コンピュータ・ネットワーク17−5
（1978. 10. 13）

DONAにおける端末の考え方

松下 温　山崎 晴明　疋田 定幸　辰己 俊文
（沖電気工業株式会社）

1. はじめに

コンピュータネットワーク技術の発達に伴い、従来ホストコンピュータの単独なリソースとしてしか使用されなかった端末が、ネットワーク内の共通リソースとして使用できるようになった。しかしながら端末には多種多様なものがあり、その制御方法も様々である。仮想端末はこの制御方式を標準化する目的で考えられている概念である([1]～[4])。端末を仮想化することにより種々のアプリケーションプログラムは容易に端末を使用することができる。ネットワーク上に端末が分散したシステムを構築するうえでもう1つ重要な事項は端末の管理である。すなわち、システムで混乱なくかつ効率良く端末をリソースとして管理することは実際のアプリケーションでは重要である。ネットワークのより高度な利用を考えると、ネットワークは端末の生成消滅だけでなく、さらに端末アドレスを管理する必要があると思われる。

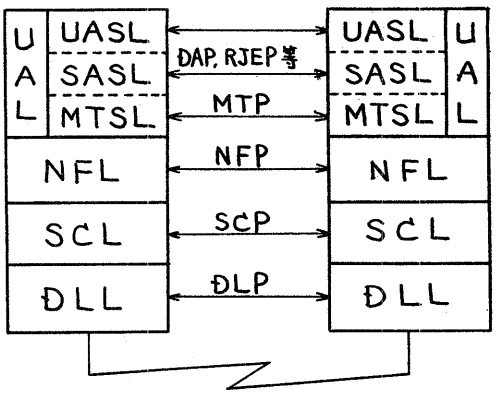
本稿では、DONAにおける端末をその管理という観点から、管理の必要性を述べ、端末アドレスの管理方式、管理のアルゴリズムについて述べる。

2. DONAの端末管理

2.1 端末管理の論理構成上の位置

本節では、DONA([5]～[9])の論理構成の概要を述べ、端末管理機能が論理構成上どのように位置付けられるかについて述べる。

DONAのプロトコル階層は、図2.1に示す如くDLL，SCL，NFL及びUALの4階層からなる。DLLは隣接ノード間でのデータ転送を実行する階層である。SCLはその上位層であるNFLに対してデータグラム型のパケット交換機能を提供する階層であり、他ノードのSCLとともに仮想的なサブネットワークを形成する。NFLはいわゆるホスト-ホスト・プロトコルを実行する階層である。UALはアプリケーション・オリエンテッドな機能を遂行する階層であり、さらに3つのサブレイヤにより構成されている。MTSLはUALで実行される各プロトコルに共通に必要な機能、すなわちプロセス相互間のメッセージ通信を実行するサブレイヤである。SASLは汎用的な標準プロトコルを実行するサブ



DLL; Data Link control Layer
SCL; Switching Control Layer
NFL; Network Facility Layer
UAL; User/Application Layer
MTSL; Message Transfer Sub-Layer
SASL; Standard Application Sub-Layer
UASL; User/Application Sub-Layer

図2.1　DONAのプロトコル階層

レイヤである。UASLは各システム毎の業務処理を実行するサブレイヤである。

DONAではアプリケーションプログラムや端末等、UALにおいて通信の主体となるものをサブスクライバと定義している。端末はそれ自身が1サブスクライバに対応させてもよいし、端末の持つ各デバイスをそれぞれ1サブスクライバに対応させてもよい。これらの様子を明確にするため、DONAにより実現されたネットワークの構成例を図2.2に示す。

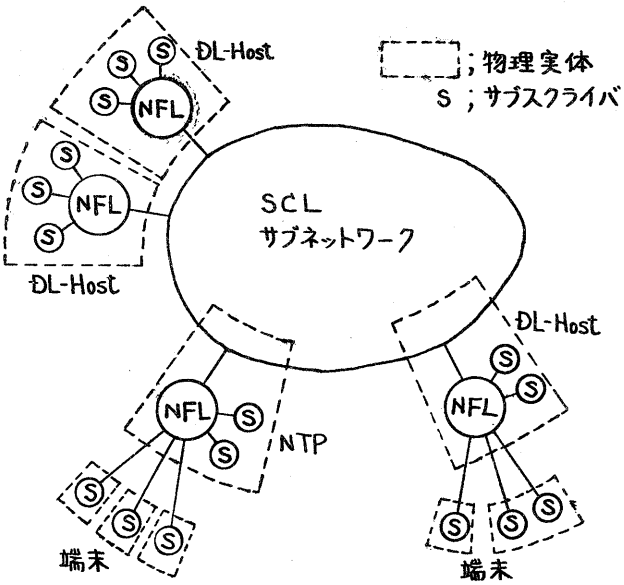


図2.2　DONAのネットワーク構成例

ネットワークにおける端末については、考慮すべき2つの大きな問題がある。

- 通信時に、多種多様な端末をいかに標準的な仕様にみせるか。
- 端末を1つのリソースとして、どのように管理するか。

前者は仮想端末の概念の導入により実現できる。DONAにおいても、仮想端末仕様が定義され、UALの各サブレイヤ内に展開されている[7]。後者について、DONAではネットワーク内の各種リソースを管理し、その円滑な運用を可能とするため管理サブスクライバの概念を導入している。この管理サブスクライバはUALのサブスクライバの1つとして位置付けられる。以下では後者の端末管理の問題に焦点をあわせて議論を進める。

2.2　端末アドレスの管理

コンピュータネットワークを介した端末への通信には、そのネットワークにおけるアドレスが必要である。これはネットワーク管理上決められたものでその使用法とか使用者名とかいう論理的な意味付けはなされていない。一方、端末と通信を行いたい人またはアプリケーションプログラムにとっては、相手のアドレスよりその論理名でアクセスした方が簡単である。すなわちネットワーク内の端末アドレスはネットワーク管理上付けられたものであり、ネットワーク運用の都合で変更され得る。一方その端末と通信する人およびアプリケーションプログラムは使用者や使用法によってその端末の論理名を決定する。このため端末のアドレスと論理名との関係を管理する必要がある。図2.3に端末へのアクセス手順例を示す。この図はAから端末BまたはCへ通信したい場合で、端末Bの論理名とアドレスはXが管理し、端末Cの管理はYがしているものとする。①Aから端末Bへ通信したい場合、XへBの論理名を使ってアクセスする。②XはBのアドレスを見つけ、これをAへ返す。③AはBのアドレスを使ってBと通信をする。次にAがCと通信したい場合、①AはCの論理名でXへアクセスする。④XにたまたまこのアドレスがなければYへアクセスする。⑤YはCのアドレスをXへ返す。②XはそれをAへ返す。⑥AはCと通信する。

図2.4に具体的な適用例を示す。ネットワークを適用した分散システムを考え、Aはシステムの状況を出力するアプリケーションプログラムとする。Aはメッセージ出力端末をαという論理名を使ってアクセスする。通常出力端末はBであるがBが障害のため出力できずCへ切替えるとする。Xはαと Bの対応関係をαとCの対応に変更する。これによりAは物理的にどの端末に出力されるかを意識せずその使用目的だけを意識してメッセージを出力できる。表2.1に端末アドレスの切替要因を示す。

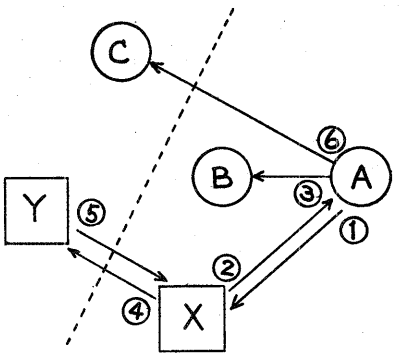


図2.3　端末へのアクセス

表2.1　端末アドレスの切替要因

要因	内　容
障害	端末障害により代替の端末へ切替える。これは人手を介する場合と自動の場合とがある。
保守	定期、不定期の保守作業により一定期間出力端末を代替端末へ切替える。
運用	運用条件により、例えば時間帯ごとに出力端末を切替える。

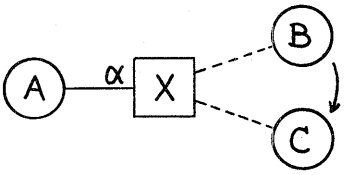


図2.4　具体例

以上の例は1つの論理名から1つの端末のアドレスが得られるものであったが、1つの論理名から複数の端末のアドレスが得られ複数の端末と通信を行う例、同一の論理名でありながら得られる端末のアドレスがサイクリックに変わる例等も考えられる。

以上述べた端末アドレス管理を個々のアプリケーションプログラム毎に考えていてはネットワーク全体として、管理が混乱し効率の良い運営が期待できない。端末管理上の問題を2.3項で述べる。

2.3　端末アドレス管理の問題点

端末アドレスをネットワーク内でどの様に管理するかはネットワークシステムを設計するうえで重要である。すなわち、

- システムのスループットを向上させる。
- システムの信頼性を向上させる。
- システムに拡張性を持たせる。
- システム運転時の通信コストを下げる。

これらの条件を考慮して設計する必要がある。端末アドレスの管理法には、各ノード間でデータを重複して持つか、重複して持たないか、そのときデータを集中して持つか、分散して持つかという管理上の問題がある。これを図2.5に示す。

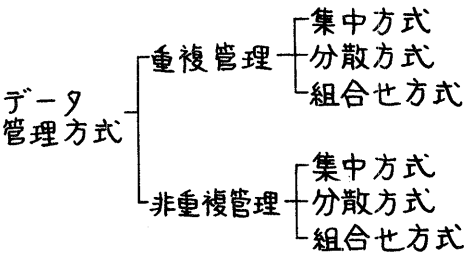


図2.5　端末データ管理方式

ネットワーク上に分散された端末へアクセスする場合、その端末アドレスはどこへ

問い合せれば得られるかという情報はネットワーク内の各管理ノードは最低限知っておかねばならない。もちろん管理ノードが1つで集中管理されていれば問題ないわけであるが、管理プロトコルを考えるうえで、管理ノードは複数を前提に考えておいた方が一般的である。すなわち各管理ノードは管理ノード間の共通データを重複して保持している。この様なデータを変更する場合、図2.6および図2.7に示す問題が発生する。

図2.6でXおよびYは論理名と端末アドレスの対応関係を管理している。AAAは論理名でKがアドレスである。端末Cのアドレス KがLに変更される場合を考える。ノードYのアドレスが変更され、ノードXのアドレスが変更される前にAからAAAという論理名でアクセスされたとする。しかし端末Cのアドレスはネットワーク内でLに変更されておりAはCと通信できない。

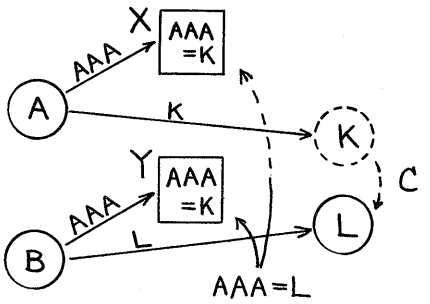


図2.6　出力端末無

図2.7において管理ノードXおよびYは論理名AAAはどのノードで管理されているかだけを知っている。AAAのアドレスはKであることはYが知っている。AAAの出力端末をKからMに変更する場合を考える。ノードYでAAA=YをAAA=Xに変更しかつノードXではAAA=Xがとどかなかった場合、ノードXで発生した論理名AAAのアクセスはノードXとノードYの間をループしてしまう。端末アドレスを管理するうえで共通データの更新では以上の問題を解決せねばならない。

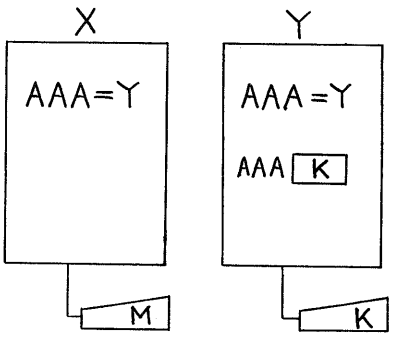


図2.7　アクセスのループ

本稿では端末アドレス管理におけるアドレス管理体系の考え方およびその際最も困難な問題となる共通データの更新プロトコルについて述べる。

3.　管理の方式

端末管理の方式としては、各端末の管理情報をネットワーク内のどのノードに持たせるかによって、以下のものが考えられる。

(1) 集中管理方式　　ネットワークの全端末の管理情報を1つの管理ノードで集中して持つ方式である。この方式は制御が容易であり、重複データを持たないので、consistencyの維持も簡単に行える。しかし、この方式は管理ノードがダウンするとネットワーク全体の通信が不可能となってしまう、といった信頼性の面における難点がある。また、すべての通信の開始要求が管理ノードに集中するため、ネットワークの規模が大きくなると、トラヒックが管理ノードのまわりに集中してスムーズな運用ができなくなってしまう可能性もあり、将来の拡張性に欠ける。

(2) 完全分散方式　　ネットワーク内のすべての端末が、他のすべての端末の管

理情報を持つ方式である。信頼性や通信開始時の効率の面からはひじょうに優れているが、全端末が重複データを持つので、管理上の変更が発生した場合には、全端末に変更通知を出さねばならず、通信量が増大するし、consistency の維持も難かしくなる。また、機能や容量に制限のある端末に、他のすべての端末の管理情報を持たせることは非現実的であり、かつ拡張性にも欠け、実際には採用し得ない方式である。

(3) 領域分割方式　集中管理方式、完全分散方式の中間形態であり、ネットワーク全体をいくつかの領域に分割し、各領域にその領域内の全端末を管理する一つの管理ノードを置く方式である。領域外の端末については、どの端末がどの領域に属するかという共通のディレクトリを持つことになる。この方式ではネットワークの規模が大きくなるに従い、領域数を増やすことにより、比較的簡単に拡張を図ることができる。

完全分散方式は、前述のように非現実的な方式であるから、ここでは評価の対象からはずし、集中管理方式と領域分割方式とについて、ネットワーク全体の通信量および領域分割方式の場合の、領域内の各管理ノードが処理しなければならない通信量を評価する。ただし、ここでいう通信量とは、管理情報の変更に伴う通信及び一般の通信において発呼サブスクライバが相手サブスクライバのアドレスを得るまでの通信である。サブスクライバ間のリンクの確立以後の通信は、管理方式とは無関係であるから、この評価の通信量には含めない。

1ノード当り、単位時間に発生する呼数をkとする。このうち、管理情報の変更に関するものの割合をpとする。従って、一般の通信要求は$k\cdot(1-p)$回発生する。また、ネットワーク全体のノード数をnとする。

集中管理方式では、1回の通信要求に対して、管理ノードと端末間に2回のインタフェース（通信要求とその応答）がある。また、1回の更新要求に対しても同様であるから、単位時間当りの管理ノードの処理する通信量（＝ネットワーク全体の通信量）は

$$2k\cdot(n-1) \qquad (1)$$

である。

次に、領域分割方式での通信量を求める。ネットワーク全体をmヶの領域に分割したとすると、各領域の管理ノードの通信量は表3.1のようになる。ここでαはローカリティ定数であり、$\alpha=1-(n-n/m)\cdot(1-\alpha)/(n-1)$（$\alpha$はノード当りのローカリティ）と定義する。従って、ひとつの管理ノードが処理しなければならない通信量は

$$\frac{n}{m}k\left[p\{2+10(m-1)(1-\alpha)\}+(1-p)(6-4\alpha)\right]-2k \qquad (2)$$

となる。また、ネットワーク全体の通信量は

$$kn\left[4-2\alpha+p(1-\alpha)\{5(m-1)-2\}\right]-2km \qquad (3)$$

となる。

(2)、(3)において、$m=1$とおくと集中管理方式となるが、これは(1)と同じになる。(2)、(3)を$k=1$, $p=0.01$の場合について、m, αをパラメータとして表わしたものを図3.1および図3.2に示す。

図からわかるように、ネットワーク全体の通信量は、領域数mとともにふえていくが、ある点をピークにして減少する。これは領域数が少ないうちは、領域間の通信量の増加のためにネットワーク全体の通信量が増加するが、領域数が多く

なると、管理ノード内のローカルな処理で済む割合が増えるため、ネットワーク全体の通信量が減少するためである。管理ノード毎の通信量は領域数と共に減少するのは自明のことであるが、ローカリティが多いと、その効果もかなり大きいことがわかる。

以上、通信量について集中管理方式と領域分割方式を比較してきたが、どちらの方式が良いかは、通信量の面からみても、アプリケーションにより異なると考えられる。従ってアーキテクチャとしては両方式を考慮する必要があるが、集中管理方式は領域分割方式の特別な場合（領域数が1）に含まれるので、一般的には領域分割方式を想定してプロトコルを構築すればよいことになる。

表3.1　領域の管理ノードの通信量

	発呼ノード		1回の発呼に要する通信回数	1ノード当りの単位時間当りの発呼度数	ノード数
通信	端末	領域内	2	$k(1-p)a$	$\frac{n}{m}-1$
		領域外	4	$k(1-p)(1-a)$	$\frac{n}{m}-1$
	管理ノード	領域内	0	$k(1-p)a$	1
		領域外	2	$k(1-p)(1-a)$	1
	他領域		2	$k(1-p)(1-a)\cdot\frac{1}{m-1}\cdot\frac{n}{m}$	$m-1$
更新	端末	領域内	2	kpa	$\frac{n}{m}-1$
		領域外	$5(m-1)+2$	$kp(1-a)$	$\frac{n}{m}-1$
	管理ノード	領域内	0	kpa	1
		領域外	$5(m-1)$	$kp(1-a)$	1
	他領域		5	$kp(1-a)\cdot\frac{n}{m}$	$m-1$

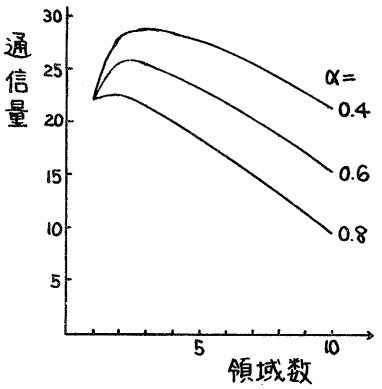


図3.1　ネットワーク全体の通信量

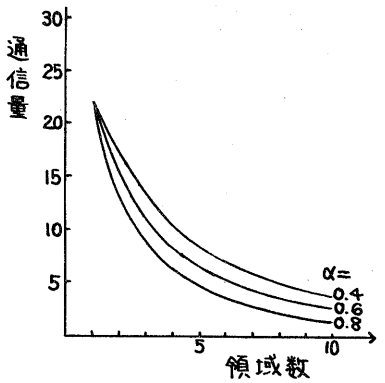


図3.2　管理ノード当りの通信量

4. 端末管理データの更新プロトコル

端末管理において、最も困難な問題は端末管理共通データの更新である。管理ノード間の共通データ更新では、以下に示す条件が満されなければならない。

(a) データ更新中の共通データに対するアクセス禁止。　更新中の端末管理共通データがアクセスされる事によるシステムの混乱を防止する必要がある。

(b) データ更新の同期。　どの管理ノードもお互い矛盾のない管理データを保有するために、データの更新がすべての管理ノードで同期して行われねばならない。

(c) 障害時の対処。　ネットワーク内のノードに障害が発生した場合、そのノードを除いてデータの更新は正常に処理されねばならない。また、障害修復時に当該ノードは共通データのconsistencyを保ってネットワークへ組み込

まれねばならない。

(d) 複数のデータ更新要求の競合制御。　共通データに対する更新が複数発生した場合、どちらの更新もお互い更新許可が得られず行われないような事がある。これを防がねばならない。

これらの要求を満足するために、領域分割型の管理方式において、端末管理データの更新は次の手順で行われる。

(図4.1参照)

①領域内ノードがその管理ノードに更新を要求する。

②管理ノードは"更新要求"を他の管理ノードにブロードキャストする。

③これを受けた各管理ノードは、これに対して"更新許可"を与え自分の領域内に対し共通データに対するすべてのアクセスを禁止する。

④全ノードから"更新許可"を受けたときのみ、管理ノードは"更新データ"をブロードキャストする。

⑤"更新データ"を受けた各管理ノードは更新を行い、"更新完了"を通知する。

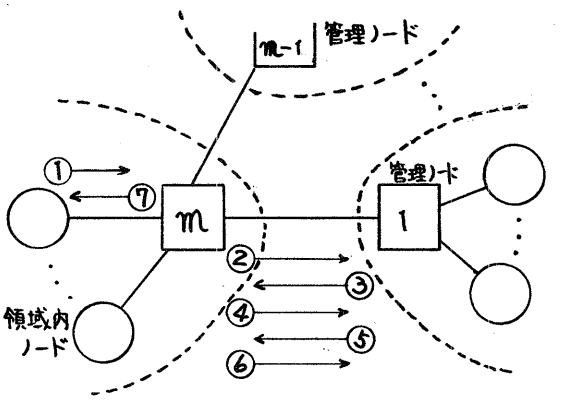


図4.1　端末管理共通データ更新手順

⑥"更新完了"を全管理ノードから受けると、"完了確認"をブロードキャストし、各ノードで共通データへのアクセスを許可する。

⑦"更新要求"を出した管理ノードは更新の終了を領域内ノードに通知する。

以上のように、共通データの更新は、領域内ノードと管理ノード間で2回の通信を行い、また各管理ノード間では各々5回の通信を行うことによって終了する。

上述の条件(a)は、③の共通データへのアクセス禁止によって実現される。また(b)は、④のように更新許可を全管理ノードから得ることによって実現される。(c)の障害発生時において、ノード障害は各ノードの時間監視によって検出され、障害ノードリストに書き込まれる。この障害ノードリストを全ノードに通知することによって、障害ノードは処理の対象からはずされる。障害修復時に当該ノードは障害修復を全ノードに通知し、障害ノードリストの修正を要求する。また障害修復ノードは隣接ノードから最新の端末管理共通データをコピーする。これらは通常のデータ更新処理と同じ手順により更新許可を得て実行され、以上の処理が終了後"完了確認"を全ノードに通知する。

(d)に示した問題への対処を以下に述べる。データ更新処理が複数発生した場合、どちらか1つの処理が優先して実行されねばならない。さもないと(d)の問題が発生する。この保証をするため、各データ更新要求に優先度を与える方法を検討する。表4.1に優先度制御の各方式を示す。ノード番号法ではノード番号に優先度を対応させ、データ更新要求に付与されたノード番号により実行の優先度が決められる。この方式はデータ更新の頻度が少ない場合は問題ないが、頻度が高くなると優先度の低いノードから発生する更新要求は実行され難いことがある。タイムスタンプ法では更新要求にタイムスタンプを付与し、この値によって優先度が決められる。絶対時刻方式はネットワーク内共通の時刻を更新要求に付与する方式である。絶対時刻を使用することによりノードによって更新がされ難いと

表4.1　優先度制御法

制御方法		内　容
ノード番号法		各ノード番号に優先度を対応させる。
タイムスタンプ法	絶対時刻	ネットワーク内各ノードは共通の絶対時刻を使用し、時刻の前後関係で優先度を決める。
	相対時刻	各ノードは相対時刻（サイクリックカウンタ）を使用し、カウンタ値の大小関係で優先度を決める。

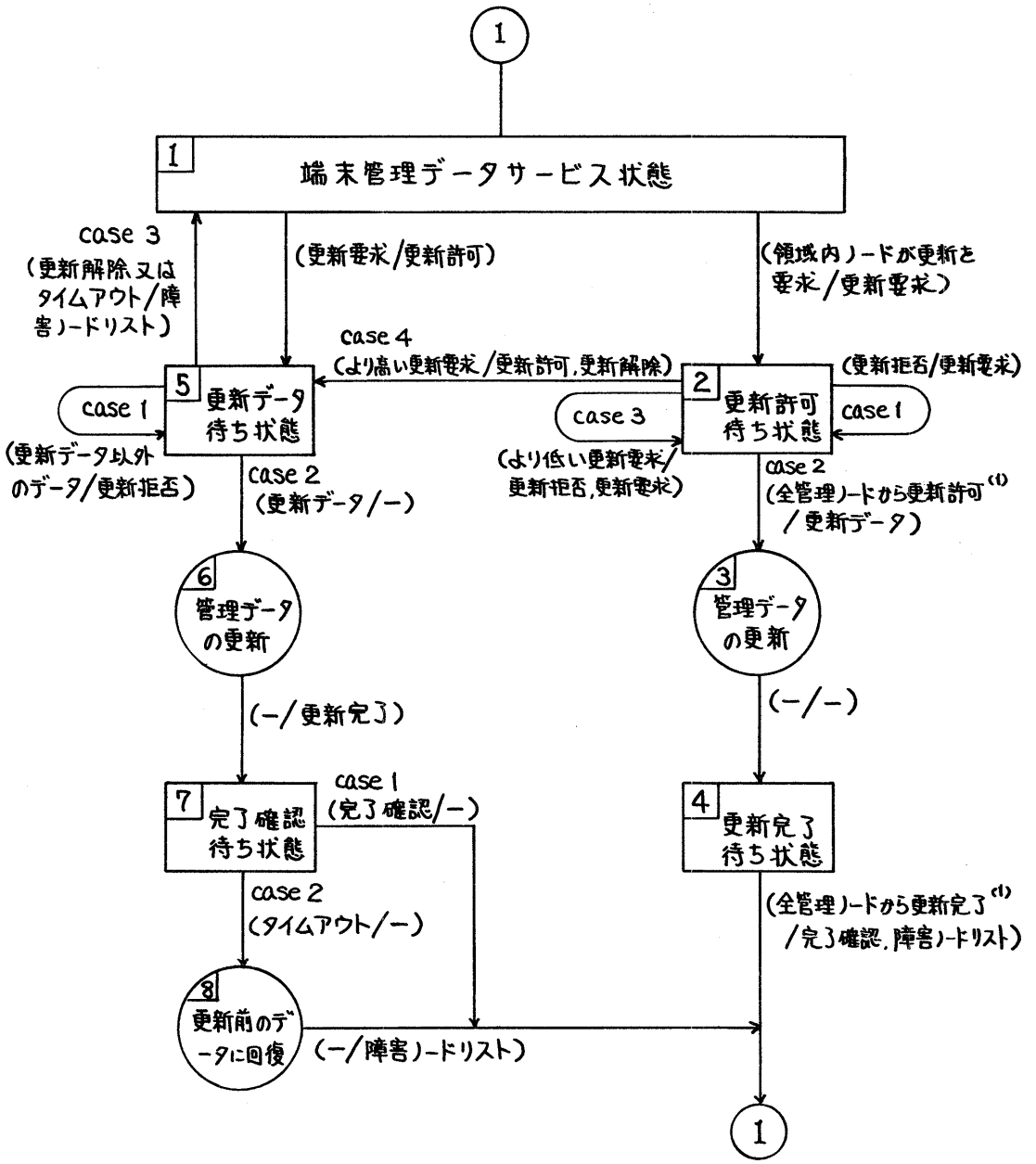
いう問題はなくなるものの、ネットワーク内に共通の絶対時刻を持つことは技術的に難しい。一方相対時刻方式は各ノードでカウンタを管理し、このカウンタ値を更新要求に付与して他ノードに送る。更新要求受信ノードは自ノードのカウンタを最新のカウンタ値に設定しておく。カウンタの歩進は共通データ更新要求を送出する度に行う。相対時刻方式ではネットワーク内に共通の物理時刻を持たないため技術的には容易であるが、更新要求の頻度が高くなると同一の相対時刻が割当てられやすい欠点がある。いずれにしてもタイムスタンプ法は同一のタイムスタンプが複数の更新要求に割り当てられる可能性があるため、ノード番号のような他の情報と組み合わせて使う必要がある。

端末データ管理において、その共通データ更新頻度は高くないと予想される。また相前後して発生した更新要求を、発生時刻順に処理しなくても端末管理データ間のinconsistencyは防ぐことができる。従って端末管理における更新の優先度制御にはノード番号法が採用できる。

以上(a),(b),(c),(d)の条件を満足するプロトコルを図4.2に示す。図4.2で長方形は安定状態を示し、円形は過渡状態を示している。状態2は、ある管理ノードが領域内ノードから更新を受け付け、更新要求をブロードキャストした後、これに対する更新許可を待っている状態である。Case 1はこの管理ノードが更新拒否を受けた場合で、このときタイマをリセットし、更新要求を再送し、状態2を維持する。Case 2は、全管理ノードから更新許可が到着した、あるいは規定時間内に更新拒否が1つも到着しなかった場合であり、この管理ノードは更新データと障害リストをブロードキャストする。Case 3はより低い優先度の要求が到着した場合で、この管理ノードはこれに対し更新拒否を返送し、自分の更新要求を再送する。Case 4はより高い優先度の要求が到着した場合で、この管理ノードはこの要求に対し更新許可を返し、自分の要求の解除をブロードキャストし、更新データ待ち状態に移行する。状態5は更新データを待っている状態であり、管理ノードは共通データに対するすべてのアクセスを禁止している。状態7は更新完了確認を待っている状態で、管理ノードはCase 1では共通データのアクセス禁止を解除し、状態1にもどる。またCase 2では、更新要求を出した管理ノードの障害と判断し、データを更新前のデータに回復し、状態1にもどる。

5. おわりに

本稿ではDONAの端末管理の考え方を述べた。端末管理方式としては領域分割型管理方式を提案し、これが実際のアプリケーションにも柔軟に対応できる方式であることを定性的、定量的に確認した。また領域分割型管理にとって最も困難な問題となる端末管理共通データの変更方式を述べた。DONAネットワークにおいて、DL-Host（Decentralized Local Host）またはNTP（Network Ter-



(注)(*/**)

*;外部からの入力又はタイムアウト

**;外部への出力

注(1);タイムアウトノードがあった場合、障害ノードリスト作成

図4.2 端末管理データの更新プロトコル

minal Processor）が端末管理ノードとなることができ、アプリケーションごとに柔軟な対応が可能である。

［参考文献］

[1] P. Schicker, H. Zimmermann "Proposal for a Scroll Mode Virtual Terminal" INWG Protocol 72 July 1977

[2] J. Davidson, W. Hathaway, et. al. "The ARPANET TELNET PROTOCOL: Its purpose, principles, implementation, and impact on host operating system design"

[3] Barber DLA "The Role and Nature of a Virtual Terminal" INWG Protocol 63 Feb. 1977

[4] "Preliminary draft version of the IFIP WG6.1 proposal for a Virtual Terminal Protocol" INWG Protocol 78 July 1977

[5] Y. Matsushita, et.al. "An Overall Network Architecture Suitable for Implementation with either Datagram or Virtual Circuits Facilities" Computer Communication Review Vol. 8 No. 3, July 1978

[6] Y. Matsushita, et.al. "An effective utilization method of public PSN in DONA" ICCC 1978

[7] 松下，他「DONAにおける仮想端末，ユーザ／アプリケーション層の設計思想」信学会，電子計算機研究会 EC 78-8 (1978.5月)

[8] 松下，他「DONAのハイレベルプロトコル」コンピュータネットワーク研究会 1978年7月

[9] 松下，他「DONAのハイレベルプロトコル」情報処理学会才19回全国大会

[10] Thomas R.H., et.al. "A solution to the concurrency control problem for multiple copy database" COMPCOM Spring, 1978