ノードの移動性を実現するモビリティ技術

高杉耕一[†] 片山穣[†]
久保田稔[†] 小柳恵一[†]

通信ノード（端末）の移動をともなった場合においても，サービスの連続性（通信切替え時にデータの欠損や通信手順をやり直すことなく連続的に行うこと）を保証する技術について提案する．ノードの移動においては従来，サービスを要求するクライアントの移動のみを考慮していたが，将来のモバイルコンピューティング環境においては，サービスを提供するサーバの移動を含めた，移動性のサポートが必須である．そこで，サーバの移動を含め，すべてのノードが移動性を有するようなネットワークにおいてもノードの移動性を確保する技術とノードの効率的な位置管理方式に関して提案する．また，プロトタイプ，計算機シミュレーションによる評価を報告する．評価の結果，短い切断時間でデータの欠損なしに，ネットワークデバイスを変更することが可能であることと，更新コストを削減したノードの位置管理が行えることを確認した．

Technology for Node Migration

Koichi Takasugi,[†] Minoru Katayama,[†] Minoru Kubota[†]
and Keiichi Koyanagi[†]

This paper proposes technology for guaranteeing continuity of service when a communication node moves and changes network device or network address. So far, only movement of the client has been considered. In the mobile computing environment in the future, however, support for mobility that includes movement of the server will be indispensable. Therefore, we propose a movement concealing system where all nodes including the server have mobility and an efficient location administration system. We report on an evaluation of a prototype and computer simulations. The prototype evaluation showed that it was possible to change network device at a short interruption time without losing any data and to perform the location administration that reduced the update cost.

1. まえがき

近年，端末の小型化，無線技術の進展にともない，情報家電，ロボット等さまざまなものがノード（PC, PDA に代表される通信装置をともなう端末）となりネットワークのさまざまな接続点に動的に接続されるようになってきている．まさに，ユービキタスコンピューティング[1] の実現が間近にせまってきている．

一方，ユーザが直接ネットワークに接続するアクセス系において，有線においては光ケーブル，無線においては IMT-2000 の導入により，広域かつ高帯域の通信環境が構築されようとしている．それにともない，音声，動画像をはじめとする高帯域通信において，サービスを中断することなく，利用可能な最適通信媒体を適宜選択し利用することがますます重要となる．

異種媒体で構成されるネットワークの例を**図 1** に示す．本論文ではこのようなネットワークにおいて，移動ノードが使用する媒体（有線 LAN，無線 LAN, PHS）を切り替えたり，同一媒体を用い異なるネットワーク接続点に切り替えたりする（ネットワークアドレスを変更する）際，サービスの連続性を保証する技術について議論，提案する．

サービスの連続性はアプリケーション間でやりとりされる通信手順をデータの欠損が発生したり，通信を最初からやり直したりすることなく，連続的に行うことである．また，ノードの移動性はノードが移動し，通信媒体やネットワークアドレスを変更した場合においても，サービスの連続性を保証できることである．

これまでに，我々はサービスの連続性を保証するノードの移動性の実現方式として，MEDLAR（Mobile Environment and Distributed Resource Collaboration by Mobile Agents）を提案してきた[2),3)]．ここでは，クライアントアプリケーションとサーバアプリケー

† 未来ねっと研究所
NTT Network Innovation Laboratories

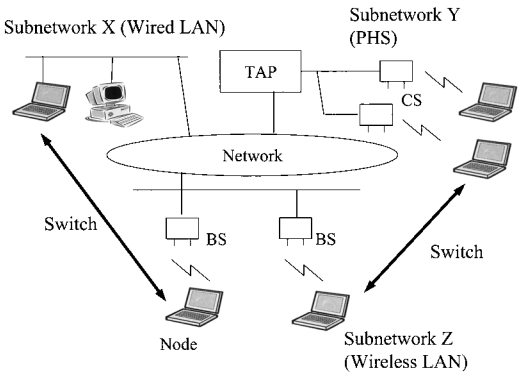

図 1 想定ネットワーク
Fig. 1 Network image.

ションの間に送信するデータの蓄積，転送を行うプロキシを 2 つ挟み，プロキシ間での通信媒体の変化や切断，再接続をプロキシで隠蔽することで，ノードの移動性を実現してきた．このとき，プロキシはアプリケーションに対して，あたかもつねに接続状態であるように振る舞うことで，サービスの連続性を保つことが可能である．ここでは，クライアントアプリケーションが存在するクライアントノード側の移動性だけに注目してきたため，サーバアプリケーションが存在するサーバノードは移動しないことを前提とし，サーバノードとクライアントノードには異なる専用の機構を持ったプロキシを想定してきた．

ところが，近年のノードの小型化，多様化，センシング技術の進展にみられるように，サーバノードが容易に移動できる環境が整いつつある．また，本来，クライアント，サーバはプロセス，スレッドレベルのものであり，ノード内に共存可能であるから，クライアント，サーバが同一ノード内に混在しているといった状況も考えられる．そこで，サーバ，クライアントの区別なく，すべてのノードが移動可能な枠組みを作ることが重要である[4]．

サーバノードとクライアントノードの双方を移動させるためのアプローチとして，(a) 強力な別の固定ノードを設置し，サーバノードの情報を固定ノードに送ることにより，サービスを提供する集中的アプローチ，(b) 対等なノード関係のもと，さまざまなノードで提供されるサービスを別のノードから直接利用するといった分散的なアプローチの 2 つが考えられる．(a) では中央に固定かつ強力なリソースを持つ蓄積ノードを介して，サービスを提供しているため安定性がある．しかし，蓄積ノードが必要であるし，蓄積ノードを経由するため，冗長なルートを通る可能性が高くなる等，柔軟性を犠牲にしている．

今後のユービキタスコンピューティングでは動的に接続形態やネットワーク構成が変化し，これに柔軟に適応していく必要がある．この場合，数万のノードを対象とするような大規模なサービスではなく，数十，数百単位の小規模でアドホック的なサービスが対象となる．ところが，(a) では，大規模サービスに対し，安定的，効率的なサービス提供ができるが，固定ノードを必ずしも前提としない小規模のサービス形態においては，固定ノードが利用できなかったり，利用できる固定ノードが非効率であったりする可能性がある．したがって，ノード間で直接通信を行う (b) に基づくサービスの形態のほうが望ましい．

そこで，本論文では分散的なアプローチをとることとし，各ノードが対等な立場で，通信するモデルを想定する．ここで対等とは一方的に管理する，されるといった従属関係でないことをいう．本論文では，すべてのノードの移動性を実現する技術について提案し，プロトタイプを用いて HTTP による動画像転送の評価実験を行い，その実現性を明らかにする．また，対等なノード関係のもと，ノードの移動性を確保するためには新たな位置管理が必要である．そこで，再接続時に再接続先のアドレスを検索可能な位置管理方法を提案し，プロトタイプによる検証と計算機シミュレーションによる位置情報更新効率の評価を行う．

2. 要求条件および解決すべき課題

ノードの移動に関する要求条件を以下に示す．

(1) すべてのノードが移動性を有している（サーバアプリケーションを搭載したノードも移動可能）．
(2) 通信しあう 2 つのノードがネットワークから切断されても，再接続時に再接続先のアドレスを検索可能な位置管理機能を備えている．

ノードの移動には瞬時に切替えが行われるリンク切替えと急な切断からの回復をするリンク断（ネットワーク離脱状態）からのリンク回復の 2 種類がある．

ノードの移動性を実現する技術に関しては，サーバも移動等に対応するため，従来技術[2] では実現されていなかった，下記の課題を新規に検討する．

(課題 a) 同時に切替え要求がきた場合の順序制御
(課題 b) 切断時間の短縮
(課題 c) 切断時のデータ欠損の防止
(課題 d) リンク断からのリンク回復 (再接続) の処理
(課題 e) 再接続の際のノードの認証

また，位置管理はすべてのノードが移動可能であるという状況に対応するために新たに検討する必要がある．

この際，位置情報を各ノードで管理する必要があるが，すべてのノードですべてのノードの位置情報を保持し，これを変更ごとにすべてを更新すると更新コストが増大するという問題がある．そこで，位置管理に関する以下の課題を検討する．

(課題 f) 更新コストの削減（位置検索成功率が大きく劣化しない範囲で位置情報の更新における通信コストを削減する）．

3. 関連研究

ノードの移動，位置管理のそれぞれに関し，関連研究との位置付けを示し，本提案の特徴を明らかにする．

3.1 ノードの移動性を実現する技術

本論文ではアプリケーション間の通信にプロキシを挿入することにより，移動性を実現する．つまり，TCP/IP より上位の層でノードの移動性を実現する．

情報発信 Toolkit [5] は移動ノードからの情報発信において，切断時に通信を回復する機能を有する．しかし，異なる通信媒体を変更してサービスを継続するものではない．

Mobile IP [6] はクライアント，サーバの両ノードが移動性を持つシステムに IP 層で IP の一意性を保ち移動性をサポートする．また，TCP 層での実現方式[7] も提案されている．しかし，これらの方式は，ノードがネットワークから離脱（リンク断）した場合にアプリケーション間のサービスの連続性を損なう可能性がある．さらに，TCP/IP 層での実装は，OS ごとの実装を必要とする．さらに，Mobile IP では移動先のノードに転送するために固定ノード（ホームエージェント）の設置が必要であるとともに，固定ノードによる転送を行うため，冗長ルートを通る可能性がある．また，異種媒体間の切替えを Mobile IP で実現するためには通信回線を構築したり，複数の通信可能な通信媒体の中でどの媒体を利用するか等，IP より上位の層で解決すべき問題は残る．

本提案による移動性の実現ではプロキシによって IP の変更を吸収するため，IP アドレスは一意性を保つ必要はない．また，LAN，電話網等 IP における送受が可能なすべての媒体で，移動性をサポートすることが可能である．さらに，Mobile IP 等の移動性でサポートしきれていない移動（リンク断からのリンク回復）等においてもサービスの連続性を実現できる．また，TCP 層の上位に存在するため，OS を選ばず，本システムを即座に導入することが可能である．

半面，Mobile IP のようにすべての IP 上のアプリケーションに即座に対処できるものではない．つまり，TCP 層の上位にあるプロトコルごと（HTTP，FTP 等）の実装を必要とする．しかしながら，TCP 層の上位プロトコルは数種類（HTTP，FTP 等）が支配的である．また，これらのプロトコルが異なってもプロキシの移動性をサポートする部分は共通に利用することが可能である．つまり，プロトコルごとに実装すべき部分はアプリケーションからの情報を取得し，MEDLAR の機構に流す部分と，MEDLAR の機構からの情報を適切なアプリケーションに転送するといったアプリケーションへのインタフェースに関する部分のみである．そのため，プロトコルごとの実装は容易に実現できる．また，プロトコルごとに最適なコンテンツのキャッシングや変換処理（QoS に応じた間引き等）を行うことも可能である．

3.2 位置管理方法

位置管理を行う仕組みとして，WIND project の INS [8]，Jini [9] のような Look-up サービスは，固定ノードの存在を前提としている．

また，階層型の名前サービス[10] のような移動ノード向けの位置管理機構もある．しかし，DNS のようにすべてのノードを階層的に管理するトップダウン的なアプローチをとっているため，すべてのノードを 1 つの位置管理機構の下に管理する必要があり，管理コストが高い．

本論文で提案する方式は，ノードの移動性を制約することなく，かつ容易に位置管理できる．そのため，それぞれのノードで全体の位置管理を分担していく手法をとっている．

4. 提案方式の構成

本章では，提案する MEDLAR の各機能とノードの移動と位置管理における課題の解決方法を述べる．

4.1 ネットワーク構成

本論文で想定する各構成要素間の関係について述べる．MEDLAR では，アプリケーション間にプロキシを挟んだセッションを構築することで，サービスの連続性を実現する．まず，セッションを構築した例を**図 2** に示す．セッションはサーバアプリケーションとクライアントアプリケーションを接続してサービスを提供する単位である．コネクションはアプリケーションとプロキシ，またはプロキシ間に設定される通信路である．また，セッションはアプリケーション間の通信路でいくつかのコネクションから構成される．コネクションの構築には TCP/IP を用いる．

4.2 ノードの移動性を実現する技術

本節では各移動ノードが持つ構成要素について説明

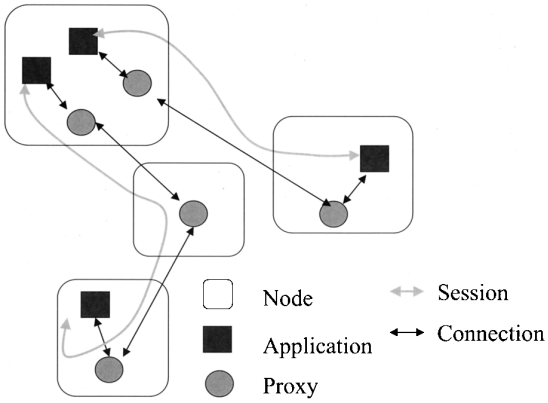

図 2 各構成要素間の関係
Fig. 2 Relationship among components.

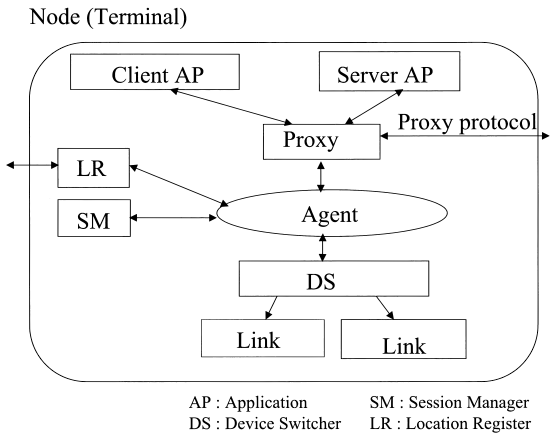

図 3 ノードにおける構成要素
Fig. 3 Functional composition in a node.

する．**図 3** にノードの構成要素を示す．

(1) ノードの構成要素

プロキシは通信の蓄積転送を行いコネクション下のリンクの切断，変更を隠蔽する．DS（Device Switcher）は新しいリンクへの接続，リンクの切断，ルーティング管理を行う．エージェントは，DS に対しリンク切替えを命令するとともに，リンク切替え後，プロキシにコネクションの再開処理を依頼する．SM（Session Manager）は，ノードの移動をサポートするため，セッションに関係する各種機能（プロキシ，アプリケーションの接続関係，プロキシで保持している情報）を管理することにより，セッションの管理を行う．LR（Location Register）はノードの位置の管理を行う．SM と LR は，全体の制御，判断を行うエージェントを介して接続されている．

(2) プロキシプロトコル

プロキシプロトコルはプロキシ間でのデータの送受，通信状態の監視（QoS 測定等）や通信再開の制御を行うことで，プロキシ間のリンクの変化に対応し，サービスの連続性を保証する．

プロキシ間は以下のような内容を設定したフレームを用い MEDLAR プロトコルによって通信する．

- 制御コマンド（通信開始の connect，通信再開の reconnect 等）
- 制御パラメータ（ack 番号等）
- セッション ID
- シーケンス番号（送信フレーム数）
- 総データ量（通信予定のデータ量があらかじめ既知である場合設定）
- 通信済データ量（現時点までやりとりしたデータ量）
- データ（プロキシで転送するデータ）

4.3 LR

位置管理を行う LR は各ノードに配置される．これは自ノードの LR を検索し，各ノードのアドレスを知ることに加え，自ノードの LR で検索に失敗し，他ノードの LR を検索する際，この検索の問合せをかけるノードのアドレスを知る必要があるからである．

現在の固定ネットワークでは IP アドレスでノード識別していたが，IP アドレスは 1 つのノードに対して，1 つ以上存在し，しかも動的に変化するため，新たに，ノードを一意に識別するノード識別子が必要となる．そして，このノード識別子を用いた検索要求に対し，LR は該当ノードの IP アドレスを返す．

このように，LR は主にノード識別子と IP アドレスのマッピングを行うが，位置管理の更新コストの削減と，リソースの少ないノードにおいても実現可能とするため，ノードの移動量と位置情報の保持可能容量の管理も行う．また，管理するノードの範囲限定するために，ノードのグルーピングを行う．

管理すべき主な属性と機能を以下に示す．属性はグループに所属する任意の複数ノードに対して作成されるテーブルとなる．

(1) 属性

- ノードの識別子
- 所属しているグループの識別子（複数）
- IP アドレス（複数）とその状態（利用可否）
- 移動量（移動，固定，半固定）
- 各ノードの LR の位置情報保持可能容量
- 更新時刻

(2) 機能

- IP アドレスの状態変更（接続中，未接続，障害）

- エージェントに変更を通知
- グループの参加，離脱

すべてのノードに対し一意性を保証したノード識別子を付ける必要がある．このような，ノード識別子を付与するには，デバイスの MAC アドレスや CPU の ID を利用する等の方法がある．

また，グループ識別子はノード識別子の一意性が保証されれば，グループ内の親ノードのノード識別子をもとに生成することができる．ただし，親ノード内で他グループの識別子と重ならないように管理する必要がある．

しかし，これらの識別子の一意性を保証する技術については適切なものを利用するとし，本論文では，識別子の一意性は保証されるものと仮定し議論する．

5. 提案方式の実現

ここでは，2 章で示した課題の解決方法について述べる．

5.1 ノードの移動性の実現

プロキシは通信の蓄積転送とともに，セッションを識別するデータ（セッション ID）の管理を行う．したがって，プロキシ間の通信方式の状態が変化してリンク（プロキシ間の通信）が切断された場合においても，リンク接続処理の後，セッション ID を用いて再接続することにより，従来の通信を再開し，コネクションを維持することが可能である．

このように，プロキシを介してセッションを構築することにより，各ノードは自由にリンクを切り替えることが可能となる[4]．また，その際，アプリケーションに通信デバイスやネットワークの変化を意識させない．たとえば，HTTP に適用した場合，Web ブラウザは 2 つ以上のプロキシを介して，サーバに接続される．したがって，Web サーバから動画等のデータが流れている間に，プロキシ間の通信デバイスを切り替えても，Web ブラウザからは何の変化もないように見える．また，プロキシ間の通信が切断されても，Web ブラウザ，Web サーバはそれぞれプロキシと接続している．このため，セッションを保持することが可能で，プロキシ間の通信の再開にあわせて，データの送受信を再開することが可能である．

従来の MEDLAR ではアプリケーションによって異なる専用のプロキシを用いていたが，提案モデルではクライアントアプリケーション，サーバアプリケーションで同一のプロキシを共有できる．また，プロキシ間に新たなプロキシをはさむようなモデルにも対応できる．それにより，両端のプロキシが移動する際に中間のプロキシを一時的に固定しておいて，安定性を確保するといったことも可能である．

次に 2 章で述べた，ノードの移動に関する課題の解決方法について述べる．

(課題 a) リンク切替え時に下記の問題がある．リンク切替えの場合でも，トランスポート層以下の通信を切断し，再接続する必要がある．そのため同時に切替えが生じると両方のプロキシから切断の命令が発信され，両方のプロキシから再接続の命令がくる．このように両方のプロキシからのリンク切替えを同時にすると同期やアドレスの通知等，処理は複雑になり処理時間の増加を招く，そのため，あらかじめどちらのプロキシの処理が優先されるかを決めておくことにより切替え要求実行の順序制御を行う．たとえば，クライアント側のプロキシを優先するとしておけば，競合した場合，サーバ側のプロキシからの再接続要求を無視することで，クライアント側のリンク切替えは達成できる．このときサーバ側のプロキシはリンク切替えが失敗に終わったことが認識できるため，再度，切替え処理を行うことで，サーバ側の切替えも完了する．

(課題 b) 切替え時の切断時間の短縮を行う．切替えの際，一番時間を要するのはルーチングテーブルの更新である．そのため，古い TCP ソケット（以下ソケットと呼ぶ）をクローズしてから，ルーチングテーブルの変更を行うと切断時間が増加してしまう．そこで，あらかじめ切替え先通信デバイスの確認，ルーチングテーブルの更新をしてから古いソケットをクローズし，新しいソケットをオープンする．この場合，ルーチングテーブルの更新が完了した時点では，上りと下りの通信で別の経路を通ることになる．

(課題 c) 切断時のデータ欠損の防止について述べる．リンク切替えはソケット通信の切断，再接続とルーチングテーブルのデフォルトルート変更によって行われる．このとき，ソケットの切断，再接続は必須である．これは，TCP/IP はホップバイホップの全二重通信なので，逆方向の IP パケットは切替え元のルートを通り，経路が変更されないからである．

通信中のソケットの切断にともなって，データの欠損が起こる．そこで，ソケットに書き込むノードで OS やデバイスのバッファ，ネットワーク上にあるデータ等，切断により欠損すると思われる量のデータを送信後にも保持しておき，再接続のとき未送信のものを再送することにより，データの欠損を防ぐ．

他の方法として，write している側からの切断，ハーフクローズといった方法もあるが，同時にソケットを read，write している場合や，突然のリンク断に対処

することができない．

(課題 d) リンク回復に関して，リンク断したノードがリンク回復（再接続）の処理を行う．これはリンク断したほうのノードが再び通信デバイスがネットワークに接続されたことを容易に認識できるためである．再接続するアドレスが分かれば，逆方向から，ポーリングすることも可能あるが，リンク断したノードはデバイスの状態を調べれば，ネットワークに接続したことが分かるので再接続のタイミングをより的確にとらえられる．また，両方のノードがリンク断した場合においても，リンク断したノードの通信デバイスがネットワークに接続したときにリンク回復を行うようにしておけば，両方のノードがネットワークに接続した時点でリンク回復の処理が完了する．

(課題 e) リンク切替えおよびリンク回復の際に行われる再接続時に，もとのノードと同一であるか認証する必要がある．そのため，最初にプロキシ間で接続したときに，認証鍵の交換をしておく（今回は実装していない）．

5.2 位置管理方法

4.3 節で述べた LR を用いて 2 章に示した位置管理に関する課題を解決方法について述べる．

(課題 f) 通信を行うノード間でグループを形成し，各ノードで管理する位置情報の数を限定する．これにより，不必要なノードに関する位置情報を管理する等，LR の容量の増加とそれにともなう更新コストの増加を防ぐ．グループは複数のサブネットをまたがり，各ノードは複数のグループに属することができる．

また，更新情報を送信する優先順位を導入し，すべてのノードへの位置情報更新命令の送信を制限する．これにより，更新にかかる通信コストの削減が可能である．

グループの構成や更新にかかる通信コストをできるだけおさえるために，位置管理方法として以下のような動作を規定する．

(a) グループ加入時

グループに加入しようとするノードは IP アドレスを取得し，グループ検索要求をサブネットワーク内にブロードキャストするか，既知のノードに対して送信する．その後，グループ検索要求を受け取ったノードからのグループ検索応答を取得し，加入するグループを選択する．その後，選択したグループに関するグループ検索応答を送信したノードに対し登録要求を送信し，登録要求を送信したノードからそのグループの位置情報を取得することでグループ加入を完了する．また，グループ登録要求受け取ったノードは，LR に記載されているこのグループに関する位置情報を送信するとともに加入したノードの位置情報を LR の位置情報に付け加えることでグループの加入登録を完了する．

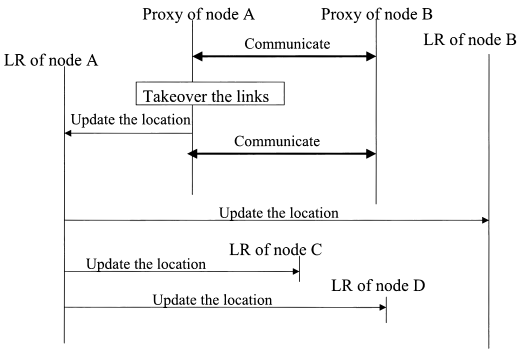

図 4 ノード移動時の位置情報更新命令
Fig. 4 Location update instructions.

(b) ノード検索時

自ノードで保持している LR の位置情報を検索する．検索できなかった場合は LR に記述されている，任意のノードに検索要求をかける．検索結果が返ってきたら，LR に付け加える．

(c) リンク切替え，リンク回復時（位置情報の更新動作）

切替えにともない保持している LR の位置情報を更新すると同時に通信相手がいれば位置情報更新命令を出す．また，移動後，LR に記載されている任意のノードに位置情報更新命令（Update the location）を出す（**図 4**）．

(d) 位置情報更新命令の優先順位

また，(c) において，以下のように位置情報更新命令を送信する宛先のノードに優先順位をつける．

- ノードの移動量が少ないノード
- ノードの位置情報を保持できる容量の高いノード

以上のように振る舞うことで，同じグループに所属しているノードに対して位置検索を行うことが可能となる．

逆にいうと，所属しているグループ外のノードに対する検索方法は有していないが，グループ内のノードで他のグループに所属しているノードが存在すれば，そのノードにグループ検索要求をかけることにより，別のグループにも加入し，新たに加入したグループのノードに対して位置検索を行うことが可能となる．

また，グループ加入時にグループ検索要求を送信する際，送信先の既知のノードが移動してしまう可能性があるので，既知のノードを複数用意したり，ある時間の経過後に再検索したり，一時的に既知のノードを

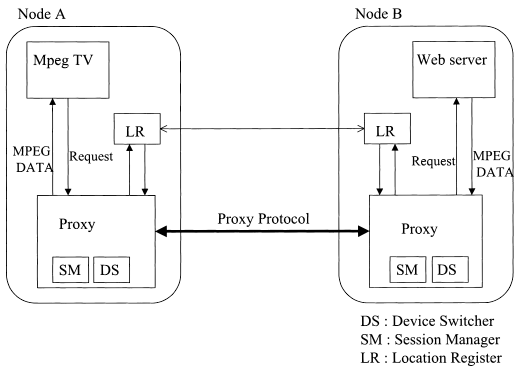

図 5 プロキシの実装
Fig. 5 Implementation of proxy.

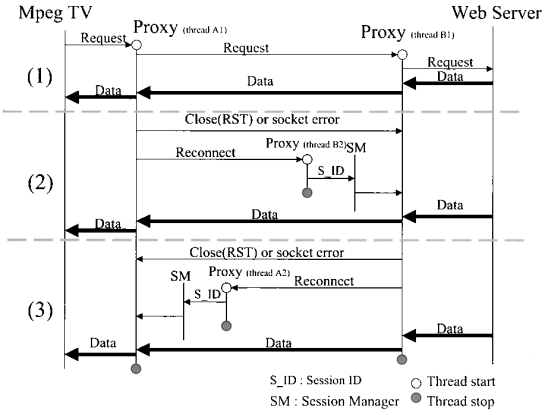

図 6 リンク切替え
Fig. 6 Link switch.

固定したりする必要がある．このような制約はあるものの，従来の固定されたノードを前提としたシステムに比べはるかに柔軟にノードの移動に対処することができ，お互いの位置を捕捉することが可能である．

6. 実装と評価

6.1 実　　装

マルチプラットホームにおける動作を考慮し，Javaで各機能を実装した．Windows，Linuxの環境で動作を確認したが，評価に関してはLinux環境で行う．

図 5 に構成図を示す．プロキシは複数の接続に同時に対応するためスレッド化を行い，複数の接続に対し並列処理を行う．

プロキシプロトコルによる通信は，クラスをSerializeすることにより行った．Serializeのオーバヘッドと MEDLARプロトコルのオーバヘッド（4.2節 (2) のフレーム構成のうちデータ以外の部分の24 byte）はクラス内の送信データ（4.2節 (2) のフレーム構成のうちデータの部分の1024 byte）と比較して3%以内に収まっている．

6.1.1 リンク切替え

切替えのシーケンスを**図 6** に示す．図6 (1) はMpeg TVとWeb Serverを接続するシーケンスである．接続方法は最初にクライアントアプリケーションであるMpeg TVからHTTPのGetコマンドを用い要求が出され，プロキシが中継することにより，Web Serverに送られる．Web Serverでは指定されたURLにあるMPEG画像ファイルを送信し，プロキシはこれをMpeg TVに転送する．

図6 (2) はクライアントの移動を示すシーケンスである．クライアントから切断し，異なる通信デバイスを使い再接続し，切り替えている．その後，データ中継が再開される．

図6 (3) はサーバの移動を示すシーケンスである．サーバから切断し，異なる通信デバイスを使い再接続している．その後データ中継が再開される．

(2) (3) の切断処理ではTCPのRSTフラグによる中断リリースを用いることにより，送信中のデータを無視し，即時に切断している．これは，切替え速度の向上に貢献する．

また，(2)，(3) の再開処理を，以下に示す．実装したプロキシはマルチスレッドで動作するコンカレントサーバであるため，再接続の要求を，以前利用していたスレッド（thread B1，A1）が直接，受け付けることができない．そこで，再接続要求を受け取ったスレッド（thread B2，A2）が再接続要求の中で指定されたSession IDを用いて再接続したソケット識別子をSMに登録する．データ中継を中断していたスレッド（thread B1，A1）は，このSMへの登録をトリガにwait状態から復帰し，SMからソケット識別子を取得することで，データの中継を再開する．

このように，リンク切替えを行うことで，Mpeg TVを利用しているユーザは通信媒体が変わっても連続的に映像を閲覧することが可能であり，連続的なサービス提供が実現する．

6.1.2 リンク断からのリンク回復

急なリンク断からのリンク回復のシーケンスを**図 7** に示す．図7 (1) は図6 (1) と同じであるので説明は省略する．図7 (2) は急なリンク断からのリンク回復のシーケンスである．このような急な切断の場合，カーネル内のソケット処理でとどまり，古いソケットをクローズすることはできない．そこで，リンク切替えの場合と異なり，再接続を受け取ったスレッド（thread B2）により，中継が再開され，それまで中継を担っていたスレッド（thread B1）を強制終了させる．いつリ

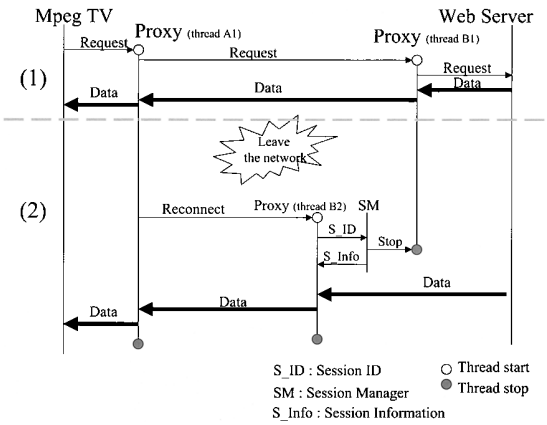

図 7 リンク回復
Fig. 7 Link recovery.

ンクが切断されても対応可能なように，SM はデータ転送再開に必要なな情報（セッション ID，ソケット識別子，通信済データ量等）をつねに保存し，その情報に基づき再接続を受けとったスレッド（thread B2）はそれまでのスレッド（thread B1）の状態を引き継ぐ．

このように，リンク回復を行うことで，Mpeg TV を利用しているユーザは通信媒体が利用できなくなった場合でも，通信媒体が再び利用できるまで待ったり，他の通信媒体に切替えたりすることで，映像の受け取りを再開することが可能であり，連続的なサービス提供が実現する．

6.1.3 LR

各ノードの LR はプロキシとは別の通信機構を用いて，お互いの位置情報をやりとりしている．また，自ノードプロキシから，自ノードの位置情報を受け取ったり，位置検索要求に対し，その結果を返したりする．

グループを構成は，サブネット内のブロードキャストまたはノードを直接指定し，その応答をもとにグループを構成していく．

プロトタイプでは 5.2 節の (a)〜(c) までを実装した．(d) の優先順位の設定，情報の交換は行えるが，実験では十分な数のノードを用いなかったので，位置情報更新命令の優先順位付けは行っていない．

6.2 評　　価

ノードの移動性および効率的な位置管理の実現に関し，プロトタイプおよび計算機シミュレーションによる評価を行う．

6.2.1 ノードの移動性

ここではリンク切替えとリンク断からのリンク回復処理の動作確認を行い，それぞれの切替え時間に関して評価する．実験環境を**図 8** に示す．評価に利用する

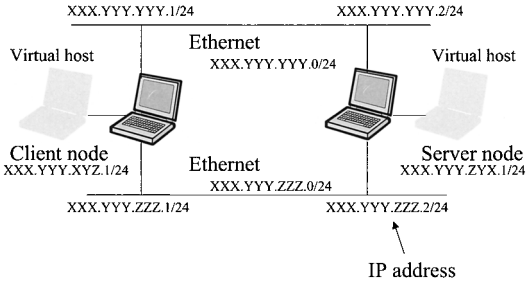

図 8 実験環境
Fig. 8 Experimental environment.

端末を減らすために IP エイリアスを用いた仮想ホストを利用している．この仮想ホストの IP アドレスを指定しプロキシに接続することで，サブネットを越えてルーチングされるためデフォルトルートの変更により，2 つの Ethernet（10BASE-T）を切り替えることができる．OS は RedHat 6.2，クライアント側のブラウザに Mpeg TV，サーバノード側の Web サーバとして Apatch を用いる．また，データは 366 kbps のフレームレートを持つ音声付きの MPEG1 の画像を用い，10 回の切替えの平均値をとった．また，プロキシが再送のために保持するバッファは 200 kByte とした．

切替え時間に関しては要求が来てから終了するまで，通信が切断されている時間の 2 つの時間間隔を測定する．ここでは前者を切替え要求処理時間，後者を切断時間とする．時間の測定はプロキシのプログラム内で行う．

デバイスのチェックには ifconfig コマンドの出力をもとに切替え先のデバイスが動作を確認する．また，ルーチングテーブルの変更は反映されるまで，Linux の場合，2 秒ほどかかるため，2000 ms の sleep コマンドを挿入している☆．

図 9 から，切替え要求処理時間はルーチングテーブル変更時間（約 2 秒）に加え，約 400 ms〜1600 ms で，切断時間は約 90 ms であることが分かる．

次にネットワークから離脱し，違う通信デバイスを用い，リンク回復したときの評価を**図 10** に示す．通信中に LAN のハブからケーブルを抜くことによって，ネットワーク離脱の状況をつくった．この場合，ソケットの切断処理を行う必要がないため，2133 ms と，リンク切替えのときより高速に再接続可能である．

評価実験の結果，動画再生時に切り替えても，アプリケーションやプロキシのバッファによる補間が 100 ms

☆ この時間は OS によって異なる．

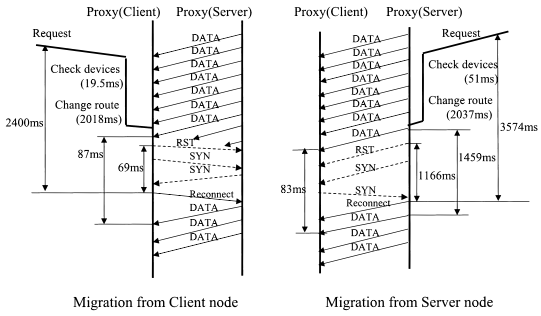

図 9 リンク切替えの評価
Fig. 9 Evaluation of link switch.

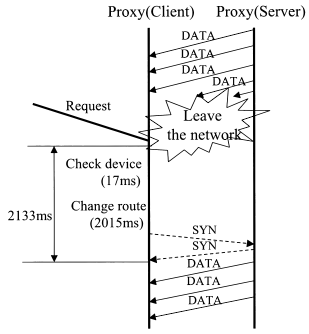

図 10 リンク断，リンク回復の評価
Fig. 10 Evaluation of link recovery.

表 1 シミュレーション諸元
Table 1 Simulation environment.

端末の移動間隔	10 min（指数分布）：1/3 のノード 30 min（指数分布）：1/3 のノード
通信待機時間	20 min（指数分布）
通信時間	50 min（指数分布）
位置情報保持可能容量	すべてのノードの位置：1/3 のノード 1/2 のノードの位置：1/3 のノード 1/4 のノードの位置：1/3 のノード

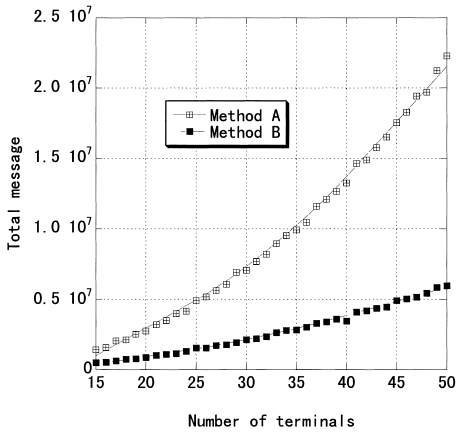

図 11 位置情報更新命令の発行数
Fig. 11 Number of location update instructions.

ほどあれば，ユーザに切替えをまったく意識させることなく切替え可能であることを確認した．また，このとき，データの欠損も起こらない．これは TCP 層で移動性を実現した場合の切替え時間 1000 ms [7] に比べても短い時間である．さらに，ネットワークからの離脱時に復旧が可能であることが分かる．

6.2.2 位置管理方法

位置管理方法の評価を行うため，LR のプロトタイプを用い 5.2 節 (a)〜(c) の動作確認をした．さらに計算機シミュレーションにより位置情報更新の通信コストと位置検索成功効率を評価する．シミュレーション諸元を**表 1** に示す．

すべての LR の情報を正しく設定した状態から，1200 分間に発行された位置情報更新命令の数と LR 検索成功率を 400 試行分，測定した．ただし，最初の 100 分間は無視している．なぜなら，最初の部分は位置情報が正しく設定された理想的な状況と考えられるからである．また，位置情報更新命令は自らが保持しているすべての位置情報を送信し，受信側で位置情報のタイムスタンプにより，取捨選択し自らノードの LR を更新するものとする．

ここでは位置情報更新命令の優先順位の異なる位置更新方式 A，B の 2 つの方式に関して評価する．位置更新方式 A は自らの位置情報を更新した際，位置情報更新命令（図 4）をすべてのノードに送信した場合である．また，位置更新方式 B は移動間隔が 30 分でかつすべてのノードの位置情報の保持可能な容量を持つノードのみがすべてのノードに位置情報更新命令を送信し，その他のノードは前述のノードのみに位置情報更新命令を送信するものである．つまり，位置更新方式 A は 5.2 節で提案した位置管理方法 (a)〜(c) に相当し，位置更新方式 B は (a)〜(d) に相当する．位置更新方式 A は各ノードの位置が更新されるごとにすべてノードに位置情報更新命令を送信する方法のため，更新に要する通信コストは最も多くなるが，位置管理の能力の一番高いものとして評価の比較に用いる．

ノードに対する位置情報更新命令の発行数を**図 11** に示す．また，自ノードの LR の情報またはその情報に基づいて，他のノードに位置情報を問い合わせて，正しい相手のノードの位置を取得できた割合を**図 12** に示す．

図 11，12 の結果から位置更新方式 A に比べ，位置更新方式 B の方が，少ない通信コスト（60〜80%減）でかつ位置検索能力の劣化も少ないことが分かる．また，この効果はグループを構成しているノード数が大

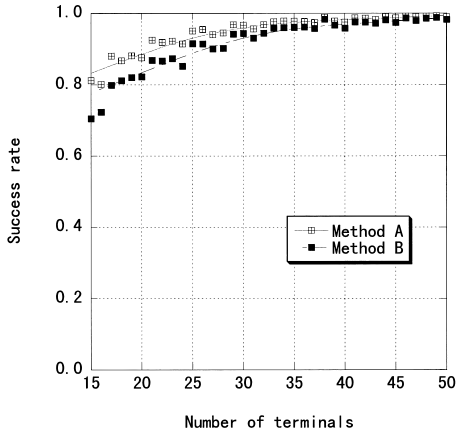

図 12 位置検索成功率
Fig. 12 Success rate for searching location.

きいほど顕著であり，位置検索成功率の差も小さくなることが分かる．各ノードの位置管理の負荷を分担し，位置情報更新に関する通信コストの抑制が図れるとともに，検索成功率の劣化も大きくないことが分かった．また，ここで，検索に失敗した場合でも，検索できたノードにさらに問い合わせていくことにより，限りなく 100%に近づけることも可能である．

また，優先度の設定により，位置管理分担の分散度合を制御することができる．そして，位置管理の分散度合に応じた，通信や位置管理コストで位置管理することが可能である．つまり，提案方式は少数の固定位置管理サーバを置く場合から，すべて移動ノードで構成され，それぞれのノードで位置管理を行う方法まで対応できる．

6.3 考　　察

評価実験より，本提案が要求条件を満たしていることを確認する．

ノードの移動性に関しては，すべてのノードにプロキシとリンク切替え機能を装備することによりリンク切替え可能であり，その際，サービスが途切れることなく，連続的に行える．また，リンク断の場合も通信セッションを維持しリンク回復することができる．したがって，要求条件 (1) を満たしている．

また，LR によりノードの移動にともなうアドレスの変更を管理し，これを高い確率で取得できる．さらに，リンク断時にも各ノードの LR を検索することにより，再接続の手がかりを得ることが可能である．したがって，要求条件 (2) を満たしている．

次に，評価実験を通して判明した課題を述べる．

移動性をサポートする仕組みを構築するとき，既存の TCP/IP の実装では必ずしもルーチングテーブルを瞬時に切り替えることができない．これは，従来ルーチングテーブルは半静的に用いられるものであったため，Linux 等ではその参照速度をあげるためテーブルそのものを直接参照するのではなく，そのキャッシュを参照しているため，キャッシュの更新まで完了しないと更新結果が反映されないからである．しかしながら，リンク切替えの高速化のためには参照速度だけでなく，更新速度も向上させる必要がある．本論文は OS 非依存を目指しているため，更新に時間を要しても切断時間に影響が出ない方式をとることにより，この問題をいくらか吸収することに成功している．しかし，この方式を用いても切替え要求処理時間の短縮にはつながらないため，さらなる性能向上を実現するためには，OS におけるルーチングテーブル更新速度の改善が必要となる．

また，アプリケーション側の課題も存在する．アプリケーションは通信がつながっているように見えても，長時間にわたり応答がなければ何らかの障害が発生したとして，終了処理を行ってしまう．一方的にデータを送信するアプリケーションの場合はプロキシにすべて吐き出すことで回避できるが，応答を待つアプリケーションの場合はタイマを十分長く設定できる必要がある☆．また，プロキシにデータを貯める場合，容量，保存期間の適正値を決定する必要がある．

次に LR について考察する．今回は，移動量（アドレスの変更）が少なく，リソースの多いノードが LR に全体の位置管理の主要な部分を担うことにより，管理コストの削減効果があることを確認した．また，グループ内のノードの数が 50 ノード程度，各ノードのアドレス変更が平均 17 分ごとの場合，グループ内を流れる更新メッセージは 13.6 メッセージ/分であり，実用上問題がないことが分かった．しかしながら，グループの単位がこれ以上大きくなったり，移動がより頻繁に起こったりするような環境ではさらに更新メッセージが増えるため，グループ間の連携方式やより上位の位置管理方式を考える必要がある．また，マルチキャスト技術による効率化も考えられる．

このように，実際にはすべてのノードが自由に移動した場合には位置管理のコスト増等の課題あるが，すべてのノードが必ずしも頻繁には移動しないが，移動させることは可能であるといった状況には十分対応できる．

このように，本提案システムはネットワーク規模が

☆ 実験で用いたブラウザ Mpeg TV はタイムアウトを無限に設定できる．

比較的小規模で，固定された強力なサーバを設置して管理するほど厳密な管理を必要とせず，多様な通信媒体を利用し自由にノードが移動していくような状況に対し最も有効である．

従来このような状況に対して，固定の管理サーバを立ち上げ，厳密に管理していく必要があったり，管理サーバ経由でしか通信できなかったりといった問題点があったが，本提案は必要とされるノードだけを利用し，より柔軟なネットワーク環境を提供することができる．

また，具体的な適用先としては，簡易な移動放送局，特定のメンバ間でアドホックに構築するオフィスネットワーク，キャンパスネットワーク，イベントホール等での情報交換ネットワーク，個人の所有する端末間で構成するプライベートネットワーク，ロボット間ネットワーク，車々間ネットワーク[11]といったものが考えられる．

7. むすび

本論文ではサーバ，クライアント双方の移動性を実現する仕組みに関し提案し，評価した．またその際，固定ノードを用いることなく位置管理を行う方法を提案し，評価した．これにより，ネットワーク上のすべてのノードが移動した場合においてもサービスを連続的に提供，享受することが可能である．また，ネットワークの構成も固定ノードを想定することなく構築することが可能であるため，より柔軟なネットワークを構築することができる．

参考文献

1) Weiser, M.: Some computer science issues in ubiquitous computing, *Comm. ACM*, Vol.36, No.7, pp.75–84 (1993).
2) 片山　穣，高杉耕一，久保田稔，小菊一三：モバイル通信におけるサービス継続機構，信学技法，SSE99-103, pp.13–18 (1999).
3) 片山　穣，高杉耕一，久保田稔，小菊一三：異種ネットワーク間におけるサービス連続性の実現方式，信学論 B, Vol.J84-B, No.3, pp.452–460 (2001).
4) 高杉耕一，片山　穣，久保田稔：ノード及びサービスの移動性を実現するモビリティ技術，情報処理学会研究報告 2000-DSP-98-4, pp.19–24 (2000).
5) 田頭茂明，最所圭三，福田　晃：移動計算機情報発信のための Tool kit の評価，DICOMO 2000, pp.614–624 (2000).
6) RFC 2002.
7) Snoeren, A.C. and Balakrishnan, H.: An End-to-End Approach to Host Mobility, *Proc. 6th ACM MOBICOM*, Boston, MA (2000).
8) Adje-Winoto, W., Schwartz, E., Balakrishnan, H. and Lilley, J.: The design and implementation of an intentional naming system, *Proc.17th ACM SOSP*, Kiawah Island, SC (1999).
9) *Jini Architectural Overview*, Sun Microsystems.
10) Balle, W.V., Vereist, W.K. and D'Hondt, T.: Location Transparent Routing in Mobile Agent Systems Merging Name Lookups with Routing, *Distributed Computing System*, *Proc. 7th IEEE Workshop*, pp.207–212 (1999).
11) Moris, R., Jannotti, J., Kaashoek, F., Li, J. and Decouto, D.: CarNet: A Scalable Ad Hoc Wireless Network System, *9th ACM SIGOPS European Workshop*, Kolding, Denmark (2000).

(平成 12 年 12 月 18 日受付)
(平成 13 年 5 月 10 日採録)

高杉　耕一（正会員）

平成 7 年東京工業大学工学部情報工科卒業．平成 9 年北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．同年，日本電信電話（株）入社．現在 NTT 未来ねっと研究所研究員．アドホックネットワーク，ソフトウェア無線，モバイル環境におけるソフトウェア技術の研究開発に従事．電子情報通信学会会員．

片山　穣

平成 2 年都立科学技術大学航空宇宙システム卒業．平成 4 年東京工業大学理工学研究科修士課程修了．同年，日本電信電話（株）入社．現在 NTT 未来ねっと研究所研究員．実時間 OS，分散処理システムの研究開発に従事．電子情報通信学会会員．

久保田 稔（正会員）

昭和 53 年東京大学工学部計数工学科卒業．昭和 55 年同大学院工学系研究科情報工学専門課程修士課程修了．同年日本電信電話公社入社．昭和 62 年から 63 年にかけて米国マサチューセッツ工科大学客員研究員．現在 NTT 未来ねっと研究所主幹研究員．電子交換プログラムの実行制御方式，実時間 OS，分散処理システム，ソフトウェア開発環境の研究開発に従事．工学博士．電子情報通信学会，IEEE，ACM 各会員．

小柳 恵一

昭和 50 年慶応大学工学部電気学科卒業．昭和 52 年同大学院修士課程修了．平成 10 年大阪大学大学院博士課程修了．昭和 52 年日本電信電話公社入社．現在，未来ねっと研究所ネットワークインテリジェンス研究部長．ディジタル加入者線交換機の研究・開発，通信網基本構想の策定，次世代通信ソフトウェア構成の研究開発，メガメディアネットワーク構築に向けた研究開発の推進およびネットワークミドルウェア研究開発に従事．平成 8 年電気通信普及財団賞奨励賞受賞．工学博士．著書「C++オブジェクト指向設計」等．電子情報通信学会員，IEEE Senior Member．