修士論文　　2003年度 (平成15年度)

コンテクストアウェアアプリケーションのための対話的な行動履歴解析ツールの構築

慶應義塾大学大学院 政策・メディア研究科

伊藤 昌毅

目 次

図 目 次

表 目 次

第1章　序論

1.1　研究背景

ユビキタスコンピューティングとコンテクストアウェア

1990 年代初頭にマーク・ワイザーによって提唱されたユビキタスコンピューティング [25] は，コンピュータの高速化や小型化，さまざまなセンサの普及により広くそのアイディアが受け入れられるようになっている．ユビキタスコンピューティングの目指す，遍在する計算機がユーザの活動を意識させることなく補助する環境を実現するためには，コンピュータ内にユーザの行動や意図が表現されたユーザモデルを表現する必要がある．コンピュータに，誰が，どのような目的でどのような行動をとっているのか，これからとろうとするのかということが表現されることでコンピュータによる無意識的な活動の補助が可能になる．

現在，マウスやキーボードを用いたコンピュータシステムとの直接的な対話によって取得できるユーザの行動や意図に加えて，さまざまなセンサ情報や，環境やユーザの属性などの既知の情報を解析し，明示されないユーザの行動や意図をコンピュータで認識する研究が広く行われている．このようにして得られた情報を利用することで，より広範なユーザの行動や意図が表現された，多様なサービスの実現の基礎となるユーザモデルが構築できる．ユーザモデルに表現された，ユーザの行動や意図はコンテクストと呼ばれ，コンテクストを利用しよりユーザに適応的に振る舞うシステムを一般にコンテクストアウェアなシステム [5] と呼ぶ．

コンテクストアウェアなシステムの中でも，ユーザの所在地からユーザのコンテクストを類推するシステムは古くから研究，開発されており，特にロケーションアウェアなシステムとも呼ばれる．センサを用いて位置を取得するために，無線や超音波，GPS，PHS や FM ラジオなど様々なデバイスや手法が考案されている．また，カーナビゲーションや GPS を備えた携帯電話の登場などで，位置に応じた情報配信やナビゲーションなどといったロケーションアウェアなシステムがすでに実現している．

こうした現行のシステムでは，一般にユーザの意図と場所とが一対一にマッピングされ，特定の場所に出向いたという情報からその場所に設定されたユーザの意図を読み取る．ロケーションアウェアを実現する研究では，一般にユーザの行動は観光や買い物，映画視聴など特定の目的に限定して設定されており，こうした条件下において有効なユーザの意図の類推が実現されている．そして，それを利用した情報提供システムやサービスの再構成システムなどが実現されている．

限定された状況ではユーザの意図はほぼ場所と対応づけられるが，より汎用的なシ

ステムのためには，単に現在地点からだけでなく，より広い情報を利用したより広範な意図を読み取るシステムが必要とされている．

1.2 研究目的

本研究の目的は，ユーザの移動軌跡や写真撮影の記録，買い物記録などの行動履歴を統合的に扱い，さまざまな行動履歴解析手法の構築を可能にすることで，行動履歴を活用した多様なコンテクストアウェアアプリケーションの構築を実現するプラットフォームを構築することである．GPS やカメラを搭載した携帯電話や鉄道料金の支払いに利用する IC カードなどの普及で，我々の行動を電子化した形で取得できる機会は増加している．また，GPS などで取得した行動情報からより高次元の意味のある情報を抽出する研究も数多く行われている．しかし，単一のアプリケーションを想定した行動履歴の解析手法の洗練だけでは，多様化する行動履歴の取得機会を生かしきれないし，また多様なアプリケーションを構築することもできない．

本研究では，さまざまな行動履歴を入力し，行動履歴の解析手法の開発者が直感的な手法でさまざまなアプリケーションに活用できる多様な解析手法を編み出せる行動履歴解析ツールを構築する．行動履歴解析ツールはまたコンテクストアウェアアプリケーションを動作させるプラットフォームとしても機能し，アプリケーションの容易な開発や配布を実現する．

1.3 本論文の構成

本論文の構成は，次のとおりである．まず第 2 章でユビキタスコンピューティング環境における行動履歴の取得について論じ，行動履歴の多様性やアプリケーションの多様性に対処した行動履歴の解析が必要であることを述べる．次に第 3 章において，提示された問題に対して新しい行動履歴解析ツールである MUSE システムを提案し，その機能や利用シナリオを示す．第 4 章では関連する研究を紹介しながら本研究を位置づけ，第 5 章において提案した MUSE システムの設計と実装について述べる．第 6 章において実装したシステムの評価を行い，最後に第 7 章において論文をまとめる．

第2章　行動履歴の取得と活用

本章では，ユビキタスコンピューティング環境において実現する行動履歴の取得について論じ，そうした環境において取得された行動履歴を扱ってコンテクストアウェアなアプリケーションを構築する際の問題点を明らかにする．

2.1 行動履歴の取得とその利用

Abowd の論じている [2] ように，環境にさまざまなセンサや計算機能が埋め込まれているユビキタスコンピューティング環境では，日常生活の情報の自動取得や記録が容易になり，記録した情報を活用することで新しいアプリケーションの可能性が広がる．本研究では，このように環境に存在するセンサや携帯デバイスなどで取得され蓄積された行動履歴を活用しアプリケーションを構築する手法を検討する．

デバイスの小型化やセンサ技術の発展により，日常生活においてすらかつてないほどの大量の行動記録を電子化した形で取得，蓄積できるようになった．特に携帯電話の進歩は，多くの人が日常から携帯するデバイスへの汎用的な計算機能や通信機能，GPS などのセンサやデジタルカメラ，買い物における決済機能などの実装を実現した．また，携帯電話と IC カードの統合も計画されており，公共交通機関への乗車が携帯電話で実現できる日も遠くない．携帯電話のこうした機能と，携帯電話が備えるストレージの大容量化と併せて考えると，今後携帯電話に GPS などで取得した行動履歴や買い物の情報，写真撮影の記録などさまざまな情報が蓄積されるような想定は現実的である．今後，これまでには想定できなかった多様で高密度な行動履歴を取り扱うアプリケーションが登場するだろう．

2.2 行動履歴の活用における問題

ユビキタスコンピューティング環境におけるシステムにおいて，行動履歴を活用する際の問題を述べる．

なお，本論文において行動履歴とは，人の，観察可能な行為を記述した情報の蓄積を意味する．具体的には移動や食事，買い物などを行動と呼び，思索や夢などは含まない．これらの行動を時刻とともに記録し，時系列に並べたものを行動履歴と呼ぶが，特に本論文においては，主にコンピュータやセンサを利用して自動的に取得され，電子的に表現された行動履歴を取り扱う．

2.2.1 行動履歴の多様化

センサを備えた電子機器を日常的に携帯し，また様々な日常の活動がそうした電子機器を利用して行われる現在，取得できる行動履歴は多様化している．GPS などの位置情報センサから取得できる移動軌跡だけでなく，FeliCa[21] のような IC カードを利用した鉄道やバスの利用履歴，デジタルカメラによる写真撮影履歴，買い物中の電子決済履歴など，様々な情報が行動履歴として取得できる．

ユビキタスコンピューティング環境の発達により電子的に取得できる行動履歴は今後もますます増加することが予想される．また行動履歴の増加や多様化により，新しい行動履歴の解析方法が登場することも予想される．そのため，特定の情報形式に依存しない，多様な行動履歴に対応するアーキテクチャが必要となる．

2.2.2 アプリケーションの多様化

通信インフラや携帯機器の進歩など，人々が計算機器により提供されるサービスを受ける機会は今後より増大し，またそのサービスの種類もより多様化することが考えられる．行動履歴や，そこから抽出された情報はこうした様々なサービスをより個々のユーザに適した形で提供するために有効な情報であるが，それぞれのサービスの形態により，必要となる情報の抽出方法は異なる．たとえば，道案内サービスに必要な情報はユーザが好んで利用する交通機関や歩く早さなどであろうが，観光ガイドにおいてはなるべく通常利用しない交通機関を提示したほうがいいかもしれない．また，取得できる行動履歴の多様化もこうしたアプリケーションの多様化をもたらすだろう．そのため，アプリケーションごと異なる情報の要求に対して，さまざまな行動履歴の解析手法を構築する必要がある．

2.3 本章のまとめ

本章では，ユビキタスコンピューティング環境において行動履歴を活用する際の問題として，行動履歴の多様化とアプリケーションの多様化を挙げた．

次章では，本章で挙げた問題に対して新しい行動履歴解析ツールを提案する．

第3章　対話的な行動履歴解析ツール

本章では，本論文で述べる行動履歴解析ツールの概要を説明し，解析ツールを利用したアプリケーションの構築例について述べる．本章で述べる行動履歴解析ツールは，前章における行動履歴に関する問題を踏まえたものであり，多様な履歴情報の入力と解析手法の柔軟な開発とを実現する．

3.1 行動履歴解析ツールの概要

本研究では 2 章において定義した問題を踏まえ，新しい行動履歴解析ツールである MUSE を設計，実装する．MUSE は地理座標に関連づけられた行動履歴を取り扱い，行動履歴の統合的な管理や，様々な解析手法の適用による新しい情報の抽出を行う．MUSE はまた，こうした情報を利用するアプリケーションを構築する際のプラットフォームとして動作し，MUSE によって解釈をされた行動履歴を視覚化するアプリケーションや，コンテクストアウェアな分散システムを MUSE を用いて構築することが出来る．MUSE の特徴は，行動履歴の解析手法を，対話的なインタフェースを通じて柔軟に変更できることである．

3.1.1 MUSE の機能

MUSE に求められる機能の詳細を述べる．

容易な解析手法の開発が可能なアーキテクチャ

行動履歴から得られる情報は様々であるが，アプリケーションによって必要な情報が異なり，各アプリケーションに対応した解析手法をそれぞれ開発する必要がある．

MUSE においては，解析手法を部品に分け，複数の部品の組み合わせによって最適なの解析手法を定義する．解析に用いる部品は，行動情報の入力部と出力部を持ち，入力された行動情報に解釈を加えて再び行動情報として出力する．開発者は，すでに用意されている部品を組み合わせたり，足りない機能だけを新たに部品として実装することで，容易にアプリケーションに適した解析手法が開発できる．また，MUSE は部品同士の組み合わせ状況を視覚化し，マウス操作でその組み合わせを容易に変更できるビジュアルプログラミングインタフェースを提供し，システム開発を容易にする．

任意の座標系による入出力

MUSE では，取得された様々な種類の行動履歴を取り扱う必要がある．行動履歴は，単一の行動内容と位置情報，時刻情報の組み合わせを最小粒度とし，この情報の集合として記述される．行動履歴の種類により，位置情報の表現形式は異なる．緯度，経度による座標系で表現された位置情報のほか，地名や施設名といった，空間につけられた文字列で表現された位置情報，また，時刻情報しか含まず，他の行動履歴と照合することで位置情報が得られる情報も考えられる．こうした様々な座標系で表現された行動履歴を入力することが必要である．また，MUSE が出力する情報においても，アプリケーションの種類により様々な座標系が要求される．

MUSE の設計に当たっては，一般に Geocoding と呼ばれる，様々な座標系を地理座標系に変換する機能を備えることで，内部的には単一の地理座標系に変換し，任意の座標系での入出力を可能にする．

Push，Pull 両用のアーキテクチャ

MUSE は，それをプラットフォームとして開発されるアプリケーションの種類によって，ストレージに保存されている行動履歴の解析にもセンサなどから動的に送られてくる行動履歴の解析のどちらにも適応できる必要がある．前者においては，本システム上に構築されたアプリケーションによって解析結果の取得要求がなされ，それに対して必要なデータが返される．このようなシステムの振る舞いを，Pull 型の情報配信と定義する．一方後者においては，センサなどから新しい情報が追加されるたびに，解析結果を利用しているアプリケーションに対してデータの変更や新しい結果を通知する必要がある．このような振る舞いを Push 型の情報配信と定義する．本システムの備える行動履歴の解析機能は，本システムは Push，Pull 両方の情報配信に対応する必要がある．

3.2 MUSE の利用シナリオ

本節では，MUSE を用いた開発手法と，アプリケーション例を示す．本システムは，単独の計算機上で動作するアプリケーションの開発にも，分散システムとして構築されたコンテクストや位置適応的なシステムの構築にも用いることができる．それぞれの状況におけるシナリオは以下のとおりである．

3.2.1 単独使用の場合

システムを単体の計算機上で動作させ利用する例を示す．開発者は，行動履歴入力部として計算機に接続されたセンサから情報を読み出すモジュールを開発し，解釈結果出力部として，アプリケーションのインタフェースを構築する．また，行動履歴解析部において，既存の解析部品を組み合わせたり新しい解析部品を開発したりして必要な解析手法を開発する．これらの組み合わせをファイルに書き出し，アプリケーションとして配布する．ユーザは，本システムを導入した後，配布されているアプリケーションを読み込ませ，行動履歴を入力することで開発者の意図した解析手法に基づき，行動履歴から抽出された情報を得られる．以下に，このようなアプリケーションの例を示す．

旅行記録

旅行記録のように，一定期間にわたるあらゆる行動を記録し，後から追体験するようなアプリケーションを構築するプラットフォームとして，本システムが利用できる．GPS で取得した旅行の軌跡や旅行中の写真やビデオ映像，また立ち寄った観光地を紹介する URL などはすべて本システムに位置情報を伴って格納される．ユーザは，本システムを操作しながら旅行情報を様々な視点から閲覧することができる．地図に軌跡

や写真を重ねて表示するだけでなく，滞在時間や撮影写真数などから興味をもって観光した場所を強調して表示する地図や，軌跡情報から割り出した移動時間を反映した時間地図といった，旅行に行くことで得られた体験が強調された情報表示が可能になる．また，観光ガイドの情報も入力することで，まだ観光していない場所のみを抽出した地図を出力すれば，次回以降の計画にも有効であろう．このように，旅行記録から様々な情報閲覧を実現するアプリケーションが本システムを利用することで構築できる．

筆者らはすでに文献 [30] においてここで述べている強調表示機能を備えた行動記録閲覧システムを構築しており，図 3.1 にその利用例を示す．こうしたアプリケーションの構築においても MUSE は有効である．

図 3.1: 旅行記録アプリケーション

街頭情報端末

商店街などを想定し，さまざまな店を回遊しながら買い物をするような場面において，Smart Furniture[14] のような街頭にある情報端末がユーザに適した情報を提示するアプリケーションを想定する．こうしたアプリケーションは，ユーザがすでに済ませた買い物や，今まで回った店，長時間滞在した店といった行動履歴から買い物の傾向や組み合わせて買う商品などを予測し，現在の位置情報や向かっている方向といった情報にこうした情報を加味した情報提供を行う．図 3.2 に，こうした使用例を示す．こうしたシステムの構築には，位置情報や買い物履歴など多種多様な行動履歴が必要であり，またその解析手法も複雑であるだけでなく，各商店街や場所に特化した汎用性の低いものとなると考えられる．また，店舗の変化や季節の変化などで，解析手法も変化することが考えられる．本システムを利用して街頭情報端末を構築することで，こうした問題に柔軟に対処するシステムの構築が実現する．

図 3.2: 街頭情報端末における利用例

3.2.2 解析結果を転送する分散システム

本シナリオでは，主に分散システムとして構築された位置適応型サービスを想定し，これらのシステムがよりユーザに適応的な動作をするために本システムで解析された情報を利用する．開発者は，本システムを利用して，目的の位置適応サービスをよりユーザに適応的にするために必要な行動履歴の解析手法を開発しユーザに提供する．ユーザは，提供された解析手法を用いて自分の行動履歴を解析し，その結果を位置適応型サービスに送信する．位置適応型サービスは，行動履歴の解析結果を用いて，より適応的な動作を実現する．

休憩場所ナビゲーター

休憩場所ナビゲータは，携帯端末を利用した，喫茶店などの近隣の休憩場所を検索し案内するサービスである．休憩に適した場所やその検索条件は各個人によって異なり，利用者に役に立つ情報を提供するのは難しい．図 3.3 に示した，MUSE を利用して構築した休憩場所ナビゲータでは，ユーザは携帯電話などに蓄積された日常の行動履歴を本システムに登録する．MUSE は，ユーザの日常の行動パタンなどから優先的に案内すべき場所を決める．また，休憩場所におけるホットスポット利用の希望など，そのほかの検索条件も登録する．こうして出力された，行動履歴に基づく休憩場所検索パラメータを休憩場所ナビゲータに登録することで，利用者の希望を反映した休憩場所の案内が実現する．

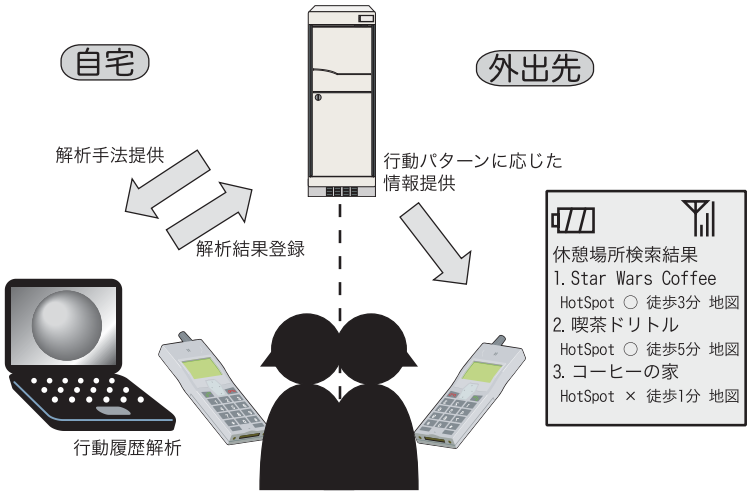

図 3.3: 解析結果を転送する分散システムの例

3.2.3 解析手法を転送する分散システム

3.2.2 節と同様の分散システムにおいて，行動履歴の解析時により短期間の行動履歴から情報抽出を行ったり，直近の行動軌跡が必要な場合，解析結果を事前に登録するのではなく，ユーザが未編集の行動履歴を直接システムに提示し，システム内で行動履歴の解析を行い情報提示を行う必要がある場合がある．こうしたシステムを構築する場合，システム開発者は本システムを用いて解析手法を開発した後，手法それ自体を位置適応型サービスなどのサービスに組み込む．図 3.4 にこうしたシステムの概要を示す．

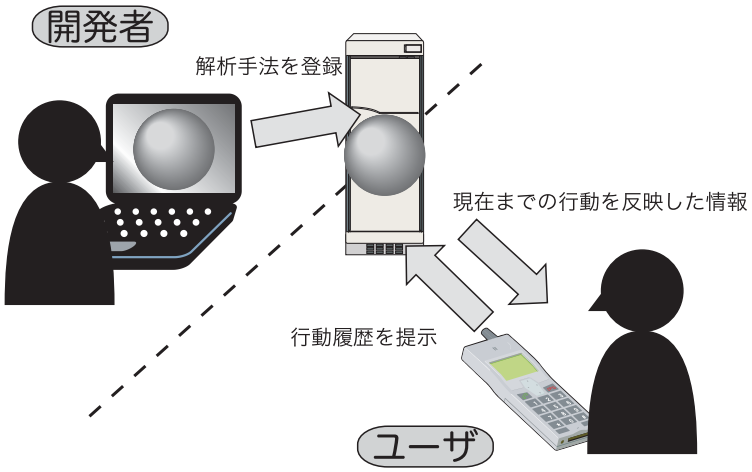

図 3.4: 解析手法をを転送する分散システムの例

3.3 本章のまとめ

本章では，2 章で定義した問題を踏まえ，対話的に解析手法の開発が可能な行動履歴解析ツールの MUSE を提案した．MUSE は，さまざまアプリケーションに対応できるよう多様な解析手法が容易に開発できる．また，さまざまな座標系で記述された行動履歴の入力を実現し，ユビキタスコンピューティング環境により取得が可能になる多様な行動履歴への対応を実現する．本章ではまた，MUSE を用いたアプリケーション開発の手順を示し，アプリケーション例として旅行記録や街頭情報端末，休憩場所ナビゲータなどを示した．

次章では，MUSE システムの関連研究について述べる．

第4章　関連研究

本章では，本研究の関連研究について述べる．本論文において構築するシステムと直接対比できるシステムはないが，本研究が前提としている研究や，本研究と一部で問題意識が重なっている研究は多数存在する．こうした研究を紹介しながら，本研究を位置づける．

4.1 位置情報の蓄積と分析

位置情報はユビキタスコンピューティングの研究の早い段階から，コンテクストを取得するための重要な情報だと考えられており，Want らによる Active Badge[24] システムなど様々なシステムが作られてきた [13]．しかし Active Badge の運用では，労働管理の強化という心配から位置情報の記録は 1 時間に限定し，永続的な保存は行わないという制約が設けられていた．

その後の研究では，位置情報を蓄積し，有効な情報を抽出しようという試みが行われるようになる．以下では，これらの研究を利用しているロケーションモデルと想定している使用者という観点から比較し論じる．

ロケーションモデル

Domnitcheva[7] や Jiang[15] の論じているように，コンピュータ上で位置を表現する手法は大きく Topological Model と Metric Model に分類できる．Topological Model は，構造や利用形態などに基づいた識別子を空間につけ，識別子名や識別子同士の構造で位置を表現する手法であり県名や市町村名，番地などからなる住所がその例として挙げられる．また，Metric Model は，空間に計量的な座標系を導入し，数値によって位置を表現する手法であり，緯度経度で表現された位置情報がその例である．Topological Model で記述された位置情報は大概場所の用途などと関連付けられた識別子が付けられており，空間が限定されている場合には Topological Model を用いたほうが容易にシステムを構築できる．大規模な空間や任意の空間における位置情報を取り扱ったり，場所同士の距離などを測定する際には，Metric Model のほうが優れているが，別に Metric Model から Topological Model へ変換する仕組みが必要となることもある．

行動履歴分析と未来予測

行動履歴の蓄積と分析によって，過去の特定の期間における行動を具体的に明らかにしてユーザや他のコンピュータシステムへ有益な情報として提供しようというシステムを，行動履歴分析型のシステムと名づける．一方，過去の行動から，現在の行動の意図を読み解いたり典型的な行動パターンなどを抽出することで，未来を予測するシステムを未来予測型のシステムと名づける．前者は，旅行における自動化された行動記録のような，より少ない手間でより大量で有益な情報を記録し，ユーザが閲覧するようなシステムが考えられる．後者は，さまざまなシステムの適応的な動作に利用できる．行動中のユーザに有益な情報を提供するだけでなく，ファイルのキャッシングや，交通量を反映した交通規制などにも応用できる．

想定している利用者数

単一の利用者の行動履歴から情報を抽出することからは，個人の経験を十分に反映した情報が得られると考えられるが，複数の使用者の行動履歴を比較することで，特定の個人に偏らない全体的な行動の傾向を把握することが出来る．システムによって，個人の行動履歴に特化したものや全体の傾向を重視したものなどさまざまな種類があるが，前者だけでは個人の，他人と異なる特徴的な行動の抽出は困難であり，後者においては得られた数値を比較する際の正規化の問題やプライバシの問題などの問題がある．

4.1.1 PEPYS

Newman らによる PEPYS[26] は，Active Badge によって取得されたデータを蓄積して，日記を生成するシステムである．PEPYS は，Xerox EuroPARC における Activity-Based Information Retrieval プロジェクト [17] の一環として構築されたシステムであり，このプロジェクト全体では行動記録を蓄積することで，行動と関連付けられた人間の記憶の検索を容易にすることを目指している．PEPYS は，Active Badge によって取得された,「誰が」「いつ」「どこに」「どうした (入った，出た，動いたなど)」といった情報から，個人の行動を類推し，時間軸で列挙した日記形式で出力するシステムである．

PEPYS は，ロケーションモデルとしては部屋などの名称を利用した Topological Model が用いられている．また，研究の主眼が，センサから得られた情報からの行動の類推であり，さまざまな類推ルールを定義することでこれを実現している．一方未来予測には触れられていない．また，利用者は単一の人であり，他人の行動履歴との比較などは行っていない．

4.1.2 Cyberguide

Cyberguide[1] は，さまざまなハンドヘルド型コンピュータ上に実装された，屋内，および屋外向けの観光ガイドシステムであり，ユーザの現在位置や過去の移動履歴から必要な情報やサービスを類推しユーザに提供する．屋外用途には GPS を備えた PDA を利用し，観光情報や地図などを提供する．

Cyberguide は，観光という限定された状況において，移動履歴を利用することでユーザにとって有益な情報を予測するという，未来予測型のコンテクストアウェアシステムである．しかし，移動履歴だけでなく現在の観光地に対する印象などを追加することでより有益な情報提供が行える可能性について言及している．また，現在のところは単一利用者を想定したシステムであるが，他の観光客の動向を取り込み，人気の観光地を判断することの有益性も議論されている．

4.1.3 Infostation-Based Hoarding

Infostation-Based Hoarding[16] は，観光地をめぐる観光客が PDA などを持ち，観光地ごとに置かれている 802.11 などによるホットスポットで観光情報を入手するという環境を前提に，ホットスポット間を移動中に必要となる情報を事前に効果的にキャッシュする手法として研究されている．ユーザがホットスポットにいるときに PDA に予め必要となる情報がダウンロードされるのだが，ホットスポットを渡り歩くすべてのユーザの行動はシステムに保存されており，この情報をもとにユーザの移動の確率モデルが構築される．そして，ホットスポットでは次に移動する確率の高い場所に関する情報から優先的に情報がキャッシュされる．

本手法では多人数の移動履歴を総合することで平均的な観光行動を算出し，個々の観光客の未来予測に用いている．本手法においては観光のような限定された条件下で行動モデルの構築が行われており，有効な行動予測が出来るとしている．しかしさまざまな意図を持った人が集まるような環境においては，本手法をそのまま適応することは困難だと考えられる．なお，本手法において，単に移動履歴だけでなく，地図のような外部の情報を利用することで予測精度を上げる手法が言及されている．

4.1.4 Activity Compass

University of Washington において研究されている Activity Compass は，アルツハイマーの患者などを想定し，現在の目的地を指し示す矢印を持つ PDA を通して日常行動を支援するツールとして開発が進んでいる．Activity Compass の研究の一環として，文献 [19] において GPS で取得した行動履歴を解析し，より高次元の情報を取得する手法が検討されている．乗り物の運行パターンなどの行動履歴に対する一般的なルールを統計的な手法で行動履歴に当てはめ，行動履歴から利用している交通手段などの情報の抽出を実現している．また，構築したモデルから行動の予測も行っている．この研究の今後の課題として，GPS データが取得できない，という情報の積極的な活用や，毎日，枚急繰り返される行動パターンの学習，交通手段よりもより上位の，行動の目的地や目的といった情報の抽出や初めて訪れる場所における利用などについて触れている．

なお，本研究の先行研究として，文献 [3] や文献 [27] の研究が挙げられる．これらの研究においても，GPS で取得され蓄積された行動履歴が対象とされ，それを解析することで行動パタンの抽出や行動予測などが行われている．

4.1.5 交通計画における GPS の利用

交通計画とは，道路ネットワークを移動する自動車や歩行者などを対象に交通分析を行い，道路や鉄道のネットワークの設計や改良を目指す研究分野である．こうした研究における基礎データとして，交通量や移動記録などのデータが必要となる．従来のインタビューなどを利用した移動記録や定点での交通量調査に加え，近年 GPS を用

いた移動記録の自動取得が注目されており，それに伴い機械的に取得したGPSデータを解析する研究が行われている (文献 [23] など)．交通計画において，人や自動車などの移動はトリップという始点，終点，その間の移動をひとまとまりとした単位で取り扱う．しかしGPSから取得したデータには，通常始点や終点は明示されていない．また，GPS特有の問題，すなわち，電源投入後から位置情報の取得を開始するために数分間を必要とする問題や，高い建物や森林中といった周辺環境によってデータの取得が不可能になる問題，そして，建物内ではデータの取得が出来ない点などから，データは必ずしも移動体の移動を完全に記録しているわけではない．そのため，GPSデータから始点や終点を自動的に抽出する手法や，GPS特有のデータの欠落を補完する手法などが研究されている．

4.2 データ取得，解析インタフェースの提案と改良

大量にあるデータやストリームデータに対してフィルタリングを行い，情報を整理し必要な情報だけを抽出する機構は数多い．本節では，そういったシステムの中から対話的で直感的な手法で情報抽出を行えるシステムを関連研究として紹介する．これらのシステムは，本研究と対象や異なったりより汎用的なシステムであったりするが，ユーザインタフェースや内部のアーキテクチャは本システムを構築する上で参考になる．

4.2.1 Navigational Blocks

Navigational Blocks[4] は，University of Washington の Ken Camara らによって開発された，タンジブルユーザインタフェースを用いた情報検索の仕組みであり，観光地にある情報端末において，直感的に情報を検索することを目標に設計されている．図 4.1 に，Navigational Blocks において利用されているブロックを示す．

Navigational Blocksはシアトルの観光ガイドをテーマに実装され，シアトルの観光名所の歴史的な由来を説明するコンテンツへの直感的なアクセスを実現している．”When”, “Who” といった問いを示すブロックをおくことで，該当する情報が表示されるほか，関連するより詳細な情報をアクセスしたいときにもう一度ブロックをおくことでデータを取得することが出来る．また，複数のブロックを組み合わせたAND,ORといったロジックの表現やブロックそのものにディスプレイを実装するといったアイディアも検討されている．

場所に関連付けられた情報に対して，エンドユーザにわかりやすい形での情報取得を目指すという点で，Navigational Blocks は本論文のシステムと同様の機能を持つ．どちらのシステムも，データベースに対する複雑なクエリーを視覚化することで直感的なデータ取得を目指している．Navigational Blocks は，限定された状況での限定された情報アクセスを主眼に設計されているため，より直感的にわかりやすいハードウェアによる情報検索が実装されている．

図 4.1: Navigational Blocks

4.2.2 Max/MSP

Max/MSP[6] は，1986 年から開発されている，音楽や映像のインタラクティブな編集を目的としたビジュアルプログラミング環境であり，現在，cycling74 社より MacOS X，および Windows XP 向けの商用ソフトウェアとして提供されている．図 4.2 に，スクリーンショットを示す.

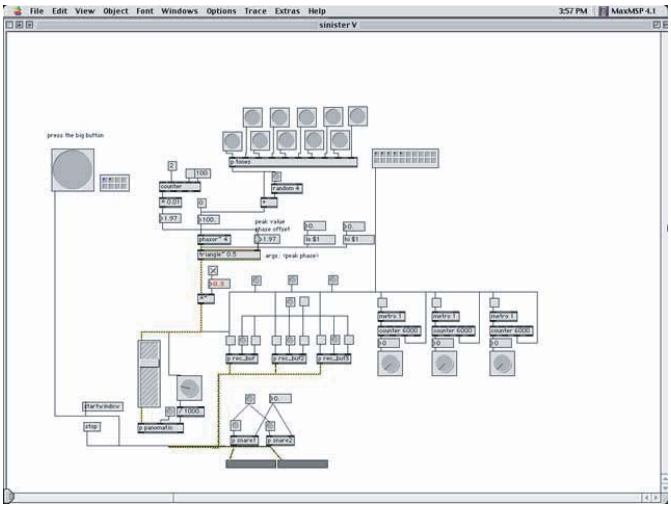

図 4.2: Max/MSP

当初は，MIDI データに対応した Max として提供されていたが，後にオーディオデータを対象とした MSP，画像や映像を対象とした Jitter が追加されており，発音アルゴリズムを自作した電子楽器の開発やアルゴリズムに基づいた自動作曲，入力した音声や MIDI イベントに対するエフェクターや Visual Jackey(VJ) の道具など，音楽やイ

ンタラクティブメディアの製作に広く用いられている．Max/MSP はこうしたメディアに携わる人々の標準言語として利用されているほか，C 言語を利用したモジュールの製作が可能なことから広くコミュニティが形成され，ユーザによって開発されたモジュールやプログラムが公開され交換されている．

ビジュアルプログラミング環境を利用して，データの編集方法をプログラムするという点において，Max/MSP は MUSE システムと類似している．Max/MSP はすでに多くの利用者を獲得しており，オーディオストリームの編集において広く受け入れられる編集の自由度とアルゴリズムやアーキテクチャの抽象化が行われていると考えられる．

4.2.3 ビジュアルデータマイニングツール

データマイニングとはPOS やクレジットカードの利用履歴など，コンピュータシステムの導入で収集が可能になった大量の生データから，ビジネスに有益な情報を発見する技術のことであり，一般の統計解析よりも大量の整理されていない情報を取り扱う点や，仮説の検証に限らず発見が困難なデータの構造や因果関係などを見つけ出すところなどに特徴がある．データマイニングには特定の解析手法があるわけではなく，ニューラルネットワークや決定木などさまざまな手法を組み合わせたり，視覚化によって情報の特徴を浮かび上がらせたりする．

IBM の Intelligent Miner や SAS の Enterprise Miner[20]，SPSS の Clementine[22] など，さまざまな企業からデータマイニングを行うツールが提供されている．これらのシステムでは，データやデータの分析結果の視覚化のほかに，データの解析手法をビジュアルプログラミングの手法で設定することが出来る．図 4.3 に，こうしたツールの一つである (株) 数理システムの VisualMiningStudio [28] においてデータの解析手法をプログラムしているスクリーンショットを示す．解析手法がアイコンとして示され，アイコン同士の流れを設定することで解析が行える．

本論文において構築したシステムでは，こうした汎用システムとは異なり地理座標と対応付けられた情報のみを扱う．取り扱える情報は限定されるが，地理座標に特化した情報操作においては優位であると考えられる．また，これらのシステムでは，事前に用意された情報のみを前提としている．しかし，本論文におけるシステムでは，動的に変動する情報を対象とした情報解析も対象としている．

4.3 本章のまとめ

本章では，関連研究を大きく二つに分け，行動履歴を蓄積，分析するシステムと改良されたデータ取得，解析インタフェースに分けて紹介した．前者の研究は，本論文では具体的に述べない個々の行動履歴の解析手法を詳細に研究しており，今後紹介されている手法を MUSE システム上に実装することで MUSE システムの柔軟性が評価できる．また，本システムでは取り上げない，複数の人々の行動履歴を総合して情報

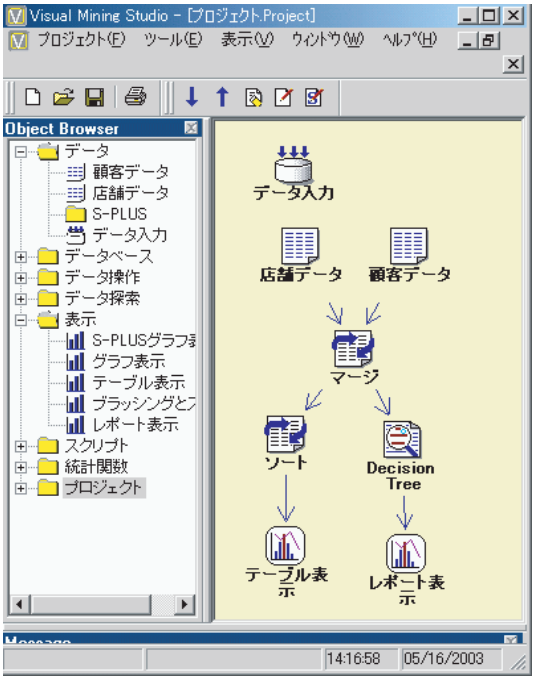

図 4.3: VisualMiningStudio

を抽出する研究も存在し，これらは本システムの今後の発展に参考になる．

後者の研究は，行動履歴に限定しないが柔軟で多様な情報を取り扱える情報解析システムとして紹介している．こうしたシステムのインタフェースや内部構造は，本システムの構築の際参考になる．

次章では，本章で提案した MUSE システムの具体的な設計と実装について述べる．

第5章　MUSEシステムの設計と実装

本章では，3章で挙げた機能をもとに，MUSEシステムの基本的な機能を実現するソフトウェアの設計，および実装を行う．特に，対話的な解析部品の接続機能を中心に設計，実装を行なう．また，実装したソフトウェア上で動作する解析部品をいくつか実装し，本システムの動作を確認する．

5.1 MUSEシステムの全体像

MUSE システムを構成するソフトウェア要素について述べ，システムの全体像を明らかにする．MUSE は，(1) 行動履歴入力部，(2) 行動履歴解析部，(3) 解析結果出力部，(4) 解析部品管理部，(5) 解析手法プログラム部からなる．図 5.1 にシステムの全体像を示す．

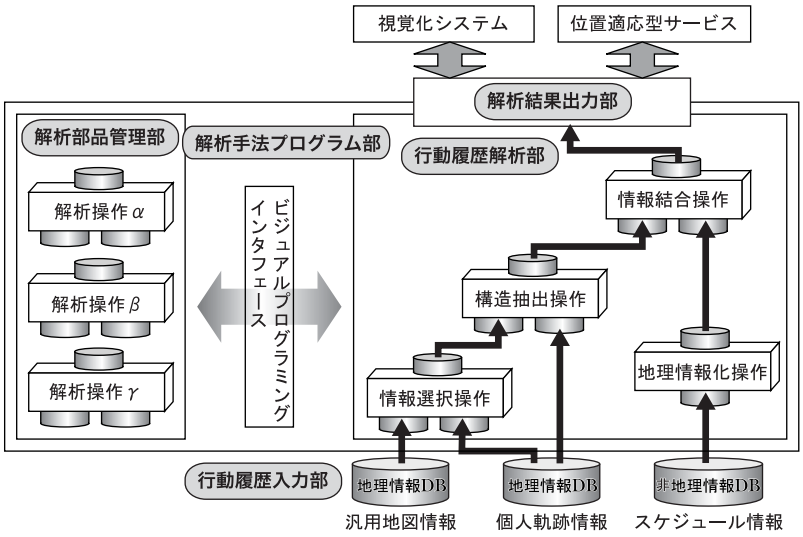

図 5.1: MUSE システム全体像

行動履歴入力部

本システムで取り扱う様々な行動履歴や，行動履歴解析の素材として用いる地理情報を入力する．このモジュールで，入力された情報には標準化された位置情報および時刻情報が付与され，本システムで取り扱う標準的な形式に変換される．位置情報の付与に際しては，様々な Geocoding の手法が用いられるほか，軌跡情報と時刻とのマッチングなども行われる．本システムが Push 型の情報配信を行うときは，行動履歴入力部が新しい行動情報の入力を検知し，情報配信の基点となる．

行動履歴解析部

行動履歴解析部は，解析手法プログラム部におけるプログラムを反映して解析部品を保持し，部品同士を接続する．行動履歴解析部に入力された情報は，各部品間を接続関係に従って通過し，解析結果出力部へ渡される．

解析結果出力部

解析結果を様々なシステムに出力する．出力機能の一部として，視覚化アプリケーションを構築するほか，ファイルに出力したり，ネットワークを用いて他のシステムに情報を転送するなど，様々な出力方法に対応する．また，アプリケーションからの情報取得要求を受け付け，そのとき本システムは Pull 型の情報配信を行う．

解析部品

MUSE において，行動履歴の解析アルゴリズムは，行動履歴の入出力を持ったオブジェクトとして実装される．本システムにおいて標準的なデータ形式を定義することで，解析部品は共通化される．また，解析結果も位置と時間とそのほかの情報からなる情報として出力することで，解析結果も行動履歴としての出力を実現する．このことで，複数の解析部品の任意の組み合わせを実現する．

解析部品管理部

解析手法を実装した部品を管理する．各部品は，行動情報を入力し，解析結果を再び行動情報として出力する．標準的な解析部品があらかじめ登録されているほか，新たに開発された解析部品を読み込んだり，本システムによって構築された複数の解析部品からなる解析手法を，新しい解析部品として出力する機能などを持つ．

解析手法プログラム部

GUI を通じた行動履歴の解析手法の開発を実現する．解析部品管理部に登録された解析部品を用い，それぞれの接続関係を設定することにより解析手法をプログラムする．解析手法に対する変更は即座にシステムに反映され，対話的な解析が実現する．GUI では，解析部品をアイコンで表現し，画面上に並べマウスで指示することにより接続関係を設定する．図 5.2 にユーザインタフェースの概要を示す．

図 5.2 内の Palette は，解析部品管理部に登録されている解析部品がアイコン化され並べられている．Canvas は，ユーザが行動履歴の解析手法を実際にプログラムする画面であり，この画面へ解析部品を Pallette からドラッグ&ドロップして登録し，解析部品同士をマウスによって指示し結線することで接続する．

5.2 設計方針

MUSE システムを設計，実装する際の設計方針を述べる．本システムは，オブジェクト指向に基づく GUI アプリケーションとして設計，実装される．

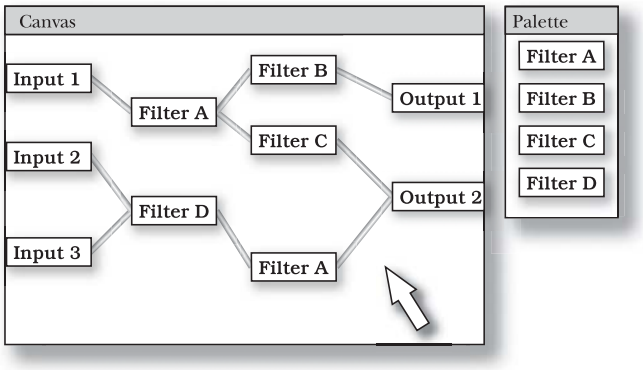

図 5.2: 解析手法プログラム部の GUI

オブジェクト指向による設計

本システムは，オブジェクト指向技術を前提に設計される．本システムを構成する解析部品などをオブジェクトとして記述することで，本システムの容易な構築を実現する．

イベントドリブンの GUI アプリケーションとしての設計

本システムは，GUI アプリケーションとして設計され，マウスやキーボードの操作，データの変更時などに発生するイベントに基づいた，イベントドリブンのアプリケーションとして設計する．

5.3 ソフトウェアの構成

本節では，本システムにおけるデータの表現方法，および，行動履歴解析部品の構造について述べる．本システムではデータはすべて ActionElement として表現される．また，行動履歴解析部品は行動履歴入力部，解析結果出力部とともに ActionFilter として表現される．以下でそれぞれの詳細を述べる．

5.3.1 ActionElement

ActionElement クラスは，本システムで取り扱うすべての行動履歴情報や地理情報のスーパークラスであり，本システムに入力される情報はすべて ActionElement クラスを継承するクラスとして表現される．図 5.3 に，クラス図を示す．ActionElement クラスは，本システムが取り扱う最小粒度の情報，すなわち地理座標で表現された形状情報，行動情報，およびタイムスタンプからなる．さらに，ActionElement クラスは入れ子状のデータ形式に対応しており，同じく ActionElement 型の要素を複数保持す

ることが出来る．たとえば，GPS から取得した軌跡情報を表現する場合，出発点から終点までの一連の軌跡が，一つの ActionHistory クラスとして表現される．さらに，一連の軌跡を構成している点の一つ一つが，InnerElement として表現される．

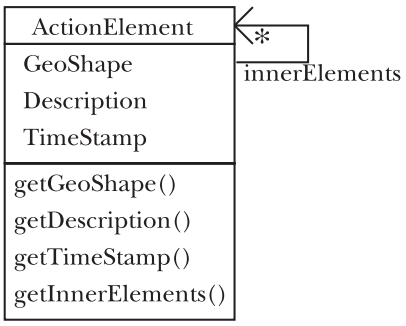

図 5.3: ActionElement のクラス図

本システムへデータを入力するモジュールを開発する際は，入力する情報は必ず ActionElement を継承した型で表現される必要がある．たとえば，デジタルカメラで撮影した写真を入力する際には，Description として写真ファイルのパスを，GeoShape として撮影地点を，TimeStamp として撮影場所を入力する．行動履歴ではない地図データのような環境データを入力する際にも，同様に ActionElement として表現する．その場合には，必ずしも TimeStamp を持たなくてもよい．また，地図データの場合地形的な構造を ActionElement クラスの入れ子構造と対応させて入力することが望まれる．

ActionElement は，その入れ子構造により柔軟なデータ表現が可能であり，さまざまな行動履歴や地理情報を入力することが出来る．また，厳密に型付けされているため後述する行動履歴解析部品の開発が容易である．

5.3.2 ActionFilter

MUSE の設計では，行動履歴入力部，解析結果出力部，行動履歴解釈部品を，すべて任意の個数の行動履歴情報の入出力機能を持った ActionFilter クラスとして設計する．個々の行動履歴入力，出力，解析機能は抽象クラスである ActionFilter クラスを継承して実装される．図 5.4 に，ActionFilter クラスと関連するクラスを示したクラス図を示す．また，以下に図中のクラスを説明する．

ActionFilter クラス

ActionFilter クラスは抽象クラスであり，ActionFilter の基本的な機能を実装する．ActonFilter の基本的な機能とは，Filter 同士の接続関係の実現であり，ActionFilter は任意の個数の入力 Socket と出力 Socket を持ち，他の ActionFilter との接続状態を表現する．また，接続された ActionFilter 間で情報を交換したり，変更を通知する機構を持

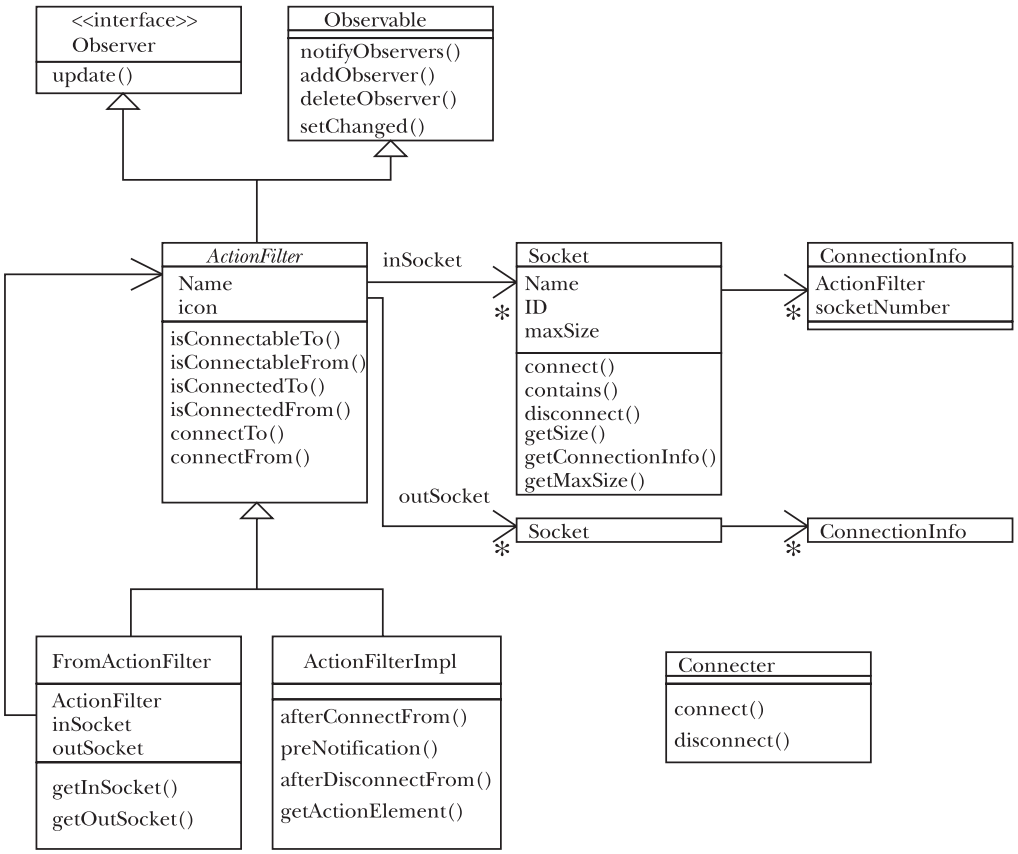

図 5.4: ActionFilter のクラス図

つ．ActionFilter 間での情報の交換は上位の ActionFilter から下位の ActionFilter へのメソッド呼び出しによって実現している．下位から上位へは変更の通知のみを行い，上位の ActionFilter は必要に応じて下位の ActionFilter から情報を取得したりさらに上位の ActionFilter へ変更を通知したりする．なお下位から上位への変更の通知は Observer Observable パターンを用いて行われる．ActionFilter が継承している Observable クラス，および実装している Observer インタフェースは，この目的のために用いられる．

Socket クラス

Socket クラスは，ActionFilter 同士の接続状態を保存するクラスであり，情報の入力を意味するの場合と出力を意味する場合がある．ActionFilter は任意の個数の入力向け，および出力向け Socket を持つことができ，個数に応じて Socket クラスのインスタンスが置かれる．ActionFilter 同士の接続は，ActionFilter が保持している Socket 同士を指定することで行われる．

Socket クラスは，自身の名前および内部的な番号を持ち，その Socket で接続を受け入れられる接続数を管理するほか，接続が行われると，接続状態を ConnectionInfo クラスとして保持する．複数 ActionFilter の接続を許可する Socket の場合，接続状況に

応じて ConnectionInfo が複数保持されることになる．

ConnectionInfo クラスは，接続相手の ActionFilter への参照と接続相手となる Socket の番号を保持し，接続状態を表現する．

FromActionFilter クラス

FromActionFilter は，ActionFilter 内で接続相手の ActionFilter の Socket 番号を隠蔽するクラスである．ActionFilter 同士の接続は，それぞれの Socket 番号によって管理される．しかし，接続相手の Socket 番号は ActionFilter 内部においては情報として意味がない．FromGeoFilter は内部に Socket 番号を保持しており，本クラスを経由することで接続相手の ActionFilter との Socket 番号を指定しない参照が実現する．

Connector クラス

Connecter クラスは，ActionFilter 同士の接続を実現するクラスである．Connecter クラスは接続と切断のためのメソッドが用意されており，入出力 ActionFilter，入出力 Socket 番号を指定することで ActionFilterd 同士の接続を行える．なお，それぞれの ActionFilter が備えている接続や切断のためのメソッドは，Connecter クラスを経由することでしか外部から呼び出すことができない．これは，接続手順を守ることや，矛盾する操作が同時に行われるのを防ぐためである．

ActionFilterImpl クラス

ActionFilterImpl クラスは，ActionFilter を継承して実装された行動履歴の入力，出力，歴解機能であり，図示されている afterConnectFrom()，afterDisconnectFrom()，preNotification()，getActionElement() の 4 つのメソッドを実装することで必要な機能を実装する．なお，詳細は 5.5 節において述べる．

5.4 ActionFilter の動作

本節では，ActionFilter 同士の接続や，ActionFilter 間のデータ交換時の動作について詳しく述べる．

5.4.1 ActionFilter 間の接続・切断時の動作

ActionFilter 同士の接続，切断はすべて Connecter クラスを通じて行われる．図 5.5 に，ActionFilter の接続時のシーケンスを示す．図に示す通り，Connector によって接続元，接続先のそれぞれの ActionFilter に対して接続の可否が検査されたあと，接続元の ActionFilter に接続先の ActionFilter が Observer として登録される．そして，接

続先の ActionFilter の接続受け入れのメソッドが呼び出され，接続先からさらに先に接続されている ActionFilter に対して変更が通知される．

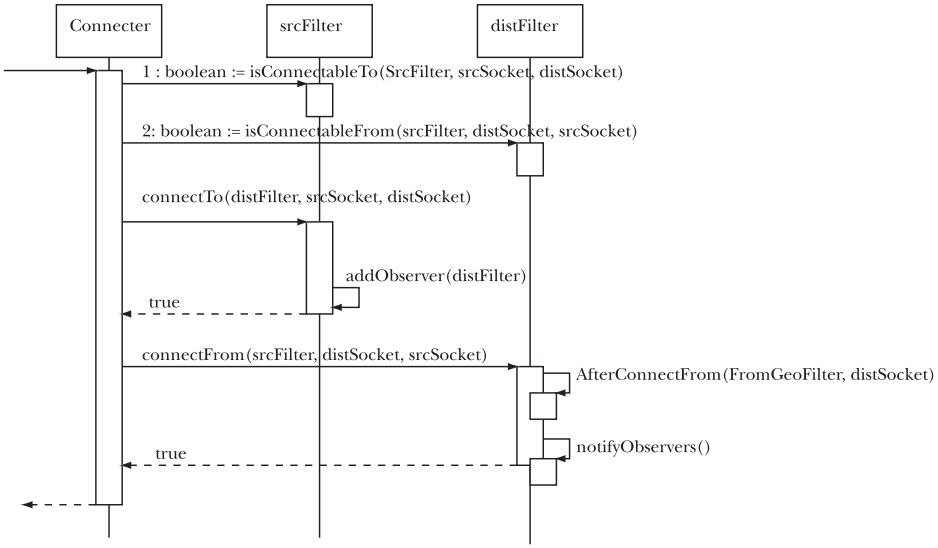

図 5.5: ActionFilter の接続シーケンス

ActionFilter 同士の切断は，接続時とほぼ同様の手順で接続状態を抹消する処理を行う．図 5.6 には，切断時のシーケンスを示す．切断の可否を確認後，接続元の ActionFilter に Observer として登録されていた接続先の ActionFilter の登録が取り消される．その後，接続先の ActionFilter の接続解消メソッドが呼び出され，接続先からさらに先に接続されている ActionFilter に対して変更が通知される．

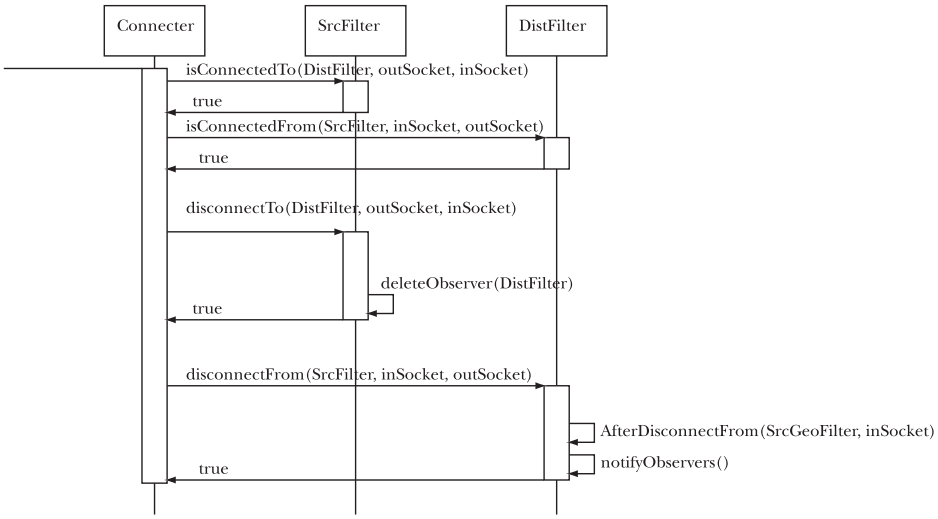

図 5.6: ActionFilter の切断シーケンス

5.4.2 データ配送時の動作

ActionFilter 間のデータの配送は大きく Push 型と Pull 型に分けられ，本システムでは両者を実現している．Push 型とは，データ上流から下流に変更を伝播させるデータ配送であり，Pull 型とは下流側から上流にデータを要求し行われるデータ配送である．Push 型の配送はデータソースの変更時や中流における接続状態の変更時などに下流に対して行われ，Pull 型はデータの終点や，中流における接続状態の変更時に上流に対して行われる．

Pull 型データ配送の動作

Pull 型のデータ配送時は，下流の ActionFilter は上流の ActionFlter の getActionElement() メソッドを呼び出し，ActionElement として表現されている行動履歴データを取得する．getActionElement() メソッドが呼び出された ActionElement では，自身がすでにデータを保持している場合には引数に応じて適切な情報を返し，自身がデータを保持していない場合には，さらに上流の ActionFliter の getActionFilter() メソッドを呼び出し，その結果に対し ActionFilter に実装されている解析手法を適応した結果を返す．多段に接続された ActionFilter 間ではそれぞれの ActionFliter がこのように振舞うことで Pull 型データ配送が実現する．

Push 型データ配送の動作

本システムでは，すべてのデータ配送処理は内部的に Pull 型として実現されており，Push 型のデータ配送処理は，Push されるデータの変更通知を受けた ActionFilter が Pull 型によるデータ転送を行うことで仮想的に実現している．データの変更などで Push 型のデータ配送が必要になった ActionFilter は，EventQueue を通じて自身を監視している ActionFilter に変更が起ったことを通知する．変更通知を受けた ActionFilter は，さらに上流の ActionFilter に変更を通知する．最新のデータが必要な ActionFilter が，変更通知の受信時に Pull 型データ配送の手順で最新の情報を取得することで，Push 型データ配送を実現している．

5.5 ActionFilter の実装方法

本節では，ActionFilter の開発の詳細を述べる．本節では，便宜的に ArcFilter を行動履歴入力機能を持った Data Source 型，解析結果出力機能を持った Data Sink 型，行動履歴の解析手法が実装された Filter 型に分類して説明する．また，Filter 型はさらに Cach 型と Dynamic 型に分類される．

5.5.1 実装するメソッド

新たに ActionFilter を実装する際に実装する必要のあるメソッドを説明する．以下の 4 つのメソッドを実装することで ActionFilter を実装することができる．

getActionElement メソッド

getActionElement(GeoShape area) メソッドは，ActionFilter にデータを要求するときに呼び出されるメソッドで，主に下流に接続されている ActionFilter によって Pull 型のデータ配送時に呼び出される．引数には必要なデータ領域を表す地理的な領域情報が渡されるので，戻り値として渡された領域と重なるすべての該当情報を ActionElement の形で返す必要がある．

afterConnectFrom メソッド

afterConnectFrom(FromActionFilter fromFilter) メソッドは上流側の Socket に新たに ActionFilter が接続されたときに呼び出されるメソッドであり，引数として新たに接続された ActionFilter に対応付けられた FromActionFilter が渡される．

afterDisconnectFrom メソッド

afterDisconnectFrom(FromActionFilter fromFilter) メソッドは上流側の Socket に接続されていた ActionFilter が切断されたときに呼び出されるメソッドであり，引数として接続されていた ActionFilter に対応付けられた FromActionFilter が渡される．

preNotification メソッド

preNotification(FromActionFilter fromFilter) メソッドは上流側の Socket に接続されていた ActionFilter の変更時に呼び出されるメソッドであり，ActionFilter に対応付けられた FromActionFilter が引数として渡される．

5.5.2 ActionFilter ごとの実装方法

各 ActionFilter の種類ごとに，実装方法を説明する．

Data Source 型

Data Source 型は，行動履歴や地理情報などのデータを本システムに取り込むために実装される ActionFilter である．本システム内では，ActionFilter 同士の接続の最上流に位置する．問い合わせに対して必要なデータを返すほか，データに変更があった

場合に変更イベントを発行する．Data Source 型の実装次第でファイルやデータベースからの情報取得や，センサなどからのリアルタイムの情報取得などさまざまな情報取得手法が可能になる．Data Source 型の ActionFilter を実装する際は，入力 Socket 数を 0 とし，getActionElement メソッドにおいて必要な情報の取得を行うような実装を行えばよい．

Data Sink 型

Data Sink 型は，本システムによって解析された結果を他のアプリケーションに出力する Data Sink であり，本システム内では ActionFilter 同士の接続の終点となる．この ActionFilter の実装によって，解析結果をファイルに出力したり，ネットワークなどを用いて他の分散システムに登録することが可能となる．また，視覚化アプリケーションなど簡単なアプリケーションは，Data Sink 型 ActionFilter として実装できる．Data Sink 型 ActionFilter の実装に当たっては，出力 Socket 数を 0 とし，afterConnectFrom，afterDisonnectFrom，preNotification メソッドそれぞれに対して必要な動作を実装すればよい．

Dynamic 型 Filter

Filter は，入出力をともに持つ ActionFilter を指し，行動履歴の解析アルゴリズムを実装する．getActionElement メソッドによってデータが要求されたときに初めて上流に接続されている ActionFilter にデータを問い合わせ，返されたデータに対して解析アルゴリズムを適応して返すように動作する．

Cache 型 Filter

Cache 型も入出力を持つ Filter 型の ActionFilter であるが，問い合わせ時に解析アルゴリズムを適応するのではなく，上流側のデータ変更時にすべてのデータを元に予めデータを生成してしまい，それに基づいてデータの問い合わせに答える ActionFilter である．複雑な演算を必要とする場合，この形式ならば問い合わせのたびに演算を行う必要がなく有利である．

本 ActionFilter の実装に際しては，afterConnectFrom，afterDisonnectFrom，preNotification メソッドそれぞれに対して予めデータの生成を行い，getActionElement メソッドに対しては本 ActionFilter 内に保持されているデータを返すように実装する．

5.6　システムの実装

本システムは Java 言語によって，全体で 14,000 行程度のアプリケーションとして実装された．図 5.7 に，システムのスクリーンショットを示す．画面右側のパレット部に

ActionFilterが登録されており，画面中央のキャンバスで，ActionFilter同士をマウスで接続することができる．実装に際しては，5.7節で述べるように基本的なGeoFilterもシステムの一部として実装している．

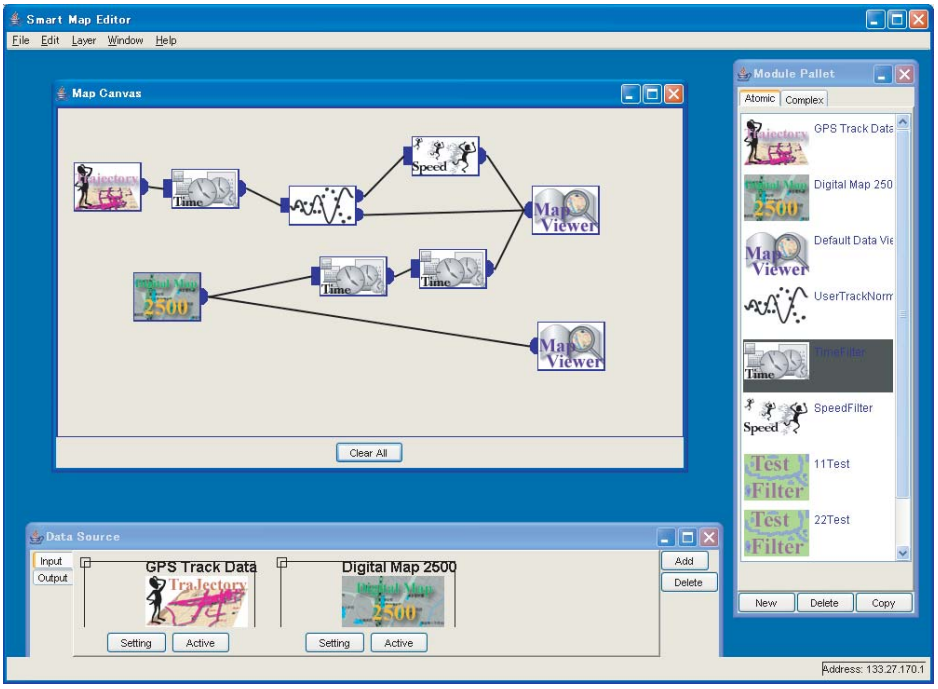

図 5.7: システムのスクリーンショット

5.6.1 実装時の利用技術

実装時に利用した座標系，および2次元データのインデックス方式について説明する．

平面直角座標系

システム内部において地理座標を表現する際には，緯度経度をそのまま用いてもデータの保持や距離の測定，地図の描画などが行えず不便であり，平面座標に変換して取り扱う必要がある．本システムの実装においては，日本国内で広く用いられている平面直角座標系 [29] と緯度経度による座標系を相互変換するライブラリを構築し，内部的には平面直角座標系で表現した平面座標ですべてのデータを取り扱っている．

R-Treeによる2次元データのインデックス

ActionFilterの保持するActionEventデータに対し地理的な近接や包含関係などでの問い合わせを実現するため，R-Tree[11] というデータ構造を利用したインデックス

を利用している．

R-Tree は 2 次元以上の次数のデータを効率よく検索するために考えられたデータ構造であり，ある点や領域に対する，近接や内包などの検索を効率よく行える．R-Tree の効率を改善するためにいくつかのデータ構造が提案されている．本システムでは，Java における R-Tree の実装として University of California において開発されている SpatialIndex Library[12] を利用している．このライブラリでは，R-Tree の改良の一つである R*-Tree[18] が用いらている．任意の次元におけるデータの格納が可能であるが，本システムでは 2 次元の地理オブジェクトの索引として利用している．

5.7 基本的な ActionFilter の実装

構築した MUSE システム上で，基本的な機能を備えた ActionFilter を実装する．

5.7.1 Data Source の実装

数値地図 2500 モジュール

国土地理院が GIS での利用を目的に発行している数値地図 2500 を読み取り，本システムに対応した形式での入力を実現する ActionFilter を設計，実装した．この ActionFilter は，DataSource として機能する．表 5.1 に，数値地図 2500 に含まれるデータを示す．これらのデータは，地図として視覚化されるほか道路ネットワークや建物の情報と行動軌跡などと組み合わせ，GPS から得られた情報からより高次元の情報を抽出する際にも用いられる．

表 5.1: 数値地図 2500 が持つデータ

項目	構造	属性
行政区域・海岸線 (町丁目/大字)	ポリゴン	点情報 (位置参照情報) 行政コード，名称
街区 (住居表示の「番」)	ポリゴン	点情報 (位置参照情報) 街区符号
道路中心線 (ネットワーク)	道路ネットワーク	主要なものの名称
鉄道，駅	ベクタ線，駅は点	名称 (路線名)
内水面，公園，飛行場等の場地	ポリゴン	名称
公共建物	ポリゴン	種別・名称
測地基準点 (三角点)	点	名称

Shape ファイルモジュール

Shape 形式 [9] は，GIS において広く利用されている米 ERSI 社の ArcGIS[8] において地図データを格納する際に用いられるファイルフォーマットであり，フォーマットが公開されているため他の GIS を含め広く利用されているデータ形式である．本システムにおいても，Shape 形式の GIS データを読み込む機能を実装した．Shape ファイル形式のデータは広く市販されており，本 Data Source を用いて市販されているデータを入力することができるほか，フリーで入手できるデータもある．

GPS 軌跡入力モジュール

GPS により取得されたデータを読み込むモジュールであり，ささまざまな GPS のフォーマットへの対応や，ファイルやシリアルポート，USB などさまざまなデータの取得形式への対応が必要である．現在の実装においては GPS データはファイル形式としては Garmin PCX5 のデータ形式である TRK 形式や Sony 社製の出力する形式である IPS 形式に対応し，またデータの取得方式はファイルにのみ対応している．今後，シリアルポートによる GPS データの標準フォーマットである NMEA 0183 形式の読み取りに対応する予定である．

5.7.2 Data Sink の実装

本システムでは，DataSink としてさまざまな DataViewer を構築することが可能なほか，他のアプリケーションに対する API を提供することも出来る．現在は，DataSink のプロトタイプとして 2 種類の Viewer を設計，実装したほか，

Default Viewer

本システムで標準的な 2D の地図表示を実現する DataSink 型 ActionFilter を設計，実装した．複数のデータの入力が可能で，それぞれを重ねた表示を実現する．一般的な地図アプリケーションに類似したユーザインタフェースを持ち，矩形状に任意の地点を任意の縮尺で表示出来る．また，接続された複数の入力のそれぞれに対し，表示の有無や表示の順序を設定することが出来る．図 5.8 に，Default Viewer に数値地図 2500 と GPS 軌跡を重ねて表示したところを示す．

Table Viewer

情報を，表として出力する DataSink 型 ActionFilter である．データの詳細を数値を表示しながら把握，検討する際に有効であり，本システム上での行動履歴解析手法の開発の際にも有効な情報を得ることが出来る．図 5.9 に，GPS データを表示したところを示す．

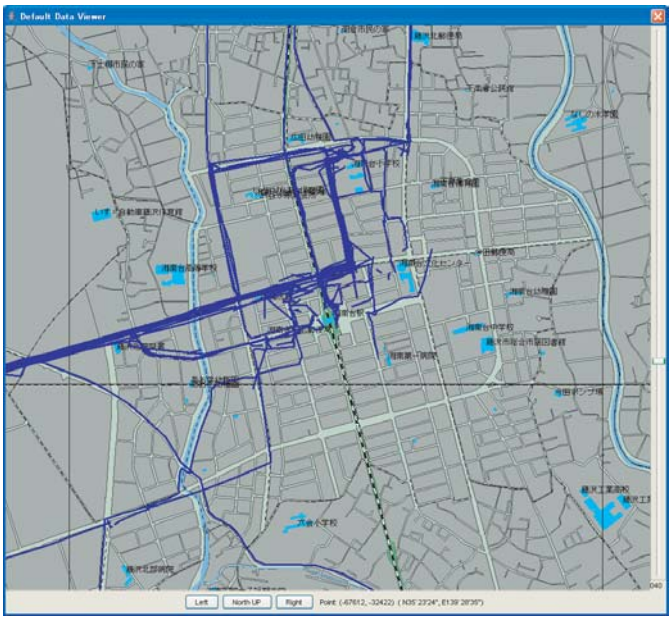

図 5.8: Default Viewer

5.7.3 Filter の実装

時刻フィルタ

データの属性として設定されているデータの時刻に基づいて情報をフィルタする．GPS 軌跡データに適応して特定の期間の軌跡のみを抽出したり，地理データに適応して特定の時点での地理情報を取得するなどの用途が考えられる．図 5.10 に，本フィルタの設定画面を示す．本フィルタは，Dynamic 型のフィルタとして実装されている．

スピードフィルタ

スピード属性を持つデータに適応して，特定の範囲内のデータのみを通過させる．GPS データに適応し，乗り物の推定などより高度な情報の抽出などの際に用いることが出来る．本フィルタは，Dynamic 型のフィルタとして実装されている．

軌跡正規化フィルタ

軌跡データから，出発点，移動軌跡，終着点からなる構造を切り出し出力するフィルタである．とくに GPS で取得したデータは電波の受信状況の変動によるデータの欠落や伽リブレーションのためのデータ取得開始までの遅れなどがあり，こうしたデータの不備を補完しながら軌跡データを正規化する．本フィルタからは正規化された軌跡情報を得られるほか，軌跡中の滞在点を点データとして得ることができる．本フィルタは，Cach 型のフィルタとして実装されている．

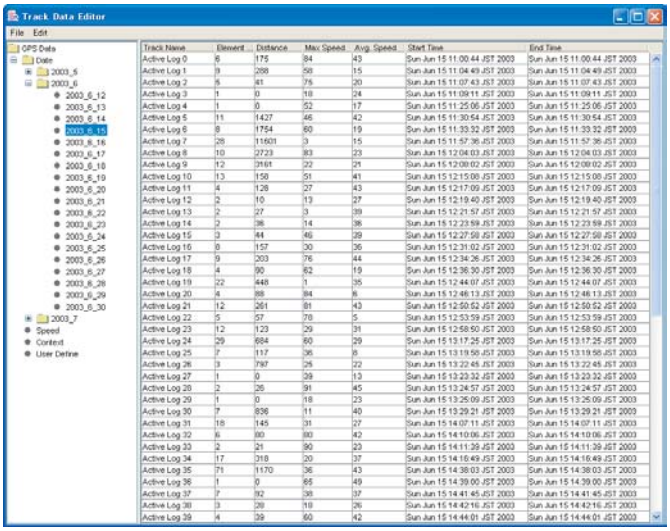

図 5.9: Table Viewer

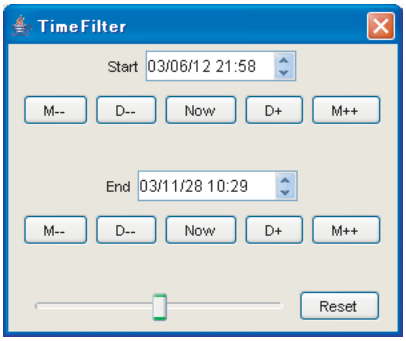

図 5.10: Time Filter の設定画面

5.8 本章のまとめ

本章では，3 章で説明されている機能やソフトウェア構成に基づき，MUSE システムの設計，実装を行った． MUSE システムはオブジェクト指向に基づく GUI アプリケーションとして設計，実装されており，本章ではシステムで用いられているデータ構造や，システムの中心的な構成要素である ActionFilter の詳細な構造や動作について説明した．本システムは Java 言語によって基本的な ActionFilter とともに実装された．

次章では，本章で実装したシステムの性能を測定し評価を行う．

第6章　評価

本章では，システムの評価を行なう．実装したシステムの動作時のパフォーマンスを測定し，本システムが目指す対話的な行動履歴解析手法の開発に支障のない応答速度が得られるか確認する．また，今後のシステム改良点を探すためのデータとして利用する．

6.1 測定環境

測定には，表6.1の環境を利用した．この環境は，現時点で市販されているコンピュータのなかで平均的な性能を持つものであると考えられる．なお，OSにLinuxを利用したのは，Javaにおける測定でms単位の測定精度を得るためである．

表 6.1: 測定環境

CPU	Pentium4 2.53GHz
memory	1024MB
OS	Linux 2.4.22
JDK	J2SDK 1.4.2_03
VGA	ATI Radeon 9000
X11	XFree86 4.2.1
Screen	1280*1024 24bit color

6.2 データ読み込みの性能

本システム上で実装したGPS軌跡入力モジュールと数値地図2500モジュールそれぞれに対して，データの読み込みにかかる時間を測定した．これらのデータの読み込みは本システムの起動時に必要であり，なるべく短時間での読み込みが望まれる．

6.2.1 GPS軌跡入力モジュールの性能

本システム上で動作するGPS軌跡モジュールにおいて，GPSで取得しファイルに保存された軌跡情報を読み込む際の性能を測定する．

利用データ

評価において利用するデータは，筆者が2003年6月からハンディGPSのGarmin Etrex Legend[10]を持ち歩いて実際に取得したデータであり，今回は11月までの半年分を利用した．データは，ある瞬間の位置情報を示すポイントと，ポイントの集合を軌跡としてまとめたトラックからなり，表6.2にその分量をまとめる．平均ポイント数は，1トラックあたりのポイント数の平均を示す．なお，取得したデータは1ヶ月ごとにファイルにまとめており，今回の測定においてもそのファイルを利用している．各々のファイルはTRK形式のテキストファイルである．

表 6.2: 評価に用いる軌跡データ

年月	データサイズ (byte)	トラック数	ポイント数	平均ポイント数
2003 年 6 月	321,310	286	5,893	20.6
2003 年 7 月	294,518	297	5,336	18.0
2003 年 8 月	473,739	412	8,676	21.1
2003 年 9 月	365,855	298	6,730	22.6
2003 年 10 月	307,187	287	5,608	19.5
2003 年 11 月	193,772	153	3,573	23.4
平均	326,063.5	288.8	5,969.3	20.7

1ヶ月分のデータ読み込みの所要時間

利用データの中から最も平均的なデータと考えられる 2003 年 6 月のデータを対象に，1ヶ月のデータを読み込む所要時間を測定した．表 6.3 に，測定結果を示す．測定は 10 回繰り返し，平均を求めている．

表 6.3: 6 月分のデータの読み込み平均時間

年月	データサイズ (byte)	読み込み時間 (ms)
2003 年 6 月	321,310	296

全データの読み込み時間

続いて，6ヶ月分すべてのデータを読み込み，その所要時間を図 6.1 に示した．測定は 10 回繰り返し，平均を求めている．図から読み取れるように，すべての軌跡データを読み込むのに 2230.6ms 必要であり，またデータの読み込みにかかる時間はほぼデータ量に比例している．

6.2.2 数値地図 2500 モジュールの性能

本システム上で実装された数値地図 2500 モジュールに地図データを読み込むのにかかる時間を測定した．地図データには，国土地理院発行の数値地図 2500 より，藤沢市のデータを利用した．数値地図 2500 は，東西 2000m, 南北 1500 メートルの区画ごとに整備されており，藤沢市を 40 区画でカバーする．表 6.4 に使用したデータについてまとめる．

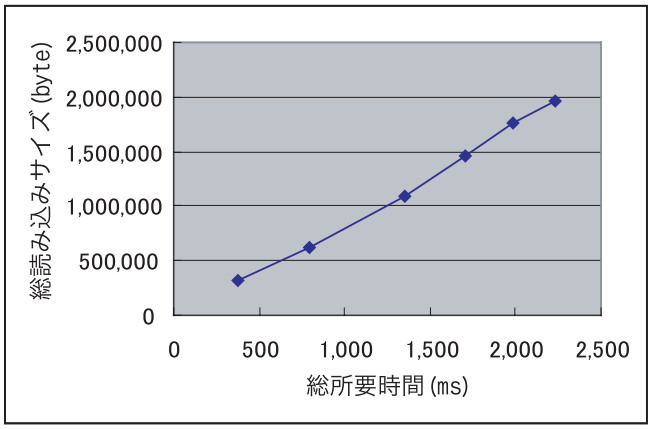

図 6.1: 6ヶ月分のデータ読み込み時間

表 6.4: 数値地図 2500 における藤沢市のデータ

	サイズ (byte)	街区数	建物数	道路データ (byte)
最小サイズ	15,005	8	0	1,462
最大サイズ	408,249	534	71	185,733
40 区域合計	9,261,000	10,688	833	4,298,193
平均	231,525	267.2	20.8	107,454.8

性能測定に当たっては，藤沢全域である 40 区域のデータを，順次読み込ませ，その所要時間を測定した．測定は 10 回行い，その平均値を採用している．図 6.2 に，測定結果を図示する．また，表 6.5 にその測定数値を示す．

表 6.5: 数値地図 2500 読み込み時間

合計読み込み時間 (ms)	3,199.3
1 区域当たり平均 (ms)	80.0

6.2.3 データ読み込み性能の考察

今回評価を行った範囲内では，データの読み込みはシステム起動時の 1 回という条件では十分な性能が示せたと考えられる．しかし，今後システムを発展させてゆく際には，より大量のデータを利用することなどが考えられる．今後は，ファイルやデータベースから必要時に動的に読み込む機構の構築と，パフォーマンスを考慮したキャッ

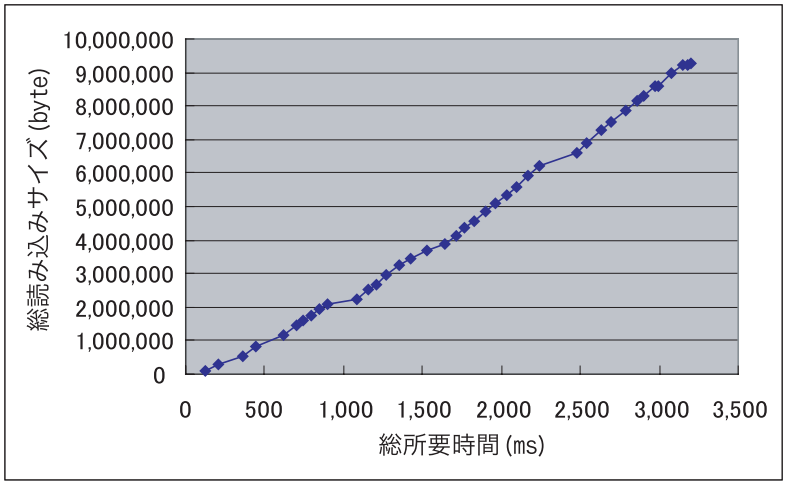

図 6.2: 数値地図 2500 読み込み時間

シングを行うことで，これ以上の起動時間の長時間化を避けながら，より大量のデータを取り扱える方策を考える．

データの読み込み手順は，1) データファイルを読み込みオブジェクトを生成，2) システム内部の地理座標系への対応付け，3)R-Tree インデックスへの登録という手順を経る．今回は詳しくは測定できなかったが，それぞれの手順ごとの時間を調べ，今後のシステムの改良に役立ててゆく．

6.3 単純な構成時の性能測定

本システムを用いて構築する最も単純なアプリケーションとして，軌跡閲覧システム，地図閲覧システム，軌跡と地図の合成閲覧システムを構築し，その性能を測定した．本測定においても，入力データとしては前節で利用した軌跡情報と藤沢市の地図情報とを用いており，軌跡情報は 6ヶ月分の軌跡を，地図情報は藤沢市全域の情報を読み込ませた状態で測定を行なう．また，出力には標準的な視覚化を行う Default Viewer を用いている．これらを組み合わせてアプリケーションを構築して性能を測定した．

6.3.1 軌跡情報の閲覧時

GPS 軌跡入力モジュールと Default Viewer を組み合わせ，図 6.3 に示すような単純な軌跡閲覧アプリケーションを構築した．

この環境において，読み込ませる軌跡の個数，および描写スケールを変化させ，描画時間を調べた．図 6.4 に軌跡の個数を変化させたときの描画時間を，図 6.5 にスケールを変化させたときの描画時間を示す．軌跡の個数に対しては，ほぼ直線的に描画時間が増加することがうかがえる．また，半年分の軌跡を描画させたときでも，300ms

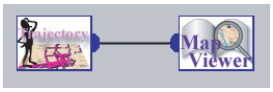

図 6.3: 単純な軌跡閲覧環境

以下の時間で描画が完了しており，ストレスのないインタラクションが実現している．スケールに対しても，同様に拡大するにつれて描画時間が減少しているが，広範囲を閲覧できる状態において 300ms 以下で描画が完了するため，十分な応答性のあるシステムであると考えられる．

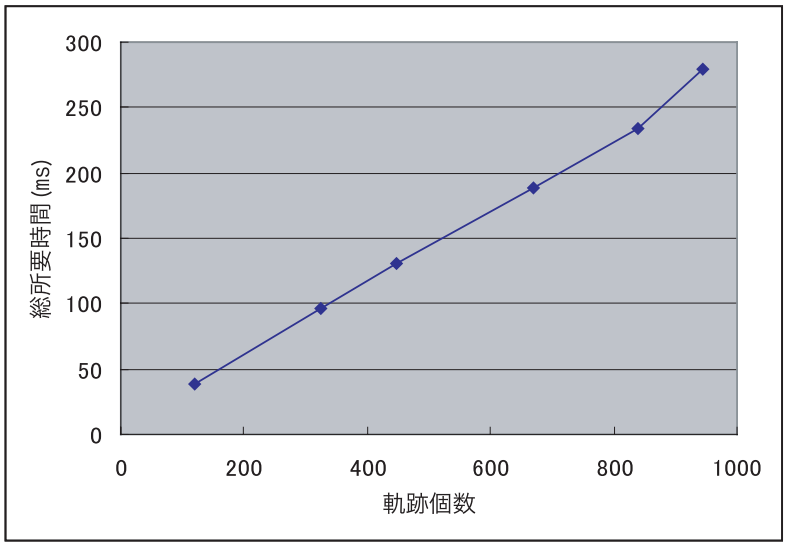

図 6.4: 軌跡個数による描画時間の変化

6.3.2 地図情報の閲覧時

数値地図 2500 モジュールと Default Viewer を組み合わせ，図 6.6 に示すような単純な地図閲覧環境を構築した．

本環境において，地図情報のレンダリングに必要な時間を測定した．地図データは単にデータとして軌跡データの解析時に用いられるだけでなく，視覚化の際に背景として用いられることも多いと考えられる．そのため，地図データのレンダリングに要する時間は本システムの応答性に大きな影響がある．本測定においては，3 種類のレンダリング手法を用意し，それぞれごとに地図閲覧に要した時間を測定した．なお，それぞれのレンダリング手法の詳細は表 6.6 にまとめてある．

各描画方式ごとに，さまざまなスケールで地図を描画し，所要時間をまとめたものが図 6.10 である．各描画方式において，スケールが 0.3 前後のときに描画時間が最低

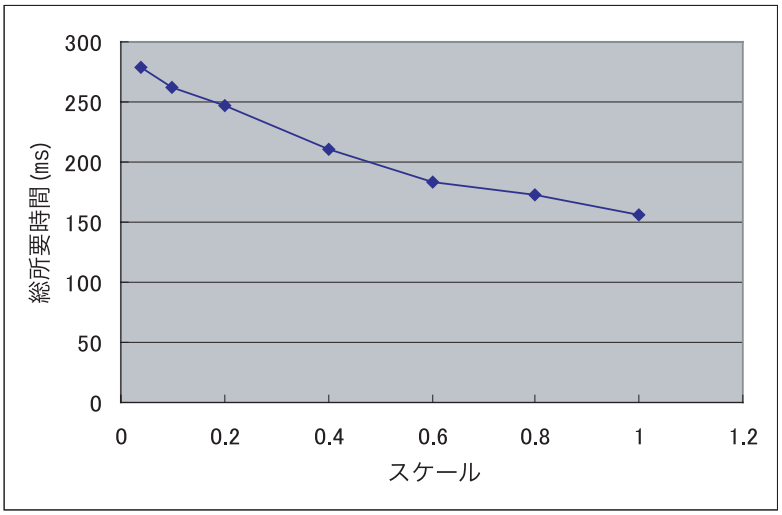

図 6.5: スケールによる軌跡描画時間の変化

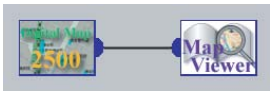

図 6.6: 単純な地図閲覧環境

となっている．小縮尺時に描画時間が非常に長くなることは，本システムの応答性に大きな影響を与えており，描画手法の改良やキャッシュの活用など，何らかの手法で小縮尺時の応答性を向上させる努力が必要である．また，スケールの 0.4 付近を境に，高縮尺時にも表示されるポリゴンや線の数が減るにもかかわらず描画時間が長くなるのは，本システムのアーキテクチャ上の問題が考えられ，より詳細な測定や分析が必要である．

それぞれの描画方式ごとに比較すると，小縮尺時には描画方式 1 がもっとも描画時間が短く，高縮尺時には描画方式 2，3 のほうが描画時間が短い．各描画方式は主に地図の見易さの観点から使い分けているが，描画時間から判断しても妥当な選択肢であるといえる．

表 6.6: 地図のレンダリング手法

名称	図	スケール	特徴
描画方式 1	図 6.7	0.01-0.28	広域向け．道路を線で描画.
描画方式 2	図 6.8	0.29-0.35	各街区をポリゴンで描写.
描画方式 3	図 6.9	0.36-1.00	拡大図向け．目印となる文字列を追加.

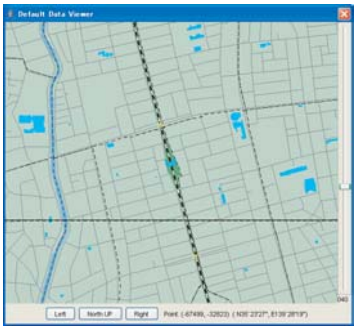

図 6.7: 描画方式 1

図 6.8: 描画方式 2

図 6.9: 描画方式 3

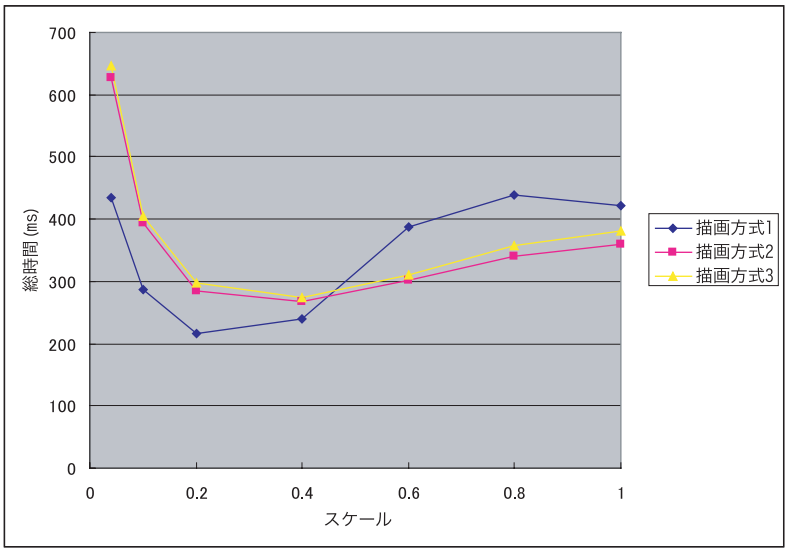

図 6.10: スケールによる地図描画時間の変化

6.3.3 地図と軌跡情報を重ねた情報閲覧時

GPS 軌跡入力モジュールと数値地図 2500 モジュール，Default Viewer を組み合わせ，図 6.11 に示すような軌跡および地図を重ねて閲覧する環境を構築した．軌跡情報を単独で閲覧することは現実的にはあまり考えられず，背景に地図を用いる本環境は本システムを用いて構築するアプリケーションの最も原始的なものであるともいえる．

本環境においてレンダリングに要する時間を測定し，表 6.7 にまとめた．また，図 6.12 にグラフとして示した．低縮尺の場合にも，描画時間は 800ms 前後かかっており，また高縮尺の場合には 1500ms 以上の描画時間を要する．地図のみ，または軌跡のみで測定した結果と比べると，特に地図の描写において地図のみの場合の 2 倍前後の時間を要しており，これが全体の描画時間を大きく損ねている．

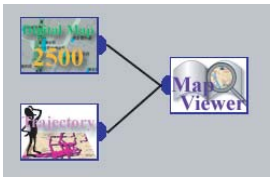

図 6.11: 軌跡および地図閲覧環境

表 6.7: 軌跡および地図のレンダリング時間

スケール	地図	軌跡	その他	合計	軌跡のみ	地図のみ
0.04	1326.1	253.1	4.9	1584.1	279.2	647.3
0.1	854.1	266.1	6	1126.2	261.9	405.5
0.2	473.2	362.5	4.4	840.1	246.3	297.4
0.4	581.9	197.8	5	784.7	210.9	273
0.6	625	170.1	4.6	799.7	182.6	311.1
0.8	661.6	162.2	4.5	828.3	172.2	356.5
1	587.8	251.2	4.7	843.7	155.8	380.9

6.3.4 単純な構成時の考察

本節では，GPS 軌跡モジュールと数値地図 2500 モジュールを入力，Default Viewer を出力として，単純な構成で構築したアプリケーションの性能を測定した．その結果，GPS 軌跡の描画時間は十分な応答性を持つこと，および特に小縮尺時の地図描写時の応答性が十分ではないことがわかった．また，GPS 軌跡モジュールと数値地図 2500 モジュールを組み合わせた場合，地図描画時間が単独の場合に比べほぼ倍となることが明らかになった．現在のところ地図の描画性能の下落の原因はわからないが，地図情報は描画時に背景情報としても用いられる重要な情報であり，さまざまな情報を重ねて表示する今回のような使い方は一般的である．そのため，本システムの改良に当たっては，描画時間の低下の原因を調べ，システムのアーキテクチャの改善をする必要がある．

6.4 行動履歴の解析時のシステムのオーバーヘッド

本節では，行動軌跡を解析するときに，ハードコーディングされた解析アルゴリズムに対し，本システムにおける部品化された解析手法を接続する手法がどの程度オーバーヘッドとなるか考察する．

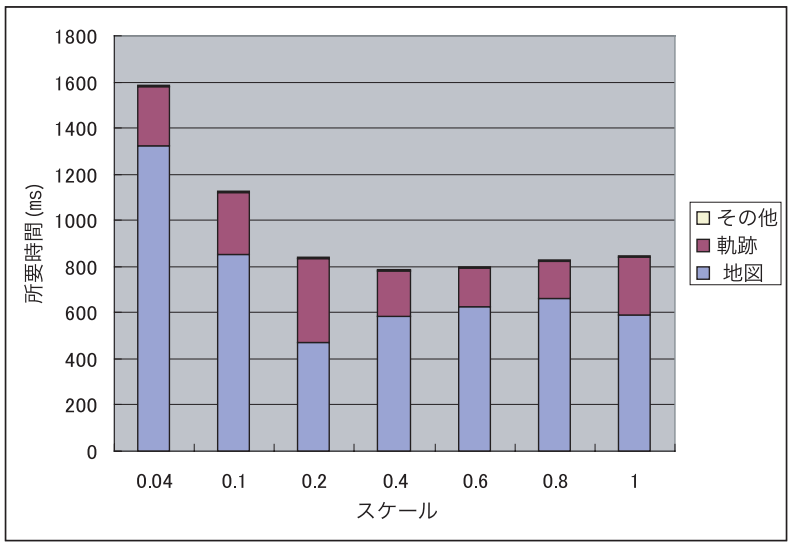

図 6.12: 軌跡および地図のレンダリング時間

6.4.1 時刻フィルタの多段接続によるオーバーヘッド測定

前節でも用いた軌跡情報の閲覧アプリケーションに対し，時間制約を加えるフィルタを経由させ特定の時間内の軌跡の閲覧を実現できるシステムを構築した．この際，図 6.13 に示すように，時刻フィルタを多段に接続し，本システムのオーバーヘッドを測定する．

図 6.13: オーバーヘッドの測定環境

ハードコーディングされたシステムにおいては，このような解析条件において，解析時間はごくわずかづつ直線的に増加すると考えられる．本システムにおいて解析に要する時間の変換を測定することで，本システムの解析アルゴリズム以外の部分のオーバーヘッドについて考察する．

本測定の結果を図 6.14 に示す．この結果に示されているように，時刻フィルタを 0 個から 15 個に増加させても，わずかな所要時間の増加しか見られない．よって，このような解析手法の多段接続における本システムオーバーヘッドはわずかなものであると考えられる．

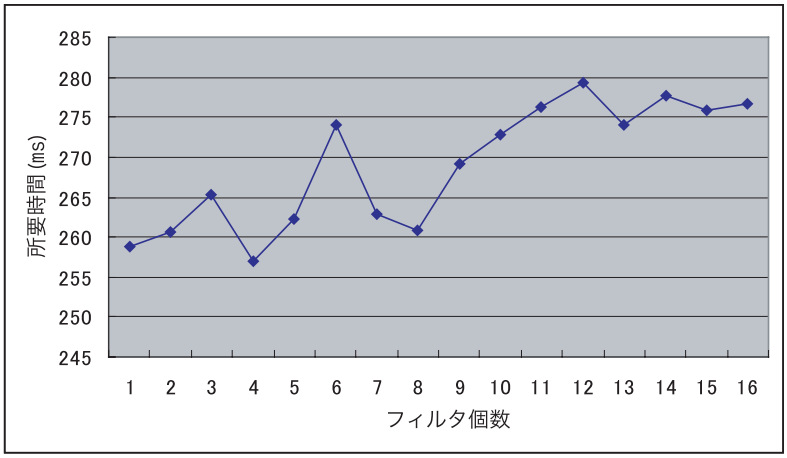

図 6.14: オーバーヘッドの測定

6.4.2 行動履歴の解析時のオーバーヘッドに関する考察

測定で示したように，単純な構造の解析部品の場合，多段に接続してもほとんどパフォーマンスの低下が見られなかった．今後，複雑なアルゴリズムを含む複数の解析部品を，直列，並列を混在させ接続したようなより実際の使用状態に近い環境においてオーバーヘッドを測定する必要がある．

6.5 本章のまとめ

本章では，5 章で設計した MUSE システムの性能を，データの読み込み性能，単純な構成における性能，行動履歴の解析時の性能という 3 つの観点から測定，評価した．データの読み込みにおいては，現在のところ十分な性能が得られているが今後大量のデータを取り扱う場合の問題が浮かび上がった．また単純な構成においては，地図の描画の性能の問題や，情報を重ねて描画したときの性能低下の問題が明らかになった．行動履歴解析のパフォーマンスは，基本的な構成においては十分なパフォーマンスがえられたが，今後より実際的な構成において性能を測定する必要がある．今後これらの問題を念頭によりシステムの性能を向上させる必要がある．

次章は，本論文を結論する．

第7章　結論

7.1　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いて取得が可能になるさまざまな行動履歴に対し，統合的な取り扱いや解析を実現するツールである MUSE システム構築した．MUSE システムでは，移動履歴だけでなく買い物や写真撮影，また駅名で表示された鉄道の乗車記録などさまざまな行動履歴を，システムで定義された表現形式に変換することですべて同一のシステム上に入力することを実現した．MUSE システムは，こうして入力した多様な行動履歴の解析をビジュアルプログラミングインタフェースを用いた対話的な手法で実現し，コンテクストアウェアネスを提供するさまざまなアプリケーションで必要となるさまざまな行動履歴の解析手法の開発を実現する．本システムでは，解析手法が部品化されているので，新しい解析手法の導入も容易である．この柔軟なメカニズムにより，単一のアプリケーションを想定した行動履歴の解析手法では実現できない多様なアプリケーションでの行動履歴の活用を実現する．また，コンピュータ技術の発達によって今後ますます増加すると考えられるさまざまな形式の行動履歴に柔軟に対応する．

本論文において設計，実装した MUSE システムは，システムへのデータの読み込みや GPS で取得したデータの選別のような解析時に，十分なパフォーマンスを示した．

7.2　今後の課題

多様な ActionFilter とアプリケーションの実装

本システム上で動作する ActionFilter として，より多様な解析手法やデータの入出力方法を実装し，本システムを用いて実用的なアプリケーションを構築する．現在，MUSE システムの骨格部分は完成しており，イベントやデータの配送などは完全に機能する．今後は，本システムを行動履歴を解析するアプリケーションの開発環境として公開し，さまざまな人による本システムを利用したアプリケーション開発を促進する．また，本格的なアプリケーションの構築から得た経験やシステムの振る舞いの評価などから，本システムを改良する．

ネットワークへの対応

本システムは現在のところ単独の GUI アプリケーションとして設計されている．行動履歴や地理情報などがネットワークに分散され保存されていたり，解析結果をネットワークを経由して他の分散アプリケーションに転送するような場合，本システム自体もたとえば個々のコンポーネントをネットワークに分散させて，分散システムとして構築したほうが柔軟なシステム構築が行えるかもしれない．今後，このように GUI ベースのアプリケーション以外のシステム構築方法を検討する．

行動履歴の共有を目指した研究

現在，本システムでは単一ユーザの行動履歴を想定して設計されている．多人数の行動履歴を考慮した場合は，行動履歴間の比較を可能にするデータの正規化の問題や，プライバシの問題などを考慮する必要が出てくる．今後，こうした点に留意しながら，複数人数を想定したシステムを構築について考察し，本システムに改良を加えてゆく．

謝辞

本研究を進めるにあたり，御指導頂き，貴重な助言を下さった慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝致します．また，本論文の副査として貴重な助言を頂いた慶應義塾大学環境情報学部教授清木康博士ならびに慶應義塾大学大学院政策・メディア研究科助教授高汐一紀博士に深く感謝致します．

折りにふれ適切な助言を頂き，本論文の執筆を支援して頂いた慶應義塾大学徳田研究室の諸先輩方，後輩の皆様に対し，感謝の意を表します．

平成 16 年 1 月 14 日

伊藤昌毅

参考文献

[1] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context-aware tour guide, 1997.

[2] Gregory D. Abowd. Software Engineering Issues for Ubiquitous Computing. In *proceedings of ICSE'99*, pp. 13–19, April 1999.

[3] Daniel Ashbrook and Thad Starner. Learning Significant Locations and Predicting User Movement with GPS. In *Sixth International Symposium on Wearable Computers(ISWC 2002)*, pp. 101–108, October 2002.

[4] Ken Camarata, Ellen Yi-Luen Do, Brian R. Johnson, and Mark D. Gross. Navigational blocks: navigating information space with tangible media. In *Proceedings of the 7th international conference on Intelligent user interfaces*, pp. 31–38. ACM Press, 2002.

[5] Guanling Chen and David Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[6] Cycling'74. Max/MSP. http://www.cycling74.com/products/maxmsp.html.

[7] Svetlana Domnitcheva. Location Modeling: State of the Art and Challenges. In *Proceedings of the Workshop on Location Modeling for Ubiquitous Computing*, pp. 13–19, September 2001.

[8] Environmental Systems Research Institute Inc. ArcGIS. http://www.esri.com/software/arcgis/index.html.

[9] Environmental Systems Research Institute Inc. *ESRI Shapefile Technical Description*. An ESRI White Paper. July 1998.

[10] Garmin Ltd. Garmin eTrex Legend, 2001. http://www.garmin.com/products/etrexLegend/.

[11] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings of ACM SIGMOD Conference on Management of Data*, pp. 47–57, 1984.

[12] Marios Hadjieleftheriou. Spatial Index Library, 2002. http://www.cs.ucr.edu/ marioh/spatialindex/.

[13] Jeffrey Hightower and Gaetano Borriello. Location Systems for Ubiquitous Computing. *IEEE Computer*, Vol. 34, No. 8, pp. 57–66, August 2001.

[14] Masaki Ito, Akiko Iwaya, Masato Saito, Kenichi Nakanishi, Kenta Matsumiya, Jin Nakazawa, Nobuhiko Nishio, Kazunori Takashio, and Hideyuki Tokuda. Smart Furniture: Improvising Ubiquitous Hot-spot Environment. In *3rd International Workshop on Smart Appliances and Wearable Computing*, May 2003.

[15] Changhao Jiang and Peter Steenkiste. A Hybrid Location Model with a Computable Location Identifier for Ubiquitous Computing. In *Proceedings of the 4th international conference on Ubiquitous Computing*, pp. 246–263. Springer-Verlag, 2002.

[16] U. Kubach and K. Rothermel. Exploiting Location Information for Infostation-Based Hoarding. In *Proceedings of the 7th ACM SIGMOBILE Annual International Conference on Mobile Computing and Networking (MobiCom 2001)*, pp. 15–27, 2001.

[17] Michael G. Lamming and William M. Newman. Activity-based Information Retrieval: Technology in Support of Personal Memory. *Information Processing*, Vol. III, pp. 68–81, 1992.

[18] Bernhard Seeger Norbert Beckmann, Hans-Peter Kriegel Ralf Schneider. The R*-tree: An Efficient and Robust Access Method For Points and Rectangles. In *Proceedings of ACM SIGMOD Conference*, pp. 322–331, 1990.

[19] Donald J. Patterson, Lin Liao, Dieter Fox, and Henry Kautz. Inferring High-Level Behavior from Low-Level Sensors. In *Proceedings of The Fifth International Conference on Ubiquitous Computing (UBICOMP)*, pp. 73–89, 2003.

[20] SAS Institute Inc. SAS Enterprise Miner. http://www.sas.com/offices/asiapacific/japan/software/enterp.html.

[21] Sony Corporation. FeliCa. http://www.sony.co.jp/Products/felica/.

[22] SPSS Inc. SPSS Clementine. http://www.spss.co.jp/product/clementine/cgp.html.

[23] Peter Stopher, Philip Bullock, and Qingjian Jiang. GPS, GIS and personal travel surveys: an exercise in visualisation. *25th Australasian Transport Research Forum Incorporating the BTRE Transport Policy Colloquium*, October 2002.

[24] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. In *ACM Transactions on Information Systems, vol 10*, pp. 91–102, January 1992.

[25] M. Weiser. The Computer for the 21st century. *Scientific American*, Vol. 265, No. 3, pp. 66–75, September 1991.

[26] Michael G. Lamming William M. Newman, Margery A. Eldridge. PEPYS: Generating Autobiographies by Automatic Tracking. In *Proceedings of ECSCW '91*, pp. 175–188, September 1991.

[27] Jean Wolf, Randall Guensler, and William Bachman. Elimination of the travel diary: An experiment to derive trip purpose from GPS travel data. *Notes from Transportation Research Board, 80th annual meeting*, January 2001.

[28] (株) 数理システム. Visual Mining Studio. http://www.msi.co.jp/vmstudio/.

[29] 国土交通大臣　林寛子. 国土交通省告示第九号, 2002. http://www.gsi.go.jp/LAW/heimencho.html.

[30] 伊藤昌毅, 徳田英幸. ユーザの行動を反映した位置履歴表示システムの構築. 情報処理学会マルチメディア、分散、協調とモバイルシンポジウム (DICOMO2003), June 2003.