社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－UBI－4 (3)
2004／4／23

センサのメタ情報を利用した センサデータ取得ミドルウェアの構築

丸山 大佑[1] 青木 俊[1] 高汐 一紀[1] 徳田 英幸[1 2]
[1] 慶應義塾大学大学院 政策・メディア研究科 [2] 慶應義塾大学 環境情報学部

既存の環境において，アプリケーションがセンサデータを取得する際のセンサの多様性，アプリケーション要求の抽象性が問題となる．本論文では，センサのメタ情報を管理するミドルウェア MARS を実現する．MARS はアプリケーションに対してセンサの多様性を隠蔽し，センサデータの取得を容易にする．センサのメタ情報にはセンサの種類のようにセンサ生産時から不変のもの，バッテリ残量のように可変であるがセンサ内で取得可能なもの，センサの位置情報のように可変かつセンサ内では取得困難なものがある．本論文では，それぞれのメタ情報について取得，管理手法を考察した．MARS はセンサのメタ情報をサーバで管理し，アプリケーションのセンサデータ要求を解決することにより，センサを多数設置，管理することを容易にする．本論文では，小型センサノードを利用して，MARS を実装し，評価を行なった．

Sensor Data Management Middleware Utilizing Sensor Metadata

Daisuke Maruyama[1], Shun Aoki[1],
Kazunori Takashio[1] and Hideyuki Tokuda[1 2]

[1]Graduate School of Media and Governance, Keio University
[2]Faculty of Environmental Information, Keio University

In this research, we propose a middleware called MARS which acquires data from sensor nodes by utilizing metadata. These nodes are becoming cheaper, and are expected to be in wide use in the future. However, current research that target wireless sensor nodes do not provide name resolution scheme that relate names of objects with their location. Thus, applications can not send requests to wireless sensor nodes based on their location. In contrast, MARS accepts requests which specifies the sensing area, and provides data acquired there. This enables applications to acquire sensor data of the target area without being aware of independent sensors. In this thesis, we have implemented MARS on wireless sensor nodes.

1 はじめに

近年，コンピュータの小型化と処理能力の向上，通信技術の進歩により，あらゆる機器にセンサやコンピュータを埋め込み，ユーザの作業を支援するユビキタスコンピューティング環境 [1] の研究が注目されている．ユビキタスコンピューティング環境の目的に，コンテキストアウェアアプリケーションや環境モニタリングアプリケーションの実現がある．コンテキストアウェアアプリケーションは環境内に設置されたセンサにより位置情報や温度情報などの環境情報を取得して，環境やユーザ，機器のコンテキストを推測し，コンテキストに適したサービスを動的に構築する．コンテキストとは，センサによって環境情報を数量化したセンサデータを基に，アプリケーションが挙動を変えるための判断基準である．そして，アプリケーションとは，センサデータやユーザの入力を基に，アクチュエータの制御を行うプログラムである．例えば３０℃というセンサデータは，オフィスの空調管理では暑いというコンテキストに当てはまり，銭湯の湯加減調節では冷たいというコンテキストに当てはまる．環境モニタリングアプリケーションは，環境情報をユーザの要求に沿った形式で提供する．コンテキストアウェアアプリケーションや環境モニタリングアプリケーションは環境情報を取得した上で，コンテキストへの変換やユーザへの提供を行うため，センサによる環境情報の取得が重要である．

現在のコンピューティング環境では，エアコンに付属する温度センサや自動ドアに付属する人体検知用焦電赤外線センサのように，環境に設置されているセンサは，機器と固定的に接続されている．機器内ではセンサ以外に，モータの動力やディスプレイの表示機能など環境に働きかけるアクチュエータと，それらを利用し，ユーザの入力に従って動作するアプリケーションが存在する（図 1-a）．センサとアプリケーションが固定的に接続されているため，ユーザがセンサを身につけることなく環境に設置されたセンサから，ユーザの移動に対応してコンテキストを抽出するような，不特定のセンサを利用することが必要とされるアプリケーションは実現が困難である．

それに対して，ユビキタスコンピューティング環境では，現在行われているセンサプラットフォームの研究 [2, 3, 4, 5] により，個々のセンサに通信機能や計算機能を持たせ，センサを機器から分離して利用することが可能である．これにより，機器とセンサ，そして，それらを制御，利用するアプリケーションを分離することで，従来のコンピューティング環境より多くの環境情報を利用することを可能とし，コンテキストアウェアアプリケーションや環境モニタリングを実現することができる（図 1-b）．

ユビキタスコンピューティング環境では，アプリケーション，センサ，アクチュエータが機器と分離し，それらが相互に接続可能である．このため，アプリケーションが利用できる機器やセンサの種類は数多く存在する．また，同じ目的で利用する機器やセンサでも機能，性能が異なる．例えば，温度や明るさなど，取得できる環境情報の種類は多様であり，センサでも，センサデータの分解能や取得範囲，サンプリングレートなどの機能が異なる．様々なセンサを１つのアプリケーションが利用するためには，そのようなセンサの種類や機能の違いを吸収する必要がある．

2 本研究の目的

本節では，ユビキタスコンピューティング環境においてセンサを利用する際の課題について考察し，本研究の目的を述べる．

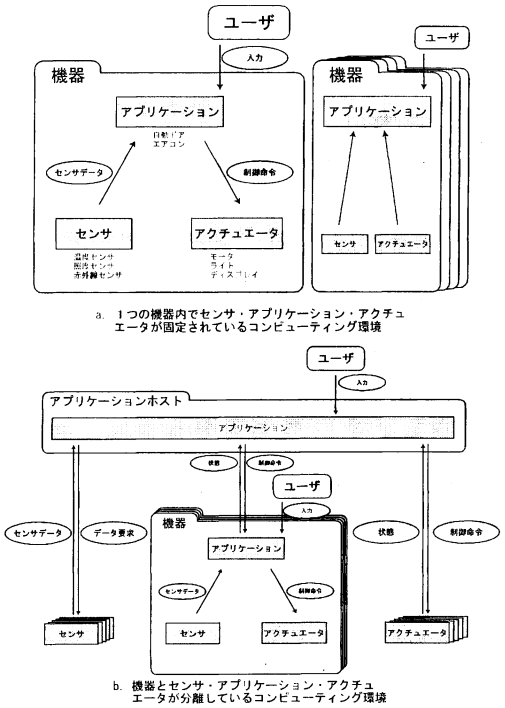

図 1: 既存のコンピューティング環境とユビキタスコンピューティング環境

2.1 ユビキタスコンピューティング環境におけるセンサ利用の課題

ユビキタスコンピューティング環境では，センサ構成が変化する環境でアプリケーションを利用することや，同じアプリケーションを異なる環境で利用することが想定される．しかし，アプリケーションとセンサの分離により，あらかじめセンサ構成を把握することは困難であるため，環境のセンサ構成に依存せずにアプリケーションがセンサデータを取得する必要がある．センサ構成を把握するということは，センサの種類や位置，機能など，センサやセンサデータに関するセンサデータ以外の情報であるセンサのメタ情報を把握することである．

センサとアプリケーションが分離している環境で，アプリケーションを利用すると次のような問題が考えられる．

センサのメタ情報の把握

環境に存在するどのセンサから必要なセンサデータを取得できるかが不明であるため，センサデータを取得するには，アプリケーションは個々のセンサのメタ情報を把握しなければならない．しかし，アプリケーション作成時には利用される環境のセンサ構成が不明であるため，要求を満たすために必要なデータと不必要なデータをアプリケーションが判別し，各センサと通信してデータを取得することは大きな負荷となる．

センサの多様性

アプリケーション構築時に，アプリケーションが利用される環境のセンサ構成は不明である．例えば，室内の温度管理を行うアプリケーションは，室内に分散している温度センサを利用する．しかし，アプリケーション構築時には，利用環境に温度センサが存在するかは不明である．また，温度センサが存在した場合でも，センサ毎に分解能や精度，通信手段などの機能が異なることが想定される．環境に存在するセンサが多様で，各センサのメタ情報が異なるため，アプリケーションがセンサのメタ情報を把握することが困難である．

センサ構成の動的変化

それぞれの環境では複数のアプリケーションが動作していることが想定される．1つのアプリケーションのために環境にセンサの数を増やして，センサデータを取得する粒度を細かくした時，そのセンサを他のアプリケーションからも利用できることがある．しかし，後者のアプリケーションは増加したセンサのメタ情報を把握していないため，そのセンサを利用することはできない．また，ユーザの移動に伴ってユーザに付属するセンサが移動する場合，既存のアプリケーションではセンサの移動に伴って提供するサービスを変化させることは困難である．このように，動的にセンサが増減，移動，変更することにより，アプリケーションがメタ情報を把握することは困難となる．

アプリケーション要求の抽象性

アプリケーションは環境に存在する多数のセンサによって取得したセンサデータの中から，目的に沿った一部のセンサデータを利用する．アプリケーションが必要なセンサデータを取得するために，センサのメタ情報を基にした抽象的な要求が生じる．その際，アプリケーションに必要なのは「テレビの周りの明るさ」というような情報であって，その情報がどのセンサで取得されたかということや，個々のセンサのメタ情報は必要ない．

アプリケーションはセンサデータを利用して，対象物のコンテキストの抽出やアクチュエータの制御を行なう．このため，対象物やアクチュエータの位置を基にした要求は特に重要である．

2.2 本研究の目的

本研究では，センサとアプリケーションが分離しているユビキタスコンピューティング環境において，センサの多様性をアプリケーションが意識することなく，抽象的な要求によってセンサデータを利用できるセンサデータ取得ミドルウェア MARS を構築する．アプリケーションがユーザや機器，およびその周囲に関する環境情報を取得することに着目したシステムを構築し，センサの種類や付属する物というようなセンサのメタ情報を利用した環境情報の取得を実現する．また，環境のセンサ構成は動的に変化するため，その変化をアプリケーションから隠蔽する．これにより，アプリケーションは個々のセンサを意識せずにセンサデータを取得できる．

以下に，本研究の目的を達成するために構築するミドルウェア MARS の機能要件を挙げる．

- **センサの多様性の吸収**
  ユビキタスコンピューティング環境には様々なセンサが存在する．利用するセンサの種類や機能，位置などのセンサの多様性をアプリケーションが全て把握するのは困難である．アプリケーションが個々のセンサを意識せずにセンサデータを取得できることでアプリケーションの負担を軽減できる．
- **センサ構成の変化への対応**
  センサが増減したり移動した場合でもセンサデータを提供するために，センサ構成の変化に対応してメタ情報を把握する必要がある．
- **アプリケーション要求の柔軟な記述力**
  アプリケーションのセンサデータの要求を容易にするため，メタ情報を利用し，要求を柔軟に記述できる必要がある．
- **センサの位置情報の把握**
  「ユーザの周囲」など，アプリケーションの位置に即した要求に対応するため，各センサのメタ情報の1つである位置情報を把握する必要がある．
- **センサデータの提供**
  アプリケーションは取得したセンサデータを平均

するなど，加工して利用することが想定される．MARS であらかじめ加工してから提供することで，アプリケーションの通信量や計算量を軽減することが可能である．

3 MARS の設計

アプリケーションが目的に沿ったセンサデータを利用するために，種類や位置などのセンサのメタ情報を基にした抽象的な要求が生じる．MARS では，センサのメタ情報を利用することにより，センサの多様性を吸収する．また，メタ情報を利用してアプリケーション要求を記述する．

3.1 MARS におけるメタ情報の定義

MARS では，アプリケーションがセンサデータを要求する際に，センサのメタ情報を利用する．本研究におけるメタ情報とは，センサに関するセンサデータ以外の情報である．以下に，本研究で扱うメタ情報を分類する．

- **不変メタ情報**
  取得できる環境情報の種類や分解能，サンプリングレートなど同じ製品なら全て等しく，また，環境に設置した後で変化しないメタ情報．製品毎に決まっているため，大量生産時にセンサに記憶させられる．
- **内部メタ情報**
  環境に設置するまで不明であったり，設置後に変化するメタ情報のうち，センサ内部に機能を持たせることにより，センサ自身で取得可能であるもの．指向性のあるセンサの向いている方角やセンサのバッテリの残量など，一つ一つのセンサで異なる．センサ設置後に動的に変化するため，各センサで取得，保持できることを想定する．
- **外部メタ情報**
  位置情報や環境に分散するセンサの密度，センサが付属しているものなど，センサ単体では取得できないメタ情報．センサの移動や追加に従って，動的に変化する．外部メタ情報は位置情報取得システムなど，センサ外部のシステムを利用して MARS が取得し，各センサには保持されない．

3.2 設計方針

以下の要件を満たすように MARS の設計を行なった．

センサのメタ情報の把握

センサの多様性に対応するため，環境に存在するセンサのメタ情報を把握する必要がある．MARS はメタ情報を把握することで各センサを管理し，アプリケーションの要求に対応するセンサの検索に利用する．アプリケーションが要求するセンサデータを取得するために，センサのメタ情報を利用した要求を受け付ける．

センサ構成の変化への対応

センサ設置後のメタ情報の変化を把握することで，センサ構成の変化に対応する．

センシングの対象物の指定

センサはアプリケーションの要求するセンシングの対象物に関する環境情報や対象物の周囲の環境情報，また，室内などの空間の環境情報を取得する．アプリケーションはメタ情報の 1 つとしてセンシングの対象物を指定し，MARS に対してセンサデータを要求する．センシングの対象物とセンサを同じ基準の位置情報で扱うために，MARS はセンサの位置情報や種類などのメタ情報を管理するメタ情報データベースを持つ．

センサデータの加工

アプリケーションの要求するセンサデータは，どのセンサで取得されたセンサデータかということは重要ではないため，全てのセンサデータをそのまま提供する必要があるとは限らない．センサで取得したセンサデータは，アプリケーションの要求によって平均などの加工をして提供することが必要である．MARS はアプリケーションの要求に従ってセンサデータを加工し提供する．

3.3 ハードウェア構成

MARS が想定するハードウェア構成を図 2 に示し，各ハードウェアの説明を行う．

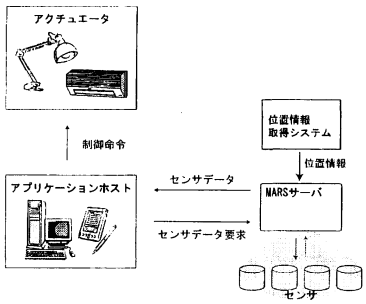

図 2: MARS ハードウェア環境

アプリケーションホスト

センサデータを利用するアプリケーションが動作するホストコンピュータである．アプリケーションは MARS からセンサデータを取得し，取得したセンサデータを利用してアクチュエータを操作する．

アクチュエータ

アプリケーションの制御命令により動作し，環境に働きかける．本研究では，情報家電などのネットワーク経由で操作可能な機器もアクチュエータに含める．

MARS サーバ

MARS サーバはアプリケーションの要求を受け付け，対象となるセンサにクエリを送信する．センサからセンサデータを受信し，アプリケーションに提供する．これらの事を実現するために，センサのメタ情報を取得，管理する．

センサ

MARS サーバのクエリにしたがってセンサデータを提供する．本研究において，センサは単一，または複数種類の環境情報を取得可能であることを想定する．クエリは，ノード ID やセンサの種類などを指定するものである．MARS サーバのクエリを解釈し，センサデータを加工するために計算機能を有する．センサは無線通信機能を有する．

位置情報取得システム

外部メタ情報取得システムの一つである．センシングの対象物とセンサの座標を取得する．MARS サーバの要求に従って，位置情報を提供する．外部メタ情報取得システムと MARS サーバは同一コンピュータ内で動作するか，広帯域の通信で接続され，高速な通信が可能であると想定する．

3.4 ソフトウェア構成

MARS サーバはメタ情報管理機構，アプリケーション要求解決機構，センサデータ提供機構，メタ情報管理データベース，および，センサとの通信を行うゲートウェイノードで構成される．また，MARS サーバは外部の位置情報取得システムを利用する．図 3 に，MARS のソフトウェア構成を示し，以下で各部の説明を行う．

メタ情報管理機構

環境に分散するセンサから，アプリケーションの要求に従ってセンサデータを取得するために，MARS はセンサのメタ情報を把握する．MARS 起動時，センサ設置時，MARS 動作時についてメタ情報の把握方法を考察した．

- **MARS 起動時**
  MARS 起動時にメタ情報管理機構はメタ情報要求メッセージをセンサに対してブロードキャストする．メタ情報要求メッセージを受信したセンサは，保持

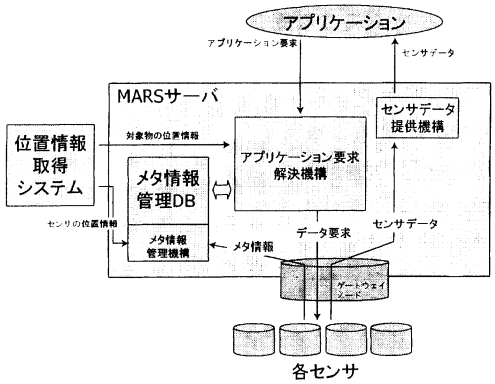

図 3: MARS ソフトウェア構成

している不変メタ情報および内部メタ情報を MARS サーバに送信する．メタ情報管理機構は受信したメタ情報をメタ情報管理データベースに登録し，位置情報システムなどから外部メタ情報を取得する．外部メタ情報取得システムは MARS 構築時に登録しておく．

- **センサ設置時**
  センサは起動時にメタ情報管理機構に対して不変メタ情報および内部メタ情報を送信する．また，メタ情報管理機構は受信したメタ情報をメタ情報管理データベースに登録し，各センサの外部メタ情報を取得する．
- **MARS 動作時**
  内部メタ情報と外部メタ情報は時間の経過とともに変化することが想定される．外部メタ情報取得システムは MARS サーバと高速な通信が可能であるため，メタ情報管理機構は定期的に外部メタ情報を取得する．センサは設置後定期的に内部メタ情報をメタ情報管理機構に送信する．これにより，MARS 起動時やセンサ設置時にメタ情報管理機構が受信できなかった場合でもセンサはメタ情報管理データベースに登録される．また，内部メタ情報が送信されなくなることにより，バッテリ不足や故障などによるセンサの減少をメタ情報管理機構が把握できる．

外部位置情報取得システム

MARS では，アプリケーションが要求するセンシングの対象物とセンサを同じ基準の位置情報で扱う必要がある．このため，MARS 外部の位置情報取得システムを利用する．MARS では位置情報取得システムは，1 つのホストに対して問い合わせることで全ての位置情報を取得する事ができる．送信機タグ型位置情報取得システムを利用する．また，MARS は外部の位置情報取得システムを利用して，センシングの対象物である機器の位置を取得する．

アプリケーション要求解決機構

アプリケーションからセンサのメタ情報による要求を受け付ける．アプリケーションの要求に該当するセンサのノード ID をメタ情報データベースから検索する．また，外部の位置情報取得システムを利用して，アプリケーションが要求するセンシングの対象物の位置を取得し，要求に対応するセンサを特定する．アプリケーション要求解決機構は，ゲートウェイノードを介してアプリケーション要求の対象となるノード ID のセンサに対して，クエリを送信する．

センサデータ提供機構

ゲートウェイノードがセンサから受信したセンサデータはセンサデータ提供機構に渡される．センサデータ提供機構では，受信したセンサデータをアプリケーションの要求する形式に変換してアプリケーションに提供する．例えば，センサデータを平均して提供することが考えられる．センサデータの変換手法は，あらかじめ API として平均や合計などを用意する．

センサ上で動作するソフトウェア

MARS が利用するセンサは，各センサ上でソフトウェアが動作し，無線通信によって MARS サーバと通信を行う．図 4 にセンサ上で動作するソフトウェアの構成を示す．

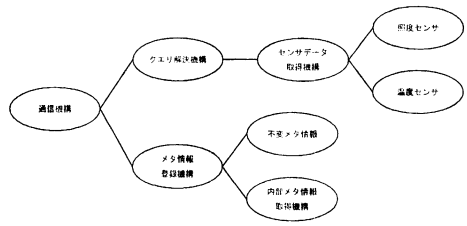

図 4: センサ上で動作するソフトウェアのコンポーネント関係図

センサを起動するとメタ情報登録機構は不変メタ情報と内部メタ情報をゲートウェイノードに送信する．また，内部メタ情報は定期的に取得，送信し，メタ情報管理データベースのメタ情報を更新する．MARS サーバからクエリを受信すると，クエリ解決機構はクエリにより指定されたセンサを判別し，センサデータ取得機構からセンサデータを取得する．センサデータ取得機構では，センサや A/D コンバータを利用してセンサデータを取得する．

4 MARS の実装

本研究では MARS のプロトタイプを実装し，慶應義塾大学 SFC Open Research Forum 2003[6] において，u-Photo[7] のデモンストレーションで使用した．本節では，MARS のプロトタイプ実装について述べる．

4.1 実装環境

MARS は様々な環境で，環境に設置されたセンサからセンサデータを取得するため，動作するハードウェアや OS も多様であると考えられる．そこで，システムのプラットフォーム独立性を確保するために，Java 言語を用いて実装した．

また，本プロトタイプ実装ではセンサとして，mote[8] のセンサノードの 1 つである mica2 を利用し，温度と照度を取得した．mica2 は各センサノードで複数のセンサによるセンサデータの取得，CPU による演算，無線による通信を行うことが可能であり，本研究におけるセンサの想定を満たす．

mica2 は 128KB のプログラムメモリにソフトウェアを書き込んで動作させる．mica2 上で動作するソフトウェアは nesc を用いて実装した．nesc は TinyOS Project[2] で開発されている mote で動作するソフトウェアを記述するオブジェクト指向言語である．

4.2 MARS サーバの実装

本項では，MARS サーバの実装について述べる．

メタ情報管理機構

MARS はセンサのメタ情報をメタ情報データベースに登録する．メタ情報データベースのテーブルには，ノード ID，センサの種類，座標，付属物のデバイス ID という項目を用意した．

アプリケーション要求解決機構

本プロトタイプは，センサの位置と種類というメタ情報を基にしたアプリケーション要求に対してセンサデータを提供した．対象物や空間に関する要求を受け付け，センサにクエリを送信し，取得したセンサデータを返す．表 1 に，本プロトタイプで用意した API を示す．

表 1: アプリケーション要求 API

メソッド	説明
double getRounddeviceData()	センサの種類，対象物の ID，半径から対象物周囲のセンサデータを取得する．
double getSphereData()	センサの種類，中心座標，半径から球形空間の内側のセンサデータを取得する．
double getCubeData()	センサの種類，始点，終点から一取得システムの座標軸に平行な立方体の内側のセンサデータを取得する．
Vector getNodeID()	センサの種類，センサデータの上限，下限から一定範囲の値を示すセンサのノード ID を取得する．
double getAllData()	一種類のセンサに関して，すべてのセンサからセンサデータを取得する．

センサデータ提供機構

本論文における実装では，対象物の周囲のセンサで取得したセンサデータを，アプリケーションの要求に従って，平均値や最大値，最小値に加工して提供した．

4.3 センサの実装

センサは，ゲートウェイノードから受信したセンシング命令に従い，センサデータをゲートウェイノードに送信する．

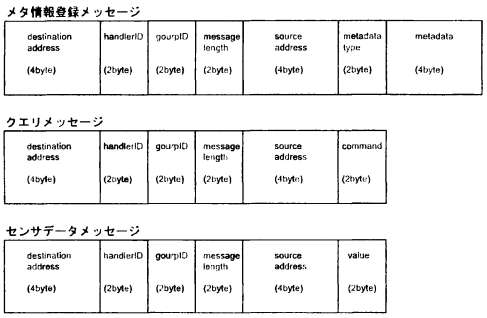

図 5: センサ間の通信

センサとゲートウェイノード間の通信の形式を，図5に示す．センサは 4byte のノード ID および，2byte のグループ ID を持つ．handlerID はメッセージの種類を示し，メタ情報登録メッセージを 0，クエリメッセージを 1，センサデータメッセージを 2 とした．metadata type はメタ情報の種類を示し，metadata は具体的なメタ情報を示す．例えばセンサの種類というメタ情報は metadata type を 0 とし，温度センサは metadata を 0，明るさセンサは metadata を 1 とした．command はセンサデータ取得命令を示し，温度の取得を 0，明るさの取得を 1 とした．

通信機構からクエリを受信すると，クエリ解決機構がセンサデータ取得機構に対してセンサデータ取得命令を行う．センサデータ取得機構は A/D コンバータの値を読む．

5 関連研究

本節では本研究の関連研究ついて考察する．

5.1 TinyDB

TinyDB[9] はセンサネットワークからデータを抽出する研究である．センサネットワークから，宣言型クエリを記述することによりセンサデータを抽出するシステム TAG[9] が mote 上で実装されている．

TinyDB では，ある値を示した 1 つのセンサとの距離が 10m というように，あるセンサの周囲の空間というセンサデータの要求が可能である．しかし，機器やユーザなどの対象物の周囲という空間のセンサデータを要求する場合，対象物とそれに対応するセンサのバインドが必要である．しかし，TinyDB では具体的な実現方法が述べられていない．TinyDB は各センサがアプリケーションの要求に対応するため，各センサが自身と様々な機器の位置を保持する方法では，スケーラビリティの点で問題がある．また，TinyDB ではセンサのメタ情報を全て自身が管理し，温度センサや照度センサのサンプリングに必要な消費電力やサンプルレートなどを定期的にゲートウェイに送信している．しかし，位置情報をメタ情報として扱うことには言及されていない．

5.2 Cougar Approach

Cougar Approach[10] はセンサネットワークからセンサデータを抽出する研究である．センサネットワークをデータベースとして扱い，宣言型クエリを用いてセンサデータを抽出する．

宣言型言語を利用することで，ユーザやアプリケーションがセンサネットワークの構成や，データの処理過程を知らなくて済む．また，各センサの持つ計算機能でセンサデータをまとめ，不要なデータを除去する．

センサのメタ情報を定期的にゲートウェイに送信するなどして，ゲートウェイが管理することにより，センサに送信するクエリの最適化を行うことが述べられている．そして，メタ情報のデータサイズやセンサ構成の動的性により，ゲートウェイで管理すべきメタ情報とセンサで管理すべきメタ情報を分ける必要があると述べられている．しかし，具体的にどのメタ情報をゲートウェイで管理し，どのメタ情報をセンサで管理するのがよいかということは述べられていない．また，センサデータを要求する際に，センシングの対象物の周囲という要求を行うことは述べられていない．

6 評価

本節では，MARS の定性的評価と定量的評価について述べる．

6.1 定性的評価

MARS の定性的評価として MARS と関連研究の比較を行う．

位置情報の把握

MARS では，超音波発振機タグを用いた位置情報取得システムのような既存の主な屋内位置情報取得システムを用いて各センサの位置を取得する事ができる．このような位置情報取得システムでは，位置情報取得の対象となるタグを安価で利用できるが，タグの位置を一箇所の基地局で計算しなければならない．TinyDB では各センサでアプリケーションの要求を解決するため，各センサ自身が位置情報を把握する必要があり，位置情報取得システムの基地局へ問い合わせる必要がある．このため，現在このような位置情報取得システムを利用される事の多い屋内では不適である．

システムの更新

ユビキタスコンピューティング環境において，センサデータを利用するアプリケーションは様々である．アプリケーション API として準備されていないセンサデータの加工を要求されるなど，アプリケーションがセンサデータ取得システムやセンサを設置する時点では想定していない要求を持つことが考えられる．こ

のような状況に対応するためには，各センサでアプリケーション要求を解決し，加工するTinyDBやCougar Approachでは，センサ上で動作するソフトウェアを更新する必要がある．しかし，環境に多数のセンサが設置されている場合，それらのセンサ全てのソフトウェアを更新することは困難である．MARSでは，アプリケーションの要求をサーバで解決し，全てのセンサデータを一度サーバに集約するため，各センサ上のソフトウェアを更新することなく，サーバ上のAPIを更新することで対応できる．

各センサのコスト

MARSではアプリケーション要求をセンサ上で解決しないため，TinyDBやCougar Approachと比べ，各センサに要求される能力は低い．このため，MARSで使用するセンサを製作する際のコストはTinyDBやCougar Approachと比べて低く抑えられる．

6.2 定量的評価

MARSがアプリケーションから要求を受信してから，センサデータを提供するまでの時間を計測した．センサが1個から8個まで増えた時にセンサデータを提供するまでの時間がどのように増加するかを観察した．計測の結果を図6に示す．

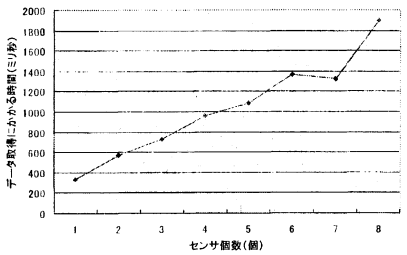

図6: 測定結果

本測定により，センサの個数が増加するに連れて，センサデータを提供するまでの時間が増加していることがわかる．平均2秒以内で，8個のセンサからセンサデータを取得し，アプリケーションに提供できた．ユーザの周囲の温度情報を利用した空調や，ユーザが睡眠中であるというコンテキストを抽出し，利用するようなアプリケーションにおいては，実用的に利用可能である．

7 まとめと今後の課題

本論文では，センサのメタ情報を利用してセンサデータを取得するミドルウェアMARSを設計，実装し，評価を行なった．MARSを利用することで，アプリケーションは環境に設置されたセンサのうち，コンテキストを抽出する対象物やアクチュエータにより制御する対象物の周囲に存在するセンサや対象物に付属するセンサからセンサデータを取得できる．

ユビキタスコンピューティング環境では，コンテキストアウェアアプリケーションが，コンテキストを抽出するために，対象物に関するセンサデータを取得する事が必要である．様々なコンテキストアウェアアプリケーションが，様々な対象物のコンテキストを抽出するため，ユビキタスコンピューティング環境では，アプリケーションとセンサが動的に接続する．しかし，環境に設置されるセンサは多様であり，アプリケーションの持つセンサデータに対する要求は抽象的である．このため，多様なセンサからアプリケーションの要求するセンサデータを取得するためのミドルウェアが重要となる．本研究では，センサのメタ情報を管理することにより，アプリケーションの要求を解決し，センサデータを提供できた．

MARSはサーバ上でアプリケーション要求の解決，センサデータの加工を行なっている．このため，アプリケーションAPIを用意する事で，アプリケーションの高度な要求や様々な要求を解決する事が可能である．今後，アプリケーションの要求を考察し，MARSに対するアプリケーション要求の記述力を向上させる．

本論文で行なったプロトタイプ実装では，センサデータは平均，最大値，最小値をアプリケーションに提供することが可能である．しかし，アプリケーションの要求するセンサデータの加工は，平均や最大値，最小値だけではなく，極端なエラー値を除去して平均することなどが考えられる．また，アプリケーションの要求によっては，各センサで取得したセンサデータとセンサの位置情報により，環境情報のマップを生成するように，全てのセンサデータを加工せずに提供する必要がある．今後，実際にコンテキストアウェアアプリケーションや環境モニタリングアプリケーションによりMARSを利用し，アプリケーションの要求するセンサデータの加工方法を考察する．

参考文献

[1] Weiser, M.: The Computer for the Twenty-First Century, *Scientific American*, pp. 94–10 (1991).

[2] Hill, J., Szewczyk, R., Woo, A., Hollar, S. and Pister, D. C. K.: System architecture directions for networked sensors, *ASPLOS 2000*, pp. 93–104 (2000).

[3] Joshua Lifton, Deva Seetharam, M. B. and Paradiso, J.: Pushpin Computing System Overview: A Platform for Distributed, Embedded, Ubiquitous Sensor Networks, *Pervasive 2002*, pp. 139–151 (2002).

[4] J. M. Kahn, R. H. K. and Pister, K. S. J.: Mobile Networking for Smart Dust, *ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom 99)*, pp. 17–19 (1999).

[5] Holmquist, L., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M. and Gellersen, H.: Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts, *Proc. of UBICOMP 2001*, p. September (2001).

[6] 慶應義塾大学: SFC Open Research Forum 2003. http://orf.sfc.keio.ac.jp/.

[7] Kohtake, N., Iwamoto, T., Suzuki, G., Aoki, S., Marudai, D., Kouda, T., Takashio, K. and Tokuda, H.: u-Photo: A Snapshot-based Interaction Technique for Ubiquitous Embedded Information, *Proceedings of The Second International Conference on Pervasive Computing (Pervasive2004)Advances in Pervasive Computing*, Vol. ISBN 3-85403-176-9, pp. **–** (2004).

[8] J., H. and D., C.: A wireless embedded sensor archtecture for system-level optimization, *In UC Berkeley Technical Report*.

[9] Maddern, S., Franklin, M. J., Hellerstein, J. M. and Hong, W.: TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks, *Operating Systems Design and Implementation* (2002).

[10] Bonnet, P., Gehrke, J. E. and Seshadri, P.: Querying the Physical World, *IEEE Personal Communications*, pp. 10–15 (2000). Special Issue on Smart Spaces and Environments.