DEIM Forum 2012 B1-2

スマートフォンを用いたセンサデータ処理収集のためのフレームワーク

義久　智樹†　　西尾章治郎††

† 大阪大学サイバーメディアセンター　〒 567–0047 大阪府茨木市美穂ヶ丘 5-1
†† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1-5
E-mail: †yoshihisa@cmc.osaka-u.ac.jp, ††nishio@ist.osaka-u.ac.jp

あらまし　近年普及しているほとんどのスマートフォンは温度センサや位置センサなどの様々なセンサを備えている．センサが計測したセンサデータに対して，平均や差分を計算するといった処理を行うことで，例えば，気温分布や人の混み具合を調べられる．スマートフォン単体で行える処理であれば各スマートフォンで行い，他のセンサデータが必要だったり，計算能力が不足している場合にはセンサデータ集約サーバなどの他の計算機で処理することで，それぞれの計算資源を有効活用できる．しかし，これまでにスマートフォンでセンサデータを処理してサーバに送信したり，サーバ側で処理するといった機能を備えたセンサデータ処理システムはなかった．そこで本研究では、記述された処理に従ってセンサデータを処理して収集するセンサデータ処理収集フレームワークを提案する．

キーワード　センサネットワーク，携帯型端末，モバイルコンピューティング

A Framework for Sensor Data Processing and Collection using Smart Phones

Tomoki TOSHIHISA† and Shoiro NISHIO††

† Cybermedia Center, Osaka University
5-1 Mihogaoka, Ibaraki, Osaka, 567–0047 Japan
†† Grad. School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka, 565–0871 Japan
E-mail: †yoshihisa@cmc.osaka-u.ac.jp, ††nishio@ist.osaka-u.ac.jp

Abstract　Recently prevalent smart phones equip various sensors such as temperature and position sensors. For instance, we can see the temperature distribution by calculating the average of the data obtained from temperature sensors and can find crowded places by calculating differenes of the data obtained from position sensors. We can exploit processing resources by executing such processes on smart phones when smart phones have sufficient processing resources, otherwise, we should excute them on other powerful computers such as a sensor data collection server when the processes require other sensor data or smart phones have insufficient processing resources. However, there is no framework that can execute processes on smart phones and send them to the server and also execute them on the server. Hence, in this research, we propose a framework that collects and processes sensor data according to the user descrived processes.

Key words　sensor networks, portable devices, mobile computing.

1. はじめに

近年の無線通信技術の発達に伴い，スマートフォンが普及している．ほとんどのスマートフォンは，インターネットに接続でき，また，温度センサや位置センサなどの様々なセンサを備えている．スマートフォンのセンサは，以下の用途で活用できる．

- スマートフォンの温度センサから温度を取得して街中の気温分布を調べる．カバンに入っている等で正確な気温を計測できないことがあるが，周囲のスマートフォンの値と比べることで特異値として認識できる．
- スマートフォンの位置センサから人の居場所を取得して街中の混み具合を調べる．居場所の誤差が大きいことがあるが，居る確率を求め、混み具合を確率的に計算できる．

- スマートフォンの圧力センサから気圧を取得して街中の天気予報を行う．気圧が低ければ雨が降りやすい．湿度センサと組み合わせることで降雨量も推測できる．

これらの用途では，複数のスマートフォンからセンサデータを取得して処理するため，センサデータを処理収集するサーバが必要になる．スマートフォンが取得したセンサデータを処理せずにそのままサーバに送信してサーバ側ですべての処理を行う方法があるが，スマートフォンで実行できる処理はスマートフォン側で処理することで，サーバの処理負荷を軽減できる．また，スマートフォンで処理するこで送信するデータを削減できることがある．

このため，多くの場合，スマートフォンがセンサからデータを取得して必要な処理を行ってからサーバに送信している．サーバは複数のスマートフォンから送信されたデータをさらに処理して結果を得る．例えば，混み具合を調べる場合を考える．各スマートフォンは定期的に位置センサのデータを取得している．位置データの誤差を r として，スマートフォンの利用者は平均速度 v でスマートフォンを持って移動しており，位置データを取得した時刻からの経過時間を t とすると，利用者は，位置データの場所を中心として半径 $r+vt$ の円内に必ず存在することになる．取得した位置データから作られる複数のこの円の重なる領域を算出することで，利用者が存在する領域を狭められ，誤差を改善できる．各スマートフォンは，送信するデータを少なくするために，算出された領域を囲む四角形を居場所としてサーバに送信する．サーバは，領域を格子状に区切り，各格子の面積に対して，格子に含まれる送信された四角形の面積の割合から，混み具合を調べられる．誤差改善の他にも，センサデータの平均や分散を計算するといった処理をスマートフォンで実行することが考えられる．

上記のようにスマートフォンでセンサデータを処理してサーバに送信し，サーバが送信されたデータをさらに処理するといった手順は，多くのシステムで使われている，しかし，スマートフォンを用いたセンサデータ処理収集システムを開発する場合，開発者はすべてのシステムを構築する必要があって開発に手間がかかっていた．これは，用途に応じてシステムを開発していたためであり，用途ごとに異なる処理は変更し，手順が同じ部分についてはフレームワークとして提供することで開発の手間を軽減できる．

そこで本研究では，記述された処理に従ってセンサデータを処理して収集するセンサデータ処理収集フレームワークを提案する．提案フレームワークでは，スマートフォンで実行するプログラムとサーバで実行するプログラムを開発者が準備し，プログラムの配信や実行を行う部分をフレームワークとして提供する．提案フレームワークを用いることで，スマートフォンでセンサデータを処理してサーバに送信してサーバが送信されたデータをさらに処理するシステムで，サーバの処理負荷軽減や通信料削減を容易に実装でき，センサデータ処理収集システムの開発手間を軽減できると考えられる．

2. 関 連 研 究

スマートフォンからセンサデータを収集して処理する以下のシステムが研究開発されている．

LifeMap では，バス停があって人が集まったり，公園があって子供がたくさん集まる，といったイベント場所を特定するために，スマートフォンから位置データを収集している（[1]）．収集したデータを集約してイベント場所を特定するために，位置に基づく集約と経路に基づく集約を行っている．位置に基づく集約は，複数のスマートフォンが近くにあれば，それらは同じ場所にあるとみなす集約であり，経路に基づく集約は，移動中のスマートフォンをイベント場所の対象から除く集約である．これらの集約を行うことで，複数のスマートフォンの位置データから，イベント場所を効率よく特定できる．この場合，経路に基づく集約を各スマートフォンで処理することで，位置データをすべてサーバに送信する場合に比べて処理負荷を軽減できる．

文献 [2] では，移動経路を特定する際に頻繁に位置データを送信するとスマートフォンの消費電力が大きくなる問題を扱っている．領域を幾つかのセルに分割して，スマートフォンが移動したセルの遷移順序のみ送信し，過去に類似した遷移順序のスマートフォンがあれば，それと同じ移動経路と特定している．類似した遷移順序を発見すれば位置データを送信する必要がなくなるため，消費電力を削減できる．スマートフォンではセルの遷移順序を算出する処理を行い，サーバで類似した遷移順序を発見する処理をしている．

SMART（[3]）では，GPS のような精度の高い位置データを取得できない屋内の環境において，無線 LAN などの電波強度を用いてスマートフォンの場所を特定し，人が通れる場所を示す地図を作成することを目的としている．文献 [4] では，スマートフォンの加速度センサと電子コンパスを用いてこれらを行っている．スマートフォンで移動経路を算出してサーバに送信し，サーバで移動経路を集約することで場所の特定や地図の作成が可能になる．

以上のように，スマートフォンを用いてセンサデータを収集処理するほとんどのシステムでは，スマートフォンでセンサデータを処理してサーバに送信し，サーバが送信されたデータをさらに処理している．

また，人に装着した複数のセンサでセンサネットワークを構築するボディセンサネットワークに関する研究が行われている．ボディセンサネットワークでは，文献 [5] のように，センサに比べて計算能力の高いスマートフォンでセンサデータを収集することが多い．そこで Dandelion では，Senselet と呼ぶプログラムを各センサに配信し，その処理結果をスマートフォンに送信するフレームワークを提案している（[6]）．しかし，スマートフォンと，計算機のようなサーバで処理を行う本研究のフレームワークとは異なる．

センサネットワークに関して，筆者らの研究グループでは X-sensor 2.0 と呼ぶモバイルエージェントを用いたセンサデータ収集処理システムを研究開発している．しかし，本研究のよ

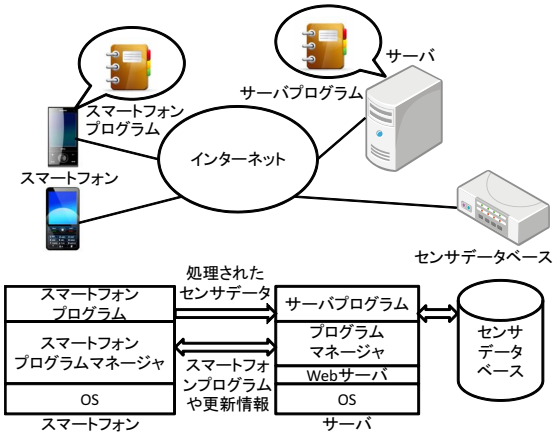

図 1　システム構成
Fig. 1　System Architecture

うにスマートフォンとの連携を対象としたものではない（[7]）．

3.　提案フレームワークの設計

提案するフレームワークのシステム構成を図 1 に示す．スマートフォンとサーバはインターネットを介して繋がっており，収集処理されたセンサデータはセンサデータベースに保存できる．サーバとセンサデータベースは同じ計算機でもよい．提案フレームワークでは，スマートフォンでセンサデータを処理してサーバに送信し，サーバが送信されたデータをさらに処理するシステムを対象としている．スマートフォンの処理を記述したプログラムを**スマートフォンプログラム**，サーバの処理を記述したプログラムを**サーバプログラム**と呼ぶ．これらのプログラムは，システムの開発者が用途に応じて作成し，サーバに登録する．

本研究では，一人の開発者がプログラムを変更して様々なシステムの性能を検証する試験用のフレームワークを想定し，使用するスマートフォンプログラムやサーバプログラムはそれぞれ 1 個とする．サーバプログラムはサーバで実行されるため，サーバに保存されていればよいが，スマートフォンプログラムはシステムに接続されるすべてのスマートフォンで実行されるため，スマートフォンに配信する必要がある．

3.1　スマートフォンプログラムの配信

電波状況が悪くなったり，電源が切れたりするとスマートフォンと通信を行えず，システムに接続されているスマートフォンを把握できない．このため，提案フレームワークでは，サーバからスマートフォンにスマートフォンプログラムを配信するプッシュ型の配信ではなく，スマートフォンがサーバからスマートフォンプログラムをダウンロードするプル型の配信を用いる．スマートフォンプログラムのダウンロード等の管理を行うソフトウェアを**スマートフォンプログラムマネージャ**と呼ぶ．

システムに接続するスマートフォンの利用者は，公開されているスマートフォンプログラムマネージャをスマートフォンにインストールする．スマートフォンプログラムマネージャの初回起動時に，認証後，サーバのアドレスを指定してスマートフォンプログラムをダウンロードする．サーバのアドレスは，インターネットで公開されていたり，システム開発者から聞くといった方法で取得できる．スマートフォンプログラムが更新された場合に，最新のスマートフォンプログラムをダウンロードできるようにするため，スマートフォンプログラムマネージャは，定期的にサーバにスマートフォンプログラムの更新の有無を問い合わせ，更新があればダウンロードする．

スマートフォンプログラムマネージャと通信してスマートフォンプログラムの配信や更新情報送信を行うソフトウェアを**プログラムマネージャ**と呼ぶ．近年，セキュリティ確保のために設置されているファイアウォールを越えるために，WebDAV や REST といった Web サーバを用いた通信がよく用いられている．そこで，提案フレームワークにおいても Web サーバを用い，Web インタフェースでプログラムマネージャの機能を提供する．プログラムマネージャを用いて利用者やスマートフォンの登録・削除も行う．

3.2　スマートフォンプログラムの実行

スマートフォンプログラムは，連続的に発生するセンサデータを処理するため，常時動作していることで即座に処理できるが，常に動作していると消費電力が大きくなる．そこで，システム開発者が指定した間隔で定期的にスマートフォンプログラムを呼び出す．定期的に呼び出す間隔は，長すぎると，発生するセンサデータに対して処理が間に合わず，短すぎると消費電力が大きくなるため，センサデータを処理し切れる範囲で短くなりすぎないように設定する．定期的に呼び出す際，コールバック関数の設定や，ファイルの作成といった処理を初めに一度だけ実行できるように，初めに呼び出すときのみ実行する処理を記述できるようにする．間隔を 0 にしてスマートフォンプログラム自体を無限ループにすることで，スマートフォンの利用者が停止するまで常時スマートフォンプログラムを動作させられる．

3.3　スマートフォン利用者のプライバシーへの配慮

スマートフォンのセンサから利用者の状況を確認できる．例えば，加速度センサの値から停止中や歩行中，車内といった状況を確認できる．スマートフォンの利用者にとって不都合であれば，これらのセンサデータを送信しないように設定できる必要がある．そこで，提案フレームワークでは，スマートフォンプログラムマネージャにおいて，スマートフォンプログラムが利用できるセンサや電話番号等のスマートフォンの情報を制限する．

3.4　サーバプログラムの実行

スマートフォンから処理したセンサデータが送信されると，プログラムマネージャはサーバプログラムにデータを渡す．何回かセンサデータが送信されてからまとめてサーバプログラムを実行することも考えられるが，連続的に発生するセンサデータを処理するためには，早く処理できる方が望ましい．そこで，提案フレームワークでは，センサデータが送信される度にサーバプログラムを実行する．サーバ側で受信したセンサデータをファイルで保存等することで，複数回分のセンサデータをまとめて処理することもできる．サーバプログラムで処理した結果，必要があればセンサデータベースに処理結果を保存する．

4. 提案フレームワークの実装

スマートフォンの OS として Android 2.3 を用い，サーバの OS として CentOS 6.0 を用いて提案フレームワークの実装を行った．スマートフォンとサーバの通信には REST を用いた．REST は，HTTP プロトコルを用いてデータを送受信する方式であり，ファイアウォールを越えやすいという特徴がある．

4.1 スマートフォンプログラム

スマートフォンプログラムは，Java で記述して Android アプリケーションパッケージとしてコンパイルしたものをサーバに保存する．サーバは，スマートフォンプログラムマネージャからスマートフォンプログラムのダウンロード要求があると，スマートフォンにスマートフォンプログラムを配信する．スマートフォンプログラムマネージャがスマートフォンプログラムを実行する際，スマートフォンプログラムの Java クラスを呼び出す．このクラスには以下のメソッドを実装した．

getStartDate　スマートフォンプログラムの実行開始日時を戻り値とする．この日時より前であれば，スマートフォンプログラムを実行しない．

getEndDate　スマートフォンプログラムの実行終了日時を戻り値とする．この日時より後であれば，スマートフォンプログラムを実行しない．

getInterval　3.2 節に記述したように，提案するフレームワークでは，定期的にスマートフォンプログラムを実行する．この間隔を getInterval メソッドの戻り値とする．

onStart　getStartDate メソッドで返す時刻になってスマートフォンプログラムを開始する時に呼び出され，実行開始時の処理を記述する．引数には，Android 自体のリソースへの識別子である context が含まれる．

onRun　スマートフォンプログラム実行時に定期的に呼び出され，実行内容を記述する．正常に実行されれば真を，異常終了すれば偽を戻り値とする．

onDestroy　getEndDate メソッドで返す時刻になってスマートフォンプログラムを停止する時に呼び出され，実行停止時の処理を記述する．正常に実行されれば真を，異常終了すれば偽を戻り値とする．

getStatus　スマートフォンプログラムの状況を戻り値とする．スマートフォンプログラムを実行中であれば真を，停止中であれば偽を返す．

Android2.3 では，Manifest ファイルと呼ばれる XML ファイルに記述することで，プログラムの機能に制限を設けられる．例えば，電話発信や電話帳の読み込みを禁止したりできる．Manifest ファイルで，位置情報取得や通信の利用といったセンサデータを取得するために必要最低限の機能のみ許可することで，スマートフォンのセキュリティを確保する．

4.2 サーバプログラム

サーバプログラムは，スマートフォンプログラムと同じく Java で記述して Java クラスとしてコンパイルしたものをサーバに保存する．プログラムマネージャがサーバプログラムを実行する際，ユーザ ID，スマートフォン ID，送信データを引数としてサーバプログラムの onRun メソッドを呼び出す．ユーザ ID は，システムに登録された利用者の識別子であり，スマートフォン ID は，スマートフォンの識別子である．提案するフレームワークでは，ユーザ ID とスマートフォン ID を紐付けて，システムに接続するスマートフォンを認証する．これらの ID はプログラムマネージャで管理される．送信データは，スマートフォンプログラムから送信されたデータである．REST を用いているため，文字列で送られる．センサデータベースへの保存処理を容易に行うため，このクラスの戻り値として，センサの種類，センサデータ取得時刻，センサデータの三つを組みにした配列リストを返すと，それらの値をセンサデータベースに格納する．

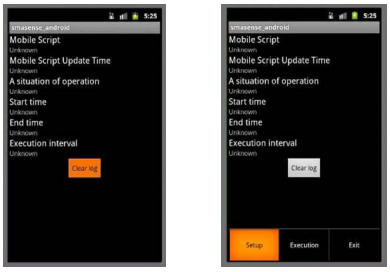

図 2　スマートフォンプログラムマネージャの初期画面
Fig. 2　A initial screen of smartphone program manager

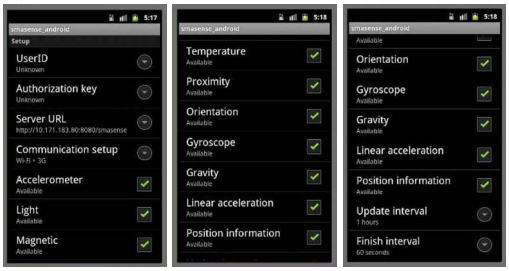

図 3　スマートフォンプログラムマネージャの設定画面
Fig. 3　A setting screen of smartphone program manager

4.3 スマートフォンプログラムマネージャ

スマートフォンプログラムマネージャの初期画面のスクリーンショットを図 2 に示す．左側が起動直後の画面であり，右側がメニューボタンを押した場合の画面である．スマートフォンの利用者がスマートフォンプログラムの詳細を確認できるように，ダウンロードしたスマートフォンプログラムの名前や更新時刻，動作状況を表示する．また，スマートフォンプログラムの実行を開始する時刻や終了時刻，実行間隔も表示している．エラー等がある場合には，下部に表示される．起動画面でメニューボタンを押すと，右側に示すメニューが表示される．Setup を押すと設定画面を表示でき，Execution を押すとスマートフォンスクリプトの実行を開始する．Exit を押すと終了する．

設定画面のスクリーンショットを図 3 に示す．設定できる項目を表 1 に示す．ユーザ ID や認証キーは，システムを利用するスマートフォンを認証するために用いる．プログラムマネージャを用いて利用者および利用者が利用するスマートフォンを登録することで作成できる．スマートフォンの利用者は，登録

表 1　設 定 項 目
Table 1　Setting items

設定項目	説明
UserID	ユーザ ID を設定する
Authorizationkey	認証キーを設定する
Server URL	サーバの URL を設定する
Communication	通信手段（3G，WiFi，または両方）を設定する
Update Interval	スマートフォンプログラムの更新を確認する間隔を設定する
Finish Interval	スマートフォンプログラムの動作状況を確認する間隔を設定する
Accelerometer	加速度センサのデータ取得可否を設定する
：	各種センサデータの取得可否を設定する

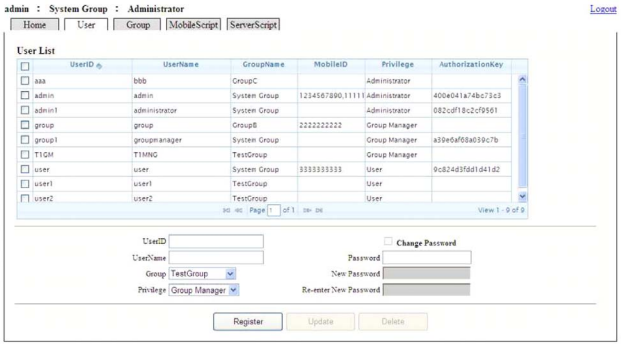

図 4　プログラムマネージャの画面
Fig. 4　A screenshot of program manager

時にシステムの開発者からこれらを入手してスマートフォンプログラムマネージャに入力する．利用者がスマートフォンの通信料金を支払うため，3G 回線による通信を行いたくないことが考えられるため，3G，WiFi，または両方のどの通信手段を用いるか設定できるようにした．スマートフォンプログラムの更新を確認する間隔や，初期画面に表示されるスマートフォンプログラムの動作状況を確認する間隔も設定できる．また，利用者のプライバシーを配慮して，加速度センサや位置情報といったスマートフォンプログラムが利用できるデータを設定できる．スマートフォンプログラムの実行を開始すると，これらの設定に基づいてスマートフォンプログラムマネージャが定期的にスマートフォンプログラムを呼び出す．ただし，前に呼び出したときの処理がまだ続いている場合には新たに呼び出さない．

4.4　プログラムマネージャ

プログラムマネージャにおいて，スマートフォンプログラムの配信などの管理を行うため，Web インタフェースを用いたプログラムマネージャを実装した．プログラムマネージャのスクリーンショットを図 4 に示す．図 4 では，利用者の登録画面を表示している．プログラムマネージャには大きく四つの機能がある．まず，システムの利用者やスマートフォンを登録し，認証に必要なキーを発行できる．この認証キーを用いてスマートフォンの認証を行える．利用者のメールアドレスや氏名も登録できるが，必須ではない．システムを使わなくなった利用者やスマートフォンの削除もできる．

次に，プログラムマネージャはスマートフォンプログラムの登録を行える．ただし，Java で記述されたスマートフォンプログラムのコンパイルには，ライブラリの依存関係の解決など複雑な場合があるため，コンパイルされアプリケーションパッケージとなったスマートフォンプログラムを登録することとした．登録されたスマートフォンプログラムは，スマートフォンプログラムマネージャの要求に応じて配信される．また，プログラムマネージャはサーバプログラムの登録を行える．スマートフォンプログラムと同じく，コンパイルされたサーバプログラムを登録する．登録されたサーバプログラムは，スマートフォンプログラムからデータが送信されると，実行される．

```
・・・略・・・
public class SmartphoneProgram extends Activity implements SensorEventListener {
SensorEvent AC, MA, OR, GY, LI, PRE, TE, PRO;
   private boolean sendFlg = false;
   public Boolean onStart(Context ctx) {
・・・略：センサデータが変わったときに発生するonSensorChangedリスナーの登録・・・
   }
   public void onDestroy() {
・・・略：onSensorChangedリスナーの解放・・・
   }
   public Integer getInterval() {      return 60;      }
   public void onRun() {      sendFlg = true;      }
   public void onSensorChanged(SensorEvent event) {
      int type=0;
      type=((android.hardware.SensorEvent)event).sensor.getType();
      switch(type) {
         case Sensor.TYPE_ACCELEROMETER:   AC=event;   break;
         case Sensor.TYPE_MAGNETIC_FIELD:   MA=event;   break;
         case Sensor.TYPE_ORIENTATION:   OR=event;   break;
         case Sensor.TYPE_GYROSCOPE:   GY=event;   break;
         case Sensor.TYPE_LIGHT:   LI=event;   break;
         case Sensor.TYPE_PRESSURE:   PRE=event;   break;
         case Sensor.TYPE_TEMPERATURE:   TE=event;   break;
         case Sensor.TYPE_PROXIMITY:   PRO=event;   break;
      }
      if (sendFlg) {
・・・略：サーバにセンサデータ送信・・・
         sendFlg = false;
      }
   }
・・・略・・・
}
```

図 5　スマートフォンプログラムの例
Fig. 5　An example of a smartphone program

最後に，REST サービスを提供してスマートフォンプログラムからデータを受信できる．スマートフォンプログラムは，処理したデータを HTTP プロトコルの GET または POST メソッドを用いてプログラムマネージャに送信すると同時に，ヘッダー部分で利用者 ID とスマートフォンの ID も送信する．これにより，誰のどのスマートフォンからデータを受信したのか判断できる．REST を用いているため，スマートフォンプログラムからだけではなく，他の REST に対応したプログラム等からでも処理したセンサデータを送信できる．

4.5　具　体　例

本章では，提案するフレームワークを用いたシステムの簡単な例として，スマートフォンからセンサデータを取得してサーバに送り，サーバはデータベースに保存するシステムを考える．全てを記述すると長くなるため，この例のスマートフォンプログラムの内容一部を図 5 に示す．Android では，センサデータを取得するために，センサデータが変化する度に発生する onSensorChanged リスナーを登録する必要がある．この登録をスマートフォンプログラム開始時に行い，終了時に解放する．スマートフォンプログラムの実行間隔は 60 秒であり，60 秒ごとに sendFlag が ON になる．センサデータが変化すると onSensorChanged 関数が呼び出され，各種センサデータを保持

```
・・・略・・・
public class ServerProgram {
  public List<String[]> ServerScript(String userID, String smarID, String sensordata){
    String[] record=new String[3];
    List<String[]> r=new ArrayList<String[]>();   /*戻り値用*/
    String[] param=sensordata.split("&");   /*paramsを&毎に分解*/
    SimpleDateFormat Format = new SimpleDateFormat("yyyymmddHHmmssSSS");
    for(int i=0;i<param.length;i++){   /*センサデータ毎にレコードを作成*/
      String[] namevalue=param[i].split("=");
      record[0]=namevalue[0];
      record[1]=Format.format(Calendar.getInstance().getTime());
      record[2]=namevalue[1];
      r.add(record);
    }
    return r;
  }
}
```

図 6　サーバプログラムの例
Fig. 6　An example of a server program

する．sendFlag が ON であれば，データをサーバに送信する．

サーバプログラムの内容を図 6 に示す．引数 params に渡されたスマートフォンプログラムから送信されたデータを，センサデータ毎に分解してレコードを作成している．作成したレコードを戻り値とすることで，これらの値がデータベースに格納される．

5. 議　　論

5.1 特　　徴

1 章で述べたように，提案するフレームワークでは，用途ごとに異なる処理については変更できるように，スマートフォンプログラム，サーバプログラムで処理を記述する．スマートフォンプログラムとサーバプログラムを変更することで，提案するフレームワークは，様々な用途に使用できる．スマートフォンでセンサデータを処理してサーバに送信し，サーバが送信されたデータをさらに処理するといった手順が同じ部分については提案するフレームワークのスマートフォンプログラムマネージャやプログラムマネージャで実現しており，開発の手間を軽減できる．Dandelion のような既存のフレームワークは，スマートフォンを対象としておらず，定期的なスマートフォンプログラムの実行や，利用者のプライバシーへの配慮を行えていない．

5.2 拡 張 性

提案するフレームワークでは，スマートフォンの OS として Andoroid 2.3，サーバの OS として CentOS 6.0 を用いている．スマートフォンプログラムについて，Android 自体のリソースへの識別子である context を引数で渡している．このため，Manifest ファイルで許可されたデータや処理であれば，OS の提供する機能をスマートフォンプログラムで使用できる．例えば，画面の明るさを変えたり，音を鳴らすといった処理も記述でき，拡張性は十分あるといえる．サーバプログラムについても，Java で CentOS の機能を使ったプログラムを作成でき，様々な処理を記述できる．

しかし，提案するフレームワークでは，スマートフォンでセンサデータを処理してサーバに送信してさらに処理するといった手順に基づいているため，スマートフォンで処理したセンサデータをさらに他のスマートフォンに送信したり，サーバからスマートフォンにデータを送信するといった処理を記述するには手間がかかる．このような手順の異なる処理の記述を軽減するためには，単純にフレームワークに機能を追加して手順の異なる処理も記述できるようにすることや，手順に応じて異なるフレームワークを用いることで対応できる．

6. ま と め

本研究では，開発手間を軽減してサーバの処理負荷軽減や通信料削減を容易に実装できるように，センサデータを処理して収集するセンサデータ処理収集フレームワークを提案した．提案フレームワークでは，スマートフォンで実行するスマートフォンプログラムとサーバで実行するサーバプログラムを開発者が準備し，これらのプログラムの配信や実行を行う部分をフレームワークとして提供する．多くのシステムで行われている同様の手順をフレームワークで実現することで，センサデータ処理収集システムの開発手間を軽減できると考えられる．

今後，スマートフォンをグループ化してグループごとに実行するスマートフォンプログラムを変えることや，複数のスマートフォンプログラムやサーバプログラムを実行できるように拡張することを考えている．

謝　　辞

本研究の一部は，科学研究費補助金基盤研究（S）「モバイルセンサネットワークのための効率的なデータ処理機構に関する研究」（課題番号：21220002）の研究助成による成果である．ここに記して謝意を表す．

文　　献

[1] Yohan Chon and Hojung Cha “LifeMap: A Smartphone-Based Context Provider or Location-Based Services”, IEEE PERVASIVE computing, pp. 58-67, 2011.

[2] Jeongyeup Paek, Kyu-Han Kim, Jatinder P. Singh, and Ramesh Govindan “Energy-Efficient Positioning for Smartphones using Cell-ID Sequence Matching”, Proc. Int’l Conf. on Mobile Systems, Applications and Services (MobiSys’11), pp. 293-306, 2011.

[3] Peng Zhuang, Dan Wang, and Yi Shang “SMART: Simultaneous Indoor Localization and Map Construction Using Smartphones”, Proc. of Int’l Joint Conf. on Neural Networks (IJCNN’10), pp. 1-8, 2010.

[4] Hyojeong Shin, Yohn Chon, and Hojung Cha “Unsupervised Construction of Indoor Floor Plan using Smartphone”, IEEE Transactions on Systems Man and Cybernetics, 2011.

[5] Matthew Keally, Gang Zhou, Guoliang Xing, Jianxin Wu, and Andrew Pyles “PBN: Towards Practical Activity Recognition Using Smartphone-Based Body Sensor Networks”, Proc. ACM Conf. on Embedded Networked Sensor Systems (SenSys’11), 2011.

[6] Felix Xiaozhu Lin, Ahmad Rahmati, and Lin Zhong “Dandelion: A Framework for Transparently Programming Phone-Centered Wireless Body Sensor Applications for Health”, Proc. Wireless Health, pp. 74-83, 2010.

[7] Yuto Hamaguchi, Tomoki Yoshihisa, Yoshimasa Ishi, Yuuichi Teranishi, Takahiro Hara, and Shojiro Nishio “A Data Aggregation System using Mobile Agents on Integrated Sensor Networks”, Proc. Int’l Conf. on Advances in P2P Systems (AP2PS’11), pp. 33-38, 2011.