OSIトランスポート･プロトコルにおけるクラス4とNCMSの実装について

加藤聰彦　小花貞夫　鈴木健二
国際電信電話株式会社 研究所

1. はじめに

CCITTやISOでは異機種計算機･端末相互間の通信を実現するため、開放型システム間相互接続(OSI)の標準化を進めており、特にトランスポート･レーヤ(TL)とセション･レーヤ(SL)についてはそのサービス定義とプロトコル仕様が確定し、両機関とも勧告化並びに標準化している[1,2]。またISOではTLがネットワーク･コネクション(NC)を有効に使用するためのサブプロトコル(NCMS)の検討も行っており、現在Draft Proposalとなっている[3]。

筆者等は、公衆パケット網においてMHSやOSIゲートウェイ機能等の高度通信処理サービスを提供する網内ホストの構築過程で、これまでにOSIトランスポート･プロトコル(TP)のクラス0から3、OSIセション･プロトコル(SP)の全仕様及び高位レーヤの簡易的なシミュレーション･プログラムをVAX11/780上に作成している[4,5]。本稿では、TPの全仕様の実装に向けて、新たにTPのクラス4及びNCMSを追加作成したので、その概要を報告する。

2. TPクラス4の概要と実装

2.1 TPクラス4の概要

TLは、SLに対して、その要求するサービス品質を満足するデータ転送機能を提供するもので、0から4までの5種類のプロトコル･クラスが規定されている。このうちクラス4は、誤り検出回復クラスと呼ばれ、

(i) 想定するNCが充分な伝送品質を持たないために生ずるTPDU (Transport Protocol Data Unit)の消失･重複･順序誤りの検出/回復機能や、NCの障害通知からの回復機能

(ii) 複数のNCを用いてTPDUを転送する機能

等に特徴がある。このため、他のクラスと比較して、表1に示すような独自のプロトコル機構が使用されており、重装備のプロトコルである。

具体的には、(i)の機能として、伝送誤りを検出するためにチェックサム、伝送誤りによるTPDUの消失からの回復にはタイムアウトによる再送、TPDUの重複･順序誤りからの回復にはリシーケンシングの各プロトコル機構が設けられている。このようなTL内の誤り回復機能と一本のTCを複数のNCに割付けるスプリッティング/リコンバイニングにより、ネットワーク･レーヤ(NL)からNCのリセット･切断指示の形で与えられるNCの障害通知から回復できる。更にNLから通知されないNCの障害から回復するために、一定時間TPDUを受信しない場合にそのトランスポート･コネクション(TC)を解放するインアクティビティ制御も有している。一方、(ii)の機能としては、スプリッティング/リコンバイニング、順序制御を行うためにリシーケンシングの各プロトコル機構が用いられる。

表1 TPのプロトコル機構

プロトコル機構	種別	0	1	2	3	4
NCへの割付け		*	*	*	*	*
TPDU転送		*	*	*	*	*
セグメンティング/アセンブリング		*	*	*	*	*
コンカティネーション/セパレーション			*	*	*	*
コネクション確立		*	*	*	*	*
コネクション拒否		*	*	*	*	*
普通解放	暗示	*				
	明示		*	*	*	*
エラー解放		*		*		
TPDUのTCへの関連付け		*	*	*	*	*
DT TPDU番号付け	普通		*	m	m	m
	拡張			o	o	o
優先データ転送	NLの普通データ		m	*	*	*
	NLの優先データ		ao			
障害後の再割付け			*		*	
TPDUの保留	NLの送達確認		ao			
	確認応答TPDU		m		*	*
再同期			*		*	
マルチプレクシング/デマルチプレクシング				*	*	*
フロー制御	有り			m	*	*
	無し	*	*	o		
チェックサム	有り					*
	無し	*	*	*	*	o
リファレンス凍結			*		*	*
タイムアウトによる再送						*
リシーケンシング						*
インアクティビティ制御						*
プロトコル･エラーの処理		*	*	*	*	*
スプリッティング/リコンバイニング						*

*：その手順を含む、 m：ネゴシエート可(実装は必須)、 o：ネゴシエート可(実装は任意)、 ao：ネゴシエート可(実装は任意、NLに依存)

クラス4のシーケンス例を図1に示す。この図に示すように、TCはCR, CC, AK(又はDT等)の3つのTPDU交換により確立され、TC確立後に必要に応じてスプリッティングに使用されるNCが確立される。またCR, CC, DT, ED, DRの各TPDUが送信されると再送用タイマT1が起動され、T1タイムアウトにより再送される。DT TPDUの応答と転送ウィンドウの伝達に用いられるAK TPDUは、AK用再送タイマWにより再送され、その転送順序を示すサブシーケンス番号によりAK TPDUの順序制御が行われる。また同一のAK TPDUを受信した場合などには、FCC (Flow Control Confirmation)パラメータを持つAK TPDUを用いて、重複したAK TPDUの受信確認を行う。更に、転送ウィンドウを開いたり逆に閉じるために用いるAK TPDUでは、これらの情報が確実に伝達されるように、Wの代わりにT1タイマを用いて再送する場合もある。

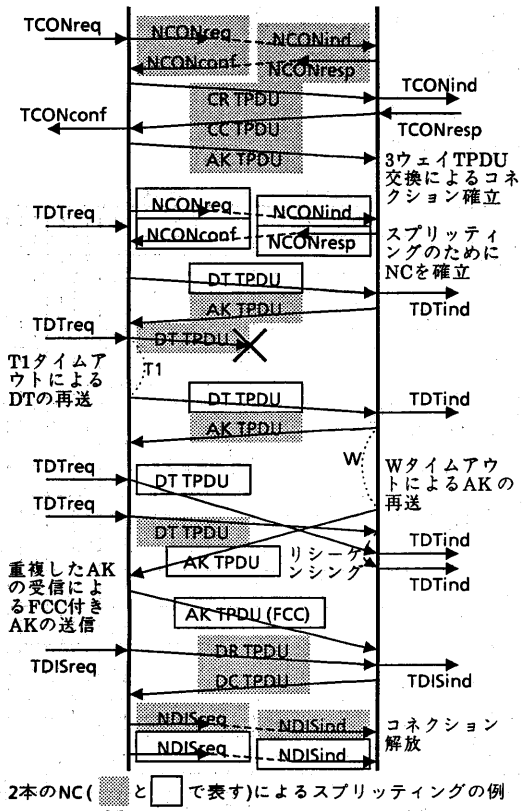


図1 TPクラス4のシーケンス例

2.2 TPクラス4実装の方針

先にVAX11/780上に実装したTPクラス0から3のプログラムは、

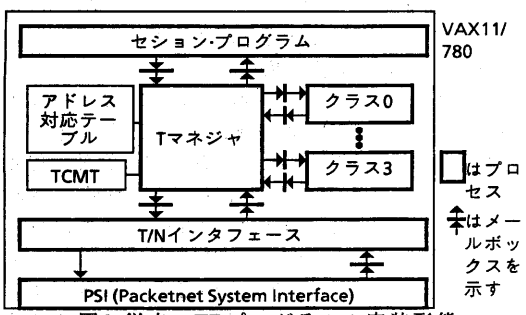


図2 従来のTPプログラムの実装形態

(i) Tマネジャ：①TCMT (TC Management Table) を用いて複数のTCの管理、②TSAPアドレスとNSAPアドレスの変換、③クラスの決定、④TCを区別するための自局のリファレンスやTC/NCのシステム内の識別子であるTS CID (Transport Service Connection Identifier) / NS CID (Network Service CID)の選択等のTLにおける一元的な情報の管理、⑤セパレーション/TPDUのTCへの関連付け/デマルチプレクシング等のプロトコル機構を実行

(ii) クラス・プロセス：各クラスのTCを管理し、クラスごとのパラメータのネゴシエーション、状態管理、TPDUやプリミティブの送出を含む、Tマネジャで行う以外のプロトコル機構等を実行

のプロセスとして実現され、これらのプロセスはVAX/VMSの提供するメールボックスにより接続されている(図2参照)[4]。

そこでクラス4を実装するに当たり、

(1) クラス4も単独のプロセスとしてTマネジャにインタフェースするように実装する。

(2) クラス4を含めた全てのクラスをサポートするように、Tマネジャを変更する。その際先に作成済みのクラス0から3のプログラムの変更については必要最小限のみを行う。

という基本設計方針をたてた。

2.3 TPクラス4実装の特徴

TPの実装では、関係の深いプロトコル機構を如何に実装するか、Tマネジャとクラス・プロセスに如何に機能分担するか、クラス・ネゴシエーション・ダウン等の複数のクラスにまたがった処理を行うTマネジャを如何に実装するか等が問題となる[4]。

以下に、クラス4を追加実装するためのTマネジャの変更、クラス4固有のプロトコル機構の実装方法等を述べる。

2.3.1 Tマネジャにおける TCとNCの管理

TPはマルチプレクシングや障害後の再割付けなどのプロトコル機構を有するためにTCとNCの対応付けが複雑で、先に作成したTPプログラムではTマネジャの処理において、NCの割付け、再割付け、コネクション確立、マルチプレクシング等のプロトコル機構が複雑に関連を持っている[4]。クラス4を追加作成するためには、これに加えてスプリッティングを実現する必要があり、TマネジャにおけるTCとNCの管理機能を見直す必要がある。

従来のTPプログラムではTCとNCを、TCMTという単一のテーブルで同時に管理していた[4]。しかしながら、スプリッティングでは1つのTCに複数のNCが割付けられるので、スプリッティングに使用されているNCに関する情報を保持するためには、NCを管理するテーブルをTCを管理するテーブルとは別個に用意する必要がある。新しいTPプログラムでは次のような3種類のテーブルを用意した。

①TCMT：TCを管理するために使用するテーブル。クラス4を追加するために図3に示すように、従来のものに、そのTCでスプリッティングに使用するNCの最大数を示す欄を追加している。TPDUのTCへの関連付けや障害後の再割付の処理を効率的に行うために自局のリファレンス順に並べられている。

②NSCID MT(NS CID Management Table):TCとNCの割当て状況を管理するためテーブル。NS CID、NCの状態、割り付けられたTCのクラスと自局のリファレンス等の情報が含まれる(図3参照)。受信したNプリミティブとNSCID MTを効率的に関係付けるためにNS CID順に並べられている。

自局のリファレンス	(2)
相手局のリファレンス	(2)
自局のTSAPアドレス	(10)
相手局のTSAPアドレス	(10)
自局のNSAPアドレス	(14)
相手局のNSAPアドレス	(14)
TS CID	(2)
NS CID	(2)
TCの状態	(2)
NCの状態	(2)
クラス 代替クラス・フラグ (class0) 代替クラス・フラグ (class1) 代替クラス・フラグ (class2) 代替クラス・フラグ (class3)	[4] [1] [1] [1] [1]
スプリッティングの最大数(4)	

TCMT

NS CID	(2)
NCの状態	(2)
割付けられたTCのクラス	(1)
割付けられたTCの数(4)	
相手局のNSAPアドレス	(14)
割付けられたTCの自局のリファレンス	(2)
⋮	
割付けられたTCの自局のリファレンス	(2)

NSCID MT

( )及び[ ]内は各々バイト数、ビット数を示す

図3 TCMTとNSCID MTの構成

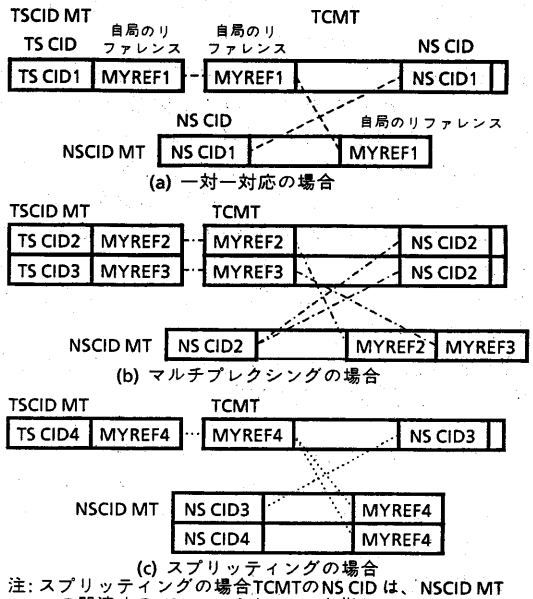
③TSCID MT(TS CID Management Table):受信したTプリミティブのTS CIDから対応するTCMTを効率的に選択するためのテーブル。TS CIDと自局のリファレンスの対が規定される。

これらの3つのテーブルは自局のリファレンスとNS CIDによりリンクされ効率的に選択される。マルチプレクシング、スプリッティングの場合を含むテーブルのリンクの状態を図4に示す。また、クラス・プロセスとの情報のやりとりは、クラス0から3に対する変更を少なくするために、従来通りTCMTを用いて行う。そのためTCMTとNSCID MTの間に、自局のNSAPアドレスとNCの状態の情報が重複している。

これらのテーブルは、次の様に確保され使用される。イニシエータ側ではSLからのコネクション確立要求(TCONreq)に対して、空きのTCMTを確保し、自局のリファレンス、クラス、使用するNCを決定する。NCを新たに確立する場合は空きのNS CIDを選択しNSCID MTを確保する。このあとテーブルに各情報をセットする。

レスポンダ側ではネットワーク・レーヤからのコネクション確立指示(NCONind)により、NSCID MTを確保する。次にそのNCから受信される最初のCR TPDUによりTCMTを確保し自局のリファレンス、クラスを決定し、テーブルに各情報をセットする。

テーブルが確保された後は、Tプリミティブ受信時はTプリミティブのTS CIDからTSCID MTにより、Nプリミティブ受信時はそのNS CIDまたは



注: スプリッティングの場合TCMTのNS CIDは、NSCID MTの関連するNS CIDのうちの1つを指している。

図4 TSCID MT, TCMT, NSCID MTのリンクの状態

TPDUに含まれる自局のリファレンスからNSCID MTにより、TCMTを選択する。

2.3.2 スプリッティング/リコンバイニング

スプリッティングを行うためにNCを複数確立する契機は、最初のNC上でTCが確立し必要なQOSが設定された直後の他に、NCの切断が通知された場合、再送のためのタイムアウトが生じた場合(T1タイムアウト)などである。このようにNCを確立する契機はクラス4の状態管理と関係が深いために、クラス4プロセスで管理する必要がある。一方、NCはTマネジャが一元的に管理しているため、NC確立に必要なNS CIDはTマネジャが割当てクラス4プロセスに通知する必要がある。通知されたNS CIDはクラス4プロセスで保持される。

そこでTマネジャ/クラス4プロセス間にNS CID要求、NS CID割当ての2つの制御情報を新たに用意した。クラス4プロセスはNCを確立する必要が生ずるとTマネジャにNS CID要求を送る。Tマネジャはそれに対し空きNS CIDを選択しNS CID割当てによりクラス4プロセスに伝える。クラス4プロセスはこのNS CIDを用いてコネクション確立要求(NCONreq)を発行する。

リコンバイニングは、TマネジャにおいてTPDUのTCへの関連付けの処理の中で実行される。

2.3.3 チェックサム

送信するTPDUへのチェックサムの設定はTPDUを作成する時点に行うため、クラス4プロセスで実行するのが適当である。しかし受信したTPDUのチェックサムの検査は、そのTPDUを

TCに関連付けるまえに行う必要があり、Tマネジャで実行される。特に本システムのように、クラス4を含む複数の異なったクラスのTCが存在するシステムでは、本来クラス4のTCに関連付けるべきTPDUが、伝送誤りにより偶然自局のリファレンスの部分に誤りが生じ、これがもとで他のクラスのTCに関連付けられないよう処理する必要がある。このため、NCが確立された直後に受信したTPDUに対しては、クラスに係わりなく、チェックサムパラメータの有無を調べ、有った場合はその検査を行う必要がある。

2.3.4 タイムアウトによる再送、リシーケンシング、インアクティビティ制御

スプリッティング/リコンバイニング、チェックサム以外のクラス4固有のプロトコル機構については、クラス0から3とは関係がないため、他のプロトコル機構とは独立にクラス4プロセスで実行するのが適当である。

2.3.5 再送等に用いられるタイマ

本プログラムでは、TPクラス4で定義しているタイマのうち、以下のものを実装している。

- 再送用タイムT1
  タイマは個々のTPDU毎には用意せず、TCに1つ用意する。但しデータ転送フェーズにおいてはDT、ED、AK用の3つが用意される。
- AKの再送用タイマ W
- インアクティビティタイマ I
- リファレンスの凍結用タイマ L

プログラムは、これらのタイマ値を自由に設定できるように作成されているが、現在T1、W、Iの3つのタイマにはT1<W<Iの関係を持たせ、それぞれ60秒、180秒、1080秒の値を採用している。またLの値は120秒を用いている。

3. NCMSの概要と実装

3.1 NCMSの概要と拡張

ISOで検討しているNCMS(Network Connection Management Subprotocol)はTLがNCを有効利用することを目的としており、TPのオプションとして位置付けられる。その主な機能は

① OSI、非OSIを含め、NLの上位のプロトコルが複数想定された場合、これらを識別し、またNC上に共存を可能とする機能
② NCの再利用や多重化を有効に行うために、NCのレスポンダ側からもTCを割付けることができるようにするNC管理機能
③ TLがNCを解放する理由を通知する機能

である。これらの機能を実現するために、①と②に対してはそれぞれ、NC確立時のNCONreq/NCONindプリミティブのユーザデータとして運ぶUN (Use of NC) TPDUとNCM (Network Connection Management) TPDUを用い、また③に対しては、NC解放時のNDISreq/NDISindのユーザデータとして運ぶDIAG (Diagnostic) TPDUを用いる。これらのTPDUの使用は全てオプションであるが、NCM TPDUを使用する場合は必ずUN TPDUと連結して送られる。

②のNC管理機能はTPクラス2、3、4に適用され、

- TCをNCに割付ける権利(割付権利：Assignment right)をどちらのトランスポート・エンティティ(TE)に持たせるかを指定する機能、
- 各NCにNCリファレンスを割当て、NCリファレンスとその割当て元を用いて、NCを識別する機能、
- NCが切断された場合、NCONreqの衝突を解決し、NCを再度確立する機能

を有している。割付権利の指定には、NCのイニシエータのみに割付権利を持たせるもの(SA)、レスポンダのみに持たせるもの(RA)、双方のTEに持たせるもの(AA)が定められている。図4にNCMSを使用した場合のTCとNCの関係を示す。

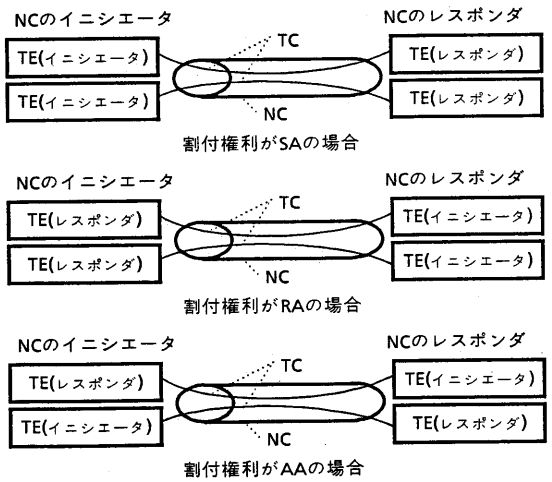


図4 NCMSを使用した場合のTCとNCの関係

但し、NCMSはオプションであるため、受信したNCM TPDUを無視してもよく、その場合はそのNCに対してNCMSの機能を使用しない。

図5に割付権利がAAの場合のシーケンス例を示す。ここではNC切断時の再度確立により、NCが切断されると双方のTEがNCM TPDUをもつNCON reqを発行している。そのNCM TPDUを受信した双方のTEは、NCM TPDUのNCリファレンスとその割当て元の情報から、コネクションの確立の衝突が生じたことを検出し、一方の確立の要求を選択する。どちらの確立の要求を選ぶかは双方のNCM

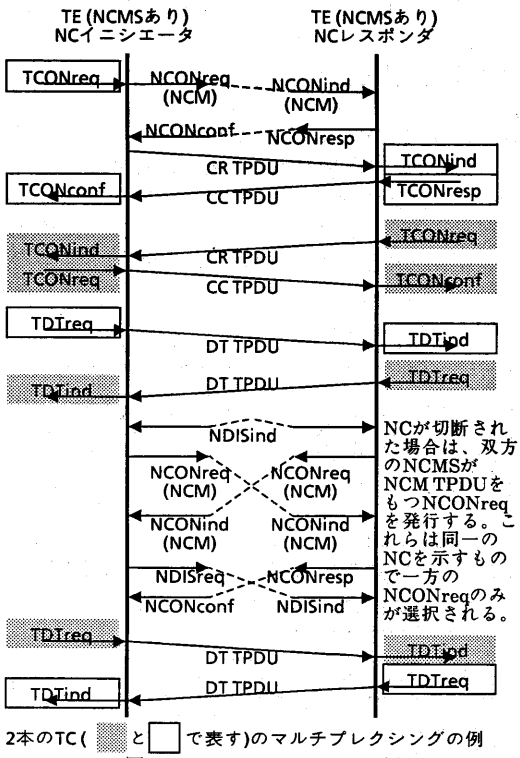


図5 NCMSのシーケンス例

TPDUの中のNCプリファレンス・パラメータの値及び双方のNSAPアドレスの比較による。

筆者らはNCMSに対して次のような問題点が存在することを先に指摘している[7]。

(i) NCM TPDUに応答が無いため、割付権利がRAのNCM TPDUを受信したTEがそれを無視した場合はTCを割付けられないNCが存在してしまう。

(ii) 通信を行うTEの間で、NCに多重化できるTCの数やTCが割付けられていないNCの寿命等のネゴシエーションが実現できない。

そこで、次のような拡張を提案してきた[7]。

- NCM TPDUの応答としてNCMC (Network Connection Management Confirm) TPDUを新たに導入し、NCM TPDUを処理したTEはNCMC TPDUをNCONresp/confのユーザデータとして転送する。
- NCM TPDUとNCMC TPDUに(ii)の情報を示すフィールドを設ける。

これらの内で(i)の問題点を解決するために、NCMC TPDUを導入することが、ISO日本案として採用された。さらにこれがISO TC97/SC6/WG4にも反映され、割付権利がRAのNCM TPDUに対しては、その応答としてNCMC TPDUを転送すると定められた。

3.2 NCMS実装の方針

NCMSをTPプログラムに追加作成する際の基本設計方針は以下の通りである。

(1) NCMSはクラス2、3、4で共通的に用いられ、またTCを割付けるNCの選択というTマネジャの機能と関係が深いために、Tマネジャの中に作成する。

(2) NCMSの追加作成による各クラスのプログラムの変更を最小限にし、またTマネジャとクラス間の内部のやりとりが増加しないようにする。

(3) NCMSのNC管理機能は、各NCで使用される場合も使用されない場合もある。クラス・プロセスはいずれの場合も同じ動作をするものとする。

(4) NCMSではクラス2が割付けられているNCが切断された場合NCの再度確立を行うことを認めているが、ここではそれは行わない。

(5) DIAG TPDUは使用しない。

3.3 NCMS実装の特徴

次に、NCMSをTマネジャに実装する方法、Tマネジャとクラス・プロセスの間のやりとり、及びクラス・プログラムの変更等について示す。

3.3.1 Tマネジャの変更箇所

Tマネジャにおいて、NCMSを使用したNCを管理するために、NC管理用のテーブルのNSCID MTを拡張して、NCMSに関連する情報を保持することにする。この様な情報には、NCリファレンスの値、NCリファレンスの状態、NCリファレンスの割当て元、送信するNCM TPDUのNCの割付権利、自局または相手局のTEがそのNCにTCを割当てられるかを示すフラグなどがある。但し、そのNCがNCMSを使用しているかどうかは、NCリファレンスの値が割当てられているかどうかで区別する。

またNCMSの機能は、Tマネジャにおいて

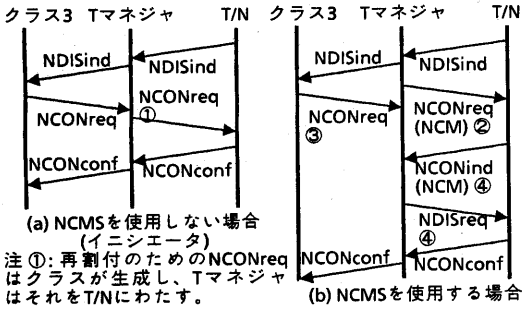
- Tプリミティブ受信処理の内、TCONreqに対して使用するNCを選択する処理
- 受信したNCONindの処理(受信したUN, NCM TPDUを解析し、コネクション確立の衝突の解決する機能)
- 受信したNDISindの処理(NDISindに対するフェーズ管理、NCを再度確立する機能)
- クラス・プロセスからのNCONreq送信要求に対する処理(UN, NCM TPDU送信処理)

等の中に実現される。またTマネジャはNCMSで使用される3種類のタイマ(TTR-NC:NCの再度確立を行う期間を示す、TPD-NC:相手からのNCの再度確立を待つ期間を示す、TFR-NC:NCリファレンスの凍結期間を示す)を実装しそれぞれのタイムアウト処理を行う。

3.3.2 Tマネジャとクラス･プロセスのインタフェース及びクラス･プログラムの変更

TPクラス3、4では各々障害後の再割付、スプリッティング/リコンバイニングのプロトコル機構によりNDISindに対してNCを再度割付ける機能を有している。一方NCMSはこれとは独立にNCを再度確立する機能を有している。これらを協調させるためには、NDISindに対してNCの再度確立はNCMSが行い、TPクラス3、4は、自らはNCの確立は行わずTCを割付けることのできるNCを見出しそれを利用するようにすればよい。

本TPプログラムではクラス3、4のプロセスは、NDISindを受信するとNCONreqをTマネジャに送出しNCONconfを待つように作成されている。そこで、NCMSを使用する場合と使用しない場合とでTマネジャとクラス･プロセスの間のインタフェースを同一にするために、図6に示すように、NCMSが使用された場合は、Tマネジャはクラス･プロセスからのNCONreqを無視して独自にNCを確立し、NCが確立されたらNCONconfとしてクラス･プロセスに通知するようにした。



注①: 再割付のためのNCONreqはクラスが生成し、Tマネジャはそれを T/Nにわたす。
②: NDISindに対して、TマネジャがNCONreqを生成し、発行する。
③: クラス3からのNCONreqは、クラスとTマネジャの間のやりとりとして用いられる。
④: 相手TEも割付権利を有する場合はNCONindを受信することがある。このNCONreqの衝突はNCMSで解決されクラスには通知されない。

図6 クラス3の再割付の手順

但し、NCMSでNC確立の衝突が解決された場合は、Tマネジャとクラス･プロセスの間でNCONreqと対応するNCONconfでNS CIDが異なることがあるため、クラス･プログラムを各NCのNS CIDとしてNCONconfの示すNS CIDを使用するように変更した。

4. 結果と考察

今回クラス4及びNCMSを追加作成したTPプログラムは、先に実装したセションプログラム、BSSシミュレーションプログラム等の上位プログラムを用い、X.25実験網上で良好に動作している。クラス4、NCMSを含むシステム全体のプログラム構成を図7に示す。以下に得られた結果について考察する。

4.1 実装上の問題点

①表2、3にTマネジャ、クラス4のソースプログラムをC言語で記述した際の実効行数を示す。但しTマネジャについてはNCMSを実装した場合としない場合の実効行数を示している。NCMS無しの場合のTマネジャが約3.1K行、クラス4が約9.4K行、NCMSが1.2K行であり、ほぼその規模が把握できる。なお表3ではプロトコル･エラーの処理は普通解放の中に含まれている。

表2 Tマネジャのプログラム規模

モジュール名	実効行数
メイン(宣言文等も含む)	932
Tプリミティブ入出力処理	374
Nプリミティブ入出力処理	239
クラス･プロセス入出力処理	314
セパレーション	65
TPDUのTCへの関連付け/デマルチプレクシング/リコンバイニング	1034
チェックサム (検査)	133
NCMS	1227
合計 (NCMS無し)	3091
合計 (NCMS有り)	4318

表3 クラス4のプログラム規模

モジュール名	実効行数
メイン(状態管理等)	2507
共通モジュール	1276
NCへの割付け	84
TPDU転送	65
セグメンティング/アセンブリング	374
コネクション確立	1586
コネクション拒否	117
普通解放(明示)	387
DT TPDU番号付け	33
優先データ転送(NLの普通データ)	464
TPDUの保留(確認応答TPDU)	539
フロー制御	176
チェックサム (設定)	44
リファレンス連結	75
タイムアウトによる再送	417
リシーケンシング	203
インアクティビティ制御	93
スプリッティング/リコンバイニング	126
クラス4独自の処理(AKの転送等)	834
合計	9400

②従来のTPプログラムにクラス4を追加作成するために、作成済みのクラス0から3のプログラムの変更箇所は、TCMTにスプリッティング最大数の一項目を追加しただけである。これは従来のTPプログラムを作成した時点における、Tマネジャとクラス･プログラムの間の機能分担の方針に従って、クラス4のプロトコル機構が実現できたためである。

③NCMSをTマネジャに追加作成する場合も、Tマネジャとクラス･プロセスのインタフェースは変更する必要がなく、クラス･プログラムの変更も少ない。

④異なるクラスの複数のTCをサポートするTPプログラムでは、複数のクラスにまたがった処理を行うTマネジャの実装方法が重要な問題となる。図8にクラス4、NCMSを含むTマネジャの処理の流れを示す。

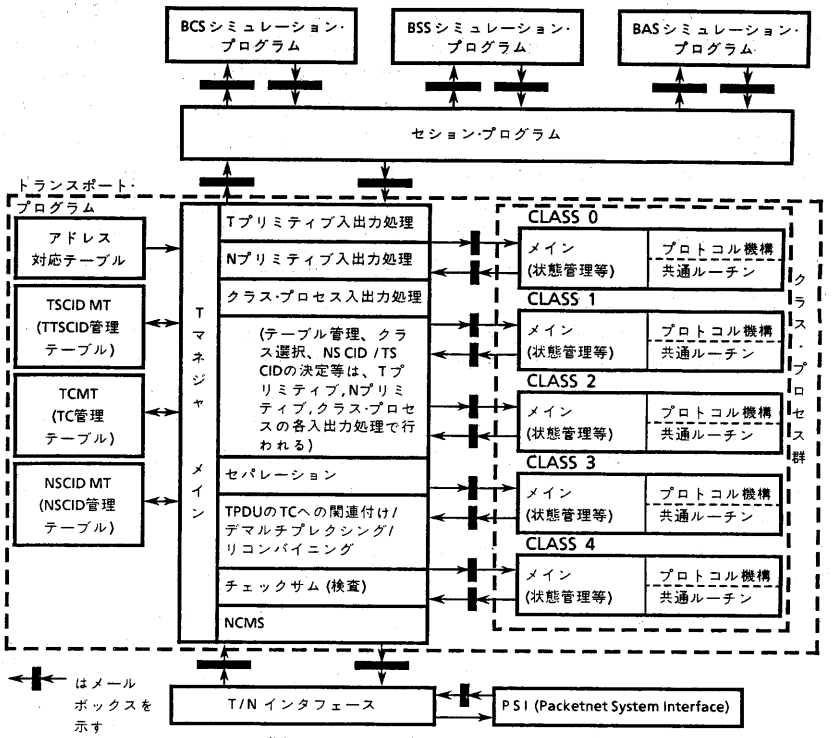


図7 システム全体のプログラム構成

⑤クラス4の実装過程で、勧告の中に若干のエラーや不明な記述があることも見出したが、プロトコルとして正常に動作することを確認した。

⑥NCMSにおいて、NCに多重化できるTCの数やTCの割付けられていないNCの寿命等のネゴシエーションは今後の重要な拡張項目であると考えている。

⑦今回作成したTPプログラムでは、TPの全仕様のうち、コンカティネーション、マルチプレクシングとスプリッティングを同時に行うこと、転送ウィ

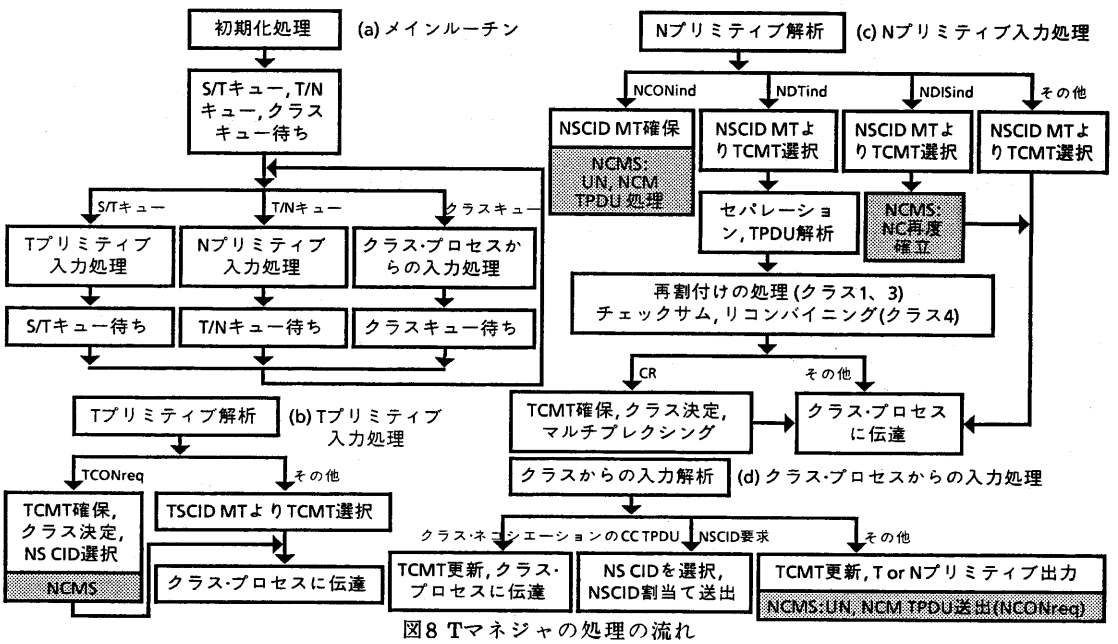


図8 Tマネジャの処理の流れ

ンドウのリダクションを実装していない。これらについては今後、その契機などについて検討して必要がある。

4.2 スループットの測定

TPプログラムのスループットを次のように測定した。

(i) X.25網折返しと内部折返しの二形態を対象とした。内部折返しでは、T/Nインタフェースの代わりに内部折返し用のプロセスを用いた。

(ii) TPプログラムの上位に簡易なファイル転送プログラムを作成し、20Kバイトのファイルを転送する。

(iii) ファイルは100バイトずつに区切ってTPプログラムに渡す。即ち、TL及びNLにおけるセグメンティングは行わない。

(iv) 転送時間の測定はファイル受信側で行い、ファイルの最初のデータの受信から最後のデータの受信までを測定した。

(v) X.25回線は4800bps、X.25レベル3のウィンドウは3である。

(vi) TPの各クラスのパラメータは以下の値を用いた。

- TPDUサイズは128バイトとした。
- クラス2はフロー制御を行う場合と行わない場合を対象とした。
- クラス2、3、4の転送ウィンドウ(クレジット)は15と充分大きい値を使用した。
- クラス4におけるタイマ値としては、伝送エラーを想定しないためT1タイマは600秒と充分大きな値を使用し、その他のタイマは2.3.5に述べた値を使用した。
- クラス4でスプリッティングに使用したNCは3本とした。

各クラスの測定結果を表4に示す。この測定ではCPUの使用率はほぼ100%であった。

表4 TPプログラムのスループット

	クラス0	クラス1	クラス2 #1	クラス2 #2	クラス3	クラス4
X.25折返し	2.4	1.4	1.4	2.3	1.4	1.2
内部折返し	3.6	2.4	2.3	3.4	2.2	1.7

注：単位はKbpsであり、クラス2#1と#2は各々フロー制御を使用した場合と使用しない場合である

①今回の実験では、一つのシステムでファイルの送受信を行っているため、行き帰り2本のTCをサポートした場合のスループットを測定していると考えられる。

②マルチプレクシングを行わない場合は、クラス1、2、3は、それぞれ類似したスループット特性を示す。なおクラス2はフロー制御を行った場合はクラス0と同様のスループットが得られる。

③クラス4も品質の良いNC上で使用される場合は、クラス1、2、3に近いスループットが得られる。

④クラス4に対しては、チェックサムの使用/未使用、スプリッティングに使用するNCの数をかえてスループットを測定したがほぼ同様な結果を得た。これはチェックサムの処理が小さく、また使用したNCの伝送速度が処理速度に比べて充分に大きいことによる。

⑤X.25網折返しでは、VAX/VMSの提供するPSIは殆ど処理時間を必要とせず、内部折返しとのスループットの差はT/Nインタフェースによるものである。今後はT/Nインタフェースの機能をTマネジャに組み入れる等の方法でスループットの向上を検討してゆく必要がある。

5. おわりに

本稿ではTPの全仕様を実装に向けて、新たにクラス4、NCMSを追加作成した結果について報告した。本TPプログラムは実装の一つの例ではあるが、異なるクラスのTCを複数サポートするシステムに対しては、一般的な実装方法を示唆するものであると考えている。今後とも、プログラム構造の変更を進め、スループットの向上、パッケージ化等について検討するとともに、OSI各種プロトコルを実装してゆきたい。最後に、日頃御指導頂くKDD研究所鍛治所長、野坂副所長、小野次長、浦野情報処理研究室長に感謝します。

参考文献

[1] : CCITT, Rec.X.214, X.224, Oct. 1984.
[2] : CCITT, Rec.X.215, X.225, Oct. 1984.
[3] : ISO, "Addendum to DIS 8073 to include a Network Connection Management Subprotocol," ISO TC97/SC6/WG4 N20, Mar. 1985.
[4] : 鈴木,加藤, "OSIトランスポート・プロトコルのインプリメントと製品検証," 情処学会 分散処理システム研究会, 22-9, May 1984.
[5] : 鈴木,加藤, "OSIセッションレーヤ標準のインプリメント",情処学会分散処理システム研究会,24-4, Nov. 1984.
[6] : 鈴木,加藤, "OSIトランスポート・プロトコル・クラス4のインプリメント", 第30回情処全大, 4P-2, Mar. 1985.
[7] : 加藤,小花,鈴木, "NCMSの拡張とその実装," 昭和60年度信学全大, 1863, Mar. 1985.