モバイルクラウドセンシングを用いた
リアルタイムデータ収集アーキテクチャの提案

2011SE084 伊藤 大貴　2011SE135 小島 一輝
指導教員　青山 幹雄

1　はじめに

近年，交通事故の対策として高度道路交通システム（ITS：Intelligent Transport Systems)などの安全運転のための技術が研究されている[2]. しかし，事故防止技術は進んでいるものの出会い頭事故の割合は高く，死亡率は交通事故の中でも最も高い[3].

本研究では，出会い頭事故の軽減を目的とした事故防止技術として，モバイルクラウドセンシング利用したリアルタイムにおけるデータ収集のアーキテクチャを提案する.

2　研究課題

近年のスマートフォンの普及に伴い，モバイルクラウドセンシング（MCS：Mobile Crowd Sensing）を用いたアプリケーションは公共交通機関や個人，企業に利用されている[4]. 本研究では，交差点での出会い頭事故の危険を予測するために，MCS を用いてスマートフォン利用者からリアルタイムに位置情報を収集する. しかし，各端末が位置情報を収集してからユーザへ通知するまでの時間は保証されていない. リアルタイムデータ収集を効率よく行うためには，データ収集に必要な処理時間を保証する必要がある.

3　関連研究

3.1　MCS

MCS とは，センサデバイスを用いてユーザや物理世界についてのデータを収集，共有するアプリケーションである. MCS のクラウドとはクラウドコンピューティングで使用されているサーバ群の Cloud ではなく，多くの人を意味する Crowd(群衆，大衆)である. MCS はデータの送受信を非同期で行い，ユーザが意識することなく，センサデバイスからセンサデータをブローカに送信する.

4　アプローチ

本研究では，MCS を用いたリアルタイムデータ収集の性能を保証する.

4.1　データ収集の定義

本研究において，データ収集の性能をデータ収集に必要な処理時間とする. 処理時間とはデータ収集からユーザへ通知されるまでの時間である.

4.2　データ収集の保証

データ収集に必要な処理時間を保証するために，人や環境などの状況の変化に対応することができるコンテキストアウェアの概念に着目した. コンテキストアウェアを用いてデータ収集アーキテクチャを提案する.

5　提案方法

5.1　設計プロセス

MCS の概念を利用した，データ収集アーキテクチャの設計プロセスを以下に示す(図 1).

提案方法は，自身の位置情報を利用して，デバイスからセンサデータを他のユーザのデバイスに送信し，再び自身のデバイスがセンサデータを受信することで，交通事故の危険通知を可能にする.

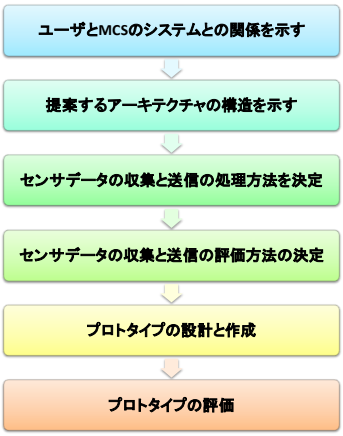


図 1 設計プロセス

5.2　前提条件

MCS におけるユーザとシステム間の関係を述べる. 相互に情報の交換を行うため，送信者は受信者の役割も意味し，受信者は送信者の役割も意味する. また，送信者と受信者はMCS アプリケーションをインストールしたデバイスを所持しているものとする.

5.3　データ収集のアーキテクチャ提案

データ収集に必要な処理時間を保証するため，MCS とコンテキストアウェアの連携モデルに基づいたアーキテクチャの構造を示す(図 2). 本研究ではデータの収集に着目しているため，ユーザとアプリケーション間での処理時間を中心として提案する.

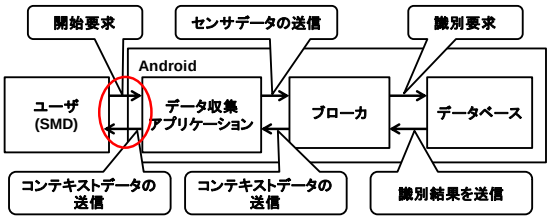


図 2 提案アーキテクチャ

5.4 データ収集の流れ

提案アーキテクチャの流れは，データ収集モードとデータ送信モードの 2 つに分けられる．本研究では，データ収集モードに着目する．MCS の特徴は非同期でデータ通信を行うことである．しかし，どこからでもセンサデータを収集できるわけではなく，MCS を利用してデータ収集に参加するユーザのみからセンサデータを収集する．

5.4.1. データ収集モード

複数のユーザの要求に対し，そのデータ群の中からユーザに応じたデータを収集し，識別するモード．ユーザが意識することなく，非同期的に行う．そのため，常にアプリケーションがセンサデータをブローカへ送る．本研究のデータ収集モードはユーザが求める情報を持つデータを識別し，その結果を返すまでの処理とする(図 3)．

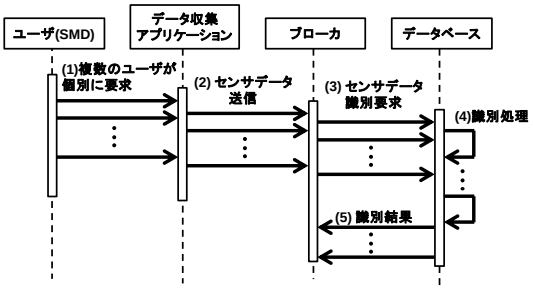


図 3 データ収集モードのシーケンス図

(1) ユーザはアプリケーションに対し，非同期で 1 人 1 人が個別にデータを要求することができる．
(2) アプリケーションは各ユーザの情報を持つセンサデータをブローカに送信する．この時，センサデータは個別に送信される．
(3) 収集されたデータから，各ユーザが求める情報を持つセンサデータを識別するため，データベースに識別要求を行う．この処理もセンサデータ毎に個別で行なわれる．
(4) 収集されたセンサデータからユーザの求める情報を持つセンサデータを識別する．この処理は必要なセンサデータが現れるまで行い続ける．
(5) 識別処理により，ユーザの求める情報を持つセンサデータを発見した場合，識別結果をブローカへ返す．

5.4.2. データ送信モード

本研究では，データの収集について着目しているが，プロトタイプの作成に当たってデータの識別方法を決める必要があるため，データの送信の流れのみを図 4 で示す．

データ収集モードによって識別されたデータを，各ユーザに送信するモード．この時，データは 1 人 1 人個別に送信するのではなく，データ収集に参加したユーザ全員に同時に送信する．

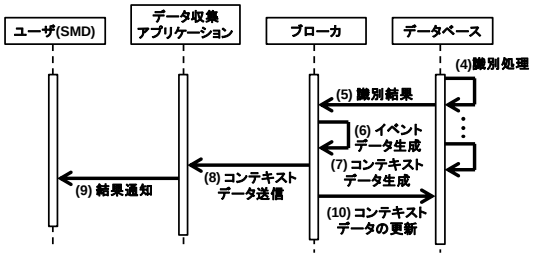


図 4 データ送信モードのシーケンス図

(4) 大量のセンサデータからユーザが求める情報を持つセンサデータを識別する．必要なデータが現れるまで処理を行い続ける．
(5) 識別によりユーザの求める情報を発見した場合に識別結果を返す．
(6) 識別されたデータを元にイベントデータの生成を行う．イベントデータとは現在のユーザの状況をリアルタイムに示したものである．
(7) イベントデータを元にコンテキストデータの生成を行う．
(8) コンテキスデータをアプリケーションに送る．
(9) アプリケーションはコンテキストデータを元にユーザに結果を通知する．
(10) ブローカは最後に新たなセンサデータの収集のためコンテキストデータの更新を行う．

6 提案アーキテクチャのプロトタイプ

作成するプロトタイプのシーケンス図とアクティビティ図を示す(図 5)(図 6)．

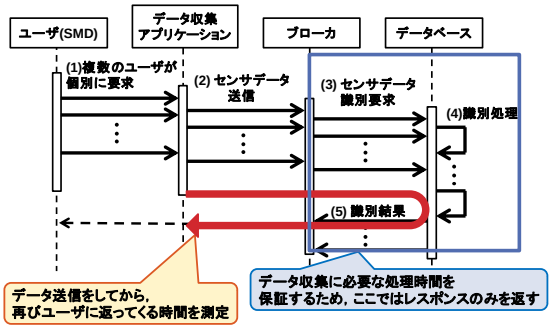


図 5 プロトタイプのシーケンス図

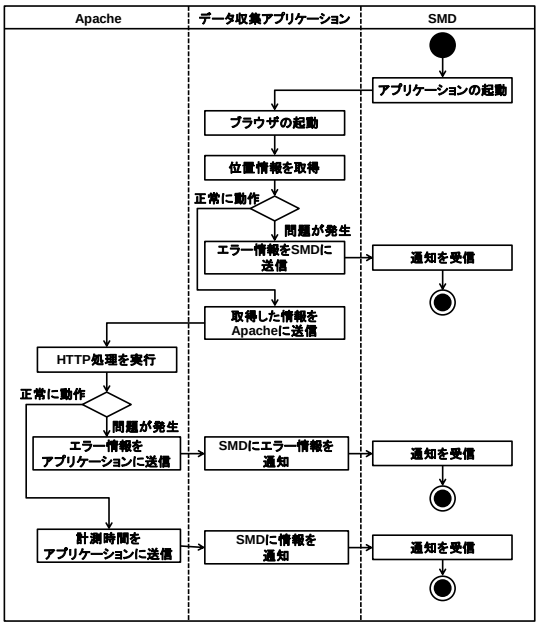
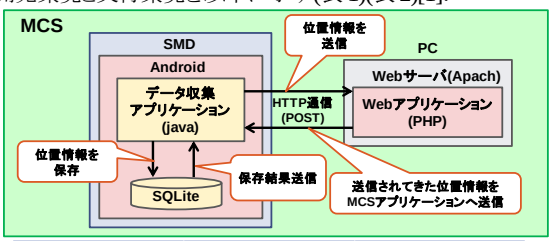


図 6 プロトタイプのアクティビティ図

6.1 プロトタイプの構成と仕様

提案アーキテクチャにおけるサーバへの POST 送信機能の確認，評価を行うため，プロトタイプを開発する．以下にプロトタイプの構成図を示す(図 7)．また，プロトタイプの開発環境と実行環境を以下に示す(表 1)(表 2)[1].



実装したアプリケーション	実装言語	LOC
Webアプリケーション	PHP	3
データ収集アプリケーション	Java	504
	XML	202
合計		709

図 7 プロトタイプの構成図と開発規模

(1) SQLite

アプリケーションが取得した位置情報が保存されるデータベースである．アプリケーションによって操作される．

(2) データ収集アプリケーション

プロトタイプでは java で実装する．データ収集アプリケーションは，SQLite に位置情報の保存，削除，閲覧を可能とし，SQLite に保存されている位置情報を Web サーバへの POST 通信を行う．

(3) Web サーバ

プロトタイプでは Apache サーバを構築し，XAMPP Control Panel で操作を行う．

(4) Web アプリケーション

データ収集アプリケーションから POST 送信されてきた位置情報を受信し，表示する．

表 1 プロトタイプの開発環境

	データ収集アプリケーション	外部システム
OS	Windows 7 Professional	Windows 7 Professional
CPU	Intel(R) Core(TM) i5 2.67GHz	Intel(R) Core(TM) i5 2.67GHz
メモリ	4.00GB	4.00GB
JDK	JDK 1.8.0_25	
EclipsesSDK	Eclipse 4.4	
Apache		Apache v3.2.1
AVD	Android 4.4	

表 2 プロトタイプの実行環境

デバイス	Nexus7
OS	Android 4.3 JellyBean
プロセッサ	Snapdragon 54 Pro(APQ8064) 1.5GHz クアッドコア
RAM	2GB
Wi-Fi	802.11a/b/g/n デュアルバンド

6.2 プロトタイプのデータ処理

データ収集アプリケーションと Web サーバ間で行われるデータ処理を図 8 で示す．

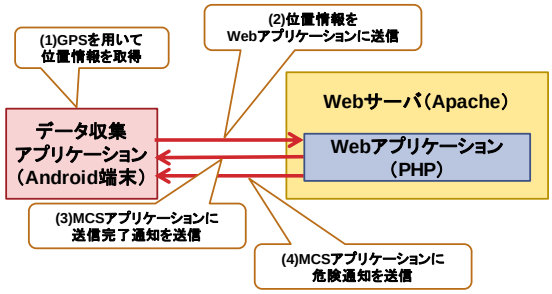


図 8 プロトタイプのデータ処理

(1) Android 端末内のアプリケーションが GPS を用いて位置情報を取得する．

(2) 取得した位置情報はデータ収集アプリケーションを通して Web アプリケーションへ送信する．
Web アプリケーションは Web サーバを用いる．

(3) Web アプリケーションはデータ収集アプリケーションから位置情報を取得し，送信完了通知を返す．

(4) データ収集アプリケーションへ危険を通知する．

7 例題への適用と評価

7.1 プロトタイプの評価基準

本研究では，出会い頭衝突を防止することを前提としており，交差点までの距離がどの程度で通知するべきか考える必要がある．

通知に必要な時間が評価時間より短いほど，出会い頭衝突防止に有効であると言えるが，現段階では通知にかかる

時間がわからず，評価をすることができない．そのため，実際に検証を行い，通知にかかる時間を測定してから交差点までの距離を決定する[5].

しかし，評価を行うに当たって，評価時間を定める必要があり，仮定として 100m を基準として検証を行う．プロトタイプの評価基準を以下で示す(図 9).

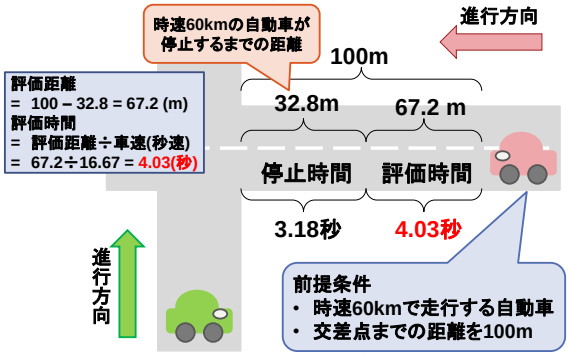


図 9 プロトタイプの評価基準

7.2 検証結果

データ収集アプリケーションで，送信ボタンを押した際に表示される経過時間を 1 秒ごとに 1 分間測定した結果，平均時間:0.093(秒)，標準偏差:0.032 となった(図 10).

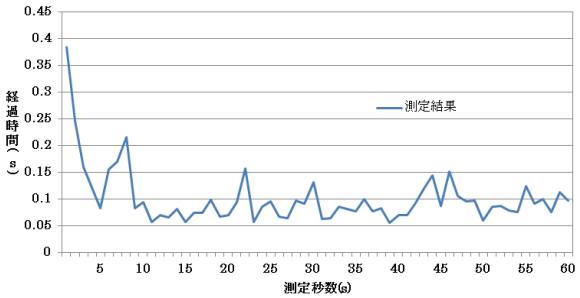


図 10 測定結果のグラフ

測定秒数 1-2 秒間の経過時間の値は，3-60 秒間までの値より大きい値となっている．理由としては，プログラムファイルの置かれている URL にデータ収集アプリケーションがアクセスする時，サーバ内部で Web アプリケーションを起動しているためである．本研究では，例外値である 1-2 秒間の経過時間の値を平均時間及び標準偏差の計算から除外する.

8 考察

検証の結果，平均送信時間は 0.093 秒となり，プロトタイプの評価基準で示した 4.03 秒以内という基準を満たす結果となった．しかしプロトタイプの評価基準は，時速 60km で走行する自動車が交差点から 100m の地点に位置していると仮定した場合であり，今回の検証結果から改めて交差点までの距離を決定し，評価基準を定める必要がある.

改めて交差点までの距離を決定する．求めた平均送信時間の誤差をなるべく小さくするため，送信時間を平均値 $\pm 3\sigma$ を用いて決定する．$0.09+3\sigma=0.18$ となるので，これを送信時間とする．この送信時間から，停止距離までの距離は 3.0m となる．したがって交差点までの距離は，データの送信時間に要する距離 3.0m+自動車の停止までの距離 32.8m=35.8m となる.

計算の結果，時速 60km で走行し続ける自動車が交差点から 35.8m 以上離れた位置で危険通知を受け取ることで，出合い頭事故を回避することが可能であると言える．そして交差点までの距離が 35.8m 以上である場合，提案したプロトタイプは評価基準を満たしていると言える.

本研究で提案したアーキテクチャにおいて上記の評価基準は，時間に対するデータ収集の性能を保証できる.

9 今後の課題

今後の課題として以下のことが挙げられる.

(1) 非同期通信での検証とプロトタイプ作成
(2) 収集データの共有の実装

10 まとめ

本稿では，出会い頭事故の軽減を目的とした事故防止技術として，MCS を用いたリアルタイムデータ収集アーキテクチャを提案した.

アーキテクチャ提案では，HTTP 通信により位置情報を送信するため，Web サーバである Apache を用いて，HTTP 通信の POST メソッドを介して位置情報を送信するプロトタイプを実装し，アーキテクチャの妥当性を示した.

データ収集に必要な処理時間を保証するため，データ収集アプリケーションをプロトタイプで実装し，Web アプリケーションへの送信時間を測定することで，通知可能な交差点までの距離の評価基準を定めた．これにより，本アーキテクチャにおいてデータ収集に必要な処理時間を保証した.

11 参考文献

[1] 秋元 累，五十川 雄太，プローブデータ解析のためのアーキテクチャ提案，南山大学2013年度卒業論文，2014.
[2] デンソー情報安全ジステム開発部，インフラ協調運転支援システム開発，2008，http://www.soumu.go.jp/main_sosiki/joho_tsusin/policyreports/chousa/its/pdf/081107_2_si1-5.pdf.
[3] 交通事故統計年報 平成 25 年版，公益財団法人 交通事故総合分析センター，2014.
[4] H. Lei, and F. Ye, Mobile Crowd Sensing and Context-Aware Real-Time Data Fusion in MCS Applications, 2011.
[5] ロバート・ボッシュ，ボッシュ自動車ハンドブック，2005