

広域センサネットワークのセンサプロフィール一貫性保証と相互接続性

落合 秀也[†] 江崎 浩[†]

[†] 東京大学

東京都文京区本郷 7-3-1 工 2 号館 102B1 江崎研究室

E-mail: tjo2lxq@hongo.wide.ad.jp, t†hiroshi@wide.ad.jp

あらまし 本研究は、広域センサネットワークにおけるセンサプロフィールの一貫性問題を解決し、アプリケーションレベルでの相互接続性を実現するアーキテクチャを提案する。プロフィールとして扱われているセンサに関する付加的情報は、種々センサアプリケーションのオブジェクトであり、権限の与えられた組織による、これらオブジェクトの管理は必須である。本研究で提案するアプリケーションドメインアーキテクチャは、アプリケーション権限のある組織に独自にアプリケーションオブジェクトを管理させることで、自律分散的なアプリケーションオブジェクトのグローバル管理を可能にし、プロフィールの一貫性問題を解決する。本研究では、プロトタイプシステムによって4つのアプリケーションドメインを作成し、アーキテクチャの検証をしている。

キーワード プロファイル一貫性、スキーマ管理、アプリケーションオブジェクト、相互接続性、広域センサネットワーク

Sensor Profile Consistency and Interoperability in Global Sensor Networks

Hideya OCHIAI[†] and Hiroshi ESAKI[†]

[†] The University of Tokyo

EsakiLab Bld2. 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan

E-mail: tjo2lxq@hongo.wide.ad.jp, t†hiroshi@wide.ad.jp

Abstract This paper proposes architecture for global sensor networks that achieve and maintain sensor profile consistency. Profile information is tightly related to sensor applications. Profile describes application objects, which should be managed by some authorities. Our proposing application domain architecture enables profile consistency globally with the administration of application objects by authorities. We demonstrate, in this paper, our architecture with our prototype implementation that deploys four application domains.

Key words profile consistency, schema management, application object, interoperability, global sensor network

1. はじめに

インターネット上に構築される広域センサネットワーク (e.g., Live E! [1], GSN [2]) においては、センサを特徴付ける目的でセンサに対し、プロフィールが定義される。このプロフィールには、センサの設置場所、機種、観測内容、設置環境、管理者、インタフェースなどの様々な情報が含まれる。各種センサアプリケーションは、これらのプロフィールを参照することで目的に合ったセンサの検索を行い、それぞれの利用形態に合わせてセンサデータを利用する。

現在のグローバルセンサネットワークにおいては、プロフィール上のデータに関して相互接続性の欠如が発生している。こ

れはプロフィールスキーマがシステム全体で管理されていないために起こる現象で、スキーマ管理機構を設けない限り、アプリケーションデータの相互接続性の点で、スケールしなくなることの意味している。

本研究では、プロフィールスキーマの管理をアプリケーション単位で独立に行い、それらの統合が可能なセンサアプリケーションのデータ管理フレームワークを提案している。グローバル環境上に様々なアプリケーションオブジェクトを展開し、それらを自律分散的に結合させることで、プロフィールスキーマの管理および、アプリケーションレベルでの相互接続性を実現すると共に、様々なセンサアプリケーションの導入を可能にするフレームワーク、すなわち、ユビキタスデータの管理フレー

ムワークを構築する。

センサの設置者をプロフィール発行者とし、アプリケーションをプロフィール利用者として捉えると、センサの検索および相互接続性の問題は、発行者と利用者のマッチング問題に帰着することができる。センサプロフィールの作成は、センサの設置者によって行われ、各種センサアプリケーションは、そのプロフィールを元に検索をすることで、利用目的に合ったセンサを抽出する。プロフィール検索などのクエリとプロフィールのマッチングを実現する、すなわち、演算可能性を確保するためには、データに関するスキーマが双方で共有されていることが前提となる。ところが、現在のグローバルセンサネットワークのアーキテクチャでは、この前提を確保する機構すら存在していない状況であり、そのため、アプリケーション層での相互接続性の欠如が発生している。

相互接続性が確保されたシステムにおいては、全体にわたって同一プロトコルが実装され、データ・スキーマも共有されている。インターネットにおいては、Internet Engineering Task Force(IETF) などがグローバルに共有されるべきプロトコルを承認し、Internet Assigned Numbers Authority(IANA) などが、グローバルに共有されるべき、データ・スキーマを管理している。これらの権威ある団体／組織から提示されるプロトコルやデータ・スキーマに従って実装することで、相互接続性のあるシステムを構築することが可能になっている。

グローバルセンサネットワークにおいても、アプリケーション層での相互接続性を実現させるためには、スキーマを管理する権威ある組織や団体を設けることは必要となるだろう。しかし、グローバルセンサネットワークにおいては、センサが複数のアプリケーションに属することが問題を難しくしている。プロフィールには、アプリケーションに応じて、センサ実装に依存できる部分と依存できない部分があり、設置者により明示的に指定すべき部分が必ず残る。したがって、プロフィールの作成登録ツールが必要になるわけだが、管理されているデータ・スキーマを IANA のようにユーザフレンドリな形式で提示し、ソフトウェアに独自の形式で組み込む従来型的方式では、プロフィール作成ツールがスキーマの更新に対し、即時に対応できるとは限らない。

管理団体が発行するスキーマを電子化し配布することで、システムおよびプロフィール作成ツールのスキーマを更新する方式が考えられる。この方法では、スキーマの更新が比較的速く伝達され、システムの柔軟性も向上する。ただし、スキーマの規模が大きくなった場合には、配信コストが大きくなると同時に、アプリケーションが利用するスキーマは全体の一部になると予想され、無駄の増大も懸念される。

本研究では、スキーマに関しては、配布するものに加えて、グローバル空間上に展開しておき、オンデマンドに読み出せる仕組みを提案している。この手法では、アプリケーションオブジェクトに様々な情報を付加させることも可能になる。

本研究で提案するアーキテクチャでは、プロフィールスキーマの管理をアプリケーションオブジェクトの管理に帰着させ、アプリケーション単位にオブジェクトの管理権限を分割委譲し

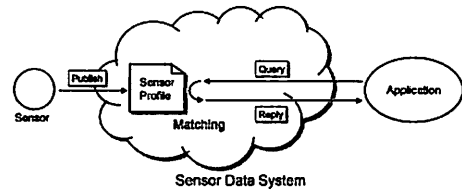


図1 センサプロフィールとセンサアプリケーションの関係

ている。この際、すべてのアプリケーションオブジェクトを抽象表現させ、アプリケーションに依存しないオブジェクト参照を可能にしている。このように抽象表現をさせることによって、アプリケーションを越えて自律分散的にオブジェクトを結合させることが可能になる。他の管理されたアプリケーションオブジェクトを参照させることで、アプリケーション層での相互接続性が確保される。

グローバル空間上にプロフィールのスキーマを構築する場合、下記条件が満足される必要がある。

- (1) アプリケーションの独立管理性
- (2) アプリケーションデータのスケラビリティ
- (3) アプリケーションに依存しない抽象表現
- (4) アプリケーションの統合性

本研究で提案するアーキテクチャでは、これらすべてを満足している。

本研究では、木構造のアプリケーションデータに限定してプロトタイプを作成し、提案アーキテクチャの動作検証を行っている。このプロトタイプシステムによりアプリケーションの独立管理性、抽象表現による統合性を検証し、相互接続性が実現される仕組みについて述べている。

本論文の構成は以下の通りである。第2章でセンサのプロフィールとその管理について述べ、第3章で提案アーキテクチャについて解説する。第4章で、プロトタイプシステムによるデモンストレーションを行い、第5章で考察を述べ、第6章でまとめる。

2. センサプロフィール管理

本章では、センサに割り当てられるプロフィールに関して問題を掘り下げる。まずセンサプロフィールとセンサアプリケーションの関係を示し、プロフィール項目の具体例を述べると共に、アプリケーションレベルでの相互接続性実現のためのスキーマ管理について解説する。

2.1 プロフィールとアプリケーション

図1に示すように、センサデータシステムにおいては、登録されているセンサプロフィールと、アプリケーションが発行するクエリとの間でマッチングが行われる。センサプロフィールには、それぞれのアプリケーションによる、検索のために必要な複数のIDや、参考程度にとどめておく情報など様々なアプリケーション向けの情報が掲載されている。検索操作 (i.e., インデックス検索, フィルタリング) はクエリ発行によって行われ、マッチしたセンサおよびプロフィールの読出しが行われる。

表 1 センサプロフィールの例

項目	意味
location	位置情報 (住所表現)
latitude	位置情報 (緯度)
longitude	位置情報 (経度)
sensorVendor	センサの製造メーカ
sensorModel	センサの型番
sensorType	観測内容 (e.g., 温度, 湿度)
environment	設置環境
organization	センサの管理者
interface	アクセスインタフェース

```

<?xml version="1.0" encoding="UTF-8"?>
<profileSchema>
  <!-- type is one of boolean, integer, float, time, string -->
  <schema name="location" type="string" value="."/* />
  <schema name="latitude" type="float" value="."/* />
  <schema name="longitude" type="float" value="."/* />
  <schema name="sensorVendor" type="string"
    value="Vaisala|AmbientWeather" />
  <schema name="sensorModel" type="string"
    value="WXT510|WM918" />
  <schema name="sensorType" type="string"
    value="Temperature|Humidity|Pressure|RainFall|WindSpeed" />
  <schema name="environment" type="string" value="out|in" />
  <schema name="organization" type="string" value="."/* />
  <schema name="interface" type="string" value="."/* />
</profileSchema>

```

図 2 プロファイルスキーマの例

2.2 プロファイルの構成

プロファイル項目の例を表 1 に示す。プロファイルに sensorVendor="Vaisala", organization="江崎研"と書いてあれば、江崎研で設置管理されている、Vaisala 社のセンサを意味する。アプリケーションはクエリを発行することで、これらの値との比較を計算し、利用目的に応じたセンサの選択を行うことができる。

2.3 プロファイルスキーマと相互接続性

図 2 にプロファイルスキーマの例を示す。プロファイルスキーマによって、システムで扱われるプロファイル項目の種類および取りうる値が定義される。図 2 では、schema タグの中の name 属性でプロファイル項目を定義し、value 属性により取りうる値の条件を正規表現で記述してある。このようにプロファイルスキーマを定義し、システム全体に配布することで、登録時のプロファイルチェックを通じて、プロファイルの一貫性が保たれる。

プロファイルスキーマが管理されていない状況下では、プロファイル入力はセンサ設置者次第で個別に行われるため、意味としては同一のものを指定しているにもかかわらず、別々の表現で実体への参照が密達まってしまう現象が起こる。これは、プロファイルの一貫性が失われた状態で、アプリケーションレベルでの相互接続性の欠如と等価である。プロファイルスキーマを上記のように表現し、システム全体で管理することで、プロファイルの一貫性が保たれ、アプリケーションレベルでの相互接続性は維持される。

ただし、図 2 の例では、sensorVendor や sensorType のように、少数のアプリケーションオブジェクトの管理は可能であるが、例えば、住所表現的位置情報 (location) のような巨大な属



図 3 アプリケーションドメインの例 (location)

性値集合を管理することはできない。そのため図 2 では、任意の値を可能にしてしまっており、location に関しては相互接続性は保証されない。

3. アプリケーションドメインアーキテクチャ

ここでは、我々の提案する各アプリケーションドメインにおける独立なデータ管理機構と、それらを自律分散的に統合し、スケーラブルなアプリケーションオブジェクト集合においても相互接続性を維持可能なアーキテクチャの解説を行っている。

3.1 アプリケーションドメイン

センサと関連付けがなされる、種々アプリケーションのオブジェクトをエンティティとして抽象化し、各アプリケーションで管理させることを考える。このように、アプリケーションごとにエンティティの管理領域を分割させることで、アプリケーションドメインを構築させる。それぞれのアプリケーションドメインにおいては、

- データ構造
- 検索アルゴリズム
- エンティティ
- 検索インタフェース/プロトコル

は、すべてのアプリケーションドメインから、参照利用できるように、抽象的に表現されている。

図 3 に、location アプリケーションドメインの例を示す。図 3 の例は、木構造のドメインで、木のノードが、エンティティに相当する。以下では、提案アーキテクチャのエンティティとデータ構造について解説し、検索の仕組みについて述べている。

3.1.1 エンティティとデータ構造

提案アーキテクチャでは、アプリケーションオブジェクトをエンティティと呼ぶ。図 3 のように、エンティティには ID が割り当てられ、グローバル環境上で一意に指定することが可能である。エンティティは、ID の他に、付加的情報を与えるアトリビュート (属性) を定義することも可能である。

図 3 では、location アプリケーションドメインは木構造で構成されているが、地表面をモデル化した 2 次元平面、シリアル番号のように 1 次元構造のものなど、様々なものが考えられる。エンティティは抽象化されており ID は、

- location=bunkyo.tokyo.jp
- earthSurface=(35.2, 139.7)

のように表現される。

エンティティへのアクセスインタフェースとしては、エンティティの関連付けインタフェースや、検索用のインタフェースがあり、システム内で標準化されている。

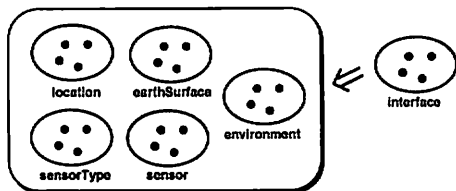


図4 インターアプリケーションドメイン

3.1.2 検索の方法

本システムにおいては、検索には Iterative サーチ [3] の手法を採用している。つまり、検索の問い合わせを受けたエンティティは、問い合わせ元に対してリダイレクションを発行する。問い合わせ元では、このリダイレクションを受けて、次のエンティティに問い合わせをする。上記の操作を繰り返し行い、目的のエンティティにたどり着くという方式である。

検索アルゴリズムは、アプリケーションごとに自由に定義され、また、データ構造ごとに異なるが、任意の ID に対して、エンティティを取り出すことは可能だろう。代表的なアルゴリズム設計指針としては、食欲アルゴリズムの適用が考えられる [4]。図3のように、木構造（トライ）であれば根から辿っていけるし、2次元平面構造であれば、GPSR [5] のようなアルゴリズムで解決できる。

3.2 インターアプリケーションドメイン

本研究では、複数のアプリケーションドメインを束ねてコミュニティを作成するアーキテクチャを提案している（図4）。すべてのアプリケーションは抽象化されたエンティティを持ち、アプリケーションをまたがって互に関連付けをすることができるようになっている。各ドメインのエントリを管理し、システム全体で共有すれば、システムの任意の場所から、それぞれのアプリケーションドメインに入り込み自律分散的に関係を結ぶことが可能になる。アプリケーションは、それぞれのドメインへのエントリを公開し、アプリケーションの統括管理機構に登録してもらう。統括管理機構では、配布するスキーマにアプリケーションドメインへのエントリを Delegate として掲載する。このようにして、各アプリケーションドメインへのエントリがシステムの隅々まで伝達される。

3.3 センサプロファイルの登録

センサには専用のドメイン（sensor ドメイン）が存在し、センサはこのドメイン上のエンティティとして管理される。ユーザがプロファイルを登録するには、センサのエンティティで親エンティティを、

- location=bunkyo.tokyo.jp
- earthSurface=(35.45,135.5)
- environment=out
- sensorType=t.wxt510.vaisala
- organization=jo2lxq.elab.wide

のように登録すればよい。このようにして、センサエンティティは、他のアプリケーションドメインのエンティティと関連付けることが可能で、エンティティID が登録されると、自動的にこ

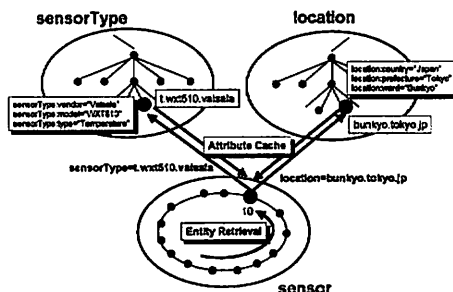


図5 動的属性キャッシュ

ModelService
String sustainLink(String chInfo);
String getEntity(String query);
String getEntityList(String query);
String addEntity(String entity);
String updateEntity(String entity);
String deleteEntity(String entity);

図6 自律分散結合/ユーザ操作サービス

れらのエンティティへ自らの通知を行う。すると、それぞれのアプリケーションへセンサが登録され、アプリケーションからもセンサにたどり着けるようになる。この双方向リンクの形成が、管理された情報との結合を意味し、プロフィールの一貫性を保証する。プロフィールの一貫性が維持されることで、アプリケーション間での相互接続性が実現される。

3.4 動的属性キャッシュ

センサの検索によって、センサを読み出すことができたときに、分散して存在している親エンティティに格納されている属性値も同時に参照しなくてはならない。このコストを削減し、パフォーマンスを向上させるために、センサプロフィールの登録によって形成された双方向リンクを利用して、属性値を前もって下位エンティティに送り、キャッシュしておくことを考える。

図5に例を示す。sensor ドメインにある sensor=10 が、sensorType=t.wxt510.vaisala, location=bunkyo.tokyo.jp を参照していて、これら間で双方向のリンクが張られている。すなわち、sensor=10 から定期的なリンク形成のための通知を、これら親エンティティに対して行っている状況である。この通知の際に、親エンティティは、自身の持つ属性値を呼び出し元（sensor=10）に公開する。この属性値を呼び出し元エンティティのローカルにキャッシュしておくことで、検索による読出しの際には、親エンティティの属性値も同時に提供できるため、検索応答のパフォーマンスが向上する。

4. プロトタイプシステム

我々は木構造のアプリケーションドメインに関して、提案アーキテクチャの実装を行い、目的の動作を確認した。

4.1 実装

基本コンポーネントのサービス/インタフェースを図6に示

```
<?xml version="1.0" encoding="UTF-8"?>
<MAMessage version="1.0">
  <Control name="sustainLink">
    <parameter name="targetParent" value="location=taito.tokyo.jp" />
  </Control>
  <Object>
    <entity
      entryURL="http://t3.live-e.org/axis/services/ModelService"
      id="sensor1.jo2lxq.org"
      model="sensor"
      validityTime="86400000"
    />
  </Object>
</MAMessage>
```

図 7 sustainLink の呼出しに使われるメッセージ例

```
<?xml version="1.0" encoding="UTF-8"?>
<MAMessage version="1.0">
  <Control name="redirect">
    <parameter name="entryURL"
      value="http://t3.live-e.org/axis/services/ModelService" />
  </Control>
</MAMessage>
```

図 8 リダイレクションの例

```
<?xml version="1.0" encoding="UTF-8"?>
<MAMessage version="1.0">
  <Control name="OK" />
  <Object>
    <entity id="taito.tokyo.jp" model="location">
      <attribute>
        location:country:Japan;
        location:prefecture:Tokyo;
        location:ward:Taito
      </attribute>
    </entity>
  </Object>
</MAMessage>
```

図 9 動的属性キャッシュの例

し、この中の sustainLink メソッドで交換されるデータ例を図 7—図 9 に示す。プロトタイプ実装では、通信インタフェースはすべて SOAP Web サービスとし、交換されるメッセージは XML で表現されている。

sustainLink メソッドは、自律分散結合の要を担うメソッドであり、呼び出しは図 7 のようなメッセージで、設定された親ノードに向けて行われる。サービスの呼び出しは、木構造の上位から行われ、呼び出し先のサービスに親ノードが存在しない場合、下位ノードを保持するサービスへリダイレクションを発生させる (図 8)。こうして、指定した親ノードを保持するサービスにたどり着き、自ノード情報の通知が完了すると同時に、親ノードが保持する動的属性を継承することができる (図 9)。sustainLink への通知は定期的に行われ、自律分散システムとして、データ構造をネットワーク上に保持することができる。

sustainLink 以外のメソッドは、グローバルに利用可能なエンティティの読出しメソッド (getEntity, getEntityList) と、サーバの管理者のみが利用を許されるべきエンティティ操作メソッド (addEntity, updateEntity, deleteEntity) がある。

プロトタイプシステムにおいては、複数のアプリケーションドメインへのエントリ管理とエントリ情報の公開は、ネットワーク上に設置されたサービスで提供することにした。

4.2 デモンストレーション

このデモンストレーションにおいては、プロトタイプ実装を用いて下記ドメインを作成し、自律分散的にこれらのエンティ

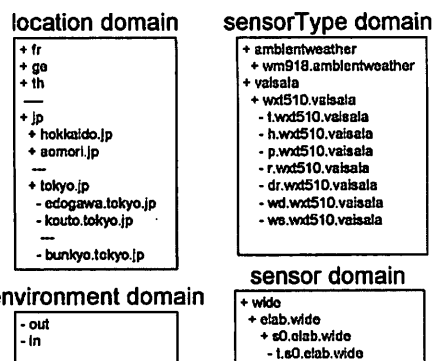


図 10 構築されたデータセット

表 2 サーバの割当て

サーバ	担当ドメイン
192.168.25.1	location
192.168.25.2	location
192.168.25.3	location
192.168.25.4	sensorType, environment
192.168.25.5	sensor

ティが結びつく様子を観察すると共に、どのようにしてセンサアプリケーションに関する相互接続性の実現されているかを示す。

- ・ 場所情報を管理する location ドメイン
- ・ センサの機種タイプを管理する sensorType ドメイン
- ・ 設置環境を管理する environment ドメイン
- ・ センサの実体を管理する sensor ドメイン

図 10 にプロトタイプ実験で構築したユビキタスデータフレームワークの構成を示す。これらは 5 台のサーバ上に構築された。割り当てを表 2 に掲載する。

この環境において、

- ・ sensor=t.s0.elab.wide

の親エンティティとして、下記を指定し、この設定前後における、親エンティティから見た子エンティティ項目 (表 4) と、子エンティティにキャッシュされる動的アトリビュートのリスト (表 5) を観察した。

- ・ location=bunkyo.tokyo.jp
- ・ sensorType=t.wxt510.vaisala
- ・ environment=out

ここで、これらのエンティティが静的に保持しているアトリビュートを表 3 に示す。

表 4 からは、sensor エンティティでの親登録により、自動的に親エンティティの子項目に自エンティティが登録されたことがわかる。

表 5 には、設定前後で sensor エンティティが保持している属性値が親エンティティから継承されてキャッシュされていることが示されている。

アプリケーションの相互接続性は、リンクの形成が保証する。アプリケーションからセンサの検索においては、(1) ある

表3 属性の例

エンティティ	属性
sensor= t.s0.elab.wide	sensor:owner="Hideya Ochiai" sensor:organization="EsakiLAB"
location= bunkyo.tokyo.jp	location:country="Japan" location:prefecture="Tokyo" location:ward="Bunkyo"
sensorType= t.wxt510.vaisala	sensorType:vendor="Vaisala" sensorType:model="WXT510" sensorType:type="Temperature"
environment= out	environment:out="true"

表4 親エンティティに子として追加される例

エンティティ	子エンティティ(前)	子エンティティ(後)
sensor= t.s0.elab.wide		
location= bunkyo.tokyo.jp		sensor= t.s0.elab.wide
sensorType= t.wxt510.vaisala		sensor= t.s0.elab.wide
environment= out		sensor= t.s0.elab.wide

表5 動的キャッシュの例

	sensor=t.s0.elab.wide の動的属性
前	sensor:owner="Hideya Ochiai" sensor:organization="EsakiLAB"
後	sensor:owner="Hideya Ochiai" sensor:organization="EsakiLAB" location:country="Japan" location:prefecture="Tokyo" location:ward="Bunkyo" sensorType:vendor="Vaisala" sensorType:model="WXT510" sensorType:type="Temperature" environment:out="true"

アプリケーションドメインから検索を開始し(例えば, location=bunkyo.tokyo.jp), それに関連付けられているセンサすべてを抽出する方法と, (2) あるアプリケーションから検索を開始し, その後, 特定の条件(例えば, environment=out)に結びついているもののみをフィルタリングする方法とがある。いずれの場合においても, 管理されたエンティティとのリンクが形成されていれば, プロファイルの一貫性が保たれ, アプリケーションでの相互接続性が実現される。

5. 考 察

本研究で実現したプロトタイプシステムでは, 独立したアプリケーションごとにデータ管理が可能で, アプリケーション間で自律分散的にエンティティ結合が行われた。管理されたアプリケーションオブジェクトとの結合により, プロファイルの一貫性が保証され, センサアプリケーションの相互接続性が実現できることが示された。

本研究で提案したアーキテクチャでは, プロファイルスキーマ配布による方法と比べて, 大規模なアプリケーションオブジェクト集合を構成できる可能性を秘めている。今後は定量的な評価試験を行うべきだと考えている。評価試験の内容としては, 規模の検証, スキーマチェックにかかる時間, 木構造以外のシステムの場合, などが考えられる。

6. お わ り に

本研究では, グローバルインターネット上に構成されるセンサデータシステムに関して, センサアプリケーション間の相互接続性を実現するためには, プロファイル一貫性の必要性を提唱した。プロファイル一貫性を実現するためにはプロファイルスキーマの共有が必須である。本研究では, スキーマをグローバル空間上にアプリケーションドメインとして実現することで, これをシステム全体で共有するアーキテクチャを提案している。本研究で提案したアプリケーションドメインアーキテクチャは,

- アプリケーションの独立管理性
- アプリケーションデータのスケーラビリティ
- アプリケーションに依存しない抽象表現
- アプリケーションの統合性

を特徴とし, 全体でセンサおよび関連するアプリケーションのデータ複合体が形成される。我々はこの複合的なデータ集合のことをユビキタスデータと呼んでいる。

本研究では, 木構造のアプリケーションに限定してプロトタイプシステムを構築し, アプリケーションの独立管理性, および抽象表現による統合性を検証した。相互接続性が達成される仕組みも提示した。

謝 辞

本研究は, 総務省の委託研究「ユビキタスネットワーク認証・エージェント技術の研究開発」の一環として実施したものである。

文 献

- [1] H. Esaki and H. Sunahara: "Live E! project; sensing the earth with internet weather stations", IEEE/IPSJ SAINT'07, p. 1 (2007).
- [2] K. Aberer, M. Hauswirth and A. Salehi: "Infrastructure for data processing in large-scale interconnected sensor networks", IEEE MDM 2007 (2007).
- [3] 首藤, 加藤, 門林, 土井: "構造化オーバーレイにおける反復探索と再帰探索の比較", IEEE/IPSJ SWoPP2006 (2006).
- [4] J. Kleinberg and E. Tardos: "Algorithm Design", chapter 4, pp. 115-208, Addison Wesley (2006).
- [5] B. Karp and H. T. Kung: "GPSR: greedy perimeter stateless routing for wireless sensor networks", ACM MOBIKOM 2000, pp. 243-254 (2000).