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

2005－MBL－32（12）
2005－UBI－　7（12）
2005／3／17

センサノードのメタ情報を利用したワールドモデル構築支援

齋藤信貴 † 高汐一紀 †††
林貴宏 †† 尾内理紀夫 †,††

ユビキタスコンピューティングの発達により，状況に応じて自律的にサービスが提供される環境が実現する．サービスを提供するアプリケーションは，空間内に存在する無数のセンサの中からサービスに適した観測領域を持つ適切な種類のセンサを発見し，環境情報を取得する必要がある．そのため，アプリケーションが空間の形状と空間内のセンサ構成を把握するための枠組が必要となる．実空間の情報を管理する手法として，コンピュータシステム上に実空間を模したワールドモデルを構築し，利用する手法がある．本稿では，空間と空間内のセンサノードの情報を自動的に取得し，実空間の形状とセンサの配置，観測範囲を模したワールドモデルを構築し，アプリケーションに対してそれらの情報を提供するシステムについて報告を行う．

World Model Construction using Sensor Node Metadata

Nobutaka SAITO ,† Kazunori TAKASHIO ,†††
Takahiro HAYASHI †† and Rikio ONAI †,††

The growth of Ubiquitous Computing makes the novel environment that provides context-aware services. Context-aware service applications have to discover sensors sensing proper context and having proper scopes, and get context data from the sensors. Therefore a new framework for decribing applications to grasp information of rooms and sensors is needed. One of ways to manage information of real world is using a world model constructed on computer system. This paper presents a world model construction system that provides 3D-shapes of room, location of sensors and sensor scopes for service applications. The system automatically constructs a world model acquiring and using information about rooms and sensors.

1. はじめに

ユビキタスコンピューティング[1]の発達は，ヒトやモノ，空間の状況に応じてサービスを自律的に提供する環境を実現する．

現在でも，自動ドアの開閉や空調設備の自動温度調節など，センサを利用し，状況に応じて我々の生活を補助するサービスは数多く存在する．しかし，これらのサービスを行う機器内では，センサと，センサが観測した情報を利用して機器を駆動させるアプリケーションが固定的に接続されている (**図 1**-a)．このため，空調設備内の適温判断アプリケーションが他の機器内の温度センサの値を使って計算精度を高めたり，自動ドアの人体検知センサを防犯アプリケーションのために利用したりすることは困難である．

これに対し，ユビキタスコンピューティング環境では，無線通信機能と演算機能をもったセンサノード機器[2][3][4]の実現により，センサインフラとアプリケーションの分離が可能となる (図 1-b)．センサインフラとアプリケーションが分離されることにより，一つのアプリケーションが複数のセンサを利用してサービスの質を高めたり，複数のアプリケーションが一つのセンサを共有することが可能になる．また，センサノードは建材や家具，家電などのあらゆるオブジェクトに埋め込まれ，空間内に遍在するようになると考えられており，様々な位置の様々な環境情報を観測することが可能となる．

一方，アプリケーションは，空間内に存在している無数のセンサの中から，サービスを実現するために利用価値のある環境情報を観測しているセンサを発見する必要がある．例えば，空調設備内の適温判断アプリケーションは部屋の四隅を観測する温度センサを発見して，それらの観測データを利用したり，自動照明ア

† 電気通信大学大学院 電気通信学研究科
Graduate School of Electro-Communications, University of Electro-Communications
†† 電気通信大学 電気通信学部
Faculty of Electro-Communications, University of Electro-Communications
††† 慶應義塾大学大学院 政策・メディア研究科
Graduate School of Media and Governance, Keio University

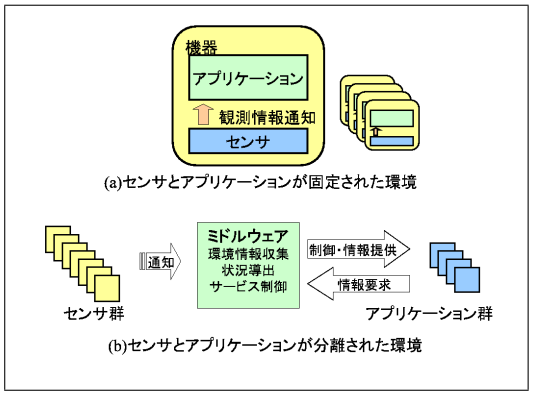

図 1 環境によるサービス形態の違い

プリケーションは部屋の天井中央部を観測する照度センサを発見して，その観測データを利用したりする．アプリケーションが具体的に観測点や種類を指定して環境情報を取得するには，空間の形状情報を取得したり，空間内に存在するセンサノードの位置や観測範囲，種類などの情報を取得できる枠組が必要となる．

本研究では，コンピュータシステム上に実空間の形状とセンサ構成を模したワールドモデルを自動構築し，空間とセンサの情報を提供するシステムを構築する．ワールドモデルを利用することで，空間の形状やセンサの位置，観測範囲などの立体的なデータの表現・管理が容易となる．アプリケーションは本システムが提供する情報取得 API を利用することにより，空間の形状情報，センサの位置や種類，観測範囲といった情報を把握し，サービス提供のために利用価値のある任意の位置の環境情報を得ることが可能となる．

本稿では，第 2 節で本研究の論点について述べ，第 3 節で本研究で構築するシステムの設計について，第 4 節で実装について述べる．第 5 節でシステムの評価を行い，第 6 節で関連研究と本研究の差異について説明する．第 7 節で結論を述べる．

2. 本研究の論点

2.1 センサノード情報の収集

アプリケーションやサービス提供者が利用するセンサノードの情報は，センサノードに搭載されたセンサデバイスの観測範囲，種類，観測データである．センサデバイスの観測範囲はセンサノードの位置を起点とした球や円柱，円錐といった形状で表される．また，観測範囲の傾斜はセンサノード本体の傾斜とセンサノード上でのセンサデバイスの設置傾斜によって決定される．

ユビキタスコンピューティング環境では，センサノードがあらゆるオブジェクトに埋め込まれた状態で無数に存在するため，これらのセンサノードごとに固有の情報を人手で確認するのは非常に困難である．また，オブジェクトの移動に伴って，そのオブジェクトに埋め込まれたセンサノードの位置や向きが変化することが想定され，これもまた人手で逐一確認するのは困難である．したがって，センサノードの情報収集を自動化する必要がある．

2.2 空間情報・センサノード情報の管理

ユビキタスコンピューティング環境におけるアプリケーションは，空間の形状や空間に存在するセンサノード情報を把握し，特定の位置を観測点とするセンサの情報を求める．空間やセンサの観測範囲の形状，大きさ，向きといった情報は立体的に構成される情報である．本システムでは，これらの立体的な情報をコンピュータシステム内で表現し，管理する必要がある．

2.3 空間情報・センサノード情報の提供

サービスアプリケーションに対し，特定の種類の環境情報を観測するセンサノードや，特定の位置を観測するセンサノードを容易に発見する手段を提供する必要がある．また，センサノードが存在している空間そのものの形状や，空間に配置されているテーブルなどのオブジェクトの形状や位置といった情報も提供する必要がある．

3. 設　　計

3.1 センサノード情報収集の自動化

センサノードの位置，種類などを自動的に取得する手法として，センサノードに自身の情報を管理・発信する機能を持たせ，センサノードが随時発信する情報を収集する手法がある[5]．本研究でも，この手法を用いてセンサノード情報収集の自動化を実現する．

本研究では，識別名，種類，位置，観測範囲といった，センサノード固有の情報をセンサノードのメタ情報と呼ぶ．

センサノードは，複数のセンサを搭載している場合や，同じ種類のセンサを搭載していても精度が異なる場合があるなど，仕様が多様である．また，センサノードが自身の位置情報を取得できない場合，外部の位置情報取得システムにセンサノードの位置の測位を依頼する必要がある．センサノードがメタ情報を整理，通信するために用いるデータ形式は以上のような特徴に対応し，柔軟に情報を表現できるものである必要がある．本研究では，これを解決するためにメタ情報のデータ形式として XML を採用し，次の概念をもつデータ形式を TinySensorML☆として定義する．

- センサノードのメタ情報を含む
- センサデバイスが観測した環境情報を含む
- 外部情報システムに解決を依頼するメタ情報について明示が可能
- 構造が簡素，データサイズが小さい

☆ SensorML[6] とは異なる構造を持つ．

また，TinySensorML は以下の情報を表現可能とする．

- センサノードの ID
- TinySensorML の情報の有効時間
- センサノードの位置座標
- センサノードの傾斜
- センサノードが所属する空間のモデル情報
- 搭載するセンサデバイスの情報 (複数記述可)
  - センサデバイスの ID
  - センサデバイスの観測範囲の形状，サイズ
  - センサデバイスの観測範囲の軸線の センサノードに対する傾斜
  - センサデバイスの取得データ名，データ型，単位
  - センサデバイスの取得データ値

センサノードの位置やセンサノードが所属する空間のモデル情報はセンサノード自身で解決できない場合がある．例えば，センサノード自身に位置情報センサが搭載されていない場合である．そこで，メタ情報の解決を外部のサーバに依頼することを明示できる仕様とする．

メタ情報は情報の解決方法によって次のように分類する．

- 内部依存メタ情報
  センサノード内で取得できる情報である．明示する際は，TinySensorML 内の当該タグに値を記述する．
- 外部情報システム依存メタ情報
  センサノード内で取得できない情報で，外部のシステムを用いて取得する情報である．明示する際は，TinySensorML 内の当該タグにおいて URL を用いて情報の参照先を記述する．
- システム依存メタ情報
  センサノード内で取得できない情報で，TinySensorML を受信した本システムに取得を依頼する情報である．システム規定の情報解決手法を用いて情報を参照する．明示する際は，TinySensorML 内の当該タグを空タグにする．

ここで，TinySensorML の要素を詳細に述べる．オブジェクトの形状，位置，傾斜といった立体的な情報の表現には，全て右手座標系を用いる．右手座標系において，X 軸の正方向を東，Y 軸の正方向を上方，Z 軸の正方向を南と規定する．

- sensornode
  以下に示す要素を全て包含する．センサノードの識別名を示す name 属性と，包含する情報の有効時間（秒）を示す ttl 属性を持つ．
- room
  センサノードが存在する空間のモデル情報を示す．モデル情報は XML でオブジェクトの形状を表現する X3D[7] 形式で示される．ref 属性に URL を指定し，外部情報システムに情報の解決を依頼することができる．また，システム依存メタ情報とすることも可能である．
- position
  センサノードの位置情報を示す．room 要素で示した空間モデル内での位置を三次元座標 (メートル) で示す．ref 属性に URL を指定し，外部情報システムに情報の解決を依頼することができる．また，システム依存メタ情報とすることも可能である．
- tilt
  センサノード本体の傾斜情報を示す．room 要素で示した空間モデル内でのセンサノードの X 軸まわりの傾斜（ディグリー），Y 軸まわりの傾斜（ディグリー），Z 軸まわりの傾斜（ディグリー）を，それぞれ pitch 属性，roll 属性，yaw 属性で示す．ref 属性に URL を指定し，外部情報システムに情報の解決を依頼することができる．また，システム依存メタ情報とすることも可能である．
- sensor
  センサノードに搭載されたセンサデバイスの情報を示す．センサノードにセンサデバイスが複数搭載されている場合は，sensor 要素が複数存在しても良い．scope，data 要素を包含する．センサデバイスの識別名を name 属性で示す．
- scope
  センサデバイスの観測範囲についての情報を示す．センサデバイスの観測範囲は，球，円柱，円錐といった形状に簡素化して表現する．これを，それぞれキーワード sphere，cylinder，cone を用いて shape 属性で示す．また，形状が球の場合は半径（メートル）を radius 属性で示す．形状が円柱，円錐の場合は半径（メートル）と高さ（メートル）を radius 属性，height 属性で示す．センサノード本体に対する観測範囲の軸線の X 軸まわりの傾斜（ディグリー），Y 軸まわりの傾斜（ディグリー），Z 軸まわりの傾斜（ディグリー）を，それぞれ pitch 属性，roll 属性，yaw 属性で示す．
- data
  センサデバイスが観測するデータについての情報を示す．要素として，観測したデータ値を持つ．brightness，temperature といった具体的なデータの種類を name 属性で示す．string，integer，float といったデータ値の型名を type 属性で示す．また，データ値の精度や単位を measure 属性で示す．

センサノードが実際に送信する TinySensorML の例を**図 2** に示す．

本研究で構築するシステムで利用するセンサノードは，全て TinySensorML を通信データ形式として用いる．

3.2 空間情報・センサノード情報の管理

本システムにおいて立体的に構成される要素を次に挙げる．

```
<?xml version="1.0"?>
<tinysensorml>
  <sensornode name="S000" ttl="200">
    <room/>
    <position/>
    <tilt pitch="30.0" roll="20.0" yaw="20.0"/>
    <sensor name="LightSensor">
      <scope shape="cylinder" radius="0.2" height="0.2" pitch="90.0" roll="0.0" yaw="0.0"/>
      <data name="brightness" type="integer" measure="lx">600</data>
    </sensor>
    <sensor name="ProximitySensor">
      <scope shape="sphere" radius="50" pitch="90.0" roll="0.0" yaw="0.0"/>
      <data name="id" type="string">Kenj/data>
      <data name="distance" type="integer" measure="cm">50</data>
    </sensor>
  </sensornode>
</tinysensorml>
```

図 2　TinySensorML の記述例

- センサノードが存在する空間
- 空間に存在するオブジェクト
- 空間に存在するセンサとその観測範囲

特定の空間とその空間内のオブジェクトの位置や形状を，コンピュータシステム上で容易に管理する手法として，ワールドモデルを用いた手法がある[8]．ワールドモデルとは実空間をコンピュータシステム上に模したものであり，実空間上のオブジェクトが，ワールドモデル内でも正確な形状で，正確な位置に配置される．そこで，本研究でも収集したセンサノードのメタ情報からワールドモデルを構築し，これを先に述べた情報の管理手法として用いる．

本研究では，実空間の形状とその空間内のオブジェクトの位置や形状を模したモデルと，センサノードの位置を表すシンボル，センサの観測範囲の形状を模したモデルから構成されるものをワールドモデルと呼ぶ．

3.3　空間情報・センサノード情報の提供

本システムでは，管理対象である空間情報，およびセンサノード情報をアプリケーションに提供する機構を備える．単に空間やセンサの位置，観測範囲の形状を提供するだけでなく，指定された種類のセンサや，指定された観測点を観測範囲に含むセンサを検索する機能も提供する．これらは，本システムがあらかじめ API として備える．

3.4　システム構成

本システムの全体像を**図 3** に示す．

3.4.1　センサノード

センサノードは自身のメタ情報と観測した環境情報を TinySensorML として整形し，対センサノード通信部に対し送信する．

3.4.2　外部情報システム

外部情報システムは，センサノードが単独で解決できないメタ情報を補完する．メタ情報解析部からの問い合わせにより，情報の提供を行う．

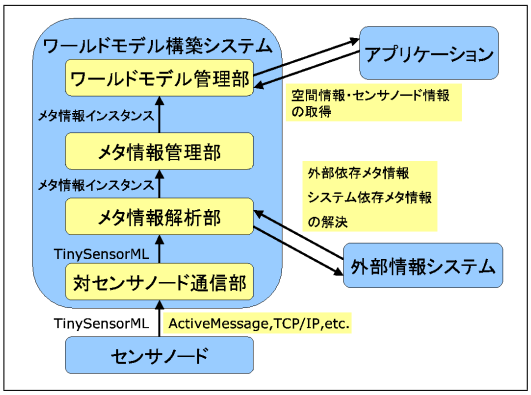

図 3　システムの全体像

3.4.3　ワールドモデル構築システム

- 対センサノード通信部

対センサノード通信部は，センサノードが送信する TinySensorML を受信する．多様なセンサノードに対応するため，複数の通信プロトコルを備える．

- メタ情報解析部

メタ情報解析部は，TinySensorML を解釈し，システム内部で管理するためのメタ情報インスタンスに変換する．このとき，外部情報システムに対して問い合わせを行い，外部依存メタ情報やシステム依存メタ情報の解決処理を行う．

- メタ情報管理部

メタ情報管理部は，メタ情報解析部から取得したセンサノードのメタ情報インスタンスを管理する．ワールドモデル管理部に対し，センサノードのメタ情報を提供する．また，各メタ情報の TTL 情報を監視し，無効となったメタ情報インスタンスを廃棄する．

- ワールドモデル管理部

ワールドモデル管理部は，メタ情報管理部から取得

したセンサノードのメタ情報をもとに，ワールドモデルを構築，保持する．また，構築したワールドモデルの情報やワールドモデル内のセンサノード情報をアプリケーションに提供する API を備える．

4. 実　　装

4.1 センサノード

本システムの実装では，センサノードとして mote[2] シリーズの一つである mica2 を利用した．mica2 は，温度，加速度，照度，磁気，音を観測することが可能であり，演算機能と無線通信機能を有する．また，プログラムメモリに任意のソフトウェアを書き込むことが可能である．

本実装では，5000 ミリ秒毎に環境情報を観測し，観測した値をセンサノードのメタ情報と共に TinySensorML として整形し，ブロードキャストするソフトウェアを書き込んだ．ブロードキャストされるデータは ActiveMessage プロトコルを介して，本システムが動作する PC に接続されたゲートウェイに通知される．ActiveMessage プロトコルは mote 独自の通信プロトコルであり，通信成功のチェックや再送処理はなく，1 パケットが最大 255 バイト長といった特徴をもつ．本実装では，TinySensorML が 255 バイトを超えた場合，複数パケットに分割して送信する．このソフトウェアは nesC を用いて実装した．nesC は mica2 で動作するソフトウェアを記述するための，コンポーネントベースのプログラミング言語である．

観測範囲の傾斜を求める処理では，加速度センサを傾斜センサとして利用した．また，mica2 は位置情報を取得する機能を持たないため，4.2 節で述べる超音波位置情報センサの送信機タグを mica2 上に設置して利用した．

4.2 外部情報システム

本実装では，センサノード mica2 の位置情報を取得する手段として，超音波位置情報センサ IS-600Mk2[9] を利用した外部位置情報取得システムを構築した．IS-600Mk2 は，mica2 上に設置された送信機タグの三次元位置座標を取得する．TinySensorML において外部依存メタ情報，またはシステム依存メタ情報として定義されたメタ情報は URL を用いて参照される．本実装では，送信機タグに与えられる識別名とセンサノードの識別名を便宜的に同一のものとし，Apache1.3.31 と PHP4.3.8 を用いて任意のセンサノードの位置情報を取得可能とした．

また，センサノードが存在する空間モデルを取得する手段として，空間モデル取得システムを構築した．本システムでは，空間モデル情報として，XML でオブジェクトの形状データを表現する X3D[7] 形式を利用する．本実装では，Apache1.3.31 と PHP4.3.8 を用いて任意の空間のモデル情報を取得可能とした．

4.3 対センサノード通信部

対センサノード通信部は Java2SE5.0 を用いて実装した．センサノード mica2 との通信処理には net.tinyos.*パッケージを利用した．このパッケージは mote 開発キットに含まれているものである．また，多様なセンサノードに対応するため，対応通信プロトコルを容易に追加可能とするインターフェイスを実装した．これを利用して TCP/IP によるセンサノードとの通信を可能とした．

4.4 メタ情報管理部・メタ情報解析部

メタ情報管理部及び，メタ情報解析部は Java2SE5.0 を用いて実装した．

4.5 ワールドモデル管理部

ワールドモデル管理部は，Java2SE5.0，Java3D SDK 1.3.1，Xj3D M9 を用いて実装した．Java3D は Java 言語を用いて仮想空間のモデルを構築可能とする開発キットである．Java3D を用いることで，仮想空間モデルの可視化も容易に行うことができる．本実装では，本システムで扱うワールドモデルを表現するデータモデルとして，Java3D のシーングラフ形式を利用した．Xj3D は，X3D 形式のオブジェクト形状データを Java3D のシーングラフ形式に変換するためのパッケージである．アプリケーションに対して空間情報，及び空間内のセンサノード情報を提供するために実装した API の一部を**表 1** に示す．アプリケーション内で実際に情報取得 API を利用したコードの例を**図 4** に示す．

5. 評　　価

本システムの評価実験は，慶応大学 SFC 徳田研究室 SSLab を実験室として行った．システムとアプリケーションを稼動させたホストのスペックは PentiumIII600MHz，196MBRAM，Windows2000，Java2SE5.0 である．センサノードとしては，mica2 と，シミュレーション用に実装した仮想センサノードを併用した．仮想センサノードは，TCP/IP を介して任意の TinySensorML をシステムに対して送信するソフトウェアである．これは Java2SE5.0 を用いて実装した．ここでは，アプリケーション構築の際の本システムの記述性と，本システムのパフォーマンスについて評価を行った．

5.1 記 述 性

本システムが持つ，センサノードから収集した情報から自動的にワールドモデルを構築する機能と，アプリケーションに対し空間と空間内のセンサノード情報を提供する機能について評価を行った．評価実験を行うために，本システムが提供する API を利用するアプリケーションを実装した．実装したアプリケーションはワールドモデルを視覚化し，サービス提供者がセンサノードの情報を容易に把握するための機能をもつ

表 1 主要な情報提供 API

返り値	メソッド名	引数	説明
List<String>	getWorldNames()	-	空間名一覧を得る
World	getWorld()	空間名	ワールドモデル情報を得る
List<SensorNode>	getSensorNodesByDataName()	環境情報の名称	センサ情報を得る
List<SensorNode>	getSensorNodesByTarget()	座標 (x, y, z)	指定座標を観測範囲に含むセンサ情報を得る

（*）World, SensorNode は Java3D の BranchGroup サブクラス

```
// 構築済みの全てのワールドモデル名を得る
for( String worldName : getWorldNames() ) {
  // 指定した識別名のワールドモデルを得る
  World world = getWorld( worldName );
  // ワールドモデルの中心座標を得る
  // 空間を構成する三次元オブジェクトの
  // 座標情報から任意の点を算出することも可能
  Point3f center = world.getCenterPoint();
    // ワールドモデル内の中心を
    // 観測するセンサノード情報を得る
    for( SensorNode node :
    world.getSensorNodesByTarget( center ) ) {
      // センサノードのセンサデバイス情報を得る
      for( Sensor sensor = node.getSensors() ) {
        // センサデバイスの観測情報を得る
        for( Data data = sensor.getDatas() ) {
          String nodeName = node.getName();
          String sensorName = sensor.getName();
          String dataName = data.getName();
          String dataType = data.getType();
          String dataMeasure = data.getMeasure();
          String dataValue = data.getValue();
      }
    }
  }
}
```

図 4 情報取得 API を利用したコーディング例

ビューアである．これらの機能は空間情報，及びセンサノード情報の取得 API を利用して実現した．まず，センサノードを実験室に設置したのち，ビューアを起動した．すると，ビューアの GUI パネル上に実験室を模した三次元モデルが表示され，実空間に存在するセンサノードの位置を示すシンボル及び観測範囲が表示された (**図 5**)．ビューア上の直方体は部屋の形状を，小球はセンサノードの位置を，大きな球や円錐，円柱は各センサの観測範囲を示す．手作業で室内を調査することなく，室内に存在するセンサノードを確認することが可能であった．また，センサノードが観測している環境情報をリアルタイムで確認することが可能であった．続いて，センサノードの移動，向きの変更を行った．これも GUI パネル上にて，センサノードの実空間での変化を反映して表示が更新された．新たなセンサノードの導入，既存のセンサノードの排除についても，導入したセンサノードの情報は即座に本システムの GUI パネルに表示され，排除したセンサノードは GUI パネル上から消失した．

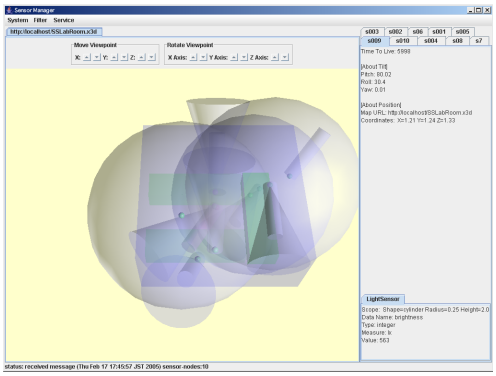

図 5 ワールドモデルのビューアアプリケーション

これらのアプリケーションの動作結果より，空間と空間内のセンサノード情報を自動的に収集してワールドモデルを構築することと，アプリケーションに対し情報を提供する機能が実現されていることを確認できた．

5.2 定量評価

本システムのスケーラビリティを評価した．評価方法としては，ターゲット空間のセンサノード数を増加させ，本システムが提供するセンサノード検索 API が検索結果を返すまでに要する時間を計測した．なお，センサノードとしては仮想センサノードを利用した．時間の計測には，システム時間を求めるメソッド☆を利用した．

センサノードの数量を横軸，検索 API が結果を返すまでに要した時間を縦軸とするグラフを**図 6** に示す．

本測定により，センサノードの個数の増加に伴い，センサノードの検索 API が処理を完了するまでに要する時間が増加していることがわかる．センサノードが 100 個増加すると，処理時間が約 50 ミリ秒増加している．これは，空調設備内の適温設定アプリケーションが，部屋中の温度センサを探索し，それらのセンサの観測値を利用するなどの場合において，十分実用的な性能である．

6. 関連研究

MARS[5] は，アプリケーションが所望の種類のセンサの観測データを取得したり，所望の位置座標にある

☆ System.currentTimeMillis()

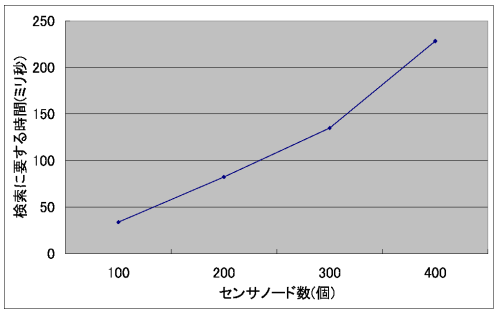

図 6 ノード数と検索 API 完了時間

センサの観測データを取得したりするための機能を実現するシステムである．この機能は，センサノードがそれぞれ保持している種類や位置といったメタ情報をサーバが収集し，アプリケーションが所望のデータを要求する際にメタ情報を条件として指定することで実現される．センサノードのメタ情報のサイズはハードウェア上の制限から 4 バイトと定めているため，扱う情報は数値情報に限られる．また，センサの種類を表すメタ情報が 0 であれば温度センサを意味し，1 であれば照度センサを意味するといった対応表をあらかじめ定義しておく必要がある．センサノードのメタ情報の表現力や柔軟性が乏しいため，空間情報やセンサの観測範囲などの詳細なセンサノード情報を扱うことはできない．アプリケーションが利用するセンサを選択する場合，センサの設置位置だけでなく，その観測範囲を考慮する必要があり，その点で MARS が提供する機能は不十分である．本システムでは，センサノードのメタ情報の通信に XML を用いることで，文字列情報や観測範囲の形状情報を表現したり，未解決メタ情報の外部依存を明示したりすることが可能であり，柔軟なメタ情報伝達を可能とした．また，本システムでは，アプリケーションが空間の形状を踏まえたうえでサービス提供のために適切なセンサノードを選択することが可能である．

SensorML[6] はセンサ情報を記述する XML 規格である．センサ機器の識別情報や，詳細な説明文，設置者の情報，センサが設置されているオブジェクトの形状，観測データの取得方法など，詳細に記述することが可能である．しかし，複数のドキュメントから構成されていたりネスト構造が深いなど，データ構造が複雑で，データサイズは巨大である．多くのセンサノードが頻繁に通信を行うユビキタスコンピューティング環境における SensorML の利用は，情報解析処理に負担を与え，通信インフラを圧迫する．また，SensorML の使用用途は，地球規模の環境観測機器の詳細をデータベースで活用することが主であり，本システムにおいての使用には不適である．そこで，本システム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ノードの有効な情報だけをコンパクトに表現可能とする TinySensorML を新規に定義した．

SensorScope[10] は，空間の見取り図とセンサノードの設置位置を GUI を用いて示し，指定されたセンサノードの情報を容易に参照することが可能なシステムである．しかし，センサノードの設置位置や，搭載しているセンサの種類などの情報はあらかじめ人手で調査し，データベースに登録する必要がある．そのため，無数にセンサノードが存在する空間において SensorScope を運用することは難しい．MobileBristol[11] は，空間の見取り図と空間に存在する位置情報センサの観測範囲を GUI を用いて示し，位置情報に基づくサービスを設定するツールである．SensorScope と同様に，センサノードの設置位置や，センサの観測範囲などの情報はあらかじめ人手で調査し，データベースに登録する必要がある．センサノードのメタ情報をセンサノードから直接収集し，ワールドモデルを構築する機能を備える本システムを利用することで，人手を介することなく SensorScope や MobileBristol のようなソフトウェアを実装することが可能となる．

7. 結　　論

本研究では，センサノードのメタ情報を収集し，収集したメタ情報をもとにワールドモデルを構築し，空間およびセンサノードの情報をアプリケーションに提供するシステムを実現した．本システムを利用することにより，空間や空間内のセンサノードの情報を人手で収集するコストを削減することができる．また，本システムが提供する情報を利用するアプリケーションは，空間の形状を把握した上で特定の位置の特定の環境情報を観測するセンサノードを発見することが可能となる．部屋の天井中央部の照度の取得，部屋の四隅の温度の取得など，具体的な観測位置を指定した環境情報の取得が可能になり，サービスの質の向上を図ることが可能となる．

また，本システムが提供する機能を利用したアプリケーションとして空間視覚化アプリケーションを実装した．このアプリケーションを利用し，本システムの機能が有効に機能していることを確認した．

参　考　文　献

1) Marc Weiser, The Computer for the Twenty-First Century, Scientific American, Vol.265, No.3, pp.94-104(1991).
2) Crossbow Technology Inc., http://www.xbow.com/.
3) Building Intelligent Environments with Smart-Its, IEEE Computer Graphics And Applications, Vol.24, No.1, pp.56-64(2004).
4) 猿渡俊介 川原圭博 南正輝 森川博之 青山友紀 篠田庄司 永原崇範, ユビキタス環境に向けたセンサネットワークアプリケーション構築支援のた

めの開発用モジュール U^3(U-cube) の設計と実装, 電子情報通信学会技術研究報告, IN2002-243, NS2002-270(2003).
5) 丸山大祐, 青木俊, 高汐一紀, 徳田英幸, 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, 情報処理学会研究報告, 2004-UBI-4, pp.11-16(2004).
6) Dr.Mike Botts, Earth System Science Center, NSSTC University of Alabama in Huntsville http://vast.nsstc.uah.edu/SensorML/.
7) Web3D Consortium, http://www.web3d.org/.
8) AT&T Laboratories Cambridge, http://www.uk.research.att.com/spirit/.
9) InterSense Inc., http://www.isense.com/.
10) EPFL, http://sensorscope.epfl.ch/.
11) Richard Hull, Ben Clayton, and Tom Melamed, Rapid Authoring of Mediascapes, UbiComp 2004: 6th International Conference, Nottingham, UK, pp. 125-142(2004).