修士論文要旨　　2003年度(平成15年度)

センサ遍在環境における
ユーザの周辺情報の収集に関する研究

論文要旨

近年，ユビキタスコンピューティング環境の浸透に伴い，現実世界での状況を取得しコンピューティングに応用するためにセンサの利用が注目されている．また，センサ情報を利用したアプリケーション実現のために，センサの管理やセンサ情報の提供を統合的に行うセンサ環境が多数構築されている．センサ情報提供環境内を移動するユーザの周辺のセンサ情報を収集することにより，ユーザの体験の記録や，記憶の補助となるアプリケーションが構築可能である．しかし，さまざまな環境を移動するユーザに対して，同一のアプリケーションでセンサ情報の要求を行うことは非常に困難である．

本研究では，多様なセンサ情報提供環境からユーザの周辺のセンサ情報の収集を実現する上での要件について考察し，アプリケーションのセンサ情報利用モデルの分析を行った．これをもとにセンサ情報収集フレームワークのプロトタイプシステムである MemoryDirectory の設計と実装を行った．MemoryDirectory はアプリケーションに対して単一のセンサ情報要求方法を提供し，センサ情報の収集に際しては，センサ情報提供環境ごとに異なる要求記述に変換することによって，アプリケーションからのセンサ情報提供非依存性を実現する．また，センサ情報環境ごとに異なるセンサ情報の時間や空間的粒度を統一的に扱う方法をアプリケーションに提供することにより，より柔軟なセンサ情報の要求を実現するアプリケーションを少ない労力と時間で構築できる．

キーワード：
1 ユビキタスコンピューティング環境　2 センサ遍在環境　3 センサ情報収集
4 記憶の補助　5 ユーザセントリック

慶應義塾大学大学院　政策・メディア研究科
岩谷 晶子

Abstract of Master Thesis

Academic Year 2003

A User-centric Approach for Personal Information Retrieval in Ubiquitous Sensing Environment

Summary

Recently, utilization of sensors in a ubiquitous computing field has received attention because of sensor's nature which can capture user's activities and local circumstances. It is suitable for ubiquitous computing to use sensors. Various sensor environments have appeared and applications which use sensor information help user's activities. One of the applications is collecting sensor information of user's circumstanses and logging the data for later use. Logged data includes time, location and content of users activities. However, sensor information retrieval from variouse sensor environments are difficult because each sensor environment implements their own method for sensor information collection and sensor naming. In addition, sensor applications most of sensor environment provides only primitive methods for sensor information collection. These difficulties make construction of sensor applications more complex.

To address this isssue, we examine existing sensor environments and specify requirements for information retrieval from heterogeneous sensor environment. Then we examine sensor information usage models of applications. We designed and implemented a prototype framework for sensor information retrieval, named MemoryDirectory. MemoryDirectory provides applications a unified method to query sensor information, and realizes sensor environment independent data collection by translating the queries according to different sensor environments. It also provides applications with methods to unify time/space granularity of sensor data, and simplifies construction of applications which require flexible sensor data collection.

Keywords:

1 Ubiquitous Computing 2 Heterogeneous Sensor Environment
3 Sensor Information Retrieval 4 memory-aid application 5 user-centric

Keio University Graduate School of Media and Governance
Akiko Iwaya

修士論文　　2003年度 (平成15年度)

センサ遍在環境における
ユーザの周辺情報の収集に関する研究

慶應義塾大学大学院　政策・メディア研究科
岩谷　晶子

目 次

図 目 次

表 目 次

第1章　序論

本章では，本研究の意義および本論文の構成について述べる．

1.1 背景

ユビキタスコンピューティング環境

近年，コンピュータや入出力デバイスが環境に遍在し，ユーザに対して多様なサービスを提供するユビキタスコンピューティング環境 [27] の浸透が著しい．ユビキタスコンピューティング環境においては，PC や携帯電話だけではなく，自動車や家電機器，机，指輪などさまざまなものが計算処理能力やメモリ，ネットワーク接続機能などを持ち，多様なサービスを実現する．遍在するコンピュータやデバイスをユーザが意識すること無く利用できるだけでなく，従来コンピュータ上で扱われていた情報に加えて，センサなどを通じて取得した現実世界の情報を用いて，ユーザのさまざまな日常的な行動に対して利便性と効率性を提供する．ユビキタスコンピューティング環境におけるサービスの例として，ユーザの位置情報に応じてナビゲーションを行うサービスや，ユーザの近傍に存在するプリンタなどのリソースを提示するサービスなどが挙げられる．ユビキタスコンピューティング環境を現実世界で実現，利用する事例も増加している．一例として経済産業省の e!プロジェクト [28] の一環として実現された六本木アカデミーヒルズ内にある六本木ライブラリにおける，無線タグによる未来型ライブラリサービス [30] が挙げられる．

センシング技術の発達

近年のセンシング技術の発達により，小型で高性能なセンサが数多く登場している．カメラや，温度計，タッチセンサ，赤外線，無線タグなど，センサの種類も多様でさまざまな利用方法が考えられている (図 1.1)．ユビキタスコンピューティング環境においては，現実世界での状況を取得し，それに応じたアプリケーションの挙動を実現するためセンサは必要不可欠である．例えば，先に説明した未来型ライブラリでは，探している本が今実際に置かれているエリアや書架などの情報を検索することが可能である．これは，本に取り付けられたタグが発信する無線電波をセンサが取得することで，本の位置を取得しているためである．センサに加えて，計算処理能力と通信機能を備えた小型のセンシングコンピュータによって，局地的にセンサネットワークを構築し，情報を収集する技術も登場している [14, 4]．センサネットワークは，センシングコンピュータを置くだけで動的にネットワークの構築とセンサ情報の取得が可能であるため，敷設が容易で低コストな技術として注目を浴びている．

ユビキタスコンピューティング環境におけるセンサアプリケーション

ユビキタスコンピューティング環境において多様なセンサを用いることが可能になったため，新しいセンサアプリケーションが登場している．新しいセンサアプリケーションではセンサ情報を人間の記憶をサポートする情報として利用する．例を挙げると，一日の行動履歴 [5, 23]，ある特定の行動における記憶 [12]，つい先程の行動 [25] などである．センサアプリケーションは，動画，静止画，音声あるいは温度などのセンサ情報を取得し，人

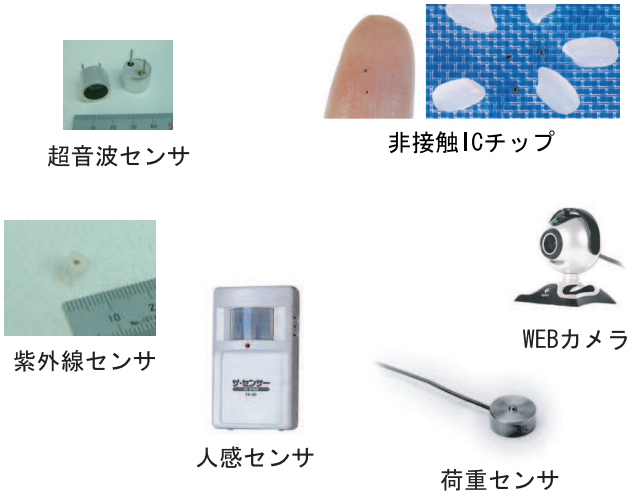


図 1.1: センサ

間にとって必要なタイミングでこれらの情報を提供することにより，ユーザの記憶を想起する手伝いをすることを目的としている．

移動するユーザの周辺情報を取得し，アプリケーションに提供する場合にも，この二種類の利用方法が考えられるが，本研究では後者の利用を特に対象とする．ユーザがどこにいたか，何を見たか，そのとき周辺はどういう状況であったか，というユーザ自身が経験したことをセンサ情報として取得・蓄積し，ユーザ自身がその経験を思い出すためにセンサ情報を参照するというのが本研究で対象とするアプリケーションの目的である．

センサ情報を利用してアプリケーションを実現する環境が増え，ささいなセンサ情報でも蓄積が可能になっているが，ユーザの周辺のセンサ情報を蓄積するアプリケーションの実現は困難である．

その理由の一つは，センサ情報を提供する環境の多くは特定のアプリケーションのみへのセンサ情報の提供を目的としていることである．その環境内でのアプリケーションの実現のみを想定しているため，環境ごとに異なるセンサ情報の提供ポリシやプロトコルを採用しており，ユーザの周辺情報を取得・蓄積するといった環境透過的なアプリケーションの実現が困難なのである．最近では，IrisNet[7] のように，組織横断的にセンサ情報の提供と取得を実現するためのアーキテクチャの提案もなされているが，こういったアーキテクチャも含めて，センサ情報の提供方法を透過的に実現するための機構が必要である．

もう一つの理由は，センサ情報を蓄積する方法にある．上記の問題を解決できた場合を仮定すると，ユーザが移動した先に存在するセンサ情報はすべて取得・蓄積が可能になる．その場合に，アプリケーションが要求するセンサ情報の量や質があるとすると，それを満たす情報の収集はどのように実現すべきであるか，またアプリケーションはあらかじめ，その場所に存在するセンサ情報を知らなければ，センサ情報の要求をできないのか，という問題が起こる．必要な時間，空間などの項目をアプリケーション側で指定しないで，すべてのセンサ情報を蓄積する場合，非常に膨大で参照が困難なものになってしまう．これを解決するためには，アプリケーションごとにセンサ情報の質や量などを指定してセンサ

情報を取得する条件を限定するための機構が必要である．

1.2 本研究の目的

本研究では，センサ情報を提供する基盤環境に透過的にユーザの周辺のセンサ情報の取得と蓄積をするアプリケーションのための，センサ情報収集システムの構築を目的とする．この目的を達成する要件として，以下の二つを挙げる．

- 多様なセンサ情報提供環境へのセンサ情報の要求機構の実現
- ユーザの周辺のセンサ情報を利用するアプリケーションのためのアプリケーションインタフェースの提供

1.3 本論文の構成

本論文は，以下のように構成する．

まず第 2 章では，既存のセンサ情報提供環境を紹介し，本研究で対象とするセンサ情報提供環境の特徴を述べる．また，アプリケーションがセンサ情報提供基盤環境からセンサ情報を収集する上で発生する問題について述べる．次に第 3 章では，第 2 章でアプリケーションセンサ情報を利用する際の特徴や要求について整理，センサ情報収集フレームワークに求められる要件を述べる．第 4 章では，第 3 章で述べた要件に基づいて，本研究でのアプローチを述べる．第 5 章にてセンサ情報収集フレームワークについて，概要と設計を述べる．第 5 章では，センサ情報収集フレームワークのプロトタイプの実装環境および，実装について詳細に述べる．第 6 章では，プロトタイプ環境において，取得されたセンサ情報の評価や，環境透過性の実現レベルに関する比較評価を行う．第 7 章では，関連研究としてセンサ情報の取得・蓄積を目的としたシステムやアプリケーションの解説と，本研究で実現した機構との比較を行う．最後に第 8 章では，今後の課題と結論を示し本論文をまとめる．

第2章　センサ情報提供環境

本章では，本研究で想定環境とするセンサ情報提供環境の説明をし，既存のセンサ情報提供環境を実現するアーキテクチャについて述べる．このような環境を横断的に移動するユーザの周辺情報を収集するアプリケーションの構築における問題点を挙げる．

2.1 センサ情報提供環境

本論分ではある特定の空間に遍在するセンサを管理し，各センサが取得したセンサ情報を蓄積し，センサ情報を提供する機構を備えた基盤を持つ環境を **センサ情報提供基盤環境** と呼ぶ．センサ情報提供環境の要件として以下の項目が挙げられる．

複数のアプリケーションへのセンサ情報の提供 単一のアプリケーションによってセンサが専有されないセンサ情報提供モデルを実現すること

センサの管理 センサ情報提供環境内に存在するセンサの種類や個数，各センサ情報を蓄積したデータベースへのポインタなどを把握し，かつそれらの情報をアプリケーションなど要求者に提供可能なソフトウェアが存在すること

センサの位置の把握 センサ情報提供環境で利用するロケーションモデルに基づいたセンサの位置を把握していること

pull 型の情報提供 センサ情報の提供は，センサ情報の要求者からの要求を受けて送信する pull 型のコミュニケーションであること

以降では，これらの基本要件を満たした既存のセンサ情報提供環境について述べる．

2.1.1 統合センサ環境

本論文では，センサシステムと詳細なロケーションシステムを統合的に備え，特に屋内でのアプリケーションの利用を想定したシステムを統合センサ環境と呼ぶ．本項では，統合センサ環境として，Active Badge System[8] と Active Bat System[9] について述べる．

Active Badge System

Active Badge System[8] は赤外線を発信する Active Badge **??**を人やモノなどに備えることによって，屋内における人やモノの位置情報を取得する．人の位置や状態に応じて挙動を変えるコンテクストアウェアアプリケーションの実現を目的としているため，より正確で詳細な位置情報や，正確な環境認識が可能である．しかし，このアプローチは敷設にコストがかかるため容易に導入できないことが最大の欠点である．

Active Badge System のデモンストレーションシステムでは位置情報システムへのアプリケーションインタフェースとして図 2.1 に示すコマンドが用意されている [26].

センサ情報の記述においては CORBA オブジェクトを利用しており，アプリケーションに対して CORBA オブジェクトによるセンサ情報の提供を行う．

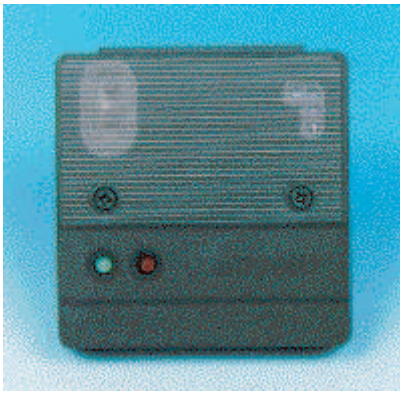


図 2.1: Active Badge

表 2.1: Active Badge System のアプリケーションインタフェース

コマンド	説明
FIND(name)	name の badge が存在する場所を通知
WITH(name)	name の badge が存在する場所およびその場所に存在する他の badge の情報を通知
LOOK(location)	location の近傍に存在する badge の情報を提供
NOTIFY(name)	name が 次に Active Badge System で観測された時に通知
HISTORY(name)	name の位置を 1 時間周期で報告

Active Bat System

Active Bat は Active Badge と同様に AT&T 研究所によって提案された屋内の位置情報システムである．Active Badge との相違点として位置認識に利用されるタグ??が超音波センサを利用している点が挙げられる．超音波センサによる位置認識により cm 単位での位置認識が可能である．

2.1.2 ワイヤレスセンサネットワーク

ワイヤレスセンサネットワーク [4] では，環境に配されたセンサ同士で，アドホック無線ネットワークを構築し，複数の超小型センサノード [14, 20] を経由してセンサ情報の取得を実現する．ワイヤレスセンサネットワークでは，基本的にセンサノードの性能が貧弱であるため，画像や音声などのリッチなセンサ情報を扱うのが難しい．また，バッテリ駆動であるため，電力消費を抑えた動作を実現する必要がある．

センサネットワークでは，アプリケーションに対してセンサ情報を提供するために，センサプロキシあるいはシンクノードを設けるのが一般的である．アプリケーションが個々のセンサノードに対して要求をするのはコストが高いためである．センサプロキシを用い

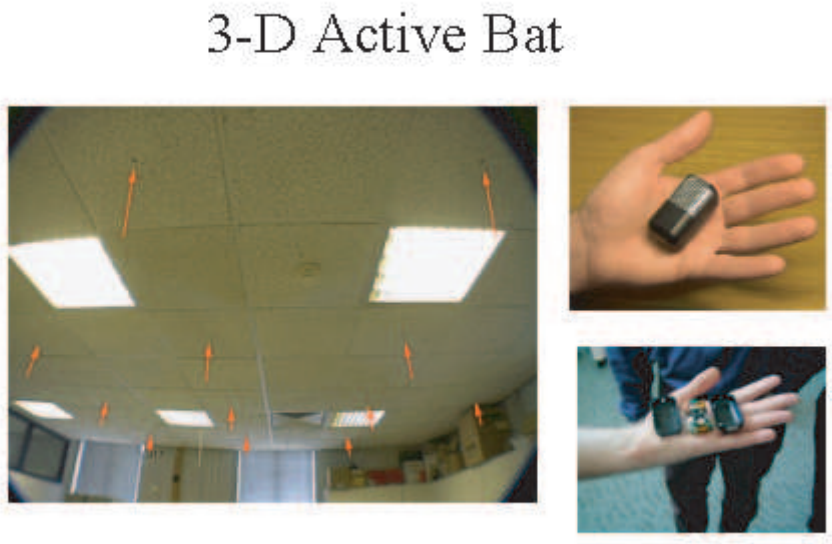


図 2.2: Active Bat

たセンサ情報要求方法を図 2.3 に示す[1].

センサネットワークに特化したセンサ情報クエリアーキテクチャとして，Fjords アーキテクチャ[18] が提案されている．Fjords では SQL 言語を拡張したクエリ方法を採用している．

ワイヤレスセンサネットワークを実現するアーキテクチャとして，Peer to Peer の無線通信 [15] によって近接するセンサのみから情報を取得するアプローチが存在する．このアプローチは，デバイスを携帯していれば，その場でセンサ情報を取得できるという点で，利用が容易である．欠点は，ワイヤレスセンサネットワークと同様で性能が貧弱なデバイスを用いて狭い無線帯域を利用しているため，画像や音声などのリッチなセンサ情報を送受信するのが困難である．

2.1.3 オープンセンサアーキテクチャ

オープンセンサアーキテクチャとは，WWW と同様に多数のセンササーバが，多数のクライアントからの要求を受けてセンサ情報を提供するモデルを実現したアーキテクチャのことである 2.4．このモデルを実現することにより，上記の二手法に比べて格段にスケーラブルなセンサ情報の取得が実現可能となる．センサの提供者が誰であるかを知らずにアプリケーションがセンサ情報の要求を可能である．現在研究が進められているプロジェク

[1]図 2.3 は [18] から引用している．

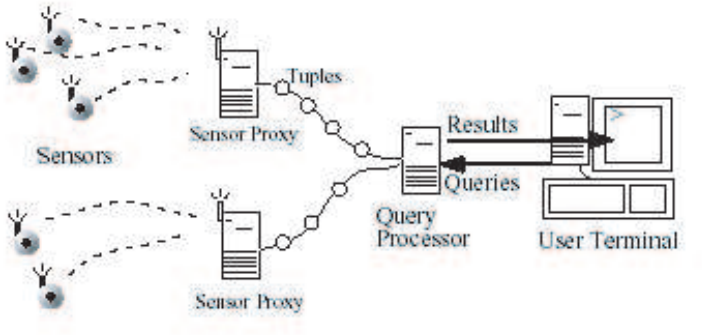


図 2.3: センサプロキシへの要求

トとして，Intel Research 社の IrisNet[7] や Open GIS Consortium による Sensor Web[2] が挙げられる．

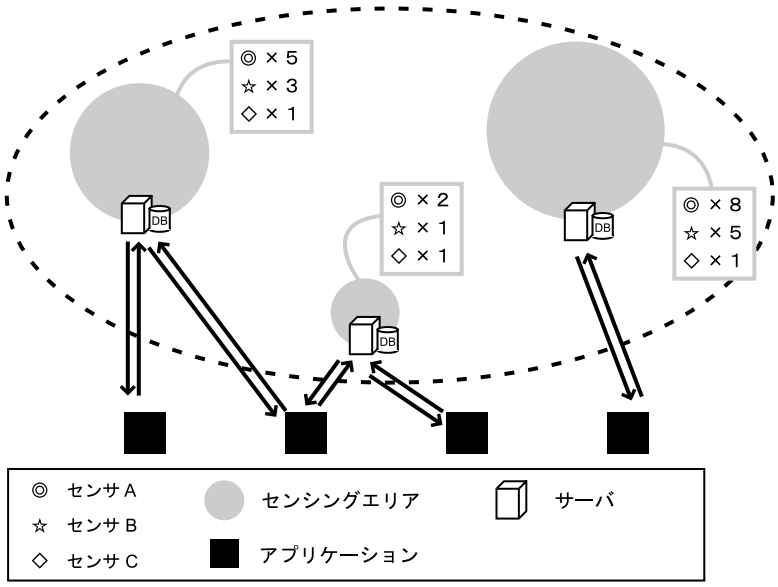


図 2.4: オープンセンサアーキテクチャの概念図

IrisNet ではアプリケーションからのセンサ情報を要求するために， XML によるデータ表現と XPATH によるクエリを提供している [3]．図 2.5 では，ニューヨークの Soho と Tribeca における駐車場の空きを尋ねるクエリである． 図 2.6 は Soho にある駐車場の空き状況を表した XML データである．

```
/usRegion[@id= 'NE ']/state[@id= 'NY ']/city[@id= 'New York ']
        /neighborhood[@id= 'Soho ' OR @id= 'Tribeca ']
        /block[@id= '1 ']/parkingSpace[available= 'yes ']
```

図 2.5: IrisNet における XPATH によるクエリの例

2.2 センサ情報収集における問題

前節で挙げたように，センサ情報提供環境は多様なアーキテクチャが存在する．また，各システムごとに異なる要求方式やセンサ情報の記述方法が提供されている．各センサ情報提供環境内でのセンサ情報のみを利用するアプリケーションならば十分であるが，本研究で想定するアプリケーションがセンサ情報収集対象とするユーザは，複数のセンサ情報提供環境間を移動する．アプリケーションがセンサ情報提供環境に横断的なセンサ情報の収集を実現するためには，以下のような問題がある．

- センサ情報提供環境の差異によるアプリケーション構築時の負荷
- 柔軟なセンサ情報指定方法の欠如

以下に，それぞれについて詳しく説明する．

2.2.1 センサ情報提供環境の差異

第 2.1 節で紹介したセンサ情報提供環境の特徴を表 2.2 にまとめる．ここで述べる特徴については，特にアプリケーションがセンサ情報を要求および利用する際に必要となる情報について取り上げる．

センサ情報の要求記述，センサ情報の記述など，それぞれが独自の方式を用いている．また，環境側がセンサ情報を提供するポリシは環境ごとに異なる．このため，各環境に横断的なセンサ情報の収集を実現するアプリケーションを構築するためには，各環境に特有の情報要求方式やセンサの記述などをそれぞれ把握する必要がある．

具体的に，アプリケーションがセンサ情報を要求する上で把握しなければならない項目について以下に説明する．

センサ情報の要求記述

センサ情報を要求するための文法やプロトコルなどが考えられる．各センサ情報提供環境ごとに，SQL やオブジェクト呼出，XPATH など多彩である．

```
<usRegion @id= 'NE '>
    <state @id= 'NY '>
        <city @id= 'New York '>
            <neighborhood @id= 'Soho '>
                <block @id= '1 '>
                    <parkingSpace @id= '1 '>
                        <available>yes</available>
                    </parkingSpace>
                    <parkingSpace @id= '2 '>
                        <available>no</available>
                    </parkingSpace>
                </block>
            </neighborhood>
        </city>
    </state>
</usRegion>
```

図 2.6: IrisNet における XML によるセンサ情報の表現

センサ情報の記述

各センサ情報提供基盤環境が持っているセンサの種類や数量は異なるため，その環境に応じたセンサ情報の要求をする必要がある．そのために，センサ情報の記述についてを知る必要がある．たとえば，同じ動画を取得するセンサに対して環境 A では「ビデオカメラ」と，環境 B では「ビデオ」と名づけられている場合，アプリケーションはそれぞれの記述方法を知り，要求する必要がある．

位置の粒度

多くのセンサ情報提供環境内では，独自のロケーションモデルに基づいたセンサの位置の管理を行っている．特に統合センサ環境では，詳細な位置情報が取得可能である．いっぽうで，TinyOS などによって実現されるワイヤレスセンサネットワークでは，位置情報自体がアーキテクチャの必須要件となっていないため，実装するノードが持つ位置情報デバイスや実装などに依存する．また，IrisNet ではセンサ情報提供者が実装する senselet と呼ばれるプログラムが位置情報の管理の実現についてを決定するため，実装に依存する．

表 2.2: センサ情報提供環境の比較

センサ情報提供環境	センサ情報要求記述	センサ情報の記述	位置の粒度	適用場所
Active Badge System	独自コマンド	オブジェクトの位置	数 10cm	屋内
Active Bat System	CORBA オブジェクト呼出	CORBA オブジェクト	数 cm	屋内
TinyDB	TinySQL	light	実装に依存	主に屋外, 自然環境
IrisNet	XPATH	XML	senselet の実装に依存	同左

センサ情報の提供モデル

おおくのセンサ情報提供環境では，アプリケーションからの要求はセンサプロキシが対応することを想定している．この場合，アプリケーションはセンサ情報が欲しい時点でセンサプロキシに対して情報の要求を発行する Query-Result のモデルである．しかし，これ以外のモデルでのセンサ情報提供を行う環境もある．Fjords[18] では，同一のセンサ情報の要求を定期的に必要とするアプリケーションを想定し，センサ情報の要求を一度だけ行うことによりセンサ情報を定期的に提供する continuous query[1] と呼ばれる方式を採っている．このように，単純な Query-Result モデルではなく，登録して情報が取得され次第提供する Publish-Subscribe モデルを採用しているセンサ情報提供環境も存在する．アプリケーション開発時には，こういったセンサ情報提供環境ごとの差異についても考慮する必要がある．

2.3 前提条件

上記の問題を解決する上で，前提となる条件について議論する．

想定するアプリケーションはさまざまな環境を移動するユーザの周辺のセンサ情報を収集する．アプリケーションがセンサ情報基盤環境からセンサ情報を収集する段階までに，以下のような手続きがユーザ，センサ情報提供環境，アプリケーションの間で実行される必要がある．

1. ユーザの位置情報の取得
2. ユーザの位置情報に応じたセンサ情報基盤環境の決定
3. 決定したセンサ情報提供環境をアプリケーションへ通知

ユーザの位置情報の取得方法には，ユーザが携帯する GPS などのデバイスが位置情報を取得する positioning system と 環境側に存在する位置情報取得センサがユーザの認識

をする tracking system の二種類が存在する [17]．本研究では位置情報の取得方法については特に規定しない．ただし，取得された位置情報に応じて，センサ情報基盤環境を決定するために，ユーザの位置情報を定期的に監視するソフトウェアが存在することを前提とする．

位置に応じたセンサ情報提供環境は実際には複数存在しうる．だが，本論文では，問題を簡単にするために，取得されたユーザの位置情報からただ一つの対応するセンサ情報提供環境が存在することを前提とする．想定として，センサ情報提供には固有の ID があり，これによってセンサ情報提供環境を識別する．これを実現するアーキテクチャの一例として，位置情報とマッピングされたセンサ情報提供環境のリストがあり，そのリストを参照することによって，センサ情報提供環境におけるセンサ情報の提供元を特定する方法が考えられる．

ユーザの位置を監視するソフトウェアは，ユーザによって実行されるものであり，実行空間もユーザが提供するものと想定するのが自然であるため，ユーザの位置情報に応じたセンサ情報提供環境の決定とアプリケーションへの通知は，同ソフトウェアが行うことを前提とする．

2.4 本章のまとめ

本章では，想定環境としてセンサ情報提供環境の説明をし，既存のセンサ情報提供基盤環境を実現するアーキテクチャについて述べた．センサ情報提供環境に横断的にセンサ情報を収集する問題点として，センサ情報提供環境ごとに仕様が異なるためアプリケーションの構築が煩雑であること，またセンサ情報提供環境自体にセンサ情報を要求における柔軟性が少ないためアプリケーションの要求する情報を収集することが難しいことの二点を挙げた．

第3章　センサ情報とアプリケーション

本章では，前章で述べた問題点のうち，柔軟なセンサ情報の要求に注目し，アプリケーションが実際に要求するセンサ情報について分析する．この分析をもとに本研究が実現するべき機構に関する要件を挙げる．

3.1 センサ情報とアプリケーション

本節では，本論文で想定するセンサ情報とアプリケーションについて説明する．

センサ情報 特定の物理空間にバインドされ，なんらかの物理的変化を表す情報

センサとは，一般的な定義として，温度や光，磁気などの物理量や変化量を検出し，その検出量を適切な信号に変換して計測系に入力するデバイスを指す．すなわちセンサ情報とは，センサによって取得された情報，物理空間で発生していることや変化したことを表す情報である．ユビキタスコンピューティング環境においては，上記のセンサ以外にも，さまざまな情報を利用して物理空間の変化や状態を取得可能である．例えば，情報家電の操作履歴を取得してユーザの行動を取得することは，物理空間における状態の取得と考えられる．当然ながら，これらの情報はある特定の物理空間において発生するものであり，センサ情報と同時にその情報が生成した場所を取得できることが前提である．

センサ情報は常に生成され続けるため，アプリケーションにとって必要な情報以外を蓄積することは非常にコストがかかる．

センサアプリケーション センサ情報を取得・利用するアプリケーション

センサアプリケーションは，一般的にセンサ情報を取得・利用するアプリケーション全般を指す．本研究においては，取得したセンサ情報を蓄積することまでを目的としたアプリケーションを特に対象としているため，本論文ではこのようなアプリケーションをセンサアプリケーションまたは単にアプリケーションと記述する．

3.2 アプリケーションシナリオ

本研究では，ユーザが移動した先々に存在するネットワークに接続されたセンサから，ユーザに関連する情報を取得・利用するアプリケーションを想定する．このアプリケーションの目的は，ユーザがいつ，どこで，何をしたか？を思い出す手がかりとなる情報を収集し，提示することである．

具体的なアプリケーション例として二つのシナリオを挙げる．

アプリケーション A ミーティングや立ち話のディスカッションで話した内容や，授業で聞いた内容を思い出すためにセンサによって記録された情報を利用する

アプリケーション B 持ち歩いていたはず の携帯電話をどこにおいたのか，という情報を思い出すためにユーザの行動履歴をセンサ 情報から生成する

本節では，上述したアプリケーションの特徴を明確にし，そこから求められるセンサ情報提供フレームワークの要件について考察する．考察する項目としてアプリケーションがセンサ情報の取得対象とする空間と時間，アプリケーションの性質などの項目について考察する．各項目で，はじめに一般的な議論をし，次に本研究で対象とするアプリケーションやセンサ情報提供フレームワークの要件について論じる．

3.3 アプリケーションとセンサの関連性

アプリケーションがセンサを利用する場合，アプリケーションとセンサの疎親の度合を考慮する必要がある．例を挙げるとアプリケーションがセンサを専有して利用する場合，その関係性は親であり，センサ情報をサーバにおいて複数のアプリケーションから利用可能にする場合には疎である．アプリケーションとセンサの関連性が疎である場合，センサ情報を複数のアプリケーションへ提供するための機構が必要である．表 3.1 に関連性についてまとめる．

表 3.1: アプリケーションとセンサの関連性

関連性の度合	利用形態
親	単一のアプリケーションがセンサを専有
疎	複数のアプリケーションがセンサ情報を利用

本研究で想定する環境では，さまざまな場所でさまざまなセンサ情報を取得するため，専用のセンサではなく，複数のアプリケーションから利用可能な状態でセンサ情報を提供するべきである．したがって，センサ情報提供環境にはセンサ情報を複数のアプリケーションへ提供するための機構が必要となる．

また，センサ情報の提供者あるいは組織は，必ずしもアプリケーションの実行者であるユーザと同一であるとは限らず，WWW などのように情報の提供者と利用者は全く関連性の無い場合が想定される．そのため，センサ情報を保存ポリシについてアプリケーションは関知できない可能性がある．よって，アプリケーションはセンサ情報をユーザに提供するためには必要なセンサ情報を収集しユーザの管理するスペースに保存する必要がある．

3.4 要求するセンサ情報

センサアプリケーションは各々の目的に基づいて，センサ情報を要求する．センサの種類はすべてのセンサアプリケーションにとって必要である．また，センサがセンサ情報を取得するサンプリングレートや，センサアプリケーションが必要とする時間帯の情報であるかどうかを判断する適時性，さらに，同じカメラセンサの場合でも，画素数の高いセンサ情報が必要な場合，あるいは，より細かい粒度で位置情報を表現できるセンサ情報が必要な場合など，センサ情報自体の質が重要なアプリケーションも存在する．表 3.2 にセンサアプリケーションがセンサ情報に要求する基準をまとめる．

先に述べたアプリケーション A の場合，要求するセンサ情報として，音声や動画などが挙げられる．またアプリケーション B の場合には，自分が携帯していたものを置き忘れた可能性のある候補を絞り込むことができるように，行動履歴が保存されている必要がある．行動履歴となりうる情報として，位置情報の変化や時間，また一緒にいた人の情報

表 3.2: センサ情報の必要性を決定する基準

基準	例
センサの種類	温度，カメラ，気象，位置情報 など
センサ情報のサンプリングレート	300 ミリ秒毎，１秒毎，１時間毎 など
センサ情報の適時性	最新の情報，要求した厳密な時間の情報
センサ情報自体の質	カメラの画素数，位置情報の粒度 など

などがある．

しかしながら，ユーザの移動先にアプリケーションが要求するセンサが必ずしも存在するとは限らない．そのため，具体的なセンサを指定して要求するのではなく，抽象的な表現で必要となるセンサ情報の指定をするできることが望ましい．抽象的な表現によるセンサ情報の取得を実現するためには，アプリケーションで利用するセンサと名とセンサ情報提供環境側で利用しているセンサ名のマッピングを行う必要がある．

3.5 アプリケーションの興味空間

本稿では，アプリケーションが興味を持つ対象が含まれる空間を**興味空間**と呼ぶ．アプリケーションがユーザの周辺情報を取得するためには，ユーザを含む興味空間の広さを決定し，その空間に含まれるセンサからセンサ情報を取得する必要がある．以降では，アプリケーションの興味空間の性質について考察する．

3.5.1 興味空間の広さ

アプリケーションの興味対象が含まれる空間の広さを決定することにより，その空間に含まれるセンサが決定される．興味空間の広さの決定は，採用されているロケーションモデルにより複数の方法が考えられる．一つはアプリケーションの興味対象の位置を基準とした絶対的な距離 (半径) によって指定する方法である．図 3.1(a) では，ユーザを中心とした半径 r を興味空間として指定した様子を表している．絶対的な距離を用いて興味空間の広さとそこに含むセンサを決定する手法として，対象からの相対的な距離を用いて空間に含むセンサを決定する方法と，各センサの位置と対象の位置を同一の座標系にマッピングし，そこから距離を取得して空間に含むセンサを決定する方法が考えられる．

興味空間の広さを決定するもう一つの方法が対象の現在位置をある広さを持つ意味的な空間に含め，その空間によって指定する方法である．この場合，興味空間の広さは，意味的な空間の広さに依存する．これを実現するためには，意味的な空間の定義と，センサが属する空間の設定があらかじめ必要である．図 3.1(b) では，アプリケーションの興味対象であるユーザが学校に滞在しており，アプリケーションの興味空間は学校全体であることを示している．図中では学校において取得できるセンサ情報は全部で三つであり，した

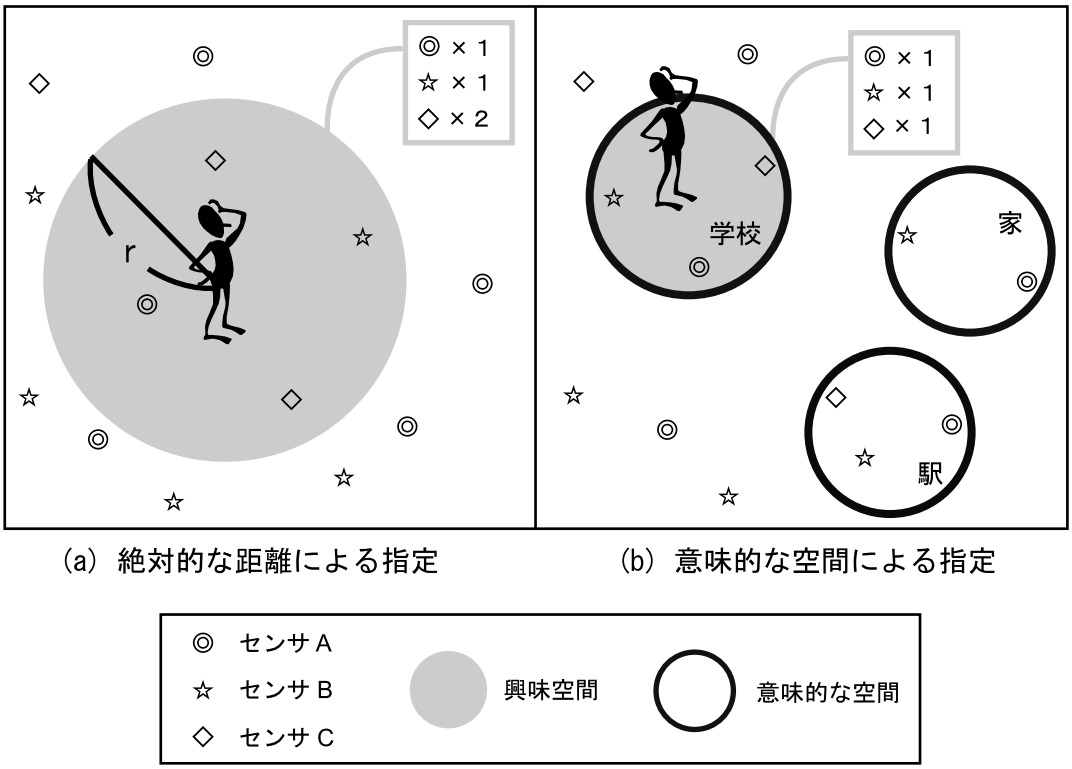


図 3.1: 興味空間の広さの指定

がって，アプリケーションの興味空間に含まれるセンサはその三つであることを表している．意味的な空間は多くの場合，階層的な構造で管理されており，同じ一つの空間でも，研究室，SFC，遠藤，藤沢市，神奈川県というように複数の名前を持つ．階層的に管理される場合，一般的に階層が上にあるほど，より広い空間である．どの階層での名前を利用するかについてはアプリケーションの要求や位置情報を取得するデバイスに依存する．

アプリケーションが興味空間の広さを指定する場合，ロケーションモデルに応じて，対象の現在位置からの距離を指定するか，意味的な空間の階層を指定する必要がある．

3.5.2 興味空間に含まれるセンサ

あるセンサアプリケーション A がセンサ情報を取得できるセンサが設置されており，かつ A の興味空間となりうる全空間を S と定義する．また，単一のセンサが情報取得対象とする範囲を s と表現する．このとき S は n 個のセンサを含む空間で $S = \{s_1, s_2, \ldots, s_n\}$ のように表現する．ある瞬間 t におけるセンサアプリケーション A の興味空間 S_I は興味空間の種類に応じて，表 3.3 で示す (a), (b), (c) のいずれかに分類できる．図 3.2 の (a), (b)，(c) は表 3.3 に対応する．単一のアプリケーションの興味対象が複数ある場合，一個以上のセンサを含む複数の空間が必要となるが，ここには記していない．

	空間が含むセンサ	表現
(a)	無し	$S_I = \phi$
(b)	一個以上のセンサを含む空間	$S_I = \exists s \in S$
(c)	全体	$S_I = S$

表 3.3: 興味空間に含まれるセンサの個数による分類

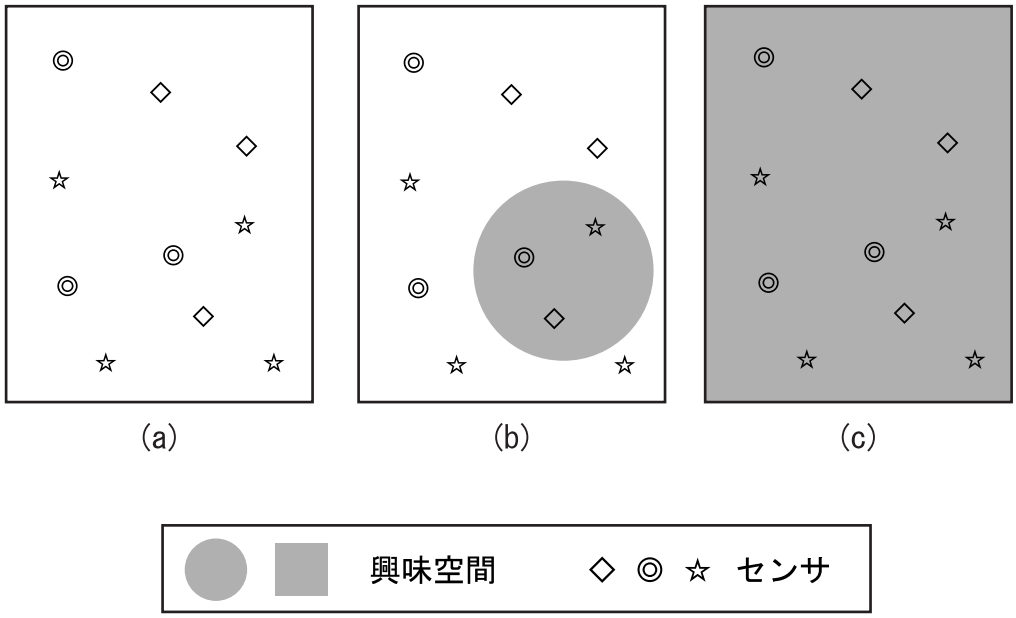


図 3.2: 興味空間に含まれるセンサ

3.5.3 興味空間の変化の有無

アプリケーションの興味対象が移動する場合には，対象の移動にあわせて興味空間も移動する．それにともない，興味空間に含まれるセンサもまた変化する．図 3.3 ではユーザの移動にともなう興味空間の変化と，興味空間に含まれるセンサの種類や数の変化を表している．

本研究で想定するアプリケーションの対象となる興味空間は，常にある特定のユーザを含む空間なので，ユーザが移動して存在する場所が変わると，興味空間に含まれるセンサが変化し，アプリケーションに提供すべきセンサ情報も変化する．

ユーザの場所を記録することで取得するセンサ情報の変化を知らせるトリガとなるため，ユーザの移動履歴およびそこに存在するセンサの種類を取得できる．

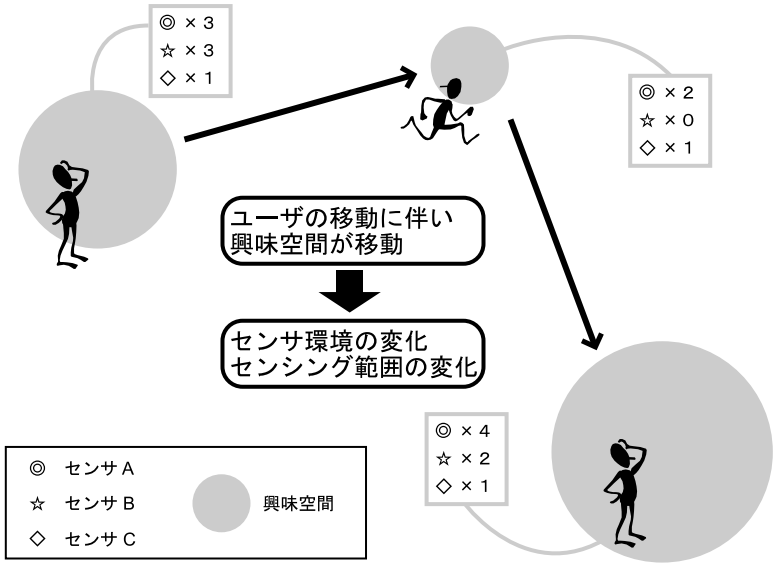


図 3.3: 興味空間の変化

3.6 センサ情報の興味時間

アプリケーションがセンサ情報を利用する場合，ある瞬間のセンサ情報のみの場合や，ある期間の複数のセンサ情報というように時間軸においても多様な要求が考えられる．

センサ s で取得できるセンサ情報を R とするとき，ある瞬間 t に取得されたセンサ情報を R_t とあらわす．このとき t はアプリケーションがセンサ情報を必要とする時間あるいは期間であり，本稿では**興味時間**と呼ぶ．興味時間は大きく表 3.4 のように分類可能である．また，図 3.4 では各時間表現のモデルを表している．

表 3.4: 興味時間の分類

図 3.4 中での記号	説明
(a)	$t = now$
(b)	$t = start$ から end
(c)	$t = l_m$から $l_n (m < n)$

現在のセンサ情報を必要とする場合 (表 3.4(a))，センサの取得開始時から終了時までのセンサ情報を必要とする場合 (表 3.4(b))，ある特定の期間のセンサ情報を必要とする場合 (表 3.4(c))) に分類した．特に，3.4(c)) については，具体的に以下に示すような期間が考えられる．

- あるイベント E が発生してから一定時間
- $start$ から now

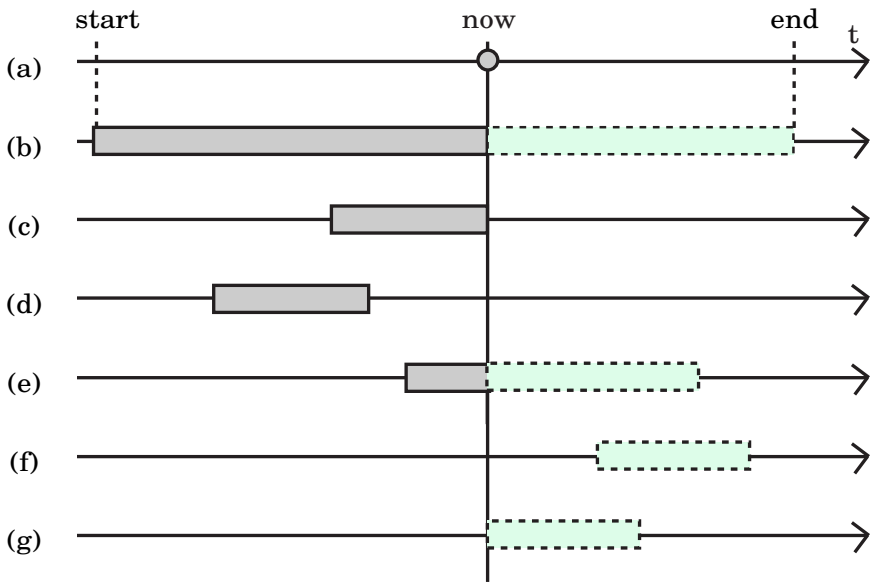


図 3.4: 興味時間のモデル

- あらかじめ決められた区間 (毎日朝 6:00 から 6:30 など．)
- $t = now - n$ から now

本研究で想定するアプリケーションでは，ユーザが興味空間に存在する間は常に興味のある時間となる．

3.6.1 センサ情報の興味時間あたりの密度

興味時間内にアプリケーションがセンサ情報を必要とする頻度をセンサ情報の興味時間あたりの密度と呼ぶ．興味時間あたりの密度が高いほどアプリケーションが利用可能なセンサ情報が増加し，アプリケーションが利用できるセンサ情報に対する選択の幅が増加するが，アプリケーション側でより多くのストレージを必要とする．アプリケーションに応じてセンサ情報の利用目的は異なるため密度の指定は多様であるが，もっとも高い密度はセンサごとのサンプリングレートに依存することになる．

本研究で想定するアプリケーションにおいて，アプリケーション A では，興味時間あたりの密度を高く設定することにより，詳細に情報を記録する必要がある．一方で，アプリケーション B の場合には，状況を詳しく思い出すという必要性よりもユーザの記録という目的が強いため，密度が高い必要はない．アプリケーションに応じてセンサ情報を必要とする周期は異なるため，アプリケーション側で指定できるようにする必要がある．図 3.5 ではアプリケーション A，B それぞれの興味時間と興味時間あたりの密度をあらわしている．

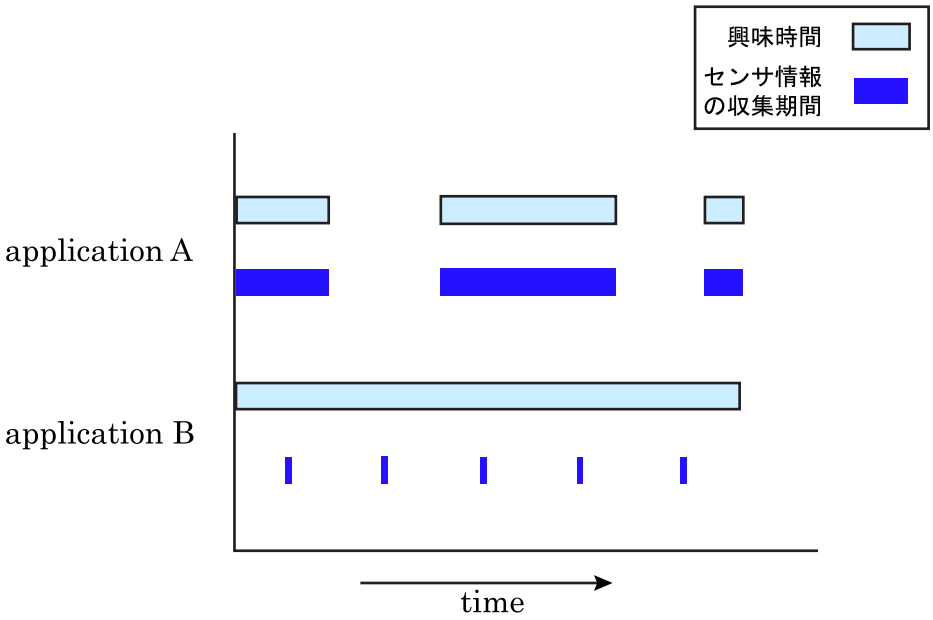


図 3.5: 興味時間あたりの密度

3.7 センサ情報の持続性

センサ情報を永続的に保存するのか，取得した直後に利用するかによってアプリケーションのセンサ情報を利用する性質を区別することができる．またセンサ情報提供フレームワークにおいても，センサ情報の保存に関していくつかの想定が考えられる．

多くのセンサ情報提供環境は，リアルタイムな遠隔の状況や変化をセンサ情報として提供することを大きな目的とするため，現在のセンサ情報の提供に主眼をおいている．IrisNet や Fjord で挙げられているアプリケーション例として，今空いている駐車場の検索や道路の混雑状況などからもその傾向は理解できる．しかし，アプリケーションによって，現在の情報を即座に参照するために収集するのではなく，のちのちの参照のために収集するものも存在する．図 3.6 では，現在のセンサ情報の提供に主眼をおくセンサ情報提供環境と 3 種類のアプリケーションが時間軸に応じてセンサ情報に置く価値を表している．(★) はセンサ情報提供環境が提供するセンサ情報をあらわし，現在に近い情報ほど，センサ情報の質や量が豊富であることをあらわしている．(a) のようなアプリケーションは現在や近い過去の情報のみを必要とし，(b) はある特定の期間，(c) は過去から現在まで一定の情報を必要とする．本研究で対象とするアプリケーションは，アプリケーション (b) や (c) に分類されるものである．

本研究で想定するアプリケーションではセンサ情報を繰り返し参照し，利用する可能性があるためアプリケーション側で保存する．センサ情報提供環境側でのセンサ情報提供ポリシを考慮した場合，アプリケーションは必要なセンサ情報をできるだけ即座に収集するべきである．

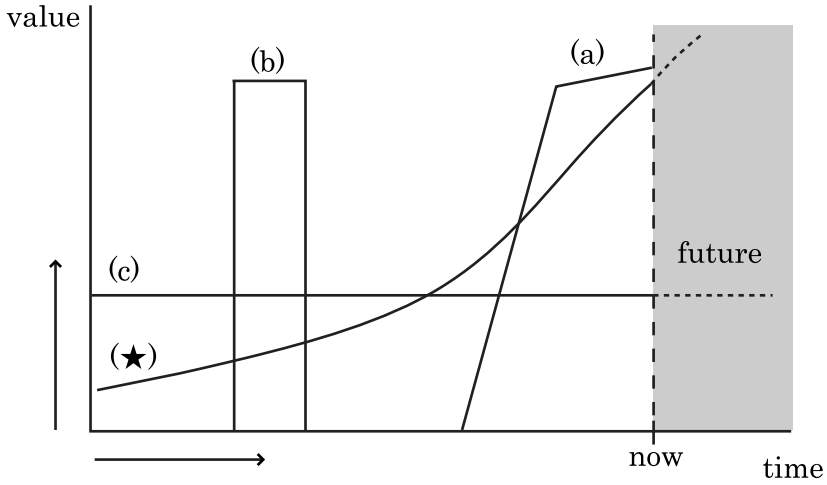


図 3.6: センサ情報の価値

3.8 想定アプリケーションから考察する要件

ここで，第 3.2 節で挙げた想定アプリケーションについて，これまでの項目からの要求を表 3.5 にまとめる．

項目	アプリケーション A	アプリケーション B
興味時間	ある特定の期間	瞬間
時間あたりの密度	大きい	小さい (指定された周期のスナップショット)
興味空間	ユーザの周辺	ユーザの周辺
センサの種類	動画カメラ，音声	位置情報，一緒にいた人
センサ情報の持続性	必要	必要

表 3.5: アプリケーションの要求

アプリケーションフレームワークに求められる要件は以下の三項目である．

アプリケーションによる興味時間におけるセンサ情報の取得密度の指定 常に生成されつづけるセンサ情報に対して，アプリケーションが要求するセンサ情報の取得周期を指定できる必要がある

アプリケーションによる興味空間の広さ指定 アプリケーションはユーザの周辺の広さを指定できる必要がある

センサ情報提供環境に非依存なセンサ情報の要求 センサ情報提供環境に存在するセンサの種類や要求のプロトコルをアプリケーションがあらかじめ知らなくても必要なセンサ情報の要求ができる必要がある

3.9　本章のまとめ

本章では，センサアプリケーションがセンサ情報を取得する上で考慮すべき項目について分析した．多様なセンサ情報提供環境に対して，本章で述べた項目を詳細に指定したセンサ情報の要求を実現するのは困難である．

次章では，この問題を解決するアプリケーションフレームワークの構築について述べる．

第4章　アプローチ

前章までで，想定環境となるセンサ情報提供環境と同環境上でユーザの周辺情報を収集するアプリケーションを実現する上での問題について述べた．

本章では，これらの問題を解決するアプローチとして，センサ情報収集のためのアプリケーションフレームワークについて述べる．

4.1　センサ情報提供環境に非依存なセンサ情報の要求

アプリケーションがセンサ情報の要求において非依存性を実現する上で解決すべき問題として，センサの種類と要求のプロトコルが挙げられる．センサの種類の指定方法として，メタデータによる指定を行うことにより，センサ情報の要求のプロトコルにおいては，プロトコル変換を行うことことにより解決する．

4.1.1　メタデータによるセンサ情報の指定

第 2 章で示したような多様なセンサ情報提供環境を想定し，必要なセンサ情報の収集を実現する場合，どのようなセンサ情報が存在するかを知ることができないだけでなく，アプリケーション側で認知できないセンサ情報を提供している場合にアプリケーションではそのセンサが必要か判断できない．この問題に対処するため，本研究 ではセンサ情報の属性のメタデータ化を行い，センサ情報の要求をする．メタデータの例を表 4.1 に示す．

アプリケーションは，メタデータによるセンサ情報の指定を行う．センサ情報提供基盤環境では，あらかじめ環境内に存在するセンサとメタデータとのマッピングテーブルを生成することにより，アプリケーションが指定したメタデータを用いて，明示的なセンサの指定をせずにセンサ情報の要求が可能となる．図 4.1 はメタデータによるセンサの指定の手順の例を表している．アプリケーションは Video と Audio データを必要としており，Location A では ウェブカメラとマイクの情報が，Location B では ビデオカメラの情報が収集可能である．

表 4.1: メタデータによるセンサ情報の指定

表現例	センサ
Audio	ビデオカメラ，マイク
Video	ビデオカメラ
Image	スチルカメラ，照度センサ
Location	圧力センサ，無線タグ検知システム，GPS
Weather	温度計，湿度計，気圧計
Object	無線タグ

メタデータはアプリケーションとセンサ情報提供環境で共通に利用されるものであり，唯一のマッピングであることが保証される必要がある．

4.1.2　プロトコル変換

センサ情報提供環境によって異なるセンサ情報の要求方式を単一のアプリケーションで実現するために，要求のプロトコルを変換する．プロトコルとは，センサ情報の要求に関

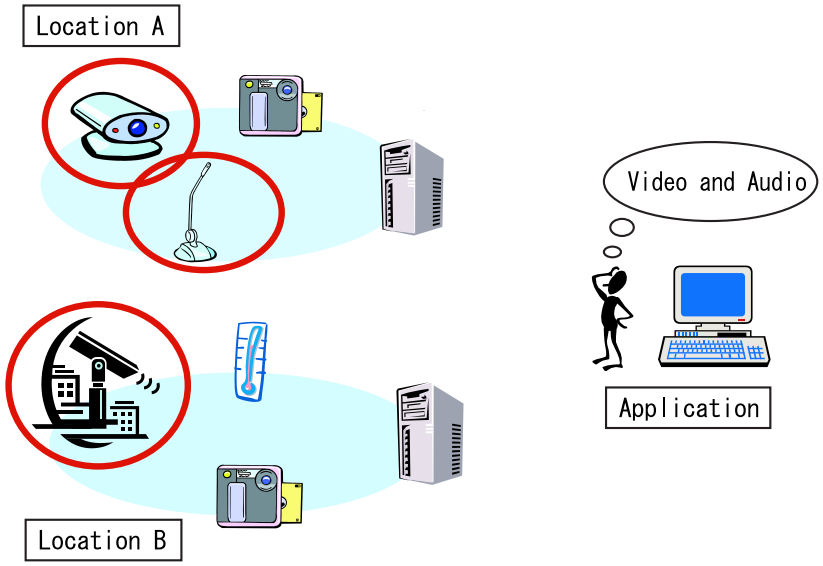


図 4.1: メタデータによる変換

する記述方式，および手順を指す．プロトコル変換の方法として，アプリケーションは本研究で定義する記述に沿ってセンサ情報要求を生成する．生成された要求記述をセンサ情報提供環境に応じて変換することにより，センサ情報の要求および収集を実現する．図 4.2 では，アプリケーションは同一の要求を発行しているが，この要求を変換することにより，Location A と Location B のそれぞれに応じたセンサ情報の要求をしている．

4.2 詳細なセンサ情報の要求

詳細で柔軟なセンサ情報の要求を実現するために特に時間と空間について，アプリケーションによる詳細な要求を実現する．

4.2.1 時間の指定

第 3.1 章で述べたように，時間に関わる項目としてアプリケーションの興味時間の長さと，興味時間あたりの密度がある．アプリケーションの興味時間の長さの指定は具体的な時間で指定する．また，興味時間あたりの密度は，指定された時間内における必要なセンサ情報の量である．この量は，取得時間に比例するため，取得周期を指定する．動画などの連続的なセンサ情報が必要な場合には，常に取得することを指定する．

通常，センサ情報の要求と返信は同期して行われる．センサ情報の要求をすると，そのときに要求された情報が存在する場合には即座に結果が返って来る (図 4.3(a))．この方式の場合，ある一定周期ごとのを取得を実現するために，アプリケーションがタイミングの管理を行う必要がある．この部分を中間機構で実現することにより，アプリケーションの

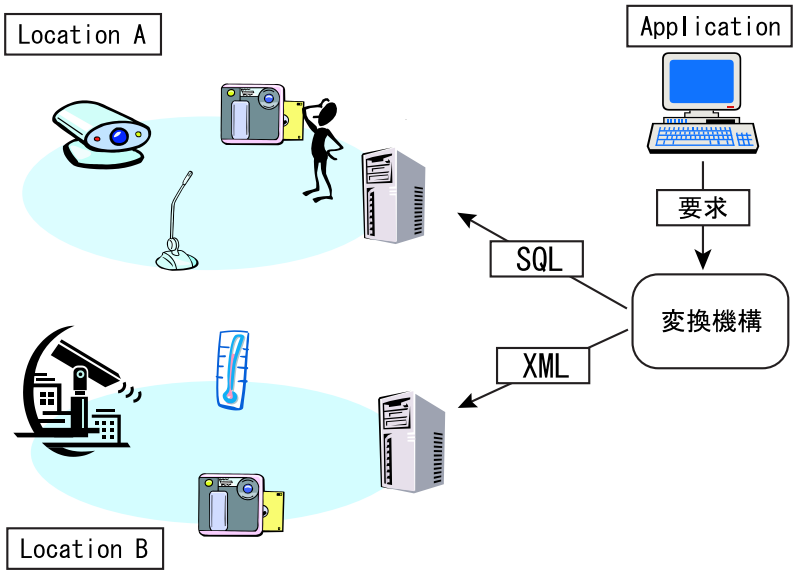


図 4.2: アプリケーションの要求の変換

負担を軽減することが可能である．中間機構とは，センサ情報要求側の変換機構や提供側のいずれかで，収集あるいは提供のタイミングの管理を行う (図 4.3(b))．これにより，アプリケーションは容易に必要なタイミングで情報の取得を実現できる．

4.2.2 空間の広さの指定

アプリケーションがユーザの周辺の広さを指定することにより，そこに含まれるセンサの数量や種類が異なってくる可能性がある．アプリケーションの目的にあわせてセンサ情報の取得範囲を指定する．

要求時には対象となるユーザの位置情報と，その周辺の広さを指定する．[1]ユーザの位置情報は

4.2.3 アプリケーションによる要求の指定

ここまでの議論をもとに，アプリケーションがセンサ情報の要求をする際に指定する項目について以下にまとめる．

- ユーザの位置情報
- センサの種類
  - メタデータによる指定

[1]ユーザの位置情報は第 2.3 節にてユーザの位置情報監視ソフトウェアから送信されることを述べた．

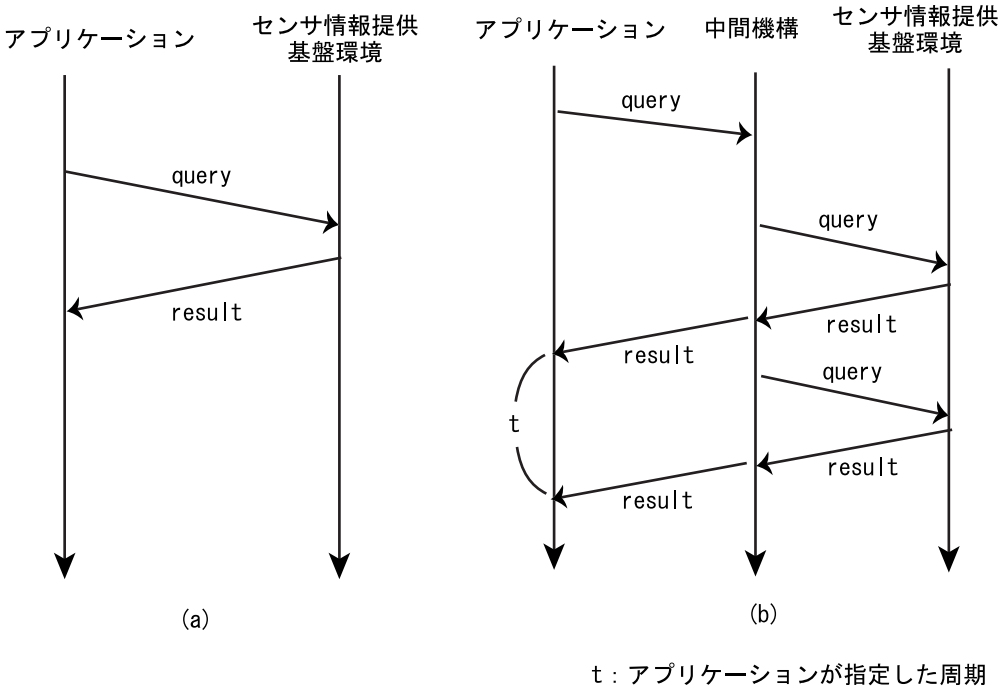


図 4.3: 要求のタイミング

- 時間の指定
  - 周期的な情報要求
    - 周期時間の指定
  - 連続的な情報要求
    - 開始時間および終了時間の指定
    - 外因的なイベントなどによる取得開始の指定
- 空間の指定
  - 数値による広さの指定

4.3 センサ情報要求の変換モデル

センサ情報の要求やプロトコル変換の機構は，動作するコンピュータネットワーク上の場所により，クライアント指向モデル，サーバ指向モデル，プロキシモデルの三つに分けられる [31]．以下に各モデルについて述べる．

クライアント指向モデル アプリケーション側で変換を行うモデル (図 4.4(a))

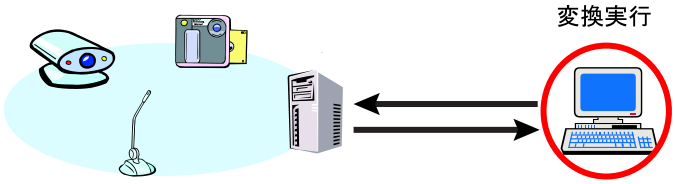
サーバ指向モデル センサ情報提供環境側で変換を行うモデル (図 4.4(b))．この場合，センサプロキシ上で動作することが想定される

プロキシモデル センサ情報提供基盤とアプリケーションの通信の中継を行うサーバで変換を行うモデル (図 4.4(c))

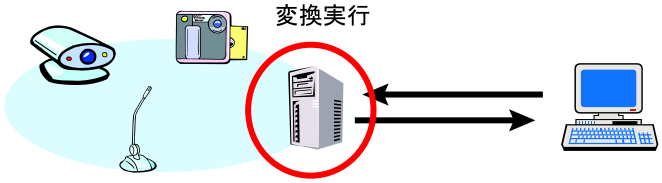
クライアント指向モデルでは，アプリケーションがセンサ情報提供環境ごとに変換モジュールを持つ必要がある．

サーバ指向モデルの場合，センサ情報提供環境側での負荷が増加する．特に，時間の指定に関する要求に対して，サーバ側で対処する必要があるため，アプリケーション固有の取得タイミングにあわせたセンサ情報提供を実現するための負荷が大きい．また，プロキシモデルの場合にも，プロキシにおける負荷が増加するため効率が低下する．

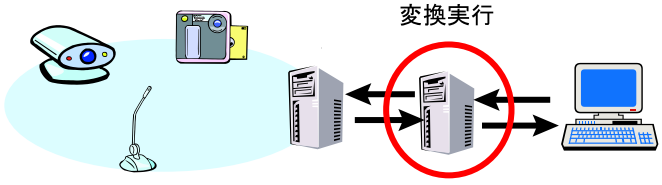
本研究では，クライアント指向モデルを採用し，各センサ情報提供環境が変換モジュールを提供する．アプリケーション側でプロトコル変換を実現するためには，センサ情報提供環境で利用されているプロトコルをアプリケーションが把握しなければならないが，センサ情報提供環境ごとに提供された変換モジュールを利用することにより，アプリケーションが直接プロトコルを把握する必要はなくなる．



(a) クライアント指向モデル



(b) サーバ指向モデル



(c) プロキシモデル

図 4.4: 取得モデル

4.4　本章のまとめ

本章では，ユーザの周辺のセンサ情報を取得する上で生じる問題を解決するアプローチについて述べた．センサ情報提供環境に非依存にアプリケーションがセンサ情報の指定を行うために，プロトコル変換やメタデータによるセンサ情報の要求を行い，より詳細なアプリケーションの要求指定を実現するために，センサ情報を取得する範囲の広さやセンサ情報のセンサ情報を取得する密度の指定方法を提供する．

次章では，これらのアプローチを採用したシステムの設計について述べる．

第5章　センサ情報収集フレームワークの設計

本章では，本研究で提案するユーザの周辺のセンサ情報を収集するフレームワークである **MemoryDirectory** の設計について述べる．まず，MemoryDirectory の概要および設計概観について述べ，次に MemoryDirectory を構成する差部システムの詳細な設計について述べる．

5.1 設計概要

本研究ではセンサ情報提供環境に透過的なセンサ情報収集システムである **MemoryDirectory** を開発した．MemoryDirectory は，アプリケーションからの要求をさまざまなセンサ情報提供環境固有の要求方式に変換し，センサ情報の要求を実現する．本節では，システムの概要をハードウェア，ソフトウェア両面から述べる．

5.1.1 ハードウェア構成

本システムでのハードウェア構成として，ネットワーク接続されたセンサ情報提供環境とセンサ情報の収集対象となるユーザに固有の端末が存在する．MemoryDirectory はユーザの端末上で動作する．ユーザはさまざまなセンサ情報提供環境を移動し，MemoryDirectory はアプリケーションの要求に応じて，ユーザの存在するセンサ情報提供環境に対してセンサ情報の要求を行う．図 5.1 にハードウェア構成を示す．

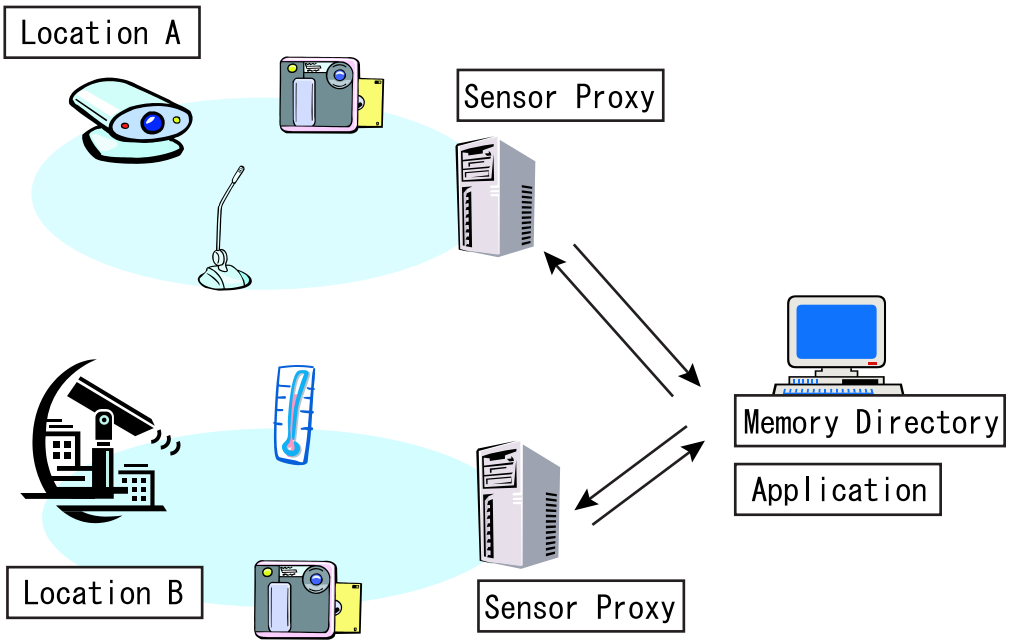


図 5.1: ハードウェア構成

5.1.2 ソフトウェア構成

本システムは位置情報管理機構，スケジューラ，センサ情報要求モジュール，センサ情報管理モジュールから構成される．図 5.2 に各差部システムの関係を示す．

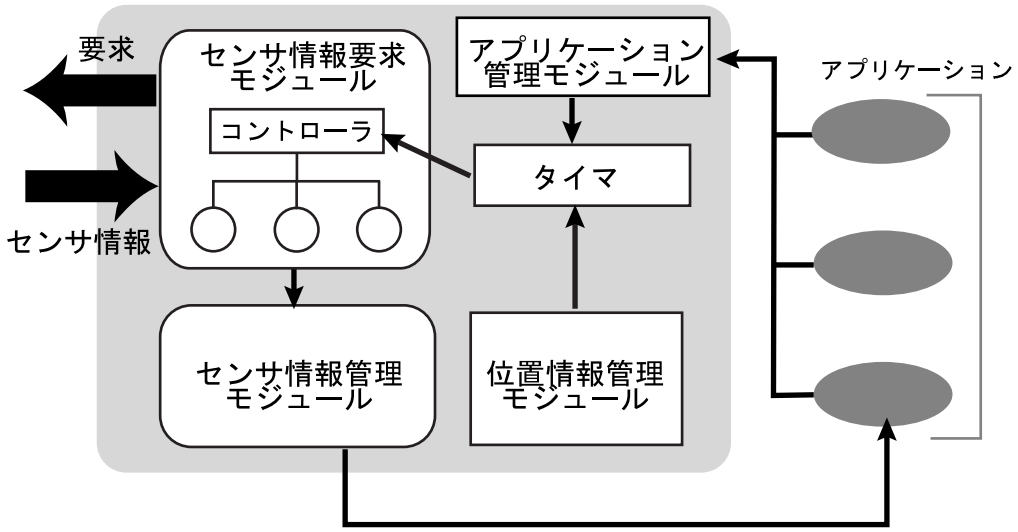


図 5.2: 設計概要

位置情報管理モジュール

本システムでは，ユーザの位置情報監視機構の存在を想定する．位置情報監視機構では，ユーザの位置情報を常に監視し，位置情報管理モジュールに対しユーザの位置情報および，センサ情報の要求先となるセンサ情報提供環境へのポインタを送信する．位置情報管理モジュールでは，受信した位置情報からタイマに対するセンサ情報要求のスケジューリングを行う．

アプリケーション管理モジュール

特定のユーザの周辺のセンサ情報を必要とする複数のアプリケーションからの要求を受け付け，アプリケーションの管理をする．センサ情報提供環境からセンサ情報を受け取った場合には，ここで管理された情報を参照し，アプリケーションに対してセンサ情報の提供を行う．

タイマ

アプリケーションごとに異なる時間のタイミングでセンサ情報を要求するため，センサ情報要求発行をスケジューリングする．

センサ情報要求モジュール

タイマの命令に基づき，センサ情報の要求変換および要求の発行を行う．センサ情報要求モジュールは，センサ情報変換モジュール群とモジュールを管理するコントローラを含む．

センサ情報管理モジュール

要求したセンサ情報の結果を受信し，アプリケーションにセンサ情報の提供を行う．

5.2 動作手順

本節では，MemoryDirectory における動作手順を述べる．以下に，アプリケーションのセンサ情報要求をもとに，センサ情報提供環境に対しセンサ情報要求を発行するまでの手順を示す．また，図 5.3

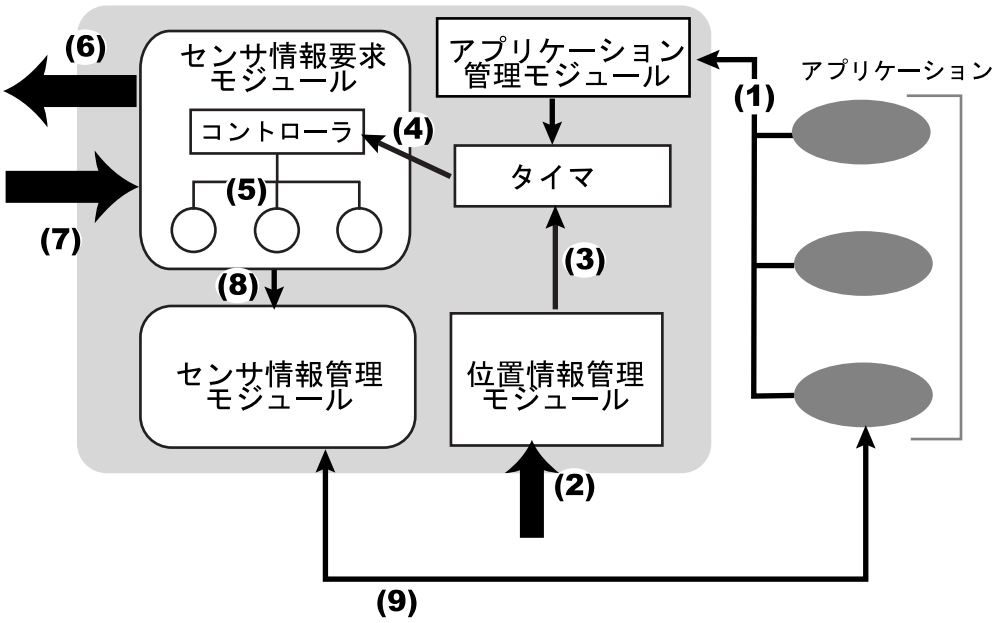


図 5.3: 動作手順

1. アプリケーションがセンサ情報要求の登録を行う
2. ユーザの位置情報およびセンサ情報提供環境へのポインタを受信する
3. アプリケーションの登録の有無を確認し，ある場合にはタイマにセンサ情報要求のスケジューリングをする
4. タイマはアプリケーションが要求するタイミングになると，センサ情報要求をセンサ情報要求モジュールのコントローラに対して発行する
5. コントローラは適切なセンサ情報変換モジュールを選択し，変換を実行する
6. 変換完了後，センサ情報提供環境へ要求を送る
7. センサ情報を受け取る

8. センサ情報要求モジュールはセンサ情報を受け取る
9. 受け取ったセンサ情報を解体，リフォーマットしセンサ情報管理モジュールへ渡す
10. センサ情報管理モジュールはアプリケーションの要求に応じてセンサ情報の提供をする

アプリケーションがセンサ情報を利用する際には，センサ情報管理モジュールに対してアプリケーションがセンサ情報の取得要求を行う．

5.3 アプリケーションによるセンサ情報の指定

アプリケーションによって指定するべき情報は，センサの属性，センサ情報が必要な周期，アプリケーション識別子，などの情報である．指定は XML データによって行う．図 5.4 に記述例を示す．

```
<Request>
    <userID>WXUDMY</userID>
    <applicationID>001</applicationID>
    <sensorAttribute>Wheather</sensorAttribute>
    <sensorAttribute>Image</sensorAttribute>
    <timeSchedule>
        <periodic>
            <interval>10 min</interval>
        </periodic>
    </timeSchedule>
    <space>
        5m
    </space>
</Request>
```

図 5.4: アプリケーションの要求例

アプリケーション識別子は MemoryDirectory システム内でアプリケーション識別のために利用する．このアプリケーションはユーザの周辺約 5 m の気候情報と画像情報を 10 分おきに収集することを要求している．

5.4　位置情報管理モジュール

位置情報管理モジュールは，以下の二つの役割を持っている．

位置情報の受信

ユーザの位置情報を監視し位置情報に応じたセンサ情報提供環境へのポインタを送信する外部のアプリケーションからユーザの位置情報を定期的に受信する．ここで受信する情報は，ユーザの位置情報，センサ情報提供環境の識別子，センサ情報提供環境のアドレスである．

タイマへの通知

ユーザの位置情報の変化があった場合，位置情報管理モジュールはタイマへ通知する．タイマ側では，センサ情報収集のタイミングで位置情報およびセンサ情報提供環境の識別子やアドレスを利用する．タイマへ通知するユーザ情報を表 5.1 に示す．

表 5.1: ユーザ情報

ユーザ情報
ユーザの位置情報
位置情報に対応したセンサ情報提供環境の識別子
センサ情報提供環境のアドレス

5.5　アプリケーション管理モジュール

アプリケーション管理モジュールは，アプリケーションの登録・削除の受付，タイマに対するセンサ情報要求実行のスケジューリング，およびスケジュールからの除去を行う．またアプリケーション識別子によるアプリケーションの管理を行う．

アプリケーションの登録とは，図 5.4 で示したような情報をアプリケーションから受け取ることを指す．ここで，必須項目のチェックや記述の正当性のチェックを行う．タイマに対するスケジューリングにおいては，アプリケーション識別子，スケジュールに関する情報の 2 種類をキーに登録を行う．アプリケーションが登録を削除すると，タイマのスケジューリングから要求実行を取り除く．

5.6　タイマ

複数のアプリケーションのセンサ情報要求のタイミングをアプリケーションの要求に合わせて予約し，センサ情報要求モジュールに対して要求実行を発行する．要求実行をする予約の種類には以下の三種類が存在する．

- 周期型
- 定時型，密度と期間指定
- イベント発火型，密度と期間指定

周期型はセンサ情報収集開始に関する記述がなく，指定された周期でセンサ情報の要求を発行する．定時型は開始時刻を明示し，指定された期間のセンサ情報を収集し続ける．イベント発火型では，イベントとして利用されるセンサ情報の収集を常に行う．

要求発行の条件が満たされると，タイマはセンサ情報要求モジュールに対して，アプリケーション識別子をキーに要求命令を発行する．

5.7　センサ情報要求モジュール

タイマで発行された要求命令をもとに，アプリケーションの要求をセンサ情報提供環境に適した要求記述に変換し，実際に要求を行う．またセンサ情報の受信を行う．センサ情報要求モジュールは，タイマからの要求を受けて適切なセンサ情報変換モジュールを選択するコントローラと，センサ情報の変換および，要求の送信と結果の受信をおこなうセンサ情報変換モジュールによって構成される．

5.7.1　コントローラ

コントローラでは，センサ情報収集時におけるセンサ情報変換モジュールの選択と実行，およびセンサ情報変換モジュールの管理を行う．

センサ情報変換モジュールの選択と実行

現在サポートしているセンサ情報提供環境を把握しており，タイマによる要求を受けて，適切なセンサ情報変換モジュールを選択する．センサ情報変換モジュールは，センサ情報提供環境一つにつき一つ必要となるため，センサ情報提供環境識別子によりセンサ情報変換モジュールを選択可能である．コントローラでは，表 5.2 に示す情報をタイマから受け取り，センサ情報要求モジュールの選択および変換の実行を行う．

表 5.2: コントローラで扱う情報

コントローラで扱う情報
センサ情報提供環境識別子
センサ情報要求に必要となる要求情報
アプリケーション識別子

センサ情報変換モジュールの登録

センサ情報変換モジュールはセンサ情報提供環境につき一つ必要であるため，ユーザが始めて行ったセンサ情報提供環境では対応するセンサ情報変換モジュールを新たに導入する必要がある．センサ情報変換モジュールはセンサ情報提供環境で提供されており，モジュールをダウンロードして MemoryDirectory システムに組み込む．この作業において，センサ情報提供環境識別子を取得し，コントローラで管理する．

5.7.2 センサ情報変換モジュール

センサ情報変換モジュールでは，センサ情報提供環境に固有のセンサ情報要求を実現するために，プロトコルや要求するセンサ情報の変換を行う．個々のモジュールでは変換ルールや変換データなど異なる処理を行う必要があるが，コントローラから統一的に扱うためにモジュール用の雛型を提供し，これにそったモジュールの構築をする必要がある．以下にモジュールに必要となる機能を列挙する．

センサ情報の種類の変換

アプリケーションが要求するメタデータから，センサ情報提供環境で提供するセンサ情報の種類への変換を行う．

センサ情報取得範囲の指定

位置情報によるセンサ情報の収集を制限可能な場合には，ユーザの位置情報および指定された範囲に基づきセンサ情報収集対象となるセンサを選択する．

センサ情報要求の構築

センサ情報の要求は，モジュールが要求命令を受けた時点で実行するため，時間軸に関する考慮は必要ない．このため，センサ情報の種類や範囲の指定情報をもとに，センサ情報提供環境への要求の構築と送信をする．

収集したセンサ情報の提供

収集したセンサ情報は，アプリケーション識別子，センサ情報の名前および値の組で提供できるように変換する必要がある．

5.8 センサ情報管理モジュール

アプリケーションに対して，センサ情報を提供するインタフェースを実現するモジュールである．センサ情報はファイルシステム上での管理を行う．一回のセンサ情報収集によって収集したセンサ情報は，アプリケーションに対して表 5.3 に示す情報とともに提供される．

表 5.3: アプリケーションへ提供する情報

センサ情報の種類
センサ情報提供環境識別子
センサ情報の収集時刻
センサ情報のファイルシステム上のアドレス
アプリケーション識別子

表 5.3 に示す情報の集合をアプリケーションに対して提供するためアプリケーションに対して提供する機能を以下に列挙する．

- 任意のアプリケーションに提供するセンサ情報一覧の提供
- 任意のアプリケーションの要求に対応するセンサ情報の開始および最終収集時刻の提供
- 指定された任意のタイムスタンプを持つセンサ情報の提供

5.9 本章のまとめ

本章では，センサ情報提供環境に非依存にセンサ情報の収集を行うアプリケーションを支援する MemoryDirectory について述べた．MemoryDirectory では，アプリケーションの要求に基づいて，現在ユーザが存在するセンサ情報提供環境におけるセンサ情報を収集する．

MemoryDirectory は位置情報管理モジュール，アプリケーション管理モジュール，タイマ，センサ情報要求モジュール，センサ情報管理モジュールの五つのサブシステムによって構成される．MemoryDirectory は各サブシステムの連携により，移動するユーザの周辺のセンサ情報を収集とアプリケーションへの提供を実現する．

第6章　プロトタイプの実装と評価

前章では，MemoryDirectory の設計について述べた．本章では，同設計に基づいて行った MemoryDirectory の実装について述べる．

まず，MemoryDirectory の実装環境について述べる．次に，MemoryDirectory を構成するサブシステムの実装について述べる．また，サンプルコードを例に，アプリケーションの構築例について述べ，基本性能の評価を示す．

6.1 MemoryDirectory の実装

本節では，MemoryDirectory のプロトタイプの実装について述べる．

6.1.1 実装概観

MemoryDirectory の実装には，Java2 SDK version 1.4.2[13] を用いた．また，XML データのパージングは SAX によって行った． SAX ドライバは Apache XML プロジェクト [11] で提供されている Crimson の SAX2 ドライバを利用した．

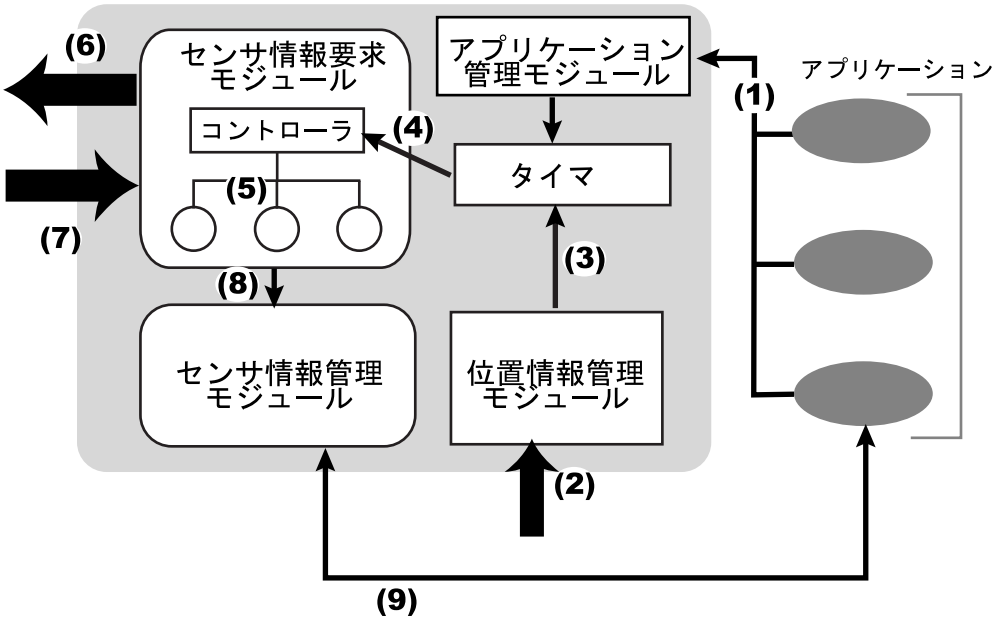


図 6.1: モジュール図

図 6.1 に示すように，MemoryDirectory は四つのモジュールで構成されている．実装として，モジュール間でやりとりされるデータの説明と各モジュールの機能を実現する Java のクラスおよびアプリケーション API について説明する．

6.1.2 データ

モジュール間でやりとりされるデータについて説明する．

ApplicationTaskInfo

アプリケーション要求における必須事項を保持するクラスである．保持される情報を以下に示す．

- アプリケーションが要求するセンサの種類
- アプリケーション ID
- アプリケーションが要求する周辺の広さ

アプリケーション ID は単一の MemoryDirectory につき一意に決定されている必要があり，アプリケーションが実際に情報を取得する場合などには，アプリケーション ID をキーに行う．

ApplicationTask

アプリケーションがセンサ情報を要求する時間周期と ApplicationTaskInfo クラスを保持している．ApplicationTask は アプリケーション管理モジュールが管理するが，定期的な要求発行は ApplicationTask が行うため，Runnable インタフェースを実装しスレッドとして実行される．

UserLocationProfile

ユーザの現在位置，ユーザの現在位置に対応するセンサ情報提供環境の識別子を含む．

6.1.3 アプリケーション管理モジュールおよびタイマ

アプリケーション管理モジュールでは，アプリケーションに対するインタフェースの提供をする ApplicationManager クラス，登録されたセンサ情報要求タスクである ApplicationTask クラス，ApplicationTask クラスより生成されるインスタンスの管理を行う

ApplicationManager クラス

ApplicationManager クラスは ApplicationTask クラスを保持し，そのクラスから生成したインスタンスの管理を行う．機能を以下に示す．

- アプリケーション要求の受け付け
  アプリケーションは XML 形式でセンサ情報要求を行う．この XML のパージング，必須項目のチェック，TimeRequest クラスのインスタンス生成を行う．
- ApplicationTask のインスタンス生成
  アプリケーションからセンサ情報の取得要求を受けて，ApplicationTask のインスタンスを生成する．また，削除や取得のために生成したインスタンスをリストに追加する．インスタンスの識別は，アプリケーションによって指定される アプリケーション識別子によって行う．

- ApplicationTask のインスタンス削除
  アプリケーションからのセンサ情報取得停止要求を受けて，対応する ApplicationTask を削除する．
- 位置情報変更の通知
  ApplicationManager は ユーザの位置情報を保持しており，この値が変更されると，ApplicationManager が保持する全ての ApplicationTask インスタンスに通知する．

ApplicationTask クラス

TimerTask クラスを継承しており，アプリケーションの要求に基づいた時間のタイミングでセンサ情報要求モジュールに対する要求命令を実行する．ApplicationTask 生成時に，ApplicationManager から渡される TimeRequest を保持し，センサ情報要求モジュールに対して ApplicationRequest を渡す．

6.1.4 センサ情報要求モジュール

センサ情報モジュールは，各センサ情報要求変換モジュールを管理するコントローラである ModuleManager クラスと センサ情報変換モジュール群によって実装する．

ModuleManager クラス

ModuleManager クラスでは，各センサ情報要求変換モジュールの保持，管理を行う．また，ApplicationTask からのセンサ情報要求命令を受けて適切なセンサ情報変換モジュールの選択と実行を行う．以下に機能をまとめる．

- センサ情報要求変換モジュールの 保持
  ModuleManager クラスでは，複数のセンサ情報要求変換モジュールを保持し，センサ情報要求変換モジュールの識別は，各クラスが保持する識別子によって行う．
- ApplicationTask によるセンサ情報要求の受けつけ
  ApplicationTask からのセンサ情報要求を受け付ける．複数の ApplicationTask からの要求が発生するため，要求はキューによる管理を行う．
- センサ情報要求変換および取得要求の実行
  ApplicationTask によって渡された ApplicationRequest が保持するセンサ情報提供環境識別子をもとに，センサ情報要求変換モジュールを選択し，変換の実行をする．

センサ情報要求変換モジュール

センサ情報要求変換モジュールはセンサ情報提供環境につき一つ必要で，ApplicationRequest クラスから実際のセンサ情報の要求命令を生成し，センサ情報の収集を行う．

センサ情報要求変換モジュールを定義するために，インタフェースを図 6.2 に示す．

```
1 public interface RequestTranslationModule{
2    public void execute();
3    public String getSensorEnvironmentId();
4 }
```

図 6.2: センサ情報要求変換モジュールのインタフェース

センサ情報変換モジュールの実装

上述したセンサ情報提供環境のセンサ情報変換モジュールの一部を図 6.3 に説明する．

6.1.5 センサ情報提供環境のプロトタイプ

センサ情報提供環境のプロトタイプの構築を行った．本プロトタイプで実装したセンサは，WEB カメラ (図 6.4 右上)，TinyOS mote[10](図 6.4 右下)，RFID タグシステム [24](図 6.4 左) である．取得できるセンサ情報はそれぞれ，静止画像，温度と湿度と照度，特定のエリアへの IN / OUT のみの位置情報である (図 6.4 右下参照)．これらのセンサ情報は常時取得され，プロキシサーバとなるホストへ定期的に送られる．プロキシサーバとなるホストでは，sql によるクエリを想定し，PostgreSQL[19] によるセンサ情報データベースを構築した．図 6.5 に実現したセンサ情報提供環境を示す．

ユーザの位置情報の取得方法に関しては，RFID のタグにふられた ID をユーザ識別子として利用し，各環境に設置されたリーダが取得した識別子は常時，位置情報監視サーバへ提供される．位置情報監視サーバではアプリケーションが登録したユーザの識別子を監視しており，ユーザの存在を検知した場合，アプリケーションへユーザ識別子とユーザが発見されたセンサ情報提供環境識別子を通知する．

アプリケーション側では，センサ情報の要求に利用するために，PostgreSQL の JDBC ドライバを用いている．

6.1.6 サンプルアプリケーション

本研究でのアプリケーション例として，6.7 にスナップショットを掲載する．本アプリケーションは，ユーザの周辺情報を 10 分ごとに収集する日記アプリケーションである．センサ情報の要求に利用した XML を 図 6.6 に示す．

```
1  import jp.ac.keio.afc.ht.memorydirectory.*;
2  import java.sql.*;
3
4  public class AccoEnvironmentTranslationModule
5                                      extends TranslationModule {
6     private String eid = null;
7     public AccoEnvironmentTranslationModule(ApplicationRequest ar){
8         // センサ情報提供環境識別子の設定
9         setId("jp.keio.ac.sfc.ht.t20.ACCO");
10        ... 省略 ...
11    }
12
13    public void execute(){
14        prepareDBAccess(); // SQL での要求のための準備作業
15        createSQLQuery();  // SQL の要求ステートメントの生成
16        doQuery(); // SQL による query の実行
17    }
18
19    public String getId(){
20        return eid;
21    }
22    .....
23 }
```

図 6.3: センサ情報変換モジュール

6.2 検証

MemoryDirectory は，アプリケーションの構築の負担を減少させることを目的としている．本節では，MemoryDirectory を利用する場合としない場合とで，アプリケーション構築におけるプログラミングの記述量についてを比較する．

図 6.8 は MemoryDirectory を利用しない場合の，センサ情報要求部分のプログラムの仮想コードである．MemoryDirectory を利用しないでアプリケーションを構築する場合，アプリケーションプログラマがユーザの移動先に存在するセンサ情報提供環境の詳細をあらかじめ把握する必要がある．アプリケーションには，センサ情報の要求記述，センサ情報の記述，提供モデルなどの情報を含め，センサ情報の収集を実現する必要がある．図 6.8 では 8 行目から始まるスイッチ文がそれにあたる．想定されるセンサ情報提供環境分のコードをアプリケーションプログラマが記述する必要があるため，非常に負担が大きい．

いっぽうで，MemoryDirectory を利用した場合，アプリケーションプログラマは，センサ情報提供環境に関する情報に一切関知する必要がない．センサ情報が取得可能な場合には，MemoryDirectory で収集・蓄積する．このため，アプリケーションプログラマが記述する量は非常に少ない．記述する項目として，センサ情報の要求およびセンサ情

図 6.4: 利用したセンサ

報の取得の二点が最低限記述されていれば十分である．図 6.9 は，アプリケーションが MemoryDirectory を利用する場合の API を示している．

上記の比較から，MemoryDirectory を利用することによって，アプリケーション開発における，センサ情報の収集に関連するプログラム記述量を減少させることができる．これにより，アプリケーションプログラマは，センサ情報を利用したアプリケーション本来の処理部分の構築に労力を当てることが可能になると考えられる．

6.3 本章のまとめ

本章では，MemoryDirectory の実装とアプリケーションの構築方法について述べた．また，構築したアプリケーションとして，ユーザの周辺情報を定期的に取得するアプリケーションについて述べた．また，検証として MemoryDirectory を利用した場合としない場合のプログラムの記述量について比較し，アプリケーション開発における負荷を軽減させることができることを示した．

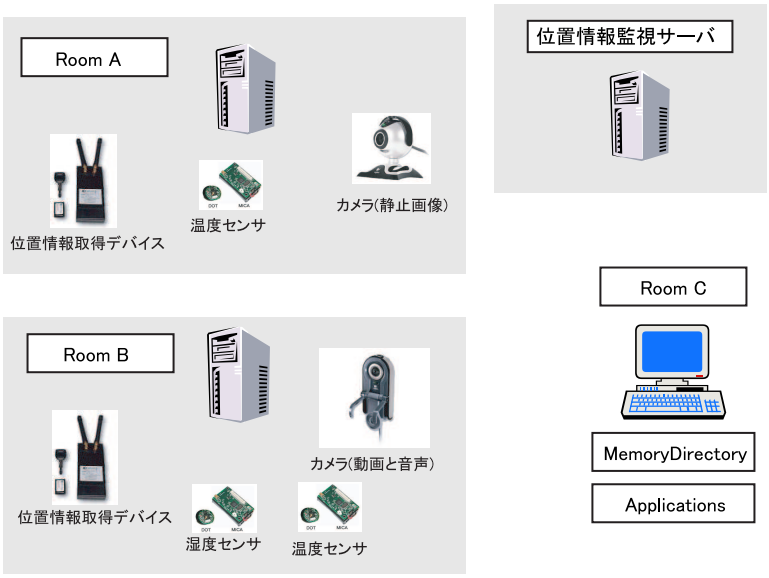


図 6.5: センサ情報提供環境

```
<Request>
    <applicationID>Logging</applicationID>
    <sensorAttribute>Wheather</sensorAttribute>
    <sensorAttribute>Image</sensorAttribute>
    <timeSchedule>
        <periodic>
            <interval>10 min</interval>
        </periodic>
    </timeSchedule>
    <space>
        3m
    </space>
</Request>
```

図 6.6: サンプルアプリケーションの要求

図 6.7: アプリケーション例

```
1
2  While(true){
3      // ユーザの現在地の取得
4      description= getSensorInfraPointer(Location);
5       /*  現在地に応じたセンサ情報提供環境の選択
6        *  とクエリの生成
7        */
8     switch(description.get InfraID()){
9     case “ t20 ”:
10             String s[] = getSensorList(“ t20 ”);
11             createSQLQuery(s);
12    case “ delta213 ” :
13  ~~~~~
14    default:
15    // 現在地に対するセンサ情報提供環境を知らない
16    }
17    sleep(interval);  // 指定周期によるセンサ取得
18 }
19
```

図 6.8: MemoryDirectory を利用しない場合

```
1  // センサ情報要求の登録
2  register(xmlDocument);
3  // センサ情報の index を取得
4  getDataIndex(appId);
```

図 6.9: MemoryDirectory を利用する場合の API

第7章　関連研究

前章では，センサアプリケーションのセンサ情報利用モデルについて整理し，

本章では，特に記憶を目的とするセンサアプリケーションおよびセンサ情報提供基盤システムに関する関連研究を取り上げ各研究の特徴と，利点および欠点について述べる．

7.1 環境に遍在するセンサ情報の収集：体験の記憶を目的とするシステムやアプリケーション

本節では，環境に遍在するセンサから情報を収集する手法によってユーザの周辺情報の取得・蓄積を実現するアプリケーションについて，目的別に紹介する．

7.1.1 短期的な蓄積

Dèjá Vu Displays [25] は Aware Home プロジェクトの一環として行われている研究である．Dèjá Vu Displays では，ユーザが料理などのシーケンシャルな手順を踏む作業を行っているときに，つい先程まで行っていた作業をディスプレイに表示して，ユーザの作業を支援することを目的としている．ユーザの作業を常に複数台のカメラでキャプチャし，キャプチャした画像を専用のディスプレイに表示するすることで，実現する．Dèjá Vu Displays では特に，作業に集中しているユーザに対して，一瞬だけその注意を逸すと，ついさきほどまで覚えていたことを突如忘れてしまう，いわゆる度忘れに対処することに焦点を当てているため，アプリケーションが必要とするセンサ情報の時間軸的スコープは現在から数十分前位と短い．アプリケーションは家庭内での利用に限定しているため，単一の空間において特定のセンサ (カメラ) のみから情報を取得する．

SPECs [15] では研究室やオフィスなどの閉じた空間ではなく公共空間において，人やものなどの検知を目的とする基盤システムの提案をしている．SPECs では付加的なインフラストラクチャを用いずに実現するため peer to peer 通信でセンサ同士の認識をする単純なデバイスを利用する．

Forget-Me-Not[16] は ParcTab[21] という小型携帯端末によって情報の収集および，閲覧を実現する．ParcTab は赤外線によって通信可能であるため，周辺のセンサや他の ParcTab と情報のやりとりを行い，ユーザの行動履歴を取得する．ユーザは自分専用の ParcTab を常に持ち歩くため，ついさきほどの出来事を思い出す，というような目的で利用するには最適である．

7.1.2 長期的な蓄積

データの取得後の利用における問題に対して取り組んでいるのが，Microsoft Research の MyLifeBits[6] である．MyLifeBits では個人にかかわるデジタル化されたさまざまな情報をデータベースに保存し，その情報を容易に参照することを目的としたプロジェクトである．保存された情報に対して，リンクを用いた連想的な情報の検索やテキストの注釈の付加，情報の再構築によるストーリの作成ができるようにインタフェースが工夫されている．センサ情報などを対象にしていないため，情報の取得すなわち，データベースへの情報の入力に関しては，自動的に行うような機構を用いてはいない．

Georgia Tech の Life Memory Box プロジェクト [22] では，家庭内での家族の想い出を蓄積するためのシステムを実現するために，家族へのインタビューに基づいて，ハード

ウェアとシステムを構築した．ここでは，特に家族の想い出の蓄積に焦点を当てているため，センサ情報の取得は必要なときに明示的に記録するイベントドリブン型である．

7.1.3 特定の経験を蓄積

Comic Diary [23] はカンファレンスなど特別な 1 日の経験を漫画として記録するアプリケーションである．カンファレンスに参加するユーザは PDA を位置情報を取得可能なデバイスを携帯しており，ユーザは参加する予定のセッション，参加したセッションの感想，出会った人などをことあるごとに PDA に対して入力する．センサ情報やユーザの入力情報，ユーザの個人的なプロファイル情報を用いて，漫画を自動生成し，1 日の体験として，保存・共有する．

Rememberer [5] は Comic Diary と同様，ある特定の経験を記憶，共有するためのツールキットである．特に，美術館や博物館などの施設で，見たもの，経験したことを，環境側に設置されたセンサで取得し，ユーザが持つセンサデバイスで位置情報の認識を行うことにより実現する．Rememberer ではセンサ情報を HTML で再構成し，ユーザに提供する．

7.2 センサの携帯によるアプローチ：体験の記憶を目的とするシステムやアプリケーション

本研究では環境に遍在するセンサ情報の収集によって，ユーザの記憶を助けるアプリケーションの実現を目指しているが，ユーザが携帯するセンサ情報を利用したアプローチによる実現を目指すプロジェクトも多数存在する．

LifeSlice[29] はユーザが首に下げた小型のデジタルカメラによって，ユーザの視点を定期的にキャプチャするシステムである．カメラによって取得した画像は，一日の終わりに PC へ取り込み，さまざまなビューワアプリケーションによって画像情報の閲覧や編集を行う．スタンドアロンなセンサデバイスとして利用されているため，利用においては一日の終わりに取り込むという作業が必要になり，自動的にユーザの記憶に関わるセンサ情報を提供する，という点で問題があるが，この点を改善し，また環境に遍在するセンサ情報との組合せによってより便利で楽しいアプリケーションが実現できるであろう．

7.3 本章のまとめ

本章では，センサ情報を利用して人間の記憶を補助することを目的としたシステムやアプリケーションについて紹介した．多くのセンサ情報を利用したアプリケーションは，ある特定の空間におけるセンサ情報のみの利用を考慮している．MemoryDirectory を利用することにより多くのアプリケーションが特定の場所に依存せずにセンサ情報を収集してアプリケーションを構築することが可能になる．

第8章　結論

8.1　まとめ

ヘテロジニアスなセンサ情報提供環境が多数存在する場合に，その環境内や環境間を移動するユーザの周辺のセンサ情報を収集することにより，ユーザの体験に関わるセンサ情報を蓄積し，ユーザの記憶を補助するアプリケーションの実現が可能である．しかし，単一のアプリケーションが要求手法やセンサ記述の異なる複数の環境に対して，センサ情報の要求をすることは，アプリケーション開発者にとって手間がかかり現実的ではない．アプリケーションがセンサ情報を要求する場合に，アプリケーションに対してセンサ情報提供環境非依存性を提供することは有用である．また，アプリケーションからセンサ情報の要求をする上で，時間や空間に関する指定を柔軟に行うことは重要である．

本論文では，これらの機能要件を満たしたセンサ情報収集フレームワークのプロトタイプシステムである MemoryDirectory を設計および実装した．MemoryDirectory は アプリケーションに対しては単一のセンサ情報要求方法を提供し，実際にセンサ情報の収集を行う場合には，センサ情報提供環境ごとに異なる要求記述に変換することによって，アプリケーションからのセンサ情報提供非依存性を実現する．

8.2　今後の課題

MemoryDirectory の今後の課題として以下の三点が挙げられる．

動的なセンサ情報要求モジュールのロード

現在の設計では，MemoryDirectory が未知のセンサ情報提供環境に出会った場合や，センサ情報提供環境側でセンサ情報などに関する変化があった場合，変換モジュールを動的にロードすることができないため，情報要求モジュールのダウンロードを再度する必要があるためユーザにとって手間がかかる．しかし，さまざまな環境へ移動するユーザの周辺情報を自動的に収集するためには，センサ情報要求モジュールの動的ロードと変換の実行を実現する必要がある．

センサ情報提供環境におけるロケーションモデルとアプリケーションの要求のマッピング

センサ情報提供環境におけるセンサ情報の位置やユーザの位置の粒度の情報が提供されると，アプリケーションが要求するセンサ情報を指定する場合により詳細で正確な指定が可能になる．これを実現するためには，ロケーションモデルの記述として一般化可能な形式を提供することにより，アプリケーションによる詳細なセンサ情報の要求を実現する必要がある．

センサ情報の保存方法

センサ情報提供環境から収集したセンサ情報は，アプリケーションに提供するために，ファイルシステムにローデータのまま保存される．ハードディスク容量を圧迫する問題とセンサ情報の閲覧・検索における問題を解決する必要がある．

参考文献

[1] J. Chen, D. J. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A Scalable Continuous Query System for Internet Databases. In *Proceedings of 2000 ACM SIGMOD Intl. Conference on Management of Data (SIGMOD 2000)*, pages 379–390, May. 2000.

[2] Open GIS Consortium. Sensor Web Enablement and OpenGIS SensorWeb. http://www.opengis.org/functional/?page=swe.

[3] A. Deshpande, S. Nath, P.B. Gibbons, and S. Seshan. Cache-and-Query for Wide Area Sensor Databases. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 503–514, Jun. 2003.

[4] D. Estrin, J. Heidemann, R. Govindan, and S. Kumar. Next Century Challenges: Scalable Coordination in Sensor Networks. In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)*, Aug. 1999.

[5] M. Fleck, M. Frid, T. Kindberg, E. O'Brien-Strain, R. Rajani, and M. Spasojevic. Rememberer: A Tool for Capturing Museum Visits. In *Proceedings of UbiComp 2002: Ubiquitous Computing, 4th International Conference*, pages 48–55, Sep. 2002.

[6] J. Gemmell, G. Bell, R.Lueder, S.Drucker, and C. Wong. MyLifeBits: Fulfilling the Memex Vision. In *Proceedings of ACM Multimedia 2002*, pages 235–238, 12 2002.

[7] P.B. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan. IrisNet: An Architecture for a Worldwide Sensor Web. *IEEE pervasive COMPUTING*, 2(4):22–33, Oct-Dec. 2003.

[8] A. Harter and A. Hoppe. A Distributed Location System for the Active Office. *IEEE Network*, 8(1):22–33, Jan. 1994.

[9] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The Anatomy of a Context-Aware Application. In *Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM '99)*, pages 59–68, Aug. 1999.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System Architecture Directions for Network Sensors. In *Proceedings of ASPLOS 2000*, Nov. 2000.

[11] http://xml.apache.org/.

[12] S.S. Intille, E.M. Tapia, J. Rondoni, J. Beaudin, C. Kukla, S. Agarwal, L. Bao, and K. Larson. Tools for Studying Behavior and Technology in Natural Settings. In *Proceedings of UbiComp 2003: Ubiquitous Computing, 5th International Conference*, pages 157–174, Oct. 2003.

[13] Standard Edition (J2SE) 1.4.2 Java 2 Platform. http://java.sun.com/j2se/1.4.2/index.jsp.

[14] J.M. Kahn, R.H. Katz, and K.S. J. Pister. Next century challenges: mobile networking for " Smart Dust ". In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 271–278, 1999.

[15] M. Lamming and D. Bohm. SPECs: Another Approach to Human Context an Activity Sensing Research, Using Tiny Peer-to-Peer Wireless Computers. In *Proceedings of UbiComp 2003: Ubiquitous Computing, 5th International Conference*, pages 192–199, Oct. 2003.

[16] M. Lamming and M. Flynn. "Forget-me-not" - Intimate Computing in Support of Human Memory. In *Proceedings of the '94 Symposium on Next Generation Human Interface*, Feb. 1994.

[17] U. Leonhardt and J. Magee. Multi-Sensor Location Tracking. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 203–214, Oct. 1998.

[18] S. Madden and M. J. Franklin. Fjording the Stream: An Architecture for Queries over Streaming Sensor Data. In *Proceedings of the 18th International Conference on Data Engineering, 26 February - 1 March 2002, San Jose, CA*, Mar. 2002.

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

[20] J.M. Rabaey, M.J. Ammer, J.L. da Silva, D. Patel, and S.Roundy. PicoRadio Supports Ad Hoc Ultra-Low Power Wireless Networking. *IEEE Computer*, 33(7):42–48, Jul. 2000.

[21] B. N. Schilit, N. Adams, R. Gold, M. Tso, and R. Want. The PARCTAB Mobile Computing System. In *Proceedings of the Fourth Workshop on Workstation Operating Systems*, pages 34–39, Oct. 1993.

[22] M.M. Stivens, G.D. Abowd, K.N. Truong, and F. Vollmer. Getting into the Living Memory Box: Family archives & holistic design. *Personal Ubiquitous Computing.*, 7(3-4):210–216, 2003.

[23] Y. Sumi, R. Sakamoto, K. Nakao, and K. Mase. ComicDiary: Representing Individual Experiences in a Comics Style. In *Proceedings of UbiComp 2002: Ubiquitous Computing, 4th International Conference*, pages 16–32, Sep. 2002.

[24] RF Code Spider RFID System. http://www.indatasys.comhtml/products/.

[25] Q.T. Tran and E.D.Mynatt. What Was I Cooking? Towards Dèjá Vu Displays of Everyday Memory. Technical Report GIT-GVU-TR-03-33, Georgia Institute of Technology, 2003.

[26] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The Active Badge Location System. *ACM Transactions on Information Systems*, 10(1):91–102, Jan. 1992.

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

[28] ｅ！プロジェクト. http://www.kantei.go.jp/jp/singi/it2/others/e-pro011213.html.

[29] ライフスライス研究所. http://www.lifeslice.net/.

[30] 森ビル株式会社. 平成１４年度 e!プロジェクト (他機能都市街区におけるマルチデバイス環境を活用した，高度な情報提供サービス基盤の構築と事業性に対する調査研究) に関する業務委託報告書 第四章, 3 月 2002.

[31] 由良淳一. 環境適応的な分散サービス変換機構の構築. Master's thesis, 慶應義塾大学, 2001.

謝辞

本研究を進めるにあたり，絶えず懇切丁寧な御指導を賜わりました，慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝致します．また，本論文の副査として貴重なご助言を頂いた，慶應義塾大学環境情報学部教授村井純博士，同大学同学部専任講師高汐一紀博士に深く感謝致します．

また，慶應義塾大学特だ研究室の諸先輩方には，折りにふれ貴重な示唆や御指導を頂きました．ここに，深い感謝の意を表します．

平成 15 年 1月 15 日

岩谷 晶子