位 (RU) カテゴリー。

セッション・データ (session data). NetView プログラムが収集する、セッションに関するデータ。セッション認識データ、セッション・トレース・データ、およびセッション応答時間データから成る。

セッション非活動化要求 (session deactivation request). SNA において、2つのネットワーク・アクセス可能単位(NAU) の間でセッションを非活動化する要求。 たとえば、UNBIND およびDACTPU などがある。 *汎用 UNBIND (generic unbind)* と同義。 *セッション活動化要求 (session activation request)* と対比。

セッション ID (session ID). Communications Server によって各セッションまたは画面へ割り当てられるアルファベットの ID (*a* から *h*)。この ID はすべての種類のホスト・セッションに適用され、ファイル転送時に使用される。*短縮名 (short name)*も参照。

セッション・レベル・ペーシング (session-level pacing). (1) 受信側ハーフセッションまたはセッション・コネクターが、通常フローでのデータ転送速度（要求単位を受信する速度）を制御できるようにするフロー制御の技法。 この技法は、送信側での要求生成の速度が受信側の処理能力を上回る場合に、未処理の要求により受信側に過負荷が発生するのを防ぐために使用される。(2) *最適セッション・レベル・ペーシング (adaptive session-levelpacing)*、*固定セッション・レベル歩調合せ (fixed session-level pacing)*、および*仮想経路ペーシング (virtual route (VR))* も参照。

セッション・レベル・セキュリティー (session-level security). 論理装置 (LU) 6.2 では、パートナー LU 検査とセッション暗号化のこと。 *会話レベル・セキュリティー (conversation-level security)* を参照。

セッション限度 (session limit). 特定の論理装置 (LU) がサポートすることのできる、一度に活動状態にできる LU-LU セッションの最大数。

共用 (shared). 1つの資源を同時に複数の用途に使用できることを表す。

シフトアウト文字 (shift-out character (SO)). 標準文字セットのグラフィック文字の代わりに、合意に達しているかまたはコード拡張手順により指定してある代替グラフィック文字セットを使用するためのコード拡張文字。 (I) (A)

短縮名 (short name). (1) Communications Server において、操作員情報域の 7 列目に表示されるセッション ID を示す文字。*セッション ID (session ID)* および*操作員情報域 (operator information area)* も参照。(2) コミュニケーション・マネージャー/2 において、表示空間またはエミュレーション・セッションの1文字の名前 (AからZ)。 *表示空間 ID (presentation space ID (PSID))* および*短セッション ID (short-session ID)*と同義。

短セッション ID (short-session ID). コミュニケーション・マネージャー/2 において、*短縮名 (shortname)* の同義語。

遮断 (shutdown). システムまたはサブシステムの操作を、定義された手順に従って終了する処理。

1 バイト文字セット (single-byte character set(SBCS)). 各文字を 1 バイト・コードで表す文字セット。*2 バイト文字セット (double-byte character set (DBCS))* と対比。

SLU. 2 次論理装置 (secondary logical unit)。

SNA. システム・ネットワーク体系 (Systems Network Architecture)。

SNA 管理サービス (SNA management services (SNA/MS)). SNA ネットワークの管理を支援するために提供されるサービス。

SNA ネットワーク (SNA network). システム・ネットワーク体系の形式とプロトコルに合わせたユーザー・アプリケーション・ネットワークの部分のこと。 エンド・ユーザー間の信頼性のあるデータ転送を可能にし、さまざまなネットワーク構成の資源を制御するプロトコルを提供する。 SNA ネットワークは、ネットワーク・アクセス可能単位 (NAU)、境界機能構成要素、およびパス制御ネットワークによって構成される。

SO. シフトアウト文字を参照。 (I) (A)

ソケット（socket）. プロセスやアプリケーション・プログラム間の通信の端点。

SPCF. サービス・ポイント・コマンド機能 (service point command facility)。

特定モード (specific-mode). VTAM において、以下のことを指す。

- ある特定のセッションからの入力を得るための RECEIVE 要求の形式。
- ある特定の待機 CINIT を受け入れることによって、セッションの確立を完了する ACCEPT 要求の形式。

不特定モード (any-mode) と対比。*継続特定モード (continue-specific mode)* を参照。

制御範囲 (sphere of control (SOC)). 1 つの管理サービス・フォーカル・ポイントから提供される制御点定義域のセット。

SSCP. システム・サービス制御点 (system services control point)。

SSCP 従属 LU (SSCP-dependent LU). LU-LU セッションの開始のためにシステム・サービス制御点 (SSCP) からの援助を必要とする LU。 SSCP-LU セッションを必要とする。

SSCP-LU セッション（SSCP-LU session）. SNA において、システム・サービス制御点（SSCP）と論理装置（LU）との間のセッション。 このセッションで、LU は LU-LU セッションを開始するための援助を SSCP に要求できる。

SSCP-PU セッション (SSCP-PU session). SNA において、システム・サービス制御点 (SSCP) と物理装置 (PU) の間のセッション。SSCP-PU セッションにより、SSCP は個々のノードとの間で要求の送信や状況情報の受信を行い、ネットワーク構成を制御することができる。

スタック (stack). *プッシュダウン・リスト (pushdown list)* の同義語。

静的 (static). (1) プログラミング言語において、プログラムを実行する前に確立できる特性に関する用語。たとえば、固定長変数の長さは静的である。 (I) (2) 事前定義された時刻または固定の時刻に行われる操作のこと。 (3) *動的 (dynamic)* と対比。

ステーション（station）. 通信機能を使用する、システムの入力または出力ポイント。たとえば、通信回線を介してデータの送信または受信を行える、特定の位置にある 1 つまたは複数のシステム、コンピューター、端末、装置、および関連プログラム。

状況（status）. 通常状況コードで表される、ハードウェアまたはソフトウェアの条件または状態。

記憶装置、格納 (storage). (1) データを入れておき、データを検索することのできる機能単位。 (T) (2) データを記憶装置に入れるアクション。 (I) (A) (3) 記憶装置 (storage device)。 (A)

サブエリア (subarea). SNA ネットワークの一部分であって、サブエリア・ノード、接続された周辺ノード、および関連の資源から構成されている。 サブエリア・ノード内では、すべてのネットワーク・アクセス可能単位 (NAU)、リンク、およびそのサブエリア・ノード内でアドレス可能な隣接リンク・ステーション（接続している周辺ノードまたはサブエリア・ノード内の）が、共通のサブエリア・アドレスを共用し、それぞれ異なる要素アドレスを持っている。

サブエリア・ノード (subarea node (SN)). 経路指定用にネットワーク・アドレスを使用し、ネットワークの構成を反映する経路指定テーブルを維持するノード。 サブエリア・ノードは、複数のサブエリア・ネットワークを接続するゲートウェイ機能、中間経路指定機能、および、周辺ノード用の境界機能サポートを提供する。 サブエリア・ノードとなることができるのは、タイプ 4 およびタイプ 5 のノードである。

サブエリア PU (subarea PU). SNA において、サブエリアの中にある物理装置。 *周辺 PU (peripheral PU)* と対比。

サブディレクトリー (subdirectory). ファイル・システム階層において、1 つのディレクトリーの中に含まれた他のディレクトリー。

サブシステム (subsystem). 2 次的または従属的なシステム。通常、このシステムは制御を行っているシステムとは独立して(すなわち非同期的に) 動作することができる。 (T)

サブベクトル (subvector). ネットワーク管理ベクトル・トランスポート (NMVT) メジャー・ベクトル。

交換ネットワーク (switched network). たとえばダイヤル呼出しによりスイッチを閉じることによって接続が確立されるネットワーク。

同期点 (synchronization point). *同期点 (sync point)* の同義語。

同期 (synchronous). (1) 共通タイミング信号のように、特定のイベントの発生に依存する複数のプロセスに関する用語。 (T) (2) 定期的または予測可能な関係で発生することに関する用語。

同期データ・リンク制御 (Synchronous Data Link Control (SDLC)). 米国規格協会 (ANSI) の拡張データ通信制御手順(ADCCP) および国際標準化機構のハイレベル・データ・リンク制御 (HDLC) のサブセットに準拠した規則であって、リンク接続を介して行われる同期、コード透過、ビット順のデータ伝送を管理するもの。伝送交換は、交換リンクまたは非交換リンクを介して二重または半二重で行う。 リンク接続の構成は、2 地点間、マルチポイント、またはループ接続になる。 (I)

同期操作 (synchronous operation). VTAM において、操作要求を受け取ってからその操作が完了するまでの間 VTAM が制御をプログラムに戻さない、通信その他の操作。*非同期操作 (asynchronous operation)* と対比。

同期要求 (synchronous request). VTAM において、同期操作の要求。*非同期要求 (asynchronous request)* と対比。

同期点 (sync point). トランザクションの処理において、そのトランザクションの 1 つまたは複数の保護された資源に対する更新または変更が、論理的に完了してエラーなしの状態となる中間点または終点。*同期点 (synchronization point)* と同義。

システム (system). データ処理において、特定の機能の集合を実行するように組織化された人、マシン、および方法より成るもの。 (I) (A)

システム・サービス制御点 (system services control point (SSCP)). サブエリア・ネットワーク内の構成要素の 1 つで、構成を管理し、ネットワーク操作員および問題判別要求を調整し、ネットワーク・ユーザーにディレクトリー・サービスおよびその他のセッション・サービスを提供する。 複数の SSCP が対等機能として互いに協調して働き、ネットワークを制御の定義域に分割することができる。この場合、各 SSCP は、それぞれの担当の定義域内の物理装置および論理装置に対して階層的な制御関係を持つことになる。

システム・サービス制御点 (SSCP) 定義域 (system services control point (SSCP) domain). システム・サービス制御点、物理装置 (PU)、論理装置 (LU)、リンク、リンク・ステーション、および 活動化要求と非活動化要求によって SSCP が制御できるすべての資源。

システム・ネットワーク体系 (Systems Network Architecture (SNA)). ネットワークを介して情報単位を伝送し、またネットワークの構成および動作を制御するための、論理構造、プロトコル、および操作順序のこと。SNA の構造は階層化されているので、情報の末端の起点および宛先（つまりユーザー）は、情報交換に使用する特定の SNA ネットワーク・サービスおよび機能に依存することも、またその影響を受けることもない。

システムの始動 (system startup). *初期プログラム・ロード (initial program load (IPL))* の同義語。

T

テーブル (table). NetDA/2 がネットワークの設計のために使用するデータのリポジトリー。各テーブルにはネットワークに関連した情報が入っている。

タスク (task). マルチプログラミングまたは多重処理環境における命令のシーケンスであって、制御プログラムが、コンピューターによって実行されるべき 1 つの作業要素として取り扱うもの。 (I) (A)

TCP. 伝送制御プロトコル（Transmission Control Protocol)。

TCP/IP. 伝送制御プロトコル/インターネット・プロトコル (Transmission Control Protocol/Internet Protocol)。

TERMINATE. SNA において、指定された LU-LU セッションを終わらせる手順をシステム・サービス制御点 (SSCP) に開始させるために、論理装置 (LU) によってその SSCP へ送られる要求単位。

TG. 伝送グループ (transmission group)。

TH. 伝送ヘッダー (transmission header)。

スレッド (thread). OS/2 オペレーティング・システムにおいて、1 プロセス内で実行される操作の最小単位。

しきい値 (threshold). (1) NetView プログラムにおいて、資源に設定され、計算した対通信量エラー率と比較される割合値。(2) NetView for AIX において、観測値が限界に達したという通知が出されるまでの最大値を指定する設定。たとえば、監視される MIB 値がしきい値を超えると、データ収集装置はしきい値事象を生成する。(3) NPM において、収集されるデータおよび統計を監視するためにユーザーが指定する最大値および最小値。(4) IBM ブリッジ・プログラムにおいて、それ以上ブリッジを通過するとエラーが生じるものとして、フレームの最大数として設定された値。これを超過すると、「しきい値の超過」の発生がカウントされネットワーク管理プログラムにそのことが示される。(5) カウンターがその値から 0 まで 1 ずつ減らされる初期値。または、カウンターが初期値からその値まで増加または減少される値。

timeout. (1) 指定されたイベントの開始時に始まった時間間隔の終わりに発生する他のイベント。 (I)(2) 特定の操作が行われるために割り当てられた時間間隔。たとえば、ポーリングまたはアドレス指定が行われてから、この時間内に応答を行わないと、システム操作が中断されて再始動が必要になる場合など。

タイム・スタンプ (time stamp). (1) 現在のシステム時刻を適用すること。(2) オブジェクトの活動記録における重要なポイントのシステム時刻を示す、オブジェクトに関する値。(3) 照会において、照会報告書が作成された日付と時刻を示すものであり、照会によって各報告書に自動的に提供される。

トークン (token). (1) ローカル・エリア・ネットワークにおいて、伝送媒体を一時的に制御しているステーションを示すために、１つのデータ・ステーションから他のデータ・ステーションへ次々に渡される、権限を示す記号。 どのデータ・ステーションにも、トークンを獲得し、それを使用して媒体を制御する機会がある。 トークンは、伝送の許可を意味する特別なメッセージつまりビット・パターンである。 (T) (2) LAN において、伝送媒体を介して１つの装置から別の装置に渡されるビット順序。 トークンにデータが付加されている場合は、全体として１つのフレームとなる。

トークンリング (token ring). (1) IEEE 802.5 によると、媒体接続ステーション間でのトークン（特殊パケットまたはフレーム）の受渡しにより媒体アクセスを制御するネットワーク・テクノロジー。(2) 接続しているリング・ステーション（ノード）から別のノードへトークンを渡すリング・テクノロジーを使用する FDDI または IEEE 802.5 ネットワーク。(3) *ローカル・エリア・ネットワーク (local area network (LAN))* も参照。

トポロジー (topology). 通信における、ネットワーク内のノードの物理的または論理的な配置。特に、ノード相互の関係や、ノード間のリンクの関係。

トポロジー・データベース (topology database). *ローカル・トポロジー・データベース (local topology database)* および *ネットワーク・トポロジー・データベース (network topology database)* を参照。

TP. トランザクション・プログラム (Transaction Program)。

トランザクション・プログラム (transaction program (TP)). SNA ネットワークでトランザクションを処理するプログラム。 次の 2 種類のトランザクション・プログラムがある。アプリケーション・トランザクション・プログラムおよびサービス・トランザクション・プログラム。*会話（conversation）* も参照。

伝送制御プロトコル (Transmission Control Protocol (TCP)). インターネット、およびネットワーク間プロトコルについての米国国防省標準に従ったネットワークで使用される通信プロトコル。TCP は、パケット交換通信ネットワークおよびそのようなネットワークを相互接続したシステムで、信頼性の高いホスト間プロトコルを提供する。 基礎プロトコルとしてインターネット・プロトコル (IP) を使用する。

伝送制御プロトコル/インターネット・プロトコル (Transmission Control Protocol/Internet Protocol (TCP/IP)). ローカル・エリアネットワークと広域ネットワークの両方で対等接続機能をサポートする、通信プロトコルのセット。

伝送グループ (transmission group (TG)). (1) 伝送グループ番号で識別される、隣接ノード間の接続。(2) サブエリア・ネットワークにおいて、隣接ノード間の単一リンクまたはリンクのグループ。伝送グループがリンクのグループから構成される場合には、これらのリンクを 1 つの論理リンクと見ることができる。この伝送グループのことを、*多重リンク伝送グループ (multilink transmissiongroup (MLTG))* と呼ぶ。*混合媒体多重リンク伝送グループ (mixed-media multilinktransmission group (MMMLTG))* は、さまざまな媒体タイプのリンクを含んでいる。(たとえば、トークンリング、切替え SDLC、非切替え SDLC、およびフレーム・リレー・リンク)。(3) APPN ネットワークにおいて、隣接ノード間の単一リンク。(4) *並列伝送グループ (parallel transmission groups)* も参照。

伝送グループ (TG) プロファイル (transmission group (TG) profile). VTAM において、APPN リンクに使用される、名前付きの特性のセット (コスト/バイト、コスト/時間単位、および容量など)。

伝送グループ (TG) ベクトル (transmission group (TG) vector). T2.1 ネットワークの端点 TG の表示。2 つの制御ベクトル (TG 記述子 (X'46') 制御ベクトルおよび TG 特性 (X'47') 制御ベクトル) から構成される。

伝送ヘッダー (transmission header (TH)). 任意に基本情報単位 (BIU) または BIU セグメントが後に続く制御情報。メッセージ単位を経路指定し、そのネットワーク内でのフローを制御するために、パス制御によって作成され、使用される。 *パス情報単位 (path information unit)* も参照。

伝送の優先順位 (transmission priority). 各ノードのパス制御構成要素により選択される優先順位を、その経路内で次のノードへ進むための経路に沿って判別するために、メッセージ単位に割り当てられるランク。

トランスポート・プロトコル (transport protocol). トランスポート・ネットワークの構成要素間の情報の交換を管理する規則の仕様。

トラップ (trap). 簡易ネットワーク管理プロトコル (SNMP) において、管理ノード (エージェント機能) が、例外条件を報告するために管理ステーションへ送るメッセージ。

チュートリアル (tutorial). 学習を指導する形で示される情報。

T1. 米国における、1.544 Mbps の公衆アクセス回線。24 の 64 Kbps のチャネルで利用できる。ヨーロッパ・バージョン (E1) の伝送速度は 2.048 Mbps。日本バージョン (J1) の伝送速度は 1.544 Mbps。

U

UNBIND. SNA において、2 つの論理装置 (LU) の間のセッションを非活動化するための要求。*セッション非活動化要求 (session deactivation request)* も参照。*BIND* と対比。

非解釈名 (uninterpreted name). SNA において、システム・サービス制御点 (SSCP) が論理装置 (LU) のネットワーク名に変換することのできる文字ストリング。通常、非解釈名は、ログオンに使用されるか、または2 次論理装置 (SLU) から要求を開始してセッションの要求先である 1 次論理装置 (PLU) を識別するために使用される。

アップストリーム (upstream). ユーザーからホストへ流れるデータ・フローの方向。 *ダウンストリーム (downstream)* と対比。

アップタイム (uptime). (1) *動作可能時間 (operable time)* の同義語。 (T)(2) *使用可能時間 (available time)* の不適語。(3) *動作時間 (operating time)* の同義語。

ユーザー (user). (1) 情報処理システムとの間でコマンドおよびメッセージを送受信する人または物。 (T) (2) 計算機システムのサービスを必要とする人。

ユーザー ID (user identifier (UID)). ネットワークまたはシステムのユーザーを個別に識別する名前。

UTC. 世界協定時 (coordinated universal time)。

V

値 (value). (1) 属性の特定の発現。たとえば、属性「色」について「青」と表示されること。 (T) (2) 定数、変数、パラメーター、または記号に割り当てられた量。

変数 (variable). (1) プログラミング言語において、異なった値を一度に 1 つだけ取ることができる言語オブジェクト。通常は、変数の値は特定のデータ・タイプに限定されている。 (I) (2) 複数の値より成る所与の集合から、任意の値を取ることができる量。 (A) (3) プログラムが実行されている間にデータ項目の値が変化する場合に、データ項目を表すために使用される名前。

ベクトル (vector). MAC フレーム情報フィールド。

verb. *LU 6.2 verb* を参照。

バージョン (version). 別個にライセンスされるプログラムであって、通常、著しく新規なコードまたは新しい機能を含んでいるもの。

仮想回線 (virtual circuit). (1) パケット交換において、ネットワークによって提供され、ユーザーに対して現実の接続であるかのような外観を呈する機構または機能。 (T) *データ回線 (data circuit)*も参照。 *物理回線 (physical circuit)* と対比。(2) 2 つの DTE の間で確立される論理接続。

仮想計算機 (virtual machine (VM)). (1) 特定ユーザーの裁量で排他的に利用できるように見えるが、実際のデータ処理システムの資源を共用することによって機能が実現される、仮想データ処理システム。 (T) (2) VM/ESA においては、1 人のユーザーに割り当てられた仮想プロセッサー、仮想記憶域、仮想装置、および仮想チャネル・サブシステム。仮想計算機には、専用の拡張記憶も含まれる。

VM/ エンタープライズ・システム体系 (Virtual Machine/Enterprise Systems Architecture (VM/ESA)). 単一コンピューターの資源を管理して、複数の計算システムが存在しているような外観を与える IBM ライセンス・プログラム。 各仮想計算機は実計算機と機能的に同等である。

仮想ノード (virtual node). *仮想経路ノード (virtual routing node)*。

仮想経路 (virtual route (VR)). (1) SNA において、次のような論理接続。(a) 特定の論理経路として物理的に実現されている、2 つのサブエリアノード間の論理接続。(b) ノード内のセッションのために、サブエリア・ノードに完全に収まっている論理接続。異なるサブエリア・ノード間の仮想経路は、基礎となっている明示経路に何らかの伝送優先順位を課し、仮想経路ペーシングを使用してフロー制御を行い、パス情報単位 (PIU) の順序番号付けによりデータ保全性を確保する。(2) *明示経路 (ER)* と対比。*パス (path)* および*経路拡張機能 (route extension (REX))* も参照。

仮想経路 (VR) ペーシング (virtual route (VR) pacing). SNA において、パス情報単位 (PIU) が仮想経路上を流れる速度を制御するため、仮想経路の両端でパス制御の仮想経路制御構成要素によって使用される、フロー制御の手法。 VR ペーシングは、経路上の任意のノードでのトラフィック輻輳（ふくそう）の程度に応じて調整できる。 *セッション・レベル・ペーシング (session-level pacing)* も参照。

仮想経路ノード (virtual routing node). ノードから、共用アクセス転送機能 (SATF) で定義された接続ネットワーク（トークンリングなど）への、接続性の表現。*仮想ノード (virtual node)* と同義。

仮想記憶通信アクセス方式 (Virtual Telecommunications Access Method (VTAM)). SNA ネットワークで通信とデータの流れを制御する IBM ライセンス・プログラム。単一定義域、複数定義域、および相互接続ネットワーク機能を提供する。

VM. 仮想計算機 (virtual machine).

VM/ESA. VM/ エンタープライズ・システム体系 (Virtual Machine/Enterprise Systems Architecture)。

VR. 仮想経路 (virtual route)。

VTAM. (1) 仮想記憶通信アクセス方式 (Virtual Telecommunications Access Method)。(2) *ACF/VTAM* と同義語。

W

WAN. 広域ネットワーク (wide area network)。

重み (weight). 経路の選択を目的とする場合に、特定のサービス・クラスによって指定される基準を資源 (ノードや伝送グループなど) がどの程度満たすことができるかを示す。 APPN 経路選択では、重みが最小の経路が選択される。

広域ネットワーク (wide area network (WAN)). (1) ローカル・エリア・ネットワークまたは大都市圏ネットワークの対象範囲より広い地域を対象として通信を提供し、公共通信施設を利用または提供できるネットワーク。 (T) (2) 数百または数千マイルの区域にサービスするように設計されたデータ通信ネットワーク。たとえば、公衆および私用のパケット交換ネットワーク、および国内電話網など。(3) *ローカル・エリア・ネットワーク (LAN)* および *大都市圏ネットワーク (metropolitan area network (MAN))* と対比。

ウィンドウ (window). (1) 表示画面の一部分であり、特定のアプリケーションに関する表示像を表示できる画面部分。 いくつかの異なるアプリケーションを、それぞれ異なるウィンドウに同時に表示できる。 (A) (2) 目に見える境界がある区域であり、そこにオブジェクトのビューが表示されるか、それを用いてユーザーがコンピューター・システムと対話を行うもの。(3) データ通信におけるデータ・パケットの数であり、データ端末装置 (DTE) またはデータ回線終端装置 (DCE) は、その数だけのデータ・パケットを、他のデータ・パケットを送る許可を待つ前に、論理チャネルを通して送ることができる。 ウィンドウは、パケットのペーシング、つまりフロー制御の主要メカニズムである。(4) ペーシング・ウィンドウ *(pacing window)* を参照。

ウィンドウ・サイズ (window size). 肯定応答を受け取る前に送信できる、特定の情報フレーム数。

WinSock アプリケーション・プログラミング・インターフェース (WinSock application programming interface (API)). Windows ファミリーのオペレーティング・システム用に開発されたソケット・スタイルのトランスポート・インターフェース。

ワークステーション (workstation). (1) ユーザーが作業を行う機能装置。 ワークステーションは何らかの処理機能を持つものが多い。 (T) (2) ユーザーに作業を行わせるプログラム式または非プログラム式装置。(3) 端末またはマイクロコンピューターであって、通常、メインフレームまたはネットワークへ接続され、ユーザーがそこでアプリケーションを実行できるもの。

X

XID. 交換識別 (exchange identification)。

XMIT. 伝送 (transmit)。

Z

Z 時 (Z time). *ズールー時 (Zulu time)* の略称。*世界協定時 (coordinated universal time (UTC))* の同義語。

ズールー時 (Zulu time (Z)). *世界協定時 (coordinated universal time (UTC))* の同義語。

索引

日本語，英字，数字，特殊文字の順に配列されています。なお，濁音と半濁音は清音と同等に扱われています。

［ア行］

［カ行］

［サ行］

［ナ行］

［ハ行］

［ヤ行］

［ラ行］

A

C

D

R

S

T

V

W

X

Printed in Japan

SC88-7728-00