卒業論文 2002 年度 (平成 14 年度)

ロケーションモデル動的生成機構の構築

指導教員

慶應義塾大学環境情報学部

徳田　英幸

村井　純

楠本　博之

中村　修

南　　政樹

慶應義塾大学 総合政策学部

村上 朝一

tomo@sfc.wide.ad.jp

卒業論文要旨　　2002年度(平成14年度)

ロケーションモデル作成支援機構の構築

本論文では，計算機器が**位置情報**を利用する際に有効な現実世界のモデル，**ロケーションモデル**の作成支援機構を提案する．本研究ではグラフ表現を用いたロケーションモデルをユーザやオブジェクトの移動履歴から動的に生成し，システム管理者によるモデル作成を支援する．本機構がロケーションモデル作成を支援することにより，位置情報を利用するシステムの導入，管理コストを下げられる．

近年の情報科学の進歩により，様々な計算機器が人々の生活空間に遍在する**ユビキタスコンピューティング環境**が実現しつつある．このような環境では計算機器が，人や物の位置情報を利用し協調動作することにより，新しいサービスを実現したり，ユーザと計算機器のインタラクションを減らすなど，ユーザの利便性を向上できる．計算機器が利用する位置情報にはセンサ取得データとロケーションモデルの２種類がある．センサ取得データは位置情報センサによって取得され，対象が存在する空間を示す．また，ロケーションモデルは空間同士の位置関係や距離を表現した現実世界のモデルである．

従来の位置情報管理システムでは，システム管理者が位置情報センサを設置した環境の情報をシステムに入力し，ロケーションモデルを作成する必要があった．しかし今後，位置情報センサが普及し，センサが認識可能な空間が増えると予想される．また一般家庭での位置情報システムの利用を考えると，ロケーションモデルの作成コストを下げる必要がある．また，ロケーションモデル作成のコストは，位置情報システムの導入，管理コストであり，これを下げることにより，位置情報システムの普及が進む．

本研究ではロケーションモデル作成を支援する機構を提案する．本論文ではグラフ表現を用いたモデルをユーザやオブジェクトの移動履歴から動的に生成する機構を構築する．本機構を用いることにより，システム管理者やユーザが空間の位置関係について明示的な入力をせずに，モデルを作成できる．これにより、システム管理者のモデル作成コストを下げられる．

本論文では，まず位置情報とロケーションモデルについて概要を整理する．次にグラフ表現を用いたモデル，**G-LoM** の動的生成について述べる．その後，モデルの動的生成を実現する **NiSMo(Node integrated Space Model)** の設計と実装について説明する．最後に本システムを評価し，全体をまとめる．

慶應義塾大学　総合政策学部
村上 朝一

Abstract of Bachelor's Thesis

Dynamic Location Model management

This thesis presents a location model generation system which produces a location model of a real world for computer usage. The system generates the location model expressed as a graph dynamically from the ambulation record of users and objects instead of system administrators. Through the dynamic generation of the location model, the system can reduce the installation and management costs of location-aware software systems.

With the breakthrough of the information technology, we are beginning to have an environment where computers are pervasively available. In such an environment, coordination of the devices adaptive to the location of humans and objects can enhance convenience of users, since the location-adaptive coordination are creating new types of software services with reduced interaction cost between the users and the computers. There are two kinds of the location information for computers: data acquired by sensors and location model. The former points a space where users and objects reside, while the latter represents a real world with relative location or distance among spaces.

In the conventional location management system, the system administrators need to input location information to generate a location model. However, since more and more sensors are embedded in our living environment, we need to reduce the cost of location model generation. Furthermore, reducing the cost of installation and management of location model generation system leads to the prevalence of location information system. The proposed system eliminates the necessity of inputting location information since it acquires the information from various sensors and dynamically generates a location model in a graph expression from ambulation record of users and objects.

This thesis, first, clarifies the relationship of the location information and the location model. It, then, presents G-Lom which is a location model expressed as a graph, and describes the design and implementation of NiSMo (Node integrated Space Model): a dynamic location model generation system. Finally, it shows evaluation of the system and concludes.

Tomokazu Murakami
Faculty of Policy Management Keio University

目 次

図 目 次

表 目 次

第1章　序論

1.1 本研究の背景

近年，情報科学の進歩により，**ユビキタスコンピューティング環境** [13] が実現されつつある．ユビキタスコンピューティング環境では，様々なセンサや情報家電などの，多機能化，小型化を遂げた情報機器がネットワークに接続し，人々の生活空間に遍在している．このような環境において，複数の情報機器が多様なセンサによって取得された情報を利用し，協調動作することにより，人々の利便性や安全性を向上させられる．

上述の環境では，計算機器が人や物の**位置情報**を利用し，多様なサービスを実現できる．現在，人や物の位置情報を取得するセンサが数多く開発され，次第に普及している．また，位置情報センサの普及に伴い，人や物の位置情報を用いたアプリケーションや位置情報を管理するミドルウェアが数多く開発されている．

アプリケーションは位置情報を利用することにより，ユーザの現在位置に最適化したサービスを提供したり，ユーザと計算機器とのインタラクションを減らすなどして，ユーザの利便性を向上できる．位置情報を利用するアプリケーションの具体例を次に挙げる．まず，美術館での適応例が挙げられる．アプリケーションは来館者の位置情報を利用することにより，来館者に展示品の説明や順路の案内を提供したり，美術館の職員に入館者状況を提供できる．次に，オフィスや大学のキャンパスなどでの適応例が挙げられる．アプリケーションは従業員や学生など，ユーザの位置情報を利用することにより，ユーザに対し最も近いプリンタの情報を提供したり，ユーザの接近によるコンピュータ端末へのログインを可能にする．

今日，多くの人々にユビキタスコンピューティング環境が我々の生活にもたらす利便性が認識されつつある．また，情報家電やセンサなど，計算機器の開発がさらに進み，高機能化，低価格化が実現すると考えられる．さらに，様々な計算機器を協調動作させるためのミドルウェアやアプリケーションなどのソフトウェアの研究開発が進んでいる．以上の要因から，今後このような計算機環境が一般に普及すると考えられる．

ユビキタスコンピューティング環境の普及は，位置情報を利用したアプリケーションの適応範囲を広げ，多様な状況，用途で位置情報の利用が可能になる．今後，位置情報を利用するソフトウェアの開発は重要な課題である．

1.2 問題意識

ソフトウェアが利用する位置情報は 2 種類ある．１つは位置情報センサが取得する情報である．本論文では位置情報センサが取得した情報を**センサ取得データ**と定義する．もう１つは**ロケーションモデル**である．ロケーションモデルとは，位置を表現するための現実世界のモデルである．

センサ取得データは人や物が位置する空間の識別子であり，ロケーションモデルはセンサ取得データの示す空間同士の位置関係を表現する．センサ取得データとロケーションモデルの関係について，図 1.2 に具体的なアプリケーション例を示す．企業のオフィスや大学のキャンパスなど，プリンタが複数存在する環境で，ユーザの位置情報を利用してユー

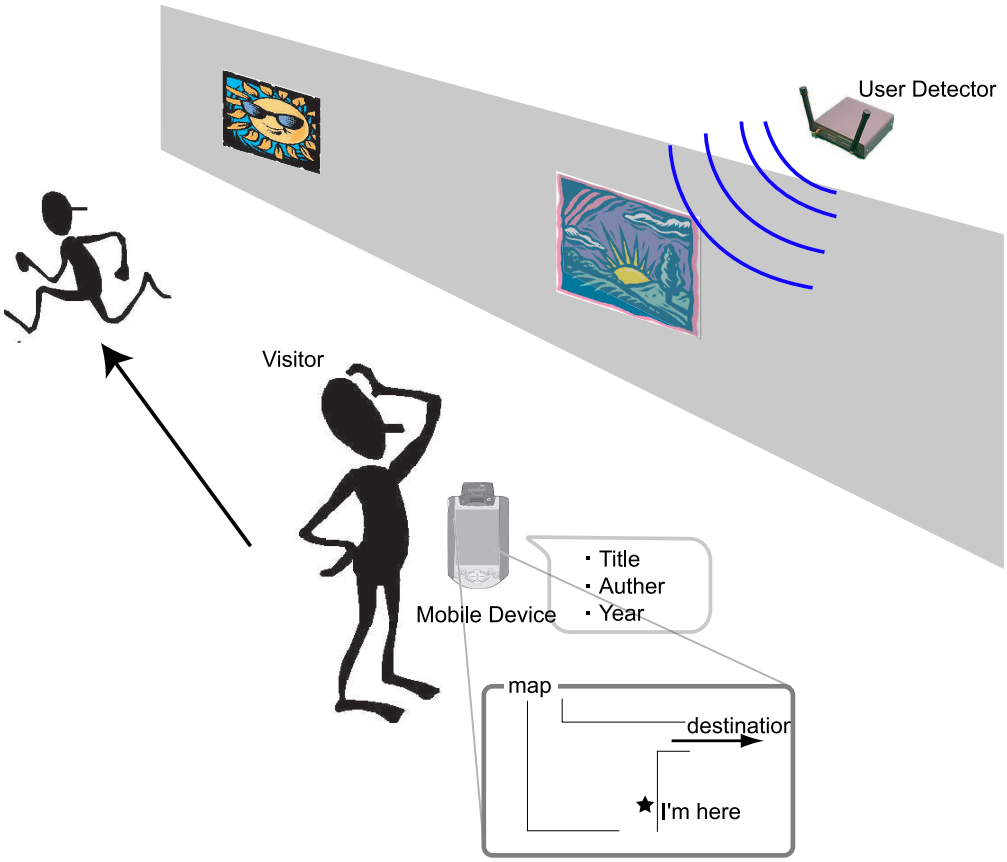

図 1.1: 美術館における位置情報の利用例

ザに最も近接するプリンタを検索する場合を考える．ユーザに最も近接するプリンタを検索する為には，2 種類の情報が必要である．まず，位置情報センサにより取得されるユーザの現在位置を示す情報が必要である．次に，利用可能なプリンタの場所と，ユーザの現在位置からプリンタある場所までの距離の情報が必要である．前者がセンサ取得データ，後者がロケーションモデルである．

一方，このようなモデルを使用せず，センサ取得データのみを使用するアプリケーションが考えられる．例えば，ある特定の位置情報センサがユーザを検出した時に，あらかじめ決められた照明を点けたり，ドアの鍵を開けたりするアプリケーションが考えられる．しかしこの場合，位置情報センサ同士の位置関係や距離に関する情報がないため，ロケーションモデルを使用するアプリケーションに比べ適応範囲が狭く，柔軟性が低い．

ロケーションモデルは，アプリケーションで利用する前に作成する必要がある．また，ロケーションモデルは位置情報センサを設置した環境のモデルであるため，システムの開発者が作成するのではなく，位置情報センサを設置した環境についての情報を持っているシステム管理者が作成しなければならない．

ロケーションモデルを作成するためには，システム管理者が位置情報センサを設置した空間の識別子や空間同士の位置関係などについての情報を入力しなければならないため，

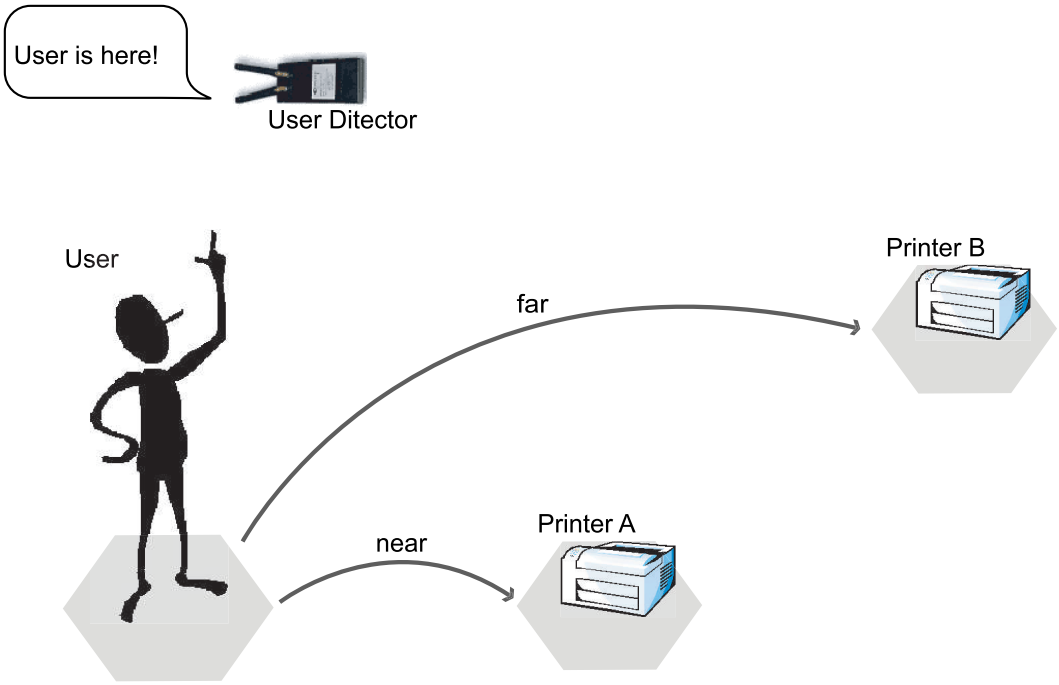

図 1.2: センサ取得データとロケーションモデル

システム管理者にとって負担である．また，位置情報を用いたアプリケーションを一般の家庭で用いる場合，システム管理者は計算機器に関する一般的な知識を持たないユーザ自身である．この場合，ユーザ自身が位置情報センサを設置した家庭の情報を入力しなければならない．

1.3 目的と意義

本研究の目的は，ロケーションモデルを作成するユーザの手間を軽減することである．本研究ではロケーションモデルの作成を動的に行う **NiSMo(Node integrated Space Model)** を構築する．本システムを用いることにより，計算機器に関する一般的な知識を持たないユーザが容易にロケーションモデルを作成できる．

また，本システムはアプリケーション開発者に対して，本システムの生成したモデルを利用する為の API を提供する．アプリケーション開発者はこの API を用いることにより，ロケーションモデルを利用するアプリケーションを容易に開発できる．

本研究は位置情報を用いたアプリケーションの普及を容易にする．本システムを用いることにより，アプリケーションを利用する前に必要なロケーションモデル作成の手間が軽減され，位置情報を用いたアプリケーションの利用が容易になる．

1.4 本論文の構成

本論文では，第 2 章において，ロケーションモデルと位置情報センサについて詳しく説明し，位置情報を用いた先行システムについて述べる．第 3 章ではまず，本研究の目的と機能要件を整理し，次に本研究の用いるロケーションモデルについて，さらにモデルの生成，管理について詳しく述べる．第 4 章でロケーションモデルを動的に生成し，管理する NiSMo の設計について，第 5 章で実装について述べる．そして，第 6 章で NiSMo を評価し，第 7 章で本論文をまとめる．

第2章　ロケーションモデル

本章ではまず，ユビキタスコンピューティング環境における位置情報の利用と，ロケーションモデルの利用について述べる．次にセンサ取得データとロケーションモデルをそれぞれ分類し，分類の対応関係を述べる．さらに，ロケーションモデルの分類とアプリケーションの関係を述べる．最後に，分類したロケーションモデルを用いた先行システムについて説明する．

2.1　モデルの利用

本節ではロケーションモデルの利用について述べる．まず，ユビキタスコンピューティング環境における位置情報の利用について述べ，次にセンサ取得データとロケーションモデルについて述べる．

2.1.1　位置情報の利用

様々な計算機器が環境に遍在するユビキタスコンピューティング環境において，計算機器がユーザを取り巻く状況の情報を利用し協調動作することにより，ユーザに計算機器の操作や存在を意識させずに，利便性を向上させることができる．このようなユーザを取り巻く状況に関する情報を**ユーザコンテクスト**と呼び，ユーザコンテクストを利用するソフトウェアを**コンテクストアウェアアプリケーション**あるいは**状況適応型アプリケーション**と呼ぶ．図 2.1 は状況適応型アプリケーションを示す．

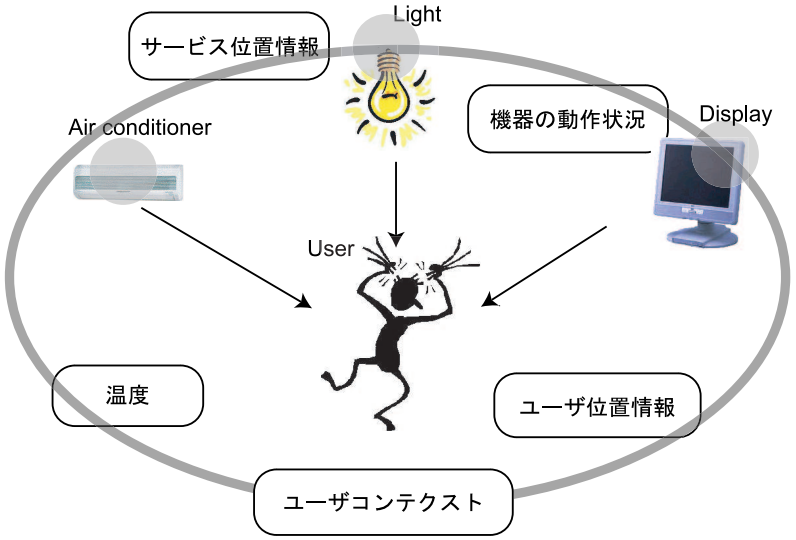

図 2.1: 状況適応型アプリケーション

ユーザやユーザの周囲に存在する計算機器などの位置情報は，ユーザの重要なコンテクストである．本論文ではユーザの周囲に存在し，ソフトウェアがその位置情報を利用する対象を**オブジェクト**と呼ぶ．ソフトウェアがユーザやオブジェクトの位置情報を利用し，ユーザの周囲に存在する計算機器を協調動作させることにより，様々な局面でユーザの活動を支援できる．

2.1.2 位置情報とモデル

位置情報には，位置情報センサが取得した情報であるセンサ取得データと，計算機器上で位置情報を扱うために現実世界を抽象化したモデルであるロケーションモデルの２種類がある．

ロケーションモデルは空間同士の位置関係や距離などを表現する．そのため，アプリケーションはロケーションモデルを利用することにより，センサ取得データのみを利用する場合に比べ，複数の位置情報センサの検出範囲を合わせた空間で位置情報を管理する機能や空間間の距離を比較する機能などの実現可能な機能が増える．

例えば，ユーザに最も近接するサービスを検索するアプリケーションや，ユーザを目的地まで案内するナビゲーションシステムなどが考えられる．

2.2 モデルの分類

ロケーションモデルには位置の表現方法によって様々な形式があり，センサ取得データの形式や利用するアプリケーションにより，用いる形式が異なる．[4]

本節ではまず，位置情報センサとセンサ取得データの形式を分類する．次に，ロケーションモデルを位置の表現方法で分類し，分類したセンサ取得データに対応させる．さらに，分類したロケーションモデルと，それを利用するアプリケーションの関係について述べる．

2.2.1 センサ取得データの形式

センサ取得データは位置情報センサに依存する．位置情報センサには様々なものがあり，取得するセンサ取得データも多様である．本項ではまず，位置情報センサについて述べ，次にセンサ取得データを分類する．

位置を取得する方法は多数ある．以下にその例を列挙する．まず，位置情報センサの例として，GPS（Global Positioning System），RF-ID[9]，Active Bats[12]，Active Badges[11][10] や Cricket[7] などが挙げられる．また，椅子に圧力センサを設置し，ユーザが座っているという情報を取得したり，ドアに磁気センサ，天井にマイクを設置することにより，その空間にユーザがいるという位置情報を取得できる [8]．さらに，無線 LAN，Bluetooth や IrDA など計算機器間の通信を目的とした技術を利用し，電波強度を測定することにより計算機器の位置を取得できる．その他に，カメラで撮影したビデオ画像を解析して位置情報を取得できる [1]．本論文ではこれらの位置情報を取得する全てのデバイスを位置情報センサと定義する．

以上に挙げたように位置情報センサは多数存在し，各々の位置情報センサによって取得するセンサ取得データの形式が異なる．センサ取得データは位置を表現する方法で大きく次の２つに分類できる．位置の表現方法には包含関係で表現するものと位置を座標で表現するものがある．以下に各表現方法の詳細を説明する．

包含表現

包含表現とは，空間毎に識別子を付け，位置をユーザやオブジェクトの含まれる空間の識別子で表現する方法である．位置情報センサの検出範囲内にユーザやオブジェクトの対象が存在するか，否を取得する．検出範囲内でのユーザやオブジェクトの場所は取得できない．

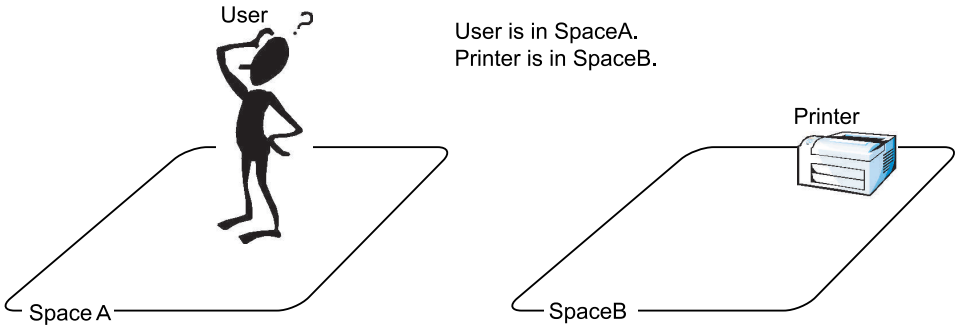

図 2.2: センサ取得データの形式：包含

位置取得センサ

包含表現のセンサ取得データを用いる位置情報センサの例として以下を挙げる．

- RF-ID

  タグが無線電波で識別子を発信する．アンテナが電波を検出し，検出範囲無内にあるタグの識別子を取得する．タグをユーザに持たせ，オブジェクトに取り付けることにより，アンテナの検出範囲内に存在するユーザとオブジェクトの識別子が取得できる．

- Active Badges

  タグが赤外線で識別子を発信する．受光器が赤外線を検出し，検出範囲内にあるタグの識別子を取得する．ユーザにタグを持たせることにより，検出範囲内にいるユーザの識別子が取得できる．

- 無線 LAN を用いた電波強度測定

  ラップトップ型コンピュータや PDA(Personal Digital Assistant) 等の携帯端末に組み込まれた無線 LAN デバイスを利用して電波強度を測定し，基地局との距離を測定する．携帯端末と基地局との距離が取得できるが，基地局から一定強度の電波が届く空間内に存在する携帯端末の識別子が取得できる．あらかじめ，携帯端末の所有者を登録すれば，基地局から一定距離の空間に存在するユーザの識別子を特定できる．

座標表現

座標表現とはユーザやオブジェクトの位置を座標で表現する方法である．ユーザやオブジェクトが位置情報センサの検出範囲内のどこにあるかを検出する．

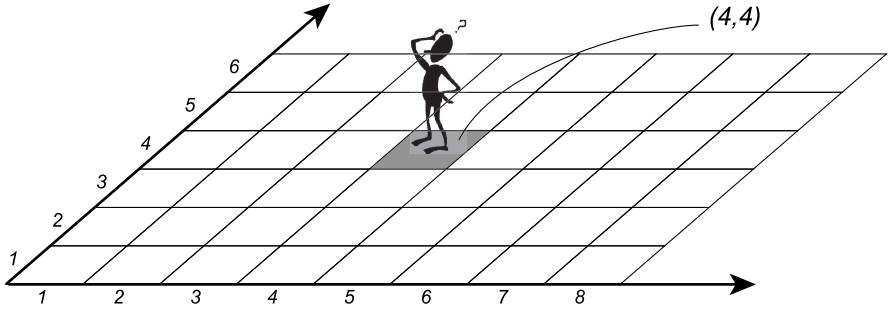

図 2.3: センサ取得データの形式：座標

位置取得センサ

座標表現のセンサ取得データを用いる位置情報センサの例として以下を挙げる．

- GPS

  複数の人工衛星が発信する信号を受信デバイスで受信し，受信器の緯度，経度や高度を算出する．受信デバイスをユーザに持たせた，オブジェクトに取り付けることにより，ユーザやオブジェクトの存在する緯度，経度や高度を取得できる．

- Active Bats

  送信機が発信する超音波を天井に設置された，受信器が受信し，部屋の中での相対的な３次元座標を算出する．超音波の送信機をユーザに持たせ，オブジェクトに取り付けることにより，ユーザやオブジェクトの部屋内での３次元座標値を取得できる．

2.2.2 モデルの種類

ロケーションモデルはユーザやオブジェクトを取り巻く周囲の現実世界をモデル化した情報である．また，前項で詳しく述べた，センサ取得データはユーザやオブジェクトの位置を表現する情報である．センサ取得データの形式によって，位置情報を管理するシステムが用いるロケーションモデルの形式が異なる．

ロケーションモデルの例として，空間をツリー構造で表現するモデル，３次元座標で表現するモデルや２次元座標で表現するモデルなどがあげられる．これらのロケーションモデルは大きく Topological Model と Metric Model の２種類に分類できる．以下に各モデルについて詳細に述べる．

Topological Model

位置情報を，識別子を付けた空間の位置関係で表現するモデルである．Topological Model はそれぞれの空間に識別子を付けて管理するため，比較的大きな粒度の位置情報を表現する場合に有効である．また，空間に識別子を付けて管理するため，人が容易に理解可能な位置情報を提供できる．

Topological Model は比較的大きな空間に識別子を付けて管理するため，包含表現のセンサ取得データを扱うシステムに適する．

Topological Model の例として，空間同士の位置関係をツリー構造で表現したロケーションモデルや，ツリー構造に付加的に，同じ階層に存在する空間同士の相対距離を定義したロケーションモデルなどが挙げられる [15]．ここで，ツリー構造のロケーションモデルを図 2.4 に示す．それぞれの節点は１つの空間を表す．このロケーションモデルは「A棟」には「１階」と「２階」があり，さらに「１階」には「１０１号室」と「１０２号室」があるというように，子の節点が示す空間は，親の節点が示す空間に含まれることを示す．また，人やオブジェクトの位置は節点の識別子で表現する．

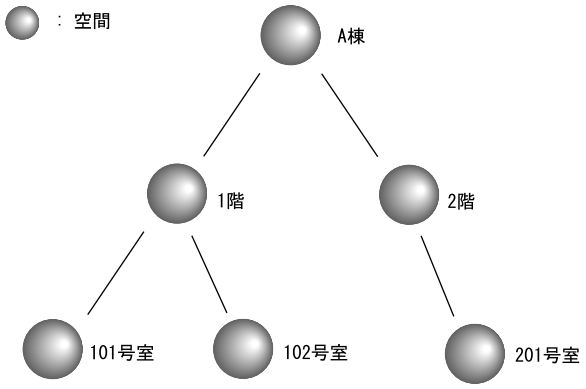

図 2.4: Topological Model の例

ツリー構造で表現したモデル上で空間間の距離は，ある空間から空間へのパスがルート方向へ戻る階層の数で表現する．しかし，戻る階層の数が同じ場合に，どちらの空間がより近いかを比較できない．また，より多くの階層を戻らなければならない空間の方が，少ない階層を戻る空間よりも近い場合が存在する 2.5.

ツリー構造に，同じ階層に存在する空間同士の相対距離を付加的に定義し，同じ階層同士の距離を比較可能したモデルが存在する．しかし，このモデルも後者の問題を解決していない．

Metric Model

位置情報を絶対的，または相対的な座標で表現するモデルである．Metric Model は連続的な数値の組み合わせで空間を表現するため，比較的小さな粒度の位置情報を表現する場

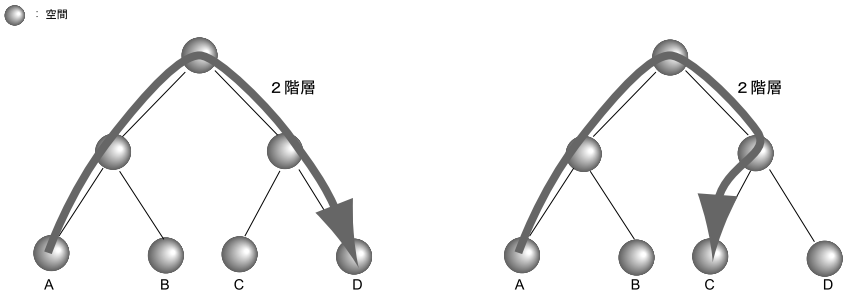

図 2.5: ツリー構造を用いたロケーションモデル上での距離

合に有効である．しかし，Metric Model は位置情報を座標で表すため，表現された位置情報が現実世界のどこを指すのかを人が直感的に理解することが難しい．

Metric Model は座標で空間を表現するため，座標表現のセンサ取得データを管理するシステムに適する．

Metric Model の例として，GPS(Global Positioning System)[5] で利用する地表面を 2 次元の絶対座標で表現したロケーションモデルや，３次元の座標の座標で表現したロケーションモデルなどが挙げられる．ここで，空間を 3 次元の座標で表現したロケーションモデルを図 2.6 に示す．このロケーションモデル上では，位置を 3 つの数字の組で表現する．

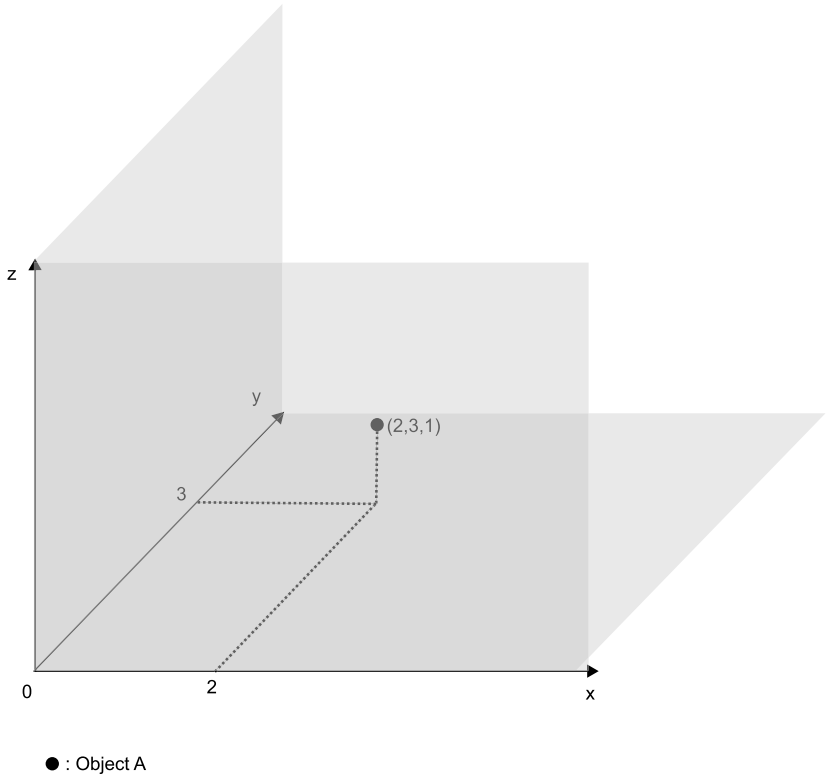

図 2.6: Metric Model の例

表 2.1: 位置取得センサとセンサ取得データの形式

ロケーションモデル	センサ取得データ形式	位置取得センサ
Topological Model	包含	RF-Code Crecket Active Badges 電波強度測定
Metric Model	座標	Active Bats GPS

また，Metric Model 上では 2 点間の直線距離を計算で求められるため，直線距離をあらかじめ定義する必要がない．しかし，2 点間に壁や障害物などが存在する場合の距離を計算する場合は，あらかじめモデル上に壁や障害物の座標を定義する必要がある．

モデルの種類とセンサ取得データ

包含表現のセンサ取得データは Topologica Model に対応し，座標表現のセンサ取得データは Metric Model に対応する．その為，位置情報を管理するシステムが用いるロケーションモデルは，利用する位置情報センサに依存する．ロケーションモデルの種類，センサ取得データ形式と位置情報センサの関係を表 2.1 に整理する．

2.2.3 モデルと利用するアプリケーション

アプリケーションによって必要な位置情報の粒度が異なり，Topological Model と Metric Model では扱う位置情報の粒度が異なる．Topological Model は扱う位置情報の粒度が大きい為，Topological Model を用いたシステムで cm（センチメートル）単位や mm（ミリメートル）単位などの比較的細かい粒度で位置情報を利用するアプリケーションは実現できない．また， Metric Model は扱う位置情報の粒度が小さい為，Metric Model を用いたシステムは部屋単位の比較的大きな粒度の位置情報を利用するアプリケーションの実現には適さない．各々のロケーションモデルによって実現できるアプリケーションが異なる．

Topological Model

比較的粒度の大きい位置情報を扱うアプリケーションに適する．デバイスやサービスの位置情報に基づいた検索や，ユーザの携帯している端末にユーザ周辺の案内を表示するなどのアプリケーションが考えられる．

また，位置情報センサの屋内への設置を考えると，包含表現の位置情報センサは，セン

サデバイスの価格や導入コストが一般に座標表現の位置情報センサと比べて安い．そのため包含表現の位置情報センサを利用する Topological Model は，検出範囲が企業のオフィスビルや大学のキャンパスなどの建物全体をカバーする必要があるアプリケーションでの利用に適している．

Metric Model

比較的粒度の小さい位置情報を扱うアプリケーションに適する．ゼスチャによる機器の制御や人の動作をキャプチャするモーションキャプチャなどのアプリケーションが考えられる．

また，屋内で利用する座標表現の位置情報センサは対象とするオブジェクトの座標を計算する為，包含表現の位置情報センサと比較して，高い密度でセンサデバイスを設置する必要がある．そのため，企業のオフィスや大学のキャンパスなどの建物全体をカバーする必要があるアプリケーションには適さない． Metric Model は検出範囲が比較的狭く，ユーザの細かい動作などを利用するアプリケーションに適している．

2.3 先行システムのモデル

本節では，位置情報を管理する先行システムについて述べる．まず，Topological Model を用いたシステムの例として Active Badges を挙げる．次に Metric Model を用いたシステムの例として，Active Bats を挙げる．さらに，Topological Model のロケーションモデルを作成管理するシステムの例として，Semantic Space[2] を挙げる．

2.3.1 Active Badges

ケンブリッジ大学及びオリベッティ研究所の開発した位置情報管理システムである [11][10].

Active Badge は部屋単位の粒度で位置情報を取得する位置情報センサを用いたシステムである．包含表現のセンサ取得データを取得し，Topological Model を用いて管理する．

Active Badges を用いたアプリケーションの例としては，受付にかかってきた電話のユーザに最も近い電話へ転送，プリンタの出力先を変更するなどがある．

2.3.2 Active Bats

ケンブリッジ大学及びオリベッティ研究所の開発した位置情報管理システムである [12].

Active Bats は約十 cm 単位の粒度で位置情報を取得する位置情報センサを用いたシステムである．座標表現のセンサ取得データを取得し，空間を 3 次元の座標で表現する Metric Model を用いて管理する．

Active Bat を用いたアプリケーション例としては，ユーザのジェスチャによる機器の制御がある．

2.3.3 Semantic Space

マイクロソフト社の開発した位置情報管理システムである．

Semantic Space は使用する位置情報センサは指定していない．部屋単位の粒度で位置情報を取得する位置情報センサを利用する．Topological Model を用いたシステムである．Semantic Spaces はツリー構造のロケーションモデルを作成，管理する．また，ロケーションモデル作成，管理のためのインタフェースとして，空間をフォルダ，空間内にあるオブジェクトをファイルとした，ファイルマネージャのような GUI を提案している．ユーザはこのインタフェースを用いてロケーションモデルを作成管理できる．

Semantic Space を利用したアプリケーション例としては，位置情報を考慮したデバイスやサービスの検索がある．

2.4 本章のまとめ

本章ではまず，位置情報を用いるアプリケーションによるロケーションモデルの利用について述べ，ロケーションモデルの有効性を述べた．次にロケーションモデルを Topological Model と Metric Model に分類した．また，ロケーションモデルの分類とセンサ取得データ，利用するアプリケーションとの関係について述べた．さらに， Topological Model と Metricl Model それぞれを用いた先行システムの例と Topological Model を作成，管理する先行システムの例を挙げた．

次章では，本研究の目的とロケーションモデルの動的生成について詳しく述べる．

第3章　ロケーションモデルの動的生成

本章ではまず，本研究の目的を整理し，目的を実現するためのアプローチについて述べる．次に本研究で生成，利用するロケーションモデルについて述べる．さらに，ロケーションモデルの生成，管理やアプリケーションへの提供について述べる．

3.1 本研究の目的

本節ではまず，位置情報を用いるアプリケーションを利用する際に必要なロケーションモデル作成，管理について述べる．次に本研究の目的である，ロケーションモデル作成の支援について述べる．

前章で述べたように，ロケーションモデルの利用は有効である．しかし，ロケーションモデルは位置情報センサを設置する環境の情報が含まれるため，システム開発者は作成できない．そのため，ロケーションモデルは位置情報センサを設置した環境についての情報を持っているシステム管理者が作成，管理する必要がある．アプリケーション，センサ取得データ，ロケーションモデルとシステム管理者の関係を図 3.1 に整理する．ロケーションモデルの作成，管理はシステム管理者の計算機器に対する入力が必要であるため，システム管理者にとって負担となる．

現在，位置情報センサが設置，利用されている環境は限られており，広く一般には普及していない．しかし今後，ユビキタスコンピューティング環境の普及に伴い，位置情報センサの数は増加すると考えられる．位置情報センサの数が増加し，位置情報センサの検出範囲が広がるとロケーションモデルの作成，管理コストは増加する．

また，位置情報を用いるアプリケーションを家庭で利用する場合，システム管理者はアプリケーションユーザ自身である．アプリケーションのユーザは，システム管理者と異なり，計算機器に関する一般的な知識を持たない場合が多い．アプリケーションのユーザにとって計算機器に対する入力は負担であり，減らす必要がある．

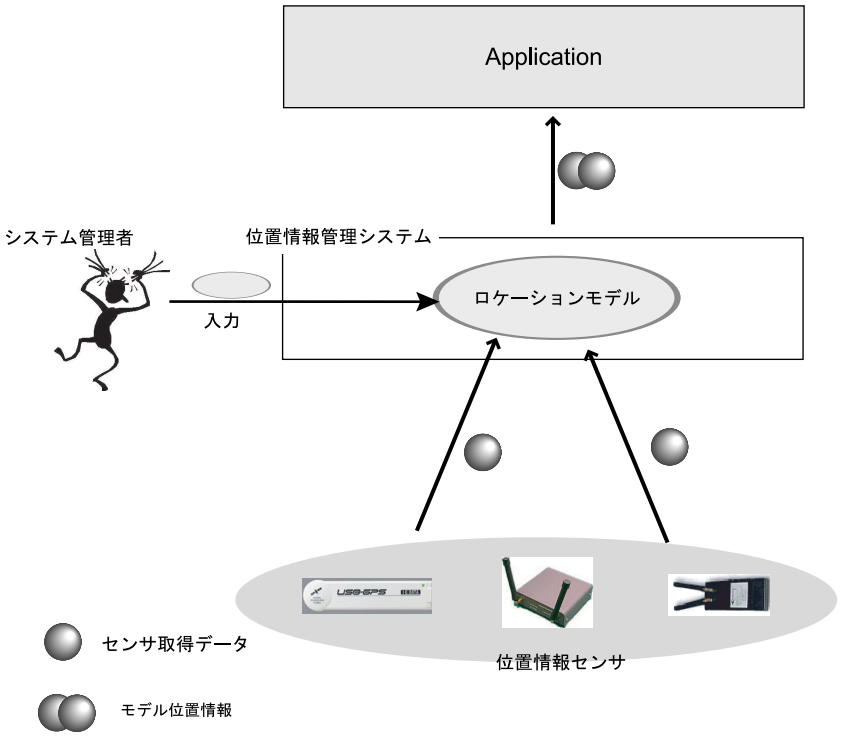

図 3.1: システム管理者によるロケーションモデルの作成

本研究は，システム管理者のロケーションモデル作成支援を目的とする．本研究により，ロケーションモデルの作成が容易になり，位置情報を用いたアプリケーションの導入，管理が容易になる．これにより，一般家庭でアプリケーションのユーザ自身が位置情報システムを導入，管理する際の，ロケーションモデルを入力する負担を減らせる．その結果，本研究は位置情報を用いたアプリケーションの普及を助ける．

3.2 機能要件

本節では，ユビキタスコンピューティング環境において，システム管理者によるロケーションモデル作成を支援し，作成されたロケーションモデルとセンサ取得データを管理し，アプリケーションに提供するミドルウェアの機能要件を整理する．システムの要件はロケーションモデル作成支援機構，ロケーションモデル管理機能，位置情報管理機能，複数アプリケーションによる同時利用可能性及び位置情報センサ非依存性である．以下でそれぞれの機能について述べる．

- ロケーションモデル作成支援機能

  システム管理者による，ロケーションモデルの作成を支援する機能である．本システムがロケーションモデル作成支援機能を実現することにより，システム管理者がロケーションモデルを作成する負担を減らす．

- ロケーションモデル管理機能

  ロケーションモデルを管理する機能である．追加的に位置情報センサを設置した場合や，位置情報センサを撤去した場合など，位置情報センサの配置を変える場合，ロケーションモデルを更新する必要がある．ロケーションモデルを更新する際，システム管理者の入力する情報を減らす．

- 位置情報管理機能

  アプリケーションに対して位置情報を提供する機能である．アプリケーションが位置情報やロケーションモデルを利用するためには，システムがこれらの情報を提供する必要がある．

- 複数アプリケーションによる同時利用可能性

  本システムが管理するロケーションモデルや位置情報を複数のアプリケーションが同時に利用できる必要がある．ユビキタスコンピューティング環境では，複数のアプリケーションが同時に同じ位置情報を利用する可能性がある．

- 位置情報センサ非依存性

  ユビキタスコンピューティング環境では，多様な位置情報センサが複数存在する．本システムは位置情報センサの差異を吸収し，異種の位置情報センサが利用可能である必要がある．そのため，本システムは位置情報センサ非依存である必要がある．

3.3 アプローチ

本研究では第 3.1 節で述べた目的を実現するため，ロケーションモデルを動的に生成，管理するミドルウェア NiSMo(Node integrated Space Model) を構築する．本システムがロケーションモデルを動的に生成することにより，システム管理者のロケーションモデル作成を支援する．また，ロケーションモデルの動的生成にはセンサ取得データを利用する．

本システムはロケーションモデルと位置情報を管理し，アプリケーションに提供する．図 3.2 は本システムによるロケーションモデルの動的生成とアプリケーションへの提供を示す．

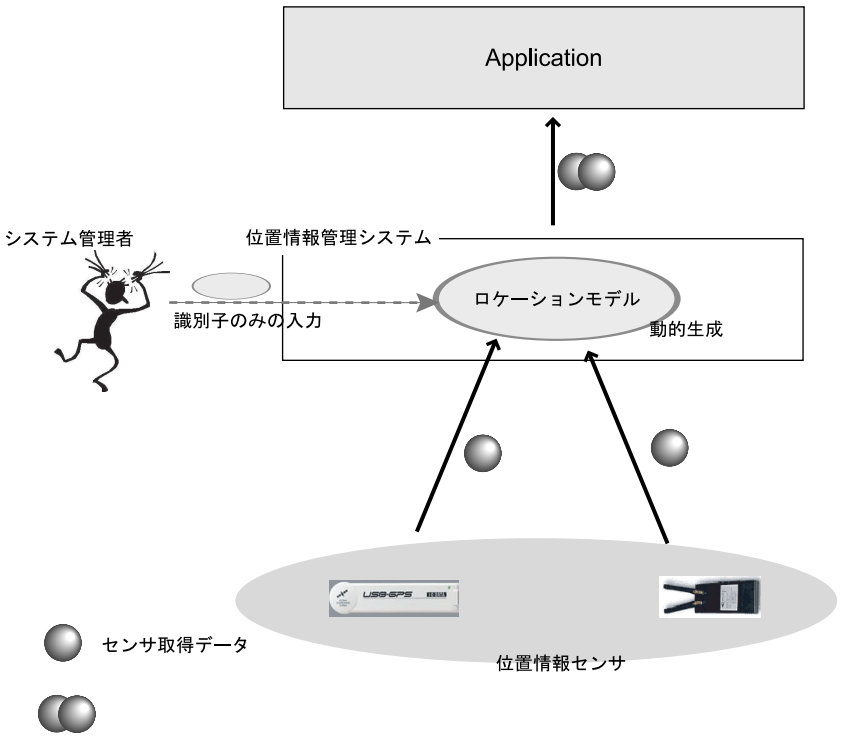

図 3.2: ロケーションモデルの動的生成とアプリケーションへの提供

3.4 生成するモデル

本研究ではロケーションモデルを動的に生成，管理する NiSMo を構築する．本節で NiSMo が用いるロケーションモデルについて詳しく述べる．

3.4.1 グラフ表現

NiSMo ではグラフ表現を用いたロケーションモデル **G-LoM(Graph Location Model)** を生成する．G-LoM では，位置情報取得センサで感知可能な空間を頂点，ユーザが移動

表 3.1: **G-LoM** の要素

要素	表現する対象
頂点	位置情報センサが感知する空間
辺	空間間に存在する経路
重み	空間間の相対距離

可能な経路を辺として抽象化する．また，移動可能な経路の相対距離を辺の重みとする．G-LoM の要素を表 3.1 に整理する．

G-LoM を用いることにより，空間同士の位置関係や距離を表現できる．G-LoM の例を図 3.3 に示す．

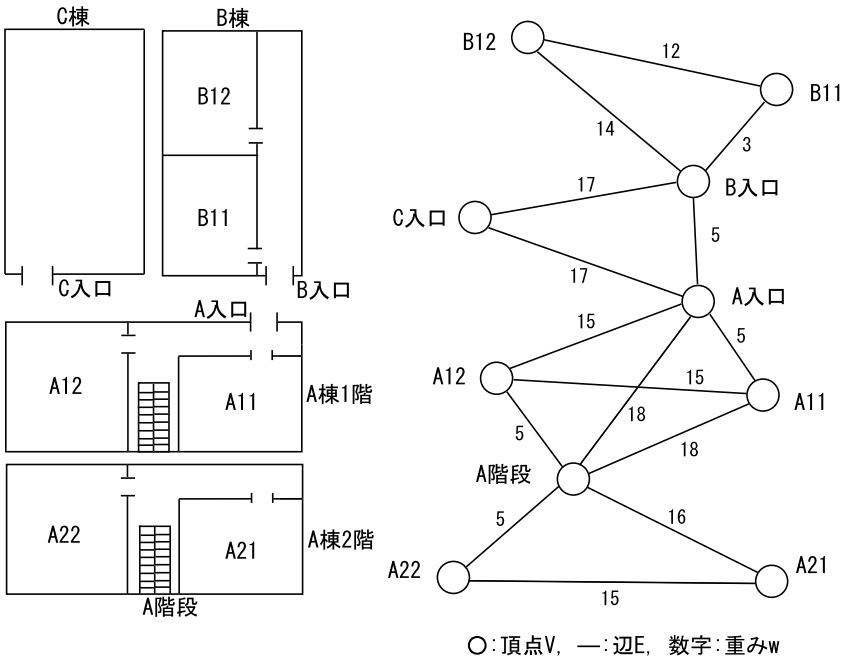

図 3.3: グラフを用いたロケーションモデル **G-LoM**

3.4.2　グラフ表現の利点

G-LoM を用いる利点として粒度の大きい位置情報への適合性，距離の表現と自動生成可能性の 3 点が挙げられる．

- 粒度の大きい位置情報への適合性

G-LoM は空間に識別子をつけ，空間間の位置関係を定義しているため，Topological Model である．そのため G-LoM は包含表現のセンサ取得データの処理に適し，広い検出範囲で比較的大きな粒度の位置情報を利用するアプリケーションに適している．

- 距離の表現力第 2.2.2 項で述べたように，空間間の位置関係をツリー構造で表現したモデルでは，空間間の相対距離を正確に表現できない．しかし， G-LoM は相対距離を辺の重みとしたグラフ表現を用いているため，空間間の相対距離をノード間の最短経路検索で求められる．

  G-LoM は他の Topological Model と比較して空間間の相対距離を正確に表現できる．そのため， G-LoM を用いることにより， Topological Model を利用するアプリケーションで空間間の距離を扱える．

- 自動生成可能性

  既存のロケーションモデルを利用する場合は自動生成できないが，G-LoM の利用によってロケーションモデルの自動生成が可能になる．ロケーションモデルの自動生成が可能になることにより，ユーザのロケーションモデル作成を容易にできる．ロケーションモデルの動的生成について次節で詳しく述べる．

3.5 G-LoM の生成・管理

NiSMo はシステム管理者のロケーションモデル作成を支援するために， G-LoM を動的に生成し，管理する．本節ではまず，ユーザやオブジェクトの移動履歴を用いた G-LoM の動的生成について述べる．次に G-LoM の更新とアプリケーションへの提供について述べる．

3.5.1 移動履歴を用いたモデル生成

NiSMo はユーザやオブジェクトの移動履歴を利用し，G-LoM を生成する．G-LoM には頂点，辺と重みの 3 つの要素がある．以下で NiSMo による G-LoM を構成する 3 要素の生成手順を述べる．

- 頂点

  G-LoM の頂点は位置情報センサが認識可能なひとつの空間である．位置情報センサが環境に設置され，新しく位置情報センサが認知可能な空間が追加されると，本システムがそれを感知し，G-LoM の頂点を作成する．

- 辺

  G-LoM の辺は空間間の移動可能な経路である．本システムは受け取ったセンサ取得データを基に，ある空間間をユーザやオブジェクトが移動したと判断すると，そ

の空間間に移動可能な経路が存在すると判断し，それぞれの空間を示す頂点を辺で繋ぐ．

- 重み

  G-LoM の重みは，頂点で表現された空間間の相対距離である．本研究では，空間間の相対距離をその空間間を**移動にかかる時間**で表現する．本システムは実際に取得されたセンサ取得データから，ユーザやオブジェクトが空間間を移動した履歴を蓄積し，空間間を移動するのにかかった時間を算出する．算出した移動時間を空間間の相対的距離として，辺の重みとする．

NiSMo は以上の手順で，ユーザやオブジェクトの移動履歴から G-LoM を生成する．また，人が理解しやすい空間の識別子が必要であれば，システム管理者が本システムの提供するモデル管理機能を用いて空間に任意の識別子を付けることができる．

3.5.2 モデルの更新

G-LoM の生成には位置情報センサを設置した環境の中で，ユーザやオブジェクトが活動し，空間間を移動した情報を利用する．本システムでは，ユーザやオブジェクトが空間間を移動する度に，その情報を用いて更新する．そのため，本システムを運用中に位置情報センサを追加的に設置した場合や取り外したりした場合に，システム管理者が G-LoM を新しく作成したり，編集する必要がない．

3.5.3 アプリケーションへの提供

本システムは，動的に生成した G-LoM をアプリケーションに提供する．アプリケーションは本システムが提供する API を利用することにより，G-LoM の情報を利用できる．また，本システムは，G-LoM をユーザやオブジェクトが空間間を移動する度に更新し，常に最新の G-LoM をアプリケーションに提供する．

3.6 本章のまとめ

本研究の目的はシステム管理者によるロケーションモデル作成を支援することである．本研究では，ロケーションモデルを動的に生成する NiSMo を構築し，ロケーションモデル作成を支援する．

NiSMo では G-LoM を使用する．G-LoM は空間を頂点，空間から空間への経路を辺，経路の距離を重みとする重み付きのグラフで表現したロケーションモデルである．NiSMo は G-LoM をユーザやオブジェクトの移動履歴から動的に生成する．また，NiSMo は生成した G-LoM や位置情報をアプリケーションに提供する．

次章では，NiSMo の設計について述べる．

第4章　NiSMo の設計

前章では，ロケーションモデル作成支援のための動的生成について述べた．本章ではロケーションモデルを動的に生成し，管理する NiSMo の設計について詳細に述べる．まず，設計方針，システム全体の概要について述べる．次に NiSMo を構成するそれぞれのモジュールについて述べる．最後に本システムのアプリケーション例について述べる．

4.1 設計方針

本研究の目的はロケーションモデルを動的に生成し，管理する NiSMo を構築することである． NiSMo の設計方針として，センサ非依存性，プラットフォーム独立性を挙げる．

- センサ非依存性

  ユビキタスコンピューティング環境では，多種の位置情報センサが設置されていることが想定される．そのため，本システムを位置情報センサ非依存にし，そのような環境で利用可能にする必要がある．ただし，本システムが用いる G-LoM は Topological Model であるため，本システムは座標表現の位置情報を取得するセンサには対応しない．

- プラットフォーム独立性

  本システムの再利用可能性を高めるため，本システムの各モジュールをプラットフォーム非依存にする．

4.2 全体概要

本節ではまず，本研究の想定する環境について述べる．次に NiSMo の概要としてハードウェア構成，システム管理者主体のユースケース，ソフトウェア構成の順に述べる．

4.2.1 想定する環境

NiSMo はオフィスビル，大学のキャンパス，イベント会場や家庭などの複数の部屋が存在する環境を想定している．また，複数の部屋が存在する場所に，部屋単位の粒度，あるいは，部屋単位より細かい粒度で包含表現の位置情報を取得できる位置情報センサが設置されている必要がある．さらに，本システムでは複数の位置情報センサが取得したセンサ取得データをサーバで管理するため，位置情報センサがインターネットなどのコンピュータネットワークに接続され，サーバと通信が可能である必要がある．また，位置情報センサが単体でネットワーク接続性を持たない場合は，PC，Duonus[14] や TINI[3] などの計算機器を介してサーバと通信をする必要がある．

4.2.2 ハードウェア構成

NiSMo では，以下のハードウェア構成を想定している．ハードウェア構成を図 4.1 に示す．

- 位置情報センサ

  位置情報を取得し，センサ取得データを NiSMo サーバに送信する．

  例 RF-ID，Active Badge，無線 LAN 基地局，赤外線通信デバイス

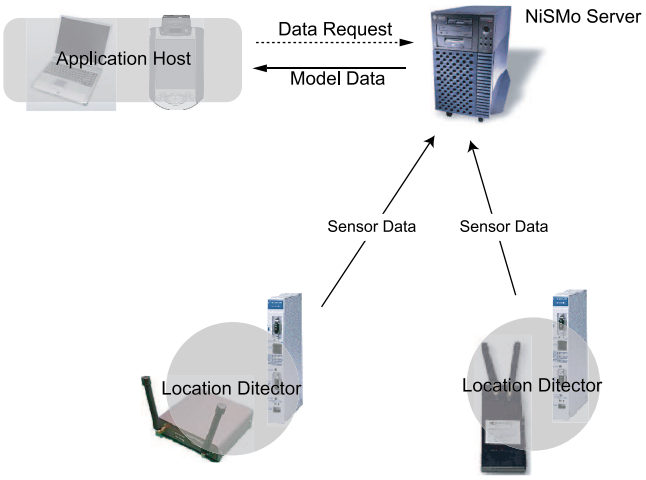

図 4.1: ハードウェア構成

- NiSMo サーバ

  センサ取得データを取得し，ロケーションモデルを生成，管理する．アプリケーション端末の要求に対して位置情報を送信する．

  例 Work Station，PC

- アプリケーション端末

  NiSMo サーバから位置情報を取得する．取得した位置情報を用いたサービスをユーザに提供する．NiSMo サーバと同一ホストである場合がある．

  例 PDA，PC，携帯電話

4.2.3　ユースケース

システム管理者が主体となって行う動作としては，空間識別子の入力と対象オブジェクト識別子の入力がある．対象オブジェクトは，位置を取得する対象である，ユーザやオブジェクトとする．ユースケース図を図 4.2 に示す．

システム管理者はロケーションモデル作成のために，空間同士の位置関係や，空間間の距離などをシステムに入力する必要がない．

アプリケーションが主体となって行う動作としては，モデル情報，履歴情報，移動情報の利用がある．ユースケース図を図 4.3 に示す．

4.2.4　ソフトウェア構成

NiSMo は位置取得部，履歴管理部，モデル管理部及び要求解析部からなる．

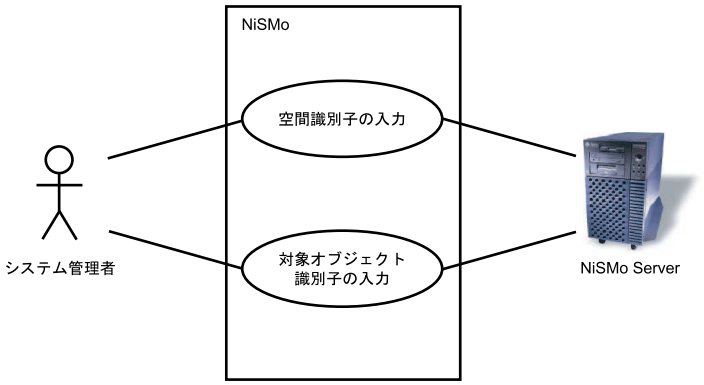

図 4.2: システム管理者主体のユースケース図

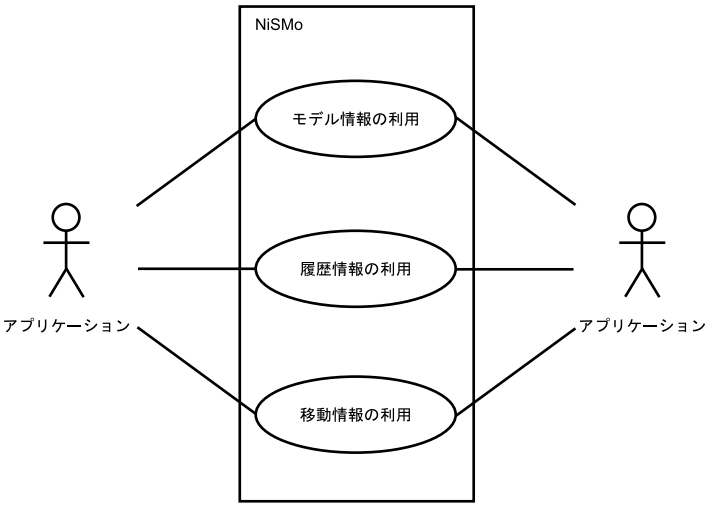

図 4.3: アプリケーション主体のユースケース図

履歴管理部，モデル管理部及び要求解析部は NiSMo サーバ上で動作する．また，位置取得部は，位置情報センサが十分な計算機能とネットワーク接続性を持つ場合は位置情報センサ上で，十分な計算機能，又はネットワーク接続性を持たない場合は位置情報センサが直接繋がれた PC や Duonus などの計算機器上あるいは，サーバ上で動作する．

システム構成を図 4.4 に示す．以下に，各モジュールの機能について述べる．

- 位置取得部

  位置情報センサからセンサ取得データを取得し，対象オブジェクトの移動を検知した時に**移動イベント**を生成し，モデル管理部，履歴管理部及び要求解析部へ配送する．

- モデル管理部

  モデル管理部は G-LoM を管理する．位置取得部から移動イベントを受け取り，その情報を基に G-LoM を更新する．また，要求解析部からの要求に対して，G-LoM

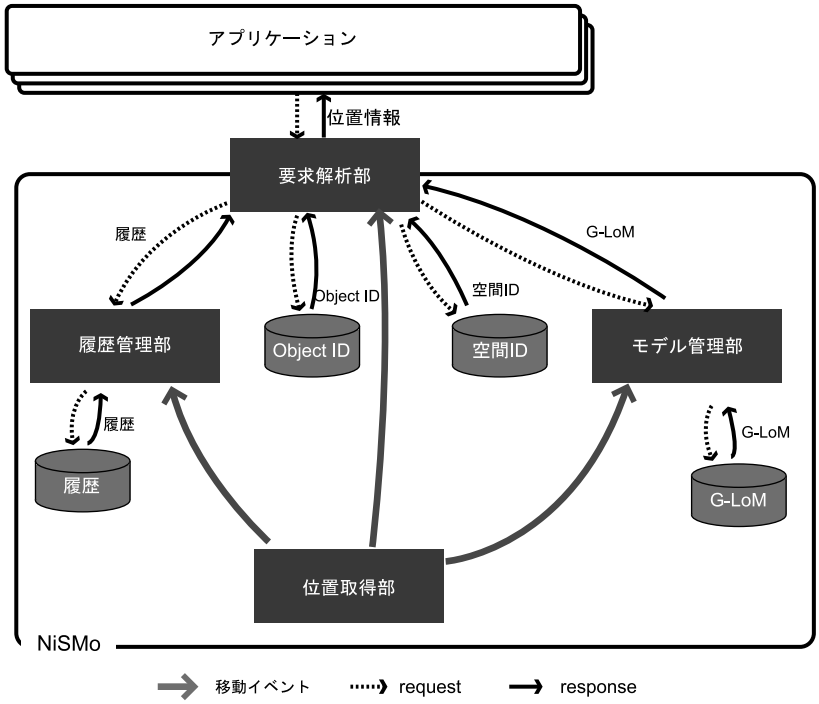

図 4.4: システム構成

の情報を返す.

- 履歴管理部

  履歴管理部は位置情報の履歴を管理する．位置取得部から移動イベントを受け取り，履歴を更新する．また，要求管理部からの要求に対して，履歴情報を返す.

- 要求解析部

  アプリケーションからの要求を受け取り，要求を解析し，位置情報を返す．位置情報を返す際に必要であれば，モデル管理部や履歴管理部に問い合わせる．また，位置取得部から配送された移動イベントを，事前に登録されたアプリケーションに対して配送する．また，空間識別子と対象オブジェクト識別子の登録を管理する．システム管理者の入力に基づいて空間識別子と対象オブジェクト識別子を登録する.

4.2.5 基本動作

前項で述べた各モジュールは次の様に動作する．動作の流れを図 4.5 に示す.

1. 位置取得部が移動イベントを生成し，モデル管理部，履歴管理部及び要求解析部へ配送する.

2. 移動イベントを受け取った各モジュールがそれぞれ以下の処理をする.

- モデル管理部が G-LoM を更新する.
- 履歴管理部が履歴を更新する.
- 要求解析部が事前に登録されたアプリケーションへ移動イベントを配送する.

3. アプリケーションが要求解析部に対して位置情報を問い合わせる.
4. 要求解析部がアプリケーションからの要求を解析し，モデル管理部あるいは履歴管理部に問い合わせ，位置情報を返す.

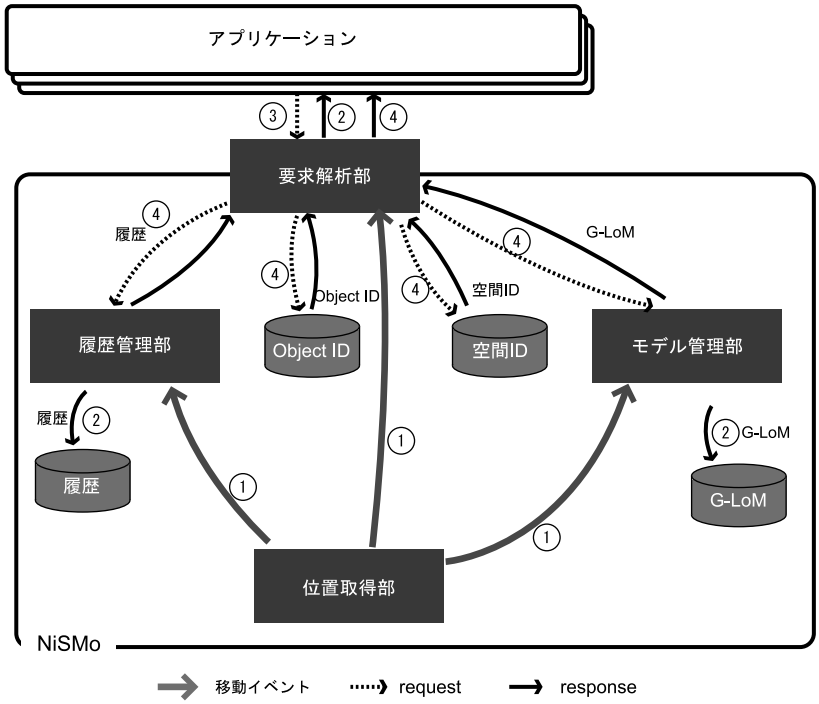

図 4.5: 基本動作

次節以降では，本節で述べた本システムを構成する各部について設計の詳細を説明する.

4.3 位置取得部

位置取得部は位置情報センサが取得したセンサ取得データを受け取り，移動イベントとしてモデル管理部，履歴管理部及び要求解析部に渡す.

システム管理者が位置情報センサを取り付け，ネットワークに接続すると，認識可能な空間が追加されたことをモデル管理部に通知する．これにより，新しい空間の識別子が登録され，モデル管理部の管理する G-LoM に新しいノードが生成される.

位置情報センサから取得したセンサ取得データはすべて，対象オブジェクトが空間内に入ったイベントと空間から出たイベントに変換して扱う．この移動イベントには以下の属

性がある．位置情報センサの種類にかかわらず，センサ取得データをこの属性を持つ移動イベントとして扱うことにより，システムを位置情報センサ非依存にする．

- 対象オブジェクト識別子
- 空間識別子
- IN or OUT
- タイムスタンプ

4.4 モデル管理部

モデル管理部は G-LoM を管理し，位置取得部が生成した移動イベントを基に更新する．移動イベントを受け取り，対象オブジェクトが空間間を移動したと判断すると，移動イベントのタイムスタンプから空間間の移動にかかった時間を算出する．算出した移動時間を用いて，G-LoM 上で空間間の距離を表現する重みを更新する．

また，モデル管理部は要求解析部の要求に対して G-LoM の情報を返す．要求解析部に提供する情報は以下である．

- 空間間を移動するのにかかる時間
- 存在する空間の識別子

4.5 履歴管理部

移動イベントの履歴を管理する．位置取得部が生成した移動イベントを受け取り，移動イベント履歴を更新する．

また，履歴管理部は要求解析部に対して移動イベントの履歴情報を提供する．要求解析部に提供する情報は以下である．

- 対象オブジェクトが存在する空間の識別子
- 特定の空間に存在する対象オブジェクトの識別子

4.6 要求解析部

アプリケーションからの位置情報要求に対して，モデル管理部や履歴管理部に問い合わせ，G-LoM や履歴の情報をアプリケーションに提供する．さらに，あらかじめ要求解析部に登録されたアプリケーションに対して移動イベントを配送する．

また，要求解析部は空間の識別子と対象オブジェクト識別子を管理する．システム管理者の入力に基づいて空間の識別子と対象オブジェクト識別子の登録を行う．

NiSMo はアプリケーションに対して以下の位置情報を提供するための API を用意する．

- 対象オブジェクトの位置

  NiSMo はオブジェクトやユーザの識別子を取得し，その現在位置を返すメソッド及び，空間の識別子を取得し，空間に存在するオブジェクトやユーザの識別子を返すメソッドを用意する．

  また，オブジェクトの識別子と時刻に関する情報を取得し，オブジェクトの過去の位置を返すメソッドを用意する．

- オブジェクトの移動イベント

  オブジェクトやユーザが移動した情報はイベントとしてアプリケーションに配送する．NiSMo はイベントを受け取るリスナを登録するメソッドを用意する．

- 空間同士の距離

  二つの空間の識別子を取得し，その間の距離を返すメソッドを用意する．空間間の距離はユーザやオブジェクトがその空間間を移動するのにかかった時間で表現する．

- 空間識別子の登録・削除

  システム内で管理する空間識別子とは異なる，システム管理者が理解し易い識別子を付ける

- 対象オブジェクト識別子の登録・削除

  システム内で管理するオブジェクト識別子とは異なる，システム管理者が理解し易い識別子を付ける

4.7 アプリケーション例

本システムを用いたアプリケーションの例としては，物理的位置が意味を持つサービスの制御や検索が挙げられる．物理的位置が意味を持つサービスとは，ユーザがサービスを受けられる物理的位置がある一定範囲に決まっているサービスである．物理的位置が意味を持つサービスには，ディスプレイ，スピーカ，照明やプリンタなどがある．本システムを用いてこれらのサービスを位置情報を用いて検索，制御するアプリケーションを利用できる．例えば，ユーザに最も近接するディスプレイを検索し利用するアプリケーションや，ユーザが部屋から出ると家電機器の電源を落とし，鍵をかけるアプリケーションなどが挙げられる．

また，ユーザの持つ携帯端末を利用し，オフィスビルやショッピングセンターなどの建物内でユーザを案内する，ヒューマンナビゲーションシステムもアプリケーションとして挙げられる．

4.8 本章のまとめ

本章では， NiSMo の設計方針として位置情報センサ非依存性，プラットフォーム非依存性を実現することを述べた．次に本システムの全体概要について述べ，その上で，各モジュールの設計について詳細に説明した．さらに，本システムを用いたアプリケーション例を述べた．次章では NiSMo の実装について述べる．

第5章　NiSMo の実装

前章では G-LoM を動的に作成，管理する NiSMo の設計について述べた．本章では NiSMo の実装について述べる．まず，実装の概要について述べ，次に各モジュールの実装について詳しく述べる．

表 5.1: 実装環境

項　　目	環　　境
CPU	Athlon 1.2GHz
Memory	256MB
OS	Windows2000 Professional
JDK	Java 2 SDK, Standard Edition Version 1.3.1-b24

5.1　実装の概要

NiSMo は表 5.1 に示す開発環境で実装した．本システムはモジュールのプラットフォーム独立性を確保するため，Java 言語を用いて実装を行った．

また，今回の実装では位置情報センサとして RF-Code を使用した．RF-Code は単体では Java の実行環境がないため，RF-Code に RS-232C ケーブルで PC を繋げ，PC 上で本システムのモジュールを実装した．実装の概観を図 5.1

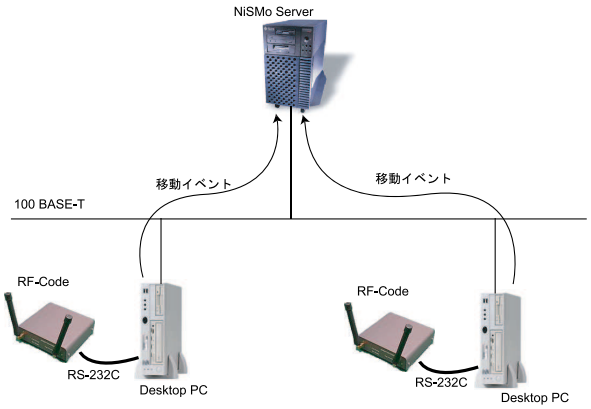

図 5.1: 実装の概観図

次節以降では，NiSMo の各モジュールの主要クラスについて詳細に述べる．

5.2　位置取得部

位置取得部は位置情報センサからのセンサ取得データを受け取り， LocationEvent を生成する．位置取得部のクラスを図 5.2 に示す．

Detector クラスを基底として，各位置情報センサ用のクラスを作成する．Detector クラスはその構成要素として LocationEventCreator クラスを持つ． Detector クラスはセンサ取得データを処理し，ある対象オブジェクトの空間への入出を検出すると，LocationEventCreator クラスのメソッドを呼び出し，LocationEvent を生成する．LocationEvent は他モジュール

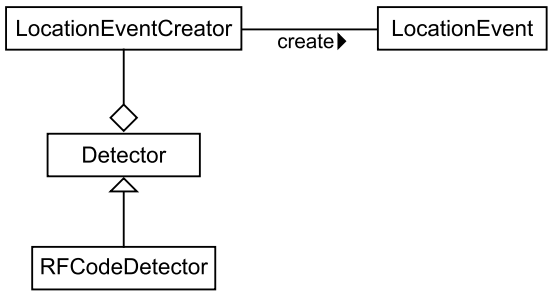

図 5.2: 位置取得部のクラス

表 5.2: LocationEvent クラスの属性

型名	フィールド名	説明
String	spaceID	空間を一意に指す識別子である
String	objectID	対象オブジェクトを一意に指す識別子である
boolean	in_or_out	対象オブジェクトが空間に入ったのか，あるいは出たのかを示す
Timestamp	timestamp	イベントの生成された時刻を示す

の LocationEventConsumer インタフェースを実装したクラスに配送する．以下に Detector クラスのメソッドを示し， LocationEvent クラスの属性を表 5.2 示す.

Detector

final void createLocationEvent(String space_id, String object_id, boolean in_or_out)
LocationEventCreator クラスのメソッドを呼び出し，LocationEvent を生成する

5.3 モデル管理部

モデル管理部は LocationEvent を受け取り，G-LoM を更新する．また，要求解析部の RequestHandler クラスからの要求に対して G-LoM を返す．モデル管理部のクラスを図 5.3 に示す.

ModelUpdater クラスは LocationEventConsumer インタフェースを実装しており，位置取得部の生成した LocationEvent を受け取る．受け取った LocationEvent を処理し，ModelDB クラスを用いて G-LoM を更新する．また ModelDB クラスは，G-LoM の重みを更新する際，CalculateWeight インタフェースを実装したクラスを利用する．今回の実装では全ての移動時間の履歴を算術平均する AverageWeight クラスを実装した．CalculateWeight インタフェースを図**??**に示す．CalculateWeight インタフェースを利用することにより，重みの計算

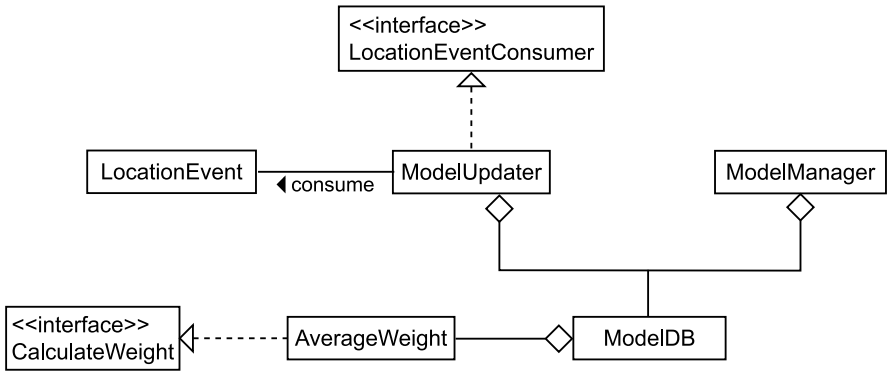

図 5.3: モデル管理部のクラス

方法を容易に変更可能になる．また，ModelManager クラスは要求解析部の RequestHandler クラスからの要求を処理し，必要な G-LoM の情報を ModelDB クラスから取得し，その結果を要求解析部へ返す．

```
public interface CalculateWeight{
    public double calWeight(double travel\_time, ModelDB modelDB):
}
```

図 5.4: `CalculateWeight` インタフェース

5.4 履歴管理部

履歴管理部は LocationEvent を受け取り，履歴を更新する．また，要求解析部からの要求に対して履歴情報を返す．

履歴管理部の主要なクラスは HistoryUpdater クラス，HistoryManager クラス及び HistoryDB クラスである．履歴管理部のクラスを図 5.5 に示す．

HistoryUpdater クラスは LocationEventConsumer インタフェースを実装しており，位置取得部の生成した LocationEvent を受け取る．受け取った LocationEvent を処理し，HistoryDB クラスを介して履歴を更新する．また，HistoryManager クラスは要求解析部からの要求を処理し，必要な履歴情報を HistoryManager クラスから取得し，その結果を要求解析部へ返す．

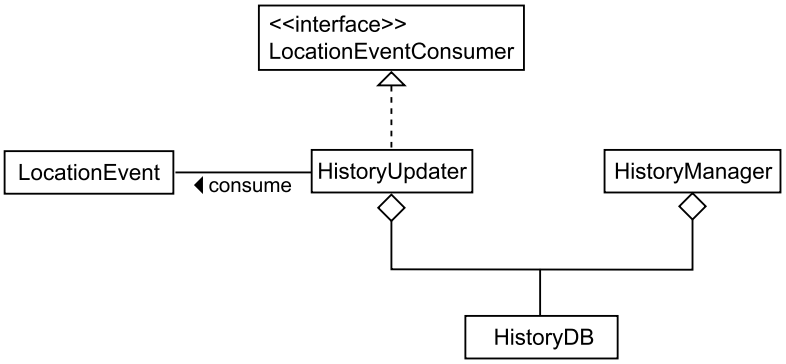

図 5.5: 履歴管理部のクラス

5.5 要求解析部

要求解析部はアプリケーションからの要求を処理し，位置情報を返すモジュールである．要求解析部の主要なクラスは RequestHandler クラス，EventFilter クラス，Registration クラス，SpaceNameDB クラス及び ObjectNameDB クラスである．要求解析部のクラスを図 5.6 と図 5.8 に示す．

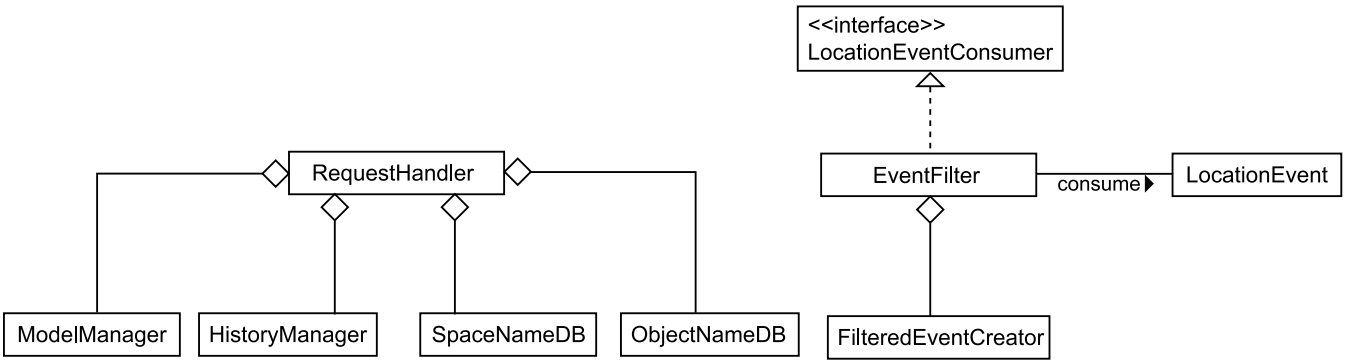

図 5.6: 要求解析部のクラス

5.5.1 RequestHandler クラス

RequestHandler クラスは ModelManager クラスと HistoryManager クラスをその構成要素としている．G-LoM や履歴の情報に関する要求を受け付け，ModelManager クラス，HistoryManager クラスに問い合わせ，その結果を返す．

RequestHandler クラスはアプリケーションが本システムの管理する位置情報を利用するために，開発者に提供する．以下に RequestHandler クラスのメソッドを述べる．

RequestHandler

Enumeration getAllSpaceName()
G-LoM に存在する全ての space_name を返す

Enumeration getAllEdges()
G-LoM に存在する全ての Edge オブジェクトを返す

int getDistance(String from_space_name, to_space_name)
２つの space_name を指定して空間間の相対距離を返す

String getCurrentSpaceName(String object_name)
object_id を指定して現在対象オブジェクトが存在する space_id を返す

Enumeration getObjectName(String space_name)
space_name を指定して現在存在するすべての object_id を返す

5.5.2 EventFilter クラス

EventFilter クラスはアプリケーションが対象オブジェクトの移動イベントを利用するためのクラスである．このクラスは LocationEventConsumer インタフェースを実装し，FilteredEventCreator クラスを構成要素として持つ．図 5.7 を実装したクラスをリスナとして EventFilter クラスに登録することにより，LocationEvent を受け取ることができる．

```
public interface LocationEventListener extends EventListener{
    public void locationEventCaught(LocationEvent evt):
}
```

図 5.7: `LocationEventListener` インタフェース

EventFilter

void addLocationEventListener(LocationEventListener location_listener)
LocationEventListener を EventFilter クラスにリスナとして登録する

void removeLocationEventListener(LocationEventListener location_listener)
登録済みの LocationEventListener を EventFilter クラスのリスナ登録から削除する

5.5.3 Registration クラス

Registration クラスは SpaceNameDB クラス及び ObjectNameDB クラスをその構成要素とする．これらのクラスを用いて，システム内で管理されている space_id と object_id のそれぞれを，ユーザが容易に理解できる識別子とマッピングする．

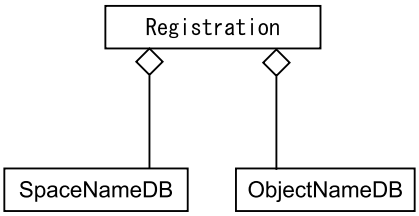

図 5.8: 要求解析部のクラス

Registration

Enumeration getAllSpaceID()
存在する全ての space_id を返す

Enumeration getAllSpaceName()
存在する全ての space_name を返す

String getSpaceName(String space_id)
space_id を指定して space_name を返す

String getSpaceID(String space_name)
space_name を指定して space_id を返す

void setSpaceName(String space_id, String space_name)
space_id を指定して space_name をマッピングさせる

Enumeration getAllObjectID()
存在する全ての object_id を返す

Enumeration getAllObjectName()
存在する全ての object_name を返す

String getObjectName(String object_id)
object_id を指定して object_name を返す

String getObjectID(String object_name)
object_name を指定して object_id を返す

void setObjectName(String object_id, String object_name)
object_id を指定して object_name をマッピングさせる

5.6 本章のまとめ

本章では，ロケーションモデルを動的に生成し，管理する NiSMo の実装について述べた．まず，実装の概要について述べ，次に各モジュールのクラス構成について述べた．今回の実装では，位置情報センサは RF-Code を利用した．また，G-LoM の重みは履歴の移動時間の算術平均で計算した．

次章では，G-LoM の定量的評価及び定性的評価，NiSMo の定性的評価を行う．

第6章　評価

本章では，ロケーションモデル作成を支援するミドルウェア NiSMo と NiSMo の用いるロケーションモデル G-LoM について評価する．まず，G-LoM の定量的評価を行う．次に，G-LoM と NiSMo の定量的評価を行う．

表 6.1: RF タグ　スペック

通信距離	約 20m (最大約 80m)
使用周波数	308.8MHz
発信感覚	0.2 秒
サイズ	40(W) × 25(D) × 10(H)mm

6.1 定量的評価

本節ではグラフ表現を用いたロケーションモデル G-LoM の定量的評価を行う．実験で得られた測定データを用いて作成した G-LoM の現実世界のモデルとしての正当性を評価する．

6.1.1 実験

2002 年 11 月に，慶応義塾大学 湘南藤沢キャンパスにて開催された ORF2002（Open Reserch Forum 2002）の会場で本システムの実験を行い，G-LoM を生成した．以下に実験の詳細を述べる．

実験方法

本実験は，慶応義塾大学 徳田研究室が研究成果の発表とデモを行っていた会場にて２日間行った．実験会場には研究成果の発表やデモを見るため多数のゲストが訪れた．来場したゲストに受付で RF-Code のタグ渡し，発表やデモを見ている間，常に携帯して頂いた．

実験環境

使用機材

位置情報センサは RF-Code 社の RF-ID を用いた．RF-ID はタグがそれぞれ固有の ID を一定間隔で発信し，リーダがその電波を捉えることにより，リーダの検出範囲内にあるタグの ID を取得する．リーダの検出範囲は約３メートルから約２０メートルまでを８段階で調節できる．また，リーダは RS-232C で PC に接続し，PC を LAN に接続した．タグのスペックを表 6.1，リーダのスペックを表 6.2 に示す．

実験会場

実験会場には建物の１フロアを使用した．会場に RF リーダを４個設置し，実験を行った．また，実験期間中，会場では７個のデモが行われていた．実験会場の見取り図を図 6.1 に示す．

表 6.2: RF リーダ　スペック

アンチコリジョン	14 個同時読み取り可能 (発信感覚が 0.2 秒の場合)
インタフェース	RS-232C
サイズ	120(W) × 120(D) × 40(H)mm

表 6.3: RF リーダ間の距離

	検出範囲間の距離	平均移動時間（G-LoM の重み）	検出範囲間の距離　÷　平均移動時間
Reception - Corridor	12 m	27.8 sec	0.431
Reception - SSLab	11.5 m	47.0 sec	0.244
SSLab - S214	3 m	19.1 sec	0.103
S214 - Corridor	5.8 m	42.3 sec	0.137

実験結果

約 1400 回のリーダ間の移動データが取れ，G-LoM が生成された．全ての移動データを使い，かかった移動時間の単純平均を用いて生成した G-LoM を図 6.2 に示す．

リーダ間の移動時間の分布を図 6.3 に示す．横軸が移動時間，縦軸が回数である．

6.1.2　生成した G-LoM の評価

本節では一日分のデータを使って生成した G-LoM のそれぞれの辺の重みを評価する．

表 6.3 にそれぞれのリーダの検出範囲間の距離，検出範囲間の平均移動時間，及び検出範囲間の距離を平均移動時間で割った数値を示す．また，検出範囲間の移動時間は反復切断法を用いて平均値 ± 2 標準偏差の範囲外をはずれ値とした．

さらに，表 6.4 に移動時間の最頻値（mode）と検出範囲間の距離を最頻値で割った数値を示す．

検出範囲間の距離を平均移動時間で割った値は Reception-Corridor 間の値が他と比べ，大きい値になっている．また，検出範囲間の距離を最頻値で割った値も Reception-Corridor 間が他の値と比べ，大きい値になっている．これは，Reception-Corridor 間は他の経路と比べデモが少なく，比較的早く通り過ぎた為と考えられる．

また，その他の平均移動時間を検出範囲間の距離で割った値は，比較的近い値になっている．このため，平均移動時間は人が検出範囲間を移動するコストを表現していると言える．

6.2　定性的評価

本節では G-LoM と NiSMo の定性的評価を行う．まず，G-LoM と既存のモデルとの比較を行う．次に，ロケーションモデルを管理する先行システムと NiSMo を比較する．

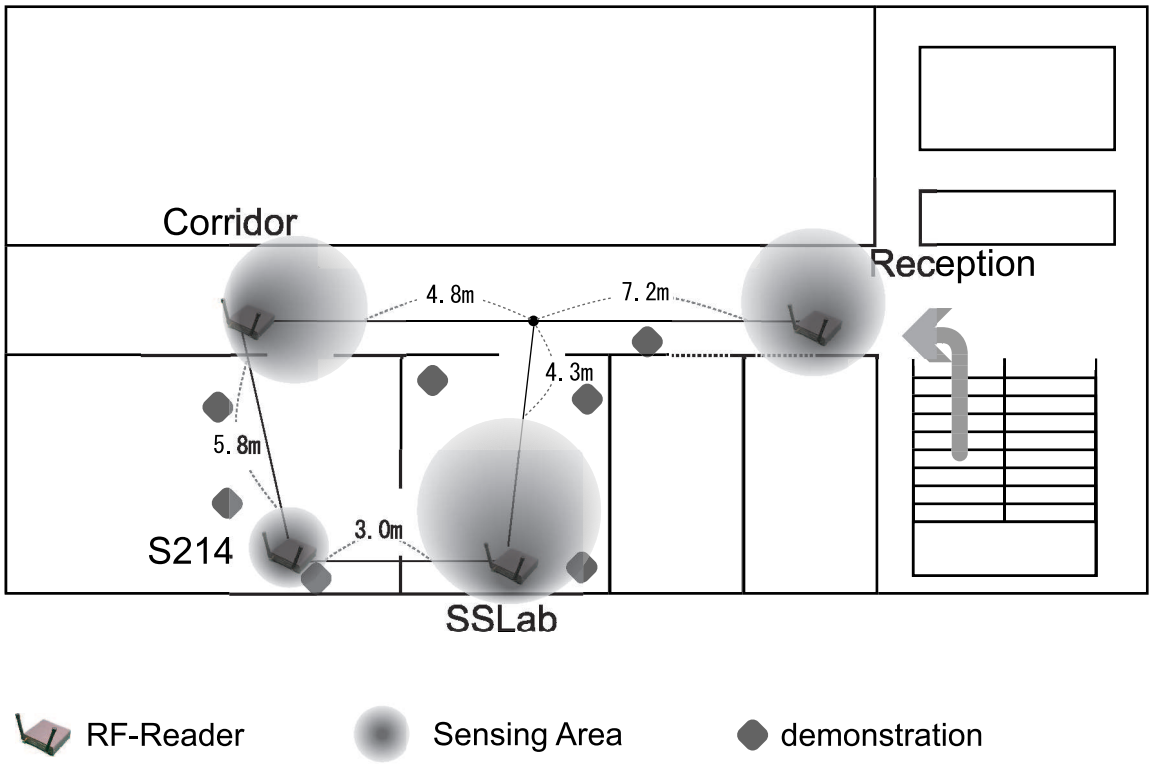

図 6.1: 実験会場の見取り図

表 6.4: RF リーダ間の距離

	検出範囲間の距離	最頻値（mode）	検出範囲間の距離　÷　最頻値（mode）
Reception - Corridor	12 m	10 sec	1.2
Reception - SSLab	11.5 m	35 sec	0.32
SSLab - S214	3 m	37 sec	0.08
S214 - Corridor	5.8 m	8 sec	0.72

6.2.1　モデルの比較

G-LoM，ツリー構造を用いたロケーションモデル，及び座標を用いたロケーションモデルを比較する．それぞれのモデルの性質を表 6.5 にまとめる．

座標を用いたモデル

座標を用いたモデルは Metric Model であり，座標表現のセンサ取得データを扱う．そのため，狭い検出範囲で比較的小さい粒度の位置情報を用いるアプリケーションに適している．また，モデル上の 2 点間の距離は計算で求められる．

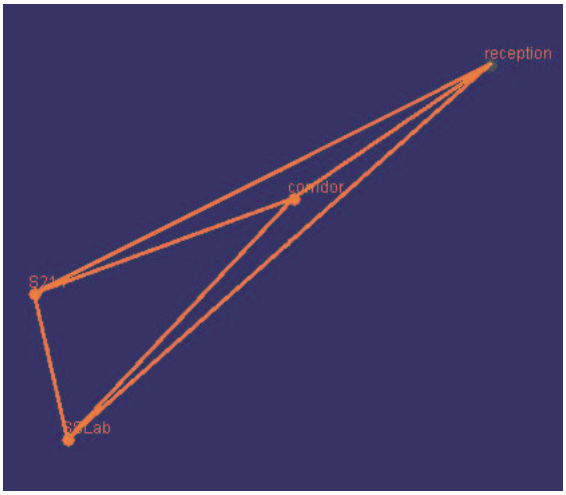

図 6.2: 生成された **G-LoM**

表 6.5: モデルの比較

		センサ取得データ	粒度	検出範囲	距離の表現
Metric	座標を用いたモデル	座標表現	小	狭い	○
Topological	ツリー構造を用いたモデル	包含表現	大	広い	△
	G-LoM	包含表現	大	広い	○

ツリー構造を用いたモデル

ツリー構造を用いたモデルは Topological Model であり，包含表現のセンサ取得データを扱う．そのため，広い検出範囲で比較的大きい粒度の位置情報を用いるアプリケーションに適している．また，ツリー構造を用いたロケーションモデルで距離を比較する場合，ある空間から空間へのパスがルート方向へ戻る階層の数で比較するが，同じ数の階層を戻る場合に比較ができない．

G-LoM

G-LoM は Topological Model であり，包含表現のセンサ取得データを扱う．また，G-LoM 上の任意の空間間の距離はそれぞれの空間を示した頂点間の最短距離経路を計算することで求められる．そのため，G-LoM は広い検出範囲で比較的大きい粒度の位置情報を用い，空間間の距離を利用するアプリケーションに適している．

6.2.2 先行システムとの比較

本システムと Topological Model を利用しロケーションモデルを管理する先行システム，Semantic Space とを実現する機能の面で比較する．

Semantic Space

ツリー構造のロケーションモデルを管理するシステムである．また，ファイルマネージャのような GUI を提供し，空間と空間内に存在するオブジェクトをそれぞれ，フォルダとファイルのように扱える．GUI を提供することにより，ユーザによるロケーションモデルの作成と閲覧を支援している．しかし，空間間の距離は定義できず，空間間の位置関係をユーザ，またはシステム管理者が入力する必要がある．

しかし，Semantic Space はツリー構造のロケーションモデルを利用しているため，空間間の距離を比較できない場合がある．

6.3 本章のまとめ

本章ではまず，実験の結果生成された G-LoM を定量的に評価した．次に，G-LoM，座標を用いたモデル及びツリー構造を用いたモデルを定性的に比較した．最後に G-LoM を生成，管理する NiSMo を定性的に評価した．次章では，今後の課題を述べ，最後に本論文をまとめる．

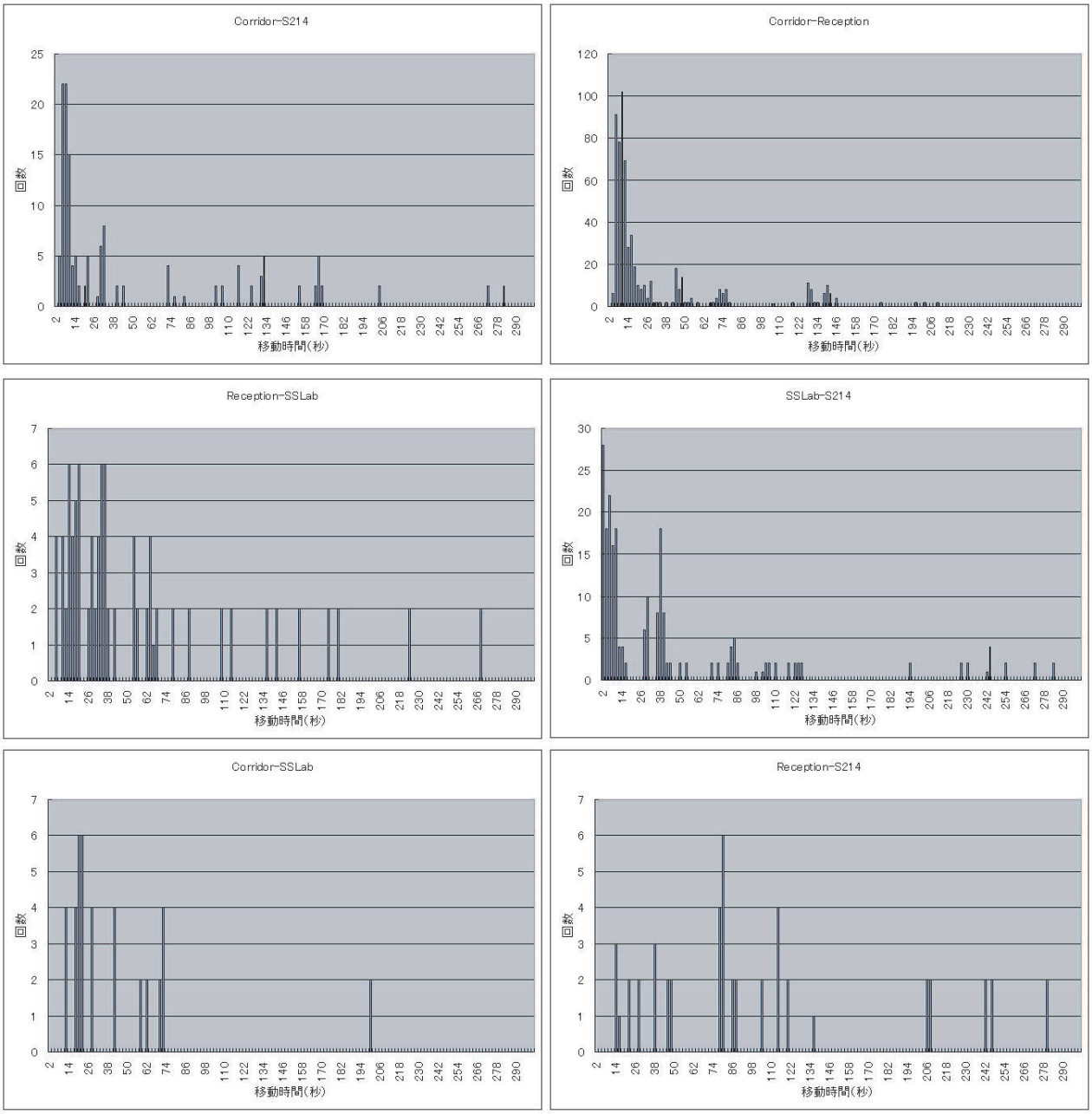
Corridor-S214
回数
移動時間(秒)
Corridor-Reception
回数
移動時間(秒)
Reception-SSLab
回数
移動時間(秒)
SSLab-S214
回数
移動時間(秒)
Corridor-SSLab
回数
移動時間(秒)
Reception-S214
回数
移動時間(秒)

図 6.3: 移動時間の分布

第7章　結論

本論文では，グラフ表現を用いたロケーションモデル G-LoM を動的に生成し，システム管理者によるロケーションモデル作成を支援する NiSMo を設計し実装した．本章では今後の課題を挙げ，最後に本論文をまとめる．

7.1 今後の課題

今後の課題として，異なる環境での再実験とプライバシの考慮を挙げる．

- 異なる環境での再実験

  本論文で行った実験の実験環境は，設置した位置情報センサの数が少なかった．そのため，本システムが生成した G-LoM のエッジ数が少なく，実際の距離と G-LoM 上の重みに関する定量的評価が十分とは言えない．今後，設置する位置情報センサの数を増やした再実験が課題となる．

- プライバシの考慮

  現状の NiSMo はプライバシ保護の機能を備えていない．位置情報はユーザの重要なプライバシを含むため，保護する必要がある．本システムにおけるプライバシの保護を今後の課題とする．

7.2 まとめ

本論文では，システム管理者によるロケーションモデル作成を支援する NiSMo を設計，実装し，評価した．

近年の情報科学の進歩により，様々な計算機器が人々の生活空間に存在するユビキタスコンピューティング環境が実現しつつある．このような環境で，人や物の位置情報を利用することにより，新しいサービスを実現したり，ユーザと計算機器のインタラクションを減らすなど，ユーザの利便性を向上できる．また，位置情報を用いる際，位置情報センサから取得されたセンサ取得データだけでなく，現実世界のモデルであるロケーションモデルを利用することにより，空間間の距離を比較する機能など，実現可能な機能が増える．

しかし従来はロケーションモデルを作成するために，システム管理者が空間同士の位置関係や距離などを入力する必要があった．今後位置情報センサが普及し，センサが認識する空間が増えることが予想される．また，位置情報を用いるシステムの家庭への普及も予想される．そのため，モデルを作成する際に必要なシステム管理者の入力を減らし，位置情報システムの導入，管理コストを下げる必要がある．

本研究では，ユーザやオブジェクトの移動履歴からロケーションモデルを動的に生成することにより，システム管理者のモデル作成を支援する **NiSMo** を構築した．また，NiSMo はグラフ表現を用いたロケーションモデル **G-LoM** を用いる．G-LoM を用いることにより，任意の空間間での距離の比較が可能になる．本システムを用いることにより，モデル作成のコストを下げ，位置情報システムの導入，管理コストを下げられる．

謝辞

本研究の機会を与えてくださり，ご指導を賜りました慶応義塾大学環境情報学部教授徳田英幸博士に深く感謝いたします.

慶応義塾大学徳田・村井・楠本・中村・南合同研究会の先輩方には折りにふれ貴重な指導と助言を頂きました．特に，徳田研究室の先生方や先輩方，ACE(Active Computing Environmets) 研究グループの方々に深く感謝いたします．また，桐原幸彦氏，青木崇行氏，中西健一氏，岩谷晶子氏には絶えざる励ましや丁寧なご指導をを賜りました．田丸修平氏と出内将夫氏の多大な協力に感謝します.

また本研究を進める過程で，WIDE プロジェクト iCARs Working Group[6] の方々が行った実験の測定データを使用させて頂きました．同グループの若山史郎氏に様々なアドバイスやご指導を賜りました．WIDE プロジェクト iCARs Working Group の方々に感謝致します.

最後に，本研究を通じて様々経験や刺激を受ける機会を頂きましたことに，深く謝意を表します.

平成 15 年 2 月 22 日

村上 朝一

参考文献

[1] Pentland. A. Machine understanding of human action. In *Proceedings of 7th International Forum on Frontier of Telecommunication Technology*, Tokyo,Japan, November 1995.

[2] B. Brumitt and S. Shafer. Topological world modeling using semantic spaces. In *Ubicomp*, Sep 2001.

[3] Dallas Semiconductor Corp. Introducing TINI: Tiny InterNet Interface, 2000. http://www.ibutton.com/TINI/index.html.

[4] S. Domnitcheva. Location modeling: State of the art and challenges. In *Proceedings of the Workshop on Location Modeling for Ubiquitous Computing*, Sep 2001.

[5] I. A. Getting. The global positioning system. *IEEE Spectrum*, Vol. 30, pp. 36–47, December 1993.

[6] iCARs Working Group. Wide project. http://www.wide.ad.jp.

[7] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *MobiCom*, Aug 2000.

[8] Hodges. S and Louie. G. Towards the interactive office. In *Proceedings of SIGCHI'94*, Boston, April 1994.

[9] InData Systems. Rf code spider rfid system. http://www.indatasys.comhtml/products.

[10] R. Want and A. Hopper. Active badge and personal interactive computing objects. In *IEEE Transactions on Consumer Electronics*, February 1992.

[11] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. No. 92.1, ORL, 24a Trumpington Street, Cambridge CB2 1QA, 1992.

[12] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. In *IEEE Personnel Communications, 4(5):42–47, October 1997.*

[13] M Weiser. The computer for the 21st century. In *Scientific American 256(3),94–104, September 1991*.

[14] 横河電機株式会社. Compact Field Server DUONUS, 2000.
http://www.yokogawa.co.jp/DUONUS/Product/Product.htm.

[15] 楠本晶彦，岩井将行，中澤仁，大越匡，徳田英幸. 物理的位置情報に基づくサービスの自動検出を実現するミドルウェアの構築. マルチメディア，分散，協調，とモバイル (DICOMO) シンポジウム, 2000.