修士論文　　2002年度 (平成14年度)

Personal Server Modelに基づく センサ情報管理機構の実現

慶應義塾大学　政策・メディア研究科
若山 史郎

修士論文要旨　　2002年度(平成14年度)

Personal Server Modelに基づくセンサ情報管理機構の実現

概要

本研究では，センサ情報をはじめとする実空間における情報を集中的に管理するモデルを提案する．また，このモデルに基づいたセンサ情報管理機構を設計および実装した．

コンピュータをさまざまなものに埋め込むことで遍在化し，それを自由に使用する環境が，ユビキタスコンピューティング環境だと言われている．本研究ではセンサデバイスを使用することにより，計算機資源を埋め込むことなく，さまざまなものおよび実空間における情報をコンピュータおよびネットワークから扱う環境を，実空間ネットワーク環境と呼ぶ．

本研究では，実空間ネットワーク環境を実現するために必要な，グローバル型センサインフラストラクチャ環境を想定環境とする．この環境は，複数種類のセンサインフラストラクチャが存在し，その中をユーザは移動しながら近隣に存在するセンサを使用するという環境である．

しかし，想定環境において，ユーザの移動による，使用されるセンサインフラストラクチャの変化問題およびプライバシ保護問題が生じる．本研究はこれら二つの問題を解決するためのPersonal Server Modelを提案する．

これは，センサ情報を含む実空間におけるさまざまな情報はすべて個人管理のサーバ，Personal Serverが管理するというモデルである．センサが取得した情報は，すべてPersonal Serverに送信され，また，そのユーザに関する情報を得ようとした第三者はPersonal Serverに対して要求する．このモデルを用いることで，Personal Serverでの集中管理に加えて，Personal Serverが持つアクセス管理機構によって，プライバシ保護を実現した．

さらに，ユーザが持つGUIDからPersonal ServerのIPアドレスを導き出す，SDP(Server Discovery Protocol)を提案することでユーザの移動に関わる問題点も解決した．また，Personal Serverで管理されているさまざまな情報を利用するためのPMLを提案した．このPMLはXMLおよびRDFを基本にして設計され，拡張性およびデータ交換性に富む．

本論文ではPersonal Server Modelを構成する各機能の実装を行った．また，Personal Serverに管理されている実空間における情報を容易に利用するためのC言語ライブラリも実装し，評価した．

本論文では，評価として実空間における情報を扱う他の関連研究と，想定環境の実現および問題点の解決の可否を比較した．また，本モデルに関する数百人規模での実証実験を行い，その成果および実験結果から得られた実空間における情報の有用性を述べた．

キーワード: 1: Real-Space Networking 2: Personal Server 3: Heterogeneous Sensors

慶應義塾大学　政策・メディア研究科
若山 史郎

Abstract of Master Thesis Academic Year 2002

Design and Implementation of Sensor Information Management System based on Personal Server Model

Abstract

This research proposes a design and implementation of a model which manages sensor information in Real-Space, called *Personal Server Model*.

Ubiquitous computing environment is embedding computers into everything. *Real-Space Network Environment* proposed in this research aims at everything as well as object that have embedded computer. Real-Space Network Environment uses sensor devices in order to provide functional equivalence of Ubiquitous Computing environment to the users.

This research's supposed environment assumes that *Global-Sensor Infrastructure Environment* to realize Real-Space Network Environment. In this environment, there are heterogeneous sensors within the infrastructure and moving users within the environment uses the sensors.

However, in the supposed environment, there are problems such as 1) modification of the sensor infrastructure caused by user's movement and 2) preservation of user's privacy. This research proposes a personal server model that will solve these two problems.

In this model, all information including the sensor information is managed in the server, that is managed by the user, called *Personal Server*. All information from the sensor is sent to the personal server. Requests from other users would be sent to the personal server as well. Privacy is protected by the central control function and access control function which is provided by the Personal Server.

Additionally, SDP (Server Discovery Protocol) is suggested in order to derive the personal server's IP address from the user's GUID. SDP can solve the problem which is caused by user movement. Furthermore, PML is introduced to exchange information between Personal Server, other users, and other applications. PML is designed based on XML and RDF, so it has extensibility and data convertibility.

This research implemented each components required for the Personal Server Model such as a library made from C-language, which makes the information easily usable in real space.

This research and other related research that uses information in Real-Space are evaluated. Personal Server Model is proved that it is able to realize the supposed environment by carrying out an experiment with some hundreds of people, leading to a conclusion that information in Real-Space is effective.

Keywords: 1: Real-Space Networking 2: Personal Server 3: Heterogeneous Sensors

Shirou Wakayama
Graduate School of Media and Governance
Keio University

目 次

第 1 章　序論 **1**
1.1 実空間ネットワーク環境 1
1.2 想定環境 2
1.3 目的 3
1.4 用語の定義 4
1.5 本論文の構成 5

第 2 章　問題点の指摘および関連研究 **6**
2.1 問題点 6
2.2 関連研究 8

第 3 章　Personal Server Model の提案 **11**
3.1 モデルに関する議論 11
3.2 Personal Server Model 13
3.3 Personal Server 14
3.4 プライバシ保護問題の解決 15
3.5 ユーザ移動問題の解決 16
3.6 実空間情報の表現手法 17

第 4 章　Personal Server Model の設計 **19**
4.1 Persnal Server 20
4.1.1 ネットワーク接続性 20
4.1.2 データベース 20
4.1.3 アクセスコントロールモジュール 20
4.1.4 拡張モジュール 22
4.2 PML(Person Markup Language) 23
4.2.1 機能要件 23
4.2.2 PML の設計 24
4.2.3 PML 24
4.3 SDP(Server Discovery Protocol) 25
4.4 本章のまとめ 28

第 5 章　Personal Server Model の実装 **29**
5.1 Personal Server の実装 29
5.2 PML の実装 31
5.3 SDP(Server Discovery Protcol) の実装 31
5.3.1 SDP ノードが保持する情報 32
5.3.2 Query の種類 33
5.3.3 Query 送信 33
5.3.4 Neighbor の追加 34
5.3.5 Query 受信時の処理 34
5.4 数百人規模での実証実験 36
5.4.1 実証実験での全体構成 37
5.4.2 センサデバイス 38
5.4.3 PML 41
5.4.4 提供したサービス 41
5.4.5 ライブラリ 42

第 6 章　Personal Server Model の評価 **44**
6.1 定性的評価 44
6.1.1 関連研究との比較 44
6.1.2 Personal Server Model におけるプライバシ保護 45
6.2 数百人規模での実証実験の結果 45
6.2.1 人の動き 46
6.2.2 部屋ごとの人の動き 46
6.2.3 全体的な人の動き 46

第 7 章　結論 **49**
7.1 まとめ 49
7.2 今後の課題 50
7.2.1 Personal Server の移動 50
7.2.2 アプリケーション 50
7.2.3 対象の拡張 51
7.2.4 アクセス管理機構の拡張 52

付 録 A 提供した C 言語ライブラリ **56**

図 目 次

1.1 グローバル型センサインフラストラクチャ 4

3.1 Personal Server Model 13

4.1 Personal Server Model に関する登場人物の整理 19
4.2 Personal Server におけるアクセス管理 21
4.3 Personal Server の拡張モジュール構造 23
4.4 Server Disocovery Protocol 概要 27

5.1 Personal Server のプロトタイプ実装概要 30
5.2 PML による表現の例 31
5.3 RDF スキーマによる PML Core 部の定義例 32
5.4 Name-Dropper アルゴリズム 34
5.5 SDP における Query 受信時動作に関する疑似コード 36
5.6 ホテル全体図 37
5.7 全体構成概要 38
5.8 RF-Code タグ (たばこ箱は比較対象用) 39
5.9 RF-Spider 基地局 (たばこ箱は比較対象用) 40
5.10 Web インタフェース：位置情報の検索 42
5.11 Web インタフェース：位置からの情報検索 43
5.12 Web インタフェース：Personal Server IP アドレスの取得 43

6.1 実空間情報の流れおよび Personal Server Model での保護 45
6.2 実際の移動履歴と検知された移動履歴 46
6.3 BOF2 における時間軸と人数とのグラフ 47
6.4 LOBBY3 における時間軸と人数とのグラフ 47

表 目 次

1.1 コンピューティング環境の分類 2
1.2 用語の定義 5

2.1 モデル比較 10

4.1 検索データベース構造におけるモデル比較 26

5.1 Personal Server のプロトタイプ実装 29
5.2 RF リーダセットの仕様 39

6.1 定性的評価 44
6.2 定性的評価 (続き) 44
6.3 ユーザの移動グラフ 48
6.4 ユーザの移動グラフの注釈 48

第1章　序論

本章では，本研究全体の背景となる実空間ネットワーク環境について述べる．また，本研究が想定している環境について述べる．これらの環境を実現することが本研究における目的となる．また，用語の定義を行う．

1.1　実空間ネットワーク環境

技術の向上により，コンピュータの小型化および軽量化が進んでいる．それに加えて，IEEE802.11b[1] に代表される無線ネットワーク技術により，モバイルコンピューティング環境が実現しようとしている．

モバイルコンピューティング環境とは，コンピュータを保持したユーザが，どこに移動しても，作業を継続することが可能な環境である．モバイルコンピューティング環境よりもさらに小型化および無線技術が向上することにより，実現されるのがユビキタスコンピューティング環境である．

ユビキタスコンピューティング環境は，さまざまなものにコンピュータが埋め込まれ，遍在している環境である．ユーザ自身は移動先にあるコンピュータを利用して作業を行うことが可能となる．また，ものにコンピュータが埋め込まれることにより，そのもの自体をネットワークを介して扱うことができる．例えば，どこにいても手元の，あるいは，手近なモニタで自宅の玄関を監視しているカメラの映像を見ることができ，さらに，そのカメラの監視角度も変えるなどが可能になるだろう．

ユビキタスコンピューティング環境は，モバイルコンピューティング環境よりも広い概念を扱っている．なぜならば，モバイルコンピューティング環境は “計算機” が移動することを考えている．一方，ユビキタスコンピューティング環境は計算機だけではなく，コンピュータが埋め込まれているものの移動や，ユーザ自身の移動も想定しているからである．

だが，現実にはユビキタスコンピューティング環境が想定する，すべてのものにコンピュータが埋め込まれているという環境が実現することはコストの面などから難しい．そのため，我々は実空間ネットワーク環境を提案している．

実空間ネットワーク環境は，センサを用いることで，ものに対してコンピュータを埋め込むユビキタスコンピューティング環境と仮想的に等価な環境を実現する．コンピュータを埋め込むというユビキタスコンピューティング環境のアプローチは，確実ではあるが，実際にはコンピュータを埋め込むことが不可能なものも存在する．しかし，センサを利用し，外側からそのものに関する情報を収集することにより，コンピュータを埋め込むことが不可能なものでさえもコンピュータ上から扱うこと可能となる．つまり，センサを用い

ることにより，現実世界に存在するすべてのものの情報を取得することが可能となる．そして，現実に存在するものの情報は，ネットワークを通じて世界中から利用可能となる．

実空間ネットワーク環境の “実空間” という言葉は，現実の空間という意味であり，その対象となる用語としては，“仮想空間” がある．仮想空間はコンピュータ上にのみ存在する空間であり，現実のユーザはキーボードやディスプレイなどのインタフェースを通じてのみ仮想空間を理解することができる．一方，実空間は仮想空間以外の，現実に我々が存在している空間であり，仮想空間上に位置するコンピュータはセンサデバイスを使用することでのみ実空間を理解することができる．

実空間ネットワーク環境は，センサデバイスを用いることにより，仮想空間に加えて実空間をもコンピュータネットワーク上で扱うことを目標とする．

表 1.1 に，各コンピューティング環境の関係を示す．モバイルコンピューティング環境は，計算機の移動のみを対象としている．ユビキタスコンピューティング環境は，移動に加えて埋め込みによるコンピュータの遍在化を行う．実空間ネットワーク環境は，ユビキタスコンピューティング環境に加えてセンサデバイスの使用により，コンピュータを埋め込めないものでさえも扱うことができる．

表 1.1: コンピューティング環境の分類

センサデバイスの使用	実空間ネットワーク環境
コンピュータの埋め込み	ユビキタスコンピューティング環境
移動	モバイルコンピューティング環境

本研究は，実空間ネットワーク環境を実現するために必要な，センサデバイスから取得した情報を管理するためのモデルを提案し，その機構を実現する．また，本論文では，実空間ネットワークが取り扱う，実空間に存在する全てのものの中でも特に人間である “ユーザ” に焦点を当てる．その理由は，人間は一意な存在であり，分離，結合，重複などが起こらず，また，実空間において最も必要とされる情報を保持しているからである．位置情報をはじめとする実空間のさまざまな情報は，対象を人間であるユーザに限定することで，その利用方法をより明確にできる．

1.2 想定環境

技術の進化により，センサデバイスは大幅に高機能化，低消費電力化，小寸法化，低価格化が進んでいる．しかし，単一種類のセンサで全ての情報を取り扱えるわけではなく，特定用途に限定したセンサデバイスがインフラストラクチャとして，多種類および多数存在するようになるものと考えられる [2]．多種類とは，位置情報や温度情報など，検知可能な情報の種類が多数存在することであり，多数とは，位置情報を検知するセンサデバイスが，大量に存在することである．多数の場合，地理的に分散していることも想定される．

例えば，位置情報を取得するセンサデバイスを想定した場合ですら，GPS で全てを網羅することはできず，屋内では異なるデバイスを使用する必要がある．また，GPS のように緯度経度情報ではなく，“慶応大学湘南藤沢キャンパス大学院棟 2F 会議室 A” など人間が

読める形式でのラベル情報が有用な場合も考えられる [3].

これらの背景をふまえ，本研究では，グローバル型センサインフラストラクチャを前提環境とする．センサデバイス単体あるいは，同一種類のセンサデバイスが集合して形成されたセンサネットワークなどをイントラ型センサインフラストラクチャとする．グローバル型センサインフラストラクチャとは，イントラ型センサインフラストラクチャ同士がセンサ情報を送信しあうことで結合し，形成される環境である．その構成例を図 1.1 に示す.

この図では，網掛け部が各イントラ型センサインフラストラクチャを表している．図中の左上には複数のセンサデバイスが結合し，センサネットワークが形成されているセンサインフラストラクチャA，右上には複数のセンサデバイスからの情報を一つのデータベースに集約しているセンサインフラストラクチャB および左下に位置する C，単一のセンサデバイスで範囲全てをカバーしている右下のセンサインフラストラクチャD などが存在している．また，お互いのセンサインフラストラクチャはネットワークで結合され，センサ情報を相互に交換することができる．

ユーザはこのような環境の中を自由に移動し，ユーザの近辺に存在するセンサデバイスを使用して実空間における情報を取得する．ユーザの移動に従い，ユーザが使用するセンサデバイスも変化する．また，複数人のユーザが同時に一つのセンサデバイスを使用することも想定される.

また，それぞれのセンサインフラストラクチャには管理者が存在する．グローバル型センサインフラストラクチャにおいてはこれらの管理者が多数存在し，センサを管理している．この管理者は，管理しているセンサインフラストラクチャが取得した全てのセンサ情報を扱うことが可能である.

このようなセンサデバイスインフラストラクチャの例としては，ユーザの位置情報を取得することができる GPS，赤外線や超音波を利用した位置情報取得システム，ユーザ周辺の温度を取得できる温度センサ，体温センサなどが挙げられる．これらは，ユーザ自身が管理者となりうる場合もあるが，多くの場合は，近隣のセンサインフラストラクチャを使用すると予想される.

以下に本研究が想定する環境をまとめる.

- グローバル型センサインフラストラクチャ
  さまざまなセンサインフラストラクチャが混在し，それが相互に接続され，情報が交換可能されている．また，それぞれのセンサインフラストラクチャごとに管理者が存在している.

- ユーザの移動
  ユーザは移動しながら，近隣に存在するセンサデバイスを使用する.

1.3 目的

本研究では，実空間ネットワーク環境において，ユーザがどこに移動しても，位置情報をはじめとする，そのユーザに関する実空間における情報をネットワーク上から取得可能

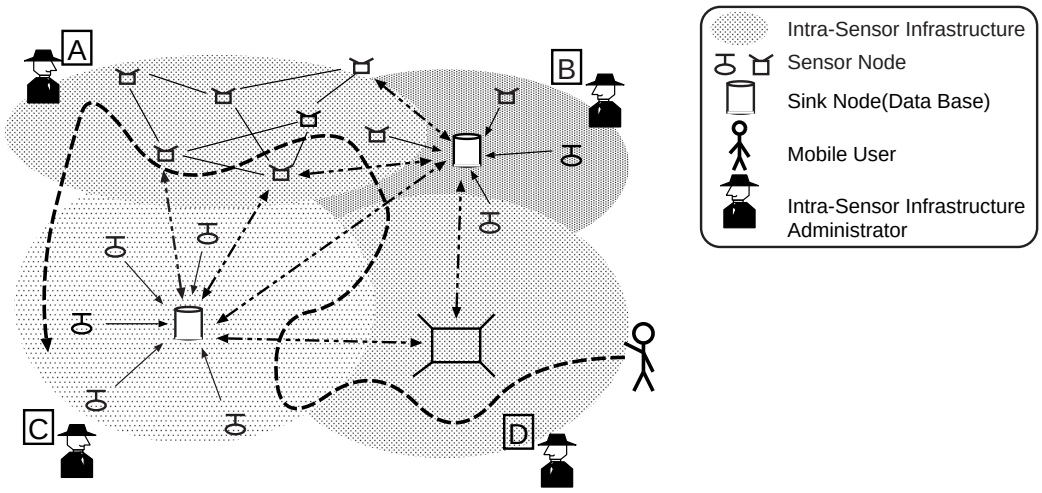

図 1.1: グローバル型センサインフラストラクチャ

にすることを目的とする．加えて，ユーザの移動先に存在するセンサが取得した情報は個人情報であるため，本研究は第三者からの個人情報保護の実現も目的とする．

本研究では実空間ネットワーク環境を実現するための情報管理モデルである，Personal Server Model を提案する．Personal Server Model は，従来はイントラ型センサインフラストラクチャ内でのみ管理されていたセンサ情報を，個人ごとに集中管理するモデルである．このモデルを適用することにより，従来では実現できなかった実空間ネットワーク環境におけるユーザの移動への対応およびプライバシ保護が実現できる．

1.4 用語の定義

本節では本論文で用いられる用語の定義を行う．

表 1.2: 用語の定義

用語	定義
実空間	現実の空間．対義語は仮想空間
もの (Object)	実空間上に存在する物体全般を指す．インターネット接続性の有無は問わない
実空間ネットワーク環境	実空間におけるさまざまな情報をネットワーク上から扱う環境
実空間情報	実空間における情報．具体的には，位置，時間，光度，騒音，温度，渋滞状況，ユーザの履歴，ユーザのプロフィール，ユーザのスケジュールなど
センサ情報	実空間情報の中で，さらにセンサデバイスから取得された情報
センサデバイス	センサを用いて実空間における情報を取得可能なデバイス．本研究では，センサデバイスは多少の計算能力，GUID 取得能力，ネットワーク接続性を持つことを前提とする．また，同一センサデバイスを複数のユーザが使用することも想定される
センサインフラストラクチャ	センサデバイス単体あるいは，同一種類のセンサデバイスが複数個集合しネットワークを形成することによって作られる，ある一種類のセンサ情報をユーザに対して取得することが可能なインフラストラクチャ
管理者	センサインフラストラクチャを維持，管理している個人あるいは組織
ユーザ	移動する個人．近隣のセンサインフラストラクチャを使用するで，自分自身の情報をネットワークから扱えるようにする．ネットワークへの接続性はなくてもよい
GUID(Globally Unique ID)	一意であることが保証されている ID

1.5 本論文の構成

本論文では，第 2 章において実空間における情報を利用する関連研究について述べ，それらの関連研究が前提環境を実現可能かどうかを述べる．また，前提環境を実現するための，解決すべき問題点を指摘する．

第 3 章においては，要求仕様を提示し，要求に適合するセンサ情報管理モデルである Personal Server Model を提案する．

第 4 章では，Personal Server Model に必要な機能について述べた後，各機能に関する要求仕様および設計について述べ，第 5 章でプロトタイプ実装について触れる．

その後，第 6 章でプロトタイプ実装の測定および定量的評価，定性的評価を行い，本研究の優位性を述べる．第 7 章において今後の課題について述べた後に，本論文をまとめる．

第2章　問題点の指摘および関連研究

本章では，まず，前章で述べた想定環境を実現するために解決すべき問題点を述べる．また，実空間情報を扱う他の関連研究をあげ，これらの関連研究が問題点を解決し，想定環境を実現することが可能かどうかを議論する．

2.1　問題点

本研究は多種多様なイントラ型センサインフラストラクチャが多数存在してグローバル型センサインフラストラクチャを形成し，その中をユーザが移動する環境を想定環境とすることを 1.2 節で述べた．この想定環境を実現するためには，以下の問題点が存在する．

1. 複数種類のセンサ機器が存在する場合に，そのユーザに関する実空間上の情報の要求先が不明
2. センサ情報という個人情報を第三者が管理することで，個人情報が漏洩する

ユーザの移動問題: 一点目の問題点は，ユーザが使用するセンサがユーザの移動に応じて変化することに起因する．イントラ型センサインフラストラクチャ内で取得したセンサからの情報は，その内部でのみ管理されるという従来想定されている環境の下では，ユーザの実空間情報は明示的に示さなくても，その管理場所は容易に判断できる．しかし，移動に応じてユーザが使用するセンサインフラストラクチャを変更するグローバル型センサインフラストラクチャにおいて，従来と同じ方式で管理すると，センサからの情報が各イントラ型センサインフラストラクチャに分散することになってしまう．つまり，ユーザの実空間情報を他のユーザやアプリケーションが得ようと思った場合，ユーザが現在使用しているセンサインフラストラクチャが不明であると，どのセンサインフラストラクチャにセンサ情報を要求すればよいか分からないという問題が生じる．この問題を “ユーザの移動問題” とする．

プライバシ保護問題: 二点目の問題点は，センサ情報が個人情報であることに起因する．想定環境において，ユーザは近隣のセンサデバイスを用いて，ユーザ自身の実空間情報を取得する．つまり，センサが取得する情報は，極めて個人的なものであり，その情報は第三者に対して隠蔽されなければならない．しかし，想定環境では，ユーザが使用しているセンサは，ユーザ自身ではなく信頼不能な第三者が管理している．そのため，個人情報であるセンサ情報がその管理者に漏洩する可能性が非常に高いという問題点が生じる．

また，グローバル型センサインフラストラクチャでは，複数のユーザが同一のセンサインフラストラクチャを使用することが考えられる．しかし，センサデバイスを使用している他のユーザは必ずしも信頼できない．そのため，信頼できない他のユーザが，共有しているセンサデバイスに対する攻撃や，センサデバイスが取得したセンサ情報を盗聴するなどの行為を行うことも想定される．これらの問題を “プライバシ保護問題” とする．

解決への要求事項

本項では，前述した問題点を解決するための要求事項を整理する．

ユーザの移動問題

ユーザの移動問題を解決するためには，他のアプリケーションが，そのユーザのセンサ情報が管理されているネットワーク上の位置を理解することが必要である．

イントラ型センサインフラストラクチャ内部でのみセンサ情報が管理されている従来手法では，ユーザが移動する都度，センサ情報が管理されているネットワーク上の位置が変更される．すなわち，ユーザの位置および，現在使用しているセンサインフラストラクチャを外部のアプリケーションが理解し，それにより，要求先を変更する必要がある．

この問題を解決する別の手法として，ユーザに関するセンサ情報は，ユーザが使用しているセンサインフラストラクチャに関わらず一定の場所に管理するという手法がある．この場合，他のアプリケーションはユーザの移動に応じてセンサ情報の要求先を変更する必要はない．しかし，逆にセンサインフラストラクチャが，取得したセンサ情報をそのユーザがセンサ情報を管理している場所に送信する必要がある．

プライバシ保護問題

プライバシ保護問題は，センサ情報を含むさまざまな実空間における情報を信頼できる相手以外から保護することにより，解決される．相手とは，自分の情報を要求する他のアプリケーションを動かしているユーザである．また，この場合の信頼とは，相手が自分を特定していると同時に自分が相手を特定していることによって形成される．そのための機能要件として相手を識別し，特定すること必要である．

ただし，信頼できない相手にとってその情報が，自分であることが分からない単なる情報のみである場合は，保護されているとする．例えば，意味のない数字の羅列であるユーザの ID とセンサ情報だけしか，相手にとって分からない場合などである．なぜなら，相手は結び付けるための情報を保持せず，ユーザの ID とその保有者とを結び付けることが相手にはできないからである．

また，想定環境内において，信頼できない相手として想定されるのは，自分と相手以外，例えば，センサインフラストラクチャの管理者あるいは同じセンサインフラストラクチャ

を使用している他のユーザなどが挙げられる．これらの相手からセンサ情報を保護することが必要とある．

2.2 関連研究

本節では，実空間情報を扱う関連研究について述べる．また，関連研究による想定環境の実現可能性について考察する．

Active Bat

Active Bat システム [4] では，ユーザは Bat と呼ばれる小型超音波受信機を保持する．建物内には超音波基地局があり，基地局からの超音波を Bat が受信することにより，システムはユーザの位置を認識できる．

Active Bat では現実世界にあるテレビやワークステーションは Ouija[5] と呼ばれるオブジェクト指向データベース言語によって記述され，オブジェクトとしてデータベースに格納される．

また，すべてのコンピュータには資源モニタがインストールされている．この資源モニタは資源の使用可能状態や使用状況などを監視している．資源モニタから得られた情報はデータベースに格納され，Ouija によって使用される．

しかし，資源モニタなどが，超音波を利用し位置情報を取得する Active Bat システムに依存しているため，Active Bat システムが存在しない場所ではシステム全体が動作しない．つまり，Active Bat システムは，イントラ型センサインフラストラクチャとして設計されている．そのため，問題点である，ユーザの移動に関する問題点を解決できない．

TEA

TEA システム [6] は，実空間の情報をレイヤリングモデルで表現している．最もセンサに近い情報を “Cue” レイヤで収集する．一つの Cue は単一のセンサを表現するが，複数の Cue を組みあわせることで，上位の “Context” レイヤでより複雑な実空間の情報を表現することができる．

TEA システムは，レイヤリングモデルにより，多種多様なセンサデバイスを表現し，扱うことが可能である．しかし，これらのセンサデバイスから取得した情報をどのように管理するかのついては述べられていない．つまり，ユーザが移動した場合，センサ情報を他のユーザがどのように取得すればいいのかについて不明である．従って，問題点である，ユーザの移動に関する問題点を解決できない．

GUIDE

GUIDE[7] システムは，観光客に対して観光案内を行うシステムである．具体的にはGUIDE システムはドイツのランカスター城周辺を案内し，歴史などをユーザに対して説明する．観光客であるユーザは GUIDE ユニットと呼ばれる端末を保持する．GUIDE ユニットには液晶画面があり，ウェブブラウザの機能を持つ．GUIDE システムサーバは，ユーザ状況に応じてさまざまな情報を HTML を用いて GUIDE ユニットに表示することにより，ユーザを案内する．

GUIDE システムは，IEEE 802.11[1] による多くの無線セルと GUIDE サーバで構成されている．GUIDE ユニットは無線基地局から位置情報が付加されたメッセージを受け取ることにより，ユーザの位置を知ることができる．GUIDE ユニットが認識したユーザの位置は，GUIDE サーバに送られ，GUIDE サーバも同様にユーザの状況を認識できる．これに基づき，GUIDE サーバは GUIDE ユニットに対して，状況に応じた情報を HTML 形式で送信する．ただし，観光案内という GUIDE システムの目的のために，ユーザ状況の中でも主に位置情報に重点がおかれる．

GUIDE システムは IEEE 802.11 以外のセンサデバイスでも，セル情報を取得できるのであれば動作する．しかし，その他のセンサデバイスをも考慮した，グローバル型センサインフラストラクチャ上で動作するためには，GUIDE サーバが複数存在しなくてはならなす，異なる GUIDE サーバ間の連携は考慮されていない．また，個人情報の保護に関しては想定されていない．

PARCTAB

PARCTAB[8] では PARCTAB と呼ばれる小型のデバイスをユーザは常に持ち歩く．PARCTAB には，小型のタッチパネル型ディスプレイが付属し，ユーザはこれにより情報を取得，あるいは送信することができる．また，PARCTAB には赤外線送信部があり，各部屋には赤外線受信部がある．この受信機が赤外線を受信することにより，TAB の位置情報やその部屋に他に誰がいるのかなどが分かる．

これらの位置情報は，各 TAB ごとに存在する，TAB-Agent と呼ばれるプロセスが管理している．ユーザは，この TAB-Agent にアクセスすることで，位置情報を利用したアプリケーションを利用することができる．

しかし，PARCTAB では，情報はあるユーザのみが使用するものとしており，他のユーザはその情報を利用することができない．従ってユーザの移動に関する問題は解決できない．また，TAB-Agent にはアクセス管理機能がなく，TAB-Agent に保存されている位置情報を他のユーザに対して保護することができない．

GLI

GLI[9] システムはプライバシ保護を実現しつつ，インターネットに接続している移動体の地理位置情報を管理するシステムである．移動体は，自分の位置情報をサーバに登録し，

検索者は移動体の識別子を鍵とした位置検索および地理的範囲を鍵とした移動体検索が可能である．検索の際には，無意味な識別子である HID(Hashed ID) を利用することにより，プライバシ保護を実現している．また，GLI システムは階層構造を元に，分散管理による規模性を保持した運用が可能である．

しかし，GLI システムは移動体はインターネットに接続していることを前提としている．従って，実空間ネットワーク環境が目指す，ネットワーク接続性を持たないものも扱うという目的は達成できない．

関連研究による問題解決の可否

関連研究がこれらの問題点を解決可能かどうかをまとめたものを表 2.1 に記す．

表 2.1: モデル比較

関連研究 / 項目	Active Bat	TEA	GUIDE	PARCTAB	GLI
センサの種類	単一種類	複数	複数	単一種類	単一種類
センサインフラストラクチャの形態	イントラ型	イントラ型	イントラ型	グローバル型	イントラ型
ユーザの移動問題	想定外	想定外	想定外	想定外	解決可能
プライバシ保護	なし	なし	なし	なし	あり

結果として，前述した関連研究では，想定環境においてユーザの移動への対応およびプライバシ保護は実現できない．

次章では，これらの問題点を解決する Personal Server Model について述べる．

第3章　Personal Server Modelの提案

前章では，実空間におけるさまざまな情報を扱う関連研究を述べ，前述した想定環境に適合するかどうかを述べた．また，想定環境における問題点を述べ，その問題点を関連研究が解決可能かどうかを述べた．本章では，これらの問題点を解決するモデルである Personal Server Model を提案する．

3.1　モデルに関する議論

本節では，前章で述べた問題点をどのように解決するか，その解決モデルに関して議論を行う．

ユーザの移動問題

ユーザの移動問題は，ユーザがどこに移動し，どのセンサインフラストラクチャを使用しているかを他のアプリケーションが理解不能なために生じる．この問題を解決する手法としては，以下の二つがある．

- ユーザが移動しても，現在使用しているセンサインフラストラクチャを既知の外部システムが把握可能にする
- ユーザに関するセンサ情報を常に同じ場所に保持する

前者はユーザが移動する都度，現在使用しているセンサインフラストラクチャを外部システムに広告することで問題を解決する．ユーザの情報を取得する外部システムが限られている場合では動作するが，さまざまなアプリケーションが考えられる場合，全てのアプリケーションに対して広告することになり，現実的ではない．

後者は，センサ情報を常に同じ場所に保持するため，他のアプリケーションにとって，ユーザの移動に関する問題は生じない．しかし，他のアプリケーションおよびセンサインフラストラクチャは，ユーザの情報を管理している場所を知る必要がある．だが，ユーザ情報を管理している場所を一定とするのならば，アプリケーションは一度その場所を把握することで問題を解決できる．

従って，本研究では，移動問題を解決する後者の手法を用いる．また，情報を管理している場所を知るための手法を 3.5 節で提案する．

プライバシ保護問題

また，プライバシ保護を行うための手法としては，主に以下の二つがある．

- 情報を分割および分散し，個々の情報だけでは意味をなさないようにする．
- 情報を集中管理して保護する．

前者は，分散管理することで個々の保護が破られた場合の危険性を低下させる．しかし，情報の整合性や，情報を管理する場所の判断など，情報の管理が困難である．
後者は，集中的に情報を管理することで保護が容易になる．しかし，保護が破られた場合，すべての情報が漏洩することになり，危険性が高い．
本研究では，後者の管理の容易さを重視し，集中管理による保護を行う．さらに，プライバシ保護のための手法には以下の手法も同時に行う．

- 情報を取得するが，それが誰のものかを分からなくする

この手法は匿名化と呼ばれる．情報は，匿名化により意味をなさない情報の集合へと変化するため，プライバシを保ったまま統計情報を抽出するなどに適している．

モデルに関する議論のまとめ

以上の議論を踏まえ，前章で述べた二つの問題点を矛盾なく解決するためのモデルとしては，センサ情報を常に同じ場所に集中管理するモデルが適切であると考える．
また，センサ情報を含む，そのユーザに関する実空間情報を集中管理するためのデータ管理モデルとして以下の二つの可能性が考えられる．

- センサインフラストラクチャ指向
- ユーザ指向

前者は，センサ情報をセンサインフラストラクチャごとに集中管理するデータ管理モデルである．また，後者はセンサインフラストラクチャを使用しているユーザごとにセンサ情報を集中管理するデータ管理モデルである．
しかし，前者のセンサインフラストラクチャ指向でのデータ管理では，ユーザが現在使用しているセンサインフラストラクチャを把握できなければ，ユーザのセンサ情報を取得できない．結果として，想定環境において，ユーザの移動に対する問題点を解決できない．
後者の，ユーザ指向でのデータ管理では，ユーザの移動に関わらず，そのユーザに関する情報を取得できる．従って，ユーザの移動に対する問題点は解決することができる．しかし，ユーザの情報を保持している場所に対して，センサインフラストラクチャが情報を送信する必要がある．つまり，センサインフラストラクチャは，ユーザを識別し，そのユーザに応じたデータ管理場所に情報を送信する必要がある．

以上の議論より，本研究では，センサ情報を含むそのユーザに関する実空間情報全般をユーザ指向で集中管理するモデルを提案する．

3.2 Personal Server Model

前節までに述べた議論に従い，本研究では Personal Server Model を提案する．これは，全てのユーザは，自分に関する情報を保持するサーバを一つ保有し，そのサーバにセンサ情報も含めたさまざまな実空間情報を集中管理するモデルである．このユーザ固有のサーバを “Personal Server” と本研究では定義する．

センサデバイスからの情報は，そのセンサインフラストラクチャ内のデータベースではなく，直接ユーザの Personal Server に送信される．また，センサ情報だけでなく，ユーザの所属や名前，連絡先，スケジュールや住所録などの個人情報も実空間情報として Personal Server で管理される．

図 3.1 に Personal Server Model の概要を示す．この図では，実空間に存在する各ユーザそれぞれに対応するように，仮想空間に Personal Server が置かれ，それぞれの Personal Server 同士はネットワークで接続されていることを表している．また，センサデバイスが取得したセンサ情報は，センサインフラストラクチャによって取得され，Personal Server に送られている．

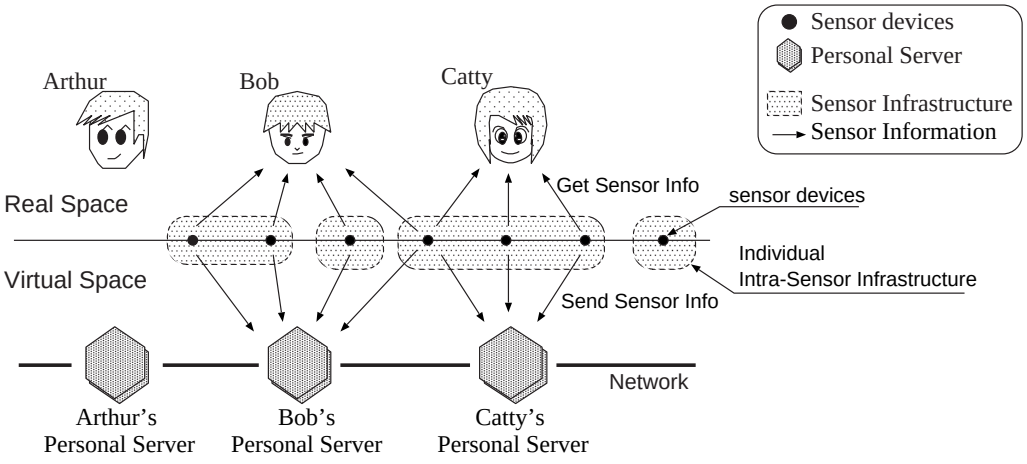

図 3.1: Personal Server Model

また，センサ情報はセンサデバイス管理者のデータベースに保持されることはなく，各ユーザの Personal Server に保持される．つまり，センサ情報はセンサインフラストラクチャに対して分散管理されることとなり，プライバシ保護を実現するとともに，ユーザ数の増加に対する規模性も提供できる．

Personal Server Model の前提機能

Personal Server Model では，全てのユーザが一意な ID(Globaly Unique ID，以下 GUID と呼ぶ) を保持していることを前提とする．この ID により，相手を特定することができる．また，センサデバイスはセンサデバイス自身が取得する位置情報などの情報に加えて，その ID も取得可能とする．この ID を利用することにより，ユーザの識別などを行える．

本研究では GUID の定義，管理，割り振りなどに関する議論は行わない．しかし，現在使用可能な ID 体系としては，Auto ID Center [10] や mu-chip[11] などがあげられる．Auto ID は，96bit の ID 空間を，mu-chip は 128bit の ID 空間を持つ．両者とも書き込み不可であり，製造段階で一意性が確保される．

この GUID は 4.3 節で後述する Personal Server 発見手法である，SDP(Server Discovery Protocol) のために使用される．SDP は，センサインフラストラクチャがそのユーザの情報を管理している場所を把握するために使用される．SDP により，前節で述べたユーザ指向でのデータ管理を行う際に生じる問題点を解決できる．また，SDP では，GUID に対応する Personal Server が存在する場合，必ずその Personal Server を発見できるものとする．

さらに，Personal Server Model におけるセンサデバイスは実空間情報を取得することに加えて，この GUID も同時に取得する．加えて，取得した情報を Personal Server へと送信するためのネットワーク接続性が必要である．そのため，本研究では，センサネットワークなどで想定している低機能センサデバイスよりもセンサデバイス自体は多少高度な処理能力を持つことを想定している．しかし，本研究では，センサデバイス自身には触れず，また，センサデバイスが取得する情報に関しては正規化などが行われていることを前提とする．

3.3 Personal Server

Personal Server はそのユーザに関する実空間情報を全て保持するサーバである．1.4 節での定義により，本研究では実空間情報とは，センサデバイスから取得したセンサ情報の他にユーザの名前や連絡先，スケジュールなど実空間におけるさまざまな情報を指す．Personal Server はそのユーザに関する実空間情報を全て保持する．実空間情報を保持するために，Personal Server はデータベース機能を持つ．

別の Personal Server あるいは他のアプリケーションは，実空間情報を取得したいユーザの Personal Server に対して接続し，そのユーザに関する実空間情報を要求する．Personal Server は，この要求に対して，データベースに保持されている実空間情報を返信する．この実空間情報を受信することにより，アプリケーションは，そのユーザに関する実空間情報をネットワークを越えて利用できる．

Personal Server は，アクセス管理機構を持ち，要求が不適切な通信相手からのものであれば，その要求を破棄する機能を持つ．センサデバイス管理者がセンサ情報を保持せず，インフラストラクチャに対して分散した Personal Server にセンサ情報が保持されることに加えて，このアクセス管理機構により，プライバシ保護が実現される．

また，Personal Server はユーザの情報を保持し，アプリケーションからの要求を待つ必要性があるため，常にネットワーク上に存在しているものとする．すなわち，Personal Server に対して接続が不可能な場合や，接続の切断などは発生しないものとする．また，Personal Server の移動も考えないと事とする．ただし，Personal Server は，インターネット上のどこにあっても構わない．

アクセス管理機構

Personal Server Model では，そのユーザに関わる情報をそのユーザの Personal Server が集中管理する．Personal Server は，他のアプリケーションなどからの要求に応じてセンサ情報を含む実空間情報を返答する．

また，プライバシ保護を実現するために，Personal Server には，アクセス管理機構を持つ．アクセス管理機構を通じてのみ，信頼関係のある相手には，そのアクセス権限に応じて適切な情報を公開することを可能とする．この際，前述した GUID を利用することで，相手の特定を行う．信頼関係のある GUID の取得は，名刺交換などで行われる．従って，GUID の取得，交換自体は本研究の範疇ではない．

また，信頼できる GUID の追加や信頼できない GUID の追加，アクセス権限の追加変更などのアクセス管理はユーザ自身が行う．そのため，管理がしやすいようにグループ単位でのアクセス管理を可能とする．例えば，“研究室” というグループを用意し，このグループに属する GUID に対しては自分の位置情報を公開するという，一括したアクセス権限の設定を可能とする．

3.4 プライバシ保護問題の解決

プライバシ保護に関しては，従来のモデルでは，センサの管理者という第三者が介在したために，ユーザの個人的な情報を第三者が保持することになっていた．しかし，本研究が提案する Personal Server Model では，センサ情報はセンサから直接 Personal Server に送信される．この通信が暗号化などで保護されることで，現在使用していない，他のセンサはセンサ情報を盗聴できない．また，ユーザが現在使用しているセンサの管理者は，センサ情報を全て盗聴可能であるが，本節で後述する GUID しか分からないため，誰のセンサ情報か分からない．

もしも，センサインフラストラクチャの管理者が誰のセンサ情報かを把握する場合は，Personal Server に直接アクセスし，名前という実空間情報の要求を行う必要がある．しかし，Personal Server には，アクセス管理機構が存在し，不適切な要求を拒否することができる．

結果として，Personal Server Model では，以下に述べる対象へのプライバシ保護が行われる．

現在使用しているセンサインフラストラクチャの管理者は センサ情報は取得できるが，誰のセンサ情報か分からない．

他のセンサインフラインフラストラクチャの管理者は センサ情報は盗聴できない．

他のユーザやアプリケーションは アクセス管理により，不適切な要求として拒否される．

ただし，本研究ではユーザが正しいユーザであるという正当性の検証および，センサデバイスあるいはセンサインフラストラクチャから Personal Server へと送信されるセンサ

情報における正当性の検証はなされているとする．これらの正当性を検証するためには，IPSec[12] や IKE[13] などを利用することが考えられる．

3.5 ユーザ移動問題の解決

本研究が想定している環境では，グローバル型センサインフラストラクチャ環境の中をユーザが移動する．そのため，ユーザに関するセンサ情報を取得する場合，どこに要求すればよいか分からないという問題が生じる．Personal Server Model では，そのユーザに関する情報は一つの Personal Server で管理されているため，常に同じ Personal Server に対して要求を行えばよい．これにより，ユーザの移動に関する問題点は解決される．

Personal Server Model では，そのユーザに関するセンサ情報を取得したセンサデバイスは，そのユーザの Persoanal Server に対して，取得したセンサ情報を送信する．また，あるユーザの情報を得ようとする他のユーザは，そのユーザの Persoanal Server に対してアクセスすることでその情報を得ることができる．

しかし，Persoanal Server は，インターネット上のどこにあっても構わないという前提がある．これは利点ではあるが，同時にセンサはセンサ情報の送信先が不明であるという欠点となる．また，他のユーザも情報の要求先が不明となる．従って，Personal Server Model では，Persoanal Server のネットワーク上の位置が分からなくなるという問題点が存在する．

本研究では，SDP(Server Discovery Protocol) を提案することでこの問題点を解決する．SDP は，各ユーザが保持している GUID を利用して，そのユーザが持つ Persoanal Server のネットワーク上の位置を取得する．これにより，ユーザの現在位置および使用しているセンサデバイスに関わらず，Personal Server のネットワーク上の位置を検出することができる．

本節では SDP に必要な機能について述べる．SDP の機能設計に関しては 4.3 節で後述する．

SDP の機能要件

SDP に対する機能要件として，以下の機能があげられる．

- 規模性

  原則として，Personal Server は一人につき一つ用意される．そのため，数十万，数百万の規模で Personal Server は存在する可能性がある．多数の Personal Server の中から，できるだけ速やかに Personal Server を発見することが SDP には求められる．

- ID の匿名性

  GUID は，そのユーザを一意に識別する識別子である．この GUID を元に，センサデバイスは Personal Server を発見する．

これはつまり，GUID さえ取得できれば，その人の関する実空間情報を保持している Personal Server を発見することができるということであり，プライバシに関わることである．従って，GUID は第三者に分からないように保護されるべきである．

一方，tcpdump や port scan などのさまざまなネットワーク層での管理ツールを使うことにより，そのホストが Personal Server である，ということは容易に識別できる．しかし，そのホストが Personal Server である，ということがたとえ判別できたとしても，その Personal Server が誰のなのかを識別できなければ構わない．そのため，Personal Server のネットワーク上の位置を示す IP アドレスに関する匿名性は，本研究では特に扱わない．

- 低機能ノードへの対応

  SDP は，Personal Server を発見する必要があるノードすべてが行う．これは，センサノードも例外ではない．しかし，一般にセンサノードは処理能力やストレージ，ネットワーク帯域などが非常に限定されている．このようなノードでも SDP を使用することが可能でなければならない．

- 確実性の保持

  SDP が収束するためには，必ず GUID から IP アドレスへと変換できなければならない．ただし，その GUID および IP アドレスの両者が登録されていることをその条件とする．

3.6 実空間情報の表現手法

Personal Server に保持されるユーザに関する実空間情報は，他のアプリケーションなどと交換，利用ができなくては意味がない．従って，Personal Server の中に保存されている実空間情報を他のアプリケーションと交換するためのデータフォーマットおよびプロトコルを規定する必要がある．

Personal Server にはそのユーザに関する実空間情報が保持される．その際，Personal Server の中でどのような形式で実空間情報が保持されるかは実装依存であるが，他の Personal Server あるいは他のアプリケーションがその実空間情報を利用する場合には，共通の形式で実空間情報が表現されていなければならない．

従って，Personal Server Model には，実空間情報の表現手法と，その交換手法が必要となる．本論文では，Personal Server 自身に加えて，Personal Server に保持される実空間情報を交換する際の表現手法についても述べる．

また，Personal Server 内に保持されている実空間情報は，今後さまざまな種類が必要に応じて追加されるであろう．しかし，現段階では今後どのような情報を扱うようになるか予測不可能である．従って，実空間情報の表現，交換手法は拡張可能であるべきである．

本研究では，全ての情報にはその情報が誰に関しての情報かを示す GUID，どこで取得されたかを示す位置情報，いつ取得されたかを示す時刻情報が暗黙的に付加されていると

考える．従って，これら三つの情報を，重要な情報とし，交換に必要な最低限の情報として，必ず実空間情報の交換に使用する．以下にその情報を列挙してまとめる．

- GUID
- 位置情報
- 時刻情報

第4章　Personal Server Modelの設計

本章では，前章で述べた Personal Server Model に必要な以下の機能群の設計を述べる．

- Persoanal Server
  各ユーザが個々に保持する，実空間情報が管理されるサーバを本研究では Personal Server と呼び，その機能を 4.1 節に述べる
- 実空間情報の表現形式
  Personal Server 内に保持される実空間情報を他のアプリケーションが利用するために必要である共通表現形式を本研究では PML と呼び，その概要を 4.2 節に述べる
- Personal Server のネットワークにおける位置取得
  ユーザ指向によるデータ管理モデルでは，センサインフラストラクチャが，ユーザを識別し，そのユーザが持つ Personal Server に情報を送信する必要がある．この問題に対する解決案を本研究では SDP(Server Discovery Protocol) と呼び，その詳細を 4.3 節に述べる

図 4.1 に Personal Server Model における機能群間の関係を図示する．図中の網掛け部分が本研究が提供する機能である．本章では各機能の設計について述べる．

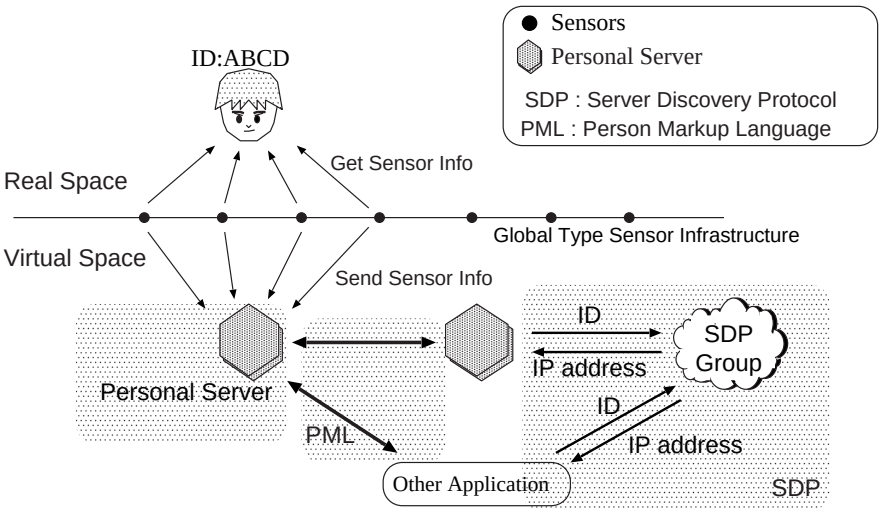

図 4.1: Personal Server Model に関する登場人物の整理

4.1 Persnal Server

本節では，実空間における情報を保持している Personal Server について述べる．

前述の通り，Personal Server は各ユーザ一人一人に一つ存在する．また，3.2 項で述べたように，各ユーザは固有の GUID を割り振られており，Personal Server もその GUID を保持している．

以下に Personal Server が有する機能について述べる．

4.1.1 ネットワーク接続性

前述の通り，Personal Server は，そのユーザに関するセンサ情報を保持するためのサーバである．本研究が想定するコンピューティング環境上では，ユーザの周辺環境の至る所にセンサデバイスが設置され，それらのセンサデバイスは常にユーザに関する情報や，周辺環境の情報を取得している．そして，センサデバイスは取得した情報を Personal Server に送信する．つまり，Personal Server は，常にセンサデバイスからの情報を受信できる環境にいなければならない．そのため，Personal Server は，無線通信環境ではなく，安定した有線通信環境が望ましい．

4.1.2 データベース

Personal Server は，センサデバイスからの情報を常に取得し，保持している．そのために，Personal Server はデータベースを必要とする．

このデータベース内部には，センサデバイスから取得した，実空間におけるユーザの情報など，Personal Server が扱うべき情報が保持される．

データベース内にどのような形式で情報が保持されるかに関しては，本研究の対象外とする．しかし，Personal Server に対して情報を送信する，あるいは Personal Server から情報を受け取る形式に関しては，PML に適合する形式でなくてはならない．

また，Personal Server 内のデータベースは過去の履歴情報を保持しなくてもよい．その理由として，Personal Server の機能は，あくまで現在のユーザ状況を仮想空間上に保持することであり，履歴情報を持つことにより，検索などによってその機能が制限されることはあってはならない．

履歴情報を保持する場合は，定期的に Personal Server に対して履歴を保持したい情報のみを要求し，Personal Server の外部にあるデータベースを保持するべきである．Personal Server の機能は必要最低限に押さえ，その他のさまざまな機能は，あくまで拡張機能とする．

4.1.3 アクセスコントロールモジュール

Personal Server に対する要求には全て，要求元のユーザが保持する GUID が付加されていなければならない．この GUID を元に，Personal Server はアクセス管理を行う．また，

そのGUIDの正当性を確認する認証機構が必要である．Personal Server内のアクセスコントロールモジュールでは，このアクセス管理を行う．また，そのために，アクセスコントロールモジュールは，アクセスコントロールリストを持つ．

アクセスコントロールリストには，GUIDと要求があった場合にそのGUIDに対して送ってもよい情報のリストとGUIDが記されている．Personal Serverは，Personal Server自身が保持するアクセスコントロールリストと受け取ったGUIDを比較し，要求された情報の中で，適切な情報のみを返答する．

また，アクセス管理リストには，個別のGUIDの他に，グループが存在する．グループには，複数のGUIDが属している．グループを用いて，アクセス管理情報を設定することで，そのグループに属しているGUIDからの要求に対しては，そのグループの送信ポリシに即した情報のみが送られることになる．ただし，3.3項で述べたように，グループはあくまでユーザに対してアクセス管理の利便性を向上させるためのものである．従って，グループは単一個人のPersonal Server内のみで定義，管理される．つまり，あるユーザにとって，あるGUIDがあるグループに属しているからといって，別のユーザでも同じGUIDが同じグループに属しているとは限らない．

アクセスコントロールモジュールの図を4.2に示す．この図では，ID:ABABというユーザに対しては，名前や連絡先，位置情報などが要求に応じて送信することが可能だが，Group:Labに属している，ID:CDCDというユーザやID:EFEFというユーザからの要求には，名前と連絡先しか送信しない．

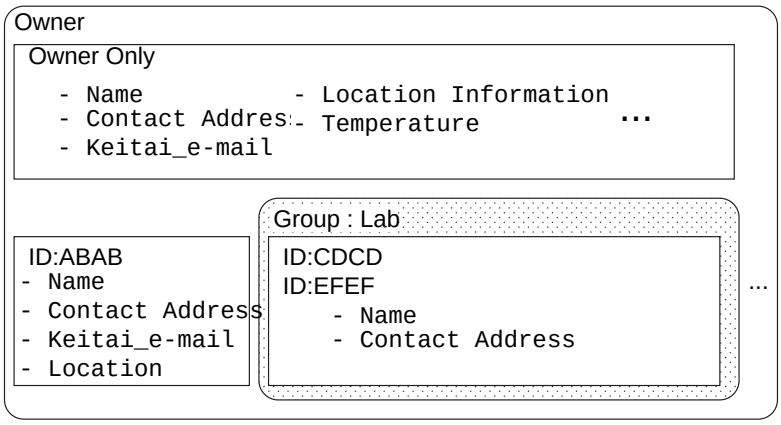

図 4.2: Personal Serverにおけるアクセス管理

アクセスコントロールリストは新たなGUIDからの接続があった場合に更新される．この時，そのGUIDを持つPersonal Serverに対してアクセスし直すことでその人の名前や所属などを取得し，それをユーザに対して提示する．これにより，ユーザはそのGUIDに対してどのようなアクセス権限を与えるかを決定することができる．

4.1.4 拡張モジュール

前述したように，Personal Server 自身は，センサデバイスの情報を保持する機能及びアクセス管理機能しか持たない．しかし，Distributed Context-Aware Systems[14] で示されているように，あるユーザの実空間情報をより正確に取得するためには，そのユーザだけでなく，近隣にいるユーザの実空間情報などが有用となる．このように，単なる情報を管理する機能だけでなく，Personal Server 自ら情報を取得するなどの機能を追加することでさらに利便性を向上できる．また，拡張モジュール自体は，Personal Server の外部および内部のどちらも設置できる．Personal Server 内部の拡張モジュール機能は，その外部および内部の区別なく拡張モジュールを使用できる．

拡張モジュールは，Personal Server を管理しているユーザと同一のユーザが管理する．拡張モジュールは Personal Server の情報を利用して，さまざまな機能を実現する．これらの拡張モジュール群は形式としては，ネットワーク上にある他のアプリケーションと同じように，アクセスコントロールモジュールを通して取得した PML を利用する．すなわち，他のアプリケーションと同じく，拡張モジュールは外部ネットワークに設置することもできる．

拡張モジュールの例として挙げられるのは，例えば，ユーザが保持している携帯電話がアクセス管理を行う拡張モジュールを持つことが考えられる．Personal Server が要求を受信すると，Personal Server は要求に付随する GUID と，GUID から導き出される Personal Server へ問い合わせた名前，要求する情報の三者を携帯電話に送信し，携帯電話はその情報をユーザに対して提示する．この情報を元に，ユーザ自身がその要求に対して情報を応答するかどうかを決定する．本研究では Personal Server におけるアクセス管理を行うが，拡張モジュールにより，アクセス管理を人の手を介在することも可能となる．ユーザの手を介在することで，より強固なプライバシ保護を行うことができる．

もう一つの例としては，そのユーザの情報を定期的に保存し，行動履歴を表示する web サーバなどに，この拡張モジュールを設置することが考えられる．この web サーバは履歴情報と今後のスケジュールの情報を持ち，過去に行った場所およびこれから行くべき場所などをグラフィカルに表示する．もちろん，web サーバは，パスワードを要求し，本人しか表示しないようにする．このように，拡張モジュールを利用することで，Personal Server 自体の機能を軽減することが可能となる．

しかし，拡張モジュールが Personal Server の情報を利用しようとするときは必ず最初に本人認証が行われる．ユーザ本人が使用する拡張モジュールであることが，鍵交換などによって確認されることで，拡張モジュールは Personal Server からの情報を取得できる．Personal Server は，拡張モジュールに対しては，上記のアクセスコントロールモジュールを経由せずに直接 PML を送信する．

図 4.3 に，Personal Server のモジュール拡張構造を示す．この図では，一つの Personal Server が五つの拡張モジュールを持ち，情報を取得していることが分かる．また，拡張モジュール同士が情報をやりとりすることで，さらに高度な処理が可能である．

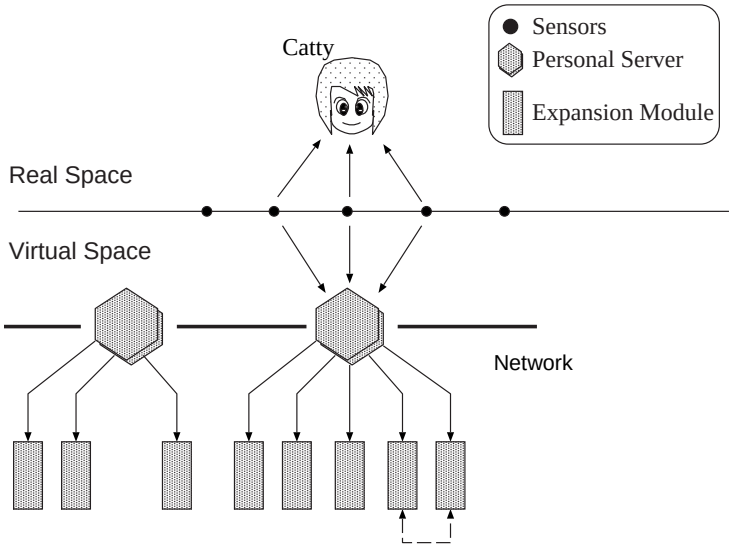

図 4.3: Personal Server の拡張モジュール構造

また，Personal Server はユーザ一人一人が保持する．ユーザの技術練度を高く想定することはできないため，Personal Server の設置や管理にかかるコストは低くなければならない．そのためにも，Personal Server が有する機能は最低限に抑える必要がある．

4.2 PML(Person Markup Language)

本研究で提案する Personal Server Model では，Personal Server には実空間情報が保持されている．本節では，実空間情報，表現する言語である，PML(Person Markup Language) について述べる．

4.2.1 機能要件

3.3 節で述べたように，実空間情報はどのような形式で Personal Server に保持されていても構わない．しかし，この実空間情報を他の Personal Server あるいは他のアプリケーションが利用する場合には，共通の形式で実空間情報が表現されていなければならない．従って，実空間情報を表現するための言語である PML にはデータの交換性が必要となる．

また，実空間情報には，さまざまな情報を含むため，現在使用不可能なあるいは有益とは考えられていない情報をも使用することが将来あり得る．このような場合に対応するために，PML には今後さまざまな情報に対応するための拡張性が必要となる．

4.2.2 PMLの設計

前述のようにPMLにはデータの交換性と拡張性が重要である．Location-aware[15]は，LDAPを用いて物体の位置情報を表現している．LDAPでは，LDIF(LDAP Data Interchange Format)というデータフォーマットを用いて階層構造化された情報を表現している．この階層構造は独自に規定できるため，拡張性に優れている．

しかし，現在LDIFはLDAPでしか用いられていない．そのため情報を取得したい他のアプリケーションは必ずLDAPを用いなくてはならず，データの交換性に劣る．そのため，LDAPおよびLDIFは，Personal Server Modelにおける実空間情報の表現手法として利用するには不適切である．

一方，インターネットにおけるデータの表現手法として多く使われているデータフォーマットとして，XML[16]がある．XMLの利点を以下に示す．

- タグを自由に決めることができ，拡張が容易である．
- 共通のDTDを使うことにより，データ交換が容易．
- 意味情報を記述することができる
- ドキュメント毎に定義された論理構造にそって記述可能．
- 様々なデータから特定の情報を抽出可能．
- 厳密な文法チェックが可能であり，プログラムからの管理が容易

前述のようにPMLでは特にデータ交換が容易な点と拡張が容易である点が重要となる．

また，XMLの拡張として，RDF(Resource Description Framework)がW3Cで勧告されている．RDF[17]は，インターネット上におけるさまざまな資源を記述することが可能なグラフベースモデルである．同時に，RDFはある資源と他の資源との関係を表現することができる．RDFを用いることにより，実空間情報を資源として容易に表現することができる．

本研究では実空間情報を表現するための手法として，XMLおよびRDFを利用してPMLを設計する．

4.2.3 PML

3.6節で述べたように，Personal Server内部の実空間情報は必ずGUID，位置情報，時刻情報が含まれる．これをPMLにおいてはcore部と呼ぶ．PMLは必ずcore部を含まなければならない．

PMLは，core部以外にさまざまな情報を記述することができる．図4.2では，名前や連絡先，携帯E-mailアドレスなどが位置情報のほかにPersonal Serverの中に保持されていることが示されている．

core 部以外のこれらの拡張情報は，RDF スキーマを新たに記述することで追加できる．XML および RDF を利用することにより，拡張が非常に容易に行える．また，PML における要件は core 部を含むことであり，それ以外の拡張部ではどのような情報を含むか，あるいは含んではいけないかは規定しない．従って，センサインフラストラクチャの管理者や実空間情報を利用するアプリケーションの作者は，XML および PML に従い，core 部を含む限り，自由に PML を拡張することができる．

4.3 SDP(Server Discovery Protocol)

本節では，実空間情報を保持する Persoanl Server のネットワーク上の位置情報である，IP アドレスを得るためのプロトコルである，SDP(Server Discovery Protocol) の設計について述べる．SDP は実空間ネットワーク環境および想定環境を実現するにあたって生じるユーザの移動問題を解決する．

SDP は，Personal Server の情報を取得する他の Personal Server，さまざまな他のアプリケーションおよび，センサデバイス情報を送信するセンサデバイスが使用する．この時，アプリケーション自体が SDP の機能を保持せずに，ネットワーク上の近隣に存在する SDP 機能を持つノードに対して GUID を送信し，受信した SDP 機能を持つノードが SDP を行うという利用形態も可能とする．

まず始めに，SDP を設計するにあたって前提となる条件を整理する．続いて，その条件に基づいた，SDP の設計を述べる．

SDP の設計

まず始めに，3.5 節で述べた機能要件に基づき，SDP を設計する．

SDP は，ID から IP アドレスを導き出すためのプロトコルである．このような検索のためのデータベースを管理する手法は大別して二種類存在する．

- 階層構造モデル
- P2P モデル

前者の階層構造モデルの例としては DNS があげられる．また，後者の P2P モデルの例としては，Gnutella や Napster などがあげられる．

階層構造モデルで SDP を設計した場合，各ホストが検索のためのデータベースを権利委譲などにより分割して保持する．各ホストは上位ホストを持ち，そのホスト内で要求されたデータが見つからなかった場合は，上位ホストに対してデータを要求する．

しかし，上位による管理は必ず行わなければならない．また，階層構造の上位に位置すれば位置するほど，その管理は厳重に厳密に行われなければならない．加えて初期段階で，上位と下位の関係を明確にし，権利関係などを定義する必要がある．そのため，管理コス

トが高いという点が生じる．従って，階層構造モデルは 3.5 節で述べた要求事項に適合しない．

一方，P2P モデルは分散型の検索を行う．つまり，各ホストは対等の関係であり，階層構造モデルのように上位と下位の関係などは存在しない．従って，特定のホストにデータベースのインデックスが集中することがなく，各ホストに対して要求される性能は低い．また，上位下位などではなく，ノード同士が対等の関係であると言うことは，管理する責任が集中することがなく，個々のホストの管理コストが低い．また，集中管理ノードが存在しないため，全体としての耐久性が高くなる．

両者の特徴を表 4.1 にまとめる．

表 4.1: 検索データベース構造におけるモデル比較

項目 / モデル	規模性	簡便性	低機能ノードへの対応	検索速度
階層構造型モデル	あり	低い	なし	早い
P2P モデル	あり	高い	あり	遅い

以上述べてきた比較の結果，SDP は P2P モデルに基づき設計する．

SDP では，SDP 機能を持つノード (以下 SDP ノードと呼ぶ) が集合し，SDP グループを形成する．SDP ノード同士は，対等の関係であり，上位下位の関係はなく，SDP グループ内にフラットに存在している．

- Personal Server の検索
- 新規 Personal Server の登録

Personal Server の検索を行うユーザは，手近な SDP ノードに対して GUID 検索要求を送信する．ユーザにとって，手近な SDP ノードとは自らの Personal Server であることが多いが，近隣に存在するセンサデバイスなども利用可能である．

GUID 検索要求を受け取った SDP ノードは，SDP グループに属する他の SDP 機能を持つノードに対してその GUID 検索要求を転送する．転送された検索要求を，GUID に対応する IP アドレスのデータを保持しているノードが受け取った場合，その IP アドレスを検索要求送信元に直接送信する．

新規 Personal Server の登録も同様に GUID と対応する IP アドレスの対を登録要求として他の SDP ノードに送信する．また，4.1 節で述べたように，Personal Server は常にネットワーク接続性を持つ．従って，Personal Server の消失は考えないものとする．これらの SDP の動作の詳細に関しては，5.3 節で後述する．

SDP の概要を図 4.4 に示す．

この図では，SDP 機能を持つノードが集合し，SDP グループを形成していることが分かる．また，ユーザが BBBBB という GUID を持つ Personal Server の IP アドレスを自身の Personal Server に要求し，その検索要求が転送されていく様子および，検索結果が直接返信されていることが分かる．

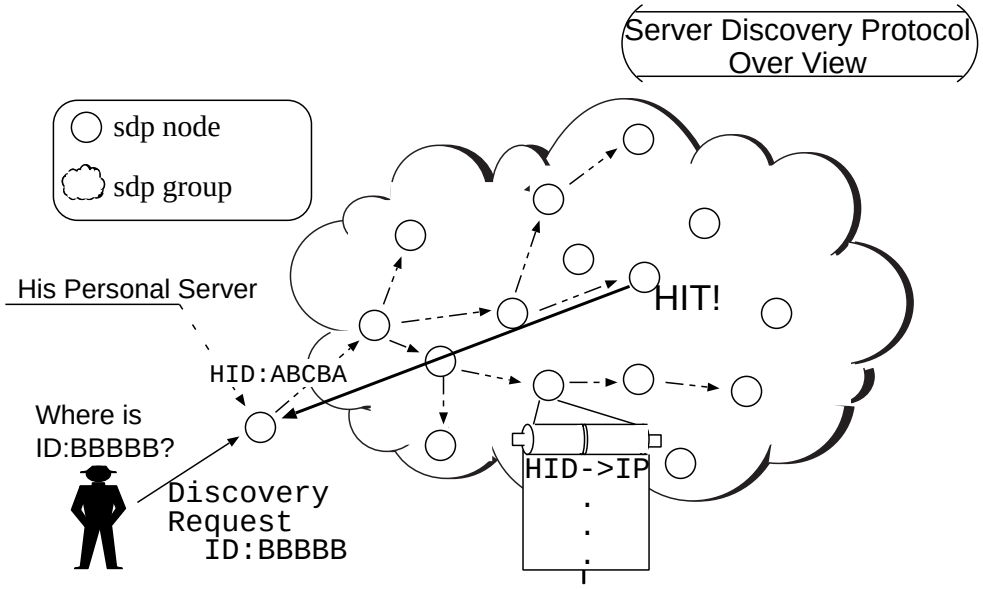

図 4.4: Server Disocovery Protocol 概要

以上述べた SDP の基本設計に続いて，3.5 節で述べた機能要件を満たす設計について述べる．

- 低機能ノードへの対応

  SDP はセンサデバイスも利用する．また，センサデバイスは一般的に低機能であるため，これらの低機能ノードでも利用可能であるように対応する必要がある．同時に，単なるアプリケーションも SDP を利用する必要がある．

  そのため，ネットワーク的に近距離にある SDP 機能を持つ高機能なノードに，検索などの機能を代替してもらう手法を取る．これにより，低機能なノード，あるいは，SDP の機能を持たないアプリケーションなどが SDP を利用することが可能になる．

  また，多数の低機能ノードによる検索要求を代替することで高機能ノードの負荷が増大した場合でも，前述したように SDP では，新規ノードの設置が簡易であるため，容易に高機能ノードを増やすことができる．

- GUID の匿名性

  P2P モデルでは，情報を複製し，多数の第三者が情報を保持することにより単一ノードでは不可能な量のデータを全体として保持することができる．SDP の場合，複製する情報の中には GUID および対応する Personal Server の IP アドレスが保持されている．しかし，情報の複製および配布は規模性という点では利点となるが，GUID の匿名性という点では欠点となる．

  SDP では，ユーザ GUID をあらかじめ MD5 などの一方向性ハッシュ関数を利用し，ハッシュ化する．つまり，元のユーザ GUID を知っているノードのみがユーザ GUID と IP を結びつけることができる．

このハッシュ化された GUID を検索に利用することにより，第三者である他のノードは，元のユーザ GUID を推測することはできず，ユーザ GUID の匿名性が保証される．

また，SDP の範囲外で GUID が漏洩した場合でも，4.1 節で述べた Personal Server でのアクセス管理機構により，信頼できない第三者に情報が漏洩することは防がれる．

- 確実性の保持

  確実性を保持するために，SDP は GUID と対応する IP アドレスのデータを持つノードが SDP で解決可能な範囲内に必ず存在することを前提条件とする．この条件により，SDP は収束し必ず IP アドレスを取得することができる．

  この要件を満たすために，新しく GUID と IP アドレスを登録したノードは必ずその両者を保持する．これにより，確実性を保持できる．

  また，検索にはタイムアウトを設定し，その GUID および IP アドレスの両者が登録されていない場合には到達不能のエラーを返す．タイムアウトは，全体ノード数に依存し，設定される．これは，SDP 全体のノードが増加することにより，IP 情報を保持しているノードへの到達に要する時間が長くなるためである．

4.4 本章のまとめ

本章では，Personal Server Model を実現するために必要な機能について述べた．Personal Server Model には，Personal Server，PML，SDP の三つの機能が必要である．

Personal Server は，そのユーザに関する実空間情報を全て保持している Personal Server であり，要求に応じて実空間情報を送信する．また，Personal Server は要求に添付されている GUID に基づいてアクセス管理を行う．

PML は，実空間情報を表現するための拡張言語仕様であり，実空間情報の交換などに使われる．本研究では，Personal Server 内に保持されている実空間情報に関しては定義しないが，他の Personal Server やアプリケーションなどとの実空間情報の交換には PML を使用する．

SDP は，GUID からそのユーザの Personal Server が持つ，ネットワーク上の位置 (IP アドレス) を取得するためのプロトコルである．SDP は P2P モデルに基づいて設計され，簡易性および GUID の匿名性を優先して設計された．SDP により，ユーザの移動問題および Personal Server Model が持つ問題点が解決される．

第5章　Personal Server Modelの実装

本章では，Personal Server Model を実現するためのプロトタイプ実装である，“D2 システム” について述べる．D2 システムは，Personal Server，PML，SDP の三つの機能から構成される．センサデバイスそのものに関しては，3.2 節で述べたように本研究の対象外とする．

5.1　Personal Server の実装

Personal Server は，以下の環境でプロトタイプ実装が行われた．

表 5.1: Personal Server のプロトタイプ実装

項目	スペック
CPU	PentiumIV 1600MHz
Memory	256M
OS	FreeBSD-4.7R
データベース	PostgreSQL-7.2.2

4.1 節で述べたように Personal Server はネットワーク的に安定した環境で運用しなければならない．そのため，プロトタイプ実装では安定した OS である，FreeBSD-4.7R および安定した有線通信環境を使用して実装された．

図 5.1 に Personal Server のプロトタイプ実装の概要を図示する．この図では Personal Server は三つのポートを保持していることを示している．一つ目は外部通信用，二つ目はユーザによる管理 (コントロール) 用，三つ目は内部データベースへのアクセス用である．このうち，コントロール用ポートおよびデータベース用ポートは Personal Server と同一ホストからしかアクセスすることはできない．これにより，セキュリティを確保している．

Personal Server からの返答はステートレス型である．すなわち，Personal Server は要求を内部に保持せず，要求がある都度該当する実空間情報を要求元に送信するだけである．ステートレス型にすることで，Personal Server の負荷は軽減される．また，送信した情報をどのように解釈するかは，Personal Server ではなく要求元に依存する．

また，Internet Car Project[18] で行われた，ワイパ動作による雨量調査など，多数の情報を集めることにより生成される情報も存在する．このような複数の Personal Server からの情報を元に，統計処理を行うことでより広範囲で高精度な情報を生成する場合は，複数の Personal Server に対してその都度要求を複数回行う必要がある．その際には，統計情報作成者は各ユーザに対してセンサ情報取得の許可を取る必要がある．

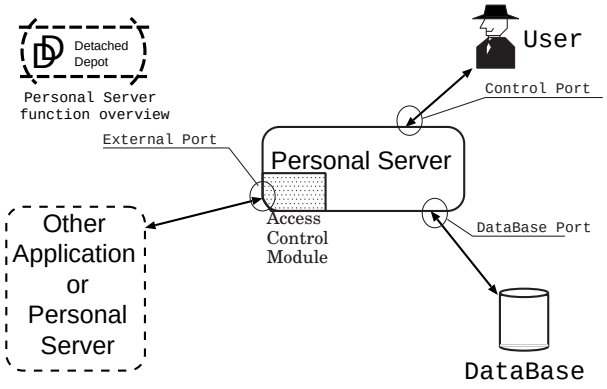

図 5.1: Personal Server のプロトタイプ実装概要

Personal Server 内部のそれぞれの機能について以下に述べる．

アクセスコントロールモジュール

他のアプリケーションとの通信は外部通信用ポートを使用する．また，外部通信用ポートの送受信はアクセスコントロールモジュールによるアクセス管理が行われる．

アクセスコントロールモジュールが使用するアクセスコントロールリストは，Personal Server 内部に存在し，ユーザ自身が追加，変更などを行うことで管理する．コントロール用ポートはユーザによるアクセス管理のために使用される．

データベース

Personal Server は，データベース用ポートを通じてデータベースを利用する．プロトタイプ実装では，データベースに PostgreSQL[19] を使用した．PostgreSQL は RDBMS(Relational Database Management System) であり，多くの UNIX 系システムで動作する．また，動作も軽量である．プロトタイプ実装では，Personal Server およびデータベースは同一のホストで動作した．

拡張モジュール

拡張モジュールは，コントロール用ポートではなく，外部接続用ポートを利用する．これにより，拡張モジュールはネットワーク上に設置することも可能となる．また，外部接続用ポートを利用することにより，Personal Server は外部と内部のどちらに拡張モジュールが設置されていても同じように使用できる．

また，拡張モジュールおよび，Personal Server の情報を利用するアプリケーションは 5.4.5 項で後述するライブラリを使用することで，容易に構築できる．

5.2 PMLの実装

図5.2にPMLの例を示す．この例では，4.2節で述べたPML のcore部が，以下のようになっていることが分かる．

GUID: HZMHMGL

Location: t20

TimeStamp: 2002-09-10 18:24:49

```
<rdf:RDF
  xmlns:pml="http://www.sfc.wide.ad.jp/~shirou/dtd/d2/pml/"
  xmlns:person="http://www.sfc.wide.ad.jp/~shirou/dtd/d2/person/">

  <pml:PMLcore>
    <pmlc:GUID>HZMHMGL</pmlc:GUID>
    <pmlc:Location>t20</pmlc:location>
    <pmlc:TimeStamp>2002-09-10 18:24:49</pmlc:TimeStamp>
  </pml:PMLcore>

  <person:person>
    <person:name>WAKAYAMA Shirou</person:name>
    <person:nickname>Shirou</person:nickname>
    <person:e-mail_addr rdf=resource="shirou@sfc.wide.ad.jp" />
    <person:organization>KEIO SFC</person:organization>
  </person:person>

</rdf:RDF>
```

図 5.2: PMLによる表現の例

また，PMLに必ず含まれるcore部の他に，Person部が定義されていることが分かる．このPerson部には，そのユーザの名前や連絡先，組織名などが含まれている．このRDFスキーマを参照することにより，他のアプリケーションやPersonal Serverもperson部で表現される，そのユーザの情報を利用することができる．このように，PMLはcore部以外の情報を拡張して表現することができる．

図5.3にRDFスキーマで表現されているPMLのCore部を記す．

5.3 SDP(Server Discovery Protcol)の実装

全てのユーザはGUIDを保持している．SDPはそのGUIDを元に，そのユーザが保有するPersonal Serverのネットワーク上の識別子であるIPアドレスを取得する．

SDPを行うことが可能なノードをSDPノードと呼び，SDPノードの集合をSDPグループと呼ぶ．また，SDPを行うことができない他のノードやアプリケーションは，SDPノードに対して後述するQueryを送信することでSDPを代替してもらうことが可能である．

```
<? xml version=1.0 encoding="EUC-JP"?>

<!DOCTYPE rdf:RDF [
        <!ENTITY rdfns   'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
        <!ENTITY rdfsns  'http://www.w3.org/2000/01/rdf-schema#'>
   ]>

<rdf:RDF xmlns:rdf="&rdf;"
         xmlns:rdfs="&rdfs;">

<rdf:Description rdf:about="http://www.sfc.wide.ad.jp/~shirou/rdf/pmlcore">
  <rdfs:label xml:lang="ja">PML Core MEtadata Element Set, Version
  0.1:Reference Description</rdfs:label>
  <rdfs:comment xml:lang="euc-ja">PML の Core 部．PML には必ず含まれる．
  </rdfs:comment>
</rdf:Description>

 <rdf:Description rdf:ID="PMLCore">
  <rdf:type  rdf:resource="&rdfs;Class"/>
  <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
 </rdf:Description>

 <rdf:Description rdf:ID="GUID">
  <rdf:type rdf:resource="&rdfs;Class"/>
  <rdfs:subClassOf rdf:resource="#PMLCore"/>
 </rdf:Description>

 <rdf:Description rdf:ID="Location">
  <rdf:type rdf:resource="&rdfs;Class"/>
  <rdfs:subClassOf rdf:resource="#PMLCore"/>
 </rdf:Description>

 <rdf:Description rdf:ID="TimeStamp">
  <rdf:type rdf:resource="&rdfs;Class"/>
  <rdfs:subClassOf rdf:resource="#PMLCore"/>
 </rdf:Description>

</rdf:RDF>
```

図 5.3: RDF スキーマによる PML Core 部の定義例

SDP では，GUID の匿名性を確保するために，GUID を一方向ハッシュした Hashed ID(HID) を使用する．プロトタイプ実装ではハッシュ化アルゴリズムとして，MD5 アルゴリズムを使用している．

5.3.1 SDP ノードが保持する情報

各 SDP ノードは以下の情報を保持する．

Convert テーブル: HID から IP アドレスへと変換するためのテーブル

Neighbor リスト: その SDP ノードが保持する他の SDP ノードのリスト

Query リスト: 受信した Query のシークエンス番号を保持するリスト

Convert テーブルは，HID から IP アドレスへと変換するためのテーブルである．Convert テーブル内には GUID そのものではなく，一方向ハッシュ化された HID しか保持せず，Convert テーブルのみが他の SDP ノードと共有される．そのため，SDP グループ内において GUID の匿名性は確保される．

また，Neighbor リストは，その SDP ノードが把握している他の SDP ノードの IP アドレスが記されている．すなわち，Neighbor リストに記載されているノードには，Query を送信することが可能だと言える．

Query リストは，受信した Query に付与されているシークエンス番号を一定数保持している．このリストに基づいた Query のチェックにより，Query のループが防げる．

これらの情報を利用し，SDP ノードは SDP を行う．

5.3.2 Query の種類

SDP における Query には，以下の種類があり，それぞれの Query にはさらに request と reply が存在する．各 SDP ノードは Query を最初に送信するときに，時間情報から生成されたシークエンス番号，その SDP ノードの IP アドレスの二つの情報を Query に付与する．

また，各 Query にはホップカウント情報が付与される．このホップカウントは，SDP ノードが Query を受信するたびに減少され，0 以下になった場合 Query は破棄される．これにより，Query のループが防げる．

- Discovery
  - Request：HID を含む Query を送信し，IP を要求する．
  - Reply：HID に対応した IP を要求元に直接送信する．
- Registration
  - Request：HID と IP の対を送信し，Convert Table に加えるように要求する．
  - Reply：Request に対し，すでに登録されている HID があれば，NACK を付加し，応答を返す．
- Neighbor
  - Request：IP を送信し，Neighbor Table に加えるように要求する．
  - Reply：Request に対して，受け取った旨を伝える．

5.3.3 Query 送信

前項で述べたように，Query を最初に送信するときにはシークエンス番号およびそのノードのアドレスを Query に付与する．

Query は，SDP ノードが保持する Neighbor リストから SDP ノードを一つ選出し送信する．プロトタイプ実装では，乱数による選出方法を使用した．また，Query 送信後，SDP ノードは一定時間待機し，その間に Reply が返ってこない場合は，再び別の SDP ノードを Neighbor リストから選出し，送信する．また，選出した SDP ノードに対して接続できない場合は，その SDP ノードは SDP グループから削除されたものとして，その SDP ノードを Neighbor リストから削除した上で，別の SDP ノードを選出し，送信する．

5.3.4 Neighbor の追加

Neighbor の追加は，Name-Dropper アルゴリズム [20] を使用する．

Name-Dropper アルゴリズムでは，Query を受信したノードは，その Query を送信したノードを Neighbor リストに加える．また，Query を送信したノードが保持している Neighbor も，受信したノードの Neighbor に加える．

図 5.4 に Name-Dropper アルゴリズムを示す．(i) は Query を送信する前の状態である．また，図中の実線矢印はそのノードが保持している Neighbor リストを示す．従って，図中の (i) では，ノード A は三つの Neighbor を，ノード B は二つの Neighbor を保持していることが分かる．

図中の (ii) は，(i) の状態からノード A がノード B へと Query を送信したあとの状態を示す．図中の点線矢印は Query を受信したノード B が新たに追加した Neighbor を示す．

この Name-Dropper アルゴリズムにより，SDP グループ内での通信経路が確保される．

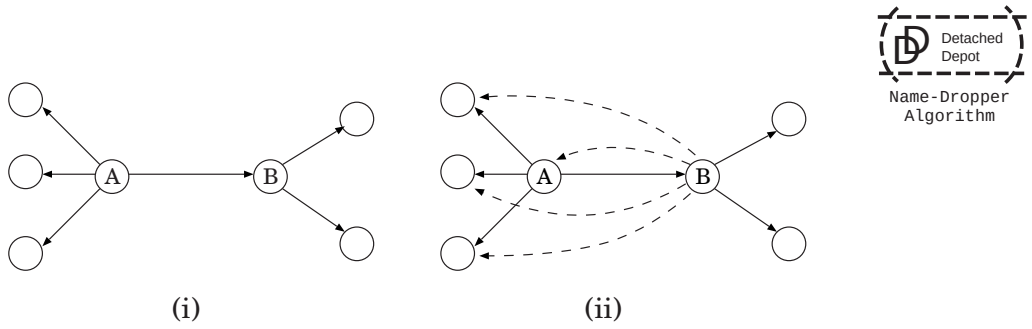

図 5.4: Name-Dropper アルゴリズム

5.3.5 Query 受信時の処理

SDP ノードが Query を受信した場合，Query のループを防ぐために，以下の二つの条件を検査する．

- ホップカウント
  ホップカウントを元に，古すぎる Query かどうかを検査する

- Query の重複
  シークエンス番号を元に，Query を重複して受信したかを検査する

これら二つの条件のどちらかに適合した Query はその場で破棄される．また，5.3.4 項で後述する Neighbor リストの更新が行われる．さらに，前項で述べた Name-Dropper アルゴリズムに従い，Neighbor リストを更新する．

その後の動作は Query の種類によって異なる．Query の転送は 5.3.3 項で述べたアルゴリズムに従う．

- Discovery Query
  Convert テーブルを検査し，Discovery Query に含まれる HID があれば，その HID に対応する IP アドレスを，同じく Query に含まれる送信元アドレスに直接送信する．なければ Discovery Query を転送する．

- Registration Query
  Convert テーブルを検査し，Registration Query に含まれている HID と IP の対を登録する．すでに HID が登録されていれば Query を破棄する．

- Neighbor Query
  Query に付与されている SDP ノードの IP アドレスを Neighbor リストに追加する．

Discovery Query を受信し，Convert テーブルに要求された HID が存在した場合，Reply を Discovery Query を送信したノードに直接返信する．これにより，検索にかかる時間が軽減される．

また，Registration Query を受信し，Convert テーブルにすでに HID が登録されていれば Query は破棄されるが，同一の IP アドレスが登録されていた場合は破棄しない．つまり，一つの IP アドレスを持つ単独のホストで複数の Personal Server を保持することは可能である．しかし，複数の Personal Server が同一ホストにある場合は Personal Server に保持されている実空間情報が漏洩しやすいので推奨されない．

Registration Query あるいは Neighbor Query を受信した場合，Convert テーブルあるいは Neighbor リストに登録が行われる．この時，Convert テーブルあるいは Neighbor リストが満杯になった場合，もっとも使用されていないエントリが削除される．

また，Convert テーブルおよび Neighbor リストの大きさは，その SDP ノードが有するメモリ容量などの機能に依存する．その SDP ノードが低機能であり，多量のエントリを保持できない場合でも，Query 転送により，SDP は SDP グループ全体では動作可能である．従って，低機能ノードへの対応も可能である．

以上述べた Query 受信時の動作に関する擬似コードを図 5.5 に示す．

```
server_Discovery_Main(query, from_Addr){
  // from_Addr は Query を送信したノードのアドレス.

  if(hop_Count_Check(query->hop_Count) == False)
    drop_Query(query);    // 古すぎる Query は破棄する.
  if(duplicate_Check(query->sequence_no) == False)
    drop_Query(query);    // 同じ Query は破棄する.

  do_Name_Dropper(from_Addr); // Name-Dropper アルゴリズムを行う.

  if(size_of_Neighbor_List == MAX_NEIGHBOR_SIZE)
    delete_Last_Neighbor_List();  // Neighbor リストがいっぱいなら,
                                  // 最も使用していない neighbor を削除する.

  swich(query->type){
    case DISCOVERY_QUERY: {
      if(cache_Include(query->hashed_ID) == True){
        send_Reply(get_IP_Addr(query->hashed_ID), query->src_Addr)
           // HID が Convert テーブル内にあれば送信元のノードに直接返信する.
      } else {
        query->hop_Count--;
        send_Discovery_Query(query,neighbor_List);
          // HID が Convert テーブル内になければ Query を近隣ノードに転送する.
      }
      break;
    } // end of DISCOVERY_QUERY
    case REGIST_QUERY: {
      if(cache_Include(query->hashed_ID) == True){
        drop_Query(query); // すでに登録されていれば Query を破棄する.
        if(size_of_Cache_List == MAX_CACHE_SIZE)
          delete_Last_Used_Cache(); // Convert テーブルがいっぱいなら
                                    // 最も使用していない cache を削除する.
      } else
        add_Cache(query->hashed_ID,query->IP_addr); // cache に登録する.
      query->hop_Count--;
      break;
    } // end of REGIST_QUERY
    case NEIGHBOR_QUERY: {
      add_Neighbor(query->neighbors); // Neighbor リストに追加する.
    } // end of NEIGHBOR_QUERY
  }
} // end of server_Discovery_Main
```

図 5.5: SDP における Query 受信時動作に関する疑似コード

5.4 数百人規模での実証実験

2002 年 9 月に，WIDE 合宿にて Personal Server Model に関する実証実験を行った．この実験は，本システムを利用して，ユーザの実空間情報を取得し，ネットワーク上へ提供することが可能であることを実証した．また，本実験ではユーザの実空間情報の中でも特に位置情報に焦点をあてた．

WIDE 合宿は，Hotel を借りて WIDE プロジェクト [21] のメンバーが集まり，BoF などが行われる．2002 年 9 月の本実験は 3 泊 4 日行われ，274 人が参加した．

図 5.6 に，実証実験が行われたホテルの地図を示す．また，“ BOF1” などのように，部屋ごとに位置情報のラベルをつけた．実験参加者の位置情報は，このラベルを利用して表現される．このラベル設定は手動で行った．また，図中の白丸は，後述する RF-Spider の基地局を設置した場所を示している．

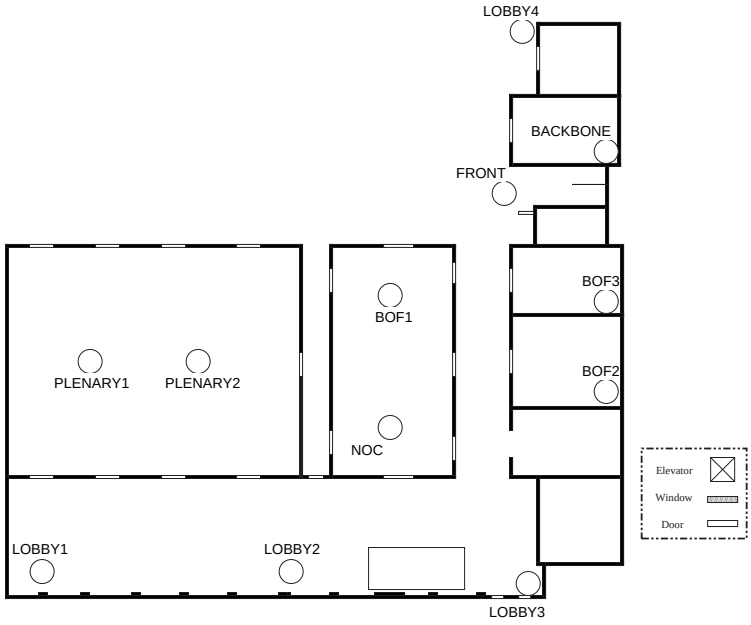

図 5.6: ホテル全体図

また，後述する LIN6 での場所検知のために，各部屋ごとにセグメントを分け，アドレスが部屋を示すようになっている．

なお，この実証実験では，各個人に対して一つづつの Personal Server を用意することが困難だったため，一つのホストに対して複数の IPv6 アドレスをエイリアスとしてつけ，仮想的に全参加者の Personal Server を用意した．

5.4.1 実証実験での全体構成

実証実験では，D2 システムに含まれる機能として，4 章で前述した Personal Server，PML，SDP に加えて，アプリケーションとしてユーザからの要求に答える web インタフェースを用意した．

図 5.7 に実験での全体概要を示す．この図では，下部に位置する多数のセンサデバイスから，Personal Server にセンサ情報が送信されていることが分かる．また，Personal Server に対してユーザが web インタフェースを通じてアクセスしている．センサデバイスの詳細については，5.4.2 項で後述する．

また，本研究で想定しているユーザが保持する GUID には，後述する RF-ID タグに振られている，GUID を利用した．

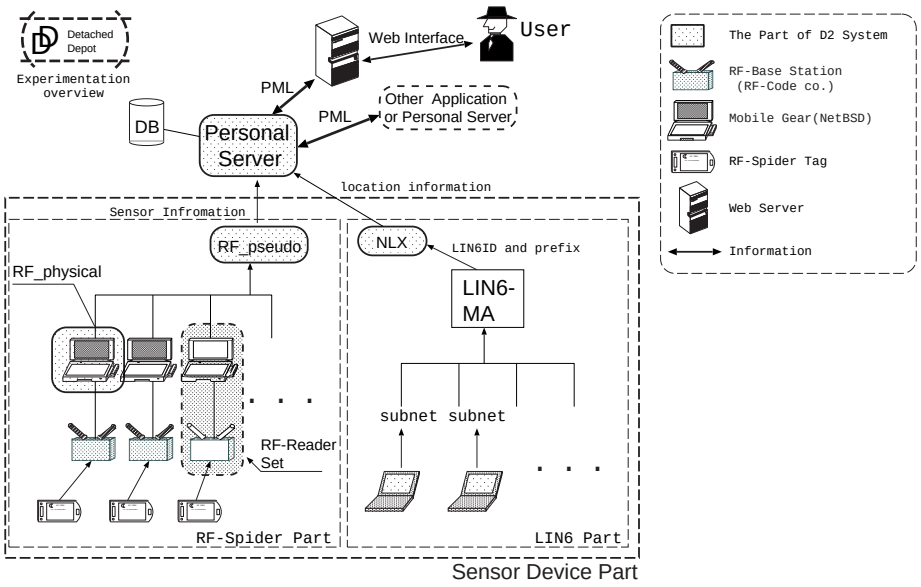

図 5.7: 全体構成概要

5.4.2 センサデバイス

本実験では人の位置情報を取得するために以下の二種類のセンサデバイスを使用した.

- RF-Spider
- LIN6

RF-Spider

本実験では位置を検出するためのシステムとして RF-Code[22] 社の Spider(RF-Spider) を利用した.

RF-Spider を選択した理由として，センサ敷設のコストがある．Active Bat[4] などのシステムは，非常に高い精度を持つが，敷設に大きなコストがかかる．また，常設実験環境ではなく，臨時実験環境であるため，設置および撤去が容易な必要がある．また，300 人弱という規模の実験であるため，各ユーザに配布するセンサデバイスのコストも重要となる．従って，個々人に高い端末が必要な自律型センサは不適当である．本実験では，位置は部屋単位で分かればよく，精度は重要ではない．そのため，RF-Spider という環境設置型センサを設置し，検知対象センサデバイスとして安価なタグを参加者全員に常時携帯してもらった．図 5.8 にそのタグを示す.

この RF-ID タグは電池を内蔵した無線タグであり，それぞれが固有の ID をもち，300MHz 帯の電波として発信している．電波の発信間隔は約 3 秒であり，この電波を RF-Spider の基地局が補足することにより，参加者がその基地局の検出範囲内にいることを検知する．図 5.9 に RF-Spider の基地局を示す.

また，我々は基地局につけるアンテナを三種類準備した．このアンテナはそれぞれ感度が異なるため，部屋の大きさに応じて使い分けた．これにより，一つの基地局が一つの部屋にほぼ相当する．また，広い部屋は複数個のリーダを設置した．

RF リーダセットは RF-Spider 基地局自体とネットワークを利用するためのモバイルギア [23] で構成される．RF-Spider 基地局およびモバイルギアは，電池駆動が可能であり，無線 LAN を使用することでわずかな時間であれば完全に無線で移動することができる．

モバイルギアと RF-Spider 基地局は RS-232C で接続されており，RF-Spider 基地局が検出したタグ ID などをモバイルギアが持つネットワークを経由して，位置情報サーバに送信される．その際，モバイルギアにつけられている，位置のラベルが付加されて，送信される．この動作をモバイルギア上で行うのが，“RF_physical” プロセスである．RF_physical から ID およびそのユーザの位置情報を受け取った位置情報サーバは，そのユーザの Personal Server へ位置情報を送信する．その様子は図 5.7 の左下に示されている．また，本システムでは，この位置情報サーバは “RF_pseudo” と呼ばれる．

また，モバイルギアは NetBSD/hpcmips で動作している．WindowsCE ではなく，NetBSD を使用することにより信頼性が増し，同時に遠隔管理が容易になった．

表 5.2 に RF リーダセットの仕様を記す．

表 5.2: RF リーダセットの仕様

本体	Mobile Gear II MC-R500
CPU	MIPS VR4111
OS	NetBSD-1.6(hpcmips)
メモリ	Compact Flash 256MBytes
RF-Spider 基地局	RF-Code SpiderIII

図 5.8: RF-Code タグ (たばこ箱は比較対象用)

RF-Spider を使用する場合に一番問題になることは，二つの RF-Spider 基地局が存在した場合に，その間に人が入った場合に，どちらを優先するかという問題である．合宿で利用したホテルは壁が薄く，電波が簡単に通り抜けてしまう．そのため，基地局が部屋をま

図 5.9: RF-Spider 基地局 (たばこ箱は比較対象用)

たいで電波を受信してしまう．つまり，人は移動していないのに，移動したように見えてしまうことがある．

しかし，合宿では，多くの BoF や発表が行われる．つまり，人の振る舞いとしては，一度部屋に入ったらしばらく (1 時間程度) はその場所を離れないことを前提とできる．本実験においてはこの事項を考慮した位置判断アルゴリズムを採用した．このアルゴリズムによって，不必要な人の移動の検知を減らすことができる．

このアルゴリズムは，位置情報サーバである，RF_pseudo で動作する．RF_pseudo は，RF_physical から受信した情報をもとに，その ID を持つユーザの場所を判断する．複数の RF_physical から同時にセンサ情報が取得した場合などの判断は，RF_pseudo が行う．

この実証実験では，位置を取得するためのもう一つのセンサとして LIN6 を利用した．

LIN6-MA

LIN6[24] では，各ホストに対して，LIN6ID という一意の ID が割り振られる．LIN6 では，ホストに prefix が割り当てられると，LIN6ID と prefix が合成され，IP アドレスが自動的に生成される．この IP アドレスを，MA(Mapping Agent) に登録することにより，ホストがどのネットワークに存在するかを通信相手に通知し，位置透過性を実現する．

参加者がノート PC を各部屋のネットワークに接続すると，LIN6ID と prefix が自動的に合成され，IP アドレスが生成される．この IP アドレスが LIN6ID とともに MA に送られる．IP アドレスを受け取った MA は NLX(Network Location eXchanger) と呼ばれるデーモンに IP アドレスおよび LIN6ID を送信する．NLX は LIN6ID によって，そのノート PC の持ち主が，prefix によって，そのネットワークが判別可能である．前述のように，本実験では各部屋ごとに異なる prefix を持つネットワークが構築されている．そのため，この

prefix およびネットワークを把握することでノート PC の位置を知ることが可能である．

5.4.3 PML

本実験では，人の位置情報に焦点をあてた．そのため，PML の Core 部に含まれる位置情報を主に利用した．しかし，それだけではなく PML の拡張データセットとして，以下の情報を利用した．これらの情報は，Personal Server から PML 形式で取得することができる．

- 名前
- ニックネーム
- 組織名
- 連絡先 (E-mail アドレス)
- WIDE 番号 (WIDE プロジェクトに参加している人に割り当てられる一意な番号)
- 宿泊部屋番号
- ユーザが現在使用しているノート PC の IP アドレス

5.4.4 提供したサービス

本実証実験では，Personal Server に問い合わせて位置情報を取得するための web インタフェースを実証実験参加者に対して用意した．web インタフェースは Personal Server に対して情報を要求し，PML を取得し，それを HTML へと変換する．また，web インタフェースは複数の Personal Server から情報を収集し，ある部屋ごとの人数などの統計情報などを提供している．

実験参加者は，この web インタフェースを通じて，以下のことができる．

- 位置情報の検索
  さまざまな条件で，希望するの利用者の位置を検索する．GUID，名前，所属組織などによって位置情報を検索できる．図 5.10 にそのスクリーンショットを示す．
- 位置からの情報検索
  位置を指定することにより，その付近に存在している利用者のリストを得ることができる．また，部屋の状況や参加者の集まり具合を遠隔からも知ることができる．図 5.11 にそのスクリーンショットを示す．この図の中では，部屋の名前をクリックすることにより，その部屋に存在している利用者のリストが表示される．

図 5.10: Web インタフェース：位置情報の検索

- ノート PC の追跡
  各参加者のノート PC に割り振られている LIN6ID を基にノート PC が置かれている場所を特定することが可能である．

- Personal Server IP アドレスの取得

  web インタフェースを通じて SDP を行うことが可能である．このサービスを利用することで Personal Server の IP アドレスをユーザが取得し，ユーザ自身が Personal Server を指定し，直接 Personal Server から情報を取得することも可能である．図 5.12 にそのスクリーンショットを示す．

5.4.5 ライブラリ

また，Personal Server にアクセスするための C 言語のライブラリを用意した．このライブラリは SDP を使用して Personal Server の IP アドレスを知る関数および Personal Server に接続して PML 形式で実空間情報を取得する関数を含む．また，SDP をコマンドラインから使用するプログラムも作成し，web インタフェースを用意した．これらのライブラリやインタフェースを使用することにより，提供した web インタフェース以外にも，合宿参加者自身が位置情報を利用するアプリケーションを容易に記述することができる．

付録 A にそのライブラリの詳細を示す．

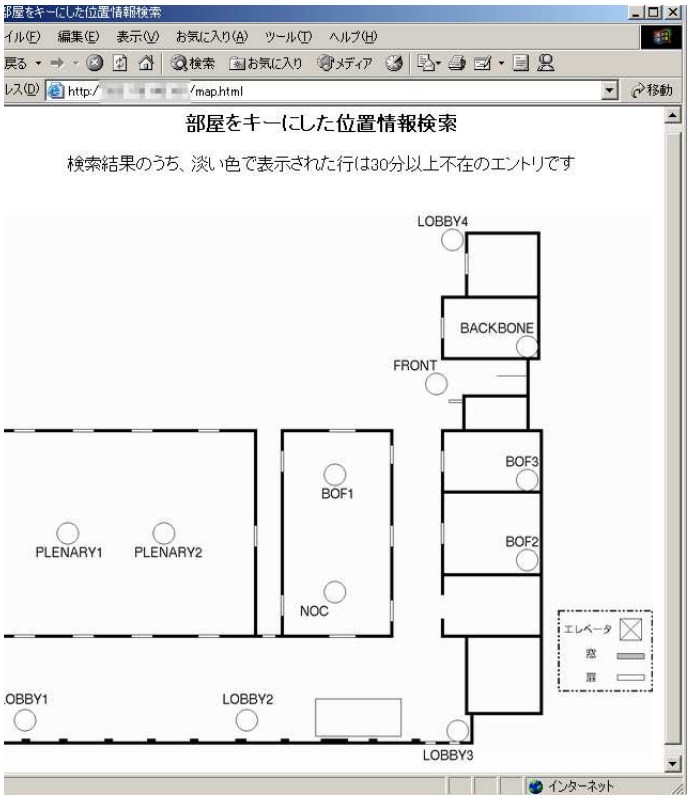

図 5.11: Web インタフェース：位置からの情報検索

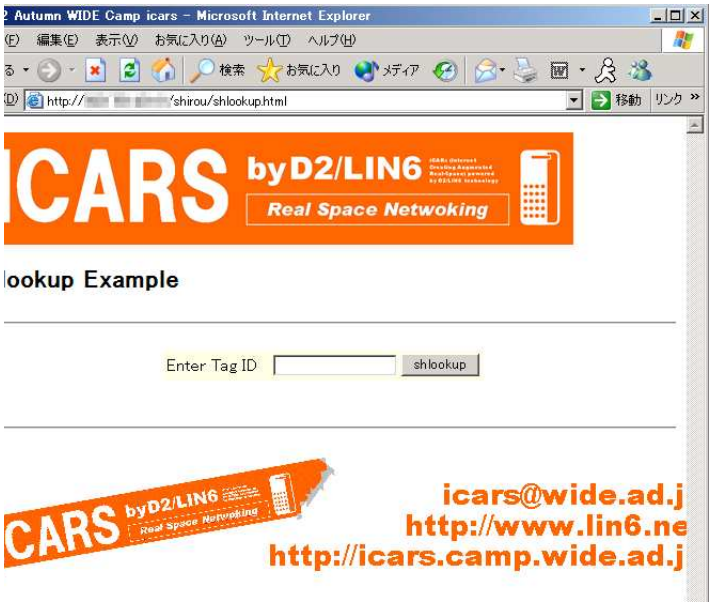

図 5.12: Web インタフェース：Personal Server IP アドレスの取得

第6章　Personal Server Modelの評価

本章では，Personal Server Model を定性および定量の両側面から評価する．また，WIDE 合宿での実証実験での結果も評価する．

6.1　定性的評価

本節では，本研究が提案する Personal Server Model の定性的評価を行う．

6.1.1　関連研究との比較

本節では，本研究が提案する Personal Server Model と 2.2 節で述べた関連研究を比較する．また，2.1 節で述べた問題点に主眼を起き，比較を行う．

前述した表 2.1 および表 2.1 に本研究が提案する Personal Server Model を追加し，比較を行った結果を表 6.1 および 6.2 に示す．

表 6.1: 定性的評価

関連研究 / 項目	Active Bat	TEA	GUIDE
センサの種類	単一種類	複数	複数
センサインフラストラクチャの形態	イントラ型	イントラ型	イントラ型
ユーザの移動	想定外	想定外	想定外
プライバシ保護	なし	なし	なし

表 6.2: 定性的評価 (続き)

関連研究 / 項目	PARCTAB	GLI	本研究
センサの種類	単一種類	単一種類	複数種類
センサインフラストラクチャの形態	グローバル型	イントラ型	グローバル型
ユーザの移動問題	想定外	解決可能	解決可能
プライバシ保護	なし	あり	あり

本研究では，単一のセンサデバイス，あるいはセンサインフラストラクチャのみを想定してはいない．多数のセンサインフラストラクチャ同士が情報を交換し合うことによって形成されるグローバル型センサインフラストラクチャを対象範囲としている．

また，本研究はグローバル型センサインフラストラクチャで生じるユーザの移動問題を指摘し，それに対する解決手法である SDP を提案している．加えて，プライバシ保護の問題点を指摘し，それに対するアクセス管理機構の必要性を指摘している．

これらの比較の結果，本研究が提案している Personal Server Model は，他の関連研究と比較し，実空間ネットワーク環境及び，グローバル型センサインフラストラクチャを実現することが可能であると言える．

6.1.2 Personal Server Model におけるプライバシ保護

本節では，Personal Server Model におけるプライバシ保護に対する評価を行う．

図 6.1 に実空間情報の流れを示す．この図では，Personal Server Model として，実空間情報が取得されてから他のアプリケーションが取得するまでの間，実空間情報を保護していることが分かる．

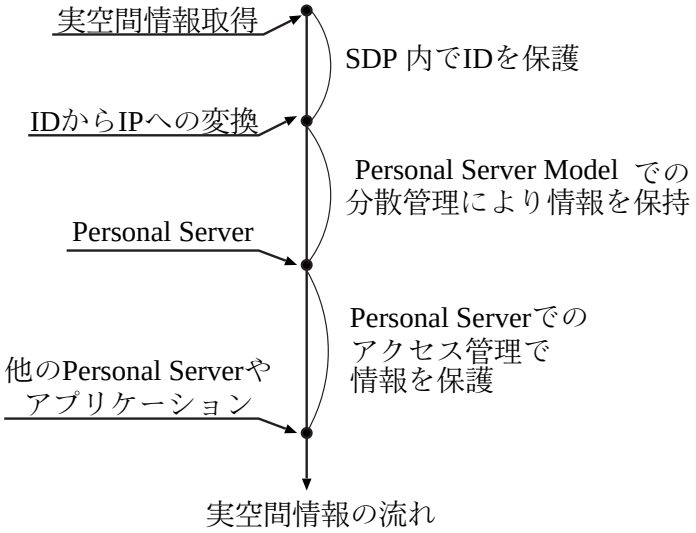

図 6.1: 実空間情報の流れおよび Personal Server Model での保護

この図により，Personal Server Model では，プライバシ保護が 3.4 節で述べられている範囲内で確立されていることが分かる．

6.2 数百人規模での実証実験の結果

本節では，5.4 節で述べた WIDE 合宿での実証実験を通じて得られた評価について述べる．

Personal Server Model では，各ユーザの情報は各 Personal Server ごとに分散している．そのため，各 Personal Server に対して要求を行うことで統計データの取得を行った．また，各 Personal Server ごとに存在するアクセス管理機構では，統計データ収集用 GUID からの要求は通過するように設定した．

6.2.1 人の動き

定量的な評価として，実際の移動とセンサを通じて取得した移動との比較を行った．図 6.2 は 24 時間 (睡眠中はセンサ検知範囲から退去したため，実質 18 時間) での実際の位置とセンサによって検出された位置を示す.

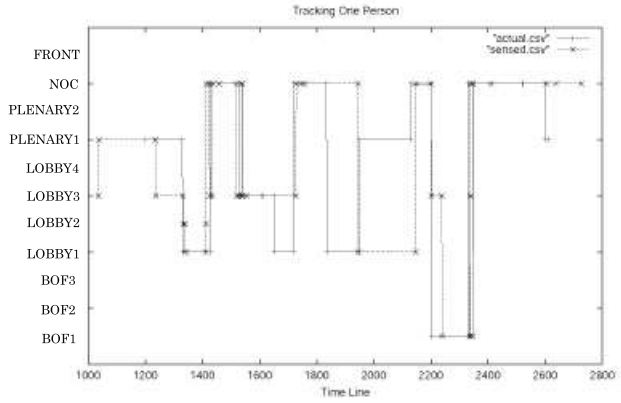

図 6.2: 実際の移動履歴と検知された移動履歴

この図から，人の大まかな場所は判別したと言えるが，まだまだ課題は多い．しかし，精度の問題は，主にセンサデバイス自身の精度に起因するものであり，センサ技術の向上により解決していくだろう.

6.2.2 部屋ごとの人の動き

人の動きが特徴的だった部屋を選択し，以下に示す．図 6.3 は BOF2 の，図 6.4 は LOBBY3 を示している．これらの図は，その部屋の人数を時間軸で表した図である．BOF2 は BOF が開始された 11:30 ごろから人が検出され始めている．また，夜中はまったく検出されていない．しかし，通路である LOBBY3 では，早朝を除き，常に検出されている.

また，両者ともに右肩下がりが繰り返されている傾向が見受けられる．このことより，BoF の開始・終了時刻とともにみなが移動し，また別の部屋へと移動するという移動パターンを繰り返している可能性が指摘できる.

6.2.3 全体的な人の動き

表 6.3 は，“どこから”，“どこへ” 移動したかを示した表である．このデータにより LOBBY と PLENARY 間の人の移動が活発であることが分かる．図 5.6 で示したホテル地図を見ると，PLENARY と LOBBY は隣り合わせであり，PLENARY からは必ず LOBBY を通る必要がある．そのため，PLENARY と LOBBY 間の移動が多く検出されたと思われる.

このように，図 5.6 で示した地図を組み合わせることにより，ロケーションモデルの提

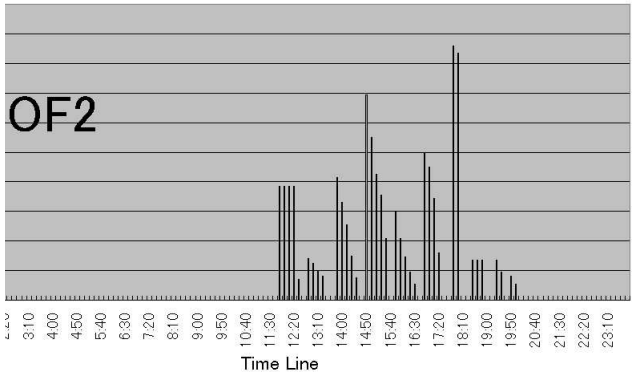

図 6.3: BOF2 における時間軸と人数とのグラフ

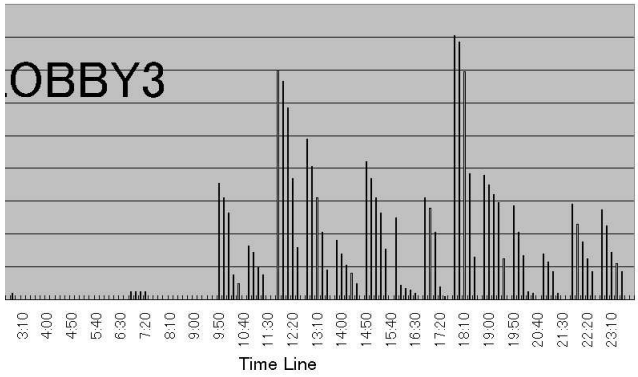

図 6.4: LOBBY3 における時間軸と人数とのグラフ

案が容易に行える可能性を示すことができた．また，今回の実証実験のように短期間ではなく，長期的に運用することで，さらに効率的な位置検出手法が使えるようになる．

ToLo[25] は，このようなデータを利用し，位置情報取得センサ間の関連性を自動生成し，センサインフラストラクチャ管理者に提案する．ToLo を利用することにより，計算機器に関する知識を持たないユーザにも容易にセンサインフラストラクチャの管理をすることができる．

表 6.3: ユーザの移動グラフ

From \ To	B1	B2	B3	P1	P2	L1	L2	L3	L4	NOC	FR	Total
BOF1	-	24	0	14	20	11	28	54	0	18	6	175
BOF2	18	-	12	24	28	24	37	51	0	13	6	213
BOF3	1	13	-	1	1	1	5	9	0	0	7	38
PLENARY1	36	32	6	-	188	83	134	173	4	43	21	720
PLENARY2	20	29	2	160	-	47	115	157	2	43	23	598
LOBBY1	11	28	1	46	61	-	129	61	0	26	10	373
LOBBY2	23	39	4	76	117	133	-	102	2	34	14	544
LOBBY3	55	65	11	78	153	59	108	-	5	113	47	694
LOBBY4	1	0	0	0	5	3	1	3	-	0	1	14
NOC	18	19	0	22	41	24	33	104	0	-	5	266
FRONT	9	5	2	10	25	9	13	46	0	5	-	126
Total	192	254	38	431	639	394	603	760	13	295	141	3760

表 6.4: ユーザの移動グラフの注釈

BOF1	B1	BOF2	B2	BOF3	B3		
PLENARY1	P1	PLENARY	P2				
LOBBY1	L1	LOBBY2	L2	LOBBY3	L3	LOBBY4	L4
NOC	NOC	FRONT	FR				

第7章　結論

7.1　まとめ

本研究では，センサ情報を個人単位で集中管理するモデルである，Personal Server Model を提案した．

本研究では，センサデバイスを利用することにより，実空間におけるさまざまなものを，コンピュータを埋め込むことなくコンピュータネットワーク上から情報を得る環境を実空間ネットワーク環境とした．この環境では，従来コンピュータネットワーク上からは扱えなかったものまで扱うことにより，ユーザに対してさらなる利便性を提供することができる．

実空間ネットワーク環境を実現するためには，センサデバイスが必須である．しかし，単一のセンサデバイスで全ての用途を網羅することはコストや性能などの面で不可能である．従って，単体のイントラ型センサインフラストラクチャだけではなく，これらのセンサインフラストラクチャがお互いに情報を交換し合うグローバル型センサインフラストラクチャが必要となる．その中で，ユーザは移動しながら，近隣に存在するセンサインフラストラクチャを利用する，という形態になる．本研究では，このような環境を想定環境としている．

このような環境では，ユーザの移動に応じてユーザの情報を取得するセンサインフラストラクチャが変化していく．従って，そのユーザのセンサ情報を他のアプリケーションが利用する時に，どのセンサインフラストラクチャでセンサ情報が管理されているのかが分からないという問題が生じる．

また，センサ情報は極めて個人的なものであり，保護されなければならない情報である．しかし，想定環境では，ユーザが使用しているセンサは，そのセンサインフラストラクチャの管理人が所有し管理している．従って，その管理人はそのセンサインフラストラクチャをユーザが使用している限り，自由にその情報を取得することが可能となってしまう．また，同一センサインフラストラクチャを複数人が共有するということがある．そのため，センサインフラストラクチャの管理人や他のユーザ個人情報であるセンサ情報が漏洩する可能性が高いという問題点が生じる．

本研究が提案している Personal Server Model では，そのユーザに関するセンサ情報を含んだ実空間情報を Personal Server が全て管理する．これにより，ユーザの移動によって生じる問題点を解決した．また，Personal Server 内にはアクセス管理機構が存在し，GUID に基づいて公開する情報を管理する．本研究では，このアクセス管理機構によってプライバシ保護の問題を解決した．また，センサインフラストラクチャの管理人には GUID しか判別できないことにより，匿名化に基づくプライバシ保護を実現した．

さらに，本研究ではPersonal Server内に蓄積されている，そのユーザに関するセンサ情報を含む実空間情報を利用するためのデータ交換言語であるPMLを規定した．この交換性および拡張性に富むPMLを利用することにより，他のアプリケーションはそのユーザの実空間情報を利用することが可能となる．

加えて，Personal Server Modelによって生じた，ユーザの情報を管理しているサーバの位置を取得する必要性を解決するための，SDP(Server Discovery Protocol)を設計した．このSDPはプライバシ保護にも考慮しつつ，GUIDからPersonal Serverのネットワーク上の位置へと変換することで，ユーザの移動に関する問題点およびPersonal Server Modelに起因するPersonal Serverの場所が不明になるという問題点を解決した．

本論文では評価として，Personal Server Modelに基づく実証実験を行い，その結果を記した．これにより，Personal Server Modelがユーザの実空間における位置情報を把握可能なことを示すことができた．

本論文により，実空間ネットワーク環境およびグローバル型センサインフラストラクチャを実現する際に生じる問題点を示すと同時に，その解決策を提示した．また，今後普及していくであろう実空間ネットワーク環境に対して，今後の発展につながるために必要なセンサ情報管理モデルを提案し，プロトタイプ実装を提示することができた．

7.2 今後の課題

本節では，残された今後の課題について述べる．

7.2.1 Personal Serverの移動

本研究では，Personal Serverは常にネットワーク接続性を維持し，消失しないものとしている．しかし，匿名性を維持し，プライバシを保護するためには，Personal Serverのネットワーク上の位置を変更するということが必要となる．

ネットワーク上の位置が変更されることにより，以前まで使用可能であったIPアドレスが使用不可能になることで，Personal Serverの匿名性が強化される．IPアドレスの変更後は，GUIDおよびSDPを利用することでのみ新しいIPアドレスを取得することができる．SDP内部では他者のGUIDを取得することは不可能なため，結果としてGUIDを知らない第三者はIPアドレスが分からない．

このように，Personal Serverの移動に対応することでより一層のプライバシ保護が図れる．Personal Serverの移動に対応するために，Mobile-IP[26]などが重要となる．

7.2.2 アプリケーション

Personal Server Modelによって実現される，実空間ネットワーク環境を利用することによりさまざまなアプリケーションが考えられる．

自動日記アプリケーション: 本人の行動履歴を元に，その日一日で何をしたか，誰と会ったかなどを記録するアプリケーション

スケジュール調整アプリケーション: 各人の Personal Server に問いかけてスケジュールの調整を自動的に行うアプリケーション

これらのアプリケーションを充実させることにより，実空間ネットワーク環境および Personal Server Model の有用性をより強調することが可能となる．

7.2.3 対象の拡張

本研究では，センサ情報を管理する対象を個人に絞っている．一方，実空間コンピューティング環境では，実空間における情報の対象を個人のみを対象としていない．例えば，“デルタ 1F 会議室における温度情報およびその場にいる人数” などの情報は極めて有用であると考えられる．

本研究が提案する Personal Server Model が管理する対象を，空間や物体などに拡張することは可能である．だが，そのためには以下の問題点を解決する必要がある．

- 対象の拡張
- 管理者問題
- 規模性

対象の拡張

本研究は，人間である個人を対象としている．その理由は，人間は一意な存在であり，分離，結合，重複などが起こらず，また，実空間において最も必要とされる情報を保持しているからである．

しかし，物体に関しては，複数の物体が結合し，一つの物体に統合することが容易に想定される．また，空間に関しては，同一の空間でも，“SFC” とするか，“デルタ” とするかなど，粒度によって管理する対象が変化する．

このように，対象を物体や空間に拡張した場合には，一つの対象に対してもさまざまな捉え方が存在し，複雑である．

管理者

本研究が提案する Personal Server Model には，必ず所有者，管理者が一人だけ存在し，その所有者の Personal Server が情報を管理する．しかし，物体や空間の所有，あるいは管理者は，実体が存在しない組織，共有，あるいは，不在な場合が多々存在する．

従って，対象を物や空間に拡張する場合には，所有者が複数あるいは存在しない場合などに対応するための別の機構が必要となる．

規模性

センサ情報を管理する対象を物体や空間に拡張した場合，本研究が想定している数に比べて数倍，数十倍の Personal Server を扱う必要性が生じる．従って，その際に生じる規模性の問題を解決する必要がある．

7.2.4 アクセス管理機構の拡張

現在の Personal Server におけるアクセス管理は，GUID を元にアクセス権限を決定し，公開する情報を決定している．しかし，[27] で述べられているように，“この時には，この人に教えてもよい” などアクセス管理機構自体に実空間情報を利用することが考えられる．

アクセス管理に実空間情報を利用することにより，粒度の細かい，よりユーザに対して簡便性を提供可能な実空間ネットワーク環境を構築することができるだろう．

謝辞

本研究の機会を与えて下さり，絶えず丁寧な御指導を賜りました，慶應義塾大学環境情報学部教授 徳田英幸博士，同学部教授 村井純博士に感謝致します．また，懇切丁寧な御助言を頂きました，同学部助教授 楠本博之博士，同学部助教授 中村修博士，同学部専任講師 教授 南政樹，奈良先端科学技術大学院大学教授 砂原秀樹博士に感謝の意を表します．また，本論文作成にあたり適切な御指摘および御指導を頂いた奈良先端科学技術大学院大学助手 羽田久一氏に深く感謝致します．

また，日頃より研究活動の御指導を頂きました，慶應義塾大学特別専任講師 植原啓介氏，慶應義塾大学 SFC 研究所研究員 渡辺恭人氏，慶應義塾大学政策・メディア研究科湧川隆次氏，川喜田佑介氏をはじめとする InternetCAR 研究グループのみなさまに感謝します．

また，権藤俊一氏をはじめとする MoVE!研究グループのみなさまに感謝します．

また，共に苦闘し，励まし，叱咤し合った，青木崇行氏，間博人氏，石井かおり氏，石原知洋氏，梅染充男氏，小川浩司氏，海崎良氏，桐原幸彦氏，菅沢延彦氏，鈴木雅子氏田原裕市郎氏，豊野剛氏，西村祐貴氏，松宮健太氏，三川荘子氏，村瀬正名氏，をはじめとする仲間たちに感謝します．

また，冬の寒い日々に何日も待っていてくれた HONDA SPADA および SUZUKI Alt に感謝し，本論文を提出後に整備を行うことを誓います．

また，執筆中の緊張感を時に高め，時にやすらぎをを与えてくれた菅野よう子氏および坂本真綾氏に感謝します．

また，忙しい執筆活動を影から支えてくれた Age Of Mythology と FLASH および 2 ちゃんねるに多大なる感謝と後悔の念を捧げます．

最後に，これまで筆者と出会い，言葉を交わしあったすべての人に感謝します．

以上をもって，謝辞といたします．

参考文献

[1] IEEE802.11WorkingGroup. Ieee 802.11 the working group for wireless lans, 1997. http://www.ieee802.org/11/.

[2] Jason I. Hong and James A. Landay. An Infrastructure Approach to Context-Aware Computing. *HCI*, 16:287–303, 2001.

[3] Mike Spreitzer and Marvin Theimer. Providing location information in a ubiquitous computing environment. In *In Proceedings of the 18th ACM Symposium on Operating Systems Principles(SOSP'93)*, pages 270–283, Ashville, NC, 1993.

[4] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The anatomy of a context-aware application. *The Fifth Annual ACM/IEEE International Conference on Mobile Conputing add Networking*, 2(1):59–68, 1999.

[5] P Steggles, P Webster, and A Harter. The implementation of a distributed framework to support 'follow me' applications. In *Proceedings of the 1998 International Conference on Parallel and Distributed Processing Technique and Applications(PDPTÁ98)*, pages 1381–1388, LasVeas NV, July 1998.

[6] A. Schmidt, K. Aidoo, A. Takaluoma, U. Tuomela, K. van Laerhoven, and W. van de Velde. Advanced interaction in context, 1999.

[7] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Experiences of developing and deploying a context-aware tourist guide: The guide project. In *Proceedings of MobiCom 2000*, pages 20–31, Boston, August 2000.

[8] Roy Want, Bill Schilit, Norman Adams, Rich Gold, Karin Petersen, John Ellis, David Goldberg, and Mark Weiser. The PARCTAB ubiquitous computing experiment. Technical Report CSL-95-1, Xerox Palo Alto Research Center, March 1995.

[9] Yasuhito watanabe, Atsushi shionozaki, Fumio Teraoka, and Jun Murai. The desibh and implementation of the geographical location information system. In *INET'96*, 1996.

[10] AutoIDCenter, 1999. http://www.autoidcenter.org.

[11] ミューチップ, 2002. http://www.hitachi.co.jp/Prod/mu-chip/jp/.

[12] S. Kent, BBN Corp, and R. Atkinson. Security architecture for the internet protocol, November 1998.

[13] D. Harkins and D. Harkins. The internet key exchange (ike), November 1998.

[14] Paolo Bouquet Massimo Benerecetti and Matteo Bonifacio. Distributed Context-Aware Systems. *HCI*, 16:213–228, 2001.

[15] Henning Maa. Location-aware mobile applications based on directory services. In *Proceedings of the third annual ACM/IEEE international conference on Mobile computing and networking*, pages 23–33. ACM Press, 1997.

[16] World Wide Web Consortium. Extensible markup language(xml) 1.0, February 1999.

[17] Ora Lassila. Resource description framework (rdf) model and syntax specification, 1999. W3C Recommendationhttp://www.w3.org/TR/REC-rdf-syntax¿.

[18] Internet CAR Project. Internet car project, 1998. http://www.sfc.wide.ad.jp/InternetCAR.

[19] PostgreSQL, 2003. http://www.postgresql.org/.

[20] Mor Harchol-Balter, Frank Thomson Leighton, and Daniel Lewin. Resource discovery in distributed networks. In *Symposium on Principles of Distributed Computing*, pages 229–237, 1999.

[21] WIDE-Project, 1988. http://www.wide.ad.jp.

[22] RF-CODE, 2002. http://www.rfcode.com/.

[23] Mobile Gear MC/R550 NEC Corporation, 1998. http://www.nec.co.jp/press/en/9803/1101.html.

[24] Mitsunobu Kunishi, Masahiro Ishiyama, Keisuke Uehara, Hiroaki Esaki, and Fumio Teraoka. Lin6: A new approach to mobility support in ipv6. In *International Symposium on Wireless Personal Multimedia Communication*, 2000.

[25] 村上朝一, 中西健一, 桐原幸彦, and 徳田英幸. ロケーションモデル作成支援機構の構築. In **日本ソフトウェア学会ソフトウェアシステム研究会** *SPA* **サマーワークショップ** *SPA-SUMMER 2002* **論文集**, 8 2002.

[26] C.Perkins. Mobility support in ipv6. Internet-Drafts(draft-ietf-mobileip-ipv6-19.txt).

[27] U. Leonhardt and J. Magee. Security considerations for a distributed location service, 1998.

付 録A　提供したC言語ライブラリ

概要	int shlookup(char * tagid,char *PS_addr);
引数	tagID, 代入先
返り値	成功ならば 0, 失敗したならば-1
説明	ID を元に，その ID を持つ Personal Server のアドレスを PS_addr に入れる.

概要	struct PS * d2_init (char *hostname);
引数	接続するホスト名
返り値	PS 構造体へのポインタ
説明	PS 構造体を作成する.

概要	int d2_connect_PS (struct PS *PS);
引数	PS 構造体
返り値	成功ならば 0, 失敗したならば-1
説明	PS に接続する.

概要	int d2_close_PS (struct PS *ps);
引数	PS 構造体
返り値	成功ならば 0, 失敗したならば-1
説明	PS への接続を切断する.

概要	char * d2_get_xml_all(struct PS *ps);
引数	PS 構造体
返り値	PML で記述された実空間情報に対する char のポインタ
説明	PS から PML で記述された実空間情報を取得する.

概要	char * d2_get_location_from_xml(char *xml);
引数	PML で記述された実空間情報に対する char のポインタ
返り値	char へのポインタ
説明	PML を解析し，最後に取得された位置情報を返す.

概要	char * d2_get_timestamp_from_xml(char *xml);
引数	PML で記述された実空間情報に対する char のポインタ
返り値	char へのポインタ
説明	PML を解析し，最後に位置情報が取得されたタイムスタンプを返す.