モーバイル環境下の自律分散通信の実現とその応用

河口信夫† 片桐秀樹† 内柴道浩† 外山勝彦† 稲垣康善†

携帯情報端末の小型化、軽量化により、情報端末を気軽に持ち運び、利用できるようになった。それとともに、モーバイル環境下で他の端末と情報の共有・交換を行いたいという要求が増えている。しかし、端末間の直接情報交換を支援する仕組みはほとんど整備されていない。本稿では、モーバイル環境下での高度で自由な情報交換を行うために、自律的にアドホックネットワークを構築し、ネットワークの動的な変化にも対応して、適切なアプリケーションを実行する自律分散通信の枠組を提案する。そこで、第１に、端末を持ち寄るだけで自律的にネットワークを構築し、直接通信できない相手とは中継により通信を可能にする自律分散通信プロトコルを提案する。このプロトコルはネットワークの動的な変更にも対応できる。第２に、プロトコルの実際の携帯端末上への実現と、具体的なアプリケーションについて述べる。

Implementation and Application of Autonomous Distributed Communication in a Mobile Environment

Nobuo Kawaguchi† Hideki Katagiri† Michihiro Uchishiba†
Katsuhiko Toyama† Yasuyoshi Inagaki†

Recently, it becomes popular to use small size computers such as notebook computers or PDAs in a mobile environment. It sometimes happens that several computers meet at the same place such as meeting rooms or conference sites. In such environment, there are demands to make a direct communication among mobile computers. However, there is no supports for such ad-hoc communications. In this paper, we propose a framework for the autonomous distributed communication sysytem in a mobile environment. We describe autonomous communication protocol for a ad-hoc network and an implementation of mobile system using the protocol.

1 はじめに

近年のハードウェア技術の進歩により、小型で高性能な携帯情報端末を自由に持ち運び、必要になったその場で様々な情報を利用することが可能になりつつある。将来的に、さらに多くの人々が電子手帳や高機能な携帯電話として携帯端末を持つであろうことは想像に難くない。この場合、パーティや会議等の人が集まる場では同時に多数の携帯端末が集まることになる。これらの端末間で直接の情報交換・共有が可能になれば、名刺交換やスケジュール調整、議事録やメモの共有、資料の配布等の、これまで人が行ってきた協同作業の支援が可能になると考えられる (図 1)。しかし、従来の携帯端末間の通信は可能であっても赤外線 [9] を使った１対１通信や、特別な機器を必要とする無線通信 [1][2] に限られているため、十分に活用されてこなかった。

この問題を解決するために、我々はすでに、出会ったその場で、いつでもどこでも誰とでも何台とでも手軽に通信を可能にする、携帯端末用の自律分散通信プロトコルを提案した [4]。このプロトコルでは、各端末は通信のために特別な設定を必要とせず、自律的に周囲の端末を認識し、アドホックにネットワークを構築する。しかし、端末の移動や障害物等により、ネットワークが動的に変化した場合には対応できていなかった。

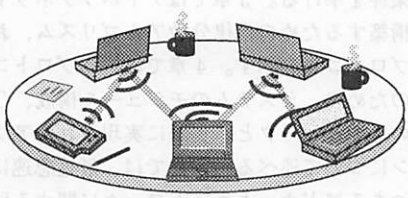

図 1: 携帯端末によるネットワーク

本研究では、第１に、このプロトコルを拡張し、ネットワークへの端末の動的な接続・切断を可能とする自律分散通信プロトコルを提案する。本プロトコルでは、直接通信が不可能な端末同士は、自律的に最適な通信路を選択し、中継を行って通信を行う。各端末はそれぞれネットワークに関する情報を保持し、動的な変更に自律的に対処できる。また、プロ

†名古屋大学大学院工学研究科 計算理工学専攻
Department of Computational Science and Engineering, Graduate School of Engineering, Nagoya University

トコルとしては任意の通信媒体の利用が可能であるが、本研究では具体的な通信媒体として、小型で軽量、小電力であり、最も普及している赤外線通信を用いる。赤外線通信は指向性が高く到達距離が短いため、隠匿性が高く、名刺交換等のプライベートな情報通信に適している。

第2に、本プロトコルを具体的なアプリケーションとして携帯端末上に実現し、その有効性を確認する。プロトコルやアプリケーションの実現・テストを容易にするために、モジュールを自律分散プロトコルの実現部と通信デバイス固有の制御部に階層化した。各階層間でのインタフェースを明確化することにより、階層毎にデバッグが可能となる。特に、実際に赤外線通信を利用したデバッグは時間と労力を要するため、TCP/IP 上で赤外線通信をエミュレートする環境を構築し、デバッグ期間を短縮した。

具体的なアプリケーションとしては、Windows 95/CE 上で、URL を交換することにより様々な情報交換を行うシステムを構築した。本システムは会議の議録の共有や、プレゼンテーション等に利用でき、本プロトコルの有効性が確認できた。

本稿では、まず2章にて、モーバイル環境下にて自由度の高い情報交換を行うためには、アドホックネットワークが必要であること述べ、その満たすべき条件を挙げる。3章ではアドホックネットワークを構築するための自律分散アルゴリズム、および通信プロトコルを示す。4章では、本プロトコルの実装のための、システムのモジュール構成、TCP/IP 上でのエミュレータと、実際に実現されたアプリケーションについて述べる。5章では、最近急速に増加しつつあるアドホックネットワークに関する研究について述べ、本研究の位置付けを行う。

2 アドホックネットワーク

まず、モーバイル環境下で、複数のユーザが互いの端末上の情報を交換・共有したい状況を考えてみる[1]。この場合、端末は他の端末に関する情報を持たないため、従来の通信方法では、ユーザが通信を行うための設定を行う必要がある。また、すべての端末が互いに通信可能であるとは限らないため、どのように中継を行うかという情報、すなわち、経路情報の交換が必要となる。このような状況で、端末が互いを自律的に認識し、通信を可能にする通信ネットワークを、ここではアドホックネットワークと呼ぶ。

[1]ここでは通信媒体として無線を考えている。

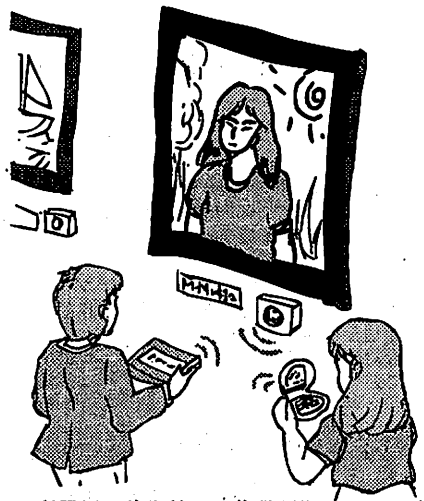

図 2: 利用例：美術館での作品説明やメモの交換

さらに、ネットワークが構築された後には、どのアプリケーションを使って通信を行うかをユーザが指定する必要がある。これまでも、IrDA を使ったファイル転送等の専用アプリケーションがあるが、汎用性は無く、ファイルの指示等は繁雑である。理想的には、あるユーザが名刺の情報の送信を指示すれば、他のユーザの端末では、自動的に名刺管理ソフトが起動し、データが送られるべきである。このような処理を適切に行う事により、名刺交換、議事録、ノートの共有、資料の配布、スケジュール調整といった、これまで人と人が行ってきた協同作業の高度な支援が可能になる。加えて、各人の個人情報や、出会いの履歴が端末に記録されていれば、共通の知人や趣味の発見、伝言の伝達など、新たなコミュニケーションが可能になる。

また、本研究では、対象とする情報端末を従来考えられている範囲から少し広げて考える。すなわち、ノートパソコンやハンドヘルド PC といった小型のコンピュータに加え、自宅のデスクトップパソコン、また携帯電話、ボイスメモ、ポケットベルといった特定用途向けの情報機器も対象とする。これらの機器はそれぞれ決まった状況で使われることが想定されるため、各端末は現在の状況を認識し、状況によって適切なアプリケーションを自動的に実行することが望ましい (context-aware)。例えば、ユーザが自宅のパソコンの近くへ携帯端末を近付けると、端末は予備的な通信により、自宅へ帰宅したことを認識し、自動的にスケジュールや名刺データベースを同期させる。他にも、端末を友人の携帯電話に近付けることにより、自動的に電話番号を交換したり、

ポケットベルから端末へメッセージの保存を行ったりすることが考えられる。

携帯端末のみでなく、固定設備にもこのような枠組を適用することにより、様々な応用が考えられる。例えば、固定ネットワークに接続された端末があれば、携帯端末からその端末を中継して固定ネットワークを利用できる。また、美術館や博物館等の部屋毎に、情報提供を行う固定端末が存在すれば、多数の人が同時に自分の端末で詳細な資料を確認することや、周囲の人との情報交換を行うことができる(図2参照)。

このように、アドホックネットワークの構築、経路制御、アプリケーション制御を統一的に扱うことは、携帯端末の高度な利用を可能にする。その実現には、以下の性質を持つ自律システムが必要となる。

1. 自律的動作により、アドホックネットワークを構築
2. ネットワークトポロジーの動的変化に対応
3. 状況依存のアプリケーション制御

この場合、ユーザは本当に望むことを指示するだけで良く、他の繁雑な処理はすべて自律システムが行ってくれる。

3 自律分散通信プロトコル

本章では、アドホックネットワークを構築し、ネットワークの動的な変化に対応するアルゴリズムについて述べる。最初に、本プロトコルの持つ前提条件を挙げる。

3.1 前提

以下では、各端末(通信局)をノード、ノード間の通信路をリンクと呼び、直接通信可能なノードを隣接ノードと呼ぶ。また、ノード発見手続きを開始するノードを開始ノードと呼ぶ。

ノード、リンクに関する条件を以下に挙げる。

1. 各ノードは固有のIDを持つ。
2. 初期状態では各ノードは他のノードに関する情報を持たない。
3. リンクは双方向に通信可能である。
4. 各ノードは自発的に隣接ノードとのリンクの存在を発見できる。

また、本稿ではアルゴリズムを簡単にするための条件として、以下の2点を仮定している。

1. ネットワークの構築中には、ネットワークトポロジーは変化しない。
2. 通信中にはデータの改変、損失は起こらない。

表1: 関係記号の意味

記号	意味
N	初期値
M	自ノード
P	隣接ノードで、自ノードを発見したノード
D	隣接ノードで、隣接リストを送信したノード
I	隣接リストを送受信しないノード
C	開始ノードの変更通知を行うノード
K	探索が終了したノード

表2: ノードn_1の情報テーブルの例

ID	in	out	rel	AL
n_1	F	F	M	$[(n_1,n_2),(n_1,n_3)]$
n_2	T	T	D	[ ]
n_3	F	T	N	[ ]

3.2 準備

各ノードn_iは2項組(I_i,T_i)を保持する。ここで、I_iは開始ノードのIDを保持する変数、$T_i = [t_i^1,\ldots,t_i^k]$は全ノードの情報を保持する情報テーブルであり、各$t_i^j = (n_j, in_i^j, out_i^j, rel_i^j, AL_i^j)$はノード$n_j$に対応する5項組である。$n_j$はノードのIDであり、$in_i^j, out_i^j$はノード$n_j$と隣接リストを受信,送信したかどうかを表すフラグである。rel_i^jはn_jとの関係を表し{N,M,P,D,I,C,K}のいずれかの値をとる。それぞれの記号の意味は表1に示す。また、AL_i^jはノードn_jから受け取った隣接リストを表す。隣接リストとは、各リンクに対応するノード対のリストである。情報テーブルの例を表2に示す。各ノードは、以下に示す各メッセージをやりとりする。ここでn_iは自ノードのIDである。

1. IDandAL(n_i,n_j,n_k,AL_i)
   ノードn_jへ自ノードの開始ノードn_kと隣接リストAL_iを送る。
2. Info(n_i,n_j,AL_i)
   ノードn_jへ隣接リストAL_iを送る。
3. Ignore(n_i,n_j)
   以後、n_jとの間では、隣接リストのやりとりをしない。
4. Change(n_i,n_j,n_k,AL_i)
   ノードn_jへ開始ノードのn_kへの変更を通知する。

また、各ノードはアルゴリズム中で以下の関数を利用する。

1. Detect()
   発見した隣接ノードのIDをリストとして返す。
2. Send(Mes)
   メッセージ Mes を送信する。宛先の情報は

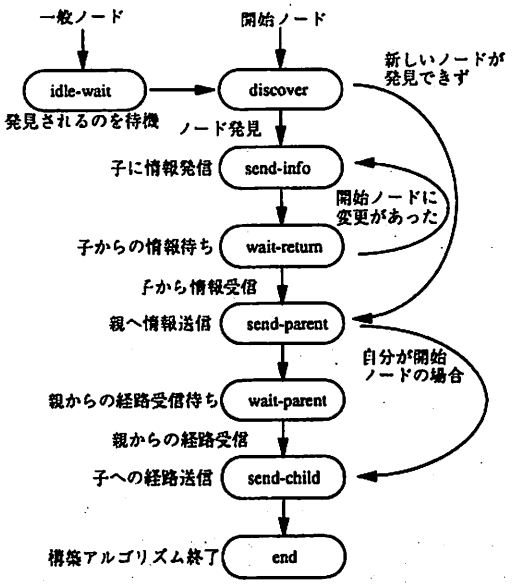

図 3: アルゴリズムの概略

Mes 内に含まれる。

3. Recv()
   メッセージを受信し、その値を返す。

3.3 アドホックネットワークの構築

具体的なアルゴリズムを付録[A]に示し、図3に動作の概略を示す。プロトコルの動作は1つ以上の開始ノードから始まる (discover)。他の一般ノードは開始ノードに発見されるのを待機している (idle-wait)。開始ノードは隣接ノード (子ノードと呼ぶ) を発見し、情報交換をする (send-info,wait-return)。発見されたノードも同様に他のノードの発見、情報交換を繰り返す。このように次々とノードの発見、情報交換を行うことにより、ネットワークの末端まで探索が到達すると、逆に親ノードへ接続情報を送り返す (send-parent)。開始ノードが全てのノードからの情報を受け取ると、隣接リストを完全にするために、子ノードへ経路情報を送り、各子ノードもそれを繰り返す (send-child)。この結果、すべてのノードがネットワーク全体の隣接リストを獲得することができる。

例えば図4の場合、n_1 を開始ノードとすると、n_2, n_3 を発見し、情報交換を行う。n_2 は n_4 を発見し、情報交換を n_5, n_6, n_7, n_8 と繰り返していく。n_3 は他のノードがいないため、親ノードへ情報を送信し、親からの経路情報を待つ。n_7 は複数のノードから発見されるが、開始ノードが同一の場合、最

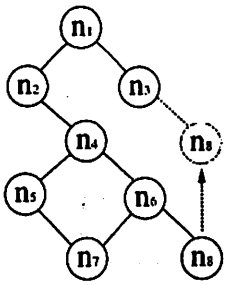

図 4: アドホックネットワークの例

表 3: ノード n_4 のテーブル

ID	rel	AL	next hop
n_1	K	[ ]	n_2
n_2	K	$[(n_1, n_2), (n_1, n_3), (n_2, n_4)]$	n_2
n_3	K	[ ]	n_2
n_4	M	$[(n_4, n_5), (n_4, n_6)]$	n_4
n_5	K	$[(n_5, n_7)]$	n_5
n_6	K	$[(n_6, n_7), (n_6, n_8)]$	n_6
n_7	K	[ ]	n_5
n_8	K	[ ]	n_6

初に発見されたノードを親とみなし、他方には情報を送信しないため、情報交換は木構造で行われ、この手続きは必ず停止する。また、開始ノードが複数あった場合、探索が衝突したノードで、双方の探索の情報を結合し、新たに開始ノードとして動作する。表3に n_4 の最終的な情報テーブルの一部を示す。なお、ネットワーク探索後の rel は自ノード以外はすべてKとなる。

3.4 ルーティングテーブルの作成

ネットワーク構築アルゴリズムにより、各ノードはネットワーク全体の隣接リストを獲得する。各ノードは隣接リストから他ノードへの経路を、中継数ができるだけ少なくなるように計算する。すなわち spanning tree の最小パスアルゴリズムによりルーティングテーブルを作成する。図4における n_4 のルーティングテーブルを表3の next hop 列に示す。

3.5 データの配送

本プロトコルでは、基本的にマルチキャストによって通信を行う。ネットワーク上を流れるデータメッセージは以下の形式を持つ。

Data(n_i,n_j,F.data)

ここで n_i は送信元、n_j は中継先、F は送信先のIDのリスト、data は上位層からの送信データである。各ノードはこのメッセージを受け取ると F 中の各ノードに対し、ルーティングテーブルにより中継先を決定し、中継先毎にメッセージを分割して送信する。

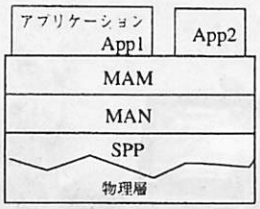

図 5: モジュールの階層構造

3.6 ネットワークの動的変化の検出・対応

ネットワークが完成すると、各ノードは付録 [B] に示す監視アルゴリズムを実行する (idle)。

ノード n_i がネットワーク上のリンク (n_i, n_j) が切れたことを検出すると、n_i は情報テーブルから、リンクに対応する隣接リストを消去し、n_j 以外の隣接ノード n_k に対しメッセージ deleteAL(n_i, n_k, F, $[(n_i, n_j)]$) を送信する。ここで F は n_k を中継に用いてデータを送信するノードのリストである。このメッセージはデータメッセージと同様に配送され、全ノードはリンク (n_i, n_j) が切れたことを知る。

逆にネットワークのノード n_i に新たなノード n_j が近付いた場合、n_i は n_j を発見し、IDandAL(n_i, n_j, n_i, AL_i) を送信する。n_j は他のノードの探索を行い、n_i へ隣接リスト AL_j を送り返す。n_i は他の隣接ノード n_k に対し、参加メッセージ appendAL(n_i, n_k, F, AL_j) を送信する。このメッセージもデータメッセージと同様に配送され、全ノードは新たなノードの参加を知る。また、リンクの変化が伝わると、各ノードはルーティングテーブルを再計算する。

図 4において、ノード n_8 が、n_6 から n_3 へ移動した場合、n_6 は n_8 の退出を検出し、deleteAL(n_6, $n_4, \{n_1, n_2, n_3, n_4\}, [(n_6, n_8)]$), deleteAL($n_6, n_7, \{n_5, n_7\}$, $[(n_6, n_8)]$) を送信し、全ノードへ転送される。次に n_3 が n_8 を発見し、情報交換を行い、appendAL(n_3, $n_1, \{n_1, \ldots, n_7\}, [(n_3, n_8)]$) を送り、全ノードが n_8 の参加を知る。このように、本プロトコルでは動的なネットワークトポロジーの変更に対応している。

4 プロトコルの実現

本プロトコルを実際に Windows 95/CE の端末上に実現した。実際に無線通信を用いたコーディング・デバッグ作業では、複数の端末を用い、プログラム更新の度に転送を行う必要があり、非常に繁雑な作業となる。そこで、プロトコルを階層化し、無線通信をエミュレートするライブラリを構築して、コーディング・デバック作業を容易にした。

図 5にモジュールの階層構造を示し、以下では各層について説明する。

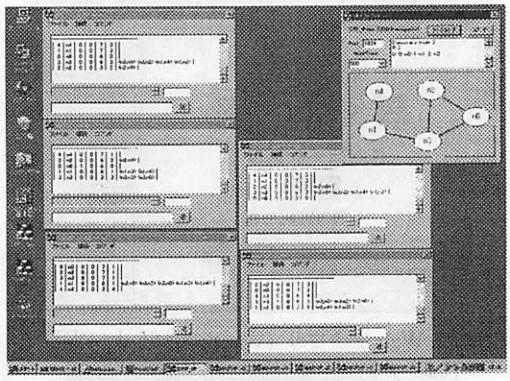

図 6: エミュレーション環境での実行画面

- **物理層：**
  具体的な通信デバイス。本稿では、アプリケーションの実現に IrDA, 開発環境のために TCP/IP によるエミュレーション環境を用いた。超音波通信や、光通信であっても良い。
- **SPP(Simple Point to Point) 層：**
  物理層を隠蔽し、隣接ノードの発見、エラー処理、高信頼の 1 対 1 通信を実現する。
- **MAN(Mobile Ad-hoc Network) 層：**
  アドホックネットワークの構築し、ルーティングにより、直接通信できないノード間での多対多通信を実現する。
- **MAM(Mobile Application Manager) 層：**
  ネットワークの確立後、アプリケーションを動的に実行したり、種類の異なるアプリケーション間の通信を支援する。
- **アプリケーション層：**
  MAM により管理される多種多様なアプリケーション。URL メモや電子黒板、名刺管理等。

4.1 エミュレーション環境

プロトコルを実現するシステムの開発、デバッグのために、SPP 層として、TCP/IP 上に赤外線をエミュレートする環境を構築した。図 6にエミュレーション環境上でのプロトコルの動作の様子を示す。右上のウィンドウはエミュレーションサーバであり、各ノードの通信を管理する。画面上には、各ノードの物理的な接続、及び赤外線通信の様子を表示する。小さなウィンドウ群は、各々がノードを実現するプログラムであり、現在の情報テーブルを表示している。各プログラムは SPP を用い、TCP/IP によりサーバに接続され、相互に通信を行ってネットワークを構築する。

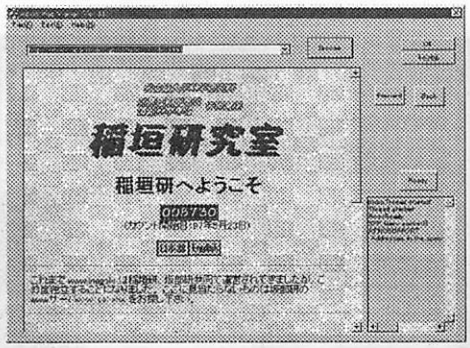

図 7: Web ブラウザ (Windows 95 上に実現)

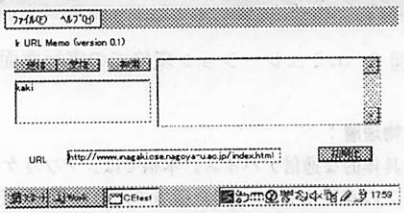

図 8: URL メモ (WindowsCE 上に実現)

4.2 モーバイルアプリケーション

モーバイルアプリケーションとして、Web ブラウザ (図 7) 及び URL メモ (図 8) を開発した。

URL メモは、自分の持っている URL 情報を他の端末に送信したり、逆に他の端末の持っている URL を取得したりすることができる。Web ブラウザは、Web ページを表示し、かつ他の端末と URL や Web ページに関する通信を行うことができる。Web ブラウザ同士では、同期してページを表示したり、URL を送ることができる。また、URL メモとも通信し、URL メモによるリモートコントロールも可能である (図 9参照)。これらのアプリケーションは MAM によって管理されており、各端末では登録されている Web 関連アプリケーションが自動的に起動する。

Web ビューア, URL メモにより、簡単な会議を行うことができる (図 10参照)。

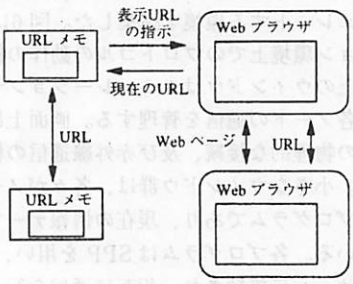

図 9: URL メモ、Web ブラウザ間の通信

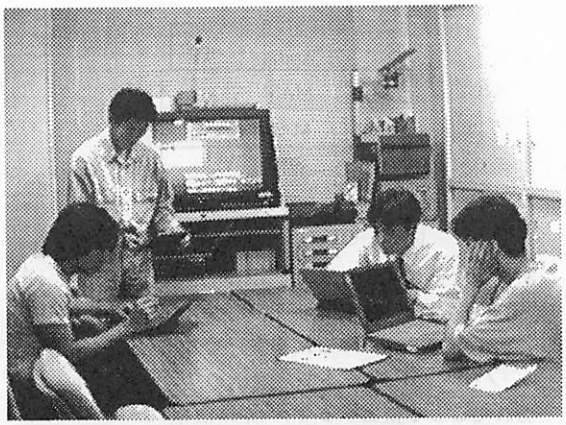

図 10: アドホックネットワークによる会議

5 関連研究

モーバイル環境に特化したアプリケーションについては、様々な研究がなされている。ParcTab[5, 6] は小型の端末であり、構内に設置された赤外線送受信機を通じてサーバにアクセスし、天気予報を問い合わせたり、メールチェック等のアプリケーションを実現している。しかし、端末間の直接通信を実現しているわけではなく、自律的なアドホックネットワークを構築しているわけではない。また、MagicCap を用い、国際会議において、携帯機器を約 100 名のユーザに持たせ、その振舞いを調べる実験 [3] が行われたが、携帯機器間の直接通信はほとんど考えられておらず、これもサーバとのやりとりを通したサービスのみが提供されている。一方 'なかよし'[1] や WirelessDAN[2] は PHS パケット通信や赤外線拡散通信によりアドホックネットワークを構築し、グループウェアを実現している。通信のために特別な機器を必要し、アプリケーション管理の機能が実現されていない点が本研究と異なる[2]。

一方、アドホックネットワークの構築プロトコルについては、IETF[10] 内で MANET というアドホックネットワークのワーキンググループにより研究が行われている。これは AODV[8] や MobileIP[7] に代表されるルーティングやデータの転送に関する研究であり、主に既存の IP を用いたアプリケーションをモーバイルコンピューティングでも利用しようという発想である。本研究とは、IP を無理にモーバイル環境下で利用するのではなく新たなモバイル用のプロトコルやその応用を考えている点で異なる。

最近では、産業界からもアドホックネットワー

[2]WirelessDAN の研究を引き継いだ SpanWorks 社 [11] からは、管理機能を持つ電子ノート等の製品が発売されている。

クに関する提案がなされている。Bluetooth[12] は 2.4GHz 帯,1Mbps、飛距離 10m 程度の無線通信機能を持つチップを 9mm×9mm 程度のサイズに実現し、様々な機器に内蔵させることを目的にした規格である。デスクトップと PDA 間のデータの同期や、ヘッドセットと携帯電話、ノートブック間の通信などを想定している。また、HomeRF[13] は家庭内で利用することを目的とした無線通信規格であり、2.4GHz 帯,1Mbps,127 台までのデータ同時接続、6 回線までの音声伝送を可能とする SWAP というプロトコルを提案している。この規格はディジタルコードレスや無線 LAN の相互接続を目指したものであり、アドホックネットワークもデータ通信に限り可能である。より家庭向きの規格としては IrBus[14] がある。これは、6-8m 程度の飛距離、75kbps 程度の速度、8 台までの同時通信可能であり、将来の家庭内のリモコンの代替となることを目指しているが、アドホックネットワークを構築することも可能であろう。これらの規格は提案のみがされている状況であり、製品化は数年後になされる予定である。本プロトコルの物理層としても応用が可能であり、今後の発展に注目したい。

6 まとめ

本稿では、ユーザの持つ携帯情報機器や固定機器の間で、適切な情報交換を容易に行うための枠組を示した。その基礎技術として、各端末が自律的にアドホックネットワークを構築し、適切なデータを交換するためのプロトコルを提案した。また、プロトコルを階層化し、開発効率を高め、具体的な URL を自由に交換するアプリケーションを実現した。

今後の課題を以下に挙げる。本プロトコルは、ネットワーク構築中の動的なトポロジー変化に対して効率的な動作を行うことができない。これは、タイムアウト等の導入によって対処が可能であろう。しかし、一時的な通信エラーとリンクの切断との区別は物理的に困難であるため、本質的な問題として残る。また、通信や経路情報に関するセキュリティの確保の必要がある。他のノードに転送するデータが中継ノードに送られるため、暗号化等による保護の必要があろう。

本プロトコルの応用により、様々なアプリケーションが考えられる。例えば Java 等のインタプリタ言語を用いることにより、アプレットそのものの配送が可能になる。また、通信媒体を赤外線から、電波や、光、超音波等に変更すれば、海中、空中、宇宙空間での動的なネットワークの構築が可能となる。例えば、海中調査をロボットで行う場合、各ロボット間で超音波等を用いて自律的に通信を行い、協同作業を行うことができる。また、固定局を必要としないことから、緊急・災害時等の通信ネットワークの基盤としても有用である。ネットワークの設定が不要なことや、動的な構成の変更が可能なことは、状況に応じた柔軟な対応を可能とし、緊急時により大きな効力を発揮することが期待される。

参考文献

[1] 倉島顕尚，市村重博，田頭繁，前野和俊，武次将徳，永田善紀：集まったその場での協同作業を支援するモバイルグループウェアシステム「なかよし」，マルチメディア，分散，協調とモーバイルワークショップ論文集 (DiCoMo97)，pp.233-238(1997).

[2] 多鹿陽介，岩村和昭，池上史彦，中村誠：携帯情報機器の通信に適した自律無線ネットワーク WirelessDAN の提案，信学技報，IN 94-161(1995).

[3] 西部喜康，和気弘明，森原一郎，服部文夫：モバイル環境下でのユーザの振舞いの解析とエージェント通信への適用法の検討 – Experiments of ICMAS'96 Mobile Assistant Project –，信学論，Vol.J81-D-I，No.5，pp.523-531(1998).

[4] 片桐秀樹，河口信夫，稲垣康善：モーバイル環境下における赤外線を用いた自律分散通信プロトコル，マルチメディア，分散，協調とモーバイルワークショップ論文集 (DiCoMo97)，pp.67-72(1997).

[5] Bill N. Schilit, Norman I. Adams, and Roy Want: Context-Aware Computing Applications,*In Proceedings of the Workshop on Mobile Computing Systems and Applications*,Santa Cruz, CA, pp.85-90(1994).

[6] Roy Want, Bill N. Schilit, Norman I. Adams, Rich Gold, Karin Petersen, David Goldberg, John R. Ellis and Mark Weiser: The PARCTAB Ubiquitous Computing Experiment, Technical Report CSL-95-1, Xerox Palo Alto Research Center(1995).

[7] Charles E. Perkins:Mobile-IP, Ad-Hoc Networking, and Nomadicity, Compsac'96 (1996).

[8] Charles E.Perkins: Ad Hoc On Demand Distance Vector (AODV) Routing , Internet Draft,draft-ietf-manet-aodv-00.txt (1997).

[9] Infrared Data Association: Serial Infrared Link Access Protocol, Version1.1(1996). (http://www.irda.org)

[10] Internet Engineering Task Force: (http://www.ietf.org)

[11] SpanWorks 社: (http://www.spanworks.com)

[12] BlueTooth : (http://www.bluetooth.com)

[13] HomeRF: (http://www.homerf.org)

[14] IrBus: (http://www.irbus.org)

[A] ネットワーク構築アルゴリズム

初期条件:

一般のノード n_i : (idle-wait) から実行

$T_i := [(n_i, F, F, M, [\,])]$

$I_i := \varepsilon$

開始ノード n_s : (discover) から実行

$T_s := [(n_s, F, F, M, [\,])]$

$i := s, I_s := n_s$

アルゴリズム :

⊕ はリストへの要素の追加及び連接を表し、$\bigcup_k L_k$ はリスト L_i の全て連接を表す。T_i 中の各ノードの ID、AL_i^j の集合をそれぞれ $ID(T_i), AL(T_i)$ と表す。

(idle-wait) ;; 探索されるのを待つ
Mes := Recv()
if Mes = IDandAL(n_t, n_i, I_t, AL_t) then
if $n_t \notin ID(T_i)$ then
$T_i := T_i$⊕(n_t, T, F, N, AL_t)
if $I_i = \varepsilon$ then
$I_i := I_t, rel_i^t$:=P
else if $I_i \neq I_t$ then
if $\exists u, rel_i^u$ = P then
rel_i^u := C, in_i^u =F
$I_i := n_i, rel_i^t$ =C
else if $I_i = I_t$ then
rel_i^t := I
goto (discover)

(discover) ;; ノード探索開始
Find := Detect()
Find := Find $-ID(T_i)$;; すでに発見済みは除く
if Find = ε then goto (send-parent) ;; 発見なし
else foreach $n_j \in$ Find
$AL_i^j := AL_i^j$⊕(n_i, n_j)
$T_i := T_i$⊕$(n_j, F, F, N, [\,])$
goto (send-info)

(send-info) ;; 探索ノードへの接続情報送信
foreach $n_j \in ID(T_i)$
if out_i^j=F then
if rel_i^j =N then
Send(IDandAL($n_i, n_j, I_i, \bigcup_k T_i^k$))
rel_i^j :=D
else if rel_i^j =I then
Send(Ignore(n_i, n_j))
else if rel_i^j =C then
Send(Change($n_i, n_j, I_i, \bigcup_{k \neq j} AL_i^k$))
rel_i^j :=D
out_i^j :=T
goto (wait-return)

(wait-return);; 子ノードからの部分情報待ち
while $\exists j$,in_i^j =F and $rel_i^j \in \{C, D\}$
Mes := Recv()
if Mes = Info(n_t, n_i, AL_t) then
in_i^t := T, $AL_i^t := AL_i^t$⊕AL_t
else if Mes = Ignore(n_t, n_i) then
in_i^t := T, rel_i^t :=I
else if Mes = Change(n_t, n_i, I_t, AL_t) then
out_i^t := F, in_i^t := T, $AL_i^t := AL_i^t$⊕AL_t
$\exists u, rel_i^u$ =P, in_i^u :=F, rel_i^u := C
rel_i^j := P
if $I_i = n_i$ then $I_i := I_t$
else $I_i := I_t$, goto (send-info)
if $I_i = n_i$ then goto (send-child)
else goto (send-parent)

(send-parent);; 親ノードへの部分情報送信
if $\exists j, in_i^j$ = F and rel_i^j = P then
Send(Info($n_i, n_j, \bigcup_{k \neq j} AL_i^k$))
out_i^j :=T
goto (wait-parent)
else goto(send-child) ;; 自ノードが開始局

(wait-parent);; 親ノードからの経路情報配信待ち
Mes := Recv()
if Mes = Info(n_t, n_i, AL_t) then
$AL_i^t := AL_i^t$⊕AL_t
goto (send-child)

(send-child) ;; 子ノードへ経路情報配信
foreach $n_j \in ID(T_i)$, where rel_i^j = D
Send(Info($n_i, n_j, \bigcup_{k \neq i,j} AL_i^k$))

(end) ;; アルゴリズム終了
forall $j, (j \neq i) rel_i^j$:= M
$I_i = \varepsilon$

[B] 動的なトポロジー変化に対応するアルゴリズム
Wait(tile) は time 単位時間だけ待機する手続き
NextHop(T_i) は隣接ノードの集合、$Next_i(n_j)$ は n_i から n_j を通って中継されるノードの集合を表す。

(idle) ;; 動的なトポロジー変化の監視
Wait(IdleTime)
Find := Detect()
if Find= $ID(T_i)$then goto (idle) ;; 変化なし
else if $ID(T_i) \subset$Find then ;; 新規参加
foreach $n_j \in$ (Find $- ID(T_i)$)
$AL_i^j := AL_i^j$⊕$(n_i, n_j), I_i := n_i$
$T_i := T_i$⊕$(n_j, F, F, D, [\,])$
Send(IDandAL($n_i, n_j, I_i, \bigcup_k AL_i^k$))
goto (wait-new)
else ;; リンク消滅
goto (send-delete)

(wait-new) ;; 新規ノードからの部分情報待ち
while $\exists j$,in_i^j =F and rel_i^j =D
Mes := Recv()
if Mes = Info(n_t, n_i, AL_t) then
in_i^t := T, $AL_i^t := AL_i^t$⊕AL_t
goto (send-new)

(send-new) ;; 新規ノードへの情報送信
foreach $n_j \in ID(T_i)$ where rel_i^j = D
Send(Info(n_i, n_j, AL_i))
foreach $n_k \in$NextHop(T_i)
$F := Next_i(n_k)$
Send(appendAL(n_i, n_k, F, AL_i^j))
goto (idle) ;; 追加終了

(send-delete);; リンクの消滅
foreach $n_j \in (ID(T_i) -$Find)
$T_i := T_i - t_j$
foreach $n_k \in$NextHop(T_i)
$F := Next_i(n_k)$
Send(deleteAL($n_i, n_k, F, [(n_i, n_j)]$))
goto (idle) ;; 終了