ユビキタスコンピューティング環境における機器間連携フレームワークに関する研究

岩崎 陽平

概要

近年，様々な機器がコンピュータとネットワーク機能を内蔵し，至る所にコンピュータが存在するユビキタスコンピューティング環境が実現されつつある．これらの機器がネットワーク経由で連携した高度なアプリケーションサービスを実現し，我々の生活を支援することが期待される．ネットワーク機器間の連携を支援する多くのフレームワークが提案，規格化されているが，設定が煩雑，動的な機能拡張が困難，連携機能の開発が困難等，様々な問題点が残されている．

本論文では最初に，ユビキタスコンピューティング環境における機器間連携フレームワークに関連した研究動向および課題について述べ，本論文の位置づけとアプローチを示す．連携させる機器を指定する際に，アドレスや名前などの間接的なシンボルを指定するのは煩雑なため，本論文では，実世界の機器そのものを直接的に指定する手法を提案する．また，従来の連携ソフトウェア開発手法では，連携機能が複数の機器に分散し，開発や保守が困難となるため，本論文では，機器間の連携機能を，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発・導入できる，分散透過な連携ソフトウェア開発手法を提案する．

本論文ではまず，機器を連携させる際に煩雑な設定を不要とするために，接続したい両機器のボタンを押すだけで２つの機器間の接続を直接的に指示できる，接続指示フレームワーク Touch-and-Connect を提案する．ボタンを押して接続を指示するインタフェースでは，一般に複数のユーザの操作が干渉して誤接続が発生する可能性がある．本手法では，状態を表示可能なボタンを用いてユーザの操作を排他制御し，複数のユーザが独立に操作を行う状況においても誤接続を防止する．ブロードキャスト通信を用いた本手法のプロトコルは，管理サーバを必要とせず，動的な端末の入退出への対応も考慮しているため，必要に応じて一時的に構築されるアドホックネットワークでも利用可能である．また，セキュリティと使いやすさの向上のため，グループを作成して機器間の接続可能性を制限できる．本フレームワークのプロトタイプシステムを実装してその実現可能性を示し，被験者実験により，インタフェースとしての使いやすさ，および本手法により誤接続が防止できることを示した．

さらに，ボタンを押すことが困難な遠くの機器に対しても，携帯端末で指し示すことにより直接的に指定可能とするために，本論文では，ユーザの位置と方向に基づいたより高度な位置情報サービスである方向依存サービスを提案する．多数の機器が存在するユビキタスコンピューティング環境では，位置のみではユーザの望む機器を特定することは難しいが，ユーザの位置だけでなく指し示している方向を用いることにより，より細かな指定が可能となる．ユーザの方向は，磁気コンパスを内蔵した方向センサによって取得できるが，ユーザの位置に関しては，現状ではどこでも利用可能かつ安価な位置取得技術は存在しない．本論文では，方向依存サービスにおいて安価に位置を取得するために，手動の位置推定手法を提案する．位置が既知の複数のマーカの方位角を，ユーザが指し示して手動で測定し，各マーカより引かれる半直線の交点をユーザの位置と推定する．マーカには色によって種類を設け，ユーザがこれをボタン等により入力することにより，ユーザが指しているマーカを特定する．本手法は，位置取得のために他の位置センサや電子ビーコン等が不要であり，安価に方向依存サービスを展開できる．

また，広い範囲でのサービス運用を可能とするために，無線 LAN の基地局情報を用いた方向依存サービスシステム Azim を提案する．本システムは，ユーザにサービスを提供するために無線 LAN を用いており，マーカの色だけでなく，無線 LAN の基地局情報から得られる位置情報を組み合わせてマーカを特定することにより，限られたマーカの色数で広い範囲での運用を可能としている．また，機器や金属等により地磁気が乱れ，正確な方向測定が困難な屋内環境においても，事前に磁気の空間的な分布を学習して MFI(磁気分布情報) を作成することにより，地磁気の乱れに対してロバストな位置推定が可能となる．方向依存サービスの有用性を判断するために，本システムを用いて屋内および屋外で評価実験を行った．また，本システムのプロトタイプシステムとして，磁気センサと加速度センサを組み合わせた方向センサ，および Linux が動作する PDA を用いて，実際に携帯可能なクライアント端末 LocPointer を作成した．

次に本論文では，事前に想定していない機器との連携機能も，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発・導入できるようにするために，連携ソフトウェアの自動分散化手法を提案する．特に，ワンチップマイコンなどの低レベルデバイス上で，端末間が直接通信を行うピアツーピア (P2P) 型の連携機構に着目する．本手法では，機器間の連携機能を，連携ソフトウェアの分散を意識せず単一のソフトウェアモジュールとして記述する．処理の流れを解析することにより連携ソフトウェアを自動的に分散化し，連携に参加する機器にインストールする．本手

法を用いることにより，従来ノードごとに別々に開発していた連携ソフトウェアを，単一モジュールで開発することができ，開発者は連携ソフトウェアの分散を意識することなく容易に連携機能を開発できる．また，遠隔のノードのプログラムコードの更新を，一箇所で集中的に行えるため，機器間の連携構成を容易に変更できる．

目 次

図一覧

表一覧

第1章　はじめに

1.1　背景

近年，ハードウェア技術の進歩により，ＡＶ機器，センサ，文房具，家電製品など，様々な機器が高性能なコンピュータを内蔵して小型化し，至る所にコンピュータが存在するユビキタスコンピューティング環境が実現されつつある．また，無線LAN[1] や Bluetooth[2] などの短距離無線通信技術が広く普及しつつあり，これらの機器をネットワーク経由で通信させて，有用なアプリケーションサービスを実現することが期待される．

しかし，多数の機器を組み合わせたシステムは，一般に複雑なものとなってしまう．Weiser[3] は，このような複雑なコンピュータシステムのことを人間が意識しなくても，コンピュータが自律的に我々の生活を支援してくれるという，透明（invisible）なコンピュータシステムのビジョンを描いた．このようなビジョンを実現するために，図 1.1 に示すように，ユビキタスコンピューティングに関する様々な研究が行われている．

機器間の**連携**とは，複数の機器を用いたアプリケーションサービスを提供することである．複数の機器を連携させることにより，単一の機器のみでサービスを提供する場合と比べ，ユーザが自分の好きな機器をユーザインタフェースとして選択できるといった柔軟性や，新たに機器を追加することによって未知の機能を利用できるといった拡張性が向上する．

多数の機器が存在するユビキタスコンピューティング環境において，機器間連携を実現するためには，様々な課題が残されている．

まず，ユーザが望んでいる連携状態をシステム側に伝えるためには，一般にアドレスや名前の入力などの煩雑な設定が必要となる．例えば，無線ネットワーク上に多くの機器が存在するような環境において，２つの小型の無線機器間の連携を指示したい場合，小さなディスプレイ上で，多くの候補の中から接続先機器の名前を選択するというような設定は直観的ではない．

Follow-me application[4] は，ユーザの位置を超音波を用いた位置センサにより取得

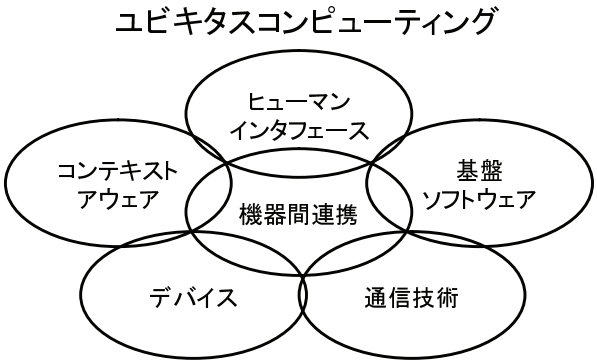

図 1.1: ユビキタスコンピューティングに関する研究分野

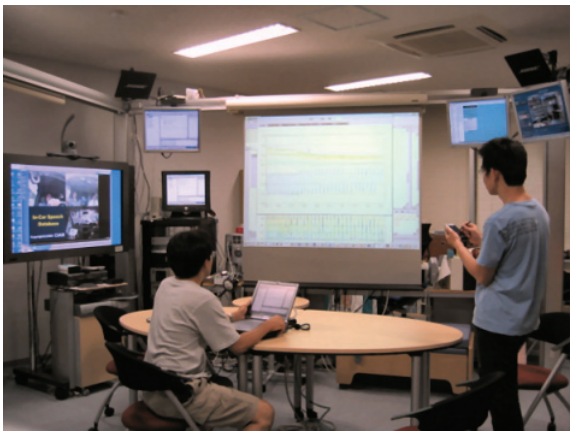

図 1.2: ユビキタスコンピューティング環境を想定した部屋 (Cogma Room)

し，ユーザに最も近いディスプレイが自動的に作業環境として選択される．STONE[5], AMIDEN アーキテクチャ[6] では，ユーザが連携の種類を選択すると，機器の種類に基づき，ネットワーク上から自動的に連携に用いる機器を検索する．しかし，多数の機器が存在する環境では，連携に用いる機器をシステム側が自動的に決定することには限界がある．例えば，図 1.2 は，ユビキタスコンピューティング環境を想定して我々が構築したテストベッドルーム [7] である．多数のディスプレイが存在するこのような環境では，ユーザが望んでいるディスプレイを自動的に決定することは難しい．従って，ユーザ自身が連携に用いる機器を直観的に指示できる手段が必要である．

一方で，連携のためのソフトウェアは複数の機器に分散するため，従来の連携ソフトウェア開発手法では，連携ソフトウェアの開発や保守が困難となる．Universal Plug and Play[8] や Jini[9] などに代表される RPC(Remote Procedure Call) 型の機器間連携フレームワークでは，端末間の通信プロトコルを隠蔽した stub クラスが自動的に生成され，開発者は分散を意識せずに連携ソフトウェアを記述できる．しか

し，一般に実行時に別途制御ノードが必要となり，また特定の連携に特化した通信効率の良いアプリケーションプロトコルを用いることが出来ない．

複雑なコンピュータシステムをより扱いやすくするためには，このような問題点を解決し，図 1.1 に示すように，ヒューマンインタフェース，コンテキストアウェア，基盤ソフトウェア，デバイス，通信技術など，複数の分野にまたがった，新たな機器間連携フレームワークを提案する必要がある．

1.2 本論文の目的

これらの問題点を踏まえ，本論文ではユビキタスコンピューティング環境における機器間連携フレームワークを提案する．特に，システムを容易に利用，管理できることを新たな目標として，利用者にとっての使いやすさと，開発者にとっての使いやすさの双方の観点に着目し，複数のアプローチを提案する．

利用者にとっての使いやすさを向上させるために，連携に用いる機器を指定する際に，アドレスや名前などの間接的なシンボルを指定するのではなく，機器そのものを直接指定できる手法を提案する．特に，２つの機器間を連携させることを**接続**と呼び，接続したい両機器を指定する際に，アドレスや名前ではなく機器そのものを直接指定できることを，**直接的な**接続指示と呼ぶ．

また，開発者にとっての使いやすさに着目し，事前に想定していない機器との連携も，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発・導入できるようにするために，連携ソフトウェアの自動分散化手法を提案する．特に，ワンチップマイコンなどの低レベルデバイス上で，機器間が直接通信を行うピアツーピア (P2P) 型の連携機構に着目する．

1.3 本論文の構成

本論文の構成を図 1.3 に示す．まず，2 章では，ユビキタスコンピューティング環境における機器間連携フレームワークに関する従来研究を挙げ，その問題点と本論文の位置づけを示す．

3 章では，比較的近くの機器に対する直接的な接続指示手法として，各機器にボタンを設け，接続したい両機器のボタンを押すだけで機器間の接続を直接的に指示できる Touch-and-Connect を提案する．本手法では，状態を表示可能なボタンを用いてユーザの操作を排他制御し，複数のユーザが独立に操作を行う状況において発

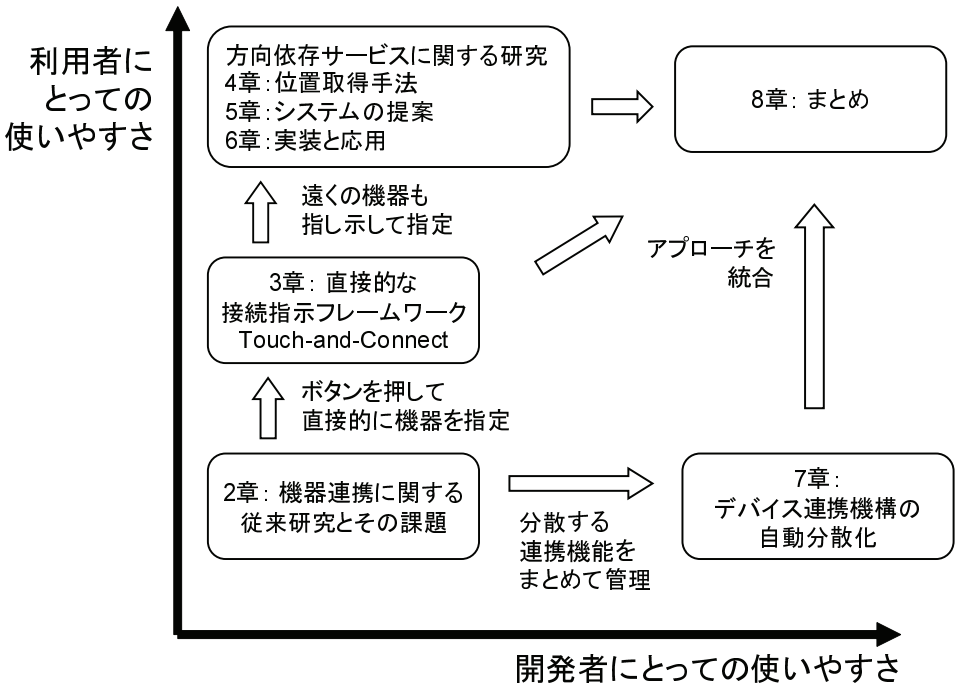

図 1.3: 本論文の構成

生しうる誤接続を防止する．

さらに，遠くの機器に対しては，携帯端末で機器を指し示すことにより直接的に指定できる手法を提案する．4 章では，指し示した機器の推定のために，ユーザの位置と方向に基づいたより高度な位置情報サービスである方向依存サービスを提案する．また，方向依存サービスにおいて安価に位置を取得するために，複数のマーカを手動で指し示すことによりユーザの位置を推定する手法を提案する．5 章では，広い範囲で方向依存サービスを利用可能とするために，無線 LAN の位置情報を用いた方向依存サービスシステムを提案する．6 章では，この方向依存サービスシステムのプロトタイプシステムの実装，および応用について述べる．

7 章では，単一のソフトウェアモジュールとして記述された連携ソフトウェアを，処理の流れを解析することにより自動的に分散化する手法を提案する．本手法を用いれば，開発者は分散を意識することなく容易に連携ソフトウェアを開発できる．

最後に 8 章では本論文をまとめる．提案したこれらのアプローチを組み合わせることにより，利用者および開発者の双方にとって，容易に利用できる機器連携フレームワークが実現する．本論文で提案するアプローチは，様々な用途に汎用的に応用可能であることを示す．また，さらなる理想的なフレームワークの実現のために，今後の課題を示す．

第2章　機器間連携フレームワークに関する研究動向

本節では，ユビキタスコンピューティング環境における機器間連携フレームワークに関連した研究動向について述べ，本論文の位置づけを示す．

機器間連携フレームワークに関する従来研究としては，以下のようなものが挙げられる．

2.1　ヒューマンインタフェース

ユーザが望んでいる連携状態をシステム側に伝えるためには，一般に煩雑な設定が必要となる．例えば，2つの小型の無線機器間の接続を想定した場合，小さなディスプレイ上で多くの候補の中から接続先機器の名前を選択するというような設定は直観的ではなく，このような煩雑な設定無しに，機器間を容易に接続できる手段が望まれる．すなわち，コンピュータシステムの複雑さをユーザに意識させることなく，ユーザの意図をシステムに容易に伝えられるヒューマンインタフェースが望まれる．

2.1.1　機器間連携の指示

ユーザの行動をセンサ等で認識し，機器間の接続などの設定を自動的に行う研究が行われている．例えば，Follow-me application[4] では，ユーザの位置をセンサにより取得し，ユーザに最も近いディスプレイが自動的に作業環境として選択される．しかし，多数の機器が存在する環境では自動設定には限界があり，ユーザが本当に望んでいる状態を，システムに容易に伝えられる手法が必要である．Reactive environments[10] では，ユーザが接続したい両機器に取り付けられたボタンを押すことにより，直接的に接続を指示できるが，複数のユーザが独立に操作を行う状況を想定していない．

2.1.2　画像認識による機器の指定

ユーザが直接的に機器を指定する手法として，画像認識技術を用いたものが提案されている．

InfoPoint[11] は，ID 情報を記述した 2 次元マトリックスバーコードを各機器に貼り付け，カメラを搭載した携帯端末でこれを撮影することにより，ユーザが指し示している機器を特定する．しかし，機器から遠く離れた場合や，バーコードを正面から撮影できない場合は，カメラの解像度の影響で機器の ID を正しく認識することが難しい場合がある．AirReal[12] では，ユーザはレーザポインタで機器を指し示す．部屋の側面に取り付けられたカメラで撮影した画像より，レーザポインタの光が照射されている位置を認識し，ユーザが指し示している機器を特定する．

しかし，これらの手法は，一般にカメラが撮影している場所・方向でしか使うことができない．このため，広い範囲で利用できるシステムを目指した場合，あらゆる場所・方向にカメラを配置する必要があり，システムの構築にはコストがかかる．また，カメラで撮影されるのはユーザにとっての心理的負担も高いと考えられる．

2.2　コンテキストアウェア

invisible なユビキタスコンピューティング環境の実現のために，ユーザの状態や行動などの状況（コンテキスト）をセンシングし，それに依存したサービスをコンピュータが自律的に提供する試みが多数行われている．コンテキストとして代表的なものが位置情報である．

2.2.1　位置情報サービスと位置取得技術

携帯電話などの携帯端末の普及や GPS モジュールの低価格化により，GPS 等により得られるユーザの位置情報を積極的に活用し，ユビキタスコンピューティング環境におけるユーザの生活を支援する研究が盛んである．例えば，ActiveCampus[13], PlaceLab[14], Locky.jp[15], Mobile Info Search[16], SpaceTag[17] などをはじめ，位置情報を用いた様々なサービスが提案されている．

位置を取得する方法としては，例えば GPS を用いる方法がある [18] が，衛星からの電波が届かないビル街や屋内などの場所での利用が困難である，電源投入後センサが利用できるようになるまでに時間がかかる (cold-start)，などの問題がある．屋内向けの測位技術として，位置が既知の超音波受信装置への超音波の到達時間を用

いた Active-BAT[4] などがあるが，至る所に超音波受信装置を配置した場合，環境側の設備が高価となってしまう．このように，屋内・屋外を問わず広い範囲で利用可能，かつ安価な位置測定手段は現状では存在しない．

2.3 基盤ソフトウェア

機器間連携のためには，連携ソフトウェアを動作させるための基盤ソフトウェアや，連携ソフトウェアの開発手法が重要である．

2.3.1 P2P 型フレームワーク

ネットワーク上の機器間を接続するための技術仕様としては，既に Universal Plug and Play[8]，ECHONET[19], Jini[9], HAVi[20] などが提唱されている．これらは，単一制御ノードが遠隔サービスノードを集中制御して連携を実現する Remote-Procedure-Call (RPC) 型のフレームである．このように，既存のシステムの多くは，一般に単一のマシンにより集中管理されており，モバイル環境や，必要に応じて一時的に構築されるアドホックネットワークでの利用は困難である．

一方で，AMIDEN[6], STONE[5] など，制御ノードを介さずに，機器間が直接通信して連携するピアツーピア (P2P) 型の基盤フレームワークが近年注目を集めている．RPC 型フレームワークと比較して，この P2P 型フレームワークは，別途制御ノードを必要としない，連携サービスに最適な通信プロトコルを使える，などの利点がある一方，連携に関するプログラムコードが複数端末間に分散し，その開発や保守は容易ではない．

2.3.2 アスペクト指向

複数のモジュールに横断する機能を抽出し，ソフトウェア開発時にはアスペクトと呼ばれる単一のモジュールとして記述するという，アスペクト指向プログラミングが近年注目を集めている．アスペクトはコンパイル時に実際のモジュールに埋め込まれる (weaving)．AspectJ[21] は，Java 上でこのアスペクト指向プログラミングを行うための代表的なツールである．また，広義のアスペクト指向言語である MixJuice[22] は，クラス間の連携処理（コラボレーション）を抽出しモジュール化して扱える．

しかしこれらは，単一マシンでの実行を想定したものであり，分散した機器間の連携処理の記述にそのまま適用することは難しい．一方で，アスペクト指向の考え

方を分散環境へ適用する試みも行われている．Remote-pointcut[23] は，遠隔の端末上のソフトウェアに対してアスペクトを適用することにより，複数の端末をまたがった連携処理を記述できる．しかし，分散を意識してアスペクトを記述する必要があり，連携ソフトウェアをより容易に開発できる手法が求められる．

2.3.3　プログラムの自動分散化

一方，分散を意識せずに書かれたソフトウェアを，自動的に分散化する機構も提案されている．

RMI[24] は，遠隔端末に存在するオブジェクトに対するメソッドの呼び出しを，ローカル端末上のオブジェクトと同様の記述で行える，RPC(Remote-Procedure-Call) のフレームワークである．端末間の通信プロトコルを隠蔽した stub クラスが自動的に生成される．しかし，一般に遠隔オブジェクトの生成や検索をハードコーディングする必要があり，機能の配置を柔軟に変更することが難しい．

J-Orchestra[25]，Addistant[26], Jacross[27] は，機能の配置を分散ポリシーファイルとして別に記述でき，単一マシンの場合と同様に書かれたソフトウェアを，自動的に分散化する．しかし，分割の単位はクラス単位・メソッド単位であり，メソッドの中身をより細粒度に分割できる自動分散化手法が望まれる．

また，これらの手法は一般的に JavaVM[28] を対象としたものであり，JavaVM を動作させることが困難な低レベルデバイスへの適用は難しい．

一方，並列コンピュータ向けの自動並列化コンパイラとして，High Performance Fortran (HPF)[29] などが存在するが，これらは一般に巨大なデータを分割して，同一のコードを並列動作させるものであり，コードを分割して異なる役割の端末同士を連携させるような用途には適していない．

2.4　通信技術・デバイス

近年，無線 LAN[1], Bluetooth[2], ZigBee[30] などの短距離無線通信技術が広く普及しつつあり，多くの身近な機器がこれらの無線通信機能を内蔵しつつある．

2.4.1　低レベルデバイス向けのソフトウェア開発環境

MOTE[31]，nRF24E1[32] など，無線通信機能を搭載した低コストな端末が開発されつつあり，センサ・家電・文房具などの様々な小型機器がネットワークに参加

することが期待される．これらのデバイスは，CPU の速度やメモリ容量が制限された**低レベルデバイス** (例えば，nRF24E1 の場合，CPU は 8bit, RAM は 4Kbytes) であり，JavaVM 等を想定した既存のソフトウェア開発手法をそのまま適用することは難しく，低レベルデバイス向けのソフトウェア開発環境が望まれる．

Simple Control Protocol (SCP)[33] は，機器の機能をネットワーク上に公開して連携させるための，低レベルデバイス向けフレームワークである．しかし，機器上のソフトウェアは固定的であり，予め定められたアプリケーションプロトコルの範囲内でしか連携を行うことが出来ない．

AhroD（ユビキタスチップ）[34] は，複数の低レベルデバイス間の連携動作を，ECA (Event，Condition，Action) ルールにより記述できる．しかし，ECA ルールのみでは記述内容に限界があり，連携動作を記述するためのより強力なプログラミング言語が求められる．

センサネットワークの分野でも，低レベルデバイス向けのソフトウェア開発環境に関する研究が盛んである．Mate[35] および文献 [36] の手法は，センサノードに VM を搭載することにより，遠隔デバイスのプログラムコードを無線経由で書き換えることができる．また，C 言語を拡張した nesC[37] は，低レベルデバイスに適したコンポーネントモデルを採用しており，サービス実現に必要な機能を組み合わせて，静的な最適化を用いたコンパクトなコードを生成できる．しかし，これらのフレームワークは，ノード間を直接ピアツーピアで連携させる状況には着目しておらず，ノード単位でソフトウェアを開発する必要があるため，ソフトウェアの開発や保守が煩雑となる．従って，低レベルデバイスにおいても，2.3.3 節で述べたような自動分散化機構などを用いて，機器間の連携処理を容易に記述できるフレームワークが望まれる．

2.5 本論文の位置付け

機器間連携フレームワークに関する幾つかの従来研究，および本論文のアプローチの性質をまとめると，表 2.1，表 2.2 のようになる．

表 2.1 は，利用者にとっての使いやすさという観点で代表的な研究を比較したものである．これら全ての性質を同時に満たすのはチャレンジングな課題であるが，3 章で提案する Touch-and-Connect フレームワーク および 5 章で提案する Azim システムを組み合わせることにより，これらの性質をバランスよく満たしたフレームワークを実現できる（6.2.1 節参照）．

表 2.1: 利用者にとっての使いやすさの比較

性質 / 研究	機器を直接的に指定可能	複数の人間が独立に操作可能	デバイス・タグ等の携帯が不要	アドホックネットワークでの利用も考慮	遠くの機器も直接的に指定可能	安価にデプロイ可能
Jini	×	○	○	×	×	○
Reactive Environments	○	×	○	×	×	○
InfoPoint	○	○	×	×	×	○
AirReal	○	×	×	×	○	×
Follow-me Application / Active-BAT	×	○	×	×	×	×
Touch-and-Connect（提案手法）	○	○	○	○	×	○
Azim（提案手法）	○	○	×	×	○	○

表 2.2: 開発者にとっての使いやすさの比較

性質 / 研究	連携時に制御ノードが不要	連携動作を単一のモジュールで開発可能	命令単位での細粒度の自動分散化	低レベルデバイスへの適用	ソフトウェアを遠隔でアップデート
RPC型連携 (UPnP等)	×	○	×	×	×
P2P型連携 (AMIDEN等)	○	×	×	×	×
SCP	○	×	×	○	×
TinyOS / nesC	×	×	×	○	×
Mate	×	×	×	○	○
J-Orchesta / Addistant	○	○	×	×	×
デバイス連携機構の自動分散化（提案手法）	○	○	○	○	○

一方，表 2.2 は，開発者にとっての使いやすさという観点で代表的な研究を比較したものである．7 章で提案する，連携ソフトウェアの自動分散化手法は，これらの性質を満たしており，開発者にとって使いやすいフレームワークであると言える．

本論文で提案するこれらのアプローチを全て組み合わせれば，ここで挙げた多くの性質を満たした，利用者および開発者の双方にとって，容易に利用できる機器間連携フレームワークが実現できる．

第3章　機器間の直接的な接続指示手法 Touch-and-Connect

近年，様々な機器が高性能なコンピュータを内蔵し，至る所にコンピュータが存在するユビキタスコンピューティング環境が実現しつつある．しかし，機器間の接続を指示するためには，一般にアドレスや名前の指定などの煩雑な設定が必要となる．

本章では，接続したい両機器のボタンを押すことにより，機器間の接続を直接的に指示できるフレームワーク Touch-and-Connect を提案する．本手法では，状態を表示可能なボタンを用いてユーザの操作を排他制御し，複数のユーザが独立に操作を行う状況においても誤接続を防止する．ブロードキャスト通信を用いた本手法のプロトコルは，管理サーバを必要とせず，動的な端末の入退出への対応も考慮しているため，必要に応じて一時的に構築されるアドホックネットワークでも利用可能である．本フレームワークのプロトタイプシステムを実装してその実現可能性を示し，被験者実験によりその使いやすさを評価した．

3.1　接続指示フレームワーク

本節では，機器間の接続指示フレームワーク Touch-and-Connect を提案する．本手法を用いれば，ユーザは接続したい両機器に直接触ることにより，アドレスや名前を指定することなく，様々な機器間の接続を容易に指示できる．機器に直接触る必要があるため，手の届く範囲にある比較的近くの機器への適用を１つの利用シーンとして想定する．

本フレームワークでは，Bluetooth や無線 LAN 等の短距離無線通信をネットワーク層として仮定する．赤外線通信と比較すると，短距離無線通信を用いることにより，自由な位置で機器を利用でき，また３台以上の機器間の接続，多対多の接続など，より多くの機器との接続を実現できる．また，基地局型の無線ネットワークだけでなく，アドホックネットワークでも利用可能とすることを，本フレームワークの要件とする．ただし，比較的近くの機器間の通信を想定しているため，マルチホップのメッセージは想定せず，全機器間は直接通信可能とする．アドホックネットワー

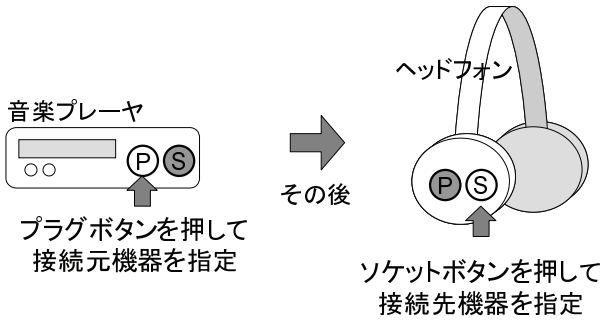

図 3.1: ボタン 2 つ版インタフェース

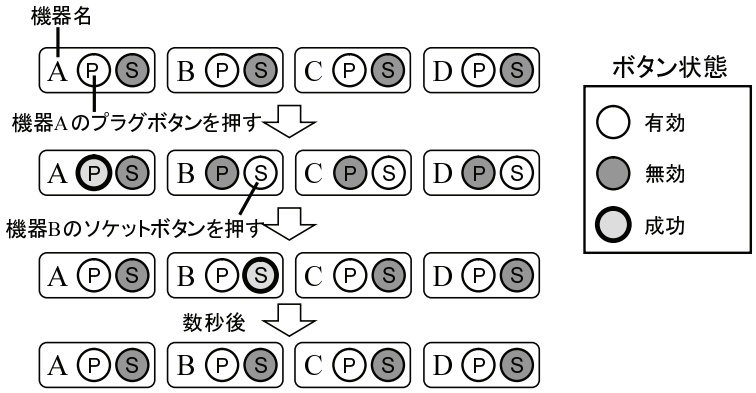

図 3.2: ボタンの状態遷移例

クとは，必要に応じて一時的に構築されるネットワークであり，(1) 管理サーバや事前の設定を前提としない，(2) ネットワークを構成する端末が，端末の動的な入退出により頻繁に変化する，などの特徴がある．アドホックネットワークでも利用可能とすることにより，いつでも，どこでも，提案する接続指示手法を使うことができる．例えば，外出先でモバイル機器間を接続する場合や，事前に設定がないノート PC 間を接続するような場合にも利用可能となる．

3.1.1　ロックメカニズム

本フレームワークでは，ユーザが接続したい機器を指示するためのインタフェースを，接続指示インタフェースと呼ぶ．各機器に 2 つのボタンを設けたシンプルな接続指示インタフェースである，ボタン 2 つ版インタフェースを用いて，本フレームワークを説明する．他の種類の接続指示インタフェースについては 3.1.5 節で述べる．

本インタフェースでは，図 3.1 のように，各機器にプラグボタン（P）とソケット

ボタン（S）を設ける．機器 A（例えば携帯音楽プレーヤ）と機器 B（例えばヘッドフォン）を接続したい場合には，まず，機器 A のプラグボタンを押すことによってそれが接続元機器であることを指示し，その後，機器 B のソケットボタンを押すことによってそれが接続先機器であることを指示する．プラグをソケットに差し込むというメタファにより，ユーザはこの操作を容易に理解することができる．

ここで，複数のユーザが独立に操作する状況を想定し，以下のような順番でボタンが押された場合を考える．

1. ユーザ 1 が機器 A のプラグボタンを押す
2. ユーザ 2 が機器 C のプラグボタンを押す
3. ユーザ 2 が機器 D のソケットボタンを押す
4. ユーザ 1 が機器 B のソケットボタンを押す

この場合，システム側は誰がボタンを押したのかを認識できないため，ユーザ 1 は機器 A と B の，ユーザ 2 は C と D の接続を意図しているにもかかわらず，A と D，C と B が接続されてしまう可能性がある．

無線ネットワークでは，このように他人の操作との干渉が発生する可能性がある．すぐ近くの他人の場合は，コミュニケーションを取って解決することも可能であるが，無線電波は壁をつきぬけるため，見えない隣の部屋の接続操作が干渉する場合もある．また，会議室などで多数のユーザが一斉に接続を行う状況では直接のコミュニケーションには限界がある．このような誤接続を防止する方法のひとつとして，ボタンを押しているユーザを認識する方法が考えられる．しかし，ユーザの識別のためには，例えば ID を持ったデバイスをユーザが携帯する必要があったり，ID を読み取るために特別なセンサが必要であったりする．

本手法では，ユーザの操作間の排他制御，すなわち，あるユーザが操作をしている間，他のユーザの操作を禁止することにより，ユーザを識別することなく誤接続を防止する．このために，状態を表示可能なボタンを用いたロックメカニズムを導入する．各ボタンは「有効」,「無効」,「成功」のいずれかの状態を表示できる．ユーザは，ボタンが有効状態の時にのみ押すことができ，押した後，成功状態になったことを確認する．この成功状態により，ユーザは自分の操作の成功を確認できる．また，あるユーザが接続操作を行っている間は，他の全ての機器はロックされ，プラグボタンが無効状態となる．これにより，誤接続の原因となる他人の操作の開始を防止できる．先ほどの例の場合，ユーザ 1 の操作中は，A 以外の全機器のプラグボ

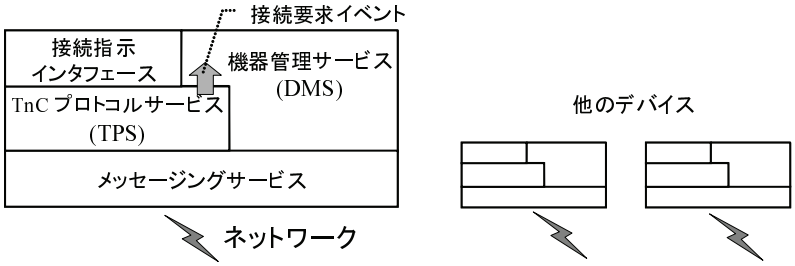

図 3.3: フレームワーク構成

タンが「無効」となり，ユーザ 2 は操作を開始できない．この場合のボタンの状態遷移例を図 3.2 に示す．

また，操作中に手動または自動で接続指示をキャンセルする機能がある．(1) プラグボタンを押して接続元を指定した後，再度プラグボタンを押すことにより，接続先の指定をキャンセルすることができる．(2) ロックしたまま放置された場合に対応するため，プラグボタンを押してから一定時間経過後，自動的にキャンセルされる．

また，接続元専用・接続先専用の機器も考えられる．この場合，片方のボタン（プラグボタンまたはソケットボタンのどちらか）のみを機器に取り付ければよい．

これらの機能を実現するプロトコルの詳細は，3.1.4 節に示す．

3.1.2　フレームワーク構成

図 3.3 に本フレームワークの構成を示す．本フレームワークは，メッセージングサービス，接続指示インタフェース，Touch-and-Connect プロトコルサービス（TPS: Touch-and-Connect Protocol Service），機器管理サービス（DMS: Device Management Service）の 4 つのモジュールからなる．

メッセージングサービス

メッセージングサービスは，様々なネットワーク媒体やネットワークプロトコルの違いを隠蔽する．また，各ソフトウェアサービスに対してユニークな識別子を割り当て，ソフトウェアサービス間の非同期メッセージ通信機能を提供する．DMS と TPS は，他の機器と通信するために，このメッセージングサービスを使う．

接続指示インタフェース

接続指示インタフェースは，本フレームワークにおいて，ユーザが機器を指定するためのヒューマンインタフェース，およびそれを管理するソフトウェアサービスである．前述したボタン２つ版インタフェースだけでなく，ボタン１つ版のインタフェースなども存在する（3.1.5 節参照）．

Touch-and-Connect プロトコルサービス（TPS）

Touch-and-Connect プロトコルサービス（TPS）は，3.1.4 節で述べるプロトコルを実装しており，本フレームワークの中心となるサービスである．TPS は，接続指示インタフェースの状態（ボタンの色等）の管理や，ユーザからの入力の処理を行い，また接続操作が行われた際には DMS に接続要求イベントを通知する．

機器管理サービス（DMS）

機器管理サービス（DMS）は，本フレームワークのアプリケーションレイヤーである．DMS は機器固有の機能を管理し，その実装は機器の種類によって異なる．

本フレームワークが DMS に対して要求する機能は以下のものである．

- DMS は**機器情報**を提供する．機器情報はデバイスの性質を表したデータ構造であり，詳細については 3.1.3 節で述べる．
- DMS は**接続要求**イベントを受け取り，それに応じた適切な動作を実行する．接続要求イベントは，ユーザが接続を指示した際に TPS から通知される．

一般に，DMS は機器間の接続状態を管理する機能を持つが，どのように接続状態を管理するかは本フレームワークでは規定していない．例えば，各機器が接続相手のアドレスのリストを持ち，接続要求イベントを受け取った際に，接続相手のアドレスをリストに追加する．既にアドレスが存在する場合には削除すれば，切断操作も可能になる．

接続要求イベント

ユーザが接続指示インタフェースにより２つの機器間の接続を指示した場合，接続する両機器各々において，接続要求イベントが TPS から DMS へ通知される．接続要求は以下の情報からなる．

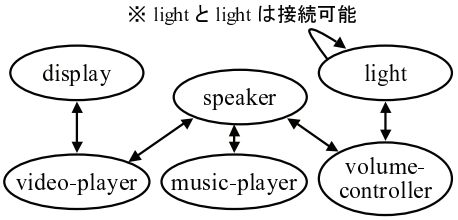

図 3.4: 接続可能性の例

```
bidirectional-link: org.cogma.tnc.display, org.cogma.tnc.video-player
bidirectional-link: org.cogma.tnc.speaker,
                    org.cogma.tnc.volume-controller
bidirectional-link: org.cogma.tnc.speaker, org.cogma.tnc.music-player
bidirectional-link: org.cogma.tnc.speaker, org.cogma.tnc.video-player
self-link: org.cogma.tnc.light
```

※ self-link は，同一タイプ間のリンクを意味する

図 3.5: 接続可能性を記述した設定ファイル

- 接続相手の**機器情報**
- **接続元フラグ:** 自分が接続元（プラグボタンにより指定された機器）であるかどうか．この情報は，接続の方向により動作を変更する場合に用いる．

3.1.3　機器情報

本フレームワークでは，各機器は機器情報というデータを持つ．機器情報は，アドレス，機器の Type，グループ ID，拡張情報からなる．機器情報は，TPS において，機器間の接続可能性の判断するために，また DMS において，接続要求時に適切な連携動作を自動的に選択するために用いられる．

機器の Type と接続可能性

本フレームワークは，様々な種類の接続に対して一般的に用いることができる．接続する 2 つの機器が指定されて接続要求イベントが発生した際に，適切な連携動作の種類（機器間連携に用いるアプリケーションプロトコルの種類など）を DMS において自動的に選択するために，各機器に Type（機器の種類）を定義する．Type は，機器の種類を表す文字列によって定義できる．例として，light（ライト），video-player（ビデオプレーヤ），music-player（音楽プレーヤ），display（表示装置），speaker（ス

ピーカ），volume-controller（ボリュームコントローラ）などの Type を考える．接続要求時には，接続する２つの機器の Type に基づいて適切な動作が自動的に選択される．例えば，video-player と display を接続する場合には映像伝送が，video-player と speaker を接続する場合には音声伝送が選択される．

使いやすさを考慮すると，接続操作中にロックを行う機器は出来る限り少ないことが望ましい．本手法では，存在しうる機器の Type の集合とそれらの接続可能性を事前に限定することにより，ロックが必要な機器の Type を制限する．これにより，プラグボタンが押された時に，全ての種類の機器をロックするのではなく，必要最小限の種類の機器のみをロックできる．ここで,「A から B へ接続可能」とは，機器 A から機器 B への接続要求（A のプラグボタンを押し，B のソケットボタンを押すこと）に対して，適切な動作が定義されていることを意味する．接続可能性は，２つの Type 間の関係として定義する．接続可能性の定義例を有向グラフに表したものを図 3.4 に示す．なお，ここでの「接続可能性の方向」は，単に「ボタン操作の順番」であり，通信の方向（実際の連携動作時に，情報がどちら方向に伝えられるか）とは関係ない．

接続操作中，プラグボタンが押された機器（接続元機器）から接続可能な機器のみが，ソケットボタンが有効状態となり接続先として指定できる．また，この「接続可能な機器」へ接続可能な機器のみがロックの対象となる．例えば，先ほどの定義の場合，light のプラグボタンが押された場合は，light から接続可能な light, volume-controller のプラグボタンのみが有効となり，またこれらに接続可能な light, speaker, volume-controller のみがロックの対象となる．

Type 間の接続可能性を判断できるようにするために，接続可能性を記述した図 3.5 のような設定ファイルを各機器に持たせる．名前の衝突を防止するために，機器の Type 名は，この図の例のように Type を定義した組織のドメイン名を含むべきである．また，今後の発展としては，将来作成されたデバイスとの接続性を確保する方法として，設定ファイルを動的に更新できるような仕組み，例えば，ネットワーク内に更新情報をブロードキャストしたり，インターネットに接続可能であれば，サーバで更新情報を配布したりするような仕組みの導入が挙げられる．

グループ ID

本手法では，プラグボタンを押した後，悪意のある他人にソケットボタンを押されることにより，意図しない接続が行われてしまう可能性がある．また，多数の接続が行われる場合には，ロックが解除されるまで長時間待たされる可能性がある．

これらの問題に対応するため，本フレームワークでは，グループを作成して接続可能な機器を制限することができる．同じグループに属する機器間のみが接続可能，およびロックが必要となる．グループを実装するために，各機器はグループ ID を持つ．同一のグループ ID を持つ機器が同じグループに属すると見なされる．

機器は初期状態では「パブリックグループ」という特別なグループに属する．機器の属するグループを変更したい場合は，機器のグループ ID を変更する必要がある．グループ ID を設定するためのユーザインタフェースとしては，ボタンなどにより直接 ID を入力する方法の他に，より簡単な方法としてグループチェンジャを利用することもできる．グループチェンジャは，グループ ID を変更する目的で使用する特別な機器であり,「設定用 ID」を持っている．設定したい機器からグループチェンジャへの接続指示を行うと，グループチェンジャの設定用 ID が接続元機器のグループ ID として設定される．従って，グループ ID を入力することなしにグループ ID の設定を行うことができる．

拡張情報

拡張情報は機器の Type に依存した情報である．接続要求時に DMS に通知される接続要求イベントのパラメータに，接続相手の拡張情報が含まれており，DMS において機器間の連携を実現するために用いられる．拡張情報の具体的な内容はアプリケーション依存であるが，例えば，アプリケーションプロトコル通信用のアドレスやポート番号等などである．

3.1.4　Touch-and-Connect プロトコル

本節では，Touch-and-Connect プロトコルサービス（TPS）で実装されるプロトコルについて述べる．本プロトコルは，必要に応じて一時的に構築されるアドホックネットワークでも利用可能とするため，管理サーバの不使用，および動的な端末の入退出への対応を要件としている．サーバでの集中管理の代わりに，ブロードキャスト通信を用いた状態の分散管理を行う．また，ブロードキャストメッセージの送信を定期的に繰り返すことにより，動的な端末の入退出に対応する．

本プロトコルのシーケンス図を図 3.6 に示す．プラグボタンが押され接続元として指定された機器（リクエスタ）は，接続可能な他の全機器に要求（**リクエスト**）を行い，ソケットボタンが押され接続先として指定された機器（リプライヤ）が，このリクエストに応答（**リプライ**）することにより，接続操作が確定する．またこの

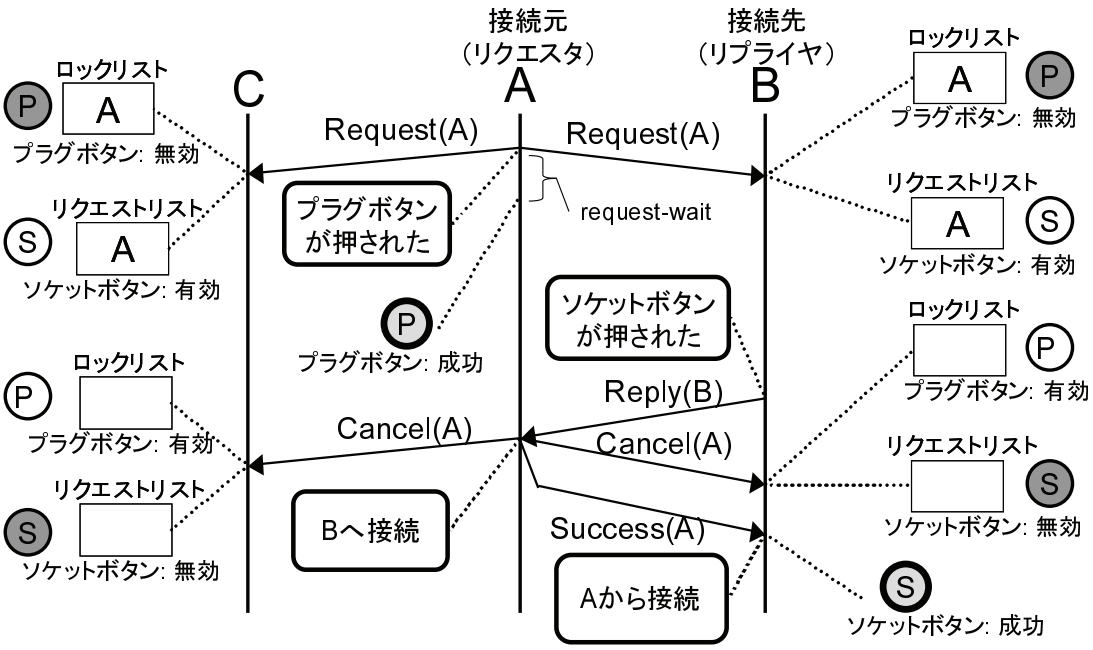

図 3.6: プロトコルのシーケンス図

際，リクエスタは，他の機器を**ロック**してプラグボタンを無効とし，他人の操作開始を禁止することにより，誤接続を防止する．

各機器はリクエストリストとロックリスト（初期状態は空）を持つ．リクエストリストには，現在自分にリクエストしている機器のアドレスが格納される．ロックリストには，現在自分をロックしている機器のアドレスが格納される．

以下，プロトコルを述べる．時間パラメータとして，request-wait, cancel-wait, success-time （デフォルト値は後述）がある．プロトコルを２つの並列状態遷移図で表したものを，図 3.7 に示す．

リクエスト

プラグボタンが押された機器（リクエスタ）は，Request メッセージをブロードキャストする．接続可能性を判断するために，このメッセージにはリクエスタの機器情報（拡張情報を除く）が含まれる．Request メッセージを受信した機器は，それが自分へ接続可能な機器からのメッセージであれば，そのメッセージをリクエストリストに追加する．

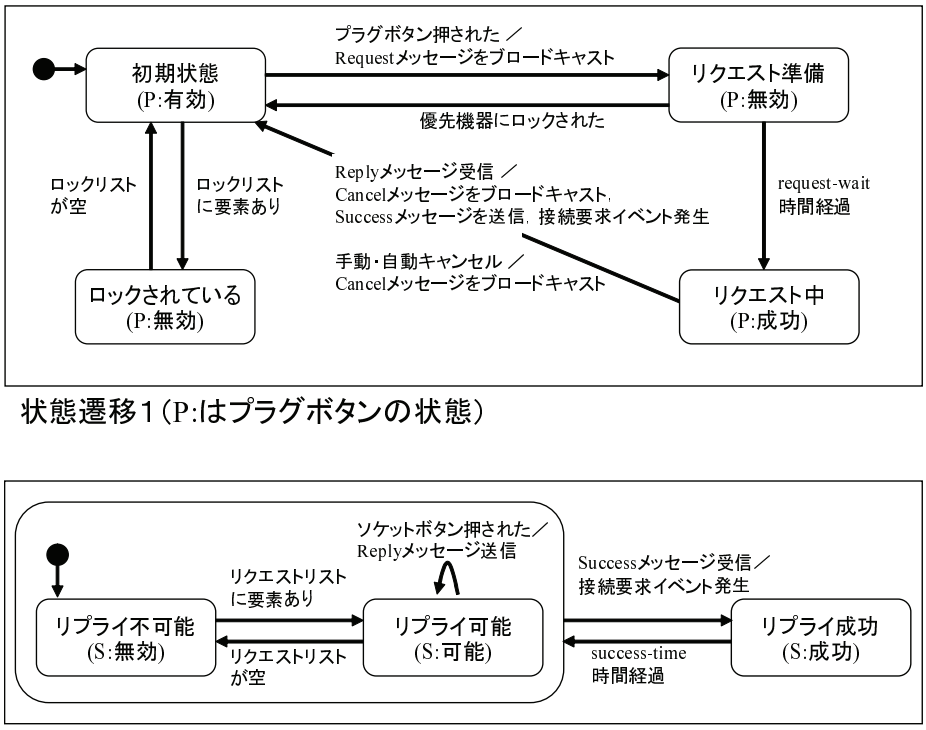

図 3.7: プロトコルの状態遷移図

ロック

Request メッセージを受信した機器は，それがロックされるべき機器からのメッセージであれば，そのメッセージをロックリストに追加する．

リプライ

ソケットボタンが押された機器（リプライヤ）は，リクエストリストの先頭要素の送信元に対して，Reply メッセージを送信する（リプライ）．

ロックが正しく行われていれば，リクエストリストの要素は 1 つである．パケットが届かなかった場合や，ネットワーク構成が動的に変化した場合などは，リクエストリストの要素が複数になる可能性がある．要素が複数の場合は，どれにリプライすればよいのかがあいまいとなり，誤接続が発生する可能性があるため，その旨をユーザに通知するべきである．

接続操作の完了

リクエスタがReplyメッセージを受信すると，リプライヤと接続することを認識する．リクエストおよびロックをキャンセルするために，Cancelメッセージをブロードキャストする．また，接続することを通知するためにリプライヤへSuccessメッセージを送信する．

Cancelメッセージを受信した機器は，リクエストリストおよびロックリストから，送信元機器の要素を削除する．

Reply, Successメッセージには，送信元機器の機器情報が含まれている．機器がこれらのメッセージを受信した際に，この機器情報を含んだ接続要求イベントが，TPSからDMSへ通知されることにより，接続を実現する．

手動キャンセル

ユーザによる手動キャンセルを実現するために，リクエスタは，リクエスト中に再度プラグボタンが押された場合は，Cancelメッセージをブロードキャストして初期状態に戻る．

自動キャンセル

ロックしたまま放置された場合に対応するため，リクエスタは，リクエスト後cancel-wait で設定された時間が経過した場合は，自動的にCancelメッセージをブロードキャストして初期状態に戻る．

リクエストの衝突への対応

複数の機器のプラグボタンが同時に押された場合，ロックが行われる前に複数のリクエストが発生する可能性がある（リクエストの衝突）．これは，リクエストリストに複数の要素が存在することになり，誤接続の原因となるため望ましくない．これに対処するため，機器にはアドレスにより優先順位が決めてあり，最も優先する機器からのリクエストのみを有効とする．

リクエストの衝突を検出するために，リクエスタは，リクエスト後request-wait時間の経過を待つ（リクエスト準備状態）．その後，はじめてリクエストが有効となりプラグボタンが成功状態となる．この待ち時間中に，より優先する機器からロックされた場合は，そのリクエスタはCancelメッセージをブロードキャストし，初期

状態に戻る．

ボタン２つ版インタフェースの状態表示

ロックリストが空の場合，新たなリクエストの発生が可能であり，プラグボタンは有効状態となる．要素がある場合，リクエストの発生が禁止されているロック状態であり，プラグボタンは無効状態となる．また，リクエストリストが空の場合，リプライが不可能なため，ソケットボタンは無効状態となる．要素がある場合，リプライが可能なため，ソケットボタンは有効状態となる．ボタンが無効状態の場合は，押すことはできず，押されても無視する．ただし，リクエストリストの要素が複数の場合は，そのままソケットボタンを押すと誤接続が発生する可能性があるため，点滅などの表示でユーザに警告するべきである．

接続元指定の成功を表示するために，リクエスタは，リクエスト後 request-wait 時間が経過してからキャンセルする（Cancel メッセージをブロードキャストする）までの間，プラグボタンが成功状態となる．また，接続先指定の成功を表示するために，リプライヤは，Success メッセージを受信してから success-time で設定された時間の間，ソケットボタンが成功状態となる．

送信の繰り返し

動的な端末の入退出やパケットの損失に対応するため，Request メッセージには，有効時間が定められている．リクエストが有効な間，リクエスタは Request メッセージの送信を定期的に繰り返す．また，Cancel メッセージも，パケットの損失に対応するために，定期的に 3 回程度送信する．本フレームワークによるネットワークトラフィックの増加を抑えるため，現在発生中のリクエストを全ての端末が管理し，その数が増加するにつれ，Request,Cancel メッセージの送信間隔も長くする．

セッション ID

A を接続元とした接続操作の直後に，再度連続して A を接続元とした接続操作が行われる場合を考えると，パケットの到着順が変わり，新しい Request(A) が，直前の Cancel(A) によって無効になってしまう可能性が考えられる．そこで，リクエストからキャンセルまでの一連の流れ（セッション）に，毎回異なるセッション ID を割り当て，セッション ID を各メッセージに含める．現在有効なリクエストとセッション ID が異なる Cancel メッセージは無視される．

表 3.1: ボタン１つ版インタフェースの動作（より上のルールが優先）

条件	ボタン状態（とその意味）	押された時の動作
リクエスト中（自分のリクエストが有効な間）	接続元指定成功（接続元機器の指定が成功）	手動キャンセル
リプライ成功（Success メッセージ受信後一定時間）	接続先指定成功（接続先機器の指定が成功）	
リプライ可能（リクエストリストに要素がある）	接続先指定（接続先機器の指定が可能）	リプライ
ロックされている（ロックリストに要素がある）	無効（押すことは出来ない）	
デフォルト（初期状態）	接続元指定（接続元機器の指定が可能）	リクエスト

時間パラメータのデフォルト値

時間パラメータのデフォルト値は，request-wait = 300 ミリ秒, cancel-wait = 30 秒, success-time = 20 秒 とする．

3.1.5 他の接続指示インタフェース

本フレームワークでの最も標準的な接続指示インタフェースは，3.1.1 節で示したボタン２つ版インタフェースである．しかし，他の形態の接続指示インタフェースも考えられる．ここでは，ボタン１つ版インタフェースについて述べる．

ボタン１つ版インタフェース

ボタンの状態表示を工夫することにより，各機器に対してボタン１つのみで本手法を実現することができる．従来の２つのボタンの状態を，１つのボタンで表現するために，ボタンには表 3.1 に挙げる 5 つの状態を設ける．ユーザは，まず，接続元機器を指示するために，接続元指定状態のボタンを押し，続いてボタンが接続元指定成功状態になったことを目視で確認する．この際，ボタンが接続元指定状態ではない場合はそうなるまで待つ．その後，接続先機器を指示するために，接続先指定状態のボタンを押し，ボタンが接続先指定成功状態になったことを確認する．

ボタンの状態変化と動作は，基本的には表 3.1 のルールに従う．ただし，複数のユーザがほぼ同時に操作を開始するような場合への対処が必要である．あるユーザが操作を開始した直後，ボタンが接続元指定状態から接続先指定状態に変わったこ

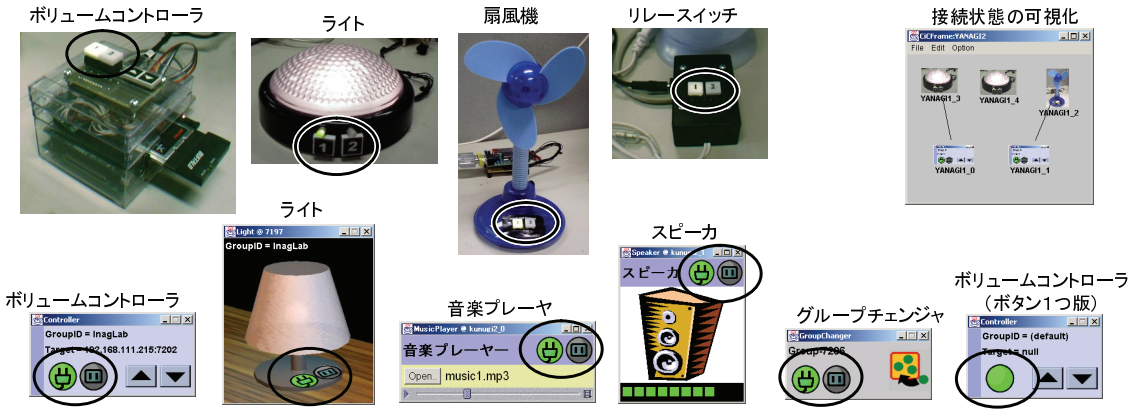

図 3.8: 実装した機器

とに気づかずに，別のユーザが誤ってボタンを押してしまい，誤接続が発生する可能性がある．従って，接続元指定状態から接続先指定状態へ切り替わる際に，一定時間，ボタンを無効状態（操作不可能）とすることにより，状態の切り替わりをユーザに認識させる．すなわち，表の 3 行目のルールにおいて，リプライ可能となった後，change-mode-wait（デフォルト値は 1 秒）で定められた時間が経過するまでの間は無効状態とし，その後に接続先指定状態とする．

ボタン 2 つ版と比較すると，ボタン 1 つ版は，インタフェースの部品コストや実装面積を削減できるという利点がある．一方，ユーザが change-mode-wait の間にボタン状態の切り替わりを認識できなかった場合に，誤操作による誤接続が発生する可能性がある．

3.2　実装

本手法に基づくプロトタイプシステムを Java（Personal Java 1.1.3）を用いて実装した．端末間の通信には無線 LAN（IEEE802.11b）を使用した．本プロトタイプシステムは，ノート PC，PDA（SHARP SL-C750），組み込み用の小型 Linux PC（LAMB-EM-01）などで動作を確認した．[38][39]

接続指示インタフェースを内蔵した機器として，図 3.8 に示すような，複数のソフトウェアによるエミュレータ，およびハードウェアを作成した．図中の円の部分が接続指示インタフェースである．上下ボタンによる操作が可能なボリュームコントローラを，対象物（ライト，扇風機，リレースイッチ）に接続し，明るさ，風速，電源スイッチなどを制御することができる．ボリュームコントローラの対象物同士を接続することにより，ボリューム値の変化（明るさの変化など）を連動させること

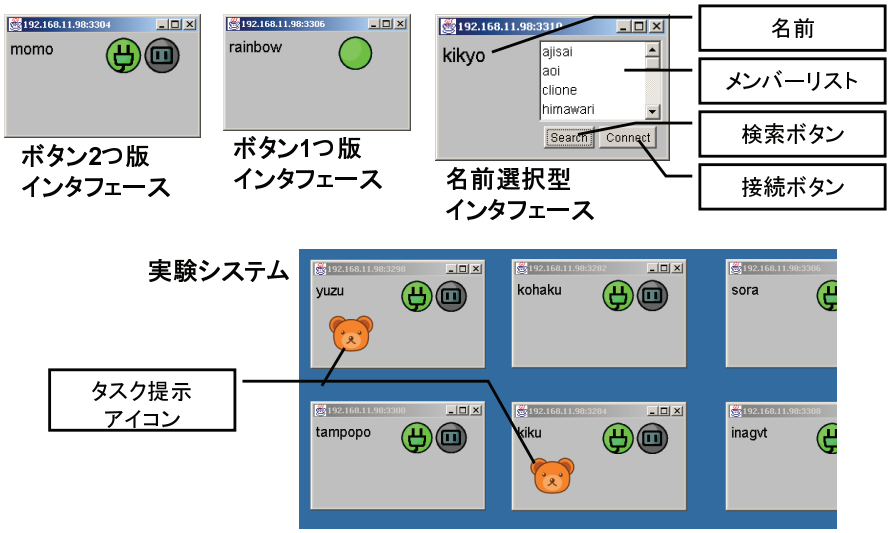

図 3.9: 実験システム

ができる．音楽プレーヤとスピーカを接続することにより，音楽プレーヤで再生された音楽を，スピーカ（の起動している端末）で再生することができる．また，任意の機器をグループチェンジャへ接続することにより，容易にグループ ID の設定を行うことができる．

Touch-and-Connect のボタンインタフェースの状態は色によって表示した．ボタン２つ版のインタフェースでは，緑が有効，赤が成功，黒（消灯）が無効を表す．ボタン１つ版のインタフェースでは，緑が接続元，赤が接続先を表し，点滅により成功を表す．

また，DMS に対して現在の接続状態を保持するような拡張を行い，接続状態を可視化できるようにした．図 3.8 右上の接続状態の可視化ソフトウェアは，ネットワーク上に存在する機器と，それらの機器間の接続状態を表示する．

3.3 被験者実験による評価

接続指示手法は，ユーザにとって使いやすいこと，また多数のユーザや機器がネットワークに存在する環境を考慮していることが望ましい．我々は，以下のような被験者実験を行い，これらの性質を評価した．

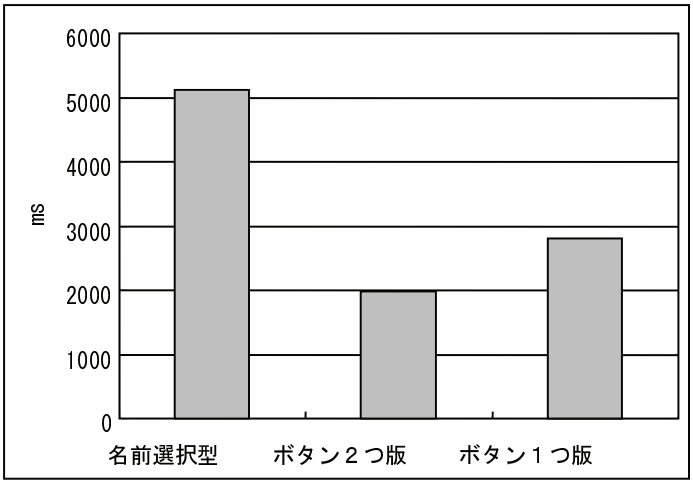

図 3.10: 平均操作時間

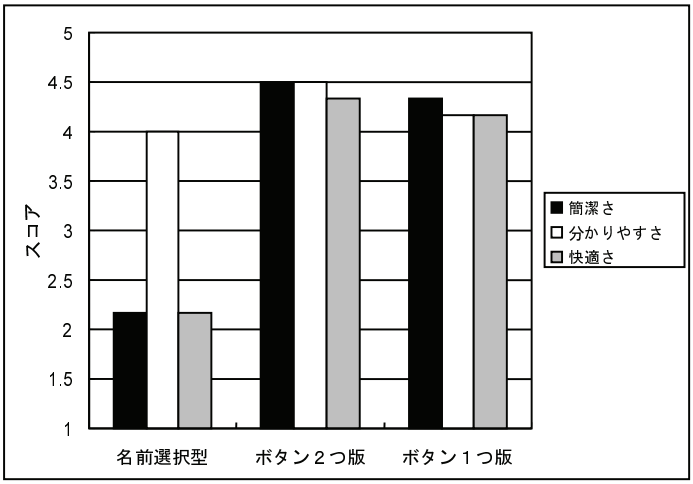

図 3.11: 使いやすさの主観的評価

3.3.1　実験 1: 使いやすさの評価

接続先の機器を選択する場合は，例えば Bluetooth Software Suite[40] の接続ウィンドウなどに見られるように，機器の名前を選択するインタフェースが一般的である．一方，提案手法では，機器そのもののボタンを押すという直接的な指定が可能である．本実験では，従来手法である名前選択インタフェースとの比較実験を行い，提案手法の使いやすさを評価した．

提案手法の使いやすさを定量的に評価するために，まずは操作時間の計測を行った．また，実際の使いやすさには，操作時間だけではなく他の要因（操作方法の分かりやすさなど）も影響すると考え，アンケートによる主観的評価も行った．被験者は 6 人の大学生・大学院生である．実験システムを図 3.9 に示す．典型的な名前選

択型インタフェース，および Touch-and-Connect フレームワークの２種類の接続指示インタフェース（ボタン２つ版，ボタン１つ版）を比較した．

名前選択型インタフェース: これは接続を指示するための最も典型的なインタフェースであり，以下のように操作する．まず，ユーザが接続元機器の検索（Search）ボタンを押すと，接続用ウィンドウを見立てたメンバーリストボックスに，ネットワーク上の全ての機器の名前の一覧が表示される．ユーザはこの中から接続先機器の名前を選択し，接続（Connect）ボタンを押す．

メンバーリストボックスには一度に４つの名前が表示でき，ネットワーク上には 24 台の機器が存在するとした．リストボックスの表示内容はスクロールバーでスクロールでき，名前の一覧は辞書順にソートされている．

ボタン２つ版インタフェース: これは，Touch-and-Connect フレームワークの最も標準的な接続指示インタフェースである（3.1.1 節参照）．

ボタン１つ版インタフェース: これは，Touch-and-Connect フレームワークの接続指示インタフェースのひとつである（3.1.5 節参照）．

まず，各インタフェースを使った場合に，接続の操作にかかる時間を計測した．図 3.9 のような実験システムを，タッチパネルディスプレイを装備したノートＰＣで動作させた．ディスプレイ上には１２個のソフトウェアエミュレータ機器が存在する．実験システムは，これらの機器のうち２つをランダムに選択し，タスク提示アイコンによってユーザに提示する．ユーザのタスクは，ディスプレイ上をペンで操作し，この２つの機器間の接続を指示することである．接続の方向（どちらが接続元機器か）は任意である．３０回のタスクを連続して与え，各タスクの操作時間を計測した．操作時間とは，タスクを提示してから，接続の操作を行いタスクを完了するまでの時間である．

各インタフェースを使った場合の平均操作時間を，図 3.10 に示す．この結果は，提案手法の操作の簡潔さを示している．Touch-and-Connect フレームワークの２つのインタフェースは，典型的な名前選択型インタフェースの約半分の操作時間しか必要としていない．ボタン１つ版インタフェースが，ボタン２つ版インタフェースよりも平均操作時間が約 0.8 秒長くなっているのは，change-mode-wait（3.1.5 節参照）の影響だと考えられる．

提案手法の使いやすさを評価するためには，複数のユーザの操作期間が重なり，ロックの解除を待たされるような状況の再現も必要である．そこで，被験者に本シ

ステムを他人と並行して使ってもらい，最後に，アンケートにより主観的評価を行った．アンケートの項目は，簡潔さ（指示に必要な手間の少なさ），分かりやすさ（インタフェース使用方法の習得の容易さ），快適さ（インタフェース使用時の心地よさ）に関する５段階評価（1–5）である．図 3.11 に，各インタフェースのスコアの，全被験者に対する平均値を示す．この結果は，提案手法が，従来の名前選択型インタフェースと比較して，簡潔かつ快適であることを示している．

本手法は，無線機器においても，物理的なケーブルにより機器間を接続する有線機器と同じような感覚で，直接的に接続指示を行えることを目指したものである．2 秒〜3 秒程度で操作ができ，操作方法も簡潔である提案手法は，物理的なケーブルと同程度の手間で接続が指示でき，インタフェースとして十分に使いやすいと言える．

3.3.2　実験 2: ロックメカニズムの有効性の評価

Reactive environments[10] や AOSS[41] のボタンインタフェースのような実世界インタフェースでは，複数のユーザの操作が干渉して誤接続が発生する可能性がある．一方，提案手法では，状態表示可能なボタンを用いてユーザの操作を排他制御（ロック）することにより，誤接続を防止できる．本実験では，それらの従来手法を仮定した「ロック無しのボタンインタフェース」との比較実験を行い，提案手法のロックメカニズムが，複数のユーザの独立操作によって発生しうる誤接続を防止できることを評価した．

ロックメカニズムの有効性を示すためには，複数のユーザの操作期間が重なるような状況が必要である．実生活での自然な接続要求に基づいて，長時間の実験を行うことが理想的であるが，被験者の実験時間には限りがあるため，アイコンでタスクを提示することにより，多くの接続要求が発生する状況を意図的に作り出して実験を行った．

被験者は８人の大学生・大学院生であり，２つの４人グループに分割して実験を行った．実験システムは実験１の場合と基本的に同様である．ただし，タスク提示間隔（タスクが完了してから，次のタスクを提示するまでの間）は，平均 10 秒の指数分布に従うランダム時間とした．

この実験システムを，４人の被験者に同時並行して使用させた．ネットワーク接続された４台のノートＰＣを，それぞれ各被験者が操作する．被験者は，他の被験者と間隔をあけて座り，他の被験者の操作を見ることはできない．

Touch-and-Connect フレームワークのボタン２つ版インタフェース（実験１と同

表 3.2: １０分間に発生した誤接続の回数と，失敗タスクの数

インタフェース	ボタン２つ版		ロック無し	
グループ	グループ 1	グループ 2	グループ 1	グループ 2
誤接続の回数	2	3	244	295
失敗タスク数	2	4	70	65
全タスク数	163	168	123	142
タスク失敗率	1.2%	2.4%	56.9%	45.8%

様），および以下のようなロック無しインタフェースとを比較し，10 分間に発生する誤接続の回数をカウントした．

ロック無しインタフェース: これは，ボタンにより接続を指示する場合の，最も平凡なインタフェースである．各デバイスは，状態表示ができない１つのボタンを持つ．最初に押したボタンが接続元機器を意味し，次に押したボタンが接続先機器を意味する．この操作により，この２つの機器間が接続される．その後，システムは初期状態に戻り，３回目に押したボタンは再び接続元機器を意味する．

ロック無しインタフェースの実装は，Touch-and-Connect フレームワークのボタン１つ版インタフェースを基にして行った．ただし，ロックを無効（ロックリストが常に空）とし，ボタンの状態表示を無くし，手動のキャンセルを無効とし，change-mode-wait を 0 とした（3.1.5 節参照）．

表 3.2 に，各グループおよび各インタフェースに対して，10 分間に発生した誤接続の回数と，失敗タスクの数を示す．タスクの遂行中（タスクを提示してから，次のタスクが提示されるまで）に，少なくとも１回の誤接続が発生した場合に，タスクは失敗したとみなす．二人の被験者の間に誤接続が発生した場合，その両方のタスクが失敗する．タスク失敗率とは，全タスク数に占める失敗タスク数の割合である．

この結果は，ロック無しインタフェースでは頻繁に誤接続が発生し，提案手法のロックメカニズムにより，これらの誤接続の大部分を防止できたことを示している．

3.4 関連研究

Touch-and-Connect フレームワークに関連した研究としては，以下のようなものが挙げられる．

3.4.1　接続先機器の自動決定

STONE[5], AMIDEN アーキテクチャ[6] では，機器の種類（Type）などに基づき，ネットワーク上から自動的に適切な機器を検索する．また，Follow-me application [4] は，ユーザの位置をセンサにより取得し，自動的にユーザに最も近いディスプレイが作業環境として選択される．しかし，多数の機器が存在する環境では自動決定には限界があり，Touch-and-Connect のように，ユーザが望んでいる機器を直接指示できる手段も必要となる．

3.4.2　直接操作技法

Reactive environments[10] では，接続したい両機器のボタンを押すことにより直接的に接続を指示できる．しかし，複数のユーザが独立に操作を行う状況を想定しておらず，複数のユーザの操作が干渉して誤接続が発生する可能性がある．

Pick-and-Drop[42] では，端末の画面をペンでタップすることにより，コンピュータ間をまたがるデータの移動を，アドレスなどを入力することなく直接的に行うことができる．コンテキストメニューに GUI のボタンインタフェースを表示することにより，Touch-and-Connect を同様の目的に使うことができる．Pick-and-Drop は，ユーザ識別のためのユニークな ID を持ったペンや管理サーバを必要とするが，Touch-and-Connect はこれらを必要としない．

FUI（Fingerprint User Interface）[43] は，コンピュータ間をまたがるデータの移動を，Pick-and-Drop と同様に直接的に指示することができる（finger-memo）．ユニークな ID を持ったペンの代わりに，指の指紋を ID として用いる．この FUI を，Touch-and-Connect と同様に，機器間の接続指示の目的に使用することもできる．しかし，一般に指紋認識モジュールは高価であり，より安価に実現できる手法が望まれる．

InfoPoint[11] は，ID 情報を記述した 2 次元マトリックスバーコードを各機器に貼り付け，カメラを搭載した携帯端末でこれを撮影することにより，ユーザが指し示している機器を特定する．

gesturePen[44] では，赤外線通信モジュールを内蔵したタグを各機器に取り付け，指向性の赤外線通信モジュールを内蔵したペンでタグを指し示して通信することにより，機器のアドレスを取得する．

これらの既存のシステムと比較して，提案手法である Touch-and-Connect は以下のような特徴がある．まず，複数のユーザによる独立操作を考慮しているにもかかわ

らず，ユーザの ID を識別する必要がなく，ユーザは特別なデバイスを持ち運ぶことなくシステムを利用することができる．接続指示インタフェースは，状態表示が可能なボタンのみで実装できるため安価であり，また GUI として実現することもできる．また，既存のホームネットワークシステムは一般に中央サーバにより集中管理されており，予め準備された環境での運用を前提としている．一方，Touch-and-Connect はサーバを用いずに実現することができ，モバイル環境や必要に応じて一時的に構築されるアドホックネットワークでの利用も可能である．

また，提案手法はボタンを押す必要があるため，手の届く範囲の機器間の接続に用いられることを想定しているが，別の実世界インタフェースとは共存でき，互いに補間関係にある．例えば，手元のデバイスは提案手法のボタンによって指示し，遠くのデバイスは他の実世界インタフェース（赤外線リモコン等）で遠隔指示を行うといった利用形態も可能である．我々は方向センサを搭載した携帯デバイスを用い，ユーザの位置とデバイスの方向から，ユーザが携帯デバイスで指し示している対象物を特定する実世界インタフェース Azim を提案した (5 章参照)．Azim のプロトタイプシステムでは，提案手法のボタンインタフェースを，携帯デバイス上で遠隔操作することができる (6.2.1 節参照)．

3.4.3 セキュリティ

Resurrecting Duckling[45][46] は，無線アドホックネットワーク環境における情報機器のためのセキュリティモデルである．機器を新規購入した際にその機器を所有者のものとして Imprinting（刷り込み）し，以後その機器は他人の機器との通信が制限される．

提案手法におけるグループチェンジャによるグループ ID の指定（3.1.3 節参照）は，Imprinting と類似しており，グループチェンジャに秘密情報を持たせることにより，同様の役割を果たすことができる．

AOSS[41] では，無線 LAN のアクセスポイントへの接続に必要な暗号鍵等の情報を，クライアント端末とアクセスポイント各々のボタンを押すことにより容易に設定できる．これはグループチェンジャによる Imprinting に類似しているが，AOSS は複数のユーザが独立に操作を行う状況を想定しておらず，多くのユーザが同時並行して操作を行うと誤接続の可能性がある．

3.5　まとめ

接続したい両機器のボタンを押すことにより，機器間の接続を直接的に指示できるフレームワーク Touch-and-Connect を提案した．本手法では，状態を表示可能なボタンを用いてユーザの操作を排他制御し，複数のユーザが独立に操作を行う状況においても誤接続を防止する．ブロードキャスト通信を用いた本手法のプロトコルは，管理サーバを必要とせず，動的な端末の入退出への対応を考慮しているため，必要に応じて一時的に構築されるアドホックネットワークでも利用可能である．

第4章　方向依存サービスのための手動の位置推定手法

ボタンを押すことが困難な遠くの機器は，Touch-and-Connect(3 章参照) の適用が難しいため，これらの遠くの機器に対しても，アドレスや名前を使わずに直接的に指定できる手法が望まれる．我々は，携帯端末で指し示すことにより，遠くの機器を直接的に指定可能なサービスを提案する．ユーザの位置と，携帯端末が指し示している方向によって，ユーザが指し示している機器を推定できる．このように，ユーザの位置だけでなくその方向も用いた，より高度な位置情報サービスのことを，我々は**方向依存サービス**と呼ぶ.

位置依存サービスと方向依存サービスの違いを図 4.1 に示す．多数の機器が存在するユビキタスコンピューティング環境では，位置のみではユーザの望む機器を特定することは難しい．ユーザの位置だけでなくその方向を用いることにより,「今見ているもの」「今指し示しているもの」などといった，より細かな指定が可能となる．

ユーザの方向は，磁気コンパスを内蔵した方向センサによって取得できるが，ユーザの位置に関しては，現状ではどこでも利用可能かつ安価な位置取得技術は存在しない．本章では，方向依存サービスのための位置推定手法を提案する．本手法では，位置が既知の複数のマーカの方位角を，ユーザが手動で指し示して測定することによりユーザの位置を推定する．この方位角に基づく位置推定手法は，他の位置センサや電子ビーコン等を必要としないため，サービスを低コストに展開できる．また，本手法は，ユーザの能動的な測定なしに位置が取得できないため，プライバシーを重視するユーザにとっても有用である．マーカの方位角の測定には誤差が伴うため，方位角の測定誤差のモデルを仮定し，ユーザの位置を，確率密度として計算した．また，本手法によって推定される位置の精度を評価するために，方向センサを用いて実験を行った.

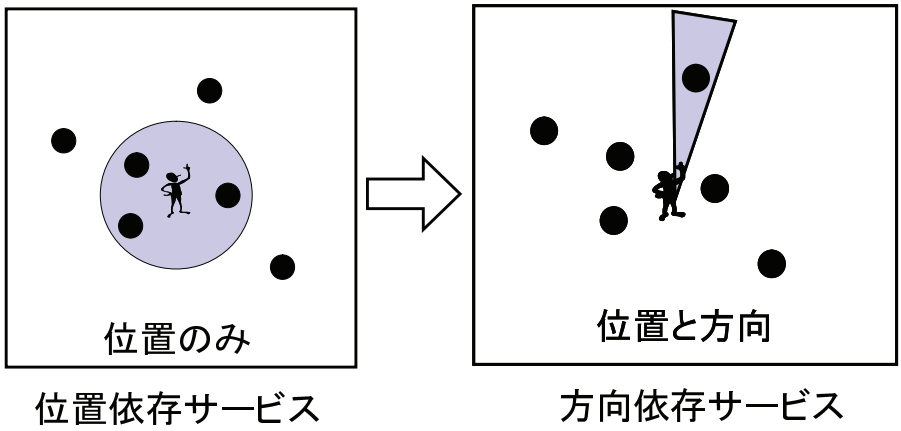

図 4.1: 位置依存サービスと方向依存サービス

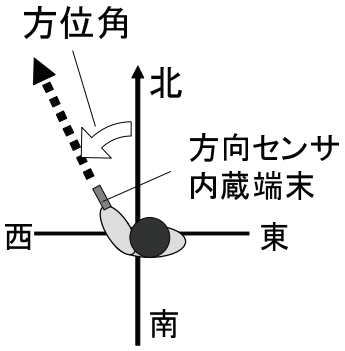

図 4.2: 方向センサにより得られる方位角

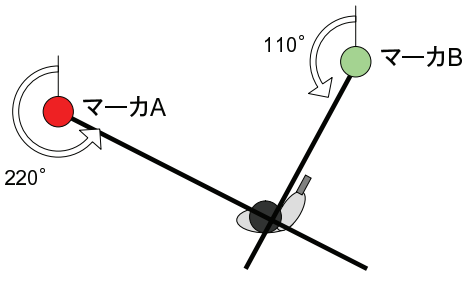

図 4.3: 複数のマーカの方位角に基づく手動の位置推定手法

4.1 方位角に基づく手動の位置推定手法

複数のマーカの方位角に基づく手動の位置推定手法の概要を述べる．簡単化のため，位置座標系として，２次元平面を仮定する．

本手法では，位置が既知のマーカ（目印）を様々な場所に配置し，ユーザが複数のマーカを指し示してその方位角を手動で測定することにより，ユーザの位置を推定する．ここで，方位角 (Azimuth) とは，図 4.2 のような，北を基準とした水平面上の絶対角であり，方向センサを内蔵した携帯端末により測定する．２つ以上のマーカの方位角が分かった場合，図 4.3 のように，各マーカより引かれる半直線の交点がユーザの位置となる．方向の測定には誤差が伴うため，本手法では，方向の測定誤差をモデル化し，ユーザの位置を確率密度として計算する．詳しくは 4.3 節で述べる．

位置推定のためには，ユーザの指し示したマーカを特定する必要がある．本手法では，マーカに色（または形状など）によって種類を設け，ユーザがこれをボタン等により入力することにより，ユーザが指しているマーカを特定する．広い範囲での利用を想定した場合，複数のマーカが同一の色を持つことある．同一の色のマーカを配置するためには，マーカは他の環境側の情報に基づき区別される必要がある．5 章で提案するシステムでは，この環境側の情報の例としては，無線 LAN の基地局 ID より得られる位置情報を用いている（5.1.2 節参照）．

マーカは単にユーザにとっての目印であり，電子機器ではないため，安価に配置することができる．また，既に存在するランドマークや建物などをマーカとして代用することもできる．この場合，色を入力する代わりに，ランドマークの名称や種類などを選択する．

4.2 方向センサ

北を基準とした絶対的な方位角を測定するための方向センサとしては，例えば，磁気センサと加速度センサを組み合わせたものが存在する．なお本論文においては，北は真北（地図上の北）ではなく，磁北を意味する．地磁気の方向を磁気センサにより測定し，また重力の方向を加速度センサにより測定することにより，ほかに特別な設備を使わなくても，デバイスの姿勢（方位角，仰角，ロール角）を取得することができる．このようなセンサデバイスの例としては，Microstrain 社の 3DM[47] や NEC Tokin の 3D モーションセンサ [48] などがある．

4.3　位置確率密度の計算

ユーザがマーカの方位角を測定する際には，誤差が伴う．誤差の原因としては，ユーザの指し示し動作のずれや，金属等による地磁気の乱れなどが考えられる．従って，単に，測定された方位角の半直線をマーカから引いて，その交点を求めるだけでは，解が存在しない場合がある．例えば，２つの半直線が交わらない場合や，３つ以上のマーカの方位角を測定した場合などである．また，解が得られたとしても，その求まった位置がどの程度正確なのか（精度）が分からない．従って，方位角の測定誤差のモデルを仮定し，ユーザの存在するであろう位置を，確率密度として計算する．

位置確率密度の計算においては，本手法を用いる応用システムに応じて，以下のような位置に関連した様々な情報を活用することができる．

- 立ち入ることのできない場所，障害物によって対象物が見えなくなる状況 (5.1.4 節参照) などの，障害物の影響
- 無線 LAN の基地局より得られる位置情報 (5.1.2 節参照)
- 事前に学習された磁気分布情報．これを用いれば，地磁気が乱れている環境においてもロバストな測定が可能となる．(5.2.2 節参照)

本節では，ユーザの位置の確率密度関数の計算方法について述べる．なお，各確率変数の実現値を，その変数を小文字にしたもので表す．例えば，確率変数 P の実現値は p で表す．また，f は，１つまたは複数の確率変数の確率密度関数（離散値の場合は確率関数）を表す．本節には様々な確率密度関数が登場するが，f の引数の型ごとに，１つの確率密度関数が対応する．

問題の定式化のために，以下の確率変数を導入する．ただし，指定マーカとは，(i 番目の測定で）指し示したマーカのことである．

- P: ユーザの位置 (２次元ベクトル)
- A_i: 指定マーカの方位角の測定値 (連続値)
- C_i: 指定マーカの色 (離散値)．複数のマーカが同一の色を持つ可能性もある．
- M_i: 指定マーカの識別子 (離散値)．ユーザが実際にどのマーカを指し示したのかを表す．各識別子は唯一のマーカに対応する．

ユーザは，マーカを選んで指し示しその色を入力する作業（測定）を複数回繰り返す．全測定回数を n とする．A_i, C_i, M_i は 1 回の測定結果である．ただし，システムが直接知ることのできるのは A_i, C_i のみであり，M_i は入力されない．

4.3.1 方位角測定モデル

方位角測定モデル $f(a_i|m_i, p)$ は，ユーザがマーカ m_i を位置 p で指し示したときに測定される方位角の確率分布である．マーカの位置は既知のため，マーカの真の方位角は，マーカの識別子 m_i と，ユーザの位置 p より定まる．もっともシンプルなモデルとしては，真の方位角を平均とする正規分布が考えられる．分散はユーザの指し示し動作の習熟度などによって調節する．地磁気の乱れによって生じる方位角の測定値 a_i の誤差を，事前に学習された磁気分布情報を用いて，修正することもできる (5.2.2 節参照)．

4.3.2 測定間の独立の仮定

ユーザの位置の確率変数 P を固定した時，i 回目の測定は，他の測定結果 (i 回目以外) の影響を受けないとする．つまり，各測定結果 (A_i, C_i) は，互いに P の条件付独立とする．すなわち，以下の式が成り立つと仮定する．

$$\begin{aligned} & f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n|p) \\ = \;& f(c_1, a_1|p)\, f(c_2, a_2|p) \cdots f(c_n, a_n|p) \end{aligned} \tag{4.1}$$

この式は，特定の位置 p において，一連の測定結果が起こりうる確率密度が，各測定結果が起こりうる確率密度を掛け合わせることにより，計算できることを意味する．独立を仮定することにより，位置確率密度の計算を単純化することができる．

4.3.3 位置推定の定式化

求める位置確率密度は，各測定結果 c_i, a_i が分かった時の位置 p の確率密度 $f(p|c_1, a_1, c_2, a_2, \cdots, c_n, a_n)$ により計算される．

$$\begin{aligned} & f(p|c_1, a_1, c_2, a_2, \cdots, c_n, a_n) \\ = \;& \frac{f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n, p)}{f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n)} \\ = \;& \frac{f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n, p)}{\int_p f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n, p) dp} \end{aligned} \tag{4.2}$$

また，式 (4.1) より

$$\begin{aligned} & f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n, p) \\ &= f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n|p)\, f(p) \\ &= f(c_1, a_1|p)\, f(c_2, a_2|p) \cdots f(c_n, a_n|p)\, f(p) \end{aligned}$$

また，

$$\begin{aligned} & f(c_i, a_i|p) \\ &= \sum_{m_i} f(c_i, a_i, m_i|p) \\ &= \sum_{m_i} \{f(c_i|a_i, m_i, p)\, f(a_i|m_i, p)\, f(m_i|p)\} \end{aligned}$$

ここで，$f(c_i|a_i, m_i, p), f(a_i|m_i, p), f(m_i|p), f(p)$ は以下のように与える．

- $f(c_i|a_i, m_i, p)$: これはマーカ m_i の色を表す．マーカ m_i の色が c_i の時には $f(c_i|a_i, m_i, p) = 1$，さもなくば $f(c_i|a_i, m_i, p) = 0$ となる．
- $f(a_i|m_i, p)$: これは 4.3.1 節で述べた，方位角測定モデルである．
- $f(m_i|p)$: ユーザのマーカ選択モデルである．すなわち，ユーザが，位置 p において，i 番目の測定の際に，どのマーカを選択するのかを表す確率関数である．例えば，位置 p から利用できる (見える) 全マーカに対する一様分布を用いる．利用可能なマーカの集合は，マーカの利用可能領域より得られる（5.1.4 節参照）．
- $f(p)$: 事前に分かっているユーザの位置の確率密度（事前確率）である．例えば，システムの利用可能な領域に対する一様分布を用いる．5 章で提案するシステムのように，無線 LAN の基地局情報より得られる位置情報（5.1.2 節参照）を適用したり，障害物などユーザが存在しえない場所をあらかじめ除外することもできる．

２つのマーカの方向を測定して位置を推定した場合の計算例を図 4.4 に示す．ユーザは，正方形の区画の中心付近におり，１回目に左 45 度方向にあるマーカを，２回目に右 45 度方向にあるマーカを指し示した．図中の正方形は，各確率密度を位置空間上にプロットしたものであり，白い部分が確率値の高い部分である．図中の正規化とは，式 (4.2) における，分母 $\int_p f(c_1, a_1, c_2, a_2, \cdots, c_n, a_n, p)dp$ による除算を表す．

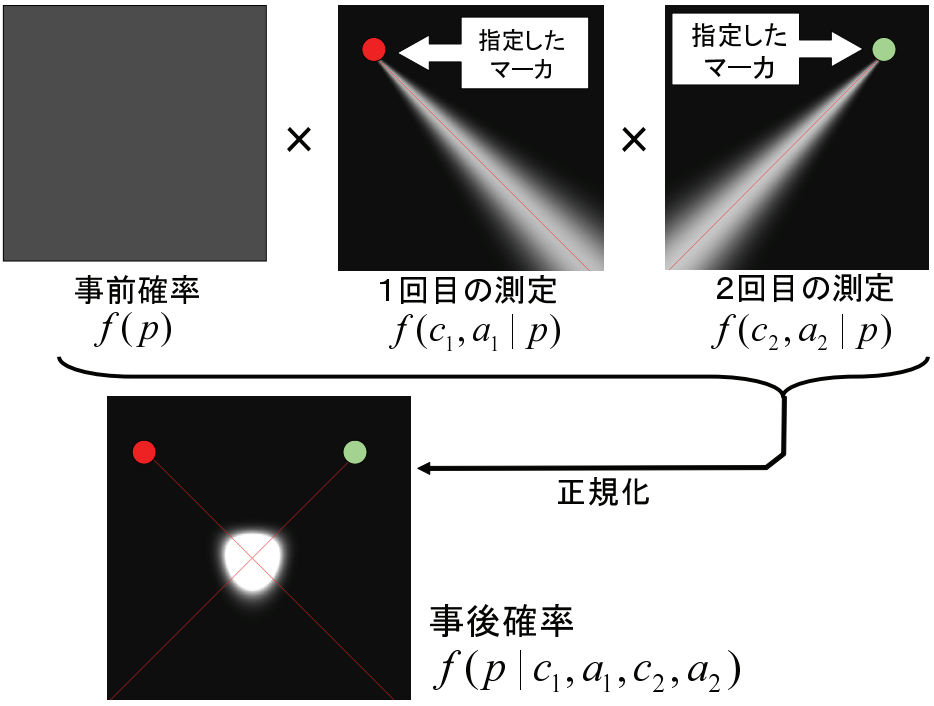

図 4.4: 位置確率密度の計算

事後確率を見ると，ユーザが正方形の中心付近にいることを推定できていることが分かる．

4.4 評価

本手法によって推定される位置の精度を評価するために，Microstraion の方向センサ 3DM[47] を用いて位置推定実験を行った [49]．本実験では磁気の乱れの少ない場所が存在する，図 4.5 のような屋外環境で実験を行った．位置確率密度は 128 × 128 の 2 次元配列として計算した．また，方位角の測定誤差のモデルは，4.3.1 節に基づき，標準偏差 σ を 5 度とした正規分布を用いた．なお，事前に予備実験として，特定の目標を指し示して，方向センサによりその方位角を測定する実験を 100 回行ったところ，方位角の標準偏差は 3.5 度となった．しかし，この方位角の測定値の平均が正しい方位角であるとは限らないため，モデルの標準偏差 σ は余裕を持たせて 5 度とした．

建物（図中の建物 A）の両端 2 箇所をマーカとした．マーカ間の距離は 30.5m である．マーカ周辺の 20 地点で，本手法に基づく位置の推定を行った．方向センサ 3DM[47] に，方向合わせを補助するための棒状のガイドを取り付けた測定装置を用いて，2 箇所のマーカを指し示し，各々のマーカの方位角を測定した（指し示し測

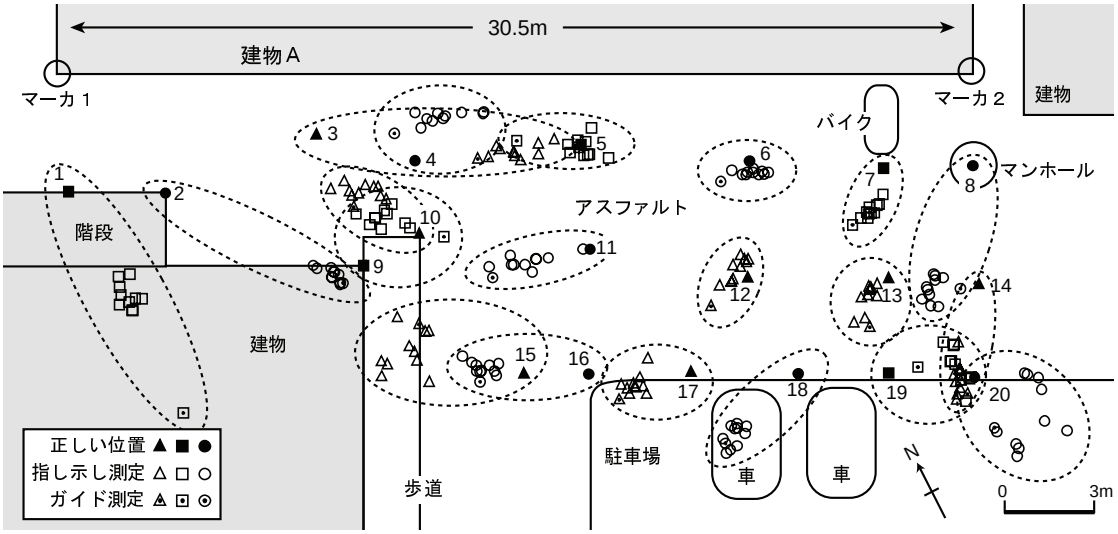

図 4.5: 推定された位置

定）．各地点において，この実験を 10 回行った．また，指し示し動作による誤差の影響を評価するために，棒状のガイドを片目で覗き込んで，測定装置を正確にマーカの方向に合わせた状態での測定（ガイド測定）も行った．測定地点の正しい位置は，超音波を用いた距離計 (STMS-850B) により，両マーカ地点からの距離を測定して求めた．

実験結果を図 4.5 に示す．数字が添えられた黒塗りの印は，測定地点の正しい位置を表す．白抜きの印は，指し示し測定により推定された位置（10 回分）を表す．本手法ではユーザの位置は確率密度として計算されるため，確率密度の重心を推定された位置とした．また，中央に点のある印は，ガイド測定により推定された位置である．図を見やすくするために，同一の地点での推定結果を破線で囲んだ．印の形状の違い（三角，四角，丸）は，単に図の見やすさを考慮したものであり，近傍で印の形状が同じものが，同一の地点での推定結果である．

本手法により推定された位置（指し示し測定の場合）は，正しい位置から平均で 2.7m 離れていた．図を見ると，多くの地点でこの程度の精度の位置情報が得られていることが分かる．しかし，特に地点 1, 2, 3, 8 などでは，正しい位置と推定された位置のずれが大きい．これは，鉄筋コンクリートや鉄板などの影響で環境の磁気が乱れ，方向センサにより測定される方位角がずれたためと考えられる．

実際，ガイド測定により正確に方向合わせをした場合の推定位置も，正しい位置から平均で 3.1m 離れており，測定される方位角に誤差が生じていることが分かる．一方で，指し示し測定による推定位置と，ガイド測定による推定位置との距離は，平均で 1.35m となり，両推定位置は比較的近くにあることが分かる．また，指し示し

測定の方位角測定値と，ガイド測定の方位角測定値とのずれ（負の値もありうる）の分布は，平均値 +0.34 度，標準偏差 3.3 度となった．すなわち，指し示し動作そのものによる誤差の影響は比較的小さく，地磁気の乱れによる方位角測定値のずれを補正することにより，位置の精度が改善することが期待される．

本手法は，屋内での利用も目標としているが，特に屋内では地磁気の乱れは大きく，例えば我々の研究室内では，磁気の方向は北方向から 40 度程度ずれている．従って，特に屋内環境では，方位角測定値のずれの補正は重要である．

4.5 関連研究

位置取得技術としては，GPS を用いる方法が広く普及している [18]．GPS は，屋外の見通しの良い場所であれば，10m 程度の誤差で位置情報を得ることができる．しかし，ビル街や屋内など，GPS 衛星からの電波が届きにくい場所や電波が反射してしまう場所での利用は困難である．また，GPS は電源投入後，GPS 衛星を補足してセンサが利用できるようになるまでに時間がかかる (cold-start) という問題もある．

携帯電話向けの位置測定技術として，基地局側で GPS 衛星を補足することにより，cold-start 時の待ち時間を削減する Assisted-GPS[50] がある．また，屋内向けの測位技術として，位置既知の超音波発生装置への超音波の到達時間を用いた Active-BAT[4] などがある．しかし，これらの手法は，一般に環境側の設備が高価となってしまう．

複数の無線 LAN 基地局からの（または基地局への）電波強度を用いて，端末の位置を推定する手法が提案されている [51][14]．5 章で提案するシステムでは，基地局 ID によってユーザの存在しうる位置を電波到達範囲 (100m 程度) 内に制限している (5.1.2 節参照) が，複数の基地局からの電波強度情報を使ってより狭い位置範囲 (10m 程度) に制限することも考えられる．これにより，マーカ配色に必要な色数を削減（もしくはより密に配置）できる．

しかし，特に屋内では，障害物による反射・吸収などにより，電波は空間的・時間的に乱れている．文献 [52] の手法では，あらかじめ，システム利用区域内の各位置において，電波強度の測定を行い，電波強度の空間的な分布を学習する．これにより，電波の乱れに対してロバストな位置推定を実現している．6 章で述べるプロトタイプシステムでも，磁場の空間的な分布をあらかじめ学習しておくことにより，地磁気の乱れに対してロバストな位置推定を行うことができる．

既存の位置測定技術と比較して，本章で提案したマーカの方位角を用いた手動の位置測定手法は，以下のような特徴がある．

- **低コストな運用：**環境側に特別な装置が必要な手法と比較すると，本手法は単にマーカを配置すればよく，またマーカは電力等の運用コストも不要なため，環境側の設備を安価にできる．また，マーカの代わりにランドマーク等を用いれば，マーカを配置する必要もない．5 章で提案するシステムのように，無線 LAN との組み合わせを考えた場合でも，無線 LAN 基地局は安価であり既に広く普及している [14]

- **どこでも利用可能：**磁気センサを用いた方向センサと磁気分布の事前学習により，屋内・屋外問わず広い範囲で本手法を運用できる．

- **位置センサが不要：**位置と方向に基づく位置情報サービス（方向依存サービス）への応用を考えた場合，方向センサだけで位置・方向ともに取得でき，新たに他の位置センサが不要なため，クライアント端末のコストを安くできる．

- **素早い起動：**GPS と比較すると，携帯端末の電源を入れてからすぐに位置情報の取得が行えるという利点がある．

- **プライバシー：**ユーザが能動的に位置を測定しない限り，ユーザの詳細な位置をシステムが知りえないため，プライバシーを重視するユーザにとって有用である．

4.6　まとめ

本章では，方向依存サービスのための位置推定手法を提案した．本手法では，位置が既知の複数のマーカの方位角を，ユーザが指し示して手動で測定することにより，ユーザの位置を推定する．本手法は，電子ビーコン等の設備が必要ないため，サービスの安価な展開が可能である．

本手法によって推定される位置の精度を評価するために，方向センサを用いて実験を行った所，30.5m 離れた 2 つのマーカを用いた場合，誤差の平均が 2.7m となった．また，一部地点では位置精度が低下したが，指し示し動作そのものによる誤差の影響は比較的小さく，地磁気の乱れによる方位角測定値のずれを補正することにより，位置精度の改善が期待されることを示した．

第5章　無線LANを用いた方向依存サービスシステム

本章では，方向依存サービスを実際に運用するためのシステム Azim を提案する．方向依存サービスの例として，ユーザが指し示した対象物を推定するサービスを提供する．本システムは，4 章で提案した位置推定手法を利用しており，限られたマーカの色数で広い範囲でのサービス運用を可能とするために，無線 LAN の基地局情報より得られる位置情報を活用している．また，本システムの有用性を判断するために，屋内および屋外で評価実験を行った．

5.1　方向依存サービスシステム Azim

5.1.1　利用形態

方向依存サービスシステム Azim の利用形態について述べる．図 5.1 にシステムの構成を示す．ユーザは，まず複数のマーカを指し示してその色を入力することにより自分の位置を測定する．次に対象物を指し示して，その指定対象物に関連した様々な応用サービスを受ける．応用サービスの例としては，指し示した機器を遠隔操作する，指し示した機器の情報をユーザの携帯端末に表示する，などが考えられる．ユーザは，同じ位置で指し示す対象物を切り替えて応用サービスを受けることができる．位置を移動した場合には，再度手動で位置を測定する必要がある．しかし，文献 [53] などの手法を応用することにより，方向センサに内蔵されたセンサを利用し，一度絶対位置を取得した後のユーザの位置を，ある程度の時間追跡する方法も考えられる．

5.1.2　無線 LAN の基地局情報

近年，ホットスポットなどの公衆無線 LAN サービス，および家庭・オフィス等での無線 LAN の利用が急速に普及しつつある [54]．本システムでは，ユーザの携帯端末と情報管理サーバとの通信のために，無線 LAN を用いる．

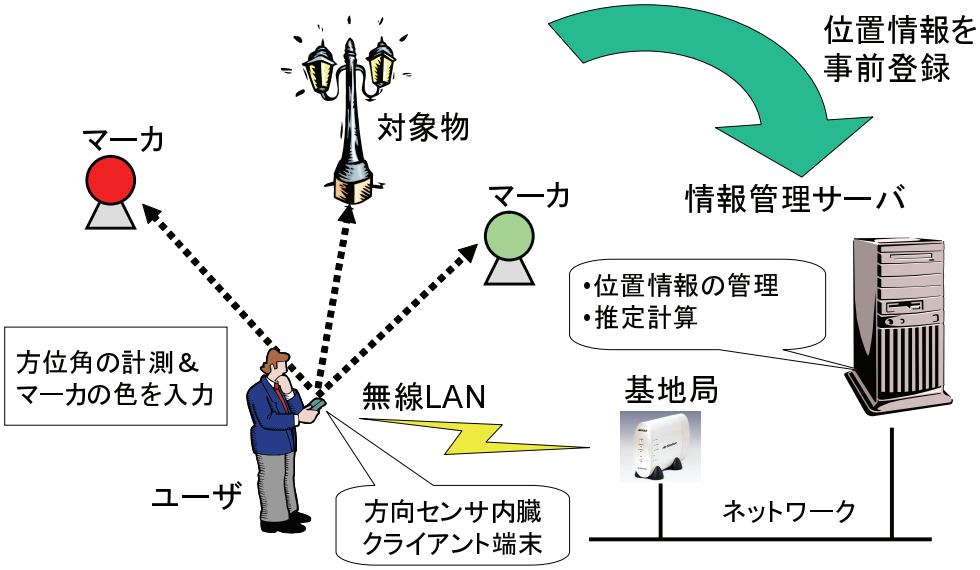

図 5.1: Azim: 方向依存サービスシステム

無線 LAN では，現在接続中の基地局 ID をクライアント端末が取得可能であり，これによりユーザがこの基地局の電波到達範囲内にいることが分かるため，粗い精度の位置情報が得られる．障害物がない場合これは半径 50-100m 程度である．4.3 節で述べる位置確率密度の計算において，事前確率 $f(p)$，および位置空間の積分範囲を，この電波到達範囲内に制限することができる．また，この電波到達範囲内で利用できるマーカのみが指定の対象となるため，同じ色のマーカがこの範囲外に存在しても区別可能となり，配色に必要な色数を削減することができる．

5.1.3　システムの構成要素

本システムは，以下のような要素により構成される．

- クライアント端末: ユーザの持ち歩く携帯端末である．PDA や携帯電話のようなものを想定している．方向センサを搭載し，ユーザがマーカや対象物を指し示した際に，その方位角を測定することができる．
- 情報管理サーバ: マーカ，対象物，基地局に関する位置情報を管理する．また，位置確率密度および指定対象物の推定などの計算もこのサーバで行う．
- 基地局: 無線 LAN の基地局である．クライアント端末は，現在接続中の基地局の ID（MAC アドレス等）を取得することができる．

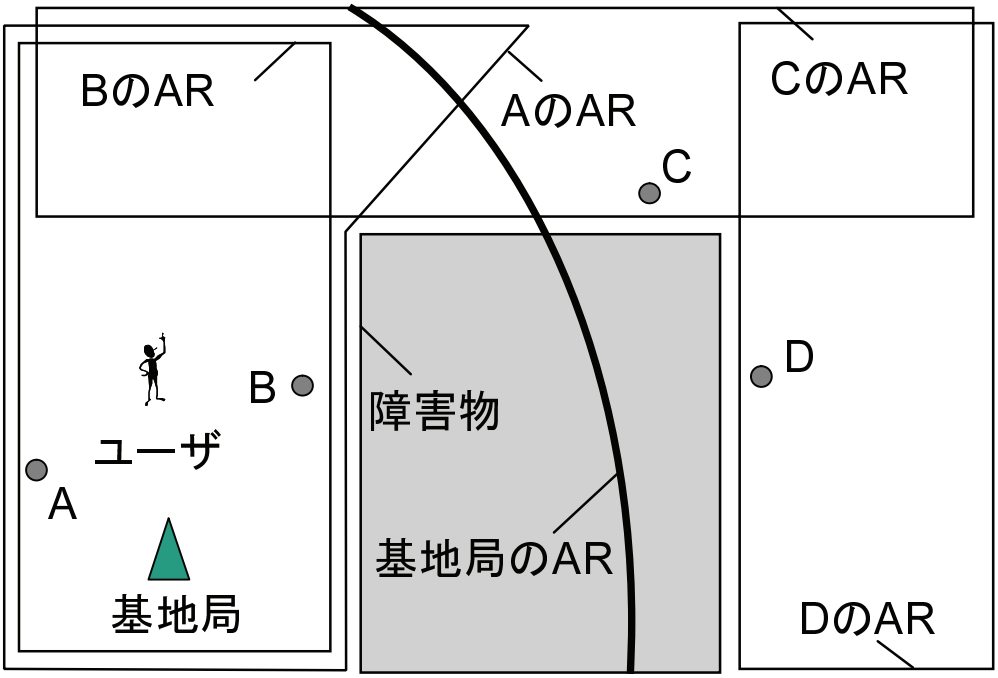

図 5.2: 利用可能領域 (AR) の例

- マーカ: ユーザが位置を測定する際に指し示す目印である．マーカには色などにより種類が設けられており，ユーザは，マーカを指し示す際にこれを入力する．ランドマークや建物などをマーカの代用とすることもできる．
- 対象物: ユーザが指し示す対象となる物体である．機器，店，ランドマーク等を想定している．

5.1.4 利用可能領域

次に，広い範囲での運用を考える．ユーザが見えないマーカや対象物を指し示すことはないため，これらの利用可能な領域を考慮することが必要となる．また，基地局の電波の到達範囲も管理する必要がある．従って，マーカ，対象物，基地局に対して，その利用可能領域 (AR; available region) を定義する．マーカ・対象物の利用可能領域により，ある位置から利用できるマーカ・対象物が特定可能となり，マーカ選択モデル (4.3.3 節参照)，対象物選択モデル (5.1.5 節参照) を具体的に計算できる．基地局の利用可能領域は，基地局の電波到達範囲を表し，5.1.2 節で述べたように，ユーザの存在しうる位置をあらかじめ制限するために用いられる．

利用可能領域の例を図 5.2 に示す．障害物に隠れた物体や，遥か遠くの物体をユーザが指し示すことは無いため，利用可能領域は障害物や物体からの距離を考慮して定

義する．また，現在クライアント端末が接続中の基地局を特定することにより，ユーザが基地局の利用可能領域内にいることが分かる．基地局の利用可能領域とマーカの利用可能領域が重なっている場合のみ，そのマーカが指定の対象とみなされる．例えば，図5.2のA–Dがマーカであるとした場合，マーカDは（ユーザが基地局の利用可能領域のどこにいたとしても）指し示されることはない．マーカの配色を考えた場合，それぞれの基地局に対して，基地局の利用可能領域とマーカの利用可能領域が重なっているマーカが別々の色に配色されていれば，色からマーカを唯一に特定できる．

次に，ユーザの位置を計算できたとする．ユーザの存在しうる位置（位置の確率密度の高い部分）に対して，対象物の利用可能領域が重なっている場合にのみ，その対象物が指定の対象とみなされる．これにより，見えない対象物が指定対象物の候補として推定されることはない．例えば，図5.2のA–Dが対象物であるとした場合，図中のユーザの位置と利用領域が重なっている対象物A,Bのみが，指定の対象となる．

5.1.5 指定対象物の推定手法

本節では，ユーザの指し示している物体を推定するための計算方法について述べる．ユーザの指し示す対象となる物体を**対象物**，実際にユーザが指し示した対象物を**指定対象物**と呼ぶことにする．対象物の位置は既知とする．4.3節の手法で計算できるユーザの位置確率密度，および対象物の方位角より，指定対象物を推定する計算を行う．問題の定式化のために，以下の確率変数を導入する．

- P: ユーザの位置 (２次元ベクトル)
- A: 指定対象物の方位角の測定値 (連続値)
- S: 指定対象物の識別子 (離散値)．ユーザが実際にどの対象物を指し示したのかを表す．各識別子は唯一の対象物と対応する．

指定対象物を推定するためには，ある対象物が指定対象物である確率の分布を計算すればよい．求める確率は，対象物の方位角の測定値 a が分かった時に，対象物 s が指定対象物である確率 $f(s|a)$ である．

$$f(s|a) = f(a,s)/f(a) = f(a,s)/\sum_{s} f(a,s)$$

$$f(a,s) = \int_p f(a,s,p)dp$$
$$f(a,s,p) = f(a|s,p)\, f(s|p)\, f(p)$$

ここで，$f(a|s,p), f(s|p), f(p)$ は以下のように与える．

- $f(a|s,p)$: 対象物の位置は既知のため，対象物の識別子 s と，ユーザの位置 p より，対象物の真の方位角が定まる．マーカの場合と同様に，$f(a|s,p)$ は方位角測定モデル (4.3.1 節参照) より計算できる．

- $f(s|p)$: ユーザの対象物選択モデルである．すなわち，ユーザが，位置 p において，どの対象物を指し示すかを表す確率関数である．例えば，位置 p から利用できる (見える) 全対象物に対する一様分布を用いる．マーカの場合と同様，利用可能な対象物の集合は，対象物の利用可能領域より得られる（5.1.4 節参照）．

- $f(p)$: ユーザの位置の確率密度である．これは，4.3 節の計算によって得られた位置確率密度である．

5.2 評価

提案したシステムの有用性を評価するために，6 章で述べるプロトタイプシステムを用いて実験を行った．本システムは屋外・屋内を問わず広い範囲で利用することを想定しているため，屋外および屋内の両方で評価実験を行った [55][56].

5.2.1 実験 1 (屋外)

まず，本システムで採用している手動の位置推定手法によって得られる位置の精度を評価するために，屋外での位置推定実験を行った．図 5.3 のような運動場の角で実験を行った．

マーカは図中 A,B の 2 箇所に 30 メートル離して配置した．そして，マーカ周辺の 14 箇所において，LocPointer(6.1.1 節参照) で 2 つのマーカを指し示してそれぞれの方位角を測定し，提案した位置推定手法により位置を推定した．それぞれの箇所でこの位置推定を 6 回行った．

位置推定の結果を図 5.3 に示す．数字が添えられた黒塗りの印は，測定地点の正しい位置を表す．白抜きの印は推定された位置（6 回分）を表す．本システムでは

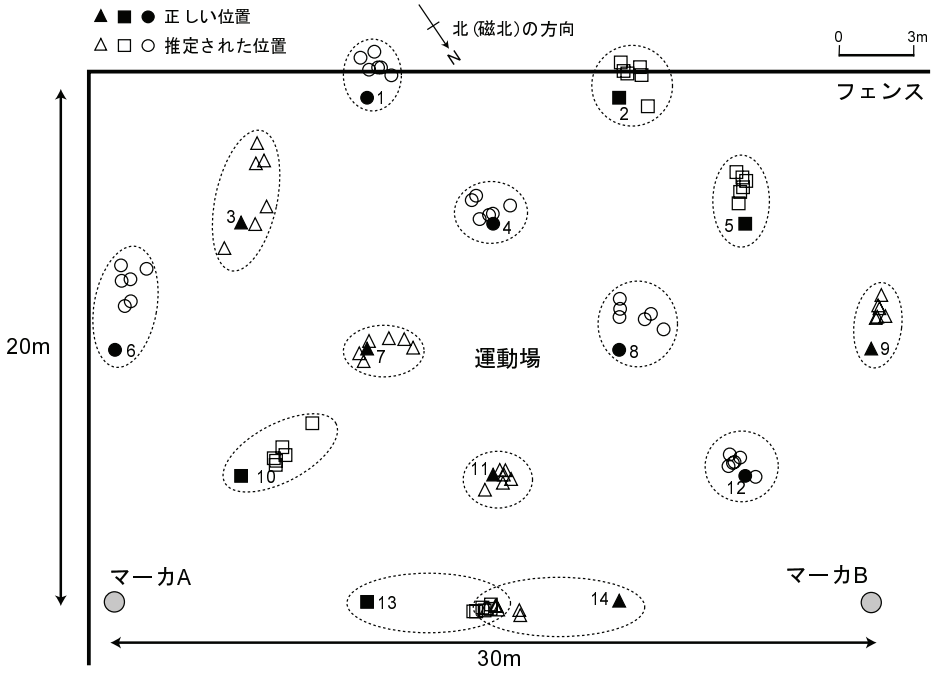

図 5.3: 推定された位置 (屋外; 2 つのマーカ)

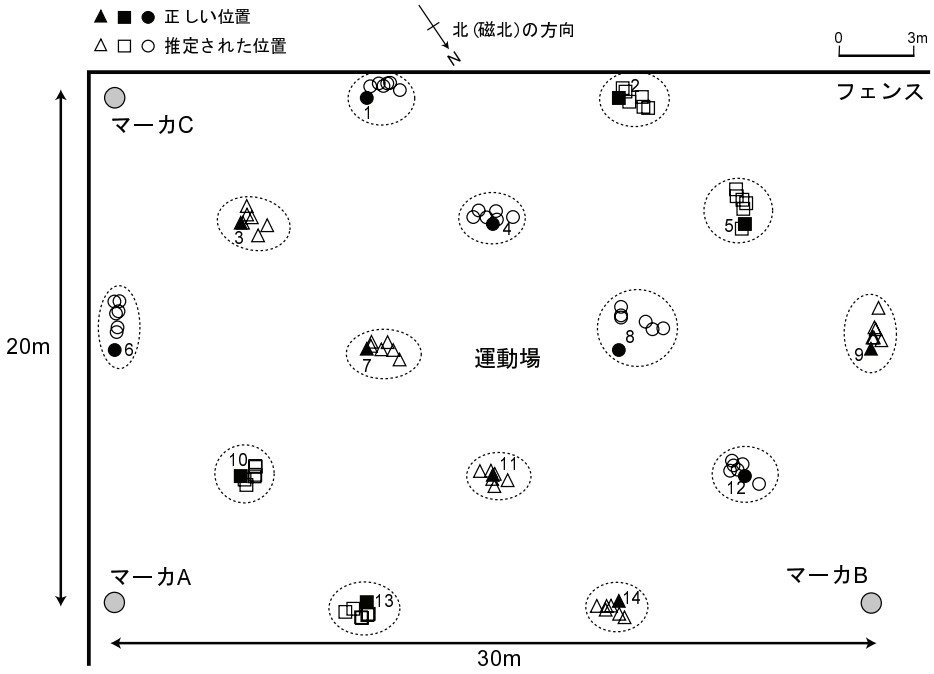

図 5.4: 推定された位置 (屋外; 3 つのマーカ)

ユーザの位置は確率密度として計算されるため，確率密度の重心を推定された位置とした．印の形状の違い（三角，四角，丸）は，単に図の見やすさを考慮したものであり，近傍で印の形状が同じものが，同一の地点での実験結果である．

本システムにより推定された位置は，正しい位置から平均で 1.9 メートル (80 %の位置推定が 2.8 メートル以内) 離れていた．この精度が得られれば，屋外において対象物を指定するような応用に対しては，十分に利用可能な精度であるといえる．また，位置の推定精度は基本的にマーカ間の距離に比例するため，より精度の高い位置推定を必要とする場合はマーカをより密に配置することで対処できる．

しかし，2 つのマーカを結ぶ線上の地点，例えば図 5.3 の地点 13,14 では， 位置推定精度は相対的に落ちている．一方，指し示す 2 つのマーカの方位角の差が直角 (90 度) に近い場合は，位置推定精度は相対的に高い．従って，利用区域に 3 つ以上のマーカを置き，ユーザが望ましい 2 つのマーカを選択するか，あるいは単に全てのマーカの方位角を測定することにより，位置推定精度の低下を防止でき，ロバストな位置推定が可能となる．図 5.4 は，実際，3 つのマーカ (図中の A,B,C) を指し示して同様の実験を行った場合の位置推定結果である．この場合，推定された位置は，正しい位置から平均で 0.78 メートル (80 %の位置推定が 1.1 メートル以内) 離れており，位置推定精度が向上している．また，マーカの間の地点を含む全ての地点において，位置推定精度が安定していることが分かる．

5.2.2 実験 2 (屋内)

次に，屋内での位置推定実験を行った．実験は，我々が構築したユビキタスコンピューティング環境のテストベッドルームである Cogma Room(図 1.2) で行った．この部屋の物体の配置を図 5.5 に示す．4 つのマーカ (マーカ A – D) と，11 個のディスプレイ (LCD1 – LCD9, Screen, PDP) が存在する．方向依存サービスの例として，これらのディスプレイのうちの１つを指し示し，そこに映像を表示させるサービスを想定して，実験を行った．

一般に，鉄筋コンクリートのビルや，機器が多数存在する屋内環境においては，金属による地磁気の乱れが，方向センサ内の磁気コンパスに影響し，方位角の測定に誤差が生じる．この方位角の測定誤差を補正するために，あらかじめ，利用区域内の様々な位置において，磁気のベクトル（方向および強さ）を測定し，磁場の空間的な分布を学習した．磁気ベクトルの空間的な分布を示す，磁気分布情報 (MFI; magnetic field information) を，図 5.6 に示す．図より，磁気ベクトルの方向は± 20

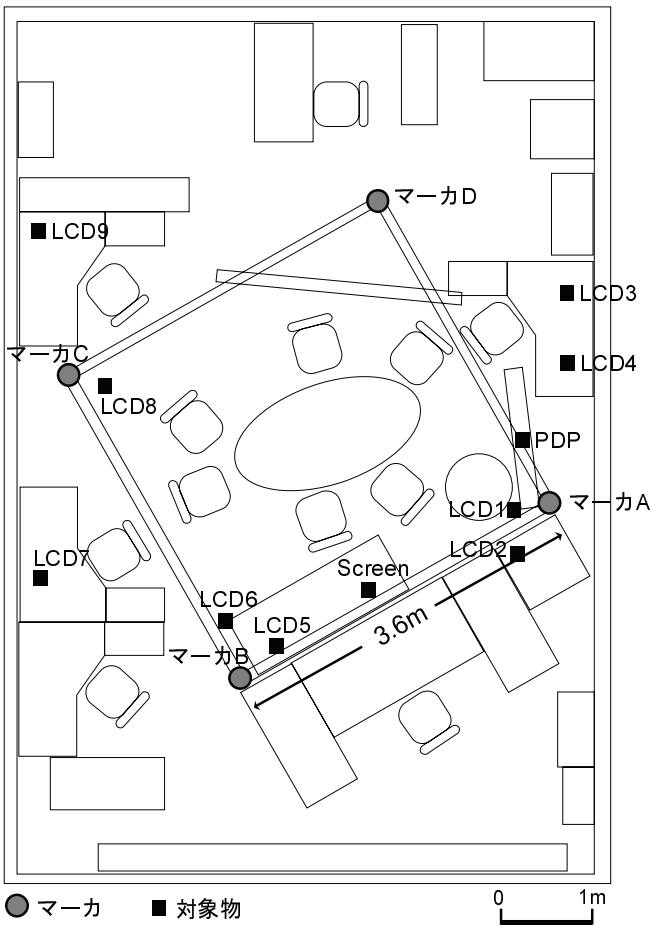

図 5.5: 屋内実験環境の物体の配置

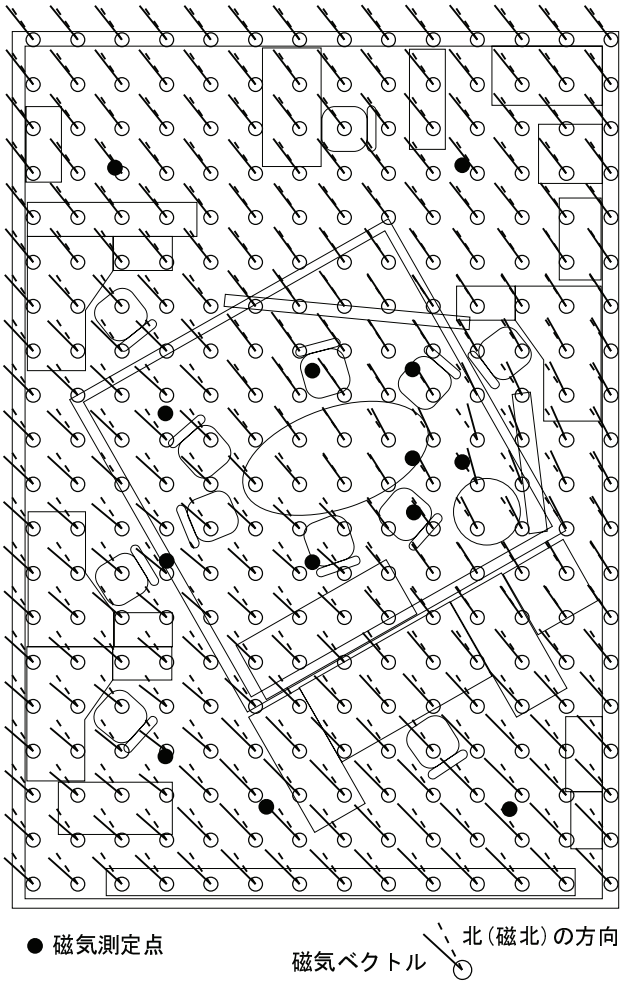

図 5.6: 予め学習させた磁気分布情報 (MFI)

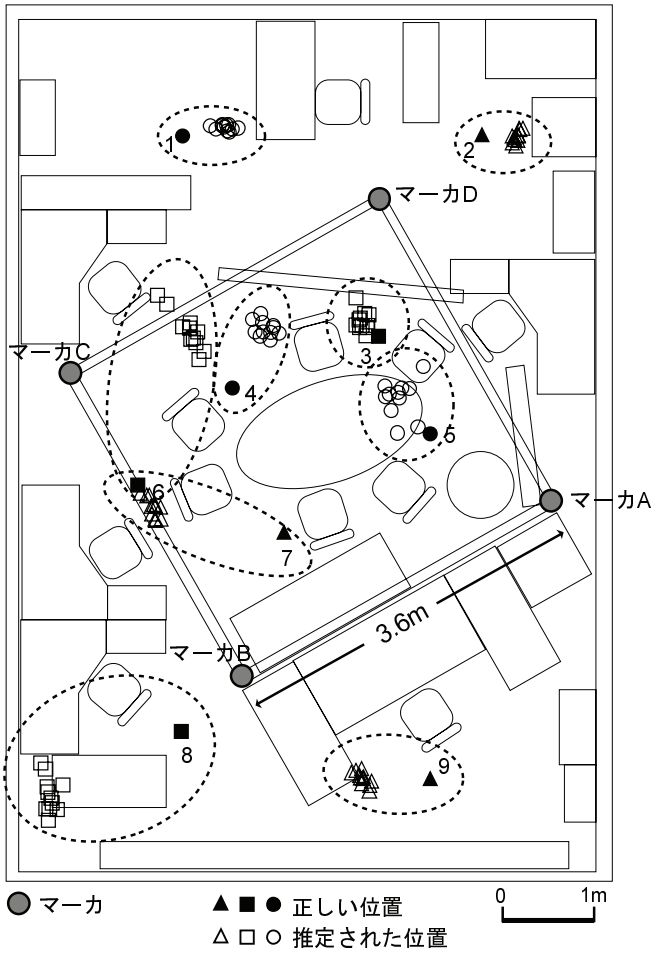

図 5.7: MFI を用いない場合の位置推定結果

度の範囲にわたって乱れていることが分かる．空間上の任意地点での磁気ベクトルの値は，測定地点までの距離の逆数の 3 乗を重みとして，測定した全ベクトルを平均することにより，補間して推定した．この補間方法は一般に Inverse Distance Weighted 法と呼ばれ，3 乗という係数は，幾つかの係数で補間を行い経験的に求めた．

予め学習させたこの磁気分布情報 (MFI) は，方位角の測定値 (4.3.3 節の $f(a_i|m_i, p)$ の a_i，および 5.1.5 節の $f(a|s, p)$ の a) を修正するために用いられる．MFI の効果を評価するために，*MFI* **を用いない場合** (磁気ベクトルの方向が全て磁北の方向に等しいと仮定する) と，*MFI* **を用いる場合**の 2 種類の位置推定結果を比較した．

図 5.7 の 1–9 の 9 つの地点において，実験を行った．2 つのマーカの方位角を Loc-Pointer で指し示して測定し，提案した位置推定手法により位置を推定する実験を，各地点で 11 回行った．用いた 2 つのマーカは，地点 3, 4, 6, 9 では A と B，地点 5, 8 では B と C，地点 1, 7 では C と D, 地点 2 では D と A である．

MFI を用いない場合の推定位置を図 5.7 に示す．実験 1 の場合と同様に，黒塗り

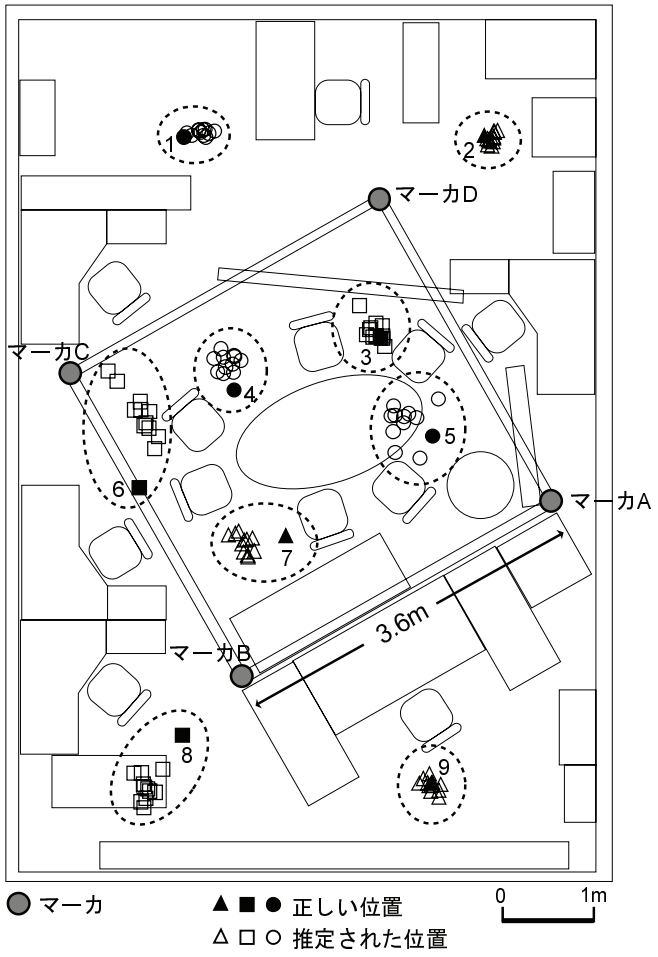

図 5.8: MFI を用いる場合の位置推定結果

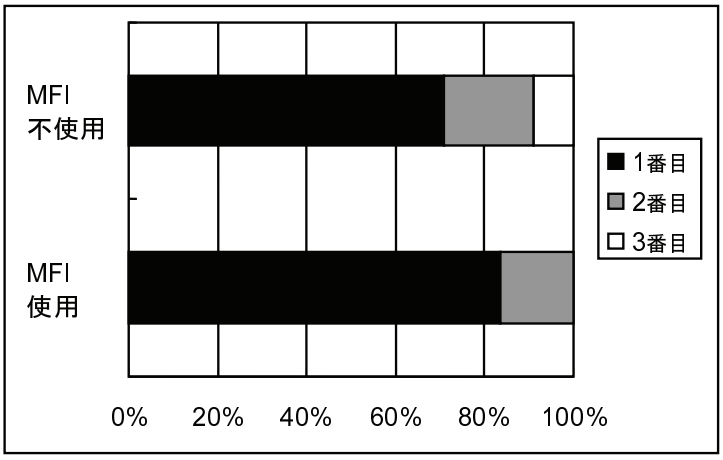

図 5.9: 指定対象物の推定リスト内での順位

の印は，測定地点の正しい位置を表し，白抜きの印は推定された位置（11回分）を表す．推定された位置は，正しい位置から平均で0.85メートル(80 %の推定位置が1.51メートル以内) 離れていた．一方，MFIを用いる場合の推定位置を図5.8に示す．この場合，推定された位置は，正しい位置から平均で0.35メートル(80 %の推定位置が0.60メートル以内) 離れていた．2つの実験結果（MFIを用いる場合と用いない場合）は，同じデータに基づいて推定を行った．これら結果より，MFIを用いることにより，推定された位置の精度が向上することが分かる．すなわち，提案システムは，MFIを用いることにより屋内においても有用であるといえる．

次に，方向依存サービスの有用性についての評価を行った．それぞれの地点で位置を推定した後，図5.5にあるディスプレイのうちの１つを指し示してその方位角を測定した．ユーザの位置確率密度と，測定されたディスプレイの方位角を元に，指定されたディスプレイ（指定対象物）がどれなのかをシステムが推定する (5.1.5節参照)．推定結果は，確率の高い順に並んだ対象物のリストとなる．この実験は，ディスプレイを見ることが出来る，図中の地点3–7のみで行った．高確率順に並んだ推定結果のリスト内での，指定対象物の順位の比率を，図5.9に示す．MFIを用いることにより，84 %の試行において，指定対象物は推定結果のリストの先頭に来ていることが分かる．一方，MFIを用いない場合でも，74 %の試行においてリストの先頭に来ている．この結果より，提案手法はここで想定した方向依存サービスを提供するために十分な指定対象物の特定精度を持っているといえる．また，現在のシステムでは２次元平面上で計算を行っているが，対象物の高さや指し示している仰角等の３次元の情報を用いれば，より結果が向上するものと思われる．これは今後の課題である．

5.3 関連研究

本システムに関連した研究としては，以下のようなものが挙げられる．

5.3.1 位置依存サービス

既に様々な位置依存サービスが提案されている．SpaceTag[17]では，位置および時間を限定して情報を発信できる．kokono Search[16]は，WWWドキュメントに含まれる住所等の位置情報を自動抽出し，位置情報に基づいたWWW検索を可能としている．Follow-me Application[4]では，ユーザに最も近いディスプレイが自動的に

作業環境として選択される．

提案したシステムの最大の特徴は，ユーザの位置だけでなく方向も利用したより高度な位置情報サービスである，方向依存サービスを提供できることである．位置のみの場合，ユーザは「今いる場所」の情報を得ることができる．方向を併用することにより,「今見ているもの」「今指し示しているもの」など，より細かな情報を得ることができる．我々は，方向依存サービスの他の例として，クライアント端末をレーダのように回転させて「どの方向に望んでいるサービスがあるか」を検索するサービスを計画している．指し示した方向にサービスが存在する場合，振動や音でユーザに通知する．また，ふらっと [57] システムのように，Augumented Reality を用いて現実空間に検索結果を付与するような方法も考えられる．

5.3.2 画像認識による指定対象物推定

指定対象物の推定のための手法としては，画像認識技術を用いたものが多く提案されている．InfoPoint[11] は，２次元マトリックスバーコードを対象物に貼り付け，カメラ内蔵端末でこれを撮影することにより，ユーザが指し示している対象物を特定する．しかし，対象物から遠く離れた場合はバーコードの認識が困難である．また，AirReal[12] では，部屋側面に取り付けられたカメラで，ユーザがレーザポインタで指し示している座標を画像認識する．文献 [58] の手法では，ユーザの人影を上面に取り付けたカメラで撮影し，その画像よりユーザの位置と腕の方位角を認識して，ユーザが腕で指し示している物体を特定する．しかし，このような手法は，カメラが撮影している場所・方向でしか使うことができない．一方，提案手法は，環境側の設備も安価であり，広い範囲での利用が可能である．

5.4 まとめ

本章では，方向依存サービスシステム Azim を提案した．本システムは，4 章で提案した位置推定手法を採用しており，より広い範囲でのシステムの利用を可能とするために，無線 LAN の基地局情報より得られる位置情報を活用する．また，方向依存サービスの例として，ユーザが指し示した対象物を推定し，その指定対象物に関連した応用サービスを提供する．

本システムの有用性を判断するために，屋内および屋外で評価実験を行った．その結果，本システムは方向依存サービスにとって十分な位置推定精度，および対象

物特定精度が得られることが分かった.

第6章　方向依存サービスシステムの実装と応用

本章では，5 章で提案した方向依存サービスシステムのプロトタイプシステムの実装，および応用について述べる．磁気センサと加速度センサを組み合わせた方向センサ，および Linux が動作する PDA を用いて，実際に携帯可能なクライアント端末 LocPointer を実装した．この端末上で，様々な応用サービスを動作させ，実際にユーザに利用してもらうことにより，提案システムの有用性を示す．

6.1　方向依存サービスシステムの実装

5 章で提案した方向依存サービスシステムの，プロトタイプシステムを実装した．ソフトウェア開発環境として，Java 2 Platform SDK 1.4, およびネットワーク機器間の連携を支援するミドルウェアである cogma[59] を用いた．ただし，現在のバージョンではマーカ・対象物・基地局の利用可能領域は考慮していない．

クライアント端末は情報管理サーバへ無線 LAN 経由で通信する．クライアント端末の動作画面を図 6.1 に示す．クライアント端末には方向センサが接続されており，図中 (1) のコンパス画面に現在指し示している方位角が表示される．位置の推定のため，ユーザはクライアント端末でマーカを指し示し，その色をボタンで入力する．これを複数回繰り返すことにより位置が推定される．図中 (2) のように，地図上で現在の位置と方向を確認できる．次に対象物の方向を指し示し，Find ボタン（虫眼鏡のアイコン）を押すと，指定対象物の推定が行われ，図中 (3) のように，指定対象物の候補がリスト表示される．自動的に確率の高い指定対象物が選択されるが，ユーザが他の候補を手動で選択することもできる．指定対象物を選択すると，それに関する応用サービスを受けることができる．具体的な応用サービスについては 6.2 節で述べる．

情報管理サーバ側では，4.3 節で提案した手法により，ユーザの位置確率密度の計算，および指定対象物の推定計算を行う．位置確率密度は 128×128 の 2 次元配列として計算した．また，方位角測定モデルは，4.3.1 節に基づき，標準偏差 σ を 5 度

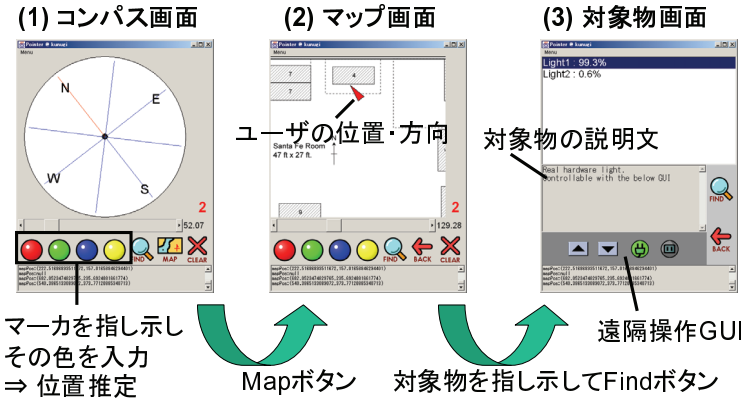

図 6.1: クライアント端末の動作画面

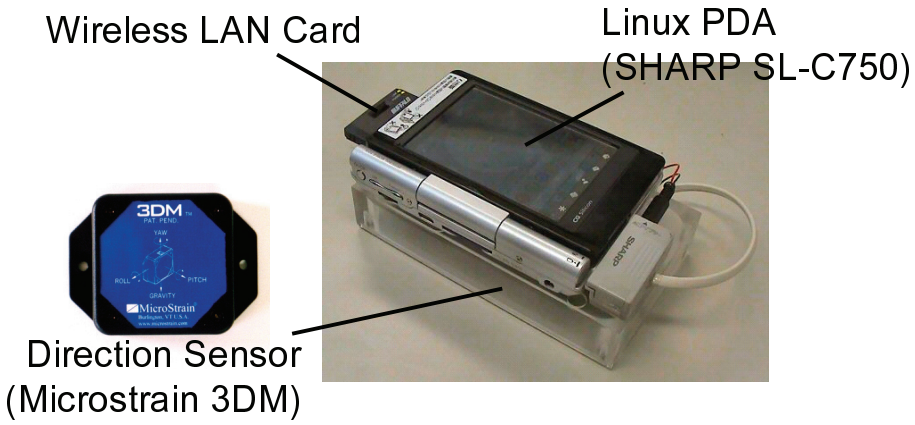

図 6.2: LocPointer: Linux PDA を用いた携帯型クライアント端末

とした正規分布を用いた.

6.1.1 LocPointer

我々は Linux が動作する PDA (SHARP SL-C750) を用い，図 6.2 のような実際に携帯可能なクライアント端末 LocPointer を製作した．方位角の測定には，3 軸の磁気センサと 3 軸の加速度センサを組み合わせた方向センサである，MicroStrain 社の 3DM[47] を用いた．J2ME(Java 2 Micro Edition) 環境で動作するクライアント端末のソフトウェアは，3DM センサをシリアル (RS-232C) 経由で制御し，無線 LAN(IEEE802.11b) 経由で情報管理サーバと通信を行う．5.2 節に示す本システムの評価実験も，この LocPointer を用いて行った.

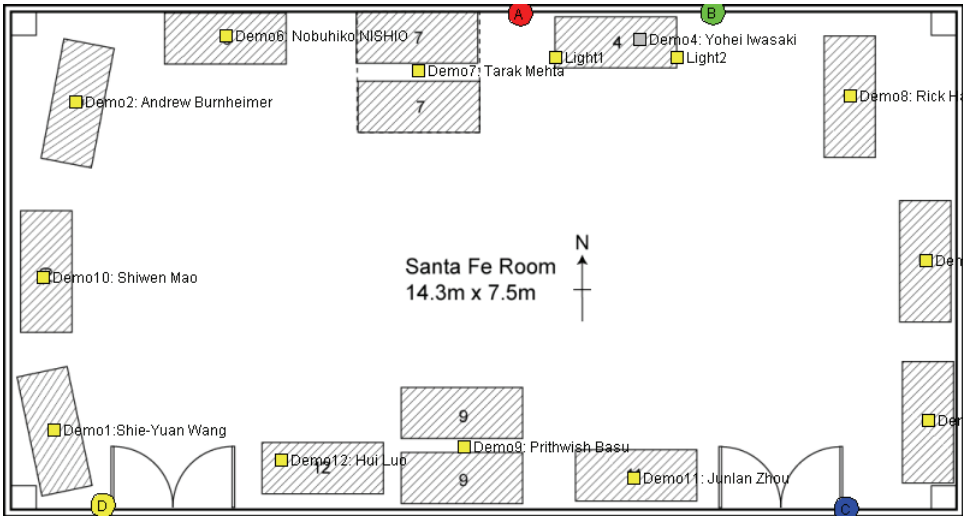

図 6.3: MobiCom2003 デモ 会場マップ

6.2 方向依存サービスの応用

方向依存サービスの具体的な応用例として，以下のようなサービスを実装した．これらのサービスを，実際にユーザに利用してもらうことにより，提案システムの実現可能性を示した．

6.2.1 MobiCom2003 Demonstration

我々は，モバイルコンピューティングに関する代表的な国際会議である MobiCom2003 において，図 6.3 のような会場で本プロトタイプシステムのデモを行った [60]．図 6.3 に示すように，２個のライト，および１１個の “デモ発表” を，対象物として登録した．まず，図中の A–D のマーカのうちの２つを指し示すと，位置が推定される．位置推定後，対象物を指し示して Find ボタンを押すと，クライアント端末である LocPointer 上に，図 6.4 のような画面が表示される．そして，対象物の種類に応じて，以下のような応用サービスを受けることが出来る．

- デモ情報サービス：他のデモを行っている机を指し示すと，著者名，概要など，そのデモに関する詳細を確認することができる．
- 万能リモコンサービス：対象物（機器）を GUI により遠隔操作できる．本デモでは，ライトを ON/OFF することができる．cogma ミドルウェアのコード移送機能により，GUI の実装コードは，対象物の機器から動的にロードされる．

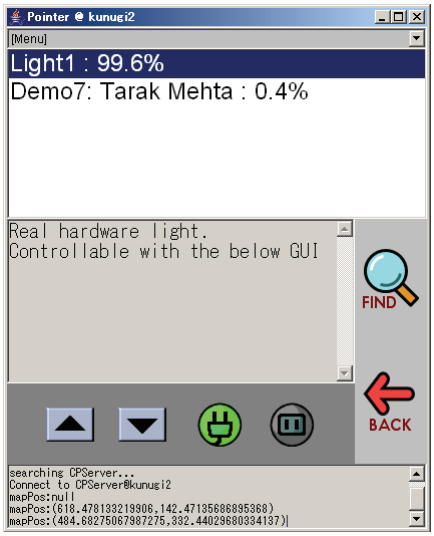

図 6.4: MobiCom2003 デモ クライアント画面

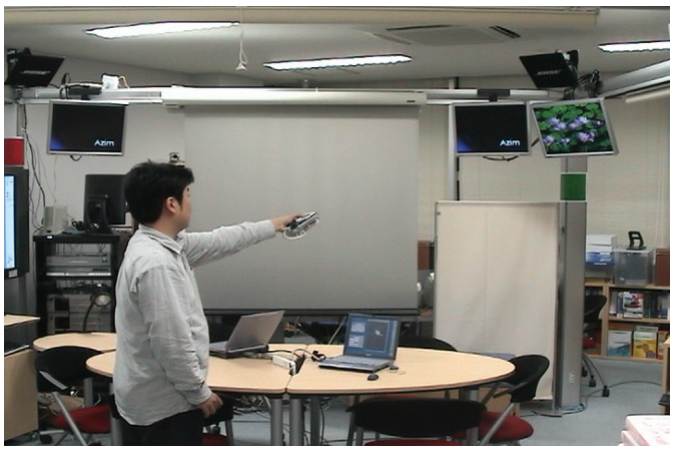

図 6.5: 映像の出力先ディスプレイを切り替えるアプリケーション

- デバイス接続サービス：ユーザは，対象となる機器を携帯端末で指し示し，Touch-and-Connect(3 章参照) のボタンインタフェースを GUI により遠隔操作できる．これにより，離れた機器間の接続を直接的に指示できる．本デモでは，２つのライト間を接続し，ライトの ON/OFF を連動させることができる．

提案手法のプロトタイプシステムの動作は，多くの見学者に理解してもらえた．一方で，Microstration の方向センサ 3DM[47] が高価である，端末の方位角だけでなく傾きも利用したほうがよい，などの意見も聞かれた．

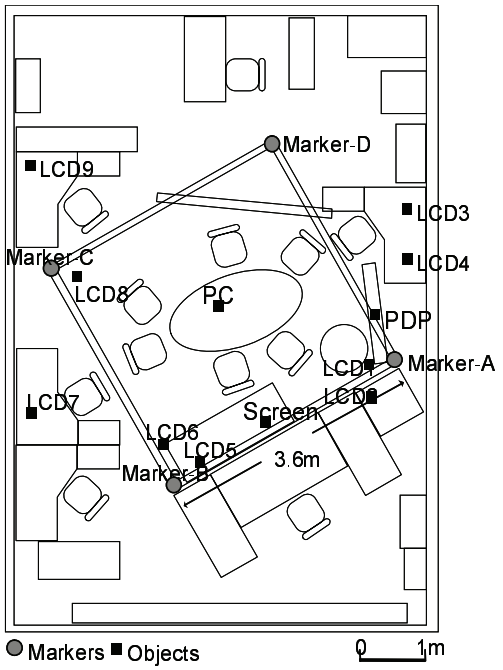

図 6.6: ディスプレイの配置図

6.2.2 cogma room

我々が構築したユビキタスコンピューティング環境のテストベッドルームである Cogma Room において，LocPointer でディスプレイを指し示すことにより，映像の出力先を直接的に切り替えられるアプリケーションを作成した．本サービスを使用している様子を，図 6.5 に示す．図 6.6 に示す１２個のディスプレイの中から１つを指し示し，(1)*get* **操作**: このディスプレイに表示されている映像 (の ID) を取得，または，(2)*put* **操作**: 取得した映像をこのディスプレイに表示，を行うことができる．これにより，例えば，小さいディスプレイに表示されている映像を大きいスクリーンに移動する，といった操作を，ディスプレイの ID や映像の ID を意識することなく，ディスプレイを指し示すだけで直接的に行うことができる．

6.2.3 RICA+

本プロトタイプシステムは，機器間連携のためのフレームワークの一部として使われることを目指しており，他のシステムと容易に融合することができる．我々は，提案システムと，動的にサービス構成の変更が可能な分散ディスプレイシステム RICA(Reconfigurable Inter-Communication Appliances)[61] とを融合させ，ディスプレイを直接的に指定できるように拡張したシステム RICA ＋を作成した．

図 6.7: RICA (Reconfigurable Inter-Communication Appliances)

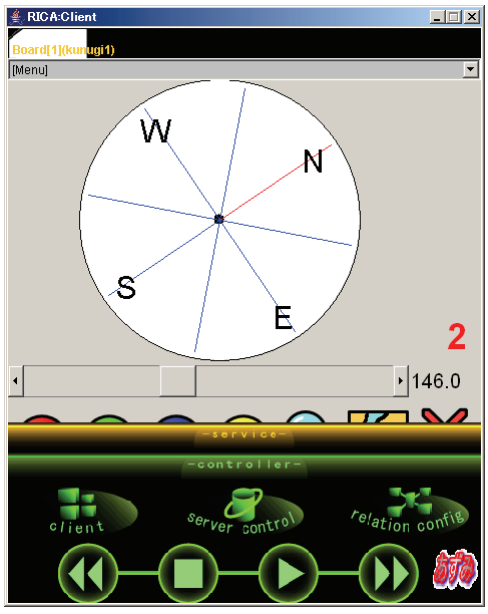

図 6.8: RICA ＋：Azim を用いた直接的指示が可能な分散ディスプレイサービス

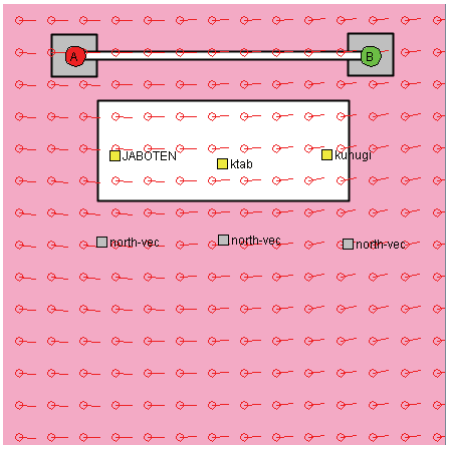

図 6.9: インタラクション 2004 デモ会場マップ

RICA は，複数のディスプレイ上に存在する様々なサービスを，図 6.7 に示すようなクライアント画面上で利用できる．しかし，ディスプレイを名前で指定する必要があるため，多くのディスプレイが存在する環境や，初めて訪れた環境では，実際のディスプレイとその名前との対応が取れず操作が煩雑となる．拡張されたシステム RICA+では，図 6.8 に示すように，RICA のクライアントシステム上に，提案システムである Azim システムが埋め込まれている．RICA+を動作させた LocPointer でディスプレイを指し示すと，RICA のサービス画面 (図 6.7) において，指し示したディスプレイが自動的に選択される．

インタラクティブシステムに関する代表的な研究会であるインタラクション 2004 で，本システムのデモンストレーションを行った [62]．図 6.9 に示すような会場で，2 つのマーカ A,B を指し示して位置を推定し，3 台のディスプレイ (JABOTEN, ktab, kunugi) を対象物とした．これらディスプレイを直接指し示して，RICA のサービスを利用することができる．地磁気が乱れている部分があったため，予め周辺の 3 箇所 (図中の north-vec) において磁気ベクトルを測定し，磁気分布情報 (5.2.2 節参照) を作成したところ，精度の高い推定を行うことが出来た．図中の丸印から伸びている線分は，磁気分布情報を表す．

6.3 まとめ

本章では，5 節で提案した方向依存サービスシステムの，実装および応用について述べた．Linux が動作する PDA を用いて，LocPointer という携帯可能なクライアント端末を実装した．また，数々の応用サービスを実装し，実際にユーザに利用し

てもらうことにより，提案システムの有用性を示した．

第7章　デバイス連携機構の自動分散化

一般に，連携のためのソフトウェアは複数の機器に分散するため，従来の連携ソフトウェア開発手法では，連携ソフトウェアの開発や保守が困難となり，また予め定められたアプリケーションプロトコルの範囲内でしか連携が行えない．本章では，事前に想定していない機器との連携機能も，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発・導入できるようにするために，連携ソフトウェアの自動分散化手法を提案する．特に，ワンチップマイコンなどの低レベルデバイス上で，端末間が直接通信を行うピアツーピア (P2P) 型の連携機構に着目する．

本手法では，機器間の連携機能を，連携ソフトウェアの分散を意識せず単一のソフトウェアモジュールとして記述する．処理の流れを解析することにより連携ソフトウェアを自動的に分散化し，連携に参加する機器にインストールすることにより，開発者は連携ソフトウェアの分散を意識することなく容易に連携機能を開発できる．

7.1　低レベルデバイス

近年，MOTE[31]，nRF24E1[32] などをはじめ，無線通信機能を搭載した低コストな端末が開発されつつある．これらを活用すれば，センサ・家電・文房具などの様々な小型機器がネットワークに参加することが期待される．

例えば，図 7.1 は，8bit の CPU，4KB の RAM，および 2.4GHz 帯での無線通信機能を搭載した，無線ワンチップマイコン nRF24E1 である．このような低レベルデバイスでは，リソースが限られているため，動的なメモリ確保やマルチスレッド等を利用することは難しく，静的なメモリ配置や最適化を考慮した設計が必要となる．

7.2　RPC 型連携と P2P 型連携

機器間連携フレームワークは，主に，Universal Plug and Play[8] や Jini[9] などの，単一制御ノードが遠隔サービスノードを集中制御して連携を実現する Remote-Procedure-Call (RPC) 型のフレームワークと，AMIDEN[6] や Touch-and-Connect(3

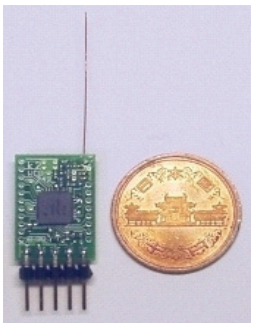

図 7.1: 無線マイコン nRF24E1

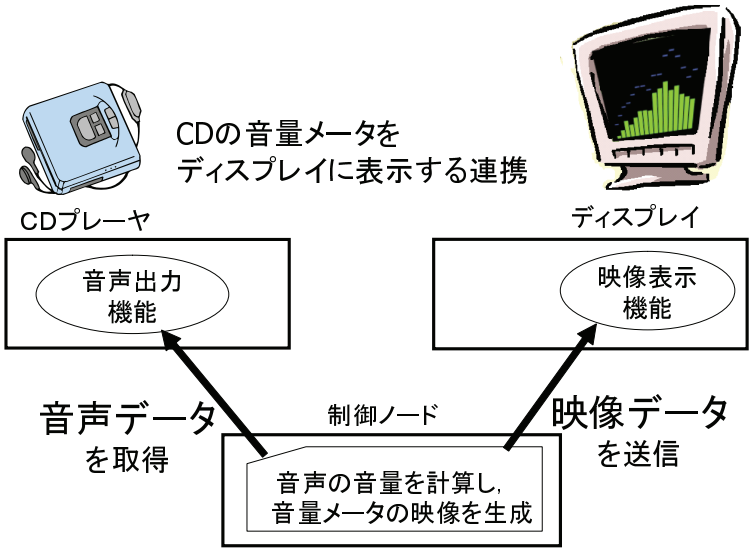

図 7.2: RPC 型連携

章参照) などの，機器間が直接通信して連携する Peer-to-Peer (P2P) 型のフレームワークに分類される．

本論文では，P2P 型フレームワークを実現する．この P2P 型フレームワークは，別途制御ノードを必要としない，特定の連携に特化した通信効率の良いアプリケーションプロトコルを使える，などの利点がある一方，連携に関するプログラムコードが複数端末間に分散し，その開発や保守は容易ではない．

ここで，連携の例として，CD プレーヤの音量を，ディスプレイに音量メータとして表示する連携を考える．図 7.2 は RPC 型の連携を，図 7.3 は P2P 型の連携を示したものである．RPC 型の連携では，制御ノードを介して，音声データと映像データを送受信している．一方，P2P 型の連携では，音声＋映像データに比べてより軽量なアプリケーションプロトコルである，音量データを送受信して連携を行うことができ，また連携時に制御ノードが不要である．しかし，連携に関するプログラム

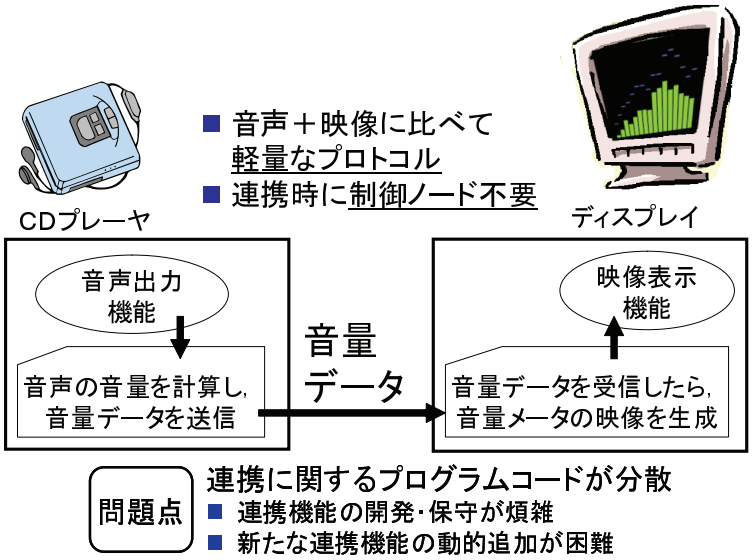

図 7.3: P2P 型連携

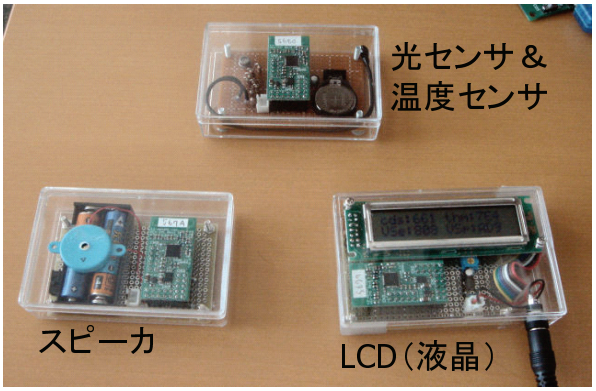

図 7.4: プロトタイプデバイス

コードが，音声の音量を計算する部分と，音声メータの映像を生成する部分に分割され，また音量データの送受信処理も記述しなければならず，ソフトウェアの開発や保守が煩雑となる．また，予めアプリケーションプロトコルを定め，連携機能を組み込んで機器を開発するため，機器生産後に連携の種類を動的に追加することが困難となる．

我々は実際，無線マイコン nRF24E1 を用いて，P2P 型の連携が可能な図 7.4 のようなプロトタイプデバイスを試作したが [63]，連携に関する機能が複数の端末に分散しているため，連携ソフトウェアの開発や保守が煩雑となった．また，P2P 型連携のフレームワークである Touch-and-Connect(3 章参照) では，機器間の連携を Java で実装しているが，イベントやアプリケーションプロトコルの定義，状態の同期処理等の処理を全て手動で記述しなければならない．

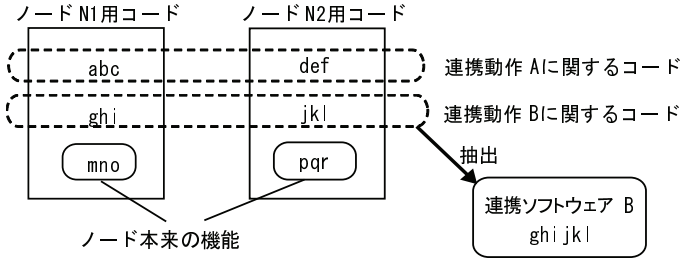

図 7.5: 複数の端末に分散する連携機構

7.3　連携ソフトウェアの単一モジュール化

本フレームワークでは，図 7.5 のように，複数のノードに分散する連携機能を抽出し，単一のソフトウェアモジュールとして扱う．連携ソフトウェアは，単一端末用のソフトウェアと同じように記述すればよく，容易に開発することが出来る．記述されたソフトウェアは，連携に参加する端末の機能に合わせて自動的に分散化され，各端末へインストールされる．すなわち，本フレームワークでは，RPC 型の連携ソフトウェアを，自動的に P2P 型の連携ソフトウェアに変換することにより，開発時には RPC 型の利点（単一モジュールで連携機能を開発可能，機器生産後に連携を追加可能）を得られ，実行時には P2P 型の利点（通信効率の良いアプリケーションプロトコルで直接通信可能，連携時に制御ノードが不要）を得られる．

7.4　連携ソフトウェアの自動分散化

本節では，単一端末の場合と同様の形式で書かれた連携ソフトウェアを，連携に参加する各端末のモジュール構成（どの機能をどの端末上で実行するべきかを示したもの）に従い，自動的に分散化する手法について述べる．命令単位での細粒度の分散化を行うため，文献 [25][26] 等の，RPC(Remote Procudure Call) を用いるクラス単位の分散化手法と比較して，効率の良いアプリケーションプロトコルを生成できる．

分散化の手順を図 7.6，および以下に示す（図中の番号に対応）．

1. 連携ソフトウェアのソースコードを，プログラムの内部表現に翻訳する．内部表現は，複数の命令と，各命令間の処理の流れを表すものである．**命令**とは，関数呼び出し，演算などを表すソースコードレベルでの要素であり，１つの命令が単一の端末上で動作する（複数の端末の機能を使わない）必要がある．

2. 各命令を実行する端末を決定する．特定の端末の機能を使う命令は，その端末

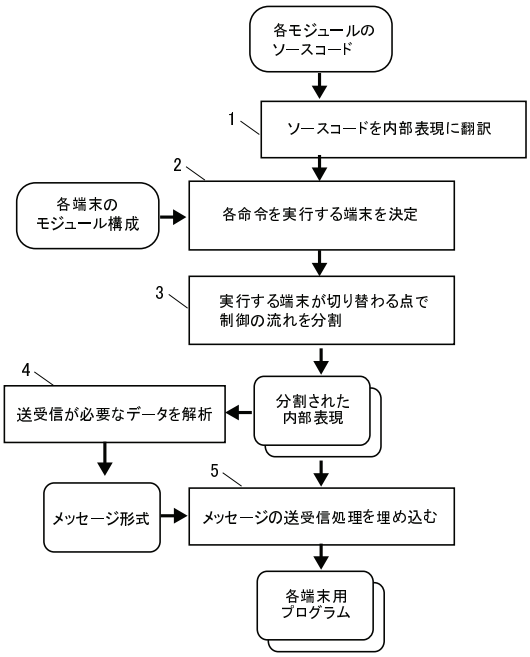

図 7.6: 分散化の手順

上で実行する．そうでないものは，何らかの方法によって実行する端末を決定する．例えば，直前の命令と同じ端末としたり，幾つかの選択肢を試してみて最適な結果を使用するなどの方法がある．また，結果が改善するように，内部表現を最適化することが望ましい．最適化のための指標としては様々なものを利用者が選択できるべきである．例えば，通信回数を減少させるために，命令の実行順序の制約を解析し，同一の端末上での処理がまとまるように，命令の順序を並び替える．

3. 実行する端末が切り替わる点で，制御の流れを分割する．これにより，プログラムの内部表現は各端末ごとに分割される．

4. 制御の流れを分割した点で，送受信するべきデータを解析し，メッセージの形式を決定する．送受信するべきデータを解析する方法とは，例えば，ソースコード内で使用する全ての変数を渡す，生きている変数 [64]（現在の値が今後使用される可能性のある変数）のみを渡す，移動先で使用される可能性のある変数のみを渡す，値が更新された（移動先と値が異なる）可能性のある変数のみを渡す，などの方法が挙げられる．また，これらの方法の組み合わせも可能である．

   ここで，データとは，ソフトウェアの実行状態のことであり，変数の値だけに限る必要はなく，例えば，サブルーチンの呼び出し履歴，プログラムの現在の

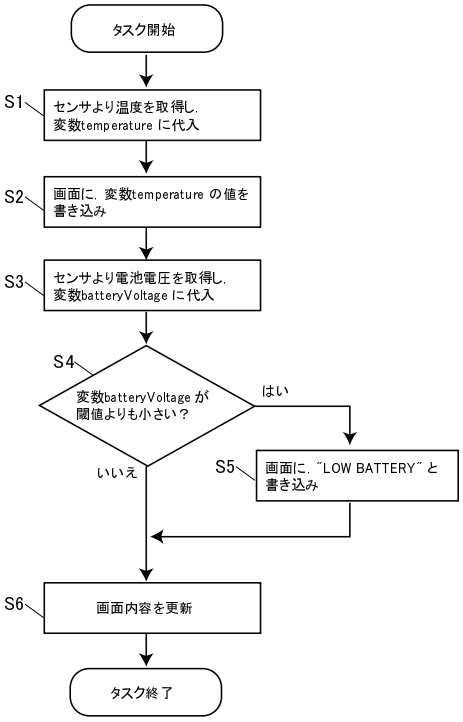

図 7.7: 分散化前のタスクのフローチャート

実行箇所などを含むこともできる．

5. 生成されたメッセージ形式に従い，プログラムの内部表現にメッセージの送受信処理を埋め込み，各端末用の最終的なプログラムを生成する．メッセージの送受信処理とは，制御の流れを分割した点において，移動元端末と移動先端末のデータ（の一部）を一致させる処理である．例えば，移動元端末では，メッセージに変数の値を入れて移動先に送信し，移動先端末では，そのメッセージを受信して，メッセージから値を取り出して変数に代入するような処理である．

   また，プログラムの処理が終了する直前に，終了後も使用する可能性のあるデータ（静的変数の値など）を端末間で一致させるように，メッセージの送受信処理を埋め込む．

   また，今後の発展としては，命令間の依存関係を解析し，互いに依存関係のない命令を，各端末間で並行して実行させることも挙げられる．

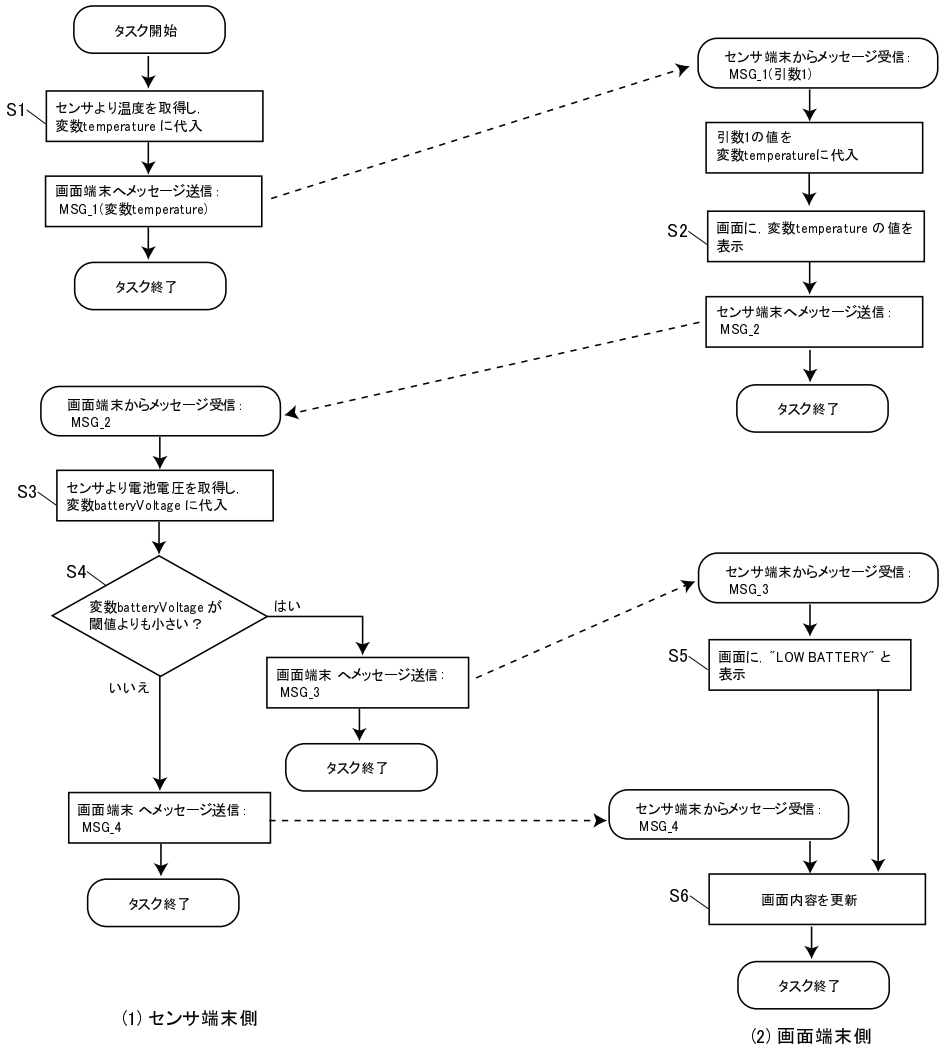

図 7.8: 分散化後のタスクのフローチャート

7.4.1 分散化の例

分散化の例として，センサより取得した温度の値を画面に表示し，またセンサより取得した電池電圧の値が，特定の閾値よりも小さい場合は,「LOW BATTERY」というメッセージを画面に表示する，という連携ソフトウェアのプログラムを考える．このプログラムを，センサから値を取得する機能を持つ，センサ側端末と，画面に値を出力する機能を持つ，画面側端末に分散化する．

図 7.7 は，分散化する前のプログラムのフローチャートである．S1〜S6 が命令を表す．S1,S3 はセンサ側端末で，S2,S5,S6 は画面側端末で実行する必要がある．S4 を実行する端末は任意であるが，ここでは直前の命令と同じセンサ側端末とした．

図 7.8 は，提案手法により分散化された，各端末ごとのプログラムのフローチャー

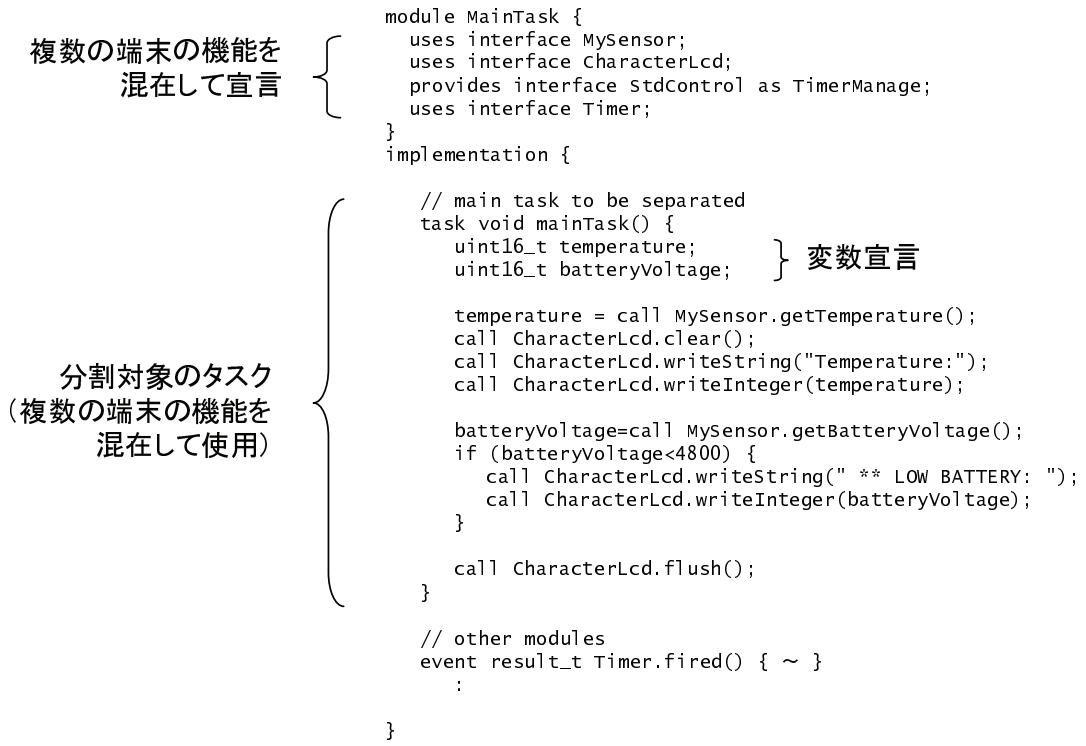

図 7.9: 分散化前の nesC によるモジュール (一部省略)

トである．実行する端末が切り替わる点は，S1 → S2, S2 → S3, S4 → S5, S4 → S6 の 4 箇所であり，それぞれの点で送受信するメッセージに MSG_1, MSG_2, MSG_3, MSG_4 という名前をつけた．各点で送信するべきデータは，生きている変数 [64]（現在の値が今後使用される可能性のある変数）のみとした．すなわち，MSG_1 では変数 temperature であり，MSG_2, MSG_3, MSG_4 では存在しない．

7.4.2　nesC による記述例

7.4.1 節に挙げた連携ソフトウェアの例を，TinyOS[65] をターゲットとしたプログラミング言語 nesC[37] のモジュールとして記述した．

分割前のモジュールのソースコードを図 7.9 に示す．MySensor, TimerManage, Timer インタフェースは，センサ端末側の機能であり，CharacterLcd インタフェースは，画面端末側の機能である．分散化の対象となるプログラムは，タスク mainTask() であり，両端末の機能を混在させた形で自然に記述できる．

nesC のモジュールを構成する要素としては，変数，task，command，event などがある．変数は基本的に全端末のモジュールで共有する．ただし，自端末で全く使用しない変数は除くことができる．また，スタックフレームを保存する機能は想定


```
module MainTask_SensorNode {
   uses interface MySensor;
   provides interface StdControl as TimerManage;
   uses interface Timer;
   //
   uses interface SendMsg;
   uses interface ReceiveMsg;
}
implementation {
   // local variables
   uint16_t mainTask_temperature;
   uint16_t mainTask_batteryVoltage;

   // separated task
   task void mainTask() {
      mainTask_temperature = call MySensor.getTemperature();
      mainTask_send_msg_1();
   }

   task void mainTask_2() {
      mainTask_batteryVoltage=call MySensor.getBatteryVoltage();
      if (mainTask_batteryVoltage<4800) {
         mainTask_send_msg_3();
      }else {
         mainTask_send_msg_4();
      }
   }

   // message stub
   TOS_Msg msg_data;

   void mainTask_send_msg_1() { ~ }
   void mainTask_receive_msg_2(msg_base_t *bmsg) {
      post mainTask_2();
   }
   void mainTask_send_msg_3() { ~ }
   void mainTask_send_msg_4() { ~ }
   event TOS_MsgPtr ReceiveMsg.receive(TOS_MsgPtr m) {
      msg_base_t *msg=(msg_base_t *)m->data;
      // dispatch
      switch(msg->msg_typeid) {
      case MainTask_mainTask_msg2_typeid:
         mainTask_receive_msg_2(msg);
         break;
      }
      return m;
   }

   // other modules
   event result_t Timer.fired() { ~ }

      :
}
```

図 7.10: 分散化後のセンサ端末側モジュール (一部省略)

画面端末側の機能を宣言

変数宣言

分割されたタスク

メッセージ送受信処理

```
module MainTask_LcdNode {
   uses interface CharacterLcd;
   //
   uses interface SendMsg;
   uses interface ReceiveMsg;
}
implementation {
   // local variables
   uint16_t mainTask_temperature;
   uint16_t mainTask_batteryVoltage;

   // separated task
   task void mainTask_1() {
      call CharacterLcd.clear();
      call CharacterLcd.writeString("Temperature:");
      call CharacterLcd.writeInteger(mainTask_temperature);
      //
      mainTask_send_msg_2();
   }

   task void mainTask_3() {
      call CharacterLcd.writeString(" ** LOW BATTERY: ");
      call CharacterLcd.writeInteger(mainTask_batteryVoltage);
      //
      post mainTask_4();
   }

   task void mainTask_4() {
      call CharacterLcd.flush();
   }

   // messaging stub
   TOS_Msg msg_data;

   void mainTask_receive_msg_1(msg_base_t *bmsg) {
      MainTask_mainTask_msg1_t *msg=(MainTask_mainTask_msg1_t *)bmsg;
      mainTask_temperature=msg->temperature;
      post mainTask_1();
   }
   void mainTask_send_msg_2() { ～ }
   void mainTask_receive_msg_3(msg_base_t *bmsg) { ～ }
   void mainTask_receive_msg_4(msg_base_t *bmsg) { ～ }

   event TOS_MsgPtr ReceiveMsg.receive(TOS_MsgPtr m) { ～ }
      msg_base_t *msg=(msg_base_t *)m->data;
      // dispatch
      switch(msg->msg_typeid) {
      case MainTask_mainTask_msg1_typeid:
         mainTask_receive_msg_1(msg);
         break;
      case MainTask_mainTask_msg3_typeid:
         mainTask_receive_msg_3(msg);
         break;
      case MainTask_mainTask_msg4_typeid:
         mainTask_receive_msg_4(msg);
         break;
      }
      return m;
   }

      :
}
```

図 7.11: 分散化後の画面端末側モジュール (一部省略)

しないため，ローカル変数はモジュール変数へと変換する．task 要素は，本節で提案した自動分散化手法に基づいて，細粒度に（命令単位で）各端末へと分散化され，対応するメッセージ送受信処理が組み込まれる．command，event 要素は，スタックフレームを保存する機能無しに制御の流れを分割することが困難なため，細粒度での分散化は想定せず，単に関数全体をインタフェースが属する端末側のモジュールへと移動する．

図 7.9 に示す分割前のモジュールを，本手法に基づき，センサ端末側モジュールと画面端末側モジュールに手動で分割したものを，それぞれ図 7.10，図 7.11 に示す．分割後のこれらのモジュールは，TinyOS エミュレータである EmTOS[66] 上で，実際に連携動作することを確認した．

7.5 遠隔端末へのソフトウェアのインストール

本手法では，実際に連携動作を行う低レベルデバイスであるシンプルノード（無線通信機能を持つ 8bit マイコン程度を想定）と，連携の設定やソフトウェアのビルドを行うための高機能なスーパーノード（PC，PDA，携帯電話などを想定）の 2 種類の端末を用いる．スーパーノードは設定時のみ必要であり，実際の連携動作時には必要ない．スーパーノード上では連携ソフトウェアの自動分散化機構が動作しており，ユーザが希望する連携構成に基づき，必要な連携ソフトウェアを統合して自動分散化（ノードごとに分割＆アプリケーションプロトコルを生成）し，各ノード用のプログラムを生成する．連携構成が変わる度に，各ノードのプログラムコードを再コンパイルするため，静的な最適化を用いたコンパクトなコードを生成でき，低レベルデバイスに適する．

図 7.12 は，自動分散化により生成された各端末ごとのプログラムを，遠隔の端末（シンプルノード）にインストールする手順の詳細を示す．ここでは，２台の端末 N1,N2 で実行する例を示しているが，１台でも，３台以上でもかまわない．まず，プログラムをコンパイルし，実行する端末向けのメモリイメージを生成する．実行する端末（シンプルノード）には，あらかじめ遠隔でメモリの読み書きを出来る機能がインストールされている．生成されたメモリイメージを，この機能を使って実行する端末に遠隔で書き込むことにより，プログラムのインストールを行う．また，実行する端末のメモリの一部を遠隔で読み出し，書き込むメモリイメージに反映させることにより，実行する端末上の現在のデータを保存したまま，プログラムを更新することができる．メモリ上のどのアドレスの内容をどのアドレスに書き込めばよ

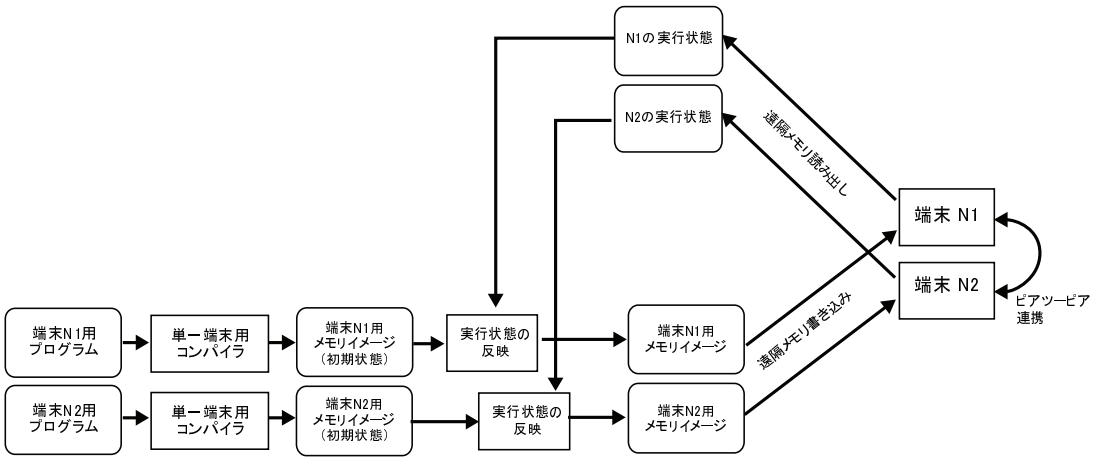

図 7.12: 遠隔端末のソフトウェアの更新

いかは，コンパイル時に生成されたメモリマップ（各データがメモリ上のどのアドレスに配置されているかを示した情報）などに基いて決定する．

7.6　まとめ

本節では，事前に想定していない機器との連携サービスも，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発，導入できるようにするために，連携ソフトウェアを自動的に分散化する手法を提案した．本手法を用いることにより従来ノードごとに別々に開発していたソフトウェアを，連携動作単位で開発することができ，連携機能の容易な開発が可能となる．また遠隔のノードのプログラムコードの更新を，一箇所で集中的に行えるため，機器間の連携動作の容易な構成，導入が可能となる．

現在，代表的な低レベルデバイスである TinyOS[65] 用に書かれた nesC[37] のプログラムモジュールを，7.4.2 節の例のように，提案手法に基づいて自動分散化するプロトタイプシステムの実装を進めている．

第8章　あとがき

8.1　まとめ

本論文では，ユビキタスコンピューティング環境における機器間連携フレームワークに関する研究について述べた．複数の機器を連携させた高度なアプリケーションサービスが実現され，我々の生活を支援することが期待できるが，多数の機器が存在するユビキタスコンピューティング環境では，設定が煩雑，動的な機能拡張が困難，連携ソフトウェアの開発が困難など，様々な問題が発生しうる．

本論文で提案した手法を用いれば，以下のような機器間連携フレームワークが実現できる．

- ２つの機器のボタンを押すだけで，機器間の接続を指示可能
  - 複数のユーザが存在しうる環境においても誤接続を防止
- 遠くの機器に対しても，携帯端末で指し示すことにより指定可能
  - サービスを安価に展開可能
  - 地磁気の乱れに対してもロバスト
- 機器間の連携機能を，連携ソフトウェアの分散を意識せずに機器生産後に容易に開発，導入可能
  - 単一マシン用の連携ソフトウェアを自動的に分散化
  - 低レベルデバイス向けのコンパクトなコードを生成可能

本論文の成果を以下にまとめる．

3 章では，接続したい両機器のボタンを押すことにより，機器間の接続を直接的に指示できるフレームワーク Touch-and-Connect を提案した．本手法では，状態を表示可能なボタンを用いてユーザの操作を排他制御し，複数のユーザが独立に操作を行う状況においても誤接続を防止する．ブロードキャスト通信を用いた本手法のプロトコルは，管理サーバを必要とせず，動的な端末の入退出も考慮しているため，

必要に応じて一時的に構築されるアドホックネットワークでも利用可能である．また，セキュリティと使いやすさの向上のため，グループを作成して機器間の接続可能性を制限できる．本フレームワークのプロトタイプシステムを実装してその実現可能性を示し，また，被験者実験により，インタフェースとしての使いやすさ，および本手法により誤接続が防止できることを示した．

4,5,6 章では，方向依存サービスに関する研究について述べた．方向依存サービスとは，ユーザの位置と方向を用いたより高度な位置情報サービスである．まず，4 章では，方向依存サービスのための位置推定手法について述べた．本手法では，位置が既知の複数のマーカの方位角を，ユーザが指し示して手動で測定することにより，ユーザの位置を推定する．本手法によって推定される位置の精度を評価するために，方向センサを用いて実験を行った．5 章では，方向依存サービスを運用するためのシステム Azim を提案した．本システムでは，ユーザにサービスを提供するために無線 LAN を用いており，無線 LAN の基地局情報を活用して広い範囲での運用を可能としている．本システムの有用性を判断するために，屋内および屋外で評価実験を行った．屋外においては，30 メートル離れた 2 つのマーカを用いた場合，平均誤差 1.9 メートルの位置精度が得られ，3 つのマーカを用いれば，平均誤差 0.78 メートルの位置精度が得られた．また屋内においては，3.6 メートル離れた 2 つのマーカを用いた場合，平均誤差 0.85 メートルの位置精度が得られ，MFI(磁気分布情報) を用いれば平均誤差 0.35 メートルに位置精度が向上した．6 章では，プロトタイプシステム実装，および応用や実際の運用について述べた．磁気センサと加速度センサを組み合わせた方向センサ，および Linux が動作する PDA を用いて，実際に携帯可能なクライアント端末 LocPointer を作成した．この端末上で，様々な応用サービスを動作させ，実際にユーザに利用してもらうことにより，提案システムの有用性を示した．

7 章では，連携ソフトウェアの自動分散化手法を提案した．本手法は特に，ワンチップマイコンなどの低レベル実行環境上で，端末間が直接通信を行う P2P 型連携機構に着目している．RPC 型の連携ソフトウェアを，本手法により自動的に分散化し，P2P 型の連携ソフトウェアに変換することにより，RPC 型の利点と P2P 型の利点の両方を持った機器間連携フレームワークを実現できる．

提案したこれらのアプローチは互いに補完的であり，組み合わせることができる．例えば，6.2.1 節で示したように，方向依存サービスシステム Azim と，直接的な接続指示フレームワーク Touch-and-Connect を組み合わせることにより，近くの機器，遠くの機器のどちらも，アドレスや名前を使わずに直接的に指定可能な機器間連携

フレームワークを実現した．さらに，7 章で提案した連携ソフトウェアの自動分散化手法を，これらと組み合わせることにより，利用者および開発者の双方にとって，容易に利用できる機器連携フレームワークが実現できる．

8.2 提案手法の応用可能性

本論文で提案した各々のアプローチは，本論文で示した実施形態だけでなく，様々な用途に汎用的に応用可能である．

従来の一般的な機器間連携フレームワーク [5][6] では，ユーザが機器間の連携を指示する際に，まず「連携（アプリケーション）の種類」を指定する必要がある．一方，3 章で提案した Touch-and-Connect フレームワークでは，ユーザは接続したい「2 つの機器」を指定し，連携の種類は両機器の種類に応じて自動的に推定されるという新しい連携指示のモデルを用いた．これは，機器間の連携指示に，オブジェクト指向の多態性 (polymorphism) の考え方を導入したものと見なせる．この連携指示のモデルは，機器の数や連携の種類が増加するユビキタスコンピューティング環境において，ユーザの望んでいる連携状態を効率よく絞り込むために応用可能である．

3.3.2 節においては，多数の人間が参加する無線ネットワークでは，他人の操作との干渉が発生し，意図しない誤接続が発生しうることを示した．これは無線ネットワーク上で動作する機器を設計するための指針となりうる．このような問題に対処するためには，ユーザに ID を持たせて識別する方法 [42] が一般的であるが，Touch-and-Connect のロックメカニズム (3.1.1 節参照) では，ユーザの操作間の排他制御を行うという新しい考え方を提案した．この考え方は機器間の連携指示に限らず，例えばユーザ識別が不要な分散共有クリップボードの実現など，多くの分野に応用可能である．

3.1.3 節では，機器間の接続性を制限するためにグループ ID を用いてグループを作り，これをユーザが簡単に設定可能とするために，グループ設定専用のデバイス（グループチェンジャ）を用いるという手法を提案した．この手法は，アドホックネットワークにおけるセキュリティモデルとして，様々なフレームワークに応用可能である．

4 章,5 章では，ユーザの位置だけでなくその方向も利用する方向依存サービスを提案し，多数の機器が存在するユビキタスコンピューティング環境においての方向依存サービスの有用性を示した．これは，コンテキストアウェアの研究分野に新たなアプローチを提案したと言え，指し示された対象物の推定だけでなく，サービス

の検索など，様々なサービスに応用可能である．

4 章では，ユーザが手動で能動的に位置を測定するという，新しい位置推定のアプローチを示した．ユーザの能動的な測定無しに詳細な位置が取得できないことが保障されるため，このアプローチは，プライバシーを重視する位置情報サービスを実現する場合に応用可能である．

4.3 節では，提案した手動の位置推定手法のために，位置に関する情報を活用した確率モデルを提案した．この確率モデルは，地磁気分布や無線 LAN の基地局情報だけでなく，他のセンシング情報にも柔軟に適用可能である．

7.3 節では，複数の端末に分散して動作する連携機能を抽出し，単一のモジュールとして管理するという新しいコンポーネントモデルを提案した．このモデルは，アスペクト指向 [21] の考え方を分散ソフトウェアに適用したものであると言え，分散ソフトウェア開発を効率化するための手法に広く応用可能である．

7.5 節では，ユーザが希望する連携状態（利用したいアプリケーションサービス）が変化する度に，スーパーノードでプログラムコードを再コンパイルし，シンプルノードに転送するというソフトウェア実行モデルを提案した．このアプローチは，再コンパイル時に静的な最適化を行うことにより，不要なコードを除いたコンパクトなメモリイメージが生成できるため，メモリサイズに制限がある低レベルデバイス向けのソフトウェア実行モデルとして，広く応用可能である．

また，7 節で提案した，ソフトウェアを自動的に分散化するアプローチは，機器間連携フレームワークだけでなく，例えば SETI@home[67] のようなピアツーピア型の並列計算など，様々な分野への発展が期待できる．

8.3　今後の課題

残された今後の課題について述べる．まず，機器間の接続指示フレームワーク Touch-and-Connect (3 章参照) に関しては，以下のような課題が挙げられる．

- **拡張性・柔軟性の向上**: 現在，本フレームワークでは，２つの機器間の接続可能性や通信プロトコルをあらかじめ定めておく必要がある．機器間の連携のためのフレームワークは，事前に想定していない機器とも連携が行えるような，拡張性や柔軟性を持っていることが望ましい．このような拡張性のあるフレームワークを実現するためには，プログラムコード移動技術が適している [68][69]．我々は既に，アドホックネットワーク環境を想定したモバイルエージェントシステム cogma[59]，ソフトウェアの動的更新手法 [70], および連携

ソフトウェアの自動分散化手法 (7 章参照) などを提案しており，本手法と組み合わせることにより，機器間連携のためのより拡張性のあるフレームワークを実現できる．

- **隠れ端末問題**: 現在，本手法のプロトコルでは，ネットワーク上の全ノードが互いに直接通信可能であると仮定している．しかし，広範囲の無線アドホックネットワークでの利用を考えた場合，隠れ端末問題を考慮する必要がある．現状では，電波到達範囲の境界付近において，リクエストリストの要素が 2 つになってしまい，誤接続を引き起こす可能性がある．この問題を解決するためには，リクエストが行われる領域への新たなリクエストの発生を禁止するために，リクエストが行われる領域の 2 倍以上の大きさの領域がロックされる必要がある．

  ProxNet[71] では，無線 LAN の電波強度を端末間の距離を測る尺度として利用している．本手法でも同様に，無線メッセージの電波強度を用いてリクエストとロックの範囲を制限することにより，隠れ端末問題の解決が期待できる．具体的には，3.1.4 節のプロトコルにおいて，Request メッセージの電波強度が閾値 p_r 以下の場合はリクエスタの Type によらず接続不可能（リクエストされない）とし，また閾値 p_l 以下の場合はロックされなくする．ただし，p_r は，本手法の適用を想定している “手の届く範囲”（5m 程度）に対応した電波強度とし，p_l は，前者の 2 倍以上の距離に対応した電波強度とする．

また，方向依存サービスシステム Azim (5 章参照) に関しては，以下のような課題が挙げられる．

- ***MFI* の効率的な学習**: 磁気分布情報 (MFI) を効率的に学習する手法を検討する必要がある．この分布を用いれば，5.2.2 節で述べたように地磁気が乱れている環境においても精度の高い測定を行うことができる．例えば，初期段階では他の位置測定方法（絶対的な方位角ではなく，マーカ間の相対角度を用いる方法など）を使用し，サービスを日々利用するにつれて，MFI が徐々に学習されていく方法などが考えられる．

- **3 次元空間への拡張**: 本手法では，まずは簡単化のために 2 次元平面を対象としたが，方向センサから得られる方位角，仰角両方を使うことにより，本手法を 3 次元空間に適用することを検討している．

- **情報管理サーバの分散化**: 現在のシステムでは，位置情報は単一のサーバで管理している．しかし，広い範囲での利用を考えた場合，位置情報やクライアント端末の管理を分散させればスケーラビリティを向上できる．

- **ユーザビリティの評価**: マーカを指し示す操作を不便に感じるユーザがいるかもしれないため，手動の位置測定手法のユーザビリティを定量的に評価することが望ましい．指し示し操作はわずが数秒程度のため，応用サービスが魅力的であれば本手法はユーザに受け入れられるものと予想している．

- **アプリケーション**: より高度な位置情報サービスとして，我々は方向依存サービスを提案した．物体を指し示す応用だけではなく，他の応用サービスも検討することが望ましい．例えば，クライアント端末を様々な方向へ向けることにより，望んでいるサービスがどちらの方向に存在するか検索するサービスが考えられる．

連携ソフトウェアの自動分散化手法 (7 章参照) に関しては，プロトタイプシステムを実装し，実際のアプリケーションを動作させて提案手法を評価するのが課題である．また，プログラムの命令間の依存関係を解析する Program Dependency Graph[72] などの手法を応用することにより，分散化を行う際に命令の実行順序を並び替え，端末間の通信回数をより少なくするような最適化を行うことも検討している．

謝辞

本研究を進め，まとめるにあたり，日頃より懇切丁寧な御指導と御鞭撻を頂いた，名古屋大学助教授の河口信夫先生に厚く感謝いたします.

また，本研究の初期の段階より御指導と御討論を頂いた，当時名古屋大学教授で，現在，愛知県立大学教授の稲垣康善先生に厚く感謝いたします.

また，本論文をまとめるにあたり，貴重な御示唆と御指導を頂いた，名古屋大学教授の阿草清滋先生，および坂部俊樹先生に厚く感謝いたします.

また，本研究を進めるにあたり，熱心に御討論くださった，名古屋大学の松原茂樹助教授，外山勝彦助教授，杉野花津江助手，山口由紀子助手，ムフタル・マフスット助手，小川泰弘助手，加藤芳秀助手に厚く感謝いたします.

また，研究に関する御討論をはじめ，研究以外の面でもいろいろお世話になりました，EL-Project の皆様，および稲垣研究室の皆様に心から感謝いたします.

発表文献リスト

種別	論文名	関連する章
論文誌	岩崎陽平, 河口信夫, 稲垣康善, “ Touch-and-Connect: ユビキタス環境における接続指示フレームワーク, ”情報処理学会論文誌, Vol.45 No.12, pp. 2642–2654, December 2004.	3 章
論文誌	Yohei Iwasaki, Nobuo Kawaguchi, Yasuyoshi Inagaki, “ Azim: Direction Based Service System for both Indoor and Outdoor, ”IEICE Transactions on Communications (Special Issue: Ubiquitous Networks), Vol.E88-B No.3, pp. 1034–1044, March 2005.	4 章 5 章
国際会議	Yohei Iwasaki, Nobuo Kawaguchi, Yasuyoshi Inagaki. “Touch-and-Connect: A Connection Request Framework for Ad-hoc Networks and the Pervasive Computing Environment,” First IEEE Annual Conference on Pervasive Computing and Communications (PerCom 2003), pp. 20–29, Mar. 2003.	3 章
国際会議	Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki, “Azim: Direction Based Service using Azimuth Based Position Estimation,” The 24th International Conference on Distributed Computing Systems (ICDCS 2004), pp. 700–709, Mar. 2004.	4 章
国際会議	Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki, “Design, Implementation and Evaluations of a Direction Based Service System for both Indoor and Outdoor,” Secound International Symposium on Ubiquitous Computing Systems (UCS 2004), LNCS 3598, pp. 20–36, Nov. 2004.	5 章

国際会議（デモ）	Yohei Iwasaki and Nobuo Kawaguchi,“LocPointer: Position estimation and pointing method using angle sensor device,” The 9th Annual International Conference on Mobile Computing and Networking (MobiCom2003), Demonstration, 2003.	6章
特許	特願２００２－３４６４６７　２００２年１１月２８日「通信ネットワークにおける通信接続関係確定方法及びシステム」	3章
特許	特願２００３－７１０９７　２００３年3月１４日「位置依存型情報処理システム、サーバ及び装置」	5章
特許	特願２００６－１５６７５　２００６年1月２４日「デバイス連携機構の自動分散化ソフトウェアおよび装置」	7章

参考文献

[1] IEEE. ANSI/IEEE Std 802.11, 1999 Edition. `http://grouper.ieee.org/groups/802/11/`.

[2] Bluetooth SIG. Bluetooth Specification Version 2.0. `https://www.bluetooth.org/spec/`.

[3] Mark Weiser. The computer for the 21st Century. *Scientific American*, Vol. 265, No. 3, pp. 94–104, 1991.

[4] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The Anatomy of Context-Aware Application. In *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking (MOBICOM'99)*, pp. 59–68, 1999.

[5] 澤畠康仁, 杉田馨, 南正輝, 森川博之, 青山友紀. ネットワークサービスシンセサイザにおける機能接続・管理機構の設計と評価. マルチメディア，分散，協調とモバイル（DICOMO 2001）シンポジウム論文集, pp. 103–108, 2001.

[6] Michihiko Minoh and Tak Kamae. Networked Appliances and Their Peer to Peer Architecture AMIDEN. *IEEE Communications Magazine*, Vol. 39, No. 10, pp. 80–84, 2001.

[7] 河口信夫. cogma : ユビキタス情報環境を構築する基盤ソフトウェア. 第 3 回情報科学技術フォーラム (FIT2004), 2004.

[8] Microsoft Corporation. Universal Plug and Play Device Architecture. `http://www.upnp.org/resources/`.

[9] Jim Waldo. The Jini architecture for network-centric computing. *Communications of the ACM*, Vol. 42, No. 7, pp. 76–82, 1999.

[10] Jeremy R. Cooperstock, Sidney S. Fels, William Buxton, and Kenneth C. Smith. Reactive environments. *Communications of the ACM*, Vol. 40, No. 9, pp. 65–73, 1997.

[11] Naohiko Kohtake, Jun Rekimoto, and Yuichiro Anzai. InfoPoint: A device that provides a uniform user interface to allow appliances to work together over a network. *Personal and Ubiquitous Computing*, Vol. 5, No. 4, pp. 264–274, 2001.

[12] Takeshi Hoshino, Yoichi Horii, Yukinobu Maruyama, Atsushi Katayama, Yoshitaka Shibata, and Takushi Yoshimaru. AirReal: Object-Oriented User Interface for Home Network System. In *Workshop on Interactive Systems and Software (WISS 2001)*, pp. 113–118, 2001.

[13] W. G. Griswold, P. Shanahan, S. W. Brown, R. Boyer, M. Ratto, R. B. Shapiro, and T. M. Truong. ActiveCampus - Experiments in Community-Oriented Ubiquitous Computing. In *Technical Report CS2003-0750*. Computer Science and Engineering, UC San Diego, 2003.

[14] Bill N. Schilit, Anthony LaMarca, Gaetano Borriello, William G. Griswold, David McDonald, Edward Lazowska, Anand Balachandran, Jason Hong, and Vaughn Iverson. Challenge: ubiquitous location-aware computing and the "place lab" initiative. In *Proceedings of the 1st ACM international workshop on Wireless mobile applications and services on WLAN hotspots (WMASH'03)*, pp. 29–35, 2003.

[15] 吉田廣志, 伊藤誠悟, 河口信夫. 無線 LAN を用いた測位ポータル locky.jp における位置情報サービス. 情報処理学会グループウェアおよびネットワークサービス研究会 (GNWS 2005), 2005.

[16] Katsumi Takahashi, Nobuyuki Miura, Seiji Yokoji, and Ken ichi Shima. Mobile Info Search: Information Integration for Location-Aware Computing. *Journal of the Information Processing Society of Japan*, Vol. 41, No. 4, pp. 1192–1201, 2000.

[17] H. Tarumi, K. Morishita, M. Nakao, and Y. Kambayashi. SpaceTag: An Overlaid Virtual System and its Application. In *International Conference on Multimedia Computing and Systems (ICMCS'99), vol.1*, pp. 207–212, 1999.

[18] Rashmi Bajaj, Samantha Lalinda Ranaweera, and Dharma P. Agrawal. GPS: Location-Tracking Technology. *IEEE Computer*, Vol. 35, No. 4, pp. 92–94, 2002.

[19] ECHONET (Energy Conservation and Homecare Network) Consortium. ECHONET Specification Ver2.11. `http://www.echonet.gr.jp/`.

[20] HAVi, Inc. The HAVi Specification Version 1.0. `http://www.havi.org/`.

[21] Gregor Kiczales, Erik Hilsdale, Jim Hugunin, Mik Kersten, Jeffrey Palm, and William G. Griswold. An Overview of AspectJ. In *Lecture Notes in Computer Science*, Vol. 2072, 2001.

[22] 一杉裕志, 田中哲. 差分ベースモジュール：クラス独立なモジュール機構. 産業技術総合研究所テクニカルリポート AIST01-J00002-1, December 2001.

[23] Muga Nishizawa and Shigeru Chiba. Remote Pointcut — A Language Construct for Distributed AOP. In *Proceedings of the 3rd International Conference on Aspect-Oriented Software Development (AOSD '04)*, pp. 7–16, March 2004.

[24] Sun Microsystems, Inc. Java Remote Method Invocation (Java RMI). `http://java.sun.com/products/jdk/rmi/`.

[25] Eli Tilevich and Yannis Smaragdakis. J-Orchestra: Automatic Java Application Partitioning. In *European Conference on Object-Oriented Programming (ECOOP)*, June 2002.

[26] 立堀道昭, 千葉滋, 板野肯三. Addistant：アスペクト指向の分散プログラミング支援ツール. 情報処理学会 研究会論文誌 Programming, Vol. 43, No. SIG03, pp. 17–25, March 2002.

[27] 須永豊. 既存 Java プログラムの機能分散化の支援を行うスクリプト言語 Jacross. 第 3 回 SPA サマーワークショップ（SPA-SUMMER 2004）, 2004.

[28] Jon Meyer and Troy Downing. *Java Virtual Machine (Java Series)*. Oreilly & Associates Inc., 1997.

[29] 湯浅太一, 中田登志之, 安村通晃. はじめての並列プログラミング. 共立出版, 1999.

[30] Zigbee Alliance. ZigBee Specification. http://www.zigbee.org/.

[31] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System Architecture Directions for Networked Sensors. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS IX)*, pp. 93–104, 2000.

[32] Nordic Semiconductor. nRF24E1 2.4 GHz Radio Transceiver with Microcontroller. http://www.nvlsi.no/.

[33] Microsoft Corporation. An Overview of the Simple Control Protocol (SCP). http://www.microsoft.com/japan/windows/scp/.

[34] 早川敬介, 塚本昌彦, 寺田努, 義久智樹, 岸野泰恵, 柏谷篤, 坂根裕, 西尾章治郎. ユビキタスコンピューティングのためのルールに基づく入出力制御デバイス. ヒューマンインタフェース学会論文誌, Vol. 5, No. 3, pp. 341–354, August 2003.

[35] Philip Levis and David Culler. Maté: A Tiny Virtual Machine for Sensor Networks. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS X)*, 2002.

[36] 鈴木誠, 鹿島拓也, 猿渡俊介, 森川博之, 青山友紀. 1 チップマイクロコンピュータにおける動的機能モジュール機構の実装と評価. マルチメディア，分散，協調とモバイルシンポジウム（DICOMO 2005）論文集, pp. 593–596, 2005.

[37] David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, and David Culler. The nesC Language: A Holistic Approach to Networked Embedded Systems. In *Proceedings of Programming Language Design and Implementation (PLDI)*, 2003.

[38] Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki. Touch-and-Connect: A connection request framework for ad-hoc networks and the pervasive computing environment. In *First IEEE Annual Conference on Pervasive Computing and Communications (PerCom 2003)*, pp. 20–29, March 2003.

[39] 岩崎陽平, 河口信夫, 稲垣康善. Touch-and-Connect: ユビキタス環境における接続指示フレームワーク. 情報処理学会論文誌, Vol. 45, No. 12, pp. 2642–2654, March 2004.

[40] Digianswer. Bluetooth Software Suite. `http://www.btsws.com/`.

[41] 株式会社バッファロー. AOSS (AirStation One-Touch Secure System). `http://buffalo.jp/`.

[42] Jun Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of the 10th annual ACM symposium on User interface software and technology (UIST'97)*, pp. 31–39, 1997.

[43] Atsushi Sugiura and Yoshiyuki Koseki. A User Interface Using Fingerprint Recognition: Holding Commands and Data Objects on Fingers. In *Proceedings of the 11th annual ACM symposium on User interface software and technology (UIST'98)*, pp. 71–79, 1998.

[44] C Swindells, K M Inkpen, J C Dill, and M Tory. Use that there! Pointing to determine device identity. In *ACM symposium on User interface software and technology (UIST 2002)*, pp. 151–160, 2002.

[45] Frank Stajano and Ross Anderson. The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In *Proceedings of 3rd AT&T Software Symposium*, 1999.

[46] Frank Stajano. The Resurrecting Duckling – what next? In *Proceedings of the 8 th International Workshop on Security Protocols*, 2000.

[47] MicroStrain Inc. . 3DM: solid state 3-axis pitch, roll, & yaw sensor. http://www.microstrain.com/3DM.html.

[48] NEC TOKIN Corporation. 3D Motion Sensor. http://www.nec-tokin.com/.

[49] Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki. Azim: Direction Based Service using Azimuth Based Position Estimation. In *The 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, pp. 700–709, 2004.

[50] Goran Djuknic and Stephen Wilkus. Geolocation and wireless multimedia. In *IEEE International Conference on Multimedia and Expo (ICME 2001)*, pp. 581–584, 2001.

[51] Paramvir Bahl and Venkata N. Padmanabhan. RADAR: An In-Building RF-based User Location and Tracking System. In *IEEE Infocom 2000*, pp. 775–784, 2000.

[52] Andrew M. Ladd, Kostas E. Bekris, Algis Rudys, Lydia E. Kavraki, Dan S. Wallach, and Guillaume Marceau. Robotics-based location sensing using wireless ethernet. In *Proceedings of the eighth annual international conference on Mobile computing and networking (MOBICOM 2002)*, pp. 227–238, 2002.

[53] Seon-Woo Lee and Kenji Mase. Activity and Location Recognition Using Wearable Sensors. *IEEE Pervasive Computing*, Vol. 1, No. 3, pp. 24–32, 2002.

[54] livedoor Co.,Ltd. livedoor Wireless. `http://wireless.livedoor.com/`.

[55] Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki. Design, Implementation and Evaluations of a Direction Based Service System for both Indoor and Outdoor. In *Secound International Symposium on Ubiquitous Computing Systems (UCS 2004), LNCS 3598*, pp. 20–36, November 2004.

[56] Yohei Iwasaki, Nobuo Kawaguchi, and Yasuyoshi Inagaki. Azim: Direction Based Service System for both Indoor and Outdoor. *IEICE Transactions on Communications (Special Issue: Ubiquitous Networks)*, No. 3, pp. 1034–1044, March 2005.

[57] 児玉哲彦, 安村通晃. ふらっと:実空間をブラウジングするためのレーダー型情報提示の提案と試作. 第 13 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS 2005), pp. 153–154, 2005.

[58] 木原民雄. 実写映像の多人数操作による情報ナビゲーションシステム. マルチメディア，分散，協調とモバイルシンポジウム（DICOMO 2002）論文集, pp. 9–12, 2002.

[59] 河口信夫, 稲垣康善. cogma: 動的ネットワーク環境における組み込み機器間の連携用ミドルウェア. 情報処理学会 コンピュータシステム・シンポジウム論文集, pp. 1–8, 2001. `http://www.cogma.org/`.

[60] Yohei Iwasaki and Nobuo Kawaguchi. LocPointer: Position estimation and pointing method using angle sensor device. In *The 9th Annual International*

Conference on Mobile Computing and Networking (MobiCom2003), Demonstration, 2003.

[61] 斉藤功治, 河口信夫, 稲垣康善. RICA:動的にサービス構成の変更が可能な分散ディスプレイシステム. 第 11 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS 2003), 2003.

[62] 斉藤功治, 岩崎陽平, 河口信夫. RICA ＋：Azim を用いた直接的指示が可能な分散ディスプレイサービス. インタラクション 2004, 2004.

[63] 岩崎陽平, 河口信夫. 低レベル実行環境向けの P2P 連携機構の動的構成. 第 4 回 SPA サマーワークショップ（SPA-SUMMER 2005）, August 2005.

[64] 中田育男. コンパイラの構成と最適化. 朝倉書店, 1999.

[65] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System Architecture Directions for Networked Sensors. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*, pp. 93–104, November 2000.

[66] Lewis Girod, Thanos Stathopoulos, Nithya Ramanathan, Jeremy Elson, Deborah Estrin, Eric Osterweil, and Tom Schoellhammer. A System for Simulation, Emulation, and Deployment of Heterogeneous Sensor Networks. In *ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, 2004.

[67] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. SETI@home: an experiment in public-resource computing. *Communications of the ACM*, Vol. 45, No. 11, pp. 56–61, November 2002.

[68] W. Keith Edwards, Mark W. Newman, Jana Sedivy, and Shahram Izadi. Challenge: recombinant computing and the speakeasy approach. In *Proceedings of the eighth annual international conference on Mobile computing and networking (MOBICOM 2002)*, pp. 279–286, 2002.

[69] Kari Kangas and Juha Roning. Using code mobility to create ubiquitous and active augmented reality in mobile computing. In *Proceedings of the 5th an-*

nual ACM/IEEE international conference on Mobile computing and networking (MOBICOM 99), pp. 48–58, 1999.

[70] 佐伯智之, 河口信夫, 稲垣康善. データ構造の変更に対応したソフトウェアの動的更新手法. 第 6 回プログラミングおよび応用のシステムに関するワークショップ (SPA2003), 2003.

[71] 暦本純一, 味八木崇, 河野通宗. ProxNet: 近接センシングによるワイヤレス環境制御のための直接操作技法. 第 11 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS 2003), pp. 103–108, 2003.

[72] Jeanne Ferrante, Karl J. Ottenstein, and Joe D. Warren. The program dependence graph and its use in optimization. *ACM Transactions on Programming Languages and Systems*, Vol. 9, No. 3, pp. 319–349, July 1987.