ネットワーク協調型コンピューティングアーキテクチャ（NCCA）

山下博之† 村主俊彦† 木下真吾†

ワークステーションやディスク等のノードを高速ネットワークにより論理リング状に接続し，サービス/情報要求メッセージをこのネットワーク上を巡回させ，要求を満たしかつ最適なノードに取り込ませることにより，負荷均衡化やフォールトトレランスを実現する，ネットワーク協調型コンピューティングアーキテクチャ（NCCA）は，分散協調システムにおける通信オーバヘッドを削減する．NCCAでは，従来アプリケーション間の通信により実現していた協調処理を，下位レイヤ通信プロトコルに水平かつ垂直にマイグレーションした．この高機能通信プロトコルは，ネットワークを共有メモリとして扱い，その制御により，各ノードが，受信メッセージの内容とノード/システムの状態とに基づき自律的に動作し，分散システム全体の協調を実現する．負荷均衡化では，各ノードが，送受信メッセージの監視により推定する自ノード負荷と，システム全体の負荷の変動に基づき更新する負荷しきい値との比較により，処理要求メッセージの取込み要否を判定する．フォールトトレランスでは，システム内に，共用待機冗長，多数決冗長等の冗長構成の混在が可能である．上記プロトコルをLSI化し，プロトタイプシステムを構築した．

Network-wide Cooperative Computing Architecture (NCCA)

HIROYUKI YAMASHITA,† TOSHIHIKO SUGURI† and SHINGO KINOSHITA†

A network-wide cooperative computing architecture (NCCA) is described that reduces communication-processing overhead in distributed processing systems. Network-wide load balancing and fault tolerance are achieved by interconnecting workstations, disks, and other nodes in a logical ring topology over a high-speed network; by circulating service and information request messages over this network; then having the messages accepted by the most suitable nodes that satisfy the requests. In NCCA, the cooperative processing that used to be executed by communications between applications is horizontally and vertically migrated to a lower-layer communication protocol. By the control of this communication protocol, system-wide cooperation is achieved through autonomous operation of each node. Load balancing is achieved as each node estimates its own load by monitoring received messages, compares that value with an updated load threshold for the overall system, and decides whether to accept processing request messages. A prototype system that includes the protocol processing LSI's is under construction. We intend to apply it to WWW and VOD servers and to evaluate their performance.

1. はじめに

インターネット/イントラネットに代表される分散システムが急速に拡大している．これにともない，分散システム内に存在する，プロセッサやストレージ，サービスや情報等のリソースを，有効利用したいという要求が高まっている．すなわち，所望の時点で，ネットワークに接続されたサーバ群の中から，最も処理負荷の小さいサーバを選んで処理をさせることや，最もアクセス負荷の小さいサーバから情報を取り出すことが望まれている．あるいは，所望のサービスを実現可能なサーバや，所望の情報を保持するサーバを容易に探してアクセスすることが望まれている．このような要求に応えるため，インターネットをコンピューティング・リソースと見なした，インターネット・コンピューティングの研究が盛んになってきている．

分散システムにおいて，このような要求，すなわち負荷分散やフォールトトレランスを実現するためには，ネットワーク内の複数の分散オブジェクトが互いに通信しながら協調して所定の仕事を実現する，分散協調処理が必要である．これらの要求は，特定のノードに制御させることによっても実現可能である．しかし，このような集中制御方式は，障害および過度な負荷に対する耐性の点で，分散制御方式に劣る．

分散協調処理においては，協調のための処理と通信

† NTT 情報通信研究所
NTT Information and Communication Systems Laboratories

処理とが要求される．従来，分散協調処理は，アプリケーションやオペレーティングシステム（OS）間の通信により実現されていた．分散システムの規模の拡大や処理の複雑化にともない，システム全体としての通信処理量が増大する．分散協調処理の効率化のためには，コンピュータ・ハードウェアやネットワークの高速化とともに，この通信オーバヘッドの削減が重要である．

通信オーバヘッド削減という観点からは，従来，軽量プロトコル（lightweight protocol）等の研究・開発が行われてきた．しかし，軽量プロトコルと通常のプロトコルとの性能差は，徐々に小さくなってきている．それは，次の理由による．近年，半導体技術の進展により，CPU 処理能力等が日増しに向上している．その結果，個々の通信プロトコル処理そのものは，見かけ上大きな問題とはならなくなってきている．そして，この傾向は今後も続くことが予想される．さらには，近い将来に，通信プロトコルにより多くの処理を分担させることも可能となろう．

以上のような背景から，我々は，分散協調処理における通信オーバヘッドを削減するための高機能な通信プロトコルを開発した．同プロトコルは，システム全体としての通信処理量の削減を目的として，ネットワークを「共有メモリ」として使用する，という考え方に基づいている．システム全体の通信処理量は，通信プロトコル処理量と通信回数との積和である．共有メモリとしての使用のために，協調のためのアプリケーション処理や上位レイヤ通信プロトコル処理を，各ノードの下位レイヤ通信プロトコル処理に水平かつ垂直にマイグレーションしている．水平マイグレーションは，スケジューリング等の集中制御的機能を各ノードに分散するものである．これにより，分散オブジェクト間の通信回数を削減する．垂直マイグレーションは，アプリケーション，OS や上位レイヤ通信プロトコルの機能を下位レイヤ通信プロトコルで実現するものである．これにより，各ノードにおける（上位レイヤ）通信プロトコル処理量そのものを削減する．

さらに，我々は，上記高機能通信プロトコル（以降，“インテリジェント通信プロトコル（ICP）”と称する）を各ノードあるいはネットワークの中に配置することにより，分散協調処理を効率良く実行する，ネットワーク協調型コンピューティングアーキテクチャ（Network-wide Cooperative Computing Architecture; NCCA）[1] を開発中である．NCCA は，マルチベンダの市販コンピュータやディスクを高速ネットワークに接続したシステムにおいて，コンピュータ間の負荷均衡化，フォールトトレランスを容易に実現する．本論文では，2 章で NCCA の基本概念を述べ，3 章で，ICP について詳述する．4 章および 5 章では，NCCA における負荷均衡化およびフォールトトレランスについてそれぞれ説明し，6 章で NCCA のインプリメンテーションについて述べる．その後 7 章で，スケーラビリティ等に関して考察した後，最後に，8 章で結ぶ．

2. 基本概念

2.1 分散協調処理

分散システムでは，ネットワークに接続されたワークステーション（WS）やディスク等のノードが，互いに協調して動作しながら要求されたサービスの処理を行う．ノード間の協調により，負荷均衡化やフォールトトレランス等を実現できる．これらの協調動作は，各ノード内のアプリケーションが他ノードのアプリケーションとそれぞれ通信し合うことにより行われる．たとえば，サービス要求元の各アプリケーションが，他ノードの性能および負荷情報を問合せ等により収集し，収集情報に基づきサービスを依頼するのに最も適したノードを選定し，そのノード宛にサービス要求メッセージを送出することにより，システムの負荷均衡化を実現する．ところが，分散システムの規模の拡大にともない，このような協調のための通信処理量は飛躍的に増大し，本来のサービス処理量によっては，協調しない方がかえって効率が良いという事態さえ起こりうる．

2.2 協調処理のマイグレーション

NCCA では，協調のためのアプリケーション処理や上位レイヤ通信プロトコル処理を，各ノードに水平かつ垂直にマイグレーションすることにより，分散システムにおける通信オーバヘッドを削減する．マイグレーション前後の通信の比較例を**図 1** に示す．

2.2.1 水平マイグレーション

水平マイグレーションでは，図 1 (a) に示すように，各ノードを論理的にリング（ループ）状のネットワークで接続し，通信メッセージをノード間で巡回させる．これにより，サービス要求元が集中的に実行していた協調処理は各ノードに分散される．各ノードでは，アプリケーションが受信した通信メッセージの内容を参照して自身の処理を決定する．したがって，アプリケーション間の通信回数，すなわち，システム全体としての通信量を小さくできる．

このような形態のシステムでは，サービス/情報要求メッセージがリング状ネットワークを巡回し，要求を

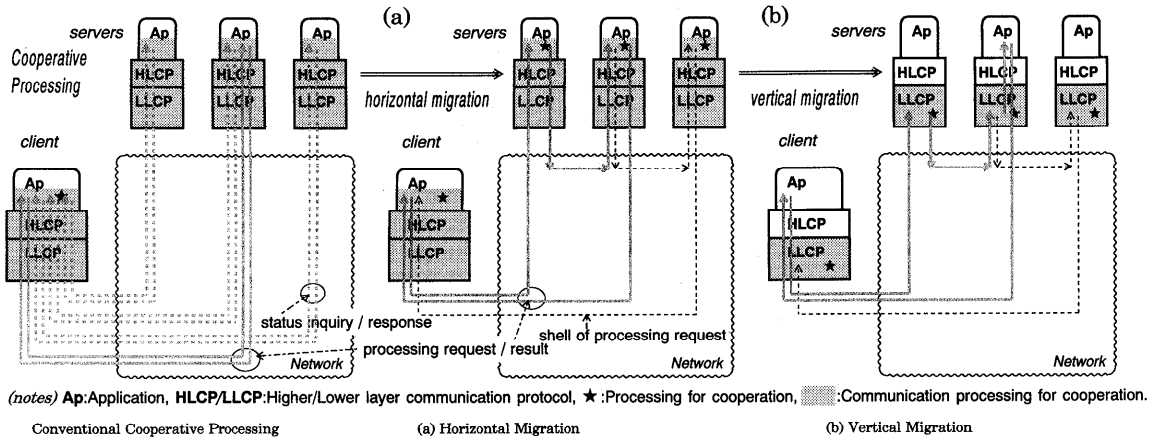


図 1 協調処理のマイグレーション
Fig. 1 Migration of cooperative processing.

満たしかつ最適なノードにより取り込まれる．たとえば，上記の負荷均衡化は，次のようにして実現される．すなわち，サービス要求メッセージを受信した各ノードのアプリケーションは，自ノードの性能と自ノード/システム全体の負荷状況とに基づき，そのサービス要求を処理するか否かを判定する．処理する場合には，受信通信メッセージを取り込むが，処理しない場合には，それをリングの隣接下流ノード宛に再送出する．なお，システム全体の負荷状況は，たとえば，受信する通信メッセージの監視により推定する．このような形態は，各ノードのアプリケーションが自律的に動作することによりシステム全体としての協調を実現することから，“**自律協調**” と呼ばれる．

2.2.2 垂直マイグレーション

垂直マイグレーションでは，図 1 (b) に示すように，ノード内の取込み処理等の協調処理を，アプリケーションから下位レイヤ通信プロトコルに移動する．処理の移動にともない，その処理の参照情報をも移動する．この結果，各ノードでは，アプリケーションまで受信メッセージを届けることなく，下位レイヤで協調処理を実行する．したがって，各ノードにおける上位レイヤ通信プロトコル処理のそのものを削除できる．

以上のようなマイグレーションによって得られた通信プロトコルを用いるシステムにおいては，各ノードが，このネットワーク中を巡回する通信メッセージの内容を参照・更新することにより，自律的に処理を行う．通信回線やスイッチに加え，各ノードの下位レイヤ通信プロトコル処理までをもネットワークと見なせば，これは，ネットワークを「共有メモリ」として使っていることにほかならない．このとき，通信メッセージは，ノード間の通信のための共有制御データということになる．また，論理リングというトポロジは，共有データの排他制御を容易に実現可能とする．

これに対して，従来のアプリケーション間の直接通信による協調処理は，「メッセージパッシング」型の制御と考えることができる．もちろん，物理的な共有メモリをシステム内に接続する形態も考えられる．しかし，この形態においても，アプリケーションと共有メモリ・ノードとの間の通信が必要となるので，「メッセージパッシング」型の制御であることには変わりはない．

2.3 構　　成

ネットワーク協調型コンピューティングアーキテクチャ（NCCA）に基づくシステム構成の一例を，**図 2** に示す．また，各ノードにおけるメッセージの流れを，**図 3** に示す．コンピュータやディスク等のノードは，高速ネットワークにより論理リング状に接続され，サービス/情報要求メッセージがこのネットワーク上を巡回し，ICP の制御により必要かつ最適なノードに取り込まれることにより，負荷均衡化やフォールトトレランスを実現する．図 2 ではクライアントが外部に存在する例が示されているが，論理リング型ネットワークに接続されたコンピュータの 1 つがクライアントとなる場合も可能である．また，図 2 には示されていないが，当該ネットワーク内で所定の時間内に取り込まれないメッセージは，ICP の有するゲートウェイ機能（後述）の利用により，他のネットワークへ転送される．

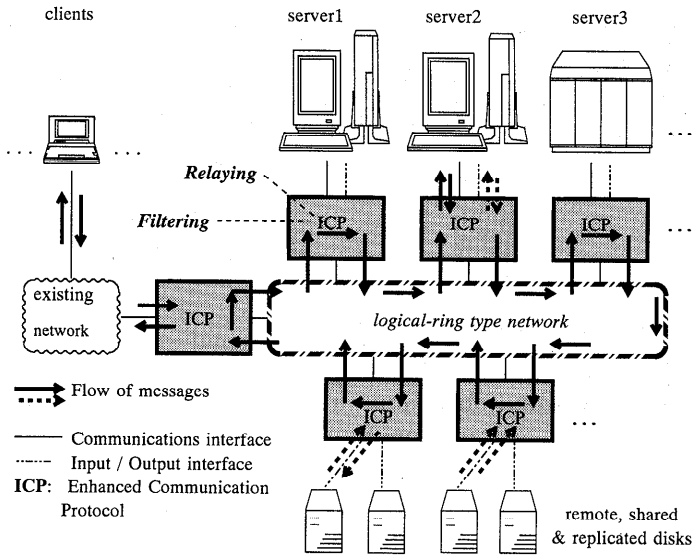


図 2 NCCA システム構成例
Fig. 2 Example of NCCA system.

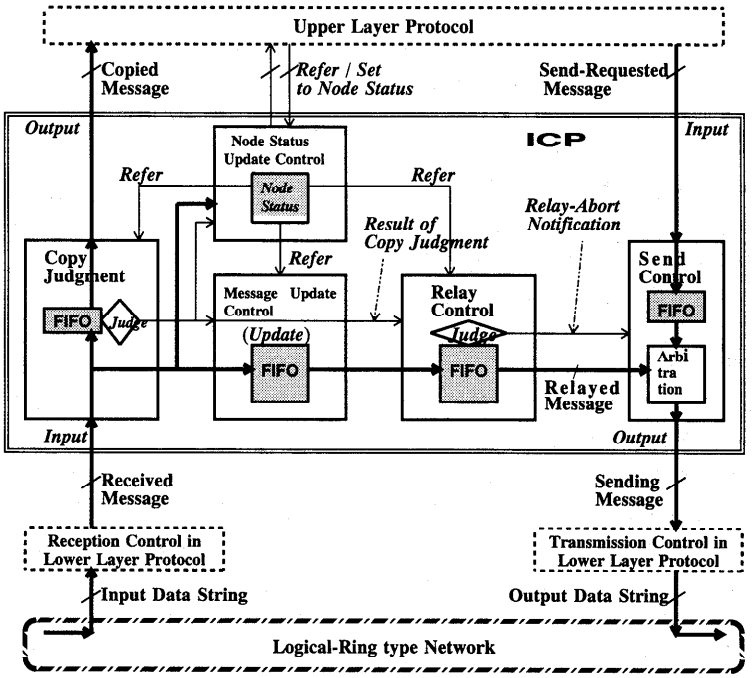


図 3 各ノードにおけるメッセージの流れ
Fig. 3 Message flow in each node.

3. インテリジェント通信プロトコル

2.1 節で述べた分散協調処理の基本的な要素は，システムの状態を知ること，その状態をもとに要求処理の実行ノードを決定すること，および要求処理を実行することである．これらを，各分散ノードが自律的に行う場合には，要求処理を実行可能なノード群のうちで，最適なノードに処理要求メッセージを届けて実行させることが必要となる．言い換えれば，複数のノードに処理要求メッセージを届ける機能とともに，各ノード

で，自ノードによる“実行可能性”や，白ノードでの実行の“最適性”を判定するための機能が要求される．したがって，これらの機能はそのまま，マイグレーションにより得られる通信プロトコルに要求されることになる．さらに，後者の機能は，排他制御を包含しなければならない．なお，要求情報の提供についても，要求処理の実行と同様に取り扱うことができる．

本章では，提案する高機能通信プロトコル（ICP）[2]への要求条件とそれを実現する機能について詳述する．

3.1 通信形態

通信プロトコルの説明に先立ち，分散協調処理を行うために必要な通信形態について述べる．

A. *Broadcast Communication*

全ノードにメッセージを届ける通信形態である．これは，論理リングの構成制御や状態情報の通知等のために必要である．

B. *Multicast Communication*

指定条件を満たすノード群によりグループを形成し，このグループの全メンバ宛にメッセージを届ける通信形態である．これを用いて，各ノードが処理要求の“実行可能性”の判定を行う．

C. *Anycast Communication*[5]

指定グループのメンバのうち，1ノードにのみメッセージを届ける通信形態である．これを用いて，各ノードが自ノードによる実行の“最適性”の判定と実行の排他制御を行う．

D. *K-reliable Communication*[3]

指定グループのメンバのうち，K 個のノードにのみメッセージを届ける通信形態である．これは，上記Cを拡張したものである．

3.2 機能

前節で示した通信形態を実現するために，通信プロトコルに要求される機能を以下に示す．

(1) 機能アドレッシング[4]

これは，通信メッセージが要求するサービスや機能，あるいは情報を提供可能なノードにそのメッセージを取り込ませるためのアドレス方式である．この機能を用いて，*Multicast Communication* を実現する．

メッセージ中の宛先アドレス域に，当該メッセージが要求するサービス，機能や情報を表すアドレス（“**機能アドレス**”）を設定する．各ノードは，当該ノードで提供可能なサービス/機能/情報に対応するアドレスを保持する．ノードの保持する機能アドレスが受信メッセージ中の機能アドレスを包含するならば，そのノードはその受信メッセージを取り込み，要求するサービス/機能を実行あるいは要求情報を提供可能である．機能アドレス形式としては，アドレス域の各ビットに“機能”を対応させたもの，機能をコード化したもの，およびこれらの組合せが可能である．システム全体の機能数，複数機能からなるサービスの有無等のシステム要求に応じ，これらの形式を使い分ける．

(2) メッセージ状態更新

これは，通信メッセージ中に保持するメッセージ状態を，ノードを通過することにより変化するごとに更新するものである．“通過”には，取込みのほかに，単なる中継をも含む．メッセージ状態は，ノード間の協調のための共有情報であり，具体的には，メッセージ中の要求の満たされ具合等がある．この機能を用いて，*Anycast Communication*，*K-reliable Communication* を実現する．

(3) ノード状態更新

これは，ノード内に保持するノード状態を，通信メッセージが通過することにより変化するごとに更新する機能である．ノード状態は，ノードで各種判定を行うための情報であり，具体的には，処理要求メッセージの受け入れにともなう自ノード負荷状態，通過メッセージ量にともなうシステム全体の推定負荷状態等がある．この機能は、次に述べるメッセージ・フィルタリング機能のために用いる．

(4) メッセージ・フィルタリング

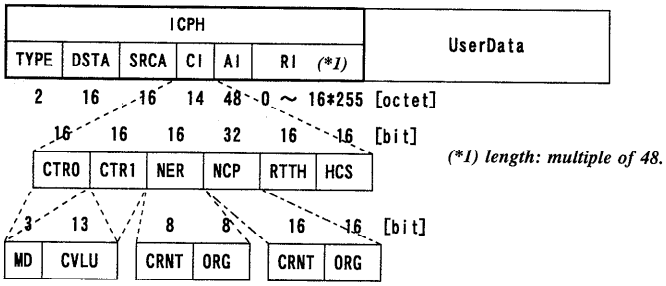
受信メッセージの取込み要否の判定を行い，次の条件をすべて満たす場合にのみ，受信メッセージを取り込み，アプリケーションに引き渡す．

(a) ノードの提供機能が，メッセージの要求機能を含む．
(b) メッセージ状態が，ノードへの取込みを要求している．
(c) ノードの提供クレジットが，メッセージの要求クレジットを満たす．
(d) ノードの負荷が，メッセージの取込みが可能な状態である．
(e) ノードの障害がない．

上記判定に用いるノード情報は，それが変化するごとに，システム管理エンティティからICP処理の保持情報域にダウンロードする．ただし，ノードの負荷状態については，上記(3)の機能により，メッセージの送受信ごとにICP処理内で更新する．

(5) コンディショナル・ストリッピング

これは，各ノードあるいは特定のノードが，受信した通信メッセージが所定の状態となっている場合にのみそれを廃棄し，そうでなければ，それを廃棄しないでさらに中継するというものである．この機能を用い



ICPH: ICP header, TYPE: message type, DSTA/SRCA: destination/source address, CI: control information, AI: additional information, RI: routing information, CTR: counter, MD: mode, CVLU: counter value, NER: execution request, NCP: rotation control, CRNT: current, ORG: original. RTTH: rotation threshold, HCS: header checksum.

図4 ICP メッセージ形式
Fig. 4 ICP message format.

て，*Anycast Communication*，*K-reliable Communication* を実現する．

3.3 メッセージ形式

前節に述べた機能を実現するための，ICP のメッセージ形式を図4に示す．ICP 処理では，刻々と受信するメッセージのヘッダ域の内容と保持するノード状態情報とにより，その受信メッセージの取込み要否，ヘッダ域の更新要否，ノード状態情報の更新要否，および受信メッセージの中継要否をリアルタイムに判定する．

物理層に ATM 網を想定し，各ノードにおけるメッセージ中継遅延を1セル時間程度とするため，プロトコル・ヘッダを 48 オクテットの倍数とするとともに，更新対象域を先頭の 48 オクテット内に配置した．また，本プロトコルで使用する機能アドレスを IPv6[6] におけるマルチキャストアドレスの体系に含めることを想定し，アドレス長を 16 オクテットとした．

主なフィールドの意味は，次のとおりである．

(1) 宛先/発信元アドレス（DSTA/SRCA）

メッセージの宛先/発信元のアドレス．宛先アドレスには，前述の機能アドレスに加えて，ブロードキャストアドレスやユニキャストアドレスを設定可能である．

(2) メッセージ種別（TYPE）

メッセージの種別，取込み抑止指示，ストリッピング時の取込み指示，論理リングを経由しない直接転送の表示，コネクション型グループアドレスの登録/抹消指示等の制御を表すフラグ類の集合．なお，直接転送は，処理結果情報の返却時のようにメッセージの宛先が確定している場合に使用する．これは，ネットワーク内に論理的にリングを形成するシステムにおいて，ネットワーク内トラヒック量の削減と，メッセージ伝送時間の縮小に有効である．

(3) 制御情報（CI）

アドレスフィールド（DSTA, SRCA）とともに，メッセージの取込み要否および中継要否を決定するための情報を含む．また，メッセージおよびノード状態の更新要否を決定するための情報を含む．さらに，中継に際しての更新対象となる．

(3-1) カウンタ 0 ～ 1（CTR0 ～ 1）

用途に対応する更新モード（MD）とカウンタ値（CVLU）とに分けられる汎用のフィールド．たとえば，後述のノード保持情報である保持カウンタ (HCTR) と組み合わせて，次の用途に用いる．

- 論理リング接続ノード数の計数と，各ノードへの通番の付与．
- メッセージ・フィルタリング機能における3番目の条件判定のための，要求クレジット値の保持．
- 入札における，現在の最大（または最小）応札値の保持．

(3-2) 実行要求（NER）

現（CRNT）と元（ORG）との2つのサブフィールドに分けられる．現サブフィールドはこれまでに当該メッセージを取り込んだノード数を表し，メッセージ取込み時に更新される．また，現サブフィールドの値が，取込み要求ノード数を表す元サブフィールドの値より小さいときに限り，このメッセージが取込み可能である．これは，先に述べたメッセージ・フィルタリング機能における，2番目の条件に該当する．なお，(ORG) = 0 は，可能な全ノードへの取込みを表す．

このフィールドを用いて，*Anycast Communication* および *K-reliable Communication* を実現する．すなわち，発信元は，NER.CRNT = 0, NER.ORG =

1 または K，としたメッセージを送信すればよい．メッセージを取り込んだ各ノードは，メッセージ中の (NER.CRNT) をインクリメントし，その値が (NER.ORG) に達しない限り，リングの下流にそのメッセージを中継する．

(3-3) 巡回制御（NCP）

現（CRNT）と元（ORG）との2つのサブフィールドに分けられ，それぞれ，メッセージの論理リング巡回済数および最大許容巡回数を表す．このフィールドは，通信メッセージの発信元ノードにより参照・更新され，先に述べたコンディショナル・ストリッピング機能に使用される．すなわち，巡回済数が最大許容巡回数を超えていれば，そのメッセージを論理リングから除去する．また，他の論理リング網へのメッセージ転送を行うゲートウェイ・ノードによっても，転送要否の判定のために参照される．

(3-4) 巡回しきい値（RTTH）

上記 NCP，後述のノード保持情報であるゲートウェイしきい値（RLTH）と組み合わせて，メッセージの他ネットワーク宛への転送のための取込み要否の判定に使用する．

(3-5) ヘッダチェックサム（HCS）

ヘッダ（DSTA/SRCA,TYPE,CI）域の誤り制御コード．ヘッダ域の誤りはシステム全体の協調動作に影響を与えうるため，誤ったメッセージはリングから除去する．なお，メッセージが除去されたことは，その発信元による監視により検出可能である．

(4) 付加情報（AI）

システム制御等のために，ICP エンティティが参照したり，上位エンティティが設定・参照する，次の付加的な情報を含む．

- ノードの保持するグループアドレスリストに登録，またはリストから削除するグループアドレス．
- 下記 RI 域に設定されているルーティングアドレス数．
- 上位プロトコル種別．
- メッセージ識別番号．
- タイムスタンプ．
- オリジナル・メッセージ中のアドレス待避データ．

(5) ルーティング情報（RI）

メッセージのネットワーク間ゲートウェイ転送のためのアドレス情報（可変長）．

3.4 ノード保持情報

各ノードの ICP 処理エンティティは，協調処理のために，ノードの状態や負荷均衡化制御情報等を保持する．これらの情報は，上位エンティティから設定/参照されるとともに，一部は，受信メッセージの取込み時および上位エンティティからの要求に基づくメッセージ送信時に，必要に応じ更新される．以下に，主な情報を示す．

(1) 自ノードアドレス（ADIA）

自ノードの個別アドレス．

(2) 提供サービス機能アドレス（NDFA）

自ノードの提供サービスを表す．たとえば，本アドレス中の '1' であるビットの位置に対応する番号のサービスを，当該ノードが提供していることを表す．

(3) 提供情報機能アドレス（IOFA）

自ノードの保持情報を表す．たとえば，本アドレス中の '1' であるビットの位置に対応する番号の情報を，当該ノードが保持していることを表す．

(4) グループアドレスリスト（GAL）

当該ノードが属するグループの識別アドレス群を表す．メッセージの取込み時，メッセージ中の TYPE 域に "グループアドレス登録" 指示があれば，次メッセージの受信に先立って，AI 域中の指定グループアドレスを本アドレスリストに追加する．また，TYPE 域に "グループアドレス抹消" 指示があれば，メッセージの取込みとは無関係に，次メッセージの受信に先立って，指定グループアドレスを本アドレスリストから抹消する．また，上位エンティティからの指示により，指定グループアドレスを本アドレスリストに登録，あるいは本アドレスリストから削除する．

これは，マルチキャスト・コネクションの設定/解放に用いる．すなわち，ある時点の *Multicast Communication* によりメッセージを取り込んだ際に，当該メッセージ中に設定されていたグループアドレス（GA）をそのノードの GAL に一時的に登録しておき，以降の通信においてその GA を用いることにより，先のメッセージを取り込んだノード群のみがそのメッセージを取り込むこととなる．

(5) ノード障害フラグ（NDFL）

当該ノードが故障等のためにサービスを提供できないことを表す．このフラグは，ノード状態が変化するごとに，システム管理エンティティ等からオン/オフされる．これは，先に述べたメッセージ・フィルタリング機能における，5 番目の条件の判定に使用する．

(6) 取込み抑止フラグ（CS）

このフラグがオンのとき，受信メッセージを取り込まない．これは，リングの上流ノードが連続して多数の処理要求メッセージを取り込むことにより，ノード間の負荷に著しい不均衡が生ずることを防ぐために使用する．

(7) ノード負荷（NDLD）
ノードの負荷状態を表す．この値に対し，特定の条件で受信メッセージが取り込まれると，対応するノード負荷加算値の分だけ加算される．また，メッセージ送信要求時の指示により指定値だけ減算される．
(8) ノード負荷加算値（LDDF）
受信メッセージを取り込んだ場合の，ノード負荷の加算値を表す．
(9) 負荷しきい値（LDTH）
ノードの負荷状態の限界を表す．メッセージ受信時，上記ノード負荷と比較され，この値がノード負荷より大きい場合にのみ，そのメッセージが取り込み可能である．これは，先に述べたメッセージ・フィルタリング機能における，4番目の条件に該当する．

また，メッセージのNER域に保持される取込みノード数に関する要求を満足せずにリングを1周してきた処理要求メッセージについて，その発信元ICPは，そのメッセージ中のNCP.CRNT域を更新して中継する．このフィールドの値が0でないメッセージの受信頻度により，各ノードのICPは，システム全体の相対的負荷の増減を推定し，負荷しきい値を更新する．
(10) しきい値変分（THDF）
負荷しきい値を更新する際の，加算および減算値を表す．
(11) 保持カウンタ（HCTR0 ～ 1）
受信メッセージ中の同一番号のカウンタフィールドの値との比較，その結果に基づく更新等を行う．たとえば，用途に応じ，次のような情報を保持する．

- メッセージの通過により設定された，リング内における自ノードの通番．
- メッセージ・フィルタリングのための自ノードの提供クレジット値．
- 入札における，自ノードの応札値．

(12) ゲートウェイしきい値（RLTH）
ICPのゲートウェイ機能が受信メッセージを取り込む場合の，メッセージ中のNCP.CRNTフィールドの値の限界値を保持する．NCP.CRNTフィールドの値が，本値とメッセージ中のRTTHフィールドの値との和以上であれば，他ネットワークへの転送のために当該メッセージを取り込む．これにより，先に述べたコンディショナル・ストリッピング機能を実現する．

4. 負荷均衡化

NCCAでは，各ノードが相互に情報交換することなく，自律的に処理要求メッセージの取込みを制御することにより，システム全体の負荷の均衡化を図る．

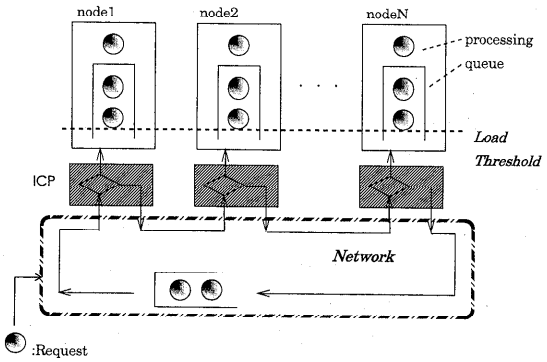


図5 NCCAにおける負荷均衡化の概念
Fig. 5 Load balancing scheme in NCCA.

その概念を図5に示す．

4.1 アルゴリズム

各ノードのICPは，処理要求メッセージ受信時，その保持する自ノードの負荷状態と負荷しきい値との比較により，当該メッセージを自ノードでの処理のために取り込むか，他ノードに処理を任せるために中継するかを判定する．

自ノードの負荷状態は，入力した処理要求メッセージ数と出力した処理結果メッセージ数との差あるいはアクティブなコネクションの数により推定する．また，OSから入手して設定することも可能である．負荷しきい値は，システム全体の相対的な負荷状況に基づき算出し，その変動に応じて更新する．

システム全体の負荷がノード間で均衡状態にあるとき，性能 P_k を有するノード k における理想的なしきい値は負荷状態と一致し，次式で与えられる．

$$\left(\sum h_i / \sum P_j\right) * P_k \qquad (1)$$

ここで　h_i ：処理要求 i のサービス時間，
P_j ：ノード j の性能，
$\sum$ ：システム全体における総和

上式から分かるように，システム全体の負荷状況は，処理要求の入出力にともなう処理量の増減に加えて，システム内のノードの故障/復旧や追加/削除/変更にともなうシステム全体の性能の増減によっても変動する，相対的なものである．NCCAにおける負荷均衡化では，システム全体の負荷状況そのものを用いることなく，送受信するメッセージの監視により均衡状態からの相対的な負荷変動量を推定し，それに応じてしきい値を更新する．このしきい値の増減幅（しきい値変分）は，ノード性能に比例する．各ノードにおける負荷均衡化制御のイメージを図6に示す．

どのノードにも取り込まれずに論理リングを1周し

てきた処理要求メッセージについて，その発信元ICPは，そのメッセージ中のNCP.CRNT域を更新して中継する．このNCP.CRNT域の値が0でないメッセージの受信頻度により，各ノードのICPは，システム全体の相対的負荷の増減を推定する．また，取込み抑止フラグ（CS）を用いて，均衡化の微調整を行う．

4.2 性能評価

NCCAにおける負荷均衡化の効果を，シミュレーションにより確認した．詳細は別途報告予定であるが，ここでは，その一例を紹介する．

図7は，文献7)で評価されている，処理要求を他ノードに移送する負荷均衡化方法との平均応答時間の比較を表すシミュレーション結果である．比較条件は文献7)と同じであり，10 Mbit/sのネットワークに接続された性能均一の40ノードのシステムにおける，処理要求呼の到着時間およびサービス時間が指数分布のサービスの場合である．また，NCCAにおける総リング長は200 mである．

図7より，文献7)で評価されているreceiver initiated方式に対しては，システム負荷のほぼ全領域にわたり，NCCAの方式が優れていることがいえる．また，adaptive symmetrically-initiated方式に対しては，システム全体が高負荷の領域において，NCCAの方式が優れていることがいえる．なお，adaptive symmetrically-initiated方式とは，高負荷のとき時にreceiver-initiatedポリシーを用い，低負荷のときにsender-initiatedポリシーを用いるsymmetrically-initiated方式に対して，ある改良を施したものである．

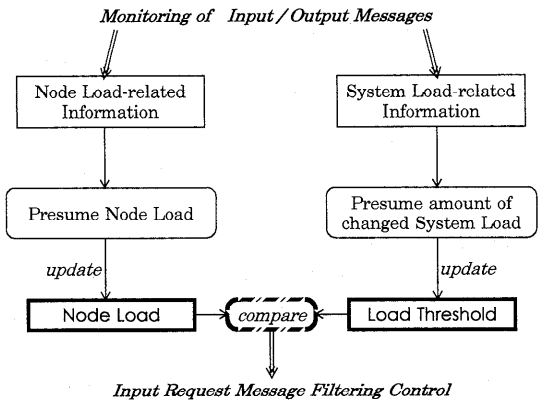


図6 各ノードにおける負荷均衡化制御
Fig. 6 Load balancing control in each node.

5. フォールトトレランス

5.1 冗長構成

3章に示した種々の通信形態の利用により，処理ごとに次の任意の（仮想的な）冗長構成が可能であり，しかも，それらがシステム内で混在する．

(1) 共用待機冗長

NCCAにおける共用待機冗長構成のイメージを**図8**に示す．受信した処理要求メッセージに対し，そのノードが故障中であれば，それを取り込まずに下流ノードに中継する．ノードごとに特定の処理を実行する場合，実行可能な処理に対応する機能アドレスを設定しておく．加えて，待機ノード（Activeであってもよい）を1つ用意し，これにはすべての処理に対応する機能アドレスを設定しておく．これにより，いずれのノードが故障した場合にも，処理要求メッセージは共通の待機ノードが取り込み，要求処理を実行する．処理実行のためのデータを格納したディスクは，ノード間で共

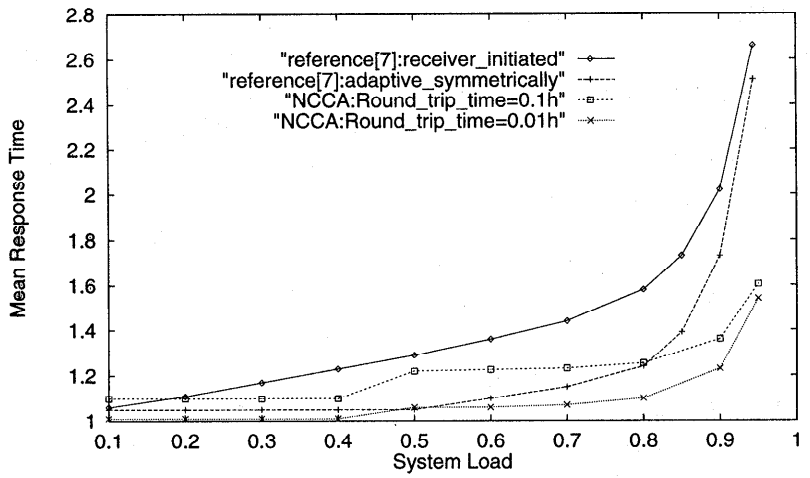


図7 負荷均衡化方式の比較評価例
Fig. 7 Comparison example of load balancing evaluation.

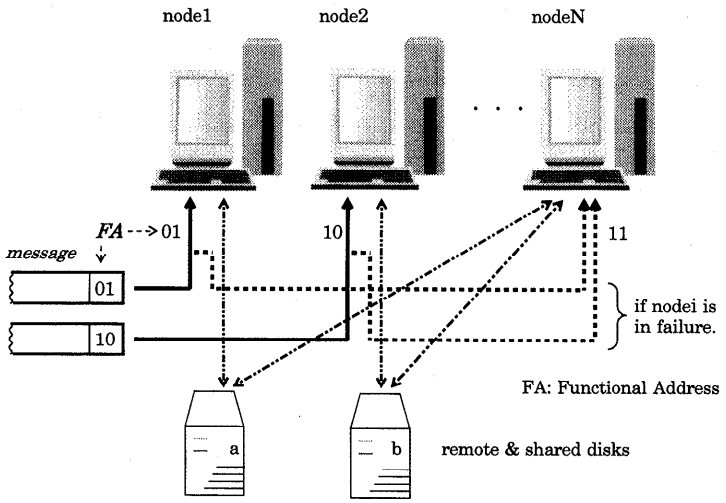


図 8 共用待機冗長構成
Fig. 8 Shared stand-by configuration.

用するので，どのノードからもアクセス可能である．

(2) 多数決冗長

処理結果やデータの信頼性向上のため，複数ノードでの処理実行結果や複数ディスクからの読み出し結果の多数決をとって要求元に返却する．NCCA における多数決冗長構成のイメージを図 9 に示す．特定のノード（要求元でもよい）に，複数の結果から多数決により 1 つを選択する Voting 機能を設け，それに対応する機能アドレス（FA）を設定しておく．要求メッセージは，*K-reliable communication* を用いて複数ノードに届ける．処理実行結果やディスク読み出し結果は，宛先アドレスとして Voting 機能を表す FA を設定したメッセージにより転送し，前記の Voting 機能を有するノードが取り込む．Voting 機能では，一定時間に到着しない結果を異常として扱うことにより，応答時間の増大を防ぐ．

なお，Voting アルゴリズムとして，多数決の代わりに先着順を採用することにより，並列冗長構成の実現も可能である．この構成では，結果の信頼性は向上しないものの，応答時間を短縮可能である．

(3) 複製化ディスク

NCCA におけるディスクへのアクセスは，ファイル名やディスクアドレスに対応させた機能アドレス（FA）をディスクノードに設定しておく．このとき，同一の FA を複数のノードに設定することにより，複製化ディスクを実現する．すなわち，*Multicast Communication* を用いて複数のディスクに同一データを書き込み，*Anycast Communication* を用いて，1 ディスクから読み出しデータを得る．あるいは，上記 (2) に示したように，多数決により読み出しデータの信頼性の向上が可能である．さらに，ディスクアクセス処理中のノードがディスク読み出し要求メッセージの取込みを抑止して中継することにより，他の無動作ディスクからデータを読み出し，アクセス時間の短縮が可能となる．

5.2 機能評価

前節に示したフォールトトレランスの各構成について，プリプロトタイプシステムを用いて機能の正当性を確認した．プリプロトタイプシステムは，複数のパーソナルコンピュータ（PC）を Ethernet で接続したものである．ICP は PC 上のプログラムにより実現し，単一の Ethernet 上で，プログラム制御により論理リングを形成した．このシステムで，図 8 および図 9 に示す各冗長構成において，擬似的な故障/エラーを発生させても，TCP/UDP プロトコルによるクライアント/サーバ型の分散処理が矛盾なく実行されることを確認した．なお，Voting アルゴリズムとしては，多数決，先行 2 一致，先着等について確認した．ディスクアクセスについては，SCSI コマンドをカプセル化して転送している．

6. インプリメンテーション

6.1 バリエーション

ICP のインプリメント形態としては，各ノードのネットワークインタフェースコントローラ（NIC）内に組み込む形態，付加装置に組み込んで通信や I/O インタフェースで接続する形態，スイッチやハブ等のネットワーク内機能として具備する形態等，種々考え

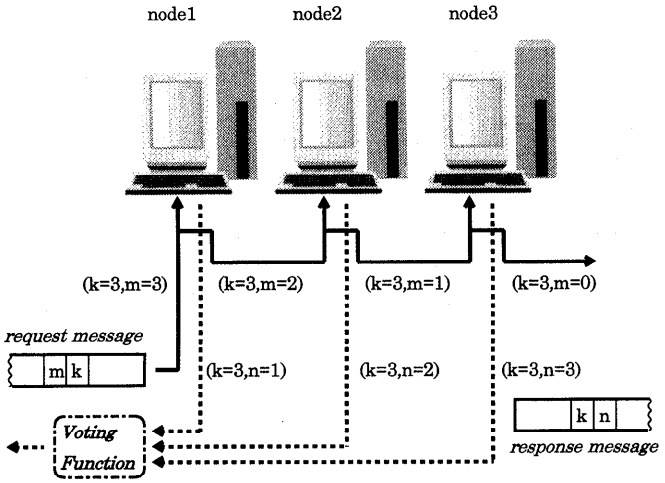


図 9 多重モジュール冗長構成
Fig. 9 Multiple modular redundancy configuration.

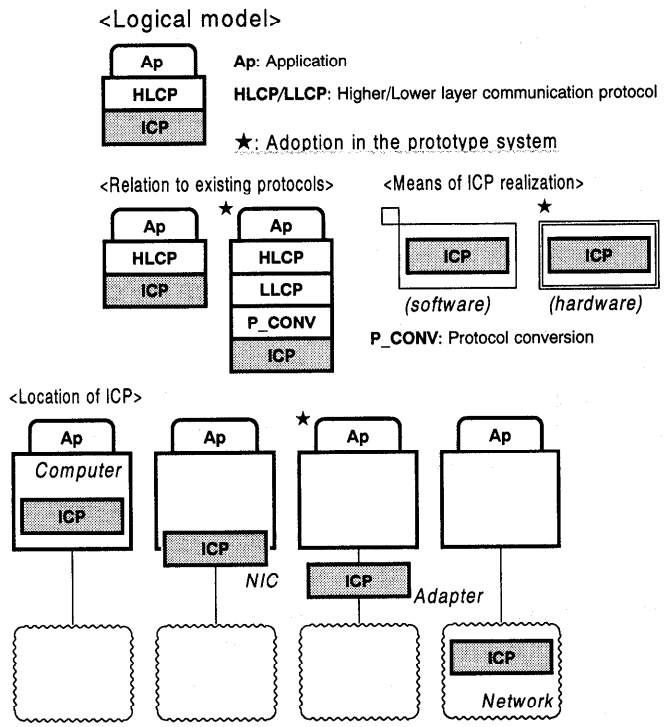


図 10 ICP インプリメンテーションのバリエーション
Fig. 10 Variation of the ICP implementation.

られる．インプリメンテーションのバリエーションを**図 10** に示す．

共有物理メディア上あるいはスイッチ網内に論理リングを形成するインプリメンテーションでは，宛先のユニークな処理結果メッセージ等の転送に，リング状にノード間を巡回しない直接転送を用いる．これにより，応答時間の短縮とネットワーク使用率の削減とが可能となる．

6.2 プロトタイプ

我々は，現在，前節で示したバリエーションのうち，ハードウェア化した ICP を埋め込んだ装置（アダプタ）を各ノードに接続する形態のプロトタイプシステムを構築中である．

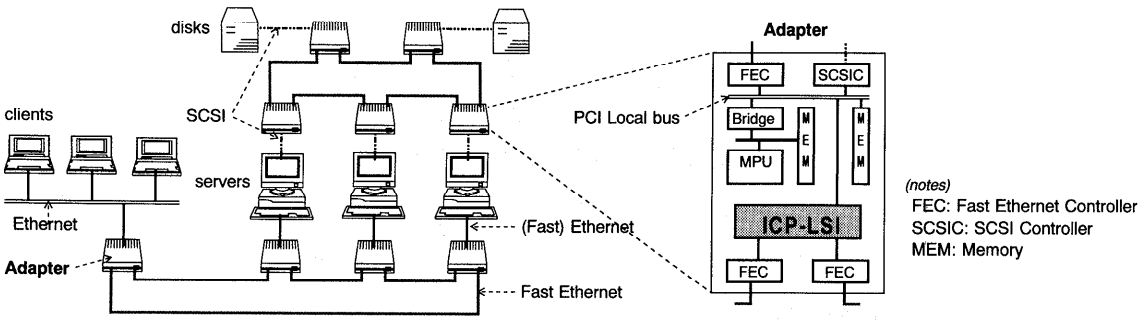


図 11 NCCA プロトタイプシステムの構成
Fig. 11 NCCA prototype system configuration.

6.2.1 システム構成

プロトタイプシステムの構成を図 **11** に示す．Fast Ethernet（100 Mbit/s）により隣接アダプタ間を接続し，物理リングを形成する．各アダプタを Fast Ethernet のシェアードハブやスイッチングハブに接続し，論理リングを形成することも可能である．各アダプタには，送受信 2 個の Fast Ethernet コントローラ（DECchip21140），ICP 処理用 ASIC，プロトコル変換処理ファームウェアが走行する MPU（SPARClite），バッファメモリ（4 MB），コンピュータとの接続のための Fast Ethernet コントローラ，コンピュータおよびディスクとの接続のための SCSI コントローラ（MB86605）が搭載され，これらは PCI ローカルバスで相互に接続されている．

本プロトタイプのインプリメンテーションでは，コンピュータ/ディスクとは，標準インタフェースで接続し，アダプタ上でプロトコル変換を行うため，コンピュータのアプリケーションや OS の変更を必要としない．また，マルチベンダ・ノードの使用が可能である．クライアントからのサービス要求メッセージは，入口のアダプタによってカプセル化されて論理リングに送出され，あるアダプタにより取り込まれると，カプセルから取り出されて IP アドレスが変換された後，接続コンピュータに転送される．負荷均衡化制御に用いるノード負荷は，前述のように，推定により求める．フォールトトレランスに用いるノード障害情報は，アダプタからの定期的なヘルスチェックにより検出する．なお，各種アドレス情報等は，特別の管理ノードから，あらかじめあるいは必要となった時点でアダプタに設定/更新する．

6.2.2 ICP-LSI

各ノードにおけるメッセージ中継遅延短縮のため，ICP 処理を LSI 化した [8]．

本 LSI では，受信メッセージを FIFO に転送しながら，必要なフィールドを横取りする形で参照し，メッセージの取込み，更新および中継の判定を “**on-the-fly**” 処理で実現する．さらに，中継判定結果が確定すると，メッセージ全体の受信が完了していなくても，送信出力を開始する **Cut-Through** 方式を採用した．最大中継遅延時間は，ICP-LSI で約 2.5 μs，両 Fast Ethernet コントローラを含めると約 12 μs である．なお，ヘッダチェックサムはサポートしていない．

7. 考　　察

7.1 ICP のプロトコルレイヤ

ICP は，各ノードにメッセージを届ける必要があるため，各ノードを認識可能なプロトコルレイヤに実装する必要がある．ただし，各隣接ノード間を伝送路で接続した物理リング形態では，メッセージの宛先は固定されるため，レイヤ 1 に実装することもできる。スイッチやバス上に論理的にリングを形成する形態では，その規模に応じて，ICP の実装可能なレイヤが異なる．ルータ等により他と接続されていない単一セグメントの LAN の場合には，MAC アドレスによりユニークにノードを識別可能であるので，レイヤ 2（MAC サブレイヤ）内あるいはその直上に実装可能である．複数セグメントの LAN や WAN の場合には，IP アドレスによりノードが識別されるため，レイヤ 3（IP レイヤ）への実装となる．

以上の考察により，いかなる形態にも共通に使用可能な実装レイヤは，IP レイヤということになる．ICP におけるアドレスは，IP レイヤへの実装を考慮して規定した．

7.2 アーキテクチャの拡張

(1) スケーラビリティ

NCCA が前提とするリング型ネットワークでは，接

続ノード数に比例してメッセージの伝送遅延時間が増大する．これは，メッセージ伝送において，各ノードにおける中継遅延が積算されるためである．

この問題の解決策は，ICPのLSI化のほかに，リングを比較的小さなグループに分割し，それらを階層的に接続することである．これら小さなリング間のメッセージ伝送は，ゲートウェイノードにより，3章で示したICPのコンディショナル・ストリッピング機能を用いて制御可能である．我々は，実験環境でこの形態について機能的に確認済みである．この実験では，上位エンティティによりメッセージのルーティングに関する制御を行った．

(2) リングの信頼性

リング型ネットワークは，また，1ノードの障害がシステム全体に致命的な影響を与えるという問題がある．これについては，次のように解決する．

物理リングの場合には，各ノードの入口にバイパス・スイッチを挿入することにより解決できる．バイパス・スイッチは，当該ノードの障害を検出すると，メッセージをそのノードに送らずに隣接下流ノードに転送するように制御する．前節で述べたプロトタイプシステムでは，このバイパス・スイッチを用いてシステムの信頼性の低下を防いでいる．

一方，論理リングの場合には，リングの再構成により解決できる．たとえば，定期的に論理リングの正常性をチェックし，異常検出時に，障害ノードの特定とそのノードの論理リングからの離脱を行う．また，必要に応じ，メッセージ伝送のリカバリを行う．これらは，ICPとは異なるシステム管理エンティティの制御により行うことになる．

(3) ネットワークの帯域

NCCAでは，システム全体の性能に関し，ネットワークの帯域が重要なポイントとなる．これについては，協調制御用にICPを用い，その結果として宛先ノードが確定すると，処理用データをそのノード宛に直接転送するという使い方をすることにより，帯域の有効利用が可能である．

7.3 関連する研究開発

(1) サーバネット[9]

プロセッサ群と周辺装置群とを，複数の，“ルータ”と呼ぶ高速のデータ交換機構で相互接続し，Any-to-Anyの直接接続を行うことにより，スケーラブルな高信頼システムの実現を可能としたものである．ただし，負荷分散の機能は有していない．装置間で専用パケットを交換することにより制御する．現在，マイクロソフト社等と共同で，PCIバス用インタフェースカードとWindows環境向けの標準API（Wolfpack）の開発が進められている．

(2) LocalDirector[10]

Ethernet接続された複数のWWWサーバに対し，外部からのアクセスを振り分ける機能を有するルータ．負荷分散が可能．ルータ上のプログラムによる集中制御により実現し，スケーラビリティはない．現在，製品が市販されている．

(3) I_2O（Intelligent Input Output）[11]

業界団体で開発中の，OSとデバイス・ドライバ間の標準インタフェース．1つのデバイス・ドライバを異なるOS間で共通に使えることを狙う．ホスト・プロセッサの介在なしに，周辺装置間でのピア・ツー・ピア通信も可能である．NCCAにこのインタフェースを適用することにより，ディスク読み出しデータを直接要求元に転送することが可能となる．さらに，ディスクのフォーマットもOSに依存しない形で標準化されれば，一層のマルチベンダ化が可能となる．

8. おわりに

アプリケーションやOSに具備していた分散協調機能を各ノードの下位レイヤ通信プロトコルに水平・垂直にマイグレーションし，各ノードが自律的に動作することにより分散協調処理の効率化を図った，ネットワーク協調型コンピューティングアーキテクチャ（NCCA）について述べた．NCCAでは，インテリジェント通信プロトコル（ICP）の有する各種のMulticast通信形態等を用いて，ノード間の協調のための通信オーバヘッドを削減し，システムの負荷均衡化やフォールトトレランスを効率的に実現する．

さらに，このプロトコルをLSI化し，同LSIを組み込んだプロトタイプシステムを構築中である．今後，同システムをWWWやVODのサーバ等に適用し[12]，これらについて評価する予定である．

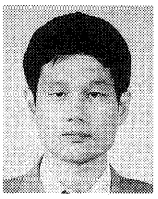
謝辞 本研究を進めるにあたり，長期にわたってご指導いただいた和佐野哲男NTT情報通信研究所長，山下正秀NTTマルチメディアシステム総合研究所研究企画部長に深謝します．また，熱心に討論いただいた上司・同僚に感謝します．

参考文献

1) Yamashita, H., Suguri, T., Kinoshita, S. and Okada, Y.: Network-wide Cooperative Computing Architecture (NCCA), *Communication and Architectural Support for Network-Based Parallel Computing* (Proc. Workshop

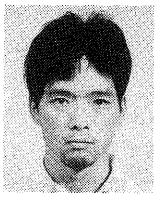
on CANPC'97, San Antonio, Texas, Feb. 1–2, 1997), Panda, D.K. and Stunkel, C.B. (Ed.), LNCS Vol.1199, pp.241–255, Springer-Verlag, Berlin (1997).
2) Yamashita, H., Suguri, T. and Kinoshita, S.: An Enhanced Communication Protocol for Flexible Interconnection, *Proc. 22nd Conference on Local Computer Networks (LCN '97)*, Minneapolis, Minnesota, Nov. 2–5 (1997).
3) Bound, J. and Roque, P.: IPv6 Anycasting Service: Minimum requirements for end nodes, Internet-Draft (1996).
4) Kandlur, D.D. and Shin, K.G.: Reliable Broadcast Algorithms for HARTS, *ACM Trans. Computer Systems*, Vol.9, No.4, pp.374–398 (1991).
5) IEEE Standards for Local Area Networks: Token Ring Access Method and Physical Layer Specification, IEEE Std 802.5-1989 (1989).
6) Deering, S. and Hinden, R.: Internet Protocol, Version 6 (IPv6) Specification, RFC1883 (1995).
7) Krueger, P. and Shivaratri, N.G.: Adaptive Location Policies for Global Scheduling, *IEEE Trans. SE.*, Vol.20, No.6, pp.432–444 (1994).
8) 山下博之, 村主俊彦, 木下真吾, 岡田靖史, 小栗清, 塩川鎮雄：分散協調通信プロトコル処理ASIC設計におけるメッセージ通過遅延最小化法, 信学論(D-I), Vol.J80-D-I, No.9, pp.745–762 (1997).
9) http://www.tandem.co.jp/technology/servernet/index.html.
10) http://www.cisco.com/warp/public/751/lodir/index.html.
11) 解説：周辺装置のデバイス・ドライバ, 異なるOSで利用可能に, 日経エレクトロニクス, No.687, pp.141–146 (1997).
12) 木下真吾, 山下博之：セションリダイレクト機構を用いたエニーキャストゲートウェイの提案と実装, 第55回情報処理学会全国大会論文集, 6T-3, p.3-725 (1997).

(平成9年 5 月 13 日受付)
(平成9年 10 月 1 日採録)



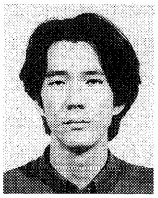
山下 博之（正会員）

1957年生. 1979年京都大学工学部情報工学科卒業. 1981年同大大学院修士課程情報工学専攻修了. 同年日本電信電話公社（現NTT）に入社. 以来, DIPS通信制御装置および通信系ファームウェアの研究実用化に従事. 1991年より, フォールトトレラントコンピューティング, ネットワークワイドのコンピューティングアーキテクチャ, 高速通信プロトコル処理等の研究に従事. 現在, 研究開発推進部担当部長. 電子情報通信学会, 応用物理学会, IEEE各会員.



村主 俊彦

1967年生. 1990年早稲田大学理工学部電子通信工学科卒業. 1992年同大大学院修士課程電気工学専攻修了. 同年NTTに入社. 以来, 負荷均衡化アルゴリズム, モバイルコンピューティング, インターネット商取引システムの研究に従事. 現在, 情報通信研究所研究主任. 電子情報通信学会会員.



木下 真吾（正会員）

1968年生. 1991年大阪大学基礎工学部物性物理工学科卒業. 同年NTTに入社. 以来, フォールトトレランス, 分散アルゴリズム, 通信プロトコルの研究に従事. 現在, 情報通信研究所勤務. IEEE会員.