モバイルコンピューティング環境における
周辺情報提示手法に関する研究

平成18年度

梅澤 猛

目 次

図 目 次

第1章

序論

1.1 はじめに

コンピュータシステムは，技術の飛躍的進歩と時代のニーズに応じて小型化とともに，クライアント・サーバシステムによる分散型ネットワーク化などダイナミックに変化している．一方で移動体を含めた通信技術の発展と普及，インターネットの拡大によって，小型・軽量化された携帯端末を活用したモバイルコンピューティングの本格的実現の基盤が固まりつつある．これらモバイルコンピューティングシステムは，情報処理技術と通信技術が効果的に融合した，今後の高度情報化社会推進のための手段として大きな期待が寄せられている [MCPC].

従来のデスクトップコンピューティングと比較して，モバイルコンピューティング環境下のユーザの最大の特徴は，実世界における物理的な移動を前提とする点にある．このため，モバイルコンピューティング分野においては，地図情報の閲覧や目的地までのナビゲーションなど，ユーザの移動を支援するために実世界情報提供を行うサービスが重視される．実世界空間におけるユーザの移動は，目的地の決定と目的地への移動の 2 つのプロセスにより構成されることから，モバイルコンピューティング環境におけるユーザの移動支援のためには，行動計画と経路計画の 2 つの観点から実世界情報を提供する必要がある．ここで，行動計画とは移動の目的地を決定する過程として，また，経路計画とは出発地点から目的地までの移動経路を決定する過程として，それぞれ定義される．

経路計画に関しては，出発地と目的地を 2 点とする最短経路問題として捉えることができる．実際には，対象となる移動環境が実世界であるため，渋滞などの経路上の混雑状況や，信号や歩道橋，坂道などの経路の移動コストを考慮した最適経路問題へと帰着する．一方で，行動計画に関しては，ユーザの意思決定によるところが大きく，特定の問題として単純化することは困難である．しかし，選考の材料として目的地の候補となり得る店舗やサービス施設に関する情報を提供することで，目的地の決定を支援することができると考える．このときの目的地候補に関連した実世界情報を，本論文では周辺情報と呼ぶ.

経路計画を支援するシステムに関しては研究事例も多く，決定された経路を正しく移動するための付加機能としてナビゲーションサービスを提供するなど，広く実用化されている．しかし，これらのサービスにおいては行動計画の支援に関しては考慮されておらず，目的地は既知もしくはシステムの範囲外においてユーザが決定することを想定している．実世界で行動するユーザに対し，周辺情報提示を行うことで目的地の決定を支援することができれば，行動計画の観点からユーザの移動支援を行うことが可能であり，さらに既存の経路計画支援・ナビゲーションシステムと合わせて，目的地の決定から移動までを包括的に支援することが可能な移動支援システムを実現することができると期待される．

1.2　目的

従来より，モバイルコンピューティング環境で利用される携帯端末は，厳しい資源制約がともなうことが指摘されており [Satyanarayanan 96]，表示能力と入力機能が貧弱であるという特徴がある．近年，関連技術およびデバイス開発の進展により，高機能かつ小型の携帯端末が普及をみせているが，携帯端末における資源制約が実用システムを構築する際の障壁であることに変わりはない [Messer *et al.* 02]．表示系に関する制限に関しては，限られた表示範囲内に登録されている周辺情報をすべて表示すると視認性が損なわれることがある．また，操作系に関する制限に関しては，デスクトップコンピューティング環境と比較して，検索キーワードなどのテキスト入力に向かず，条件を変更するために入力を繰り返す用途には特に不向きである．

また，モバイルコンピューティング環境下においては，ユーザは物理的な移動をともなった行動を取るため，周辺情報を提供する際に提示する店舗やサービス施設についての位置情報を考慮しなくてはならない．提供される周辺情報は，行動計画における目的地決定を支援するためのものなので，ユーザが移動不能なほど遠隔地の情報を提供したり，逆に目的地として有力な近接地の情報が欠落したりすることは避けなくてはならない．一方で，店舗やサービス施設の規模・サービスの質など，周辺情報として建物の地理的な位置とは独立に評価されるべき要素もある．この場合，遠隔地であっても重要度の高い情報や，逆に近接地であっても重要度の低い情報が存在し得る．

さらに，一般に利用されている関連技術・サービスにおいては，地図データ規格の相違や携帯端末間の機能差が大きいことが，位置情報を利用したモバイルコンピューティング向け実用システムを構築する障害となる．地図データにおいては，一般に利用されているだけでも数種類の座標系が存在し，異なる座標系で記述されたデータの統合は容易ではない．このため，異なる座標系を元に登録された複数のリソースからデータを統合して周辺情報提示を行うには，こうした差異が問題とな

る．また，地図データは提供するベンダによってカスタマイズが行われる傾向にあり，異なるベンダ間の地図データには互換性がない．したがって，継続してサービスを提供する実用システムの場合，運用上の理由から使用する地図データを変更することが考えられるため，リソースの互換性が問題となる．

本研究の目的は，実世界で行動するユーザの移動支援を行うために，行動計画を立てる上で必要な周辺情報の提示手法を開発することである．具体的には，前述の問題点を踏まえて次の 3 つの要素を満たす手法の開発を目的とする．

1. 携帯端末特有の表示系・操作系に適応可能なインタフェースとしての周辺情報提示手法
2. ユーザの物理的移動を考慮した，ユーザと周辺情報の位置関係を反映した周辺情報提示手法
3. データリソースや携帯端末の多様性に適応可能な，データの相互利用を可能とするネットワークデータ処理手法

モバイルコンピューティング環境に適した周辺情報提示手法，および多様なリソース間で相互利用可能なネットワークデータ処理手法を開発することで，実世界で行動するユーザの目的地決定を支援する周辺情報提示サービスを実用システムとして構築することができると考える．

1.3　アプローチ

本論文では，モバイルコンピューティング環境において活動するユーザの行動計画支援する周辺情報提示手法を開発する上で，第 1.2 節であげた 3 つの要素を満たすために，次の 3 つのアプローチをとる．

領域可変型情報検索手法 Baum

まず，実世界向けの汎用検索手法として，領域可変型情報検索手法 Baum を提案する．Baum では，検索対象と検索範囲それぞれに特定の幾何形状を仮想的な「領域」として設定し，互いの領域の重なり度合いによって対象の評価値を決定する．これにより，求められた評価値を基準にして，検索対象に順位付けを行うことで情報の絞込みを行うことが可能である．さらに，検索範囲の拡大・縮小に従い検索対象の評価値が変動するため，検索範囲の大きさに応じて網羅的表示と概観的表示の両方が可能となる．

Baum によるモバイルユーザ向け周辺情報提示

次に，Baum による情報検索手法を，モバイルユーザ向けの周辺情報提示に適用する．Baum の適用により，ある対象に設定された重要度と，現在地からその対象までの物理的距離とを総合的に評価することが可能となる．求められた評価値を基準にして提示する周辺情報を選択することで，表示範囲に応じた粒度での周辺情報提示が可能となる．また，検索範囲を拡大・縮小させると，Baum の手法による対象の評価値が変動するため，検索範囲が狭い状況においては網羅的表示を，検索範囲が広い状況においては概観的表示を行うことが可能であり，これら相反する表示方式の両方を情報検索手法 Baum という単一の評価方式によって実現することができる．また，ユーザの移動によって検索の基準点が移動すると，検索範囲も地図平面上を移動するため，これにしたがい対象の評価値も変動するので，移動に適応してユーザに近接した地域の情報を提示することができる．

Baum を適用することにより，ユーザの物理的移動を考慮した，ユーザと周辺情報の位置関係を反映した周辺情報提示が可能となると考えられる．さらに，Baum の手法においては，検索範囲の位置・大きさを変更することで各表示候補の評価値が再計算されるため，ユーザは地図の縮尺変更という簡易な操作のみで適切な情報の絞り込みを行うことができる．また，携帯端末の限られた表示能力を考えたとき，網羅的表示と概観的表示を適宜使い分けることができる Baum の手法は，携帯端末のもつ表示系・操作系の特性に適応した周辺情報提示インタフェースとして有用であると考えられる．

構造可変モバイルエージェント機構

さらに，環境の変化やアプリケーションの要求に応じて動的に構成・機能を変更できるミドルウェアとして，構造可変モバイルエージェント機構 AgentStack を提案する．AgentStack は，システム内のデータフローをストリームとして捉え，データ処理過程を（1）リソース制御,（2）データ変換,（3）データ選別,（4）データ統合,（5）ノード間通信の 5 つの基本機能に分割し，それぞれネットワークデータ管理・運用を行うための階層構造を持つミドルウェアとして構成される．各階層の機能は，モバイルエージェントをベースとするコンポーネントとして実現され，モバイルエージェントのコンピュータ間移動性を通して，データ処理機能を動的に変更・拡張することが可能となる．各階層のコンポーネントには統一されたインタフェースを既定することにより，コンポーネント置換時に他の階層へ与える影響を最小化することが可能である．

AgentStack の適用により，データ処理の対象となる環境の変化やアプリケーションの要求の応じて，動的な構造変化による機能の変更・拡張が可能となる．また，モバイルエージェントのもつ自己複製能力や自律的移動能力を利用して，遠隔ノー

ドに対して動的にコンポーネントを配置し，必要なデータストリームを形成することが可能となる．したがって，構造可変モバイルエージェント機構 AgentStack は，データリソースや携帯端末の多様性に適応し，データの相互利用を可能とするネットワークデータ処理手法として有用であると考えられる．

1.4　論文の構成

以下，2 章では関連研究として，実世界環境におけるモバイルユーザ支援を目的とした情報提供サービス，および，実世界指向のデータ処理を目的としたミドルウェア技術について述べる．3 章では実世界向けの汎用情報検索手法として，領域可変型情報検索手法 Baum を提案する．4 章では，3 章で提案した情報検索手法 Baum を実世界情報サービスに適用し，モバイルコンピューティング環境の特性を考慮した周辺情報提示システムを提案する．5 章では，モバイルエージェントによる階層コンポーネントで構成される構造可変モバイルエージェント機構 AgentStack を提案する．6 章で，本論文で提案した各手法についての議論を行い，今後の課題について述べたのち，最後に 7 章で本論文の結論を述べる．

第2章

関連研究

本章では，本論文で提案する手法に関して関連のある研究について述べ，提案する手法との差異を明確にする．まず，モバイルコンピューティング環境において，実世界で行動するユーザに対して情報提供を行うシステムについて述べる．次に，実世界指向システムで利用されるデータ処理用のミドルウェアに関する研究について述べる．特に，本論文で提案する AgentStack と同じ，モバイルエージェントを利用したミドルウェアについて取り上げる．

2.1 モバイルユーザ支援システム

2.1.1 地図情報システム

地図情報サイトの多くでは，地図情報と連動して地理的な位置情報に基づいた様々な情報を提供している．

Mapion[Mapion] や Its-mo Guide[Its-mo Guide] では，地図表示の付加機能として予め登録されている情報カテゴリを選ぶことで表示範囲内のスポット情報をアイコン表示させることができる．また，Google マップ [Google Map] では，表示した地図範囲内の店舗やサービス施設をキーワード検索することが可能であり，検索結果はアイコンとして地図上へオーバーレイされるとともに，結果一覧がリンクのリストとして表示される．図 2.1 は Google マップによる周辺情報提示の画面出力例である．さらに，MapFanWeb[MapFan Web] では，任意の 2 地点を出発地・目的地として設定することで，ルート検索を行うことが可能なナビゲーション機能を提供している．

これらの地図情報サイトでは，携帯電話を対象としたモバイルデバイス向けのサービスも提供されている．しかし，いずれのサービスも Web 向けのサービスを携帯端末の表示系に合わせて提供しているにとどまり，操作系への配慮は十分でない．また，提供する情報の内容も，カテゴリによる分類があるのみで，検索結果はすべて等価に表示される．

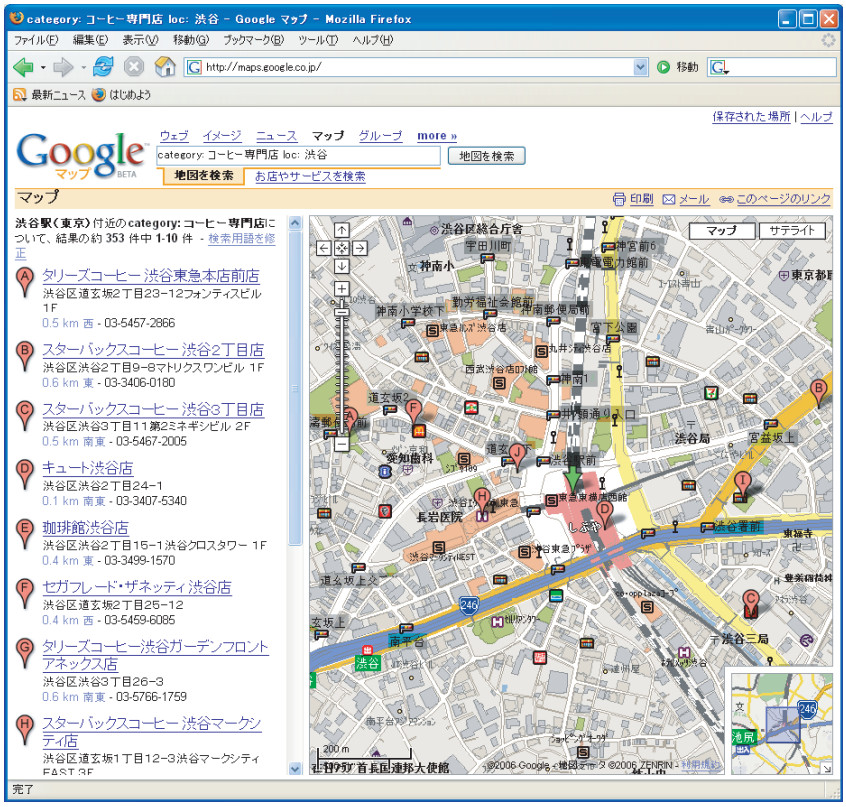

図 2.1: Google マップによる周辺情報提示

2.1.2　パーソナルナビゲーションシステム

モバイルコンピューティング分野における，通信環境や携帯端末，位置情報技術の発展により，モバイルユーザに対する情報提供を目的としたシステム [Hohl *et al.* 99, Cheverst *et al.* 00] の開発が盛んである．

REAL プロジェクトは，屋内外で活動するモバイルユーザを対象として，利用できるリソースを考慮したナビゲーションを目的としている [Baus *et al.* 02]．とくに，位置検出の精度変化に対する適応や，ユーザが使用する携帯端末の変化による出力内容の適応が特徴的である．

GUIDE[Cheverst *et al.* 00] は，Lancaster 大で開発されたコンテキストへの適応を重視したパーソナルナビゲーションシステムである．コンテキストをユーザ依存のものと環境依存のものに分類し，状況に応じた情報提示を行うことを目的としている．ユーザ依存のコンテキストとしては，ユーザの興味，現在位置，休息の欲求に関するものがあげられる．また，環境依存のコンテキストとしては，現在時刻，アトラクションの開始時刻に関するものがあげられる．

European Media Lab による DEEP MAP プロジェクト [Malaka *et al.* 00] は，ウェアラブルデバイスを利用して，自然言語処理を含めたマルチモーダルインタフェースによるパーソナルナビゲーションを目的としている．

いずれのシステムも，目的地までのナビゲーションを主な用途として想定しているため，経路計画を重視している．周辺情報の提示という観点では，ナビゲーション用に地図を表示する際に経路誘導のランドマークとして利用されている程度である．

2.1.3　モバイルユーザ支援システムのまとめ

本論文では，モバイルコンピューティング環境において利用される携帯端末の表示系・操作系両方の特質を考慮し，簡易な操作で視認性に優れた表示を目指す．また，ユーザの動的な移動や検索範囲の変化に応じて検索対象を評価し直すことで，実世界で移動しながらサービスを利用するモバイルコンピューティングに特化したシステムを目指す．

さらに，本論文では，モバイルコンピューティング環境におけるユーザの移動支援において，とりわけ行動計画の支援に着目する．既存システムの多くが，経路計画の支援を重視し，目的地を既知もしくはシステムの範囲外でユーザが決定するのに対して，周辺情報を効果的に提示することにより移動の目的地を決定する過程をシステムの目的としている点に特徴がある．

これらの目的を実現するための要求事項をまとめると，次に示す通りである．

1. 携帯端末の特質への対応

(a) 簡易なオペレーションによる操作

(b) 表示系の制限を考慮した情報提示

2. 検索対象の動的な再評価

(a) ユーザの移動への対応

(b) 検索範囲の変化への対応

3. ユーザの行動計画の支援

第 2.1.1 項では，既存の地図情報システムについて述べたが，現在サービスを提供している地図情報システムはデスクトップでの利用を前提としているため，モバイルコンピューティング環境で主に利用される携帯端末での利用を考えると，特有の表示系・操作系への対応という点で上記要求を満たしていない．既存の地図情報システムの中には，主に携帯電話を対象としてモバイル対応をうたっているサービスが存在するが，画面サイズを携帯電話に合わせて変換することで表示可能にしたものであるため，地図の表示領域制限に対する対応がない点が問題である．また，サービス利用の際の操作に関しても，デスクトップでの操作を踏襲しているため，携帯端末の操作系制限への対応がないという問題がある．

第 2.1.2 項では，モバイルコンピューティング分野の中でも，モバイルユーザに対し携帯端末を利用して情報提供を行う研究例について述べた．これらのシステムでは，携帯端末の使用を前提としているため，表示系や操作系に一定の配慮がされた実装となっている．また，ユーザの移動に従って提示する情報の種類や提示方法を変化させる手法も取り入れらている．しかし，あらかじめ用意されたいくつかの提示方法を切り替える方式であるため，ユーザの位置や検索範囲の変化に対して細かく反応することはできないという問題がある．行動計画支援の観点からみると，既存のシステムにおいては目的地までの経路計画の支援を重視しており，目的地に関してはシステムの取り扱う範囲外においてユーザが決定することが前提となっているため，移動の目的地決定という行動計画の支援については考慮されていないという問題がある．

2.2　実世界指向データ用ミドルウェア

2.2.1　アドホックネットワーク向けミドルウェア

近年，アドホックネットワークを対象としたアルゴリズムやミドルウェアがいくつか研究されており，移動ノードを含むネットワークを非集中制御で管理し，ネットワークトポロジの変化に適応する機能を提供している．

MIT による Directed Diffusion[Intanagonwiwat *et al.* 00] は無線通信で接続されたセンサノード間転送経路の発見機構を実現する．また，ETH による Smart Context-Aware Packets (sCAP)[Michahelles *et al.* 02] はデータ転送用のパケットにそのパケットが通過したノードの識別子を保持することにより，効率的なデータ転送を実現する．文献 [Li *et al.* 00] は，多数の移動ノードから構成されるアドホックネットワークにおいて，ネットワークの一部に切断が発生する場合にも，動的な経路変更を行うことによってメッセージを目的のノードに到達させる手法を提案している．文献 [Ramanathan *et al.* 00] においては，移動ノードを含むネットワークにおいて，動的なトポロジ変化に適応するために，各ノードが送信に用いる電力を変化させるヒューリスティックなアルゴリズムを提案している．

2.2.2　モバイルエージェント

一般に，モバイルエージェントはコンピュータ間の自律移動能力を持つプログラムである [Fuggetta *et al.* 98, Lange *et al.* 98]．モバイルエージェントが移動する際には，プログラムコードとともにプログラムの実行状態も転送される．このため，エージェントは移動先のコンピュータにおいて，移動前の実行状態を復元して処理を継続することができる．

コンピュータ間移動

モバイルエージェントのコンピュータ間移動は，移動の前後で保存される実行状態の内容によって次の 3 つの方式に分類することができる [佐藤 99]．

Code Migration プログラムの実行コードのみを転送する．このため，エージェントの実行状態は移動の前後で保存されず，エージェントは転送先で初期状態から実行を開始する．なんらかの形での実行状態の保存をモバイルエージェントの必須要件と考えると，Code Migration は単なるコード転送に過ぎず，厳密にはモバイルエージェントの範疇には入らないが，実行状態をどこまで保存するかという観点からすると最も単純な転送方式と考えることができる．Code Migration によって実行されるプログラムには，Java Applet，JavaScript，PostScript などがある．

Weak Migration プログラムの実行コードとともに，インスタンス変数などのヒープ領域内の情報を実行状態として転送する．したがって，実行中のメソッド内のローカル変数などのスタック内の情報や，プログラムカウンタは保存の対象とはならない．Java 言語の提供するシリアライズ機能は Weak Migration に基づいているため，Java 言語で実装されたモバイルエージェントシステムの多くは Weak Migration の方式となっている．

Strong Migration プログラムの実行コードとともに，ヒープ領域内の情報，スタック内の情報，プログラムカウンタを実行状態として転送する．このため，エージェントは特別な手続きなしに転送先において移動前に実行していたメソッドの実行を継続することができる．

Strong Migration 方式は，移動の前後でローカル変数やプログラムカウンタの内容まで保存されるため，移動に備えた変数退避・復帰のコードが不要であり，プログラム記述が簡単化されるという利点がある．しかし，スタック領域内の情報を保存するための実行コストが大きく，またエージェントシステム自体の実装が複雑になるという問題点もある．また，ファイルやウインドウなどの計算リソースは実行環境に依存したものであるので，Strong Migration 方式であっても明示的に解放・獲得の処理を行う必要がある．一方で，モバイルエージェントにおいては，エージェントプログラム自身が移動のタイミングを制御できることが多く，スタック内の情報に関しても明示的に保存するコードを用意することによって，擬似的に Strong Migration に近い移動を実現することができる．このため，一般的に実用的なアプリケーションを記述する上では，Weak Migration 方式で十分であるとされている [Lange *et al.* 98, Strasser *et al.* 96]．

一般的な Weak Migration 方式によるモバイルエージェントシステムにおける，エージェントのコンピュータ間移動は次の手順で実行される．

1. 移動対象となるモバイルエージェントの実行状態の一時停止．
2. エージェントの実行状態のシリアライズ．
3. シリアライズされた実行状態とプログラムコードを転送可能な形式に符号化．
4. 任意のデータ通信プロトコルによる符号化データの転送．
5. 受信したデータの復号化によるエージェントの再構成．
6. 移動前の実行状態の復元．
7. 転送先でのエージェントの実行再開．

Weak Migration 方式においては，転送先で移動前に実行していたメソッドの処理が継続されることはない．したがって，転送後にエージェントの実行を再開する機構が必要となる．既存のモバイルエージェントシステムにおいては，コールバックメソッドと呼ばれる移動後に明示的に実行されるメソッドを定義し，この中に転送先でエージェントの実行を再開するためのプログラムコードを記述する方式が採用されている．

代表的なモバイルエージェントシステム

Aglets IBM の Aglets[Lange *et al.* 98] は Java 言語によるモバイルエージェントシステムである．各エージェントは生成後，移動前，移動後，終了前など既定のタイミングで実行されるコールバックメソッドから構成されるオブジェクトとして実現される．Aglets は Java 言語の提供するシリアライズ機能を利用しているため，メソッド内のローカル変数やプログラムカウンタは移動の前後で保存されない．このため，各種リソースの管理を行うコードをコールバックメソッド内に記述し，明示的な解放や獲得を行う必要がある．図 2.2 に Aglets のエージェントプログラム例を示す．エージェントがシステムにロードされると，該当するエージェントの `onCreation()` メソッドが実行される．tt onCreation() メソッドの中身としてエージェント初期化時の処理を記述することによって，適切にエージェントをロードすることができる．ここで，`dispatch("atp://dest.host.url")` はコンピュータ `dest.host.url` にエージェント自身を移動させる API コールであり，そのあと移動前の処理のために `onDispatching()` メソッドが実行されてから実際の移動が行われる．`onDispatching()` メソッドの中身として，移動先で必要なローカル変数の退避や使用していたリソースの解放の処理を記述することによって，該当コンピュータ上からエージェントを適切にアンロードすることができる．移動先コンピュータに到着したエージェントは，`onArrival` メソッドが実行され，メソッド内に記述された処理により，移動前に保存された変数の復元や，新たなリソースの獲得を行う．

Voyager ObjectSpace 社で開発された Voyager[Recursion Software, Inc.] は Java 言語による著名なモバイルエージェントシステムである．最大の特徴は独自の ORB（Object Request Broker）[1]を基礎としている点である．遠隔コンピュータ上でのエージェント生成・移動はこの ORB の機能によって実現される．エージェントの移動は Aglets と同様にインスタンス変数のみを対象とするが，移動先で実行されるメソッドを明示的に指定することができるという特徴を持つ．図 2.3 に Voyager のエージェントプログラム例を示す．システムにロードされたエージェントは，実行を開始するためのメソッドである `start()` メソッドを外部から明示的に呼び出されることにより処理を開始する．ここで，`moveTo("dest.host.url", "callMeOnArrival"`はコンピュータ `dest.host.url` にエージェント自身を移動させる API コールであり，特徴的なのは引数として移動後に実行されるメソッド `callMeOnArrival()` を指定できることである．この機能により，移動前の状態や移動先に応じて，移動後に実行するメソッドを任意に選択するプログラムの記述が可能である．

[1]分散オブジェクト間のメソッド呼び出しを仲介し，各メソッド呼び出しにおいて，実際に呼ばれるオブジェクトやメソッドを選択・実行する機構.

```
public class AgletsAgent extends Aglet
    imlements MobileListener {
    public String name;

    // 生成直後に実行される
    public void onCreation(Object obj) {
        addMobilityListener(this);
        name = new String("AgletsAgent");
        try {
            // dispatch() メソッドにより自身を転送
            dispatch("atp://dest.host.url");
        } catch(Exception e) {
            // エラー処理
        }
    }
    // 移動直前に実行される
    public void onDispatching(MobilityEvent e) {
        // ローカル変数の保存，各種リソースの解放操作
    }
    // 到着直後に実行される
    public void onArrival(MobileEvent e) {
        // ローカル変数の復元，各種リソースの獲得操作
    }
}
```

図 2.2: Aglets のエージェントプログラムコード例

```
public class VoyagerAgent extends Agent {
    public String name;
    // エージェント実行のトリガとして外部から呼び出される
    public void start() {
        name = new String("AgletsAgent");
        // moveTo() メソッドにより自身を転送
        moveTo("dest.host.url", "callMeOnArrival");
    }
    // 到着直後に moveTo() メソッドで指定されたメソッドが呼び出される
    public void callMeOnArrival() {
        // ローカル変数の復元，各種リソースの獲得操作
    }
}
```

図 2.3: Voyager のエージェントプログラムコード例

AgentSpace AgentSpace[佐藤 98] は，Aglets と同様 Java 言語によるモバイルエージェントシステムである．エージェントの移動は Java 言語の提供するシリアライズ機能を用いて実現されるが，圧縮による転送速度の高速化や移動失敗時のリカバリ処理などの機能が提供されている．エージェントの移動に用いる通信プロトコルを多くの方式から選択可能であるなど，システムの柔軟性に特徴があり，モバイルエージェント研究のテストベッドとして広く利用されている．図 2.4 に AgentSpace のエージェントプログラム例を示す．メソッド `create()`,`destroy()`,`leave(URL dst)`,`arrive()`,`clone()` はエージェントの生成直後，終了直前，移動直前，移動直後，複製直後にモバイルエージェントのランタイムシステムにより呼び出されるモバイルエージェント側のコールバックメソッドである．エージェントプログラム作成には，これらのメソッドにそれぞれの実行モード遷移に必要な処理を定義することになる．

2.2.3　モバイルエージェントミドルウェア

モバイルエージェントをセンサネットワークへ応用した事例もいくつか存在するが，それらは通信トラフィックの削減を目的としている．一般にセンサによって測定されるデータ量は大きくなるが，その一方でセンサネットワークは無線通信が前提となったり消費電力の制約を受けたりすることなどから，通信帯域に対する制約が多い．これを解消する方法として，アプリケーション要求に合致する測定データのフィルタリング用プログラムコードをモバイルエージェントにより各ノードへ運

```
public abstract class Agent
    imlements Serializable {
        // 生成後に実行される
     void create();
        // 消滅時に実行される
     void destory();
        // 移動時に実行される
     void leave(URL dst);
        // 到着時に実行される
     void arrive();
        // 複製後に実行される
     void clone();
     ....
        // 自身を転送
     void go(URL dst) throws
       NoSuchLocationException ... { ...}
        // 自身を複製
     void duplicate(URL dst) throws
       NoSuchLocationException ... { ...}
        // ライフタイムを設定
     void setTimeout(int time);
     ....
}
```

図 2.4: AgentSpace のエージェントプログラムコード例

び，そのコードを実行することにより必要なデータだけをノード間で交換・選別する方法が知られており，たとえば Qi らのセンサネットワーク [Qi *et al.* 01] ではモバイルエージェントによる通信トラフィックの低減および測定データの合成手法を提案している．

2.2.4　実世界指向データ用ミドルウェアのまとめ

本論文では，個々の通信制御を前提とせず，データリソースの制御や取得データの選別，ノード間通信などを動的に変更する枠組みを提供するミドルウェアを提案する．

また，提案するミドルウェアは既存手法と同様にモバイルエージェントを通してデータリソースによるネットワークのスケーラビリティを向上させると同時に，モバイルエージェントを移動・置換可能なコンポーネントとして導入することにより，ミドルウェアの汎用性や適応性，独立性を高めることを目的とする．

これらの目的を実現するための要求事項をまとめると，次に示す通りである．

1. 適応性
   (a) データリソースの変化
   (b) アプリケーション要求
2. 汎用性
   (a) 多様な用途への対応
3. スケーラビリティ
   (a) ノード数の増加
   (b) 通信トラフィックの増加
4. 独立性
   (a) データリソースの差異
   (b) オペレーティングシステム
   (c) ハードウェア

第 2.2.1 項では，アドホックネットワークを対象としたミドルウェアについて述べたが，既存研究の多くはセンサノード間の経路制御やネットワーク管理に関するものである．これらはネットワーク構成の変化を通信方法に反映することができるが，測定対象やアプリケーション要求の変化に対する適応性を提供するものではない点に問題がある．

第 2.2.3 項では，モバイルエージェントを利用したミドルウェアについて述べたが，既存研究はモバイルエージェントによる通信トラフィックの低減を通してスケーラビリティの向上を図っているものの，多様な用途に対応する汎用性や，アプリケーションの要求変化への適応性に関しては想定していない点が問題となる．

2.3　まとめ

本章では，本論文で提案する手法に関連した研究について述べた．まず，モバイルコンピューティング環境においてユーザへ情報提供を行うシステムについて説明した．地図情報システムの分野では，既に多くの地図情報サービスが提供されており，ルート検索機能や表示範囲内の店舗やサービス施設の検索機能など，デスクトップコンピューティング環境における位置情報利用サービスとして広く利用されている．モバイルコンピューティング環境においても，携帯電話向けのサービスを提供するなど，位置情報利用サービスの展開は促進される傾向にあるが，ほぼデスクトップ向けのサービスを携帯電話で表示可能にしただけのものであり，携帯端末特有の表示系・操作系の特質に適応したものではない．パーソナルナビゲーションに関する研究では，ウェアラブルデバイスの利用により携帯端末の特殊性を考慮する取り組みがみられる．デバイス毎の表示特性に合わせた情報表示や，コンテキストに適応した情報表示など，モバイルコンピューティング環境に固有の特徴を捉えた情報提示手法が提案されている．しかし，パーソナルナビゲーションを目的としているため，ユーザの経路誘導は重視されているものの，目的地の決定過程は支援対象として扱われていない．

次に，実世界指向データ処理用を目的としたミドルウェアについて説明した．実世界指向のミドルウェアとしては，移動ノードを含む無線ネットワークを扱ったアドホックネットワーク分野での研究が盛んであるが，多くは移動ノードの通信切断に起因するトポロジ変化に適応するノード間の通信経路の制御やネットワーク管理に関するものである．本論文で対象とするモバイルコンピューティング環境における周辺情報提示サービスには，データリソースの相違やアプリケーションプログラムの要求変化に適応して，データの処理や通信方式自体を動的に変更する機能が必要となる．モバイルエージェントは，コンピュータ間を自律的に移動する能力を持ち，エージェントの置換による動的な機能変更によりこうした要求を満たすことのできるプログラミング手法である．センサネットワークの分野においては，モバイルエージェントの移動機能を利用したミドルウェアが存在するが，通信トラフィックの削減や計算負荷の分散を目的としており，データの処理内容そのものを動的に変更することは想定していない．

本論文では，実世界環境で行動するユーザに対して周辺情報を提示することにより，目的地の決定過程を重視した行動計画支援に注目する．モバイルコンピュー

ティング環境において利用される携帯端末の表示系・操作系両方の特質を考慮し，簡易な操作と優れた視認性を目指す．ユーザの位置の動的変化や検索範囲の変化に応じて検索対象を評価する情報検索手法を提案し，実世界で移動しながらサービスを利用するモバイルコンピューティングに特化したシステムを目指す．提案する手法は，既存のモバイルユーザ支援システムで実現されている経路計画支援機能と合わせることで，目的地の決定から経路誘導までを含めた総合的なモバイルユーザ支援システムが実現できると考える．また，各ノードにおいてデータ処理を行うソフトウェアをプラグインコンポーネントとして構成することで，構造を動的に変更可能なミドルウェアを実現する．コンポーネントを階層的に構成し，各層のコンポーネントをモバイルエージェントとして実現することで，エージェントの移動機能を利用したコンポーネントの置換によりデータ処理機能の動的変更を可能とする．

第3章

領域可変型情報検索手法Baum

本章では，実世界向けの汎用検索手法として，領域可変型情報検索手法 Baum を提案する．Baum は，検索対象と検索範囲それぞれに特定の幾何形状を仮想的な「領域」として設定し，互いの領域の重なり度合いによって対象の評価値を決定する．求められた評価値を基準にして，検索対象に順位付けを行うことで情報の絞込みを行うことが可能である．さらに，検索範囲の拡大・縮小に従い検索対象の評価値が変動するため，検索範囲の大きさに応じて網羅的表示と概観的表示の両方が可能となる．

3.1 Baumの検索手法

本論文で対象とする一般的な情報検索のイメージを図3.1に示す．表示候補となる情報は，あらかじめ位置情報とともに登録されているとする．O_i $(i = 1, 2, 3, \cdots, n)$ は，これらの情報を仮想空間内にマッピングした様子を示している．P で表される地点は検索の基準点である．

一般的な情報検索の場合，検索の基準点 P を含む検索範囲 R_P を設定した場合の表示候補は，次の （3.1）式で表される．

$$\{O_i \subseteq R_P \mid i = 1, 2, 3, \cdots, n\} \tag{3.1}$$

このとき，検索範囲内に存在する情報を，絞り込みなしにすべて表示している状態であるといえる．

Baum は，検索範囲と検索対象の双方について領域を設定し，互いの領域の重なり度合いを対象の評価値とする検索手法である．Baum の検索イメージを図3.2に示す．P で表される点は検索の基準点であり，O_i で表される点は検索対象である．

P，O_i は同じ仮想空間内にマッピングされている．このときの仮想空間は，実世界指向システムであれば3次元空間が考えられ，対話語彙検索であれば単語の意味情報にもとづく意味空間が考えられる．また，ネットワーク管理に関係したシステムであれば，ホップ数を元にしたネットワーク空間となり得る．

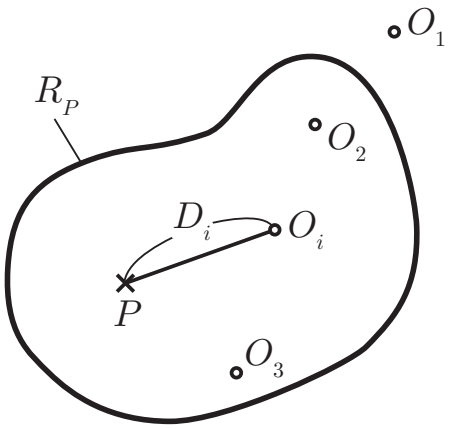

図 3.1: 一般的な情報検索イメージ

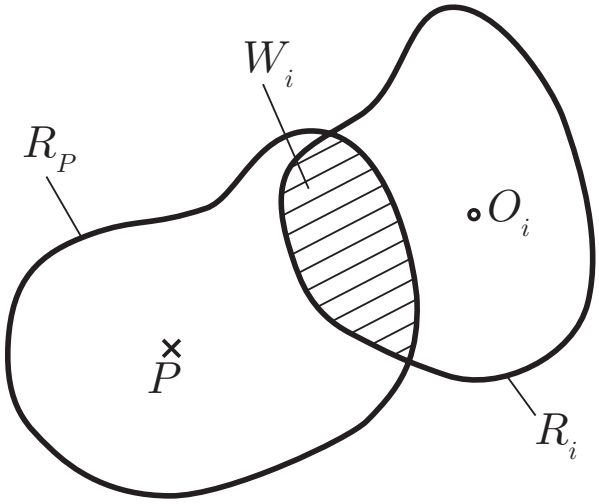

図 3.2: Baum による検索概念図

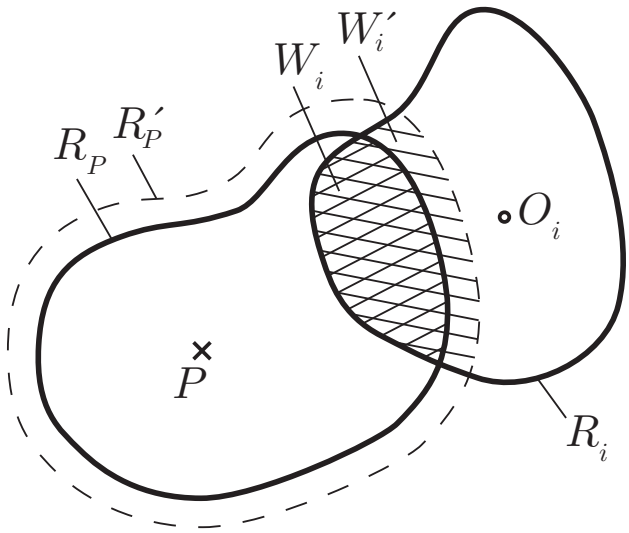

図 3.3: 検索領域の拡大

この意味空間内で，検索の対象とする範囲を検索領域 R_P で，O_i のもつ重要度を R_i で定義する．

R_P は，仮想空間内で検索の対象とする範囲を示す領域であり，R_P が広いほど検索の条件が緩く，R_P が狭いほど検索の条件が厳しいことを示している．実世界指向システムであれば検索の対象となる地区・地域に相当し，対話語彙検索であればどの程度単語の類似度を許容するかという指標となる．また，ネットワーク管理に関係したシステムであれば，基準となる地点から何ホップの機器までを対象にするかという指定を表すことになる．

一方，R_i は検索対象の持つ重要度を基準に重み付けを行うために設定する領域であり，実世界指向システムであれば，ターミナル駅に他の駅よりも領域を広く設定したり，対話語彙検索であれば,「寒い」という単語に「涼しい」よりも領域を広く設定するということが考えられる．また，ネットワーク管理に関係したシステムであれば，重要なルータには他よりも広い領域を設定する場合が考えられる．

Baum では，これらの検索領域 R_P と検索対象の重要度を示す領域 R_i の重なり度合いに応じて，各対象の評価値を決定する．図 3.2 においては，W_i で表される領域が検索対象 O_i の評価値を表している．このとき，O_i の評価値 W_i は（3.2）式によって表すことができる．

$$W_i = |R_P \cap R_i| \tag{3.2}$$

Baum においては，検索対象 O_i の評価値は領域 R_P，R_i の変化にともない，動的に変化する．今，R_i を一定に保ったままで，検索領域 R_P を拡大させたときの様子を図 3.3 に示す．検索領域 R_P を拡大して R_P' とすると，検索領域と O_i の重要度

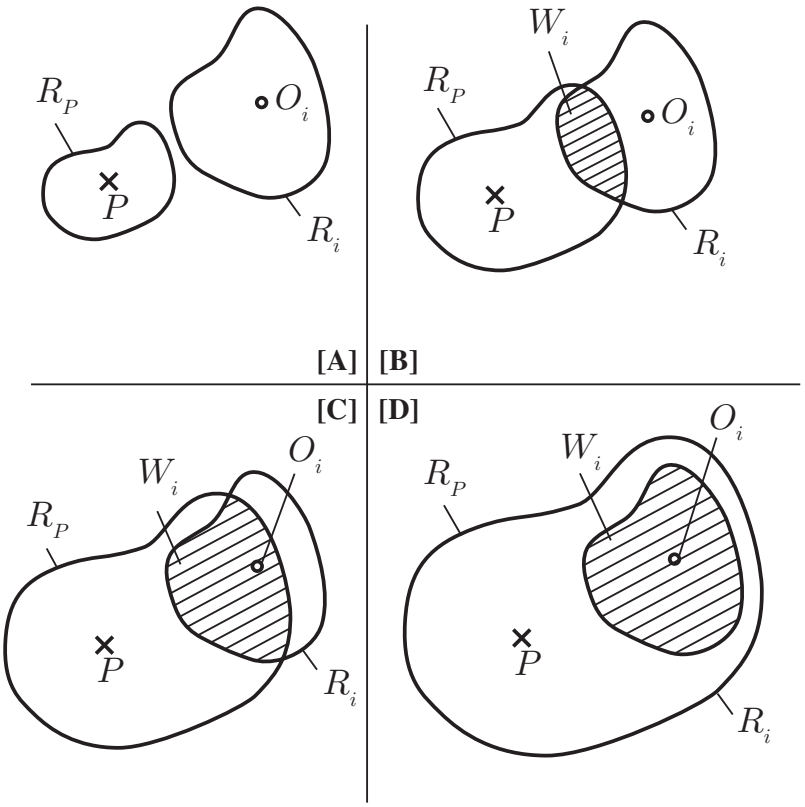

図 3.4: 検索領域の変化と評価値

を示す領域との重なる領域も W_i から W_i' へと変化する．したがって，このときの O_i の評価値は $W_i' = \left|R_P' \cap R_i\right|$（$W_i' > W_i$）となる．

3.2　検索領域の拡大・縮小

Baum では，検索範囲と検索対象それぞれに設定された仮想的な領域の重なり度合いによって評価値を決めるため，領域の変化に応じて，各検索対象の評価値も変動する．図 3.4 は，検索領域 R_P を拡大していったときに候補 O_i の評価値が変化する様子を示している．このとき，検索の基準点 P および検索対象 O_i の仮想空間内の位置は固定とした．図 3.4 において［A］から［D］へ順に検索領域 R_P を拡大していくにしたがって，検索対象 O_i の評価値が増加していく．

[**A**] 検索範囲が十分狭く設定されているため，R_P は検索対象 O_i の重要度を示す領域 R_i まで達していない．このため，O_i の評価値は $W_i = |R_P \cap R_i|$ $(R_P \cap R_i = \phi)$ となる．したがって，このとき検索対象 O_i は検索結果には含まれないことになる．

[B] 検索領域を拡大して R_P が R_i と重なる状態になると，O_i の評価値は $W_i = |R_P \cap R_i|$ となる．この間，検索領域の拡大に従って，O_i の評価値 W_i は単調に増加する．特にこのとき，点 O_i が検索領域 R_P には含まれておらず，(3.1) 式を満たしていない．これは，検索対象 O_i の重要度を示す領域 R_i を定義したことにより，点 O_i が R_P の領域外であっても表示候補になり得ることを示している．

[C] さらに検索領域を拡大していくと，検索対象 O_i の重要度を表す領域 R_i が検索領域 R_P に含まれる状態になる．このときも，［B］の状態と同じく，検索対象 O_i の評価値は $W_i = |R_P \cap R_i|$ で表すことができる．また，この間の W_i の変化も［B］の状態と同じく単調増加を示す．

[D] 検索範囲が十分広く設定されているとき，$R_P \supseteq R_i$ となる．このとき，検索対象 O_i の評価値は $W_i = |R_P \cap R_i| = |R_i|$ で表すことができる．また，これ以上検索領域を拡大しても O_i の評価値 W_i は変化しなくなる．これは，検索対象 O_i の重要度を示す領域 R_i の大きさと比較して，検索領域 R_P が十分大きくなった状態を示している．このため検索の基準点 P と検索対象 O_i の論理距離に関わらず，O_i に設定された重要度 R_i の大きさに従って評価値 W_i が決定される．

Baum では，検索領域 R_P を設定した場合の表示候補を次の（3.3）式で定義することができる．

$$\{O_i \mid R_P \cap R_i \neq \phi, i = 1, 2, 3, \cdots, n\} \tag{3.3}$$

ここで，(3.1）式で示した一般的な情報検索の場合の表示候補は，(3.3）式において O_i に設定された重要度 R_i の大きさが十分に小さい場合に相当すると考えることができる．

3.3　検索基準点の移動

図 3.5 は，論理空間内において検索の基準点 P を移動させたときに検索対象 O_i の評価値 W_i が変化する様子を示している．このとき検索領域 R_P および検索対象 O_i の重要度を示す領域 R_i は固定とした．図 3.5 で［A］から［D］へ順に基準点 P が移動するにしたがって，検索対象 O_i の評価値が増減する．

[A] 検索基準点 P が検索対象 O_i から十分遠く，検索領域 R_P が O_i の重要度を表す領域 R_i まで達していない状態である．このとき，検索対象 O_i の評価値は $W_i = |R_P \cap R_i|$ $(R_P \cap R_i = \phi)$ で表すことができる．したがって，このとき検索対象 O_i は検索結果には含まれないことになる．

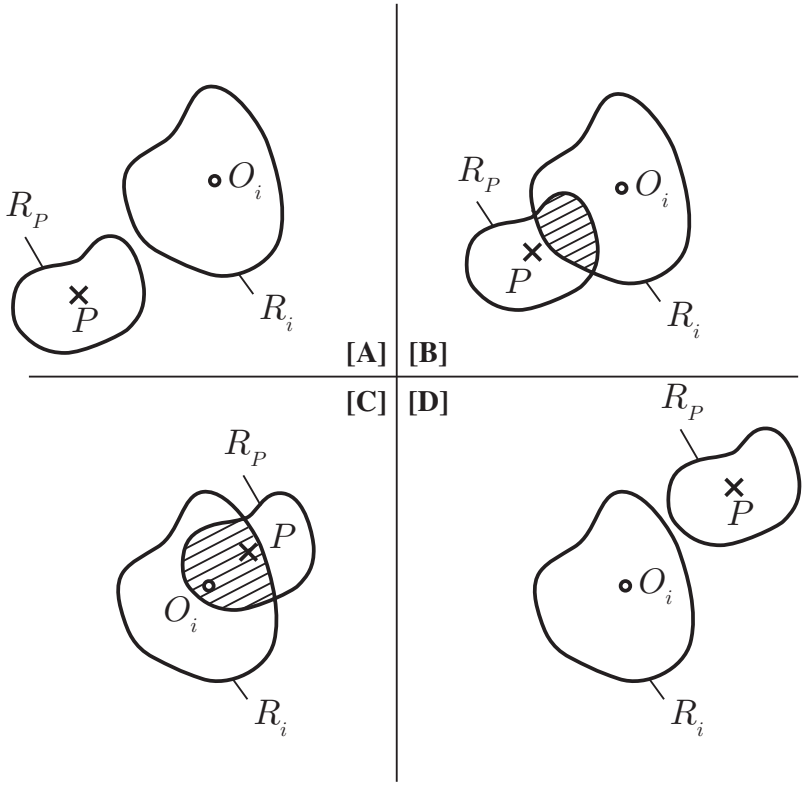

図 3.5: 検索基準点の移動と評価値

[B] 検索基準点を検索対象 O_i に近づけていって，検索領域 R_P が R_i と重なる状態となると，検索対象 O_i の評価値は $W_i = |R_P \cap R_i|$ となる．この間，検索領域の移動に従って，O_i の評価値 W_i は増加する．特にこのとき，点 O_i が検索領域 R_P には含まれておらず,（3.1）式を満たしていない．これは，検索対象 O_i の重要度を示す領域 R_i を定義したことにより，点 O_i が R_P の領域外であっても表示候補になり得ることを示している．

[C] さらに検索基準点を移動させると，P が検索対象 O_i から遠ざかり始める．このときも,［B］の状態と同じく，検索対象 O_i の評価値は $W_i = |R_P \cap R_i|$ で表すことができるが，この間 W_i は検索基準点の移動に従って減少する..

[D] 検索基準点 P が再び検索対象 O_i から十分遠くまで離れると，検索領域 R_P と O_i の重要度を表す領域 R_i との重なり合う部分がなくなる．このとき，O_i の評価値は再び $W_i = |R_P \cap R_i|$ $(R_P \cap R_i = \phi)$ となる．

3.4　まとめ

本章では，実世界向けの汎用検索手法として，領域可変型情報検索手法 Baum を提案した．Baum は，検索対象と検索範囲それぞれに特定の幾何形状を仮想的な「領域」として設定し，互いの領域の重なり度合いによって対象の評価値を決定する検索手法である．Baum の検索手法について説明し，検索領域の拡大・縮小および，検索基準点の移動によって検索対象の評価値に与える影響について示した．Baum の手法により求められた評価値を基準にして，検索対象に順位付けを行うことで情報の絞込みを行うことが可能である．さらに，検索範囲の拡大・縮小に従い検索対象の評価値が変動するため，検索範囲の大きさに応じて網羅的表示と概観的表示の両方が可能となる．

第4章

Baumによるモバイルユーザ向け周辺情報提示

本章では第3章で提案した領域可変型情報検索手法Baumを，モバイルユーザ向けの周辺情報提示へ適用する．Baumの適用により，位置情報をともなった周辺情報を，あらかじめその対象に設定された重要度と，基準点からの物理的距離の双方を考慮して評価することが可能となる．

4.1 周辺情報提示システム

4.1.1 想定環境

本論文で対象とする周辺情報提示サービスの一般的な情報検索イメージを図4.1に示す．

表示候補となる周辺情報は，あらかじめ位置情報とともに登録されているとする．O_i $(i = 1, 2, 3, \cdots, n)$ は，これらの周辺情報を位置情報に基づいて地図平面上にマッピングした様子を示している．地図上にオーバレイ表示された O_i はそのまま店舗・施設の位置を示している．P で表される地点は検索の基準点であり，P を含む R_P は検索の対象となる範囲を示す領域である．

検索基準点 P を設定する方法によって，次の2通りの利用法が考えられる．

リアルタイム表示 ユーザはGPSによる位置取得サービスを利用することで，検索基準点を常に自己の現在位置として設定することが可能である．このとき，表示される情報はユーザの現在位置周辺の情報となり，ユーザの移動に従って表示内容が更新される．

ウォークスルー表示 任意の位置座標を検索基準点として指定することで，ユーザの現在位置とは関係なく，遠隔地の情報を参照することが可能である．この

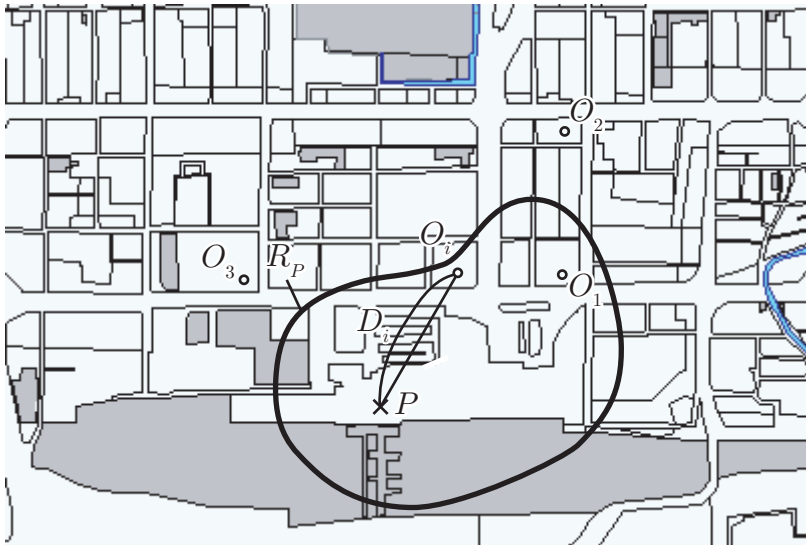

図 4.1: 地図平面上の周辺情報提示イメージ

とき，表示される情報は任意の地点を起点とする周辺情報となり，ユーザは訪れる予定のある場所周辺の情報を参照することができる．

一般的な地図情報サービスにおいては，R_P は地図表示範囲と一致する矩形で表されることが多く，このとき表示される情報は表示範囲内のすべての登録情報となる．

4.1.2　アプリケーション要求

モバイルユーザを対象とする実世界指向システムにおいては，その行動特性や用いる携帯端末の機能制限の観点から，アプリケーションには次に示すような特徴が要求される．

1. 検索範囲に応じた粒度の結果表示
   検索範囲 R_P を設定することで，多くの検索対象の中から，表示候補を基準点 P 周辺のものに絞り込むことはできるが，$O_i \subseteq R_P$ となる O_i が多数存在する場合にはさらに表示候補の絞り込みを行う必要がある．目的地への移動という，モバイルユーザの特徴的行動を考慮すると，表示候補の絞り込みの際に候補を評価するのにあたり，検索の基準点 P からの O_i までの物理的距離 D_i は重要な要因である．一方で登録されている周辺情報はすべてが同じ有用性を持っているわけではないので，各情報の重要度も評価する必要がある．検索範囲に応じて表示する情報を適切に絞り込むためには，これら 2 つ

の要因，対象までの物理的な距離と，対象自体の情報の重要度を適切に評価する手法が求めれる．

2. オペレーションの簡便さ
   モバイルユーザが利用する携帯端末は入力機能が充分ではなく，更にモバイルコンピューティング環境は一般に複雑な入力を行うのには不向きである．したがって，表示候補の絞り込みは簡易なオペレーションで利用できる必要がある．

本論文では，第 3 章で提案した領域可変型情報検索手法 Baum をモバイルユーザ向け周辺情報提示に適用することによってこれらの要求に応える．Baum の適用により，対象までの物理的な距離と対象自体の重要度を総合的に評価することができ，検索範囲に応じた粒度での結果表示が可能となると期待できる．また，検索領域の大きさを変化させることで適切な表示が可能であるため，簡易なオペレーションによってサービスの利用が可能であるといえる．

4.1.3　システムアーキテクチャ

ここでは，第 4.1.2 項であげたアプリケーション要求を満たすために，システムに必要な機能について整理するとともに，それによって構成されるシステムアーキテクチャについて述べる．

- クライアントアプリケーション
  モバイルユーザ向け周辺情報提示システムは，携帯端末での利用を前提とするため，フロントエンドとなるアプリケーションは携帯端末上で動作する必要がある．また，Baum による周辺情報提示システムは地図への周辺情報のオーバーレイという形式をとるため，対象となる地域の地図データと検索対象となる登録情報を利用できなくてはならない．既に述べたように，携帯端末は厳しい資源制約があるため，これらのデータはサーバにおいて管理し，フロントエンドアプリケーションはサーバとの通信によって周辺情報を提示するクライアントアプリケーションとして構成する必要がある．第 4.1.2 項で要求事項に挙げた簡便なオペレーションによる操作を実現するためには，クライアント側で取り扱うデータを最小限に抑える必要がある．このため，Baum 方式による検索対処の評価処理や表示範囲の地図データの切り出し処理はサーバ上で行うこととし，クライアントアプリケーションはシンクライアントとして動作することとする．

- 地図データベース
  第 4.1.1 項で述べたように，本論文で想定する周辺情報提示システムは，地

図平面上への周辺情報のオーバーレイという形式をとる．このため，システムの対象となる地域を網羅する地図情報データを利用できることが必要であり，クライアントアプリケーションの要求に応じて，必要な範囲の地図を適切な縮尺で提供する必要がある．一方で，このような地図情報データは市販のものを含め多くのものが存在する．本研究では，特定の地図情報データに依存することなくシステムを構成するため，地図情報データは独立したデータベースとして構成する．

- 周辺情報データベース
  第 4.1.1 項で述べたように，Baum による周辺情報提示システムにおいては，検索基準点と検索対象との物理的距離を元に，検索対象の評価値を動的に再計算する必要がある．このため，周辺情報として登録される検索対象のデータには重み付けの数値だけではなく，地図平面上の座標値が必要である．また，ユーザの位置移動や検索範囲の変化にともなって，評価計算の対象となる検索対象も変化するため，クライアントアプリケーションからの要求に従って必要な範囲の検索対象を抽出する機能が必要である．

- 仲介エージェント
  上記のように，クライアントアプリケーションをシンクライアントとして構成した場合，必要なデータ処理はサーバ側で行う必要がある．Baum による周辺情報提示システムにおいては，周辺情報データベースから登録情報を得て検索対象の評価値を計算し，地図情報データベースから，クライアントアプリケーションでの表示に適した地図データを取得することが必要となる．また，クライアントアプリケーションとの通信，複数のクライアントアプリケーションに対するセッション管理も必要となる．これらの処理を行うモジュールを仲介エージェントとして構成する．この仲介エージェントは第 5 章で提案する構造可変モバイルエージェント機構 AgentStack を利用して構成することにより，汎用性，適応性などの機能拡張が期待できる．

4.1.4　Baum による提示候補の評価

第 3 章で提案した領域可変型情報検索手法 Baum は実世界向けの汎用検索手法である．Baum をモバイルコンピューティング環境における周辺情報提示に適用するには，検索の基準点および検索対象を地図平面上にマッピングすればよい．このときの検索イメージを図 4.2 に示す．

表示候補となる周辺情報は，あらかじめ位置情報とともに登録されているとする．O_i $(i = 1, 2, 3, \cdots, n)$ は，これらの周辺情報を地図平面上にマッピングした様子を示している．P で表される地点は検索の基準点である．また，R_P は検索の対象となる範囲を示す領域であり，R_i は登録されている周辺情報 O_i のもつ重要度を

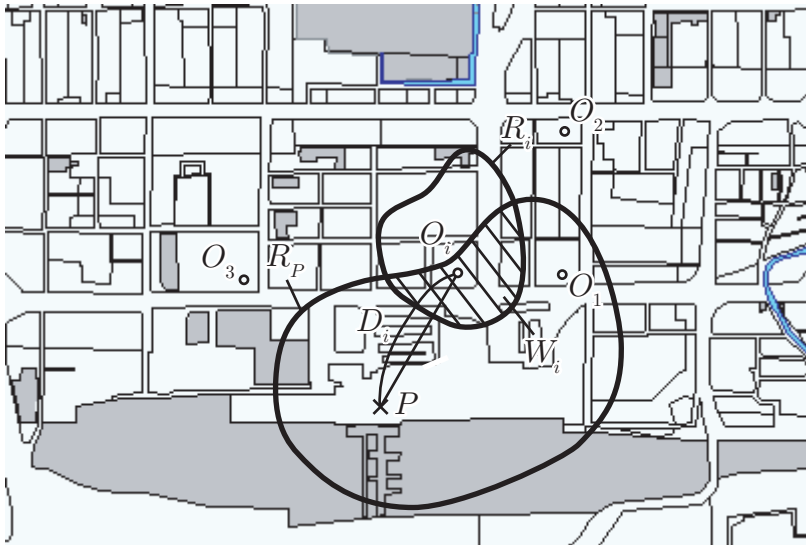

図 4.2: 地図平面への Baum 適用イメージ

示す領域である．Baum では，検索領域 R_P と検索対象の重要度を示す領域 R_i の重なり度合いに応じて，対象の評価値を決定するため，(3.2）式によって表わされる領域 W_i が検索対象 O_i の評価値となる．

4.1.5　検索領域モデル

領域可変型情報検索手法 Baum においては，検索領域の形状が重要な要素のひとつである．検索対象の評価値は，検索対象と検索範囲それぞれに設定された領域の重なり度合いによって決定されるため，検索領域の形状による影響が大きい．したがって，検索領域モデルをどう設定するかによって，検索基準点を移動させた場合や，検索領域を拡大・縮小させた場合における検索対象の評価値の変化の仕方が大きく異なる．

単純円形

まず，簡単化のため検索領域を単純な円形とするモデルを利用することにした．このときの，地図上にマッピングした検索領域のイメージを図 4.3 に示す．

検索領域は，検索の基準点 P を中心とする単純な円形として定義され，検索領域を拡大・縮小したときには円の半径を変化させることで検索領域を得る．

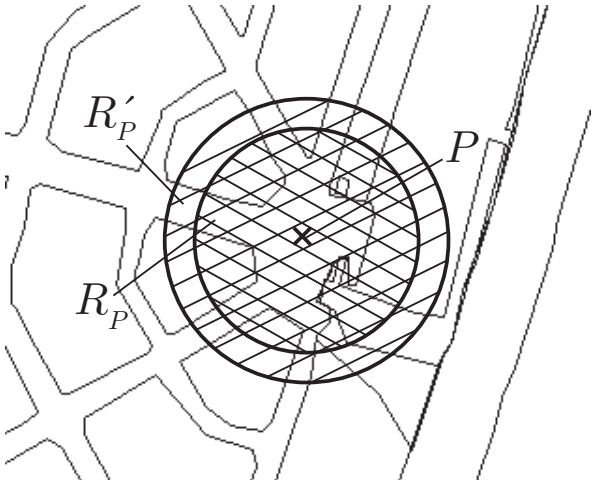

図 4.3: 単純な円形による検索領域

地形情報の考慮

検索領域を単純円形とした場合，表示や対象の評価値計算の上で簡単化を図ることができるが，表示結果に地理的な条件が反映されないという問題がある．地図平面上で検索基準点から近い対象であっても，間に建物が存在したり，広い道路や川により隔てられている場合には，地理的にみると近いとはいえない場合がある．実世界を対象とした情報提示システムの場合，こうした地理情報の考慮はユーザの利便性の観点から非常に重要であるといえる．

そこで，検索領域の形状として地形情報を考慮した検索領域モデルを定義する．検索領域は検索の基準点 P を中心とする円形を基本として，地形情報を反映した形状とする．地形情報を考慮した拡張を行った検索領域のイメージを図 4.4 に示す．

検索基準点 P を中心とする円を描き，そこから街区や建物など障害物となる地形を取り除き，検索領域とする．検索領域を拡大・縮小するときには，まず基準となる円を拡大・縮小し，同様に地形情報を反映して検索領域を得る．

その他の領域形状

領域モデルの候補としては，ほかにもいくつか参考になる研究がみられる．移動体向けの情報配信サービスを行う研究 [武井 他 04] においては，移動方向に領域を拡大した楕円形の領域を検索範囲として用いている．また，バリアフリーな移動支援を目指す [矢入 04] においては，領域としてとらえるのではなく，検索対象も含めた多数のノードをつなぐネットワークによって移動経路を構成し，厳密に経路ごとのコストを算出している．

しかし，文献 [武井 他 04] ではユーザが約 30km/h の速度で移動している仮定で

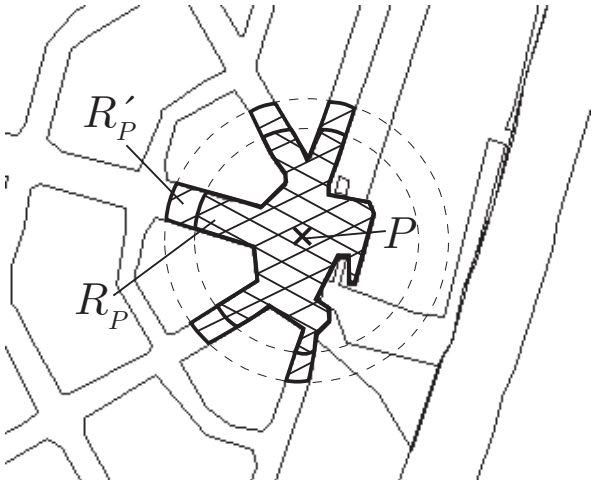

図 4.4: 地形情報を考慮した検索領域形状

あり，主に車載端末を対象としたシステムである．本論文における実装では主に歩行者をユーザとして想定しており，商店街を含めた街路を実験環境としている．このため，検索領域を移動方向に変形させる程高速で移動しながらのシステム利用は考慮する必要がないと考える．また，文献 [矢入 04] では車椅子や高齢者など，階段や悪路の通行に困難をともなう場合を想定している．本論文で扱うシステムでは対象外の用途であり，ノードによるネットワークですべての経路情報を管理する手法には，データを作成するための手間がかかりすぎるという問題があり，提案するシステムには適していない．

4.2　実装

4.2.1　システム構成

モバイルユーザ向け周辺情報提示システムに，領域可変型情報検索手法 Baum を適用したシステムの構成を図 4.5 に示す．

システムは表示および操作を行う携帯端末によるクライアント（Client）と地図データベース（Map DB）および周辺情報が登録されたデータベース（Cite Info. DB）を管理するサーバ（Server）から構成される．サーバ上にはクライアントからのリクエストを処理する仲介エージェント（Baum Agent）が配置される．

クライアントとサーバは，次の手順による処理を繰り返すことで周辺情報の表示を行う．

1. クライアント（Client）は検索基準点の位置情報（Position Info.）と検索領域の情報（Search Region）をサーバ（Server）に送信する．

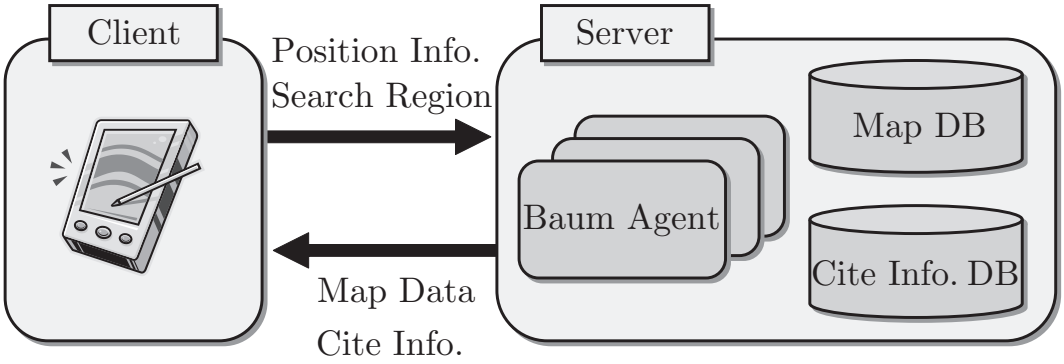

図 4.5: システム構成

2. サーバ側では，仲介エージェント（Baum Agent）がクライアントの要求に応じて必要なデータを収集する．

   (a) 地図データベース（Map DB）から適切な範囲のデータを取り出し，クライアントの要求に適した縮尺の表示となるようにデータ（Map Data）を生成する．

   (b) 周辺情報が登録されたデータベース（Cite Info. DB）から得られた表示候補を，Baum による手法で評価する．

   (c) 評価結果を元に，表示候補の評価値と位置情報から，クライアントに提供する情報（Cite Info.）を作成する．

   (d) 生成された地図表示用データ（Map Data）と周辺情報データ（Cite Info.）をクライアントに通知する．

3. データを受け取ったクライアントは，地図表示用データを元に地図の表示を行い，周辺情報データをオーバーレイ表示することでユーザに提供する．

また，クライアントとサーバの実装環境は次の通りである．

- サーバ
  サーバ側プログラムの実装は，Java プログラミング言語 [Arnold *et al.* 00]（Java 2 Platform, Standard Edition, v1.42）を用いて行った．地図情報には国土地理院 [国土地理院] 発行の数値地図 2500 を利用した．周辺情報としては，あらかじめ大学周辺の飲食店の情報を用意し，各店舗に 10 段階評価による重要度を付加して登録した．

- クライアント
  クライアントとなる携帯端末には，PalmOS5.0 で動作する PDA を利用した．

図 4.6: PDA への実装

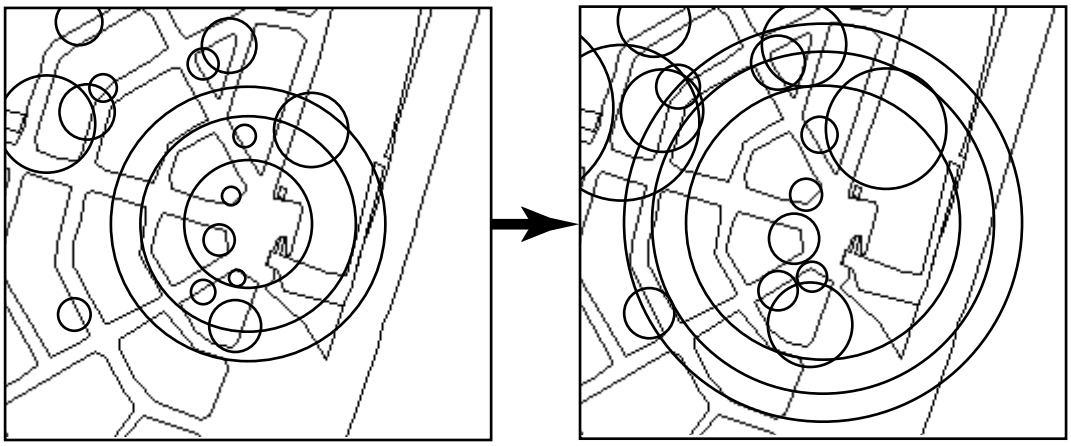

図 4.7: Baum による周辺情報の表示

プログラム実装には，Java VM として Mobile Information Device Profile（MIDP） for PalmOS 1.0[Sun Microsystems, Inc.] を用いた．また，サーバ・クライアント間の通信には IEEE802.11b の無線通信を利用した．

4.2.2　システム動作例

実装したプロトタイプシステムを動作させたときの様子を図 4.6 に，クライアント PDA 上に表示される画面例を図 4.7 に示す．

画面中央を中心とする同心円として描かれているのは，検索領域を表現するイメージである．表示系の処理および視認性の観点から，クライアントの画面に表示される検索領域イメージは単純円で構成される同心円とした．この同心円は検索基準点を分かりやすく示すための表示であり，実際の評価計算に用いられる検索領域

の形状とは無関係である．地図上に描かれたそのほかの円は登録されている飲食店を表し，円の大きさは Baum の手法によるその店舗の評価値を反映している．

各円は波紋が広がる要領でアニメーション表示させている．これにより，店舗位置の視認性を高め，近接する店舗を区別しやすくする効果がある．評価値の大きさを表示する円の大きさで表現するため，静止画像で表示すると評価値の高い（円の大きい）店舗に関しては店舗の位置（円の中心）が分かりにくくなる．アニメーションによる表示を行うことで，円の中心の確認と円の大きさの把握を同時に行うことが可能となり，近接して存在する店舗に関しても情報の区別が容易になる．

検索領域の拡大・縮小は PDA に搭載されたジョグダイヤルによって操作可能である．検索領域を変更すると範囲の情報をサーバへ送信し，新たな検索領域に適した縮尺の地図データと検索対象の新たな評価値を受信し，再表示する．地図データはポリゴンデータを集めた数値地図を利用しているため任意のサイズでの地図描画が可能である．

また，検索基準点の移動はスタイラスを使って画面をドラッグすることで操作が可能である．新たな検索基準点が決まると，基準点の座標と現在の検索範囲をサーバへ送信し，必要な地図データと更新された検索対象の評価値を受信して表示する．なお，あらかじめ表示矩形範囲の外側の地図データも受信しておけば，ドラッグ操作の際にサーバとの通信なしで移動先の地図データの表示が可能となるため有利と考えられるが，サーバ・クライアント間のデータ通信に関して予備実験を行ったところ，表示範囲外の地図データ転送がシステム全体のパフォーマンスに大きな影響を与えることが分かったため，今回の実装では表示矩形範囲と交差するポリゴンデータのみを転送する方針をとった．

検索領域の拡大・縮小

まず，検索基準点を一定に保った状態で検索領域を拡大させたときの表示結果の変化を示す．このときの実行例を図 4.8 に示す．

このとき，検索領域の変化が表示結果に与える影響について図 4.9 に示した例を用いて述べる．今，基準点 P の周辺には 3 つの検索対象 O_1，O_2，O_3 が登録されている．P から各検索対象までの距離 D_i は $D_1 < D_2 < D_3$ であり，それぞれの重要度を示す領域は $R_1 < R_2 < R_3$（ただし $R_1 \simeq R_2$）の関係である．

このとき，検索領域 R_P を拡大していったときの，各検索対象の評価値 W_i の変化を図 4.10 に示した．

図 4.10 において，O_1 と O_2 に注目すると，検索範囲 R_P が十分小さいときには基準点から近い O_1 のみが表示されることが分かる．R_P を拡大するに従い，O_2 が表示され，$R_1 < R_2$ であるので，拡大をつづけると O_2 が O_1 を順位で逆転する．この振る舞いは，基準点からの距離 D_i と検索対象の重み付け R_i を総合的に評価していることを示している．

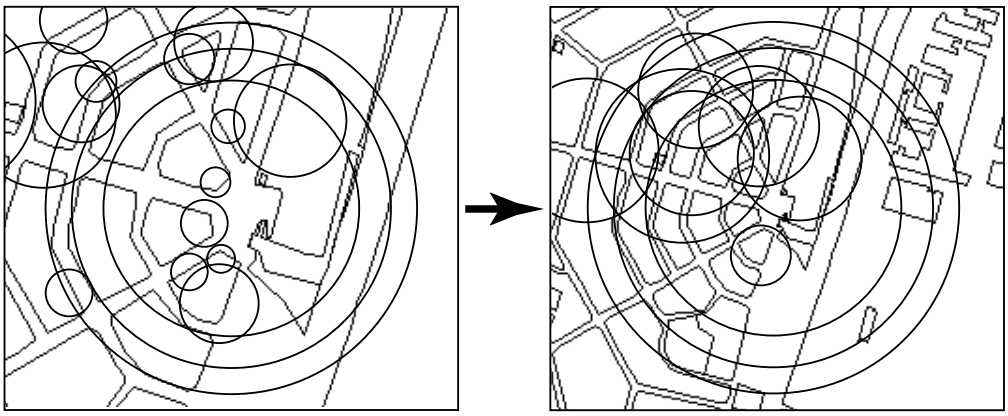

図 4.8: 検索領域の拡大（実行例）

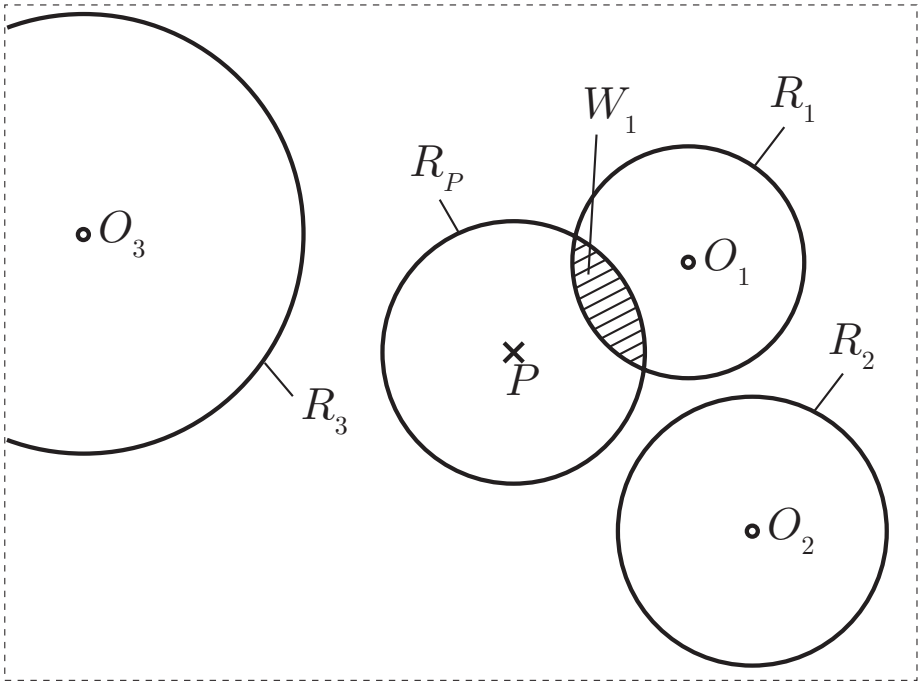

図 4.9: 検索領域の変化と評価値（モデル図）

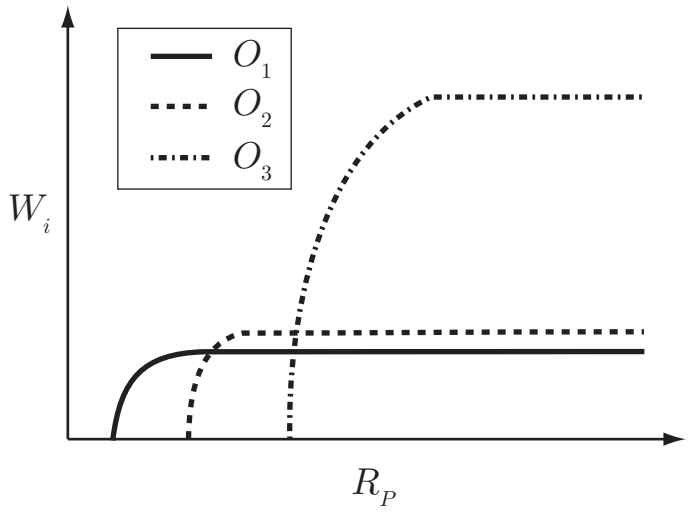

図 4.10: 検索領域の変化と評価値（グラフ）

次に，図 4.10 において O_1 と O_3 に注目する．重要度を示す領域の設定は $R_1 < R_3$ であり，一方で基準点からの距離は $D_1 < D_3$ である．つまり，O_3 は重要度は高いが遠い対象であり，O_1 は重要度は低いが近い対象である，検索領域 R_P が狭いうちは，距離の近さが評価に与える影響が大きいため，O_3 より O_1 が高評価となる．R_P を拡大していくと，やがて O_3 が表示され，ある区間で $W_1 \simeq W_3$ となる．このとき，基準点からの距離も重み付けも全く異なる O_1 と O_3 の評価値は，距離 D_i と重み付け R_i を総合的に評価した結果，同程度と判断されていることになる．

更に R_P を拡大し続けると，O_3 の重み付けの影響が大きくなり O_1，O_2 よりも遥かに高い評価値となり，やがてそれぞれの評価値は一定値に収束する．これは，検索範囲が十分に大きくなったため，表示範囲に比べて距離 D_1，D_2，D_3 それぞれの差はわずかな違いであると扱われていることを示している．

検索基準点の移動

次に，検索領域を一定に保った状態で検索の基準点を移動させたときの表示結果への影響を調べた．このときの実行例を図 4.11 に示す．

このとき，検索基準点の移動が表示結果に与える影響について，図 4.12 に示した例を用いて述べる．基準点 P 周辺の検索対象は O_1，O_2，O_3 の 3 種であり．P から各検索対象までの距離 D_i は $D_3 < D_1 < D_2$ であり，それぞれの重要度を表す領域は $R_1 << R_2 < R_3$ の関係である．

図 4.12 内の破線矢印で示されたルートに沿って，R_P を移動させたときの各検索対象の評価値 W_i の変化を図 4.13 に示した．これは，ユーザが検索領域を固定した状態で移動した場合に相当する．

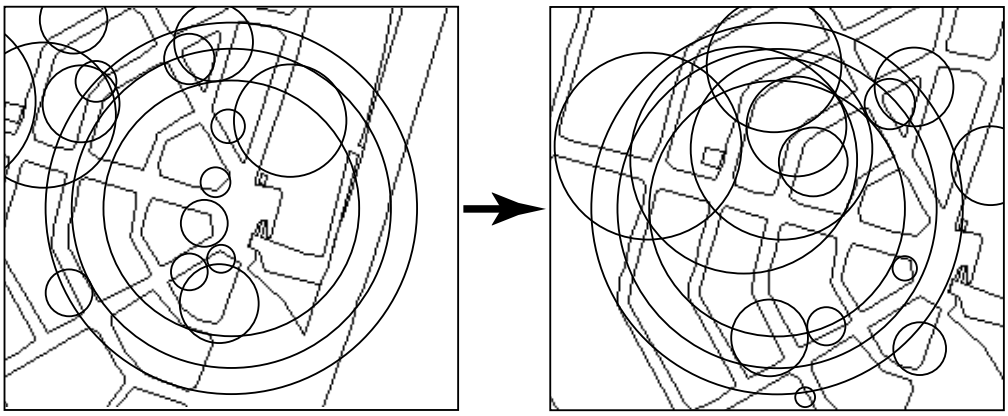

図 4.11: 検索基準点の移動（実行例）

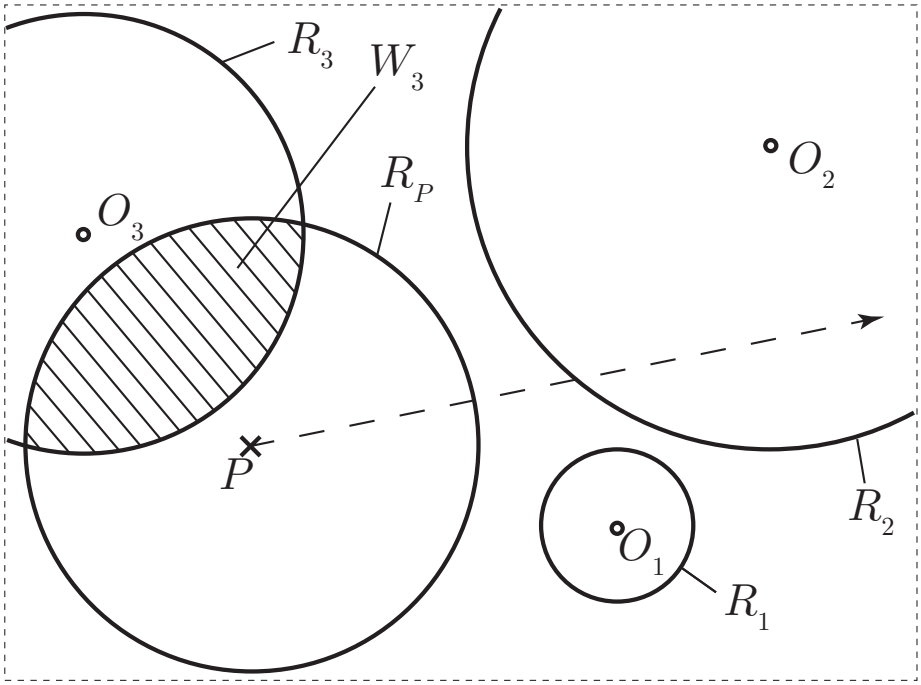

図 4.12: 基準点の移動と評価値（モデル図）

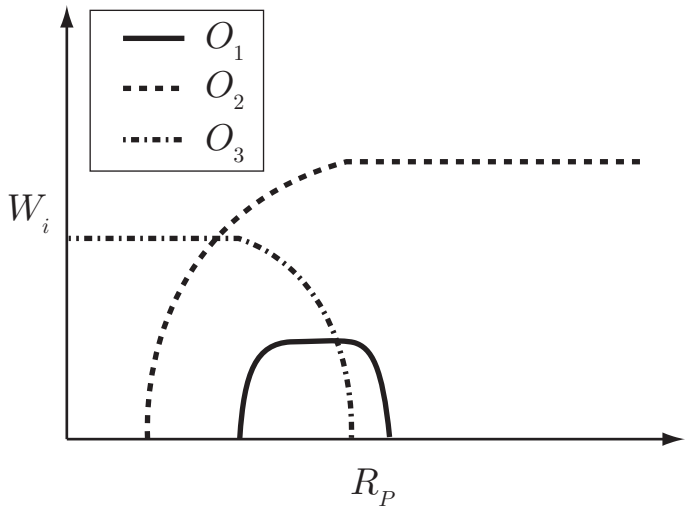

図 4.13: 基準点の移動と評価値（グラフ）

図 4.13 において，O_2 と O_3 に注目すると，初期状態では最も近くに存在する O_3 の情報が表示されていることが分かる．P が矢印方向に移動すると，O_3 から遠ざかり O_2 に近づいていくことになる．ある地点でユーザの端末には O_2 が表示され，O_2 に近づくにつれて O_3 の評価値は減少し，O_2 の評価値は増加する．この振る舞いは，Baum による対象評価が検索の基準点からの距離 D_i をよく反映していることを示している．

次に，図 4.13 において，O_1 と O_2 に注目する．$R_1 << R_2$ であるので，O_1 の評価値は O_2 に比べてかなり低い．特徴的なのは，O_1 の評価値がわずかな間しか観測されていないことである．これは，O_1 の重要度 R_1 が小さいため，ユーザがすぐ近くまで来ないと O_1 が表示されないことを示している．したがって，検索範囲が小さい場合，比較的重要度の低い対象に関しては，近くにあるものだけが表示され，遠くにあるものは非表示となる．

4.2.3　Baum 評価アルゴリズム

実装したプロトタイプシステムは第 3 章で提案した Baum の評価手法に従って作成されたものである．ここではプロトタイプシステムにおいて Baum の評価手法がどのような手順で実現されるかを説明する．

検索対象の評価手順

1. クライアントアプリケーションから，検索基準点 P の座標（x, y）および検索範囲の大きさが，サーバアプリケーションに伝えられる．検索範囲の大きさは，検索領域 R_P を単純円と見立てたときの半径に相当する数値である．

2. 　サーバには，その位置座標および 1 ～ 10 の 10 段階評価による重み付けが，検索対象として登録されている．予め登録されている検索対象各々について，重み付けの数値に比例した半径を持つ円として，検索領域 R_P との重なり部分を算出する．算出には Java 言語が提供する `java.awt.geom.Area.intersect()` メソッドを利用した．重なり部分の算出については後述する．

3. 算出された重なり部分の大きさを元に検索対象を並び替え順位付けを行う．

4. 重なり部分の大きいものから上位の一定数（実装したプロトタイプシステムでは上位１０位まで）を表示候補としてクライアントに転送する．このとき転送されるデータは検索対象の位置座標および評価値である．転送される評価値は算出された重なり部分に特定の係数（予備実験により携帯デバイスの画面サイズに合わせて求めた定数）を掛けたものである．

5. クライアントアプリケーションは受け取った検索対象の座標位置を基準として，評価値に応じた半径を持つ円として検索結果を表示する．

重なり部分の大きさの算出

図 4.14 は，領域の重なりを求める際に利用されるクラスのプログラムコード例である．`CircleRange` クラスは主として検索範囲や検索対象を示す円形を定義したクラスであり，地図平面上の x 座標，y 座標および検索範囲の大きさを示す半径で表される．重なり部分の大きさは，メソッド `score()` によって算出される．まず，楕円を定義した `java.awt.geom.Ellipse2D` のインスタンスとして検索範囲を `range1`，検索対象を `range2` として設定する．次に，`range1` と `range2` から `java.awt.geom.Area` のインスタンスを作成する．`java.awt.geom.Area` は，任意の形状の領域について，Add，Subtract，Intersect，ExclusiveOR などの構成領域ジオメトリ演算を提供するクラスである．`java.awt.geom.Area.Intersect()` メソッドにより検索範囲と検索対象の重なり部分を示す形状を得ることができる．そして，得られた重なり部分のバウンディングボックスを求め，その中に含まれる任意座標を走査して重なり部分に含まれる座標をカウントすることで，重なり部分の大きさを求めることができる．

サーバに登録されている検索対象の情報は静的なものであるが，Baum の評価方式により，検索基準点との相対距離が結果に反映されるため，ユーザの位置移動ま

```
public class CircleRange {

  int x;       // x 座標
  int y;       // y 座標
  int radius;  // 円の半径

  // コンストラクタ 座標と半径を設定する
  public CircleRange(int x, int y, int radius) {
    this.x = x;
    this.y = y;
    this.radius = radius;
  }
  // 円のバウンディングボックスを返す
  Rectangle getBounds() {
    return new Rectangle(x-radius, y-radius, 2*radius, 2*radius);
  }
  // もうひとつの円 range との重なり部分の面積を算出する
  public int score(CircleRange range) {
    return score(range.x, range.y, range.radius);
  }
  // 座標値 (x, y)、半径 radius の円との重なり部分の面積を算出する
  public int score(int x, int y, int radius) {
    Ellipse2D range1 = new Ellipse2D.Double();
    Ellipse2D range2 = new Ellipse2D.Double();
    // 2 つの円の java.awt.geom.Ellipse2D 表現を得る
    range1.setFrameFromCenter(this.x, this.y,
                              this.x+this.radius, this.y+this.radius);
    range2.setFrameFromCenter(x, y, x+radius, y+radius);
    // 2 つの円の java.awt.geom.Area 表現を得る
    Area area1 = new Area(range1);
    Area area2 = new Area(range2);
    // 重なり部分を抽出する
    area1.intersect(area2);
    // バウンディングボックスを取得する
    Rectangle bounds = area1.getBounds();
    // 重なり部分のカウントを行う
    int score = 0;
    for (int i = bounds.x; i < bounds.x+bounds.width; i++) {
      for (int j = bounds.y; j < bounds.y+bounds.height; j++) {
        if(area1.contains(i,j))
          score++;
      }
    }
    return score;
  }
}
```

図 4.14: 検索対象の評価値算出プログラムコード例

たは検索範囲の拡縮によって，表示結果は動的に変化する．

4.3　評価実験

4.3.1　評価目的

領域可変型情報検索手法 Baum を適応した周辺情報提示システムにおいて，検索対象がどのように評価されるのかを確かめるために，実装したアプリケーションを実際に動作させて次の 4 つの評価項目について評価実験を行った．

(1) 検索領域の拡大・縮小による評価値変動 ··· （第 4.3.2 項）

(2) 検索基準点の移動による評価値変動 ··· （第 4.3.3 項)

(3) 地形情報の考慮による表示への影響 ··· （第 4.3.4 項）

(4) 応答時間 ··· （第 4.3.5 項）

Baum を適用した周辺情報提示手法の最大の特徴は，地理的な位置関係に関して，依存する要素と独立な要素の双方を総合的に評価することができる点にある．モバイルコンピューティング環境における，目的地への移動というユーザの特徴的行動を考慮すると，提供される周辺情報はユーザの現在地と対象となる店舗・サービス施設との地理的な位置関係を重要な評価項目のひとつとして評価を行うべきである．しかし一方で，店舗・サービス施設の規模やサービスの質などを元にしたその対象の重要度は，周辺情報として地理的な位置とは独立に評価されるべき要素であるといえる．Baum では，後者の評価要素を各対象に設定された重要度を表す領域として表現することで，大域的にみたときに地理的条件とは独立に評価している．前者の評価要素に関しては，検索の範囲を表す検索領域を設定し，この領域をユーザの移動および拡大・縮小に合わせて位置・大きさを変えながら，各対象に設定された領域との重なり度合いを元に評価を行うことにより，ユーザの動的な位置変化・設定された検索領域の変化により変動する対象との地理的な位置関係を反映した評価としている．

したがって，本章における評価実験における目的は，Baum の適用により周辺情報提示システムにおいて，前述の理論的動作に合致した対象評価が行われることを検証することにある．このために，(1) 検索領域の拡大・縮小による評価値変動，(2) 検索基準点の移動による評価値変動についてそれぞれ評価実験を行う．(1) の評価項目においては，領域を拡大するにしたがって各対象の評価値が線形に増加し，検索対象の地理的位置および設定された重要度と比較して，検索領域が十分に大きくなったときに該当する対象の評価値が収束する振る舞いをみせると考えられる．(2) の評価項目においては，検索基準点の移動にしたがって，各対象の評価値

が線形に増減し，個々の対象について検索基準点に近づいた際に評価値が増加し，反対に離れた際に減少する振る舞いをすると考えられる．

また，Baum の検索手法においては，領域の重なり度合いが評価値を決定するため，領域をどのような形状で定義するかが表示結果に大きな影響を与える．この影響を調べるために，(1)，(2)，(4) の評価項目において第 4.1.5 項で定義した 2 つの領域モデルによって評価実験を行い，地形情報の考慮が対象の評価値へ与える影響を調べる．また，(3) 地形情報の考慮による表示への影響について評価実験を行い，地形情報を考慮したときの，地図表示を行った際の画面表示に与える影響を調べる．(3) の評価項目においては，地形情報の考慮によって直線距離では検索基準点から近くみえるが，地形的な距離で離れている対象に関して評価値の減少がみられると考えられる．さらに，システムの基本性能評価を示す付加的な参考評価実験として，(4) 応答時間の測定を行う．

ところで，一般にユーザインタフェースの研究においては，被験者実験によるシステムの印象評価などが行われることが多いが，システムの性質上，利用者の印象は登録データの充実度や情報としての品質，また地図表示系など提案内容以外の要素による影響が大きい．このような印象評価実験は，本論文の提案による行動計画支援の成果を，既存の経路計画支援を行うサービスと統合した実用システムの評価として行うべき評価項目であると考えられる．したがって，本章の評価実験としては，被験者によるシステムの印象評価実験は評価項目に含めなかった．

4.3.2　検索領域の拡大・縮小による評価値変動

まず，検索基準点を固定とした場合に検索領域を拡大・縮小することで，検索対象の評価値がどのように変動するのかを調べた．

検索基準点には駅前のロータリを選び，図 4.15 左の状態から同右の状態まで一定の割合で検索領域を広げていった．このとき，登録されている検索対象群の中で特徴的な変動が観測された 6 つ（Site 1〜Site 6）に関して，評価値の変化を図 4.16，図 4.17 に示した．また，これら 6 つの検索対象の地図上での位置を図 4.15 中に　1〜6 として示した．

図 4.16 において，検索領域が拡大するに従い各対象の評価値 W_i は次第に増加した後やがて一定値に収束する．これは，検索領域を十分大きくしたときに W_i は各対象の重要度 R_i に等しくなることを示している．Site 1 と Site 2 の評価値は初期状態においてすでに一定値に収束した状態であり，図 4.15 左の検索領域と比較して，Site 1 と Site 2 が検索基準点から非常に近いことを示している．

また，$R_2 < R_3$ の関係ではあるが，検索領域が小さいうちは $W_2 > W_3$ となっている．これは該当範囲においては，Site 3 の重要度に比べ Site 2 の基準点からの近さの方が評価値に与える影響が大きいことを示している．検索領域を拡大していくと，やがて $W_2 \simeq W_3$ となる区間がある．このとき，基準点からの距離も重要度も

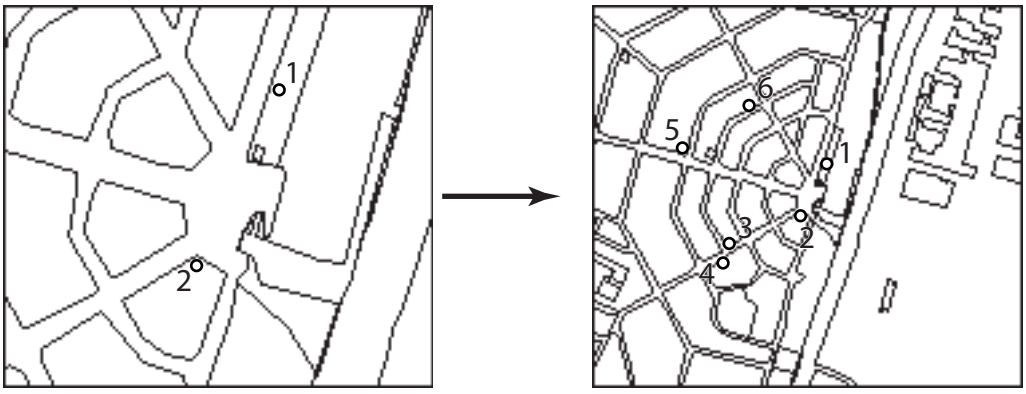

図 4.15: 検索領域の拡大

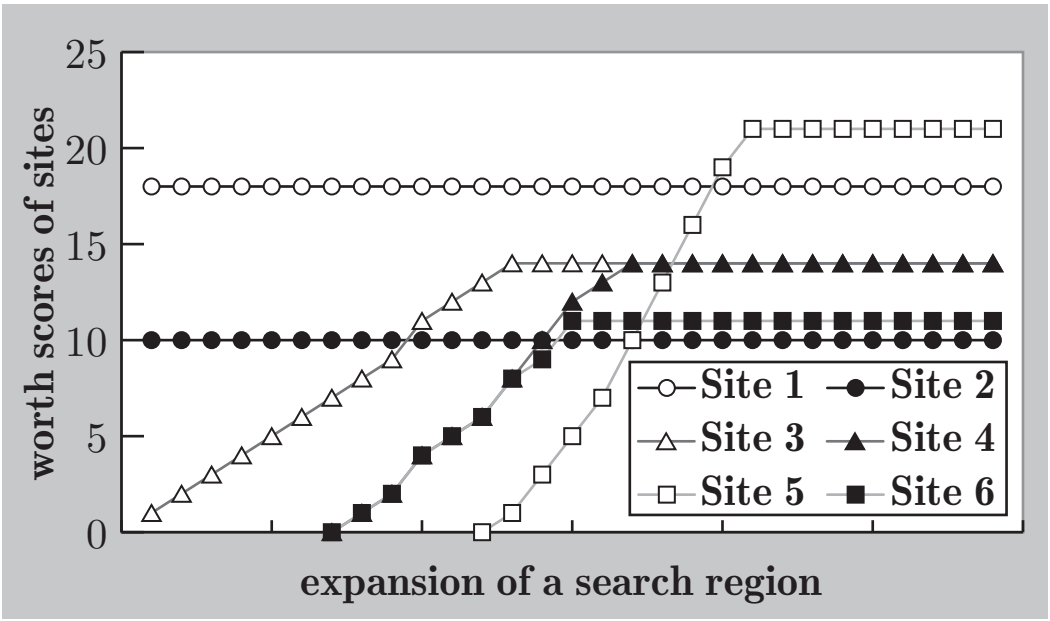

図 4.16: 検索領域の拡大による評価値変化（地形情報の考慮なし）

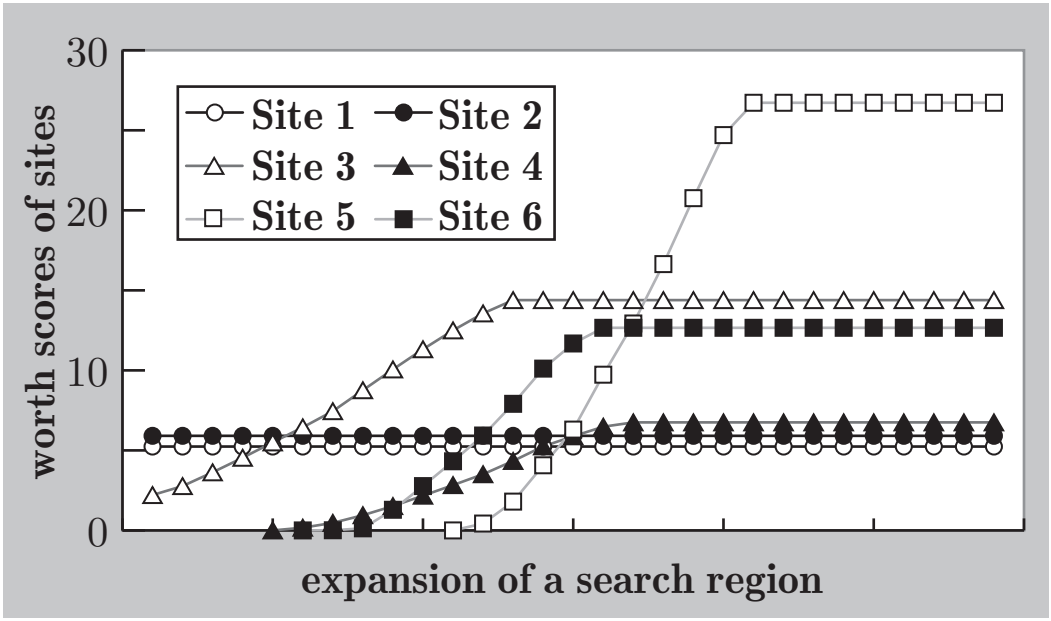

図 4.17: 検索領域の拡大による評価値変化（地形情報の考慮あり）

異なる Site 2 と Site 3 の評価値は，距離と重要度を総合的にみて同程度と評価されていることになる．さらに拡大を続けると Site 3 の評価値が Site 2 の評価値を上回る．これは検索範囲が十分に大きくなったために基準点から両対象までの距離の違いがわずかなものとして扱われていることを示している．

Site 4 と Site 6 に注目すると，基準点からの距離はほぼ同じであり重要度にも大差がないため，ある地点までは $W_4 \simeq W_6$ で推移する．しかし，検索範囲が十分大きくなると $R_4 > R_6$ であるため，$W_4 > W_6$ へと変化する．これも，検索範囲が小さいときは基準点からの近さの強い影響で拮抗していた両対象の評価値が，検索範囲の拡大によって重要度に応じたものへと変化するという点で，対象の重要度と基準点からの距離を総合的に評価している例であるといえる．

初期状態では評価値は Site 1，Site 2，Site 3 の順位である．検索領域を拡大していくのに従い Site 4，Site 6 が下位に現れ，Site 2 と Site 3 の順位が入れ替わる．最終状態においては，基準点から最も遠くに位置する Site 5 が最も高い評価値となり，Site 2 は最下位となる．実際に端末へ結果を表示する際には，このときの評価値を基準に上位から一定の件数を採用することになるため，結果として検索領域が小さいときには重要度が低くても基準点から近い対象が表示されることになる．一方で，検索領域を拡大すると重要度の低い対象は間引いて表示され，広範囲での概観的な表示が可能となる．

図 4.17 において，検索領域に関して地形情報を考慮した結果，全体的に評価値の低下がみられるが，おおむね変化曲線には変化が少ない．これは検索基準点を固

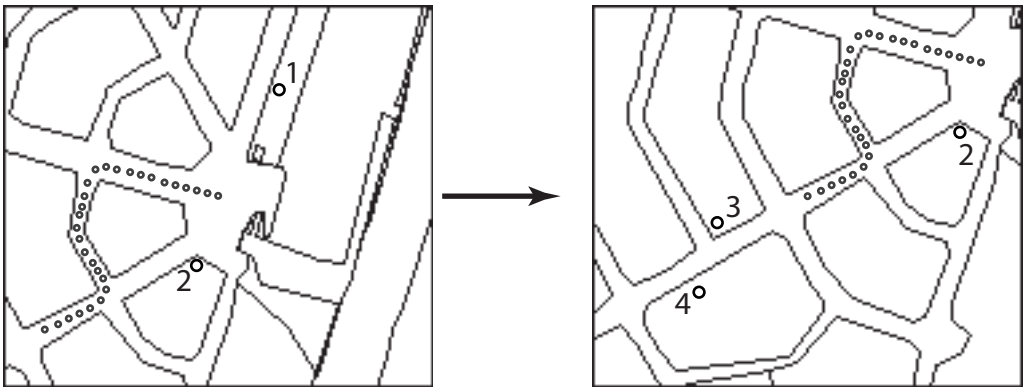

図 4.18: 検索基準点の移動

定して検索領域のみを拡大させているため，地形情報を考慮したことによる全般的な数値の変動はあるものの，領域の拡大にともなう評価値の変化特性には大きな影響がないことを示している．

4.3.3　検索基準点の移動による評価値変動

次に，検索領域の大きさを固定した状態で検索の基準点を移動させた場合に，検索対象の評価値がどのように変化するかを調べた．

移動の初期状態を図 4.18 左に，最終状態を同右に示した．図中の点線で示されているものは基準点の移動経路である．移動の初期位置には駅前のロータリを選び，初期状態は図 4.15 左と同一とした．図 4.16 でとりあげたのと同じ候補群 Site 1〜Site 6 に関して，評価値の変化を図 4.19，図 4.20 に示した．

図 4.19 において，Site 1 は初期状態で最も評価値が高いが，検索基準点の移動が Site 1 から離れていく経路をとるため，次第に評価値が減少していく．一方で，Site 3 については基準点の移動に従って検索基準点に近くなっていくため，評価値が増加していく．基準点を移動していくと，$W_1 \simeq W_3$ となる区間があり，与えられた検索範囲においてその場所を検索基準点とすると Site 1 と Site 3 を同程度に評価できることを示している．さらに基準点を移動させていくと，ある地点で Site 3 の評価値 Site 1 を上回り，そのときの基準点と検索範囲においては，遠くにある重要度の高い Site 1 よりも，近くにある Site 3 の方を高く評価していることになる．これは Site 1 が基準点から遠くなったため，その重要度の大きさが評価に与える影響が小さくなったことを示している．

Site 2 は基準点の移動による影響を受けていない．これは移動経路が Site 2 の周辺に限られていて，与えられた検索範囲に比べ十分に基準点から近いため $W_2 = R_2$ となっていることを示している．一方で，Site 5 と Site 6 はわずかな間しか観測さ

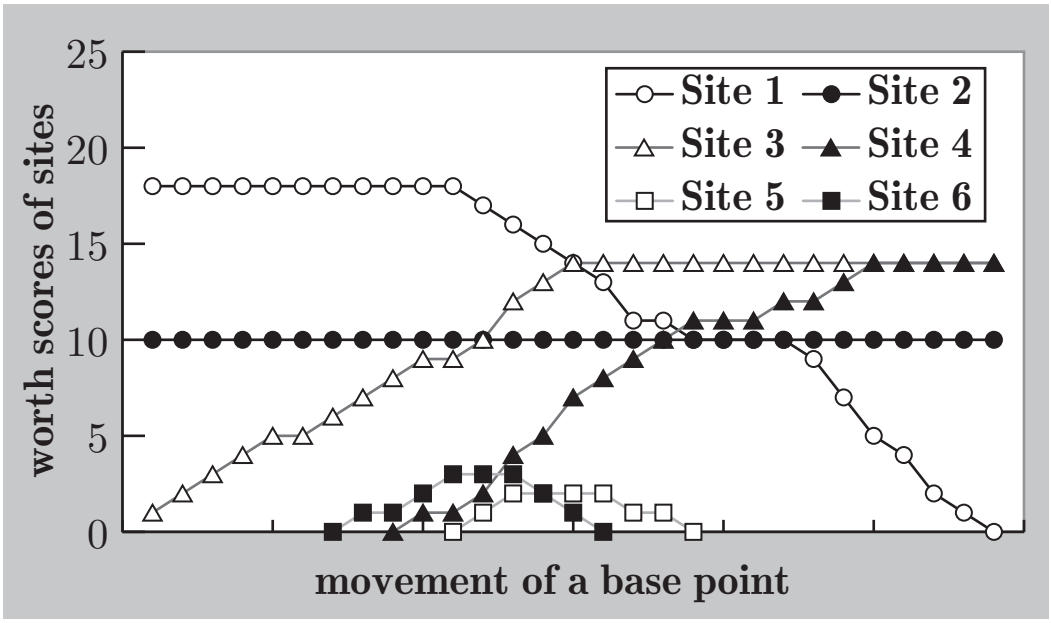

図 4.19: 検索基準点の移動による評価値変化（地形情報の考慮なし）

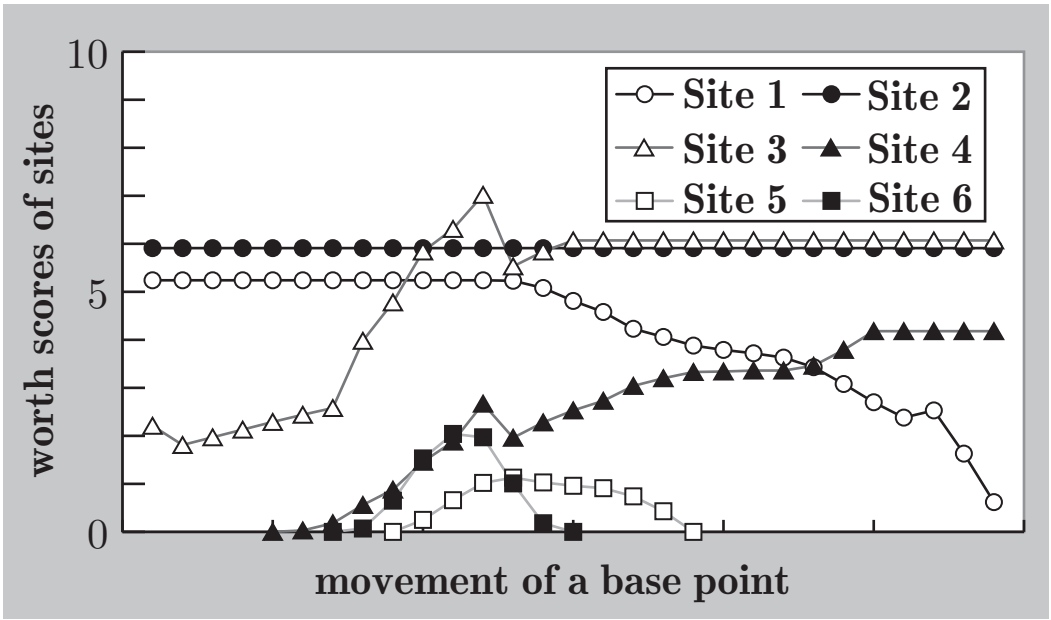

図 4.20: 検索基準点の移動による評価値変化（地形情報の考慮あり）

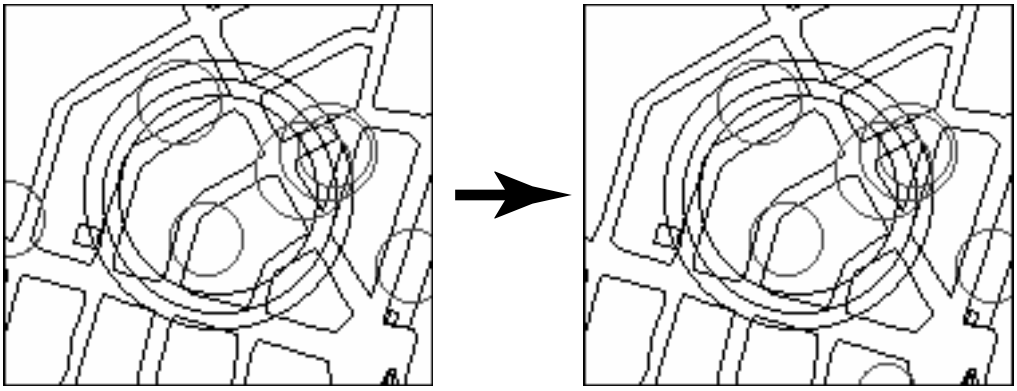

図 4.21: 地形情報の考慮による表示への影響

れていない．これは，両対象の重要度や検索範囲に比べ検索基準点からの距離が大きいため，両対象に接近したときにのみ評価値が増加することを示している．

初期状態では評価値は Site 1，Site 2，Site3 の順である．検索基準点の移動に従い，基準点から遠くなってしまった Site 1 は評価値が減少し，基準点に近くなった Site 3 や Site 4 の評価値が高くなる．また，この間つねに基準点近くとなる Site 2 は一定の評価値を保ち，さらに Site 5 と Site 6 は基準点が近くなったときにだけ評価値が上昇する．評価値の上位を端末へ表示することを考えると，結果として指定した検索基準点周辺の対象が表示されることになる．

図 4.20 において，第 4.3.2 項と同様に，検索領域に関して地形情報を考慮した影響による全体的な評価値の低下がみられる．さらに，検索基準点を移動しているために，第 4.3.2 項の場合よりも，個々の評価値に与える影響が大きく，変化曲線の形状に変化がみられる．これは，検索領域の形状が複雑になったために，検索基準点の移動に対して評価値が大きく変化することを示している．

4.3.4　地形情報の考慮による表示への影響

検索領域について，地形情報を考慮したときのクライアント上の周辺情報表示について図 4.21 に示した．地形情報を考慮せず，検索領域を単純な円形としている図 4.21 左では，検索基準点と検索対象との直線距離の影響が強いため，画面左端に経路的には比較的距離のある対象が表示されてしまっている．一方で地形情報を考慮した検索領域モデルを使用した図 4.21 右では，同じ対象は表示されていない．これは地図平面上での経路距離に近い評価を行ったことで，検索領域を単純円形としたときよりも厳密な評価を行うことができたといえる．

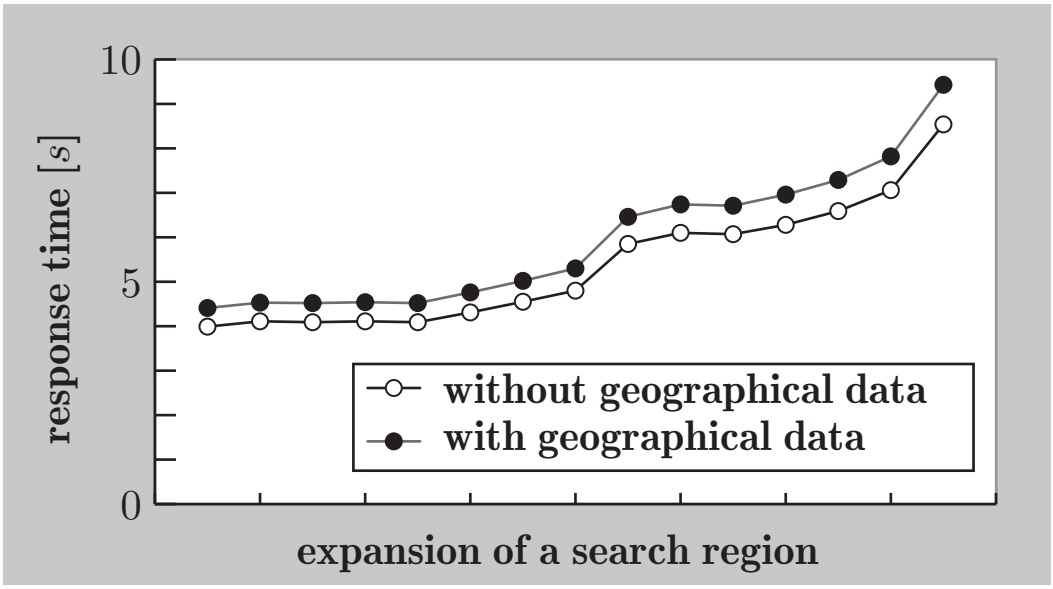

図 4.22: 応答時間（検索領域の拡大）

4.3.5　応答時間

クライアントアプリケーションの基本性能評価として，応答時間を調べた．第 4.3.2 項および第 4.3.3 項の測定を行った際に，携帯端末上で地図データと検索結果を受け取るのにかかる時間を測定し，それぞれ図 4.22，図 4.23 に示した．なお，測定には MIDP for PalmOS 1.0 の提供する `java.lang.System.currentTimeMillis()` メソッドを利用した．

図 4.22 において，検索範囲を拡大した際には領域の拡大にともなって応答時間が増加している．これは検索範囲を拡大することで，表示すべき地図範囲が拡大したことにより，サーバから受け取るポリゴンデータに含まれるポイント数が増加したことが原因と考えられる．図 4.23 において，検索基準点を移動した際には，ある地点で応答時間の減少がみられる．これもサーバから受け取る地図データに依存したもので，基準点を移動したことにより表示する地図範囲が変わり，ポリゴンデータに含まれるポイント数が減少したことが原因と考えられる．また，図 4.22，図 4.23 両方において，検索領域に地形情報を考慮したモデルを用いた方は単純な円形とした方に比べ応答時間が増加している．これは形状が複雑になった分評価値の計算時間が増加しているものと考えることができる．

検索領域を拡大したときに比較して，検索基準点を移動したときには応答時間の変動が小さく，ほぼ一定となっていることから，応答時間は受信する地図データにおけるポリゴンを構成するポイント数に大きく依存していると考えられる．現在の実装では，サーバはクライアントから要求があるたびに，指定された座標と検索領

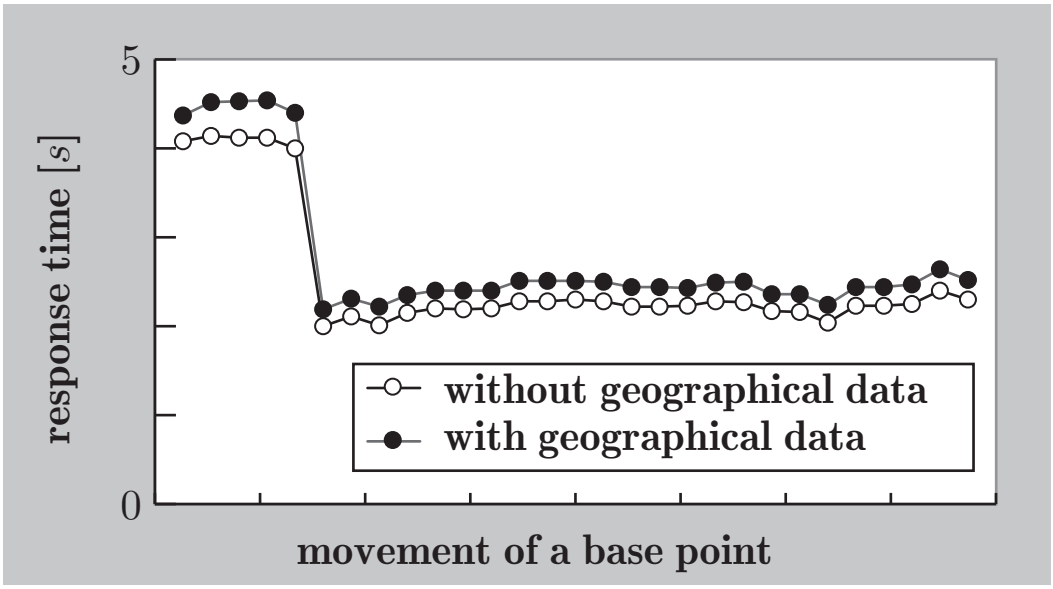

図 4.23: 応答時間（検索基準点の移動）

域に適した地図データをすべて送信している．パフォーマンス向上のためには，地図データの差分転送化やデータの圧縮などを検討する必要があると考える．

4.4　まとめ

本章ではモバイルコンピューティング環境において，ユーザに周辺情報提示を行うシステムについて述べた．多数の登録情報の中から表示範囲の大きさに応じた情報を抽出して表示するために，領域可変型検索手法 Baum を適用した．Baum 方式による対象の評価により，検索の基準点から対象までの距離と対象の重要度を総合的に評価できることを示した．また，先行研究において課題であった検索領域の形状モデルについて，地形情報を考慮する拡張を行った．さらに，拡張したモデルを適用したプロトタイプシステムによる評価実験を行い，定量的評価として，検索対象の評価値変動およびシステムの応答時間を測定した．

Baum による周辺情報提示システムは，検索基準点と検索領域のみを指定すれば，表示範囲に応じた周辺情報表示が可能である．基準点から対象までの距離と，対象に設定された重要度とを総合的に評価することで，検索領域が広い場合には概観的な，狭い場合には網羅的な周辺情報提示が可能である．提案した周辺情報提示アプリケーションによって，ユーザは街中において行動範囲や現在地に応じた周辺情報を参照することができる．

第5章

構造可変モバイルエージェント機構

本章では，構造可変モバイルエージェント機構 AgentStack を提案する．AgentStack の特徴は，データ処理の対象となる環境の変化やアプリケーション要求に応じて，その構成や機能を変更できることにある．具体的には，リソース制御，データ変換，データ選別，データ統合，ノード間通信のネットワークデータ管理・運用における基本機能からなる階層構造を持つミドルウェアであり，さらに各階層をモバイルエージェントをベースとするコンポーネントとして実現する．この結果，モバイルエージェントのコンピュータ間移動性を通して，ミドルウェアの機能を変更・拡張することが可能となり，その際に各コンポーネントに対して統一されたインタフェースを規定することにより，他の階層への影響を最小化することもできる．本章では提案するミドルウェアのアーキテクチャを示すとともに，その実装に関して説明する．また，二つの応用事例を通して，提案するミドルウェアの有用性と柔軟性を議論する．

5.1 モバイルエージェントによるコンポーネント階層

5.1.1 設計指針

本章で提案する AgentStack は，ネットワーク上で分散管理されるデータリソース（実世界情報データベース）から得られるデータを統合することで，広範囲な周辺情報提示サービスを実現することを目的としている．また，アプリケーションの大幅な改変なしに，多種多様かつ機能制限の厳しい携帯端末に対応することができるミドルウェアを目指す．

5.1.2　システム要求

各ノードは多様なデータリソースを扱うとともに，他のノードとデータを円滑に交換する必要がある．これらをアプリケーションに対して代行する方法として，ネットワークデータを管理・運用するためのミドルウェアが必要となる．なお，このミドルウェアはネットワークの特性から次のような要求を満足する必要がある．

汎用性： 前述のように各ノードが持つ計算能力や記憶容量などへの制限が多い一方で，あらかじめ想定した用途だけでなく，開発時に未知な用途を含む多様な用途に利用できることが求められる．このため，各ノードが必要なソフトウェアを必要なときだけ保持・実行することが求められる．

適応性： 処理対象のデータリソースやアプリケーションの要求が変化した際には，該当するノードが提供するデータ処理方法やデータ選別などの機能を動的に変更することが求められる．

スケーラビリティ： ノード数増加による各ノードやネットワーク全体への影響を最小化するため，非集中制御が求められる．各ノードが収集するデータは多量となる傾向があるが，その一方で各ノードは無線通信などで接続されることも多く，その通信帯域は広いとはいえない．したがって，必要なデータだけを転送する機構が求められる．

独立性： データリソースの種類は多様であり，そのデータ形式も多様である．データリソースの制御方法においてもデータリソース自身がデータを常時または変化時に出力するもの（プッシュ型），外部からの問合せに応じてデータを出力するもの（プル型）などがある．ミドルウェアはこうした様々なデータリソースを扱える必要がある．また，オペレーティングシステムやハードウェアに独立であることが望まれる．

5.1.3　動的拡張性を持つミドルウェア

提案するミドルウェアの基本構成および機能について概説する．前述のネットワークデータ向けミドルウェアに対する要求事項を満足するため，各ノード上で必要とされる各種機能をモバイルエージェントによるプラグインコンポーネントとして実現する．ここで，モバイルエージェント [Fuggetta *et al.* 98, 佐藤 99] はコンピュータ間を移動できる自律的なプログラムであり，その移動に際してはプログラムコードだけでなく，変数などの実行状態もデータ化して転送できるため，移動先では移動直前の実行状態から処理を継続できる．この結果，従来のプログラムコードを主体としたコンポーネントと比較して，コンポーネントに対する各種設定を行った状態での遠隔コンピュータへの配布に適している．これは，データリソース

や通信制御などにおいて多くの設定が必要となるネットワークデータ管理においては有用な特徴となる．また，このコンポーネントはモバイルエージェントの移動性や自律姓を継承することから，コンポーネント自身が配布先のノードを発見し，そのノードにそれ自身または複製を移動させることで，遠隔ノードへのコンポーネントの配布・稼働を実現できる．

コンポーネント拡張・置換では依存関係によるミドルウェア全体への影響を考慮する必要がある．ところで，一般的にデータ統合の処理は，次のような手順で実行されることが多い [Brooks *et al.* 98]．

1. データリソースの制御・監視によるデータ取得
2. 取得したデータの形式変換
3. データの変化量などによる選別
4. 選別されたデータの統合処理

そこで，提案するミドルウェアではこれらの 4 つの処理に加えて，統合済みデータを遠隔ノードに転送する

5 ノード間通信機能

を含む 5 つの機能に分けて構成する．さらに，これらの機能は測定環境やアプリケーション，ネットワークの変化に応じて変更が要求されることから，その機能をモバイルエージェントによるプラグインコンポーネントとして実現し，拡張・置換を可能とする．なお，これらの 5 つの機能はデータ処理の流れ（つまり各機能の番号順）に応じた依存関係を持つことから，それに対応した図 5.1 のような階層モデルとしてコンポーネントを構成する．さらに階層間インタフェースを定めることにより，コンポーネントの拡張・置換時の影響を最小化・明確化できるようにする．

- **リソース制御層**（Reception Layer）のコンポーネントは，ノードに接続されているデータリソースの制御・監視を行うものであり，データリソースに対して所定の間隔でデータ要求を行い，必要なデータを収集する．
- **データ変換層**（Presentation Layer）のコンポーネントは，リソース制御層のコンポーネントから受け取ったデータリソースに依存した表現データをアプリケーションによる処理に適した表現データ形式に変換する．データの正規化を行うことで，これより上の階層ではデータリソースの相違によるデータ表現の違いを意識せずに処理が可能となる．
- **データ選別層**（Filtering Layer）のコンポーネントは，特定の閾値を超えるデータや，一定期間内に大きな変動があったデータのように，ある与えられた条件に合致するもののみを選び，選別済みデータとして出力する．

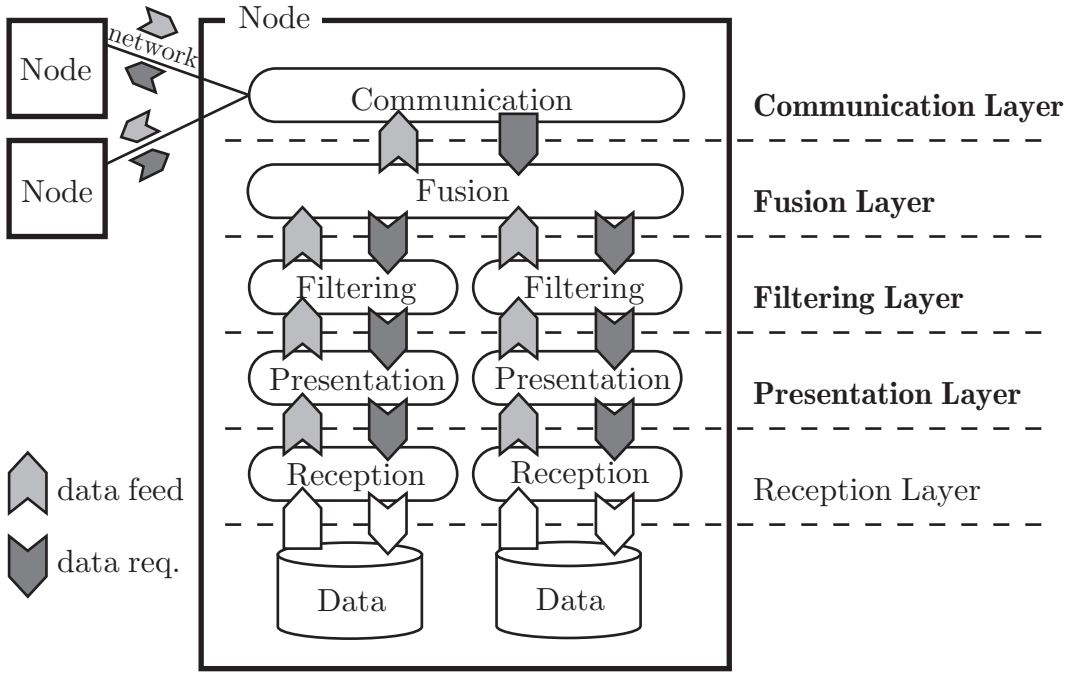

図 5.1: コンポーネントアーキテクチャ

- **データ統合層**（Fusion Layer）のコンポーネントは，ノードに接続された複数のデータリソースから集められたデータを統合することで，データの精度向上や，相違な対象による複合データを生成し，統合済みデータとして出力する．

- **ノード間通信層**（Communication Layer）のコンポーネントは，ノード間の通信プロトコルを定義し，他のノードとのデータ交換や，特定のホストへのデータ送信を行う．なお，このコンポーネント自身がその移動性を利用してデータを収集・移送することも可能である．

たとえば，ノード間の通信プロトコルを変更するときは通信対象となるノード上のノード間通信層のコンポーネントを，またデータリソースの出力する単位系を変えるときはデータ変換層のコンポーネントを，そして収集すべきデータを変えるときはデータ選別層のコンポーネントを入れ替える．なお，この入れ替えは特定サーバまたは他のノードから必要なコンポーネントを複製・移動させることにより実現する．また，各層は他の層に非依存な単一の機能を提供するとともに，階層間には統一されたインタフェースを提供することから，各階層のコンポーネントを置換・拡張した際の影響はその層または依存関係がある下位層に限定される．

5.2　設計

5.2.1　モバイルエージェント

本論文で提案する，構造可変モバイルエージェント機構 AgentStack のミドルウェア構成を図 5.2 に示す．ミドルウェアは各ノードに用意され，コアシステムとコンポーネント，そしてコンポーネントの実行・移動を実現するモバイルエージェントのランタイムシステムの 3 つの部分で構成される．前述のように，AgentStack ではネットワークデータを制御・管理する主要機能を Java 言語で実装されたモバイルエージェントによるコンポーネントとして実現する．モバイルエージェントは独立した計算実体として導入され，モバイルエージェントの合成や置換などの機能は提供しない．これらの機能をコアシステムおよびフレームワークライブラリとして実現することにより，多様なモバイルエージェントシステムを拡張することなく利用できるようにする．なお，ネットワークデータの制御・監視は移動先コンピュータのリソースを利用することから，エージェントの移動前や終了前には獲得したリソースの解放，また生成先や移動先では必要なリソースの獲得などの処理を明示的に行う必要がある．したがって，このミドルウェアから利用可能なモバイルエージェントシステムはセキュリティ機能が許す限りリソースを利用できることと，エージェントの実行モードが遷移した前後に明示的にエージェントにそれを通知する機能が必要である．また，後述するようにエージェントの移動手段は Weak Migration を前提とする．ただし，これらは既存の多くの Java 言語モバイルエージェントシステムの利用を制限するものではない．

5.2.2　コアシステム

コアシステムは各階層間にインタフェースを規定し，測定環境やアプリケーション要求の変化に応じて必要なコンポーネントの配布・置換を制御する．

階層コンポーネントのインタフェース

AgentStack ではコンポーネントを図 5.3 のように階層的に構成する．この階層間インタフェースは隣接する階層のコンポーネントに対する 1 つ以上の入出力ストリームとして与えられる．上層から下層へのインタフェースは制御命令の転送用ストリームとなり，下層から上層へのインタフェースはデータリソースから得られたデータの転送用ストリームとなる．各ストリームは Java 言語の I/O Stream を利用したものであり，その接続・切断はコアシステムにより制御される．

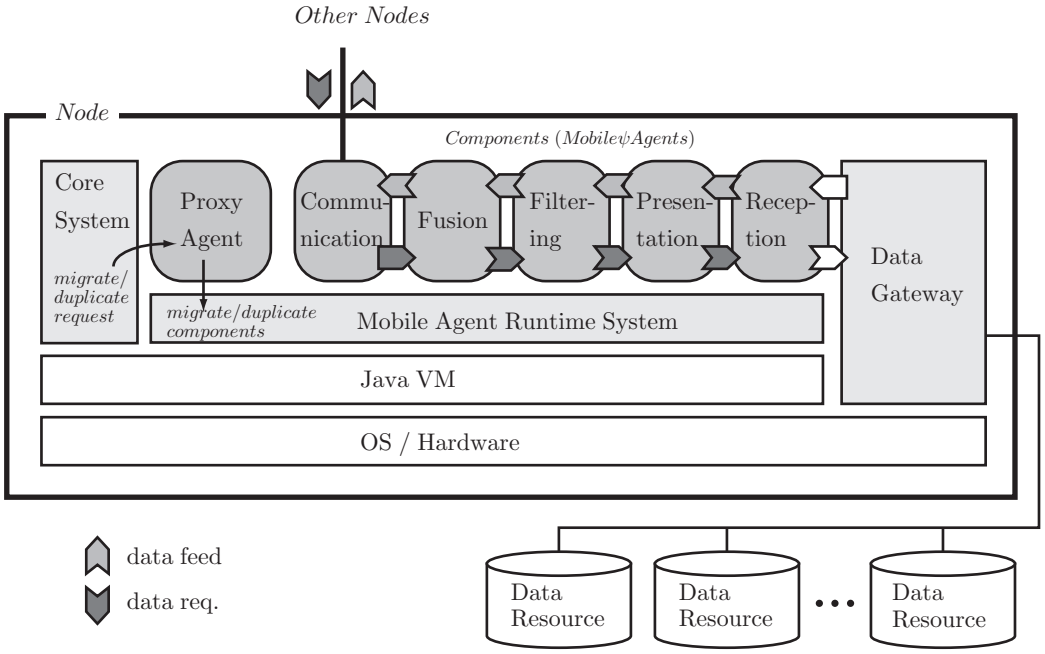

図 5.2: ミドルウェアの構成

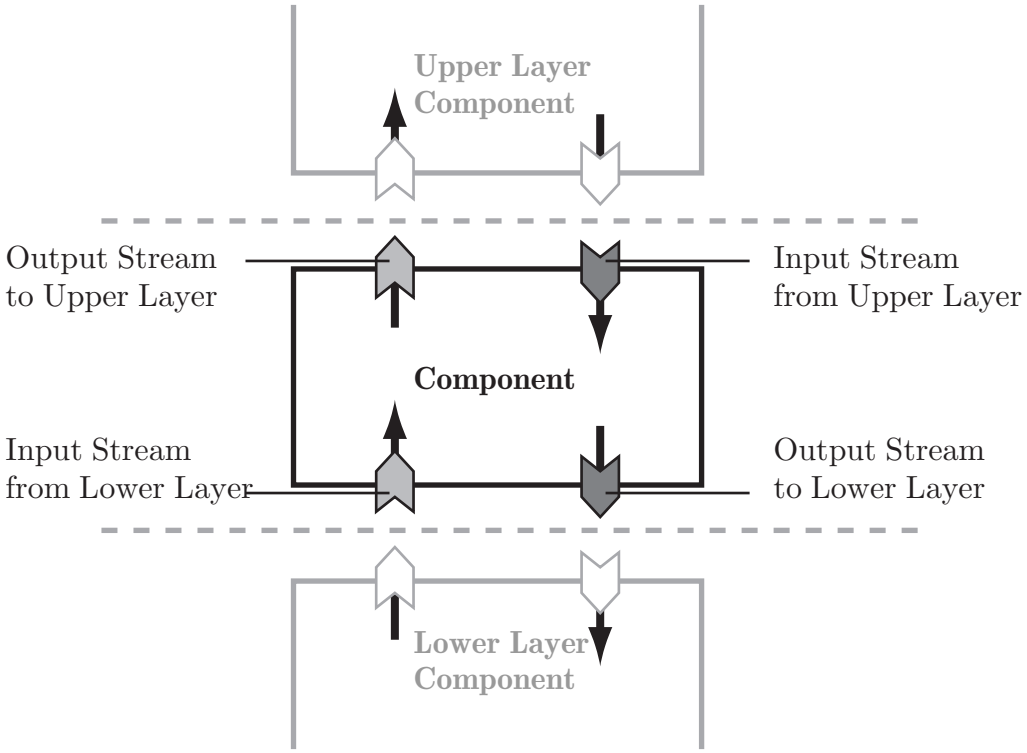

図 5.3: コンポーネント間インタフェース

コンポーネントの配布

コンポーネント配布には次の 3 つの方法がある．

1. あらかじめコンポーネントを配布しておく．
2. コンポーネントを実現するモバイルエージェントの自律性を利用して，コンポーネント自身が移動・複製を行うことにより，遠隔ノードにそれ自身を配布する．
3. 対象となるデータリソースやアプリケーション要求の変化に応じて必要なコンポーネントを配布・取得する．

2 つめおよび 3 つめの配布方法について，その手順を次に述べる．

エージェントの自律性を利用した配布

1. 特定のノードにおいて新たにデータ処理を行う必要が生じた場合，隣接するノード上で既に実行中のコンポーネントは，自己を複製して該当サーバへ転送する．
2. 転送されたコンポーネントは移動先のノードにおいて，転送元ノードで実行されているのと同様のデータ処理を行い，ノード間通信層コンポーネントを介してデータを共有する．
3. コンポーネントによる処理が実行中のノードにおいて，環境の変化やアプリケーションの要求により，該当ノードでのデータが不要となった場合には，コンポーネントは処理を停止して自己をノード上から削除する．

この配布方法によって，コンポーネントをデータ処理が必要なノードにだけ配布することが可能であり，各ノードのリソースやネットワーク上のデータ転送量を最小化することができる．また，不要になったコンポーネントは，実際にはデータ処理の停止後すぐに削除されるわけではなく，実行を停止したままで一定時間該当ノード上にキャッシュされる．したがって，隣接するノード間において，データ処理の停止と開始にタイムラグが生じた際にも，キャッシュされたコンポーネントの転送によって，円滑なコンポーネント配置が実現できる．

環境変化に適用した必要コンポーネントの配布

1. ノードにおいてデータリソースの環境やアプリケーション要求，自身のシステム構成の変化を検出したとき，隣接するノードまたは所定のサーバにメッセージを送信する．このときのメッセージには，その変化を検出に必要なコンポーネントすべての識別子を含んでいる．

2. コンポーネントの要求メッセージを受け取ったノードまたはサーバは，そのコンポーネントを保持していれば複製を作って送り返す．保持していない場合はその要求メッセージを他のサーバまたはノードに転送する．なお，メッセージの増加を避けるため転送回数は制限可能である．

この配布方法によって，システム構成時にあらかじめ想定していなかった機能であっても，コンポーネントとして必要に応じて導入することができる．また，各コンポーネントは TTL（Time-To-Live）フィールドを持ち，有効期間を明示的に制限することもできる．したがって，所定の時間内に利用されなかったコンポーネントは自動的に削除または所定のサーバに期間させることもできる．

コンポーネントの置換

AgentStack の目的の 1 つはノードにおけるのデータ処理機能を動的に変更することにある．各ノード内のコンポーネント置換手順を示す．

1. ノードに新しいコンポーネントとなるエージェントが到着する．
2. ミドルウェアはそのコンポーネントの入るべき階層を特定し，該当する階層の既存コンポーネントにおける上層および下層とのストリームを切断する．
3. 新しいコンポーネントが同じ階層の既存コンポーネントと排他的は関係になるときは既存コンポーネントを取り除く．
4. ミドルウェアは新しいコンポーネントの入出力ストリームを上層および下層のコンポーネントと接続する．

なお，現在の実装ではコンポーネント置換過程に検出されたデータの補完は行っていない．ストリーム内部にバッファリングし，接続後に補完することも可能であるが，想定環境における処理データは即時性が要求されることに加えて，一般にモバイルネットワークではノードのネットワーク切断やエラーが頻繁に起きるため，それを利用するアプリケーションはこれらを前提に構成されていることが多いこと，本論文の対象はモバイルコンピューティング環境におけるユーザの行動計画支援にあることから，データ欠損に寛容であることが多いことなどからデータの補完は行っていない．また，取り除かれたコンポーネントは将来の利用に備えてコアシステム内のキャッシュに所定の時間だけ保持することもできる．

5.2.3　コンポーネント階層

AgentStack では前述のようにノードにおけるデータ処理の機能をデータリソース制御，データ変換，データ選別，データ統合，ノード間通信の 5 つの機能に分割する．それぞれの階層コンポーネントが提供する機能を説明する．

データリソース制御

数多くのデータリソースが存在するため，それらの制御方法は多種多様となる．この層ではデータリソースを含めたハードウェアと低レベルソフトウェアに関する処理を行い，こうしたデータリソースによる相違を吸収する．たとえば，データリソースにはデータを受動的に取り出すもの（プル型）と，能動的に出力するもの（プッシュ型）がある．前者のデータリソースには問合せコマンドを所定間隔や上層から要求に応じて送信してデータを取得する．一方，後者のデータリソースはデータリソース自身が所定間隔やデータ変化時にデータを送信してくることから，各ノードにバッファを設けて最新のデータを格納することで，上層からのデータ取得要求時にただちに結果を返せるようにする．

データ変換

データリソースはその制御方法だけでなく，得られるデータの表現方法も相違する．たとえば地図情報データにおいては，その座標系によって TD（Tokyo Datum），WGS84（World Geodetic System 1984），ITRF（International Terresrial Reference Frame）など様々なデータ形式が存在する．この層では，精度や単位などについてデータ表現の変換を行う．この層のコンポーネントは，下層のデータリソース制御層のコンポーネントから 1 本の入力ストリームで結ばれ，データを受け取ると自動的にデータ表現を変換し，その変換結果を上層のコンポーネントに出力ストリームを介して送る．なお，上層からの制御命令はそのまま下層に転送する．コンポーネントはモバイルエージェントとして実装されているので，各ノードはアプリケーションや他のノードの要求に応じて，このデータ変換用コンポーネントを取得・置換することができる．

データ選別

モバイルコンピューティング環境におけるアプリケーションでは，取得できるデータのすべてを必要とはしないことが多い．必要なデータだけを選別することにより，ノード間で交換されるデータ量を減らすことができる．ただし，選別方法はアプリケーション要求や環境の変化によって変わる．したがって，選別を行うコン

ポーネントを用意し，下層のコンポーネントから受け取ったデータから必要なものだけを上層のコンポーネントに転送させる．なお，他層と同様にコンポーネントはモバイルエージェントとなることから，そのノード間移動を通じて配布・置換を行うことができる．また，データ選択ポリシは入力ストリームから出力ストリームへのフィルタプログラムとして定義され，コンポーネントは下層のコンポーネントに対して所定の間隔でデータを要求することもできる．

データ統合

1 つのノードに複数のデータリソースが接続されている場合，それぞれのデータリソースから得られたデータを統合することで，データの精度向上や条件の絞り込みなどが可能である．この層におけるコンポーネントは上層に対する出力ストリームは 1 つであるが，1 つ以上の下層コンポーネントと接続するための複数の入力ストリームとそこから得たデータを統合するためのプログラムを持つ．また，このデータ統合用コンポーネントは後述のノード間通信用コンポーネントに入力ストリームをつなぐことができ，その場合，他のノードによるデータの統合にも利用できる．

ノード間通信

モバイルコンピューティング環境では，多数のノードから得たデータを特定のホストへ集めて処理を行うアプリケーションが多くみられる．また，他のノードからデータを得ることで，データの補完や統合を行うこともある．この層では，これらのホストやノード間でデータ交換を行う機能を提供する．各コンポーネントは，下層のコンポーネントに接続された入力ストリームを持ち，これを介して遠隔ノードとデータ交換を行う．AODV[Perkins 97] や DSR[Broch *et al.* 98] のように，無線アドホックネットワーク向けに多くのルーティングやフォワーディングのアルゴリズムが提案されており，他ノードとの通信プロトコルとして，各コンポーネントにこれらを定義することでシステムの柔軟性を高めることができる．なお，モバイルエージェントの移動性や自律性を利用することにより，コンポーネント自身がデータを他のノードに移送することもできる．この場合，プログラム自身による経路制御ができるため，柔軟な径路制御が可能となる．

5.3　実装

AgentStack は Java 言語 [Arnold *et al.* 00] の Java 仮想マシン（JDK1.1 以上，Personal Java を含む）上で稼働する．以下ではそのモバイルエージェントシステム

```
public abstract class Agent
    imlements Serializable {
     // エージェント生成後のコールバックメソッド
     void create();
     // エージェント消滅時のコールバックメソッド
     void destory();
     // エージェント移動時のコールバックメソッド
     // void leave(URL dst);
     // エージェント到着時のコールバックメソッド
     void arrive();
     // エージェント複製後のコールバックメソッド
     void clone();
     ....
     //  自分自身を移動
     void go(URL dst) throws
       NoSuchLocationException ... { ...}
     //  自分自身を複製
     void duplicate(URL dst) throws
       NoSuchLocationException ... { ...}
     // ライフタイムを設定する
     void setTimeout(int time);
     ....
}
```

図 5.4: Agent クラスのプログラムコード例

として AgentSpace[佐藤 98] を利用している場合について概説する．なお，Aglets [Lange *et al.* 98] などの他の Java 言語によるモバイルエージェントシステムを利用しても容易に実現できると考えられる．

5.3.1　モバイルエージェント

前述のようにコンポーネントはモバイルエージェントとして実装され，その実行や移動はモバイルエージェントのランタイムシステムによって実現される．各エージェントは Java オブジェクトであり，`Agent` クラスのサブクラスとして定義される（図 5.4）．ここでメソッド `create()`，`destroy()`，`leave(URL dst)`，`arrive()`，`clone()` はコンポーネントの生成直後，終了直前，移動直前，移動直後，複製直後

にモバイルエージェントのランタイムシステムにより呼び出されるモバイルエージェント側のコールバックメソッドである．エージェントはこれらのメソッドにそれぞれの実行モード遷移に必要な処理を定義することになる．ところで，コアシステムはモバイルエージェントとして実現されたコンポーネントの移動や複製を行う際に，モバイルエージェントのランタイムシステムを制御する必要がある．ただし，ランタイムシステムへの拡張を回避するため，コアシステムの代理となるエージェントを用意し，コアシステム制御要求を受け取ると対応したランタイムシステム上の対応する API を呼び出すこととした．

なお，モバイルエージェントの移動では，移動対象となるエージェントプログラムの範囲に応じて Strong Migration 方式と Weak Migration 方式に分かれるが，AgentStack ではコンポーネントは Weak Migration 方式によって移動するものとする．この結果，移動時にインスタンス変数は保存されるが，ローカル変数やプログラムカウンタは対象外となる．ただし，モバイルコンピューティング環境では各コンポーネントはデータリソースを含む多様な計算資源を獲得・解放する必要があり，仮に Strong Migration 方式を用いたとしても，移動透過の実現は困難となることから，Weak Migration 方式の選択による制限は少ない．また，Java 言語による多くのモバイルエージェントシステムは Weak Migration 方式に基づいていることから，結果として多くのモバイルエージェントシステムが利用できることになる．エージェントの認証やノードの保護はモバイルエージェントシステムおよび Java 言語の提供するセキュリティ機構を適用している．

5.3.2　コンポーネント

各階層のコンポーネントは `Agent` クラスのサブクラスとなり，ここではデータ変換層とデータ選別層コンポーネントの定義を宣言する `AbstractComponent` 抽象クラスを図 5.5 に示す．具体的なコンポーネントは同クラスの実装となる．

`AbstractComponent` クラスでは，下位層および上位層へのインタフェースとして合計 4 つの入力または出力ストリームを持ち，下位層からの入力ストリーム `inputLower` からデータの読み出し，それぞれのデータ処理を行った結果を上位層への出力ストリーム `outputUpper` に書き込む．そして，引数なしの `connect` メソッドを呼び出すことにより階層間接続を実現するが，このメソッドはコンポーネント到着後にコアシステムにより自動的に呼び出される．なお，引数付き `connect` メソッドは接続するストリームを明示的に指定するものである．また，`connect` と同様に移動直前にはストリームの切断処理を行うメソッドが自動的に呼び出される．

また，各コンポーネントにはライフタイム `time` が定義される．コンポーネントは自身がロードされているランタイム内でコンポーネントの転送が行われる度にライフタイムをデクリメントする．実際のデクリメント操作はコアシステムによって行われる．`time == 0` になったコンポーネントは，次回のライフタイムデクリメ

```
public abstract class AbstractComponent
    extends Agent {
  // エージェントのライフタイム
  protected int time;

  //  上層からの入力ストリーム
  protected PipedInputStream inputUpper;
  //  上層への出力ストリーム
  protected PipedOutputStream outputUpper;
  //  下層からの入力ストリーム
  protected PipedInputStream inputLower;
  //  下層への出力ストリーム
  protected PipedOutputStream outputLower;

  public void connect() {
    // 上層に接続される入力ストリームを開く
    inputUpper = new PipedInputStream();
    // 上層に接続される出力ストリームを開く
    outputUpper = new PipedOutputStream();
    // 下層に接続される入力ストリームを開く
    inputLower = new PipedInputStream();
    // 下層に接続される出力ストリームを開く
    outputLower = new PipedOutputStream();

    // エージェントの参照を取得して接続を行う
    AgentContext ac = getAgentContext();
    Enumeratioin agents =
      ac.getAgents(lowerAgent);
    if(agents.hasMoreElements()) {
      AgentIdentifier aid
      = (AgentIdentifier)agents.nextElement();
      ac.send(aid, "connect",
        inputLower, outputLower);
    }

    ....
  }
  public void connect(PipedInputStream in,
          PipedOutputStream out)
  throws IOExceptioin{
    try {
      // 隣接する 2 層のストリームを接続する
      inputUpper.connect(out);
      outputUpper.connect(in);
    } catch (IOException ex) {
      throw(ex);
    }
  }
}
```

図 5.5: AbstractComponent クラスのプログラムコード例

ント時にコアシステムによってランタイム上より消去される．なお，time < 0 となる設定を持つコンポーネントは永続的に存在し，時間経過による消去は行われない．

5.4　評価実験

5.4.1　基本性能評価

ミドルウェアの実装は Java（JDK1.1）互換の実行環境で動作し，オペレーティングシステムやハードウェア，ネットワークプロトコルなどのプラットフォームには依存しない．現在の実装は必ずしも最適化されたものではないが，その基本性能として 2 つのノード間のコンポーネントを移動・置換する際のコストを示す．ノード間が 100BASE-T Ethernet による有線ネットワークにより接続されている場合のコストは 61ms となり，IEEE802.11b 無線ネットワークにより接続されている場合は 64ms となる．ここで，転送対象は空のコールバックメソッドから構成された最小化されたコンポーネントであり，各ノードは Windows2000 および JDK1.1 が稼働する PC(PentiumIII-750 MHz) である．また，コンポーネントの実現に用いるモバイルエージェントシステムとして前述の AgentSpace を用いている．また，ノード間転送は TCP/IP を用いており，上述の 2 つのコストはそれぞれコンポーネントのシリアライズ処理，TCP コネクションの確立，シリアライズされたエージェントコードの転送，転送確認，エージェントコード復号化，型検証，安全性の確認処理が含まれる．エージェントの転送には，Java 言語の提供するソケットライブラリを利用しており，実行状態に加えて，エージェントを構成するコードも一括して転送している．両コストは AgentSpace システムのエージェント転送コストに対して，それぞれ 10%以下となり，ミドルウェアによるオーバヘッドは小さいといえる．なお，本論文が対象としているモバイルコンピューティング環境におけるユーザの行動計画支援においてはコンポーネントの移動・置換は頻発しないことから，十分な性能を持っているといえる．また，測定に用いたコンポーネントのサイズは 8KB となり，実際の運用時にはコンポーネントの処理に応じてサイズが増加することになるが，一般に個々のノードが処理するデータ量と比較すると十分に小さく，適切にデータを選別することによりネットワークトラフィックを軽減することが可能となる．

5.4.2　動的な構造変化による機能変更・拡張

AgentStack は，5 つの処理階層を構成する各コンポーネントをモバイルエージェントを用いて構成しているため，モバイルエージェントの移動性を利用してコン

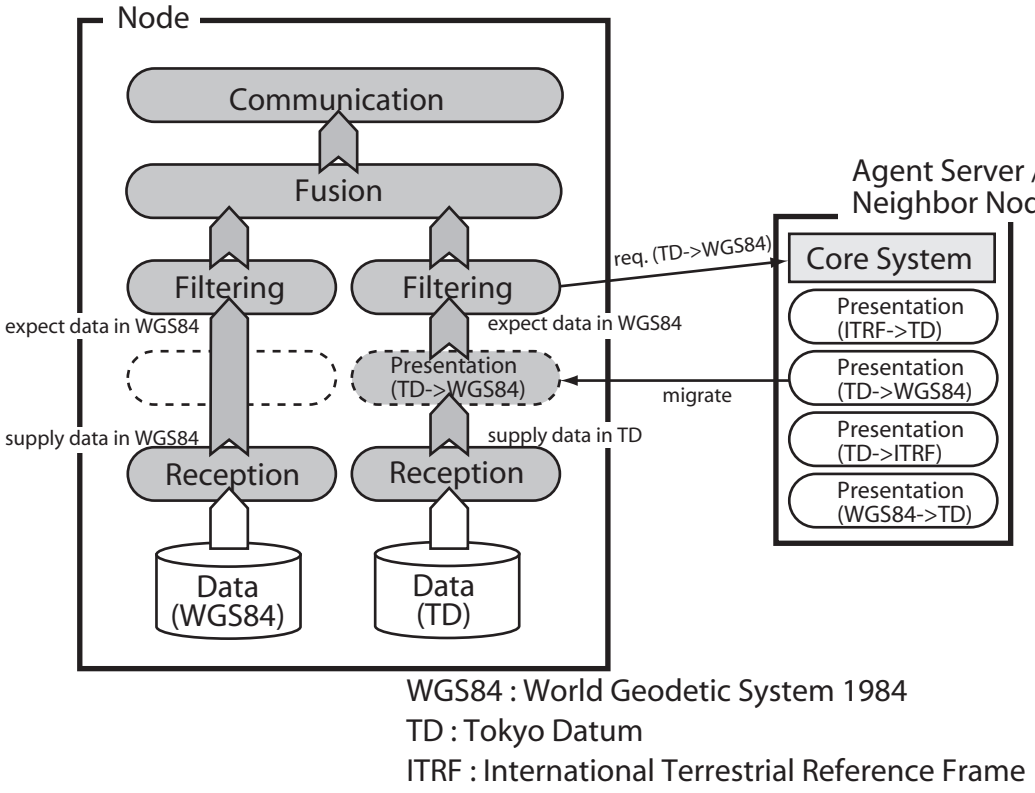

図 5.6: コンポーネントの動的置換

ポーネントを置換することにより，データ処理機能を動的に変更・拡張することが可能である．

ここでは，データリソースとして地図情報データを例として，コンポーネントの置換によってデータ処理機能が動的に変更される手順について説明する．

各ノードは図 5.6 に示すように，1 つ以上の地図情報データベースをデータリソースとして，地図情報データを参照しているとする．今，ノード内には座標系の異なる 2 つのデータリソースがあり，これらから得られる地図情報データを統合して利用することを考える．アプリケーションのプログラムが GPS などで広く利用される世界測地系（WGS84）を前提として記述されている場合，当然ミドルウェアはアプリケーションの要求に合わせて WGS84 形式でのデータを提供する必要がある．図 5.6 においては，2 つあるデータリソースのうち，1 つは WGS84 形式でデータが取得できるが，もう 1 つは日本周辺を記述するのに適した座標系として知られる日本測地系（TD）によってデータが蓄えられている．このとき，コンポーネントの置換によって，動的にデータ処理機能を変更する手順を次に示す．

1. データ選別層のコンポーネントは，自身の WGS84 形式によるデータ要求に対し，下層コンポーネントがこれとは異なる TD 形式によるデータを供給す

ることを検知する．このときの検知の様子は後述する．

2. 該当するデータ選別層コンポーネントは，隣接ノードまたはエージェントサーバに対し，TD 形式を WSG84 形式に変換する機能を持った，データ変換層コンポーネントを転送するように要求を送る．

3. 転送要求を受けた隣接ノードまたはエージェントサーバは，自身の管理下にあるコンポーネントの中に要求に合致するものがあるかを検証する．

   (a) 要求に合致するコンポーネントがある場合には，転送要求を受け付け該当コンポーネントを要求元ノードへ転送する．

   (b) 要求に合致するコンポーネントがない場合には，さらに他の隣接ノードまたはエージェントサーバに要求を転送する．

4. 要求元ノードへ転送されたデータ変換層コンポーネント（TD 形式を WSG84 形式に変換）は，移動後に上下各層のコンポーネントとの間に入出力ストリームを接続し，データ処理が開始される．

図 5.7 は，AgentStack におけるコンポーネントのプログラムコード例である．各コンポーネントは，データ処理を始める前の初期化手続きの中で、`init()` メソッドを呼び出す．まず、下層コンポーネントから得られる出力データのフォーマット情報を読み込み、自身の要求するデータフォーマットと一致しているかを確かめる．一致していた場合には，さらに上層コンポーネントに対して自身の提供するデータのフォーマットを通知する．フォーマットが不一致の場合，コアシステムに対して該当するデータフォーマットをサポートするエージェントの転送を要求する．

このとき，コンポーネント間のインタフェースが明確に定義されているため，機能の変更については変更が必要な処理階層のコンポーネントを置換するだけでよく，その際に他の階層の処理には影響を与えない．さらに，利用するデータリソースを変更することによってデータリソース制御層コンポーネントから得られる地図情報データの座標系が変わってしまった場合や，アプリケーションの要求・仕様変更により選別層コンポーネントの要求するデータの座標系が変わってしまった場合にも，同様にデータ変換層のコンポーネントを置換することにより，それより上層・下層のコンポーネントには影響を与えることなく処理機能の変更が可能である．

5.4.3　複製・転送による動的配置

AgentStack は，モバイルエージェントによるデータ処理コンポーネントによって構成されているため，モバイルエージェントの自己複製能力や自律的移動能力を利用して，遠隔ノードに対して動的にコンポーネントを配布し，必要なデータストリームを形成することが可能である．

```
public class AgentStackComponent  extends AbstractComponent {

  String indataform;
  String outdataform;

  public void init() {
    connect();
    String lowerdataform;
    lowerdataform = inputLower.readUTF();
    if(dataform.equal(lowerdataform)) {
      outputlowe.writeUTF(outdataform);
      return;
    }

    AgentContext ac = getAgentContext();
    ac.send("retrieve", dataform);

        return;
  }
}
```

図 5.7: AgentStack コンポーネントのプログラムコード例

ここでは，データベースが分散管理されている周辺情報提示システムを例にして，コンポーネントが自己の複製・転送によって遠隔ノード上に動的配置される手順について説明する．

実世界情報サービスシステムにおいては，サービスに利用される情報は集中管理であるとは限らない．個々の情報の物理的な位置情報とは別に，情報リソースとして論理的に配置され管理されている [Hohl *et al.* 99]．図 5.8 は実世界空間における物理的な存在と，情報空間における論理的な存在の関係を模式的に示している．今，実世界環境（Augmentd Area）において，検索の基準点 P と検索対象 O_i $(i = 1, 2, 3 \cdots, n)$ をシステムで扱うことを考える．実世界情報は，空間的位置を基準としてシンボル化されたモデル（Augmented Area Model）としてコンピュータで取り扱われる．それぞれの実世界情報は，情報空間（Information Spaces, Services）の観点からみると，異なるコンピュータ上で管理されるデータとして存在する．

このとき，検索範囲を段階的に拡大・縮小することによって，コンポーネントの複製・転送による動的配置がされる手順を次に示す．

1. 検索範囲が十分に狭く，登録されている検索対象が範囲内に存在しないとき，AgentStack によるコンポーネントはユーザの利用する携帯端末上にのみロードされ，GPS などのリソースから現在の座標位置を取得するためにデータ処理を行っている．

2. 検索範囲を拡大し，範囲内に検索対象が現れると，ユーザの携帯端末上のコンポーネントは，検索対象に関する情報を管理するサーバに対して，周辺情報収集用のコンポーネント群を配置する．

   (a) このときのコンポーネント配置には，第 5.4.2 項と同様に， 隣接ノードまたはエージェントサーバへコンポーネントの転送要求を行う手順が取られる．

   (b) 転送されたコンポーネント群は周辺情報の収集のためのデータ処理を開始し，ユーザの携帯端末との間でノード間通信層コンポーネントを介してデータのやり取りを行う．

3. さらに検索範囲を拡大し，範囲内に新しく検索対象が現れると，同様にコンポーネント群の配置が行われる．

   (a) このときのコンポーネント配置は，既に周辺情報の収集を行っているサーバ上のコンポーネント群によって，自己複製と複製されたコンポーネントの転送によって行われる．

   (b) O_i と O_3 のように，同じサーバ上の同じデータリソースによって検索対象の情報が管理されている場合には，コンポーネントの動的配置は必要とならない．

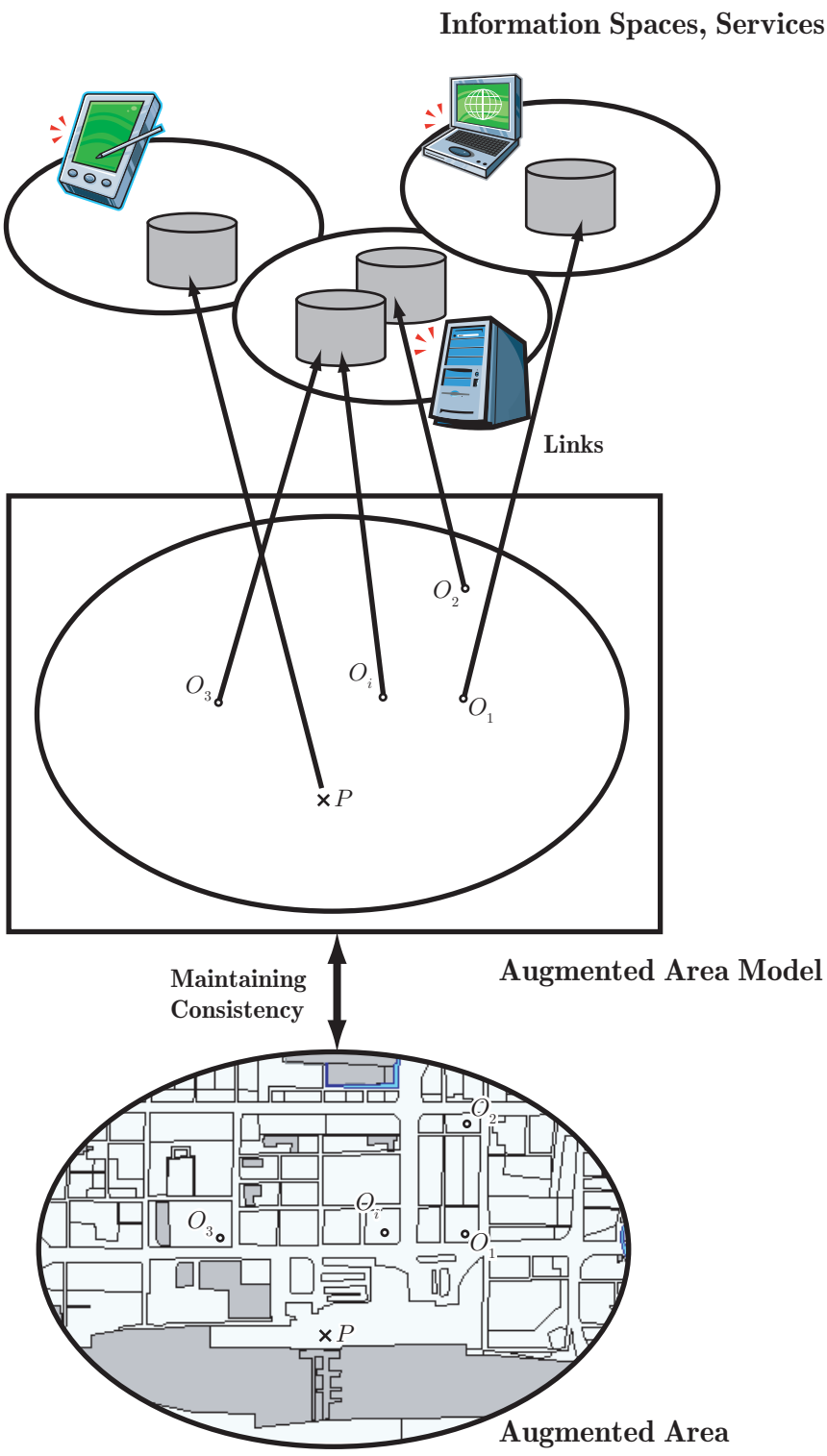

図 5.8: 実世界空間と情報空間の対応関係

(c) O_2 と O_3 のように，同じサーバ上の異なるデータリソースによって検索対象の情報が管理されている場合には，コンポーネントの複製・転送による配置ではなく，第 5.4.2 項と同様の手順によりコンポーネントの配置が行われ，サーバ上でデータ統合コンポーネントによる統合処理が行われる．

4. 検索範囲を縮小し，特定のサーバ上で取り扱っている検索対象が検索範囲内に含まれなくなると，該当サーバ上のコンポーネント群はデータ処理を停止し，サーバ上から自己を削除する．

検索範囲の拡大・縮小だけでなく，検索基準点の移動が起こった場合にも，同様の手順をとることによって，ユーザ周辺に関係するサーバのみにコンポーネント群を配置することができる．このとき，動的なコンポーネント配置を行うことで，各ノードの計算能力，記憶領域，ネットワーク通信帯域に掛かる負荷を最小化することが可能である．

5.5　まとめ

本章では，ネットワークデータ処理を対象としたミドルウェア AgentStack を提案した．これはネットワークデータの管理・運用における 5 つの基本機能（データリソース制御，データ変換，データ選別，データ統合，ノード間通信）に対応した階層型アーキテクチャとなる．そして各階層はモバイルエージェント技術を利用したプラグインコンポーネントとして実現され，階層間にはストリーム通信を基礎とする統一的インタフェースを提供する．この結果，データ処理対象となる外部環境やアプリケーション要求の変化に応じて，対応するコンポーネントを交換・拡張することができ，またその際他の階層に対する影響も最小化することができる．提案した AgentStack は特定のアプリケーションに限定したものでなく，汎用的な枠組みである．

第6章

考察および今後の課題

6.1 Baumによるモバイルユーザ向け周辺情報提示

地図の拡大・縮小に合わせた情報表示粒度

本論文では，モバイルユーザ向けの周辺情報提示システムにおいては，携帯端末の表示能力の制約を考えると表示情報の絞り込みが必要とされることを指摘した．また，ユーザの利用プロセスを考慮すると，様々な縮尺で情報を参照する必要があり，地図の拡大・縮小に合わせ，適切な粒度で情報を表示する必要があることを述べた．ここで，Baumの利用による効果を検証するために，Baumを用いた場合と用いない場合のプロトタイプシステムの実行例を図6.1に示す．

図6.1中央は表示範囲内に登録されている店舗の位置と重要度を示している．Baumを用いない場合の表示には，既存のシステムによく見られる方法として，地図の縮尺に応じて表示する情報の重要度に閾値を設ける方法を用いた．図6.1において，閾値を用いた場合には表示範囲内の表示候補のうち一定の重要度以下のものが全て非表示になる．これに対しBaumを用いた場合には，重要度の低いものの中でも基準点から遠いものは表示を省略し，同時に基準点近辺のものについては

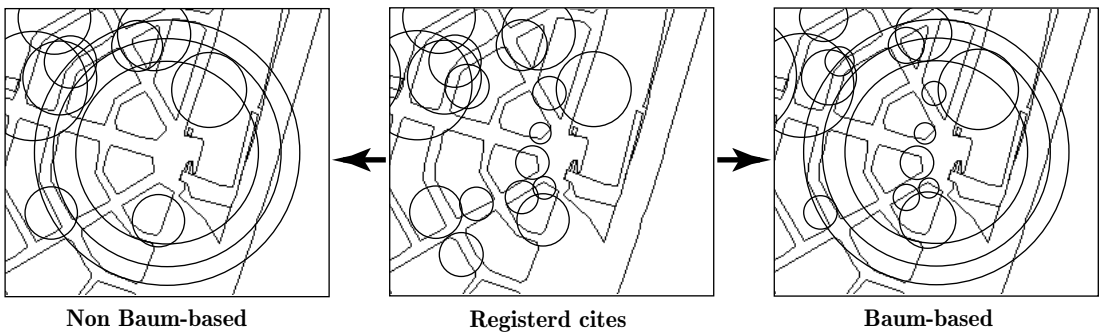

図 6.1: Baumの有無による表示への影響

表示することができる．モバイルユーザにとって，同じ重要度の候補であっても基準点からすぐ近くの候補と遠くはなれた候補では存在価値は異なると考えられるので，このような振る舞いは要求に合ったものであると考えられる．地図を利用した情報検索においては，基準点を徐々に移動させながら情報を探す行為がよく見られ，このような場合にはある程度の範囲が見渡せるように地図表示を比較的広域に保つことが有効である．閾値を用いて一律に表示を抑制してしまうと，このときに必要な情報が表示されないことが起こり得る．一方で，Baum を用いれば比較的広域な表示を行っていても，基準点近くの候補については重要度が低くても表示される．これを利用すれば，基準点を特定のルートで移動させながら周辺情報を検索するウォークスルーが可能であり，また実際にユーザが移動している際には，通り掛かった場所にある小さな店舗を近くを歩いている間だけ表示することもできる．

さらに Baum を用いることの利点として，地図の表示範囲外の情報についても扱うことができることが挙げられる．図 6.1 右において，図左上部に明らかに地図表示範囲外に中心を持つ円弧が表示されているのが分かる．図 3.4［B］および図 3.5［B］に示した通り，Baum では検索領域と検索対象それぞれに設定された領域の重なり度合いを検索対象の評価値とするため，検索対象の位置座標が地図の表示範囲に含まれなくても表示候補となるためである．これにより，ユーザに対して地図の表示外に有力な候補があることを伝えることができ，地図表示の広域化あるいは基準点の移動を促すことができる．

Baum による周辺情報提示システムを使えば，次のような手順を繰り返すことで情報検索を行うこともできる．

1. 広域表示の概観的な表示により有望な情報が集まっている地点を探索
2. 検索基準点の移動により目的の地域を表示
3. 地図を詳細表示して狭い地域に関して網羅的に情報を参照

Baum を利用することで，登録されている情報に対し，対象の重要度と対象までの距離を総合的に評価することができた．これにより，地図を広域表示して広範囲に情報を閲覧したい場合には，重要度の低い情報を間引き概観的な表示ができるようになった．一方で，狭い範囲で詳細に情報を見たい場合には，比較的重要度が低いが身近である情報を表示させるとともに，重要度の高い情報を効果的に表示することができるようになった．また，Baum を利用すれば，基準点から近くて重要度の高い候補を優先させた上で，基準点から遠いが非常に重要度の高い情報も表示したいという要求に対応することが可能であると考えられる．

オペレーションの簡便さ

一般に携帯端末は入力機能が乏しい上に，モバイルコンピューティング環境は複雑な入力を行うのには不向きであるといえる．したがって，モバイルユーザ向けの情報提示システムはできる限り簡易なオペレーションのみで利用できる必要がある．Baum による周辺情報提示システムにおいては，次の 2 つの入力でサービスを利用することができる．

- 検索の基準点
- 検索領域

このうち，検索の基準点については携帯端末上に表示されている地図の中心とみなすことができるので，地図の表示区域を移動させることで指定可能である．また，本論文で示したプロトタイプでは使用しなかったが，GPS による位置情報サービスを利用すれば，ユーザの現在地をリアルタイムに検索の基準点として利用することが可能である．

検索領域については，地図の縮尺と対応づけることができる．Baum によって表示される周辺情報は「そこまでなら行く価値がある」というニュアンスを表現しており，その検索領域の広さは表示している地図の範囲とほぼ一致していると考えることができる．

以上の入力は一般に文字の入力や複雑なメニュー操作を伴わない，簡易なオペレーションとして実現することが可能である．

重要度の動的変化

本論文で示したプロトタイプシステムにおいては，登録された店舗情報に設定された重要度は静的なものとし，重要度を表す領域は固定であった．しかし，店舗検索を例に取れば，休業日や営業時間，更に混雑時間帯など様々な要因によってその重要度は時間変化すると考えられる．検索対象の重要度を表す領域の可変対応に関しては今後の課題であると考える．

検索領域の形状

領域可変型情報検索手法 Baum においては，検索対象の評価値が検索対象と検索範囲それぞれに設定された領域の重なり度合いによって決定されるため，検索領域の形状が表示結果に与える影響が大きい．本論文では，検索領域の形状として，地形情報を考慮した領域検索モデルを定義した．地形情報の考慮により，地図平面上で検索基準点から近い対象であっても，間に建物が存在したり，広い道路や川に

より隔てられている場合には対象の評価値を低くする振る舞いを実現することができた．実世界を対象とした情報提示システムの場合，こうした地理情報の考慮はユーザの利便性の観点から非常に重要であるといえる．地形情報を考慮した領域検索モデルは，単純な円形領域モデルに比べ，ユーザの行動計画支援としての周辺情報提示に適した実験結果を得ることができた．移動ユーザが屋外で携帯端末を利用して周辺情報を得る目的に限定すれば，この検索領域モデルで妥当であると考えられる．しかし，デパートやオフィスビルの中で利用することを考えるならば，領域の形状を 3 次元に拡張する必要があると考えられる．

6.2　構造可変モバイルエージェント機構

次に，提案した構造可変モバイルエージェント機構 AgentStack が，第 5.1.2 項であげたネットワークデータを管理・運用するためのミドルウェアに対する要求を満たしているかどうかについて考察を行う．

汎用性

提案した AgentStack は，モバイルエージェントによって実現されたデータ処理コンポーネントノードのコンピュータ間移動能力により，ノード上のデータ処理プログラムを動的に変更・拡張することができる．これは，ノード上のデータリソースやアプリケーションの要求に特化したデータ処理プログラムをモバイルエージェントベースのコンポーネントの集合として構成できることを示す．構成されたデータ処理プログラムは，遠隔ノードに動的に転送することができ，またデータ処理が不要となったノードからは動的に削除することができる．AgentStack の提供する階層コンポーネントはコンポーネント間インタフェースが明確に定義されているため，ノード上の他の階層のコンポーネントに影響を与えることなく，動的に置換することが可能である．

適応性

AgentStack においては，処理対象のデータリソースやアプリケーションの要求が変化した際には，新たなデータリソースやアプリケーションの要求を満たすコンポーネントと動的に交換することで適応することができる．コンポーネントは隣接ノードあるいはエージェントサーバから要求に応じて転送される．ノード間通信層のコンポーネントにおいては，ノード間でデータを交換する際の通信プロトコルを自身で保持できるため，自己複製を行うことによって，あらかじめシステムによって提供されていない通信プロトコルも含め，任意のプロトコルでの通信が可能で

ある．

スケラビリティ

モバイルエージェントの移動性を通して，データ処理を局所化することができるので，ネットワーク上を大量のデータが流れることがない．データ選別層のコンポーネントのはたらきにより，ノード上のデータリソースから得られる生データの中から必要なデータのみを選び出して利用することができる．また，コンポーネントの動的配布によってネットワークの負荷分散を図ることができる．

独立性

AgentStack のデータ処理コンポーネントは Java 仮想機械上で動作する．したがって，オペレーティングシステムやハードウェアに独立である．また，データリソースや携帯端末に依存した処理を行うリソース制御層のコンポーネント以外は，ノード上のデータリソースや携帯端末のデバイスに対しても独立である．

以上より，提案した AgentStack は，モバイルエージェントをベースとした 5 つの処理階層による階層コンポーネントによってミドルウェアを構成することにより，想定するネットワークデータ管理・運用のためのミドルウェアに要求される事項を満足すると考えられる．

6.3　実世界におけるユーザの移動支援

第 1.1 節で述べたように，実世界におけるユーザの移動支援は，行動計画支援と経路計画支援の 2 つ観点から捉えることができる．本論文では，従来の研究では取り組みが十分ではなかった，行動計画を支援する情報提示手法について取り扱った．領域可変型情報検索手法 Baum の導入によって，モバイルユーザ向けユーザの地理的位置に合わせて周辺情報を提示することが可能となり，さらにユーザの移動に合わせて提示する情報を変化させることが可能となった．また，構造可変モバイルエージェント機構 AgentStack の導入によって，モバイルエージェントのコンピュータ間移動性を通して，ミドルウェア機能の変更・拡張が可能となり，モバイルコンピューティング環境における携帯デバイスやデータリソースの多様性に適応したデータ処理が可能となった．これらの成果は，実世界における移動ユーザの移動支援を目的とするサービスを構築する上で，行動計画支援の重要性を示唆するも

のである．行動計画の支援を目的とした本論文の成果は，既に多くの研究例があり，実用サービスも多く存在する経路計画支援のシステムと統合することによって，目的地の決定から移動までを包括的に支援することが可能な，真の移動支援システムを実現することができると期待される．

第7章

結論

本論文では，モバイルコンピューティング環境下で活動するユーザに実世界情報を提供するための，周辺情報提示手法を提案した．まず，実世界向けの汎用検索手法として，領域可変型情報検索手法 Baum を提案した．Baum は，検索対象と検索範囲それぞれに特定の幾何形状を仮想的な「領域」として設定し，互いの領域の重なり度合いによって対象の評価値を決定する手法である．次に Baum による情報検索を，モバイルユーザ向けの周辺情報提示に適用した．Baum の適用によって，ある対象に設定された重要度と，現在地からその対象までの物理的距離とを総合的に評価できることを評価実験によって示した．Baum を利用したモバイルユーザ向け周辺情報提示によって，ユーザの地理的位置に合わせ，店舗・施設のような周辺情報を提示することが可能であり，これによってユーザの行動計画を効果的に支援できると考えられる．さらに，環境の変化やアプリケーションの要求に応じて動的に構成・機能を変更することができる，構造可変モバイルエージェント機構 AgentStack を提案した．AgentStack は，システム内のデータフローをデータストリームと捉え，データ処理過程を 5 つの基本機能に分割した階層構造を持つミドルウェアとして構成される．各階層をモバイルエージェントベースのコンポーネントとして実現することで，モバイルエージェントのコンピュータ間移動性を通して，ミドルウェア機能の変更・拡張を可能とする．提案するミドルウェアを Java 言語により実装し，基本性能の評価を行った．また，動的な機能の変更・拡張および複製・転送による動的配置が可能であることを，事例を用いて示した．

モバイルユーザは物理的な移動をともなうため，動的な位置変化と許容される移動距離を考慮して適切な情報提示を行う必要がある．また，一般に利用されている位置情報サービスにおいては，地図データ規格の相違や携帯端末間の機能差が大きく，実用システムの構築には，リソースを相互利用するためのデータ処理行機構が必要となる．本研究では，領域可変型情報検索手法 Baum および構造可変モバイルエージェント機構 AgentStack を提案し，これらの要求に応える．

本研究の目的は，実世界で行動するユーザの移動支援を行うために，行動計画を立てる上で必要な周辺情報の提示手法を開発することである．周辺情報提示アプリ

ケーションを Palm OS で動作する PDA 上に実装し，ユーザの動的な位置変化と許容される移動距離に応じた周辺情報の提示が可能となることを評価実験により示した．

本論文で提案した周辺情報提示手法は，実世界情報システムの分野において有用であると考える．当該分野は，通信環境や携帯端末，位置情報技術の発展により，モバイルコンピューティング環境下で活動するユーザに実世界情報を提供するサービスとして注目されており，今後ますます重要になると考えられる．とりわけ，領域可変型情報検索手法 Baum によって，ユーザの地理的位置に合わせ，店舗・施設のような周辺情報を提示することが可能となり，ユーザの行動計画を効果的に支援することが可能であると考えられる．さらに，構造可変モバイルエージェント機構 AgentStack によって，環境の変化やアプリケーションの要求に応じてシステムの構成・機能を動的に変更・拡張することが可能となり，クライアントとなる携帯端末や位置情報を提供するデータ形式が多様な傾向にある実世界指向システムにおいて適応性に優れたシステム構築が可能となると考える．

謝辞

本研究を行う機会を与えて下さり，また研究を進めるにあたり厳しくも温かく御指導下さいました 慶應義塾大学理工学部教授 安西 祐一郎 先生 に心より感謝致します．安西先生には，お忙しい中いつも気に掛けていただき，格別の配慮を頂きました．幾度も挫折しそうになる私を励まし，温かく見守って下さいました．安西先生の薫陶を受けることができたことが，私にとっての最大の幸運だと思っています．

慶應義塾大学理工学部助教授 今井 倫太 先生 には，多くのディスカッションを通して研究内容に関して多くのアドバイスを頂きました．また，研究の進め方，論文の書き方など多くの御指導を頂きました．自らの経験に基づく今井先生の御指導は，論理的かつ非常に理解し易く，研究に行き詰まり悩んでいた私のブレイクスルーとなるものでした．心より感謝致します．

慶應義塾大学理工学部教授 寺岡 文男 先生 には，論文の細部にわたり多くの貴重な御意見を頂きました．また，何度も時間を割いて頂き，快くディスカッションに応じて頂きました．慶應義塾大学理工学部教授 櫻井 彰人 先生 には，お忙しい中，本論文の審査を御快諾頂き，有益なコメントを数多く頂きました．特に，研究内容について御理解を頂き，温かい御意見に大いに励まされました．ここに謹んで感謝致します

国立情報学研究所教授 佐藤 一郎 先生 には，モバイルエージェントシステムの研究にあたり多くの御指導を頂きました．そもそも，モバイルエージェント技術との出会いは佐藤先生の研究紹介プレゼンテーションでした．その後御縁があり，研究を御一緒させて頂き，大変貴重な経験を積ませて頂きました．私の力不足により，佐藤先生には色々と御迷惑をお掛けしましたことを大変申し訳なく思っております．また，研究を御一緒させて頂いた期間，もっと多くのことを吸収できたのではないかということが非常に悔やまれます．佐藤先生の存在なくしては，現在の研究者としての私は在り得ません．心より感謝致します．

名古屋大学教授 長尾 確 先生 には，長尾先生が日本アイ・ビー・エム東京基礎研究所に御在籍の折，学生研究員として研究を御一緒させて頂き，多くの貴重な経験をさせて頂きました．長尾先生の研究に対するひたむきさ，純粋な情熱には多大な影響を受けました．学生研究員と対等な目線で議論を交わす姿勢のお陰で，様々なことを学ぶことができました．思えば，私が研究者の道を選んだ決め手は，長

尾先生と研究を御一緒させて頂いた経験であると云えます．ここに謹んで感謝致します．

また，日本アイ・ビー・エム学生研究員として時間を共にした，大平茂輝氏，東中竜一郎氏，細谷真吾氏，福岡俊樹氏，片桐由希子さんには，長尾プロジェクトメンバーとしての研究活動を通して，多くの貴重な経験をさせて頂きました．一つのシステムを共同作業によって完成させる過程や，展示会への出品準備など，様々な苦労を共にした日々は私の大きな財産です．ここに，心より感謝致します．

安西研究室の先輩である白石陽氏には，卒業研究時より大変お世話になりました．研究内容への有益なコメントはもちろん，大学院での研究生活や様々な手続きに至るまで，実に多くの助言を頂きました．また，共に博士課程に進んだ同期である，川島英之氏，大村廉氏，宮崎崇史氏には，様々なディスカッションを通して多くの意見を頂きました．皆で集まって話をする際には，たわいない雑談である程有益な意見が飛び出したように思えるのも，良い思い出です．特に，それぞれ修了した後も機会のあるたびに，残された私に励ましの言葉を頂きました．心より感謝致します．

安西研究室の卒業生である，谷澤佳道氏，白井俊紀氏，大前寛子さん，油本宗久氏，鳴海真里子さん，広田裕氏には，研究グループの活動を通して貴重な経験をさせて頂きました．後輩と一緒に研究に取り組むことで，一人で学んだり，他人から教えを受けたりすることでは得られない，貴重なことを習得することができたように思います．頼りない先輩を慕ってくれ，大変ありがたく思っています．心より感謝致します．

そして，向井淳氏，石井健太郎氏，佐竹聡氏，大澤博隆氏をはじめとする，安西・今井研究室の現役学生の皆様には，多くの御協力と御意見を頂きました．また，元研究室秘書の永坂弘子さんには，研究室生活を様々な面から支えて頂くとともに，何度も励ましの言葉を頂きました．ここに，心より感謝致します．

そのほか，本論文は本当に多くの方々に支えられて完成することができました．ここに全ての方の御名前を挙げることは叶いませんが，謹んで感謝致します．

最後に，深い理解と様々な協力により，最も身近に研究生活を支えてくれた家族に感謝します．とりわけ，一人上京したまま長年にわたり勉学と研究の生活を続ける息子に理解を示してくれた母に心より感謝します．

2007 年 2 月
梅澤 猛

参考文献

[Arnold *et al.* 00] Ken Arnold, James Gosling, and David Holmes. *The Java Programming Language*. Addison-Wesley Pub, 3 edition, 2000.

[Ashbrook *et al.* 02] Daniel Ashbrook, and Thad Starner. Learning significant locations and predicting user movement with GPS. In *Proceedings of 6th IEEE International Symposium on Wearable Computers*, 2002.

[Author 97] Author. Programming Languages for Mobile Code. *IEEE Personal Communication Magazine*, Vol. 29, No. 3, pp. 213–239, 1997.

[Baus *et al.* 02] Jörg Baus, Antonio Krüger, and Wolfgang Wahlster. A Resource-Adaptive Mobile Navigation System. In *Proceedings of the International Conference on Intelligent User Interfaces (IUI02)*, pp. 15–22, 2002.

[Boll *et al.* 04] Susanne Boll, Martin Breuning, Birgitta Köning-Ries, Florian Matthes, and Thomas Schwarz. Mobile Application development – now and then Towards a Handbook for User-Centered Mobile Application Design. Vision Paper, 2004.

[Broch *et al.* 98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*, pp. 85–97, 1998.

[Brooks *et al.* 98] Richard R. Brooks, and S. S. Iyengar. *Multi-Sensor Fusion: fundamentals and applications with software*. Prentice-Hall, Inc., 1998.

[Chess *et al.* 95] David Chess, Benjamin Grosof, Colin Harrison, David Levine, Colin Parris, and Gene Tsudik. Itinerant Agents for Mobile Computing. *IEEE Personal Communication Magazine*, Vol. 2, No. 5, pp. 34–49, 1995.

[Cheverst *et al.* 00] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Developing a Context-aware Electronic Tourist Guide:

Some Issues and Experiences. In *Proceedings of the ACM Conference on Human Factors in Computing Systems, 2000*, pp. 17–24, 2000.

[Fuggetta *et al.* 98] Alfonso Fuggetta, Gian Pietro Picco, and Giovanni Vigna. Understanding Code Mobility. *IEEE Transactions on Software Engineering*, Vol. 24, No. 5, pp. 342–361, 1998.

[Google Map] Google Map. http://maps.google.co.jp/.

[Hohl *et al.* 99] Fritz Hohl, Uwe Kubach, Alexander Leonhardi, Kurt Rothermel, and Markus Schwehm. Nexus – An Open Global Infrastructure for Spatial-Aware Applications. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, pp. 249–255, New York, NY, USA, 1999. ACM Press.

[Intanagonwiwat *et al.* 00] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pp. 56–67, 2000.

[Its-mo Guide] Its-mo Guide. http://www.its-mo.com/.

[Kingston *et al.* 00] Richard Kingston, Steve Carver, Andrew Evans, and Ian Turton. Web-Based Public Participation Geographical Information Systems: An Aid To Local Environmental Decision-Making. *Computers, Environment and Urban Systems*, Vol. 24, No. 2, pp. 109–125, 2000.

[Lange *et al.* 98] Danny B. Lange, and Mitsuru Oshima. *Programming and Deploying Java Mobile Agents with Aglets*. Addison-Wesley Pub, 1998.

[Li *et al.* 00] Qun Li, and Daniela Rus. Sending Messages to Mobile Users in Disconnected Ad-hoc Wireless Networks. In *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pp. 44–55, 2000.

[Malaka *et al.* 00] Rainer Malaka, and Alexander Zipf. DEEP MAP – Challenging IT Research in the Framework of a Tourist Information System. In Daniel R. Fesenmaier, Stefan Klein, and Dimitrios Buhalis, editors, *Information and Communication Technologies in Tourism 2000*, Springer Computer Science, pp. 15–27. SpringerWienNewYork, 2000.

[MapFan Web] MapFan Web. http://www.mapfan.com/.

[Mapion] Mapion. http://www.mapion.co.jp/.

[MCPC] MCPC. モバイルコンピューティング推進コンソーシアム. http://www.mcpc-jp.org/.

[Messer *et al.* 02] Alan Messer, Ira Greenberg, Philippe Bernadat, Dejan Milojicic, Deqing Chen, T.J. Giuli, and Xiaohui Gu. Towards a Distributed Platform for Resource-Constrained Devices. In *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems (ICDCS'2002)*, pp. 43–56, 2002.

[Michahelles *et al.* 02] Florian Michahelles, Michael Samulowitz, and Bernt Schiele. Detecting Context in Distributed Sensor Networks by Using Smart Context-Aware Packets. In *Proceedings of the International Conference on Architecture of Computing Systems 2002 (ARCS '02)*, pp. 34–50, 2002.

[Nelson 99] Jeff Nelson. *Programming Mobile Objects with Java*. Wiley, 1999.

[Perkins 97] Charles E. Perkins. Ad Hoc On Demand Distance Vector (AODV) Routing. IETF MANET, Internet Draft, 1997.

[Qi *et al.* 01] Hairong Qi, S. S. Iyengar, and Krishnendu Chakrabarty. Distributed Multi-Resolution Data Integration Using Mobile Agents. In *Proceedings of the IEEE Aerospace Conference*, Vol. 3, pp. 1133–1141, 2001.

[Ramanathan *et al.* 00] Ram Ramanathan, and Regina Hain. Topology Control of Multihop Wireless Networks Using Transmit Power Adjustment. In *Proceedings of the 19th Annual Joint Conference of IEEE Computer and Communications Societies (INFOCOM 2000)*, pp. 404–413, 2000.

[Recursion Software, Inc.] Recursion Software, Inc. Voyager Java Development Platform. http://www.recursionsw.com/voyager.htm.

[Sato 00] Ichiro Sato. MobileSpaces: A Framework for Building Adaptive Distributed Applications Using a Hierarchical Mobile Agent System. In *Proceedings of the 20th IEEE International Conference on Distributed Computing Systems (ICDCS'00)*, pp. 161–168, 2000.

[Sato 01] Ichiro Sato. Network Processing of Mobile Agents, by Mobile Agents, for Mobile Agents. Workshop on Mobile Agents for Telecommunication Applications (MATA '2001), LNCS, pp. 81–92. Springer, 2001.

[Satyanarayanan 96] Mahadev Satyanarayanan. Fundamental Challenges in Mobile Computing. In *Proceedings of the 15th ACM Symposium on Principles of Distributed Computing (PODC '96)*, pp. 1–7, 1996.

[Strasser *et al.* 96] Markus Strasser, Joachim Baumann, and Fritz Hohl. Mole: A Java Based Mobile Agent System. In *Proceedings of the 10th European Conference on Object-Oriented Programming (ECOOP '96)*, 1996.

[Sun Microsystems, Inc.] Sun Microsystems, Inc. Mobile Information Device Profile (MIDP). http://java.sun.com/products/midp/.

[Waldo 99] Jim Waldo. The Jini Architecture for Network-Centric Computing. *Communications of the ACM*, Vol. 42, No. 7, pp. 76–82, 1999.

[Warneke *et al.* 01] Brett Warneke, Matt Last, Brian Liebowitz, and Kristofer S. J. Pister. Smart dust: Communicating with a Cubic-millimeter Computer. *IEEE Computer*, Vol. 45, No. 1, pp. 44–51, January 2001.

[阿部 他 04] 阿部昭博, 佐々木辰徳, 小田島直樹. 位置情報を用いて地域コミュニティ活動を支援するグループウェアの開発と運用評価. 情報処理学会論文誌, Vol. 45, No. 1, pp. 155–163, 2004.

[阿部 他 05] 阿部昭博, 狩野徹, 大信田康統, 小田島直樹, 宮井久男. 住民参加型アプローチによるユニバーサルデザイン活動支援システムの開発. 情報処理学会論文誌, Vol. 46, No. 3, pp. 753–764, 2005.

[国土地理院] 国土地理院. http://www.gsi.go.jp/.

[佐藤 98] 佐藤一郎. AgentSpace：モバイルエージェントシステム. In *Workshop on Multi-Agent and Cooperative Computation 98 (MACC98)*, 1998.

[佐藤 99] 佐藤一郎. モバイルエージェントの動向. 人工知能学会論文誌, Vol. 14, No. 4, pp. 598–605, 1999.

[山本 他 06] 山本浩司, 片上大輔, 新田克己, 相場亮, 桑田仁. 地図上の推薦情報システムにおける投稿情報の信頼度. 人工知能学会論文誌, Vol. 21, No. 3, pp. 276–285, 2006.

[中西 他 05] 中西健一, 高汐一紀, 徳田英幸. 粒度の動的変更による位置匿名性についての考察. 情報処理学会論文誌, Vol. 46, No. 9, pp. 2260–2268, 2005.

[武井 他 04] 武井一朗, 佐藤潤一, 谷口幸治, 渡辺成光, 山口孝雄. ユーザ要求の多様性を加味した位置適応型移動体向け放送方式の性能評価. マルチメディア，分散，協調とモバイル（DICOMO2004）シンポジウム論文集, pp. 213–216, 2004.

[矢入 04] 矢入（江口）郁子. 歩行空間のバリアフリー化を目指すユビキタス移動支援システム. 情報処理学会会誌, Vol. 45, No. 9, pp. 912–917, 2004.

[矢入 他 05] 矢入（江口）郁子, 奈良博之, 猪木誠二. 歩行者のアクセシビリティ向上のための GIS と地域コミュニティによる運用の提案. ヒューマンインタフェース学会論文誌, Vol. 7, No. 4, pp. 463–475, 2005.

論文目録

【 主論文に関する公刊論文 】

1. 梅澤 猛，今井 倫太，安西 祐一郎，“Baum を用いたモバイルユーザ向け周辺情報提示アプリケーション”，情報処理学会論文誌，Vol.47，No.7，pp.2117–2124 (2006 年 7 月).

2. 梅澤 猛，今井 倫太，安西 祐一郎，“領域可変型検索手法 Baum によるモバイルユーザのための周辺情報提示”，情報処理学会論文誌，Vol.46，No.12，pp.3008–3016 (2005 年 12 月).

3. 梅澤 猛，佐藤 一郎，安西 祐一郎，“モバイルエージェントを用いたセンサネットワーク向けフレームワーク”，情報処理学会論文誌，Vol.44，No.3，pp.779–788 (2003 年 3 月).

【 国際会議発表 】

4. Takeshi Umezawa, Ichiro Satoh, Yuichiro Anzai, “A Mobile Agent-based Framework for Configurable Sensor Networks”, *The 4th International Workshop on Mobile Agents for Telecommunication Applications (MATA'2002)*, LNCS, Vol.2521, pp.128–140 (October 2002).

【 国内学会発表 】

5. 梅澤 猛，今井 倫太，安西 祐一郎，“領域可変型検索手法 Baum によるモバイルユーザのための周辺情報提示”，マルチメディア，分散，協調とモバイル (DICOMO2005) シンポジウム論文集，pp.625–628 (2005 年 7 月).（優秀論文賞受賞）

6. 梅澤 猛，今井 倫太，安西 祐一郎，“領域可変型情報検索システム Baum”，マルチメディア，分散，協調とモバイル (DICOMO2004) シンポジウム論文集，

pp.161–164 (2004 年 7 月).

7. 谷澤 佳道，梅澤 猛，佐藤 一郎，安西 祐一郎，“モバイルエージェントによる移動ユーザへのサービス実現”，第 60 回情報処理学会全国大会，No.3, pp.521–522 (2000 年 3 月).

8. 梅澤 猛，佐藤 一郎，安西 祐一郎，“移動エージェントによるユーザ追跡型サービス”，Workshop on Multi-Agent and Cooperative Computation 99 (MACC99)，Online Proceedings (1999 年 11 月)

【 同一著者による他の研究成果 】

9. 広田 裕，川島 英之，佐竹 聡，梅澤 猛，今井 倫太，“セマンティック・センサネットワークの実現に向けた実世界指向メタデータ管理システム MeT の設計”，合同エージェントワークショップ&シンポジウム 2005（JAWS2005）(2005 年 11 月).

10. 梅澤 猛，安西 祐一郎，“センサ情報を扱う移動エージェントシステムにおける ORB 機構”，日本ソフトウェア科学会第 17 回大会，CD-ROM 版 (2000 年 9 月).

11. 梅澤 猛，白石 陽，安西 祐一郎，“センサネットワーク利用のためのモバイルエージェントプラットフォームの設計と実装”，日本ソフトウェア科学会第 16 回大会，pp.417–420 (1999 年 9 月).