分散環境における諸資源の命名法および名前管理機構の構成法

古宇田　フミ子　　田中　英彦　　元岡　達
（　東京　大学　　工学部　）

１　はじめに

計算機システムにおける名前は対象を指し示すものとして、重要な役割を担っている。名前の付け方は、処理効率に大きく影響する。ところが、この点に関しては、従来、それほど考慮が払われてこなかった。

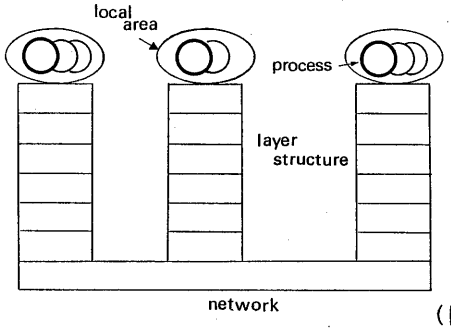
そこで、本研究では、計算機システムにおける名前の機能や役割を明確にし、また、分散環境に向いた名前管理方式の一案を提案する。

２　考察の前提

２－１　設定環境

ここでは、網環境と分散環境とを次のように区別する。網環境はhostの集まりとそれらを結ぶ回線からなる。分散環境は、網環境の上に立ってソフトウェア機能まで含めたものを言う。

名前の決定方式で必要な網環境の諸要素は、概念的には次の通りであり、これらを前提とする。



（図１）

１）　構造：　階層構造（layer ）を採用する。同じ層内でのprotocolと層間でのinterface の構成法は与えられていると仮定する。

２）　形態：　機種・構成・接続形態・規模、等の形態は問わない。

３）　制約：　分散環境では、網上の伝達遅延や回線障害は避けられない。換言すると、情報の伝達には時間がかかり、又、途中で障害にあって、目的地まで辿りつかないことがある。従って、伝達遅延と回線障害は必ず起り得るものとする。

これを各ノードから眺めた場合、次のことも前提とする。

４）　プロセス：　プロセスはそれぞれのローカルな環境にあり、これらは網環境の端に接している、と考える。（図　１）

５）　プロセス間通信：　プロセス間のコミニュケーションは一ブロックのデータ（メッセージ）授受によって行なう。コネクションを張って行なう場合やコネクションレスにも対処できる様に考慮する。

ユーザからは、プロセス間通信を、メッセージをやりとりする形で、同一のローカル環境内でも、異なる環境間でも同じ手法で行なえるものとする。

今後、上記の環境を前提にして考察を進める。

２－２　分散環境における名前

網環境が整備された分散環境における名前の付け方に必要な事項を挙げる。

１）　命名：　名前の初期化や更新等に起因する網上の通信のオーバヘッドができるだけ少なくなる様にする。即ち、名前を付ける為（又は変える為）の操作はできるだけ通信回数が少なくなることが望ましい。

階層構成を採ることにより名前は層間でmappingをする必要がある。そのために名前の構成は制約を受ける。

回線を伝わることから生ずる遅れの為に名前の再使用には制限を受ける。特にresponseを必要とする場合等では、名前の一意性を時間的、空間的に保障できることが大切である。

又、回線障害の時には、単なる遅延との違いが見極められるようにすることや、障害発生場所での対処の仕方や他所での対処の仕方も重要である。

２）　別名：　同一対象に対する名前の別名（alias ）を許し（対象：名前＝１：複数）使用者毎の目的に合った名前を付ける。別名を含めて、名前付けは、利用しやすく、また使用目的に応じて適切な名前付けが行えるようにする。また、layer 構造に見合った名前が必要となる。例えば、ローカル・エリア内で用いる名前と、通信路で用いる名前を異なった形式にすることや、名前の表現をbit 列で示すか、文字列にするか、等が挙げられる。

３）　管理：　分散環境での名前管理を容易で明確にすることが必要である。特に、別名を用いることによる名前間の対応関係や、名前の変換が容易にできる様にすることや、分散環境という制約から来るコピーや更新の管理問題を明確にする必要がある。

更に、名前構成法の中に、他の機能の一部も取り込み、全体としての処理効率を上げることを目指す。即ち、4） 保護機構： 対象に対する名前の付け方の一面として、protectionやcapabilityをも名前機構に組み入れる。

5） プロセス間通信： 通信効率が上がるような名前付けを行なう。

2－3 考察の対象と範囲

或る対象があれば必ずそれに対する名前を付け得る。名前を付ける対象は限り無くあるが、ここでは、主に計算機におけるプロセスの名前を考察する。具体的には、次のようになる。

1） プロセスが仕事を行なう場合、相手プロセスの区別を効率的に行なうためのプロセスの名前の付け方。

2） プロセス間通信を行なう場合、効率が上がるような名前。

3） 分散環境でのプロセスの名前管理法。

4） 将来、プロセス以外の対象に対する命名にも役立つ一般的な名前。

3 名前の目的と特性

3－1 名前を使う目的

名前とは何であろうか。名前は対象を示す指標となり、複数のものが区別できる。名前を用いると対象を間接的に呼び出すことができる。このことから、名前は参照（reference ）のために用いられ、又、名前を用いることにより、対象を区別（distinct）することが可能になることがわかる。対象を正しく区別して参照するためには、名前と対象との間や名前と利用者との間で対応付けが正しくなされなければならない。そこで、区別と参照という観点と、対応付けという観点から考察を進める。

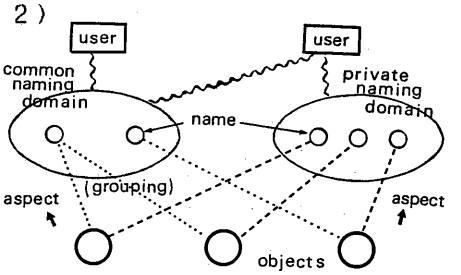
3－2 名前の一意性と二面性

利用者から見て、対象の区別がつくこと とは、対象と名前が一対一に対応し、かつ名前が一意に定まること と、言い換えられる。対象を参照する場合は、対象の持つ或る性質を区別する名前が考えられる。この場合は対象の持つ性質の一部と名前とが一対一に対応し、この名前が一意に区別できることから、参照可となる。区別用の名前の例としては、区別する対象に通し番号を割り当て、この番号を名前として用いる方法がある。参照の名前は意味まで含めて区別できるもので、例えば、ＴＳＳ 等のプロセスの機能名がある。

一意であることを言うためには、どの範囲で一意かを明確にする必要がある。さらに、1） 空間（space ）における一意性と、2） 特定の時刻（time）での一意性との両方を考慮しなければならない。

空間や時間での一意性は、全空間、全時間で一意になるように決めることもできる（集中管理方式）が、こうすると、特に分散環境において、その管理が難しくなりがちなので得策とは言えない。

そこで、名前が一意になる範囲を限定・制限し、その枠内では時間的、空間的に一意性を保障することを考える。名前の一意性が保障された範囲を、ここでは naming domain と呼ぶ。naming domain では、要素は、すべて異なる。（図 2）



対象と名前の対応付けの方法について考える。対象はそれを見る観点（aspect）によって種々の性質を持つものとして捉えることができる。捉えた個々の要素に名前を付けたものが、対象を或る観点から見た名前となる。

そこで、見方を或る特定の観点に限定し、複数の対象を眺めると、これらの対象はこの観点に見合った名前を付け得る。観点によっては、複数の対象が同一の要素に含まれることもある。即ち、複数の対象は一つの観点の中で見ると、同じ属性を持つものも生ずる。この場合、これらは一つのグループに纏め、グループ毎に各々の名前を与えることができる。この名前は同じ観点を持つ要素として一纏まりにして区別できるので、これを一つのnaming domain として構成できる。即ち、対象を或る見方（aspect）によって捉え、この側面に見合った対象の名前を付け、これらの名前の集合を一つのdomainとしている。見方は単一では無いので、一つの対象に対して複数個の名前が対応する。（図 2） 以上を纏めると、naming domainは一つの対象を或る目的に沿った特殊なフィルタを通して見た結果、区別し得る要素に名付けた名前の集合と言える。

ここで対象の見方について調べてみる。見方としては、前述の特定の機能・性質に注目したものや、対象を識別するための名前があるが、働きは異なる。

機能や性質を表す名前は幾つかの対象の持つ共通

なものを抽出したものであり普遍性が強い。一方、対象参照・区別用の名前は利用者にとって識別できればよいものである。

このことから、名前の用法としては、

（１） 性質を表す・・・対象の属性・機能を説明するための表象であり対象を参照するための指標となる。（対象の一部のみを捉える）

（２） 識別子として・・・対象を区別するための指標を表わす。（対象を全体として捉える。）

の二面があることがわかる。

尚、ここで述べた名前の二面性のうち、識別子としての名前は、ＳＣ１６／ＷＧ１ で検討されている primitive name に、また、性質を表わす名前は descriptive name に近い、と考えられる。［Ｎ１３９０］

従って、naming domainについても二種類考えられる。即ち、プロセスの捉え方として、１） プロセス自身の性質を表わすもの ２） プロセスの単なる識別子を表わすもの、とに対応して、naming domainにも１） プロセス自身の性質を表わすもの、例えばプロセスの機能、プロセスの存在場所、プロセスに関するcapabilityを表わすもの、等と、２） 他プロセスの識別用に用られるもの、例えば、プロセスの状態制御用、他プロセスの呼称 等との二種類が考えられる。

naming domain の利用者について調べる。利用者はnaming domainの二面性を組み合せて使い分ける。また、個人で用いるnaming domainと、共同で用いるnaming domainがある。

識別子としてのnaming domain は「誰から見て区別が付くか、」という観点の違いで意味が異なってくる。対象を識別しようと意図する利用者が、この識別子としてのdomainの持ち主となる。持ち主は個人の場合も共同の場合もある。同じ対象に対して、識別子 domain の持ち主毎に異なる名前が付けられることもあり得る。

一方、性質を表す domain は、普遍性が強いことから、どの利用者からも利用できる様にしたい。その為には、domain の利用者としての各プロセスに性質を表す名前に関する知識の一部をa priori に与えておく必要がある。この意味から、性質を表わす naming domain は、考えている範囲内の網（intra －network ）で共通であると仮定する。

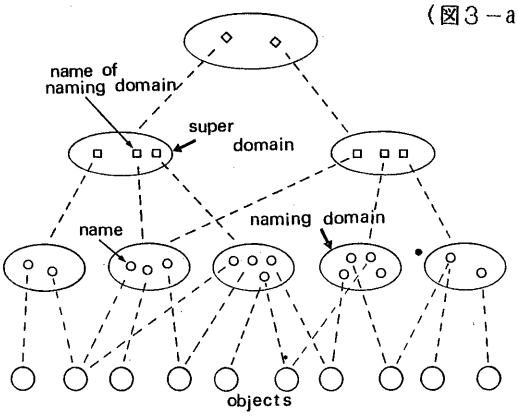
◊脚注： naming domainは名前が一意になる範囲であり必ずしも対象が一意に識別できる範囲ではない。対象を一意に識別する範囲は利用者が区別用に用いる naming domainにおいてである。

３－３ 名前の構造と関係

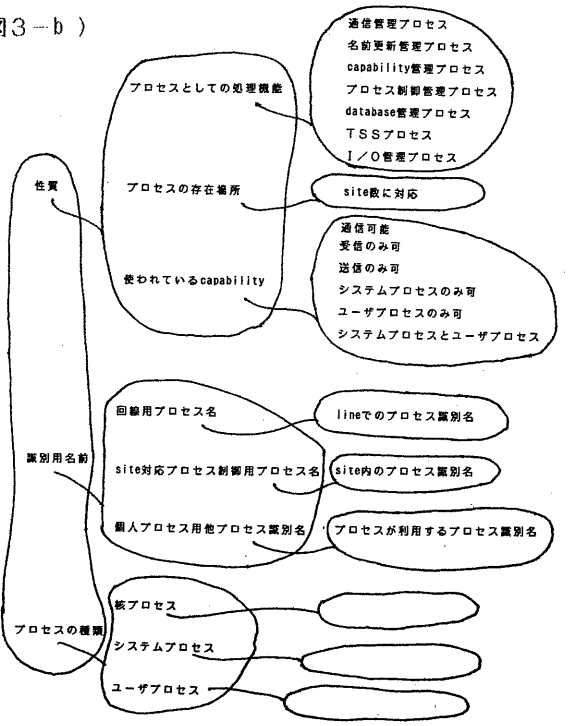
名前の構造について考える。 名前は naming domain を用いることによって分類される。前節までの考察では、naming domain間の関係は明らかになっていなかった。そこで、この節では、naming domain間の関係を、「縦（上下）」と「横（同一レベル）」の面から、調べる。

名前は対象を抽象化して表わしたものである。前節では対象の見方に応じて、分類し、一纏まりにした。そこで、この見方に名前を付け、これらを統合

（図３－a ）



（図３－b ）



する概念によって包括することを考える。即ち、各naming domainに、これを表わす名前を付ける。この名前を共通概念で纏めて、新たなnaming domainを構成する。（図 3）

naming domain の naming domainは、super domainと呼ぶことができる。最下層の naming doaminの要素は対象の名前を表わすinstanceと考えられ、上位に行く程、抽象度が高いと言える。これを言語 smalltalk −80 におけるclass の概念と比較してみると、対象の名前はinstanceに、domainはinheritance を伴なわないclass に似ていると言える。

次に、同じ抽象度のnaming domainの間の関係を調べる。関係には対象を指す名前の間での関係付けと、抽象的な名前間の関係付けとがあるが、先づ前者を考える。

同じ対象に対して複数の見方があるので、それぞれの見方に対応するnaming domainの名前の間には対応関係が付く。対応付けの方法としては、1）考えている網内で一意に区別の付く global name を導入し、global nameと各naming domain の対応する名前との間で関係を付ける方法。global nameの成分としては、［host名＋機能・属性＋同一host内同一機能に対する通し番号（index ）］の形とする。2） 一組の別名の間だけで分かる写像を用いる方法。写像としての中間名を導入し、（一つの名，中間名）の形の組と（中間名，他の名）の組によって名前間の対応を付ける。

方法1は直接global nameと対応付けているので目的とする対象がすぐ分かるが、global nameは長いので、各名前との対応付けは容易でなく、無駄が多い。方法2は、対象は写像を通して間接的に対応が付く。

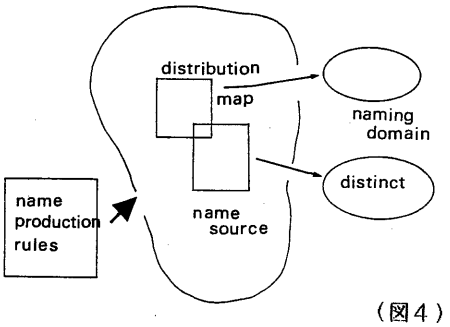
ここでは、簡便さに着目し、主に、方法2による考察を行なうが、必要に応じて、方法1も採り入れる。

抽象化レベルでの名前の間では必ずしも対応が付かないが、特に網間での対応付けは必要に応じて設定できる必要がある。

3−4 名前管理について

これまでは、名前の運用の面を調べた。この節では、名前管理に目を向ける。

名前の管理に必要な要素としては、1） 名前生成規則。各種の名前を発生させる機構。2） 名前の集合。 1）によって発生した名前のプール。3）名前分配表。 2）の名前をどのnaming domainに配ったかを記録する表。（図 4）



（図4）

プロセスの状態の変化に伴なってそれに対する名前も変化するので、名前管理は、静的な面だけでなく、動的な考慮も必要となる。また、異なるnaming domainにおいて同じ形の名前が違う意味で使われることもある。naming domainはその中で一意を保障するものであるから、異なるnaming domainでは同じ名前が使われてもよい。大切なことは、或るnaming domainで要素（名前）が変化した時に如何にして一意を保つかであり、そのための分配表は重要となる。（特に障害の時）

名前の再使用の問題は、は分配表の管理問題に帰着する。

名前の発生規則は用い方によって効率に影響する。処理機能に応じた規則、例えば、文字列やbit 列を用いる場面の選択を如何にするかは重要である。この管理は更新時の管理法のこと等も考慮に入れて、専用の名前管理プロセスを設ける。

分散環境での名前管理では、名前に関する情報を伝えるために、プロセス間通信が必要となる。共通の naming domainは利用者共通の知識と言えるので実際にはコピーを持つ形となる。コピーの方法としては naming domainの中身そのものを持つ方法とnaming domain自体の名前のみを持ち、必要に応じて参照に行く方法とがある。具体的方式は次章で述べる。

4 分散環境での名前と名前管理

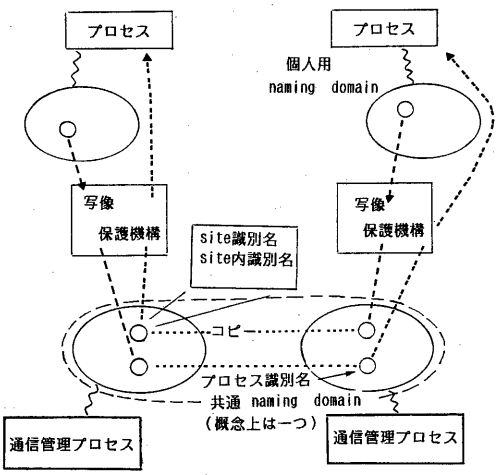
4−1 名前とプロセス間通信

プロセス間通信は、分散環境では唯一の情報伝達手段である。これをサポートする通信管理プロセスは、効率等を考慮して、各hostに置くことにする。

通信管理プロセスは、host内のプロセスから通信要求が出された場合（通信）目的のプロセスがどのhostに存在するか や、host内のどのプロセスかを認識することが問題となる。そこで、通信管理プロセスの間で共通に理解できるプロセス識別名が必要

になる。この識別名は、通信回線上を移動するので、通信用の負担が大きくならず最小限区別できる名前構成が望ましい。　そのためには、host名の識別とhost内での識別ができる形になっていれば十分である。この識別名は必ずしも利用者が使用する形態を採らなくてよい。プロセス識別名全体は、一つのnaming domainとして纏めることができ、これは各通信管理プロセスが共通に利用するnaming domainとして、各々にコピーされる。（図　5）

他の考え方として、通信管理プロセスは自host内のプロセスが通信にかかわる相手プロセスのみを持つ方法もあるが、これは上記の拡張と考えられるので、ここでは一つの網の中で共通なものを用いる場合に限定する。



（図５）

４－２　名前と伝達遅延

分散環境における情報は、或る場所で発生してから他の地点に伝わるまでに必ず伝達遅延を伴なう。情報の受け手ではそれが到着するまでは、古い情報を使うことになり、相互の間に情報の不一致　（mutual inconsistency）が生ずる。この状態の時にこの情報を利用すると不一致による混乱が起こり得るので対策が必要となる。

データのアクセスに対する権利に関しては、保護機構（protection mechanism ・capability）がよく知られている。これを用いることによりアクセスの資格チェックができ、不正な利用を防止できる。

capabilityにはaddress としてのcapabilityと保護としてのcapabilityとの二種類ある。［Fab７４］

保護としての　capabilityの制御法を少し触れる。capabilityは表形式で表わし、meta－operation を用いて操作する方法が一般である。

表形式の作り方には、アクセス行列、C－list方式とA－list方式がある。後の二つの機能は同値であることが知られている。［計セ昭和５１］

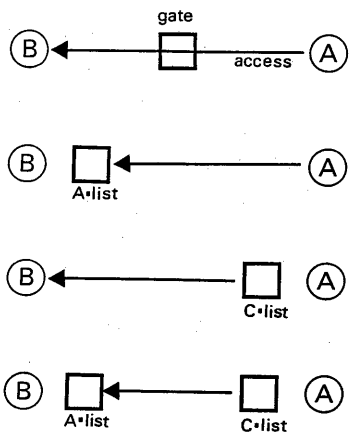
C－listは、　或る対象が利用し得る　相手側のcapabilityを書いた表である。相手を直接調べなくとも、使用の可否がわかるので、処理効率は良い。

A－listは、或る対象が　受け入れ可能な相手のcapabilityを書いた表である。受け入れ側にあるので、常に、正確な情報を保つことができる。

網環境では、誤り処理は、集中的環境における場合よりも難しくなる。誤りを事前に防ぐ方策として、網環境にcapabilityを導入することは、効果がある、と思われる。そこで、以下の方法で、capabilityを導入し、伝送遅延や回線障害にも対処し得る方式を提案する。

この場合、第一に考慮しなければならないことは、capabilityの変更・更新が集中環境ほど容易ではない、という点である。例えば、変更情報を知らせる相手の状態が変化した場合は遅れて伝わる。伝達遅延による情報の食い違いも起こり得る。

伝達遅延の問題を避ける為には、capabilityは、アクセスされる対象のみがA－listの形で持ち、そのコピーを相手に渡さないようにする。こうすることで、capability管理は対象のcapabilityの変化に応じて、確実に行なわれる。或る対象が他のノードにある対象をアクセスする場合、必ず到着ノードの対象の下にあるA－listで確める必要がある。着いた場所でアクセス権が無ければ、ここで拒否され、網上の往復が無駄になる。そこで、事前にアクセスの可否がわかる方法があるとよい。特に、網に出る前に調べる機能があると便利である。その為には、網上での不正操作を予防することも考え合わせて、各ノードに他のノードの相手に対するC－listも置く。（図　6）

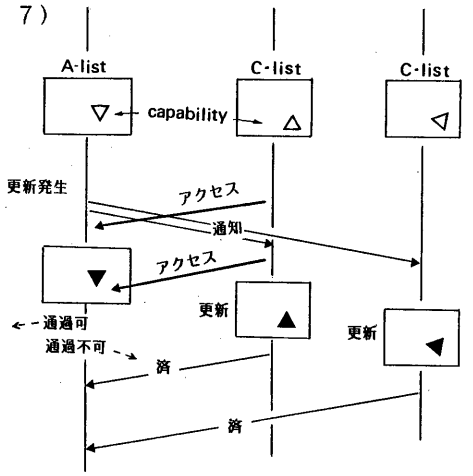


C－listを用いると、コピーを持つことになるので、伝達遅延によるcapabilityの不一致が起こり得る。この場合、A－listとC－listとを併用する方式は、A－list側で対処でき、誤りに陥らない。（C－listのみのチェックに比べて効率は落ちる。C－listのみ使用する場合は、誤りが生ずる。）この頻度はそれ程多くはないと考えられる。

これらのことから、ここではC－listとA－listとを併用する新しい方式を採用する。

具体的には、capabilityは2つの対象の間でのアクセス権を規定するものであるから、C－list とA－list とを関連ある対象間で置く形を採る。

capabilityが変化した時の更新方法は、変更発生箇所のA－listを更新し、続いて、対応C－listのある場所へその旨を通知する。この期間に対象へのアクセスが生じた場合、capabilityの不一致が起こるが、capabilityの変化の仕方によって、アクセス可な場合と不可な場合がある。可な場合はそのまま通過でき不可の場合は拒否することで対応する。（図 7）



C－listの変更が途中で失敗した場合は、capability管理者（具体的には各siteにcapability管理プロセスを置く）にその旨を通知し、原因調査及び回復処理を依頼する。

capabilityの変更中に、情報不一致の為、実際にはアクセスが可能であるのに、できないことも起こり得る（過保護問題）が、この点は許容することにし、むしろ制御の容易さの方を重視する。

C－list変更処理は、ノード対応に、C－list管理プロセスをおき、これらの間で通信を行うことで、実現する。

上記の方式は、伝達遅延にも対処でき、又、回線障害の時も不正なアクセスは防げる。換言すると、伝達遅延による不一致は生ずることがあっても、それがただちに誤りに結び付かないような構成が可能である。複雑なcommitment control を用いなくとも、C－listと A－listとを併用することで、効率良く、誤りも生じにくいシステムが構築できる、と言える。

4－3 名前管理と更新

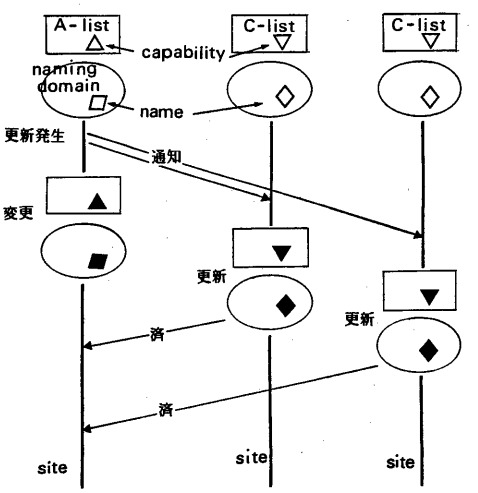
名前管理はnaming domainが基本単位となる。各プロセスは、個人用のnaming domainを専有し、他プロセスと共通に利用しているnaming domainについては、共用する。後者は各プロセスが名前に関する知識を共有し、具体的にはnaming domainをコピーして持つことになる。名前、即ちnaming domainの要素に変更があった場合は、すべてのコピーに知らせる必要があり、名前の更新が問題となる。

更新の種類としてはnaming domain中の名前の消滅、機能変更などによる名前の一部更新、新しく対象が生成された場合naming domainへの登録法、の三種類ある。

更新時の状況は正常時においても伝達遅延は避けられず、時には、回線障害やhostの障害にも考慮を払う必要がある。

以上のことから、一つの対象に対する別名間の写像機構があることを利用して、写像の中に前節で述べた保護機構を組み込み障害にも耐え得る方法を以下のように提案する。

名前の更新方式は、名前管理プロセス、通信管理プロセスとcapability管理プロセスとを活用する。その手順は、1） 更新発生箇所のnaming domainとA－listとを変更し、2） 名前管理プロセス間で連絡し、該当C－listを更新する。3）更新が済んだら発生箇所に返事する。（図 8）



名前の一部更新や名前の消滅の場合は、ほぼこの手順に従う。

新たにプロセスが生成された場合、その旨名前管理プロセスに通知すると、必要に応じて、名前管理プロセス間で連絡しあう。各プロセスはアクセス資格があればそれを個人用のnaming domainに加える。

生成されたプロセスについての情報は通信回数のオーバヘッドを減らすために、この場合はglobal nameの形を用いる。即ち、新に生成された旨の表示、host名、機能、capability、（index）等の要素から構成される。通信管理プロセス間の伝達には、これらのみ分かる名前に変換してもよいが、名前管理プロセスでは、共通naming domainに載っている名前を用いる。個人用のnaming domain には、名前生成規則と名前分配表に従って、識別名を一意性を保つように登録する。

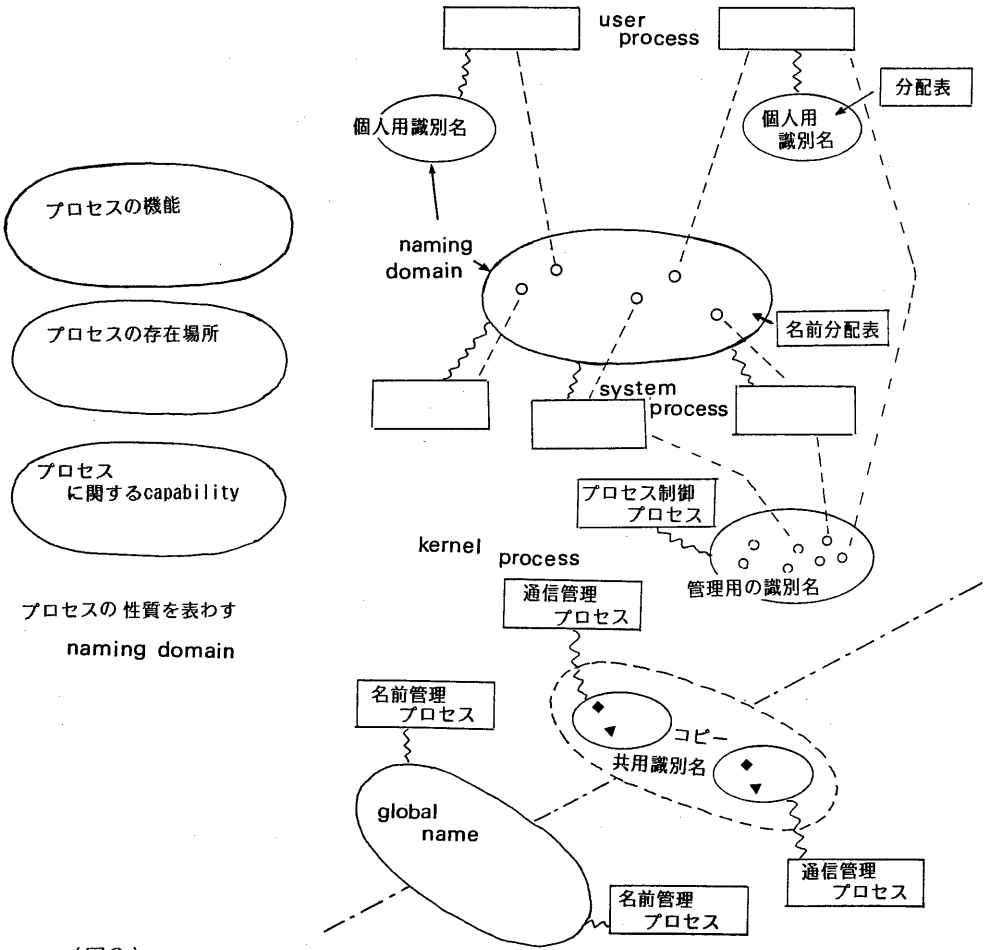
障害等で更新が完了しなかった場合は組み込んであるC－listとA－listとの機能により、不一致箇所の処理を停止する。

この方法の利点は、分散データベースシステムにおける two phase commit 方式［Gray 78］と比較して、1） 厳密な制御が不要なので、処理方式が簡単になる。2） 一部の更新不一致に対しては対処法があるので網全体を犠牲にしなくて済むこと、そのため、処理効率が全体として上がることである。

5 Model System

これまでの考察に基づいて、分散環境における、名前管理の面から見た構成を示す。（図 9）

プロセスの種類は、図3－b のように核プロセス、システムプロセス、ユーザプロセスの三種類を



（図9）

考える。

プロセスは、個人用のnaming domainと、共用のnaming domainとを持つ。名前生成規則と名前分配表は一つのnaming domainの共同の利用者の間で持つ。この管理は複数でする場合と個人でする場合とがある。

核プロセスの中で、管理に関係するプロセスは各々管理のための識別用naming domainを持つ。この利用者の単複は管理の範囲に応じて、異なる。例えば、プロセス制御プロセスはhost内のみの管理なのでこのnaming domainは一人で所有する。通信管理プロセスは、網用の識別名 naming domainを、他hostの通信管理プロセスとの間で共用する。名前管理プロセスはglobal nameを扱う。

システムプロセスは処理用の識別naming domainを持つ。これは個々で持っても、共用してもよい。図の例では、共用した場合を示した。

ユーザプロセスは個々にnaming domainを持つが、プロセスによっては共用することもあり得る。

以上のnaming domainの他に性質を表わすnaming domainは、各々のプロセスで共通に用いる。

6 今後の方針

以上の考察で行なった分散環境での名前の付け方や管理方式に基づいて、障害時のsimulationや具体的設計を行なう予定である。

最後に、本研究は文部省昭和５８年度科学研究費（課題番号５８７５０２７３）を利用して行なっている。

参考文献

R.S.Fabry:Capability-Based Addressing 1974.July CACM Vol.17,No.17.

Gray,J.N.:Notes on database Operating Systems ;Advanced Course Vol.60,1978

Saltzer,J.H.:Naming and Binding of Objects,Operating Systems; Advanced Course Vol.60,1978

ISO/TC 97/SC 16 N1390 March 1983