

無線センサネットワーク用ミドルウェアの検討

竹島 亮祐[†] 久保田 稔[‡]千葉工業大学 大学院工学研究科^{†‡}

1. はじめに

企業などの間で、センサネットワークを利用したプラントや工場のラインの状態を監視するシステムの開発事例が多くなっている。センサネットワークシステムには、センサノード上で動作する制御ソフトウェアとホスト PC 上で動作するアプリケーションの 2 つが必要となる。これらの知識を十分に習得していない技術者にとって、無線センサネットワーク用のアプリケーション開発が困難なものとなる。

本研究では、上記のアプリケーション開発を容易化することを狙いとしたミドルウェアを提案する。また、ホスト PC 上で動作するサンプルアプリケーションを実装し、実環境で動作させた上でそれらを評価する。

本ミドルウェアは、

- センサノード上で動作し、ホストからクエリに対して搭載されたセンサの値の応答や、環境に対して変化をもたらすアクチュエータの動作を行う制御ソフトウェア
- ホスト PC 上で動作し、アプリケーションにセンサネットワークへアクセスするためのインターフェース(API)を提供し、仲介を行うソフトウェア

の 2 つから構成される。

2. 関連研究

センサネットワークを DB とみなしてクエリを発行し、データの取得を行う TinyDB [1] や Cougar [2] といったミドルウェアが提案されている。これらのミドルウェアは、ホスト PC 側のソフトウェアに PostgreSQL などの DBMS を組み込むことが多い。このため、ユーザは利用するミドルウェアの他に使い方を習得すべきソフトウェアの数が多くなってしまう。ユーザがセンサネットワークへアクセスするための API については、文字列のクエリをバイナリへ変換するといった機能はあるものの、ユーザの負担を軽減させる機能に関しては十分ではない。

Study on Middleware for Wireless Sensor Network

[†]Ryosuke TAKESHIMA, [‡]Minoru KUBOTA

^{†‡}Chiba Institute of Technology

3. ミドルウェアの設計方針

提案するミドルウェアの概要を図 1 に示す。ミドルウェアはセンサネットワークの制御あるいは状態の取得をするための機能のインターフェース(API)をオブジェクトのメソッドとして提供する。このオブジェクトはセンサノードの機種に依存しない形式で提供されるため、異なるセンサノードであっても再利用が容易となる。

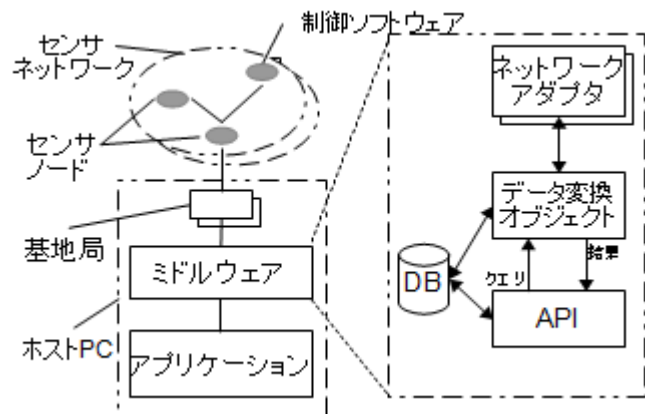


図 1 ミドルウェアの概要

本ミドルウェアでは、センサネットワークを TinyDB のように仮想的な DB とみなし、センサの値は仮想的なテーブルに格納されているものとする。API を通してアプリケーションからの要求を受けた時、その要求を仮想的な DB を制御するクエリへ変換しデータ変換オブジェクトへ送る。

データ変換オブジェクトは、クエリを要求先のセンサネットワーク毎に対応した形式の制御データへ変換し、これをネットワークアダプタを介して各センサネットワークへ送る。制御データを受け取ったセンサノードの制御ソフトウェアが処理を行い、結果をホスト PC へ返信する。結果はデータ変換オブジェクトでセンサ測定値の単位の変換などが行われ、API を通してアプリケーションへ戻される。センサネットワークに対応して行われる上記の変換処理のためにセンサネットワーク個々の情報を格納した DB を保持する。

本ミドルウェアは複数のクエリをあらかじめ組み合わせたラッパーを提供する。これは、ア

アプリケーションを開発する上で頻出する処理要求を簡潔に記述することを可能とし、開発者の負担の軽減を狙う。例としては、特定のノードからセンサの値を取得や、特定のノードの近傍にあるセンサから一括してセンサの値を取得する、といった要求がある。

4. ミドルウェアのインターフェース(API)

ネットワーク内の全てのノードから温度センサの値を取得する処理を例に、本ミドルウェアのAPIについて説明する。

```
~
TinyDBMain.initMain();
q = SensorQueryer.translateQuery("SELECT
light", (byte)1);
~
public void addResult(QueryResult qr) {
    Vector v = qr.resultVector(); //print
the result
    for (int i = 0; i < v.size(); i++) {
        System.out.print("\t" + v.elementAt(i)
+ "\t|");
    }
    System.out.println();
}
}
```

図 2 TinyDB の JavaAPI を使用した記述

図 2 で示している TinyDB の JavaAPI を利用した記述では、センサネットワークを仮想的な DB として抽象化するため、プログラム内に TinyDB のクエリ(図 2 の下線部分)を記述する必要がある。これはユーザの自由な形式でデータが取得できる反面、複雑な要求を記述する場合には 1 行の長いクエリを記述しなければならず、エラーを生じやすい。また、プログラム中では文字列として記述するため、IDE の補完機能や文法チェック機能を利用するには辞書データを作成することが必要となる。

```
~
sensorNetwork.discovery();
SensorList list =
sensorNetwork.getNodes();
SensorList lightSensors =
list.haveSensor(Sensor.LIGHT);
for(Sensor sens : lightSensor){
    System.out.println(sens.sensorValue);
}
```

図 3 提案方式での記述

図 3 で示す本ミドルウェアでの記述はクエリをプログラムに埋め込む必要がない。クエリに用いるセンサの種類を指定する定数などは、API のオブジェクト内に定義済みである。このため、オブジェクトがもつメソッドやプロパティ名の

補完といった IDE が提供する開発支援機能を利用することが可能となっている。

5. 実装と評価

提案したミドルウェアの予備的実装を行った。動作確認および評価のため、センサノードから温度センサの値を取得するアプリケーションを作成し、本ミドルウェアの動作確認を行った。動作確認のための環境として、基地局として Crossbow Technology 社製の IRIS に USB インターフェースボードである MIB520 を接続したものを使用した。また、センサノードとして IRIS にセンサ基板である MTS300 を接続したものを使用した。

本ミドルウェアを用いた場合と比較対象として既存のミドルウェアである TinyDB を用いた場合の両方の場合において、温度センサの値を取得するプログラムを動作させた時に同様の値が取得できることを確認した。さらに、センサノードへ書き込まれるソフトウェアのサイズ、通信の量、アプリケーションのソースコードを比較し、ノードへの負担および開発者への負担を評価する。また、アプリケーションの要求から値を得るまでに要した時間を遅延として比較し、評価をする。

6. まとめ

センサネットワークに関する詳細な知識をもたなくてもアプリケーションの開発を容易化するミドルウェアについて提案をした。本ミドルウェアは、ユーザへ API として提供するオブジェクトの構造をユーザの開発容易性に着目して構成することや、IDE の開発支援機能を積極的に利用することにより、アプリケーション開発の容易化を実現した。また、本ミドルウェアの予備的実装と評価を行った。今後はミドルウェアの実装を進め、より詳細な動作実験および評価を行う。

参考文献

- [1] Madden, S. and Hellerstein, J. and Hong, W.: TinyDB: In-network query processing in TinyOS, Intel Research, IRB-TR-02-014, 2002.
- [2] Yao, Y., and Gehrke, J.: The cougar approach to in-network query processing in sensor networks, ACM SIGMOD Record, 31(3), pp.9-18, 2002.