非同期通信を利用したWeb上のマップ描画による情報共有システムの設計

廣瀬研究室 4 年 c1151073 鈴木光明

平成 30 年 1 月 15 日

概要

近年、GPS 内蔵型携帯電話等の位置情報を計測できる情報端末が広く普及している。しかし、近年の熊本地震や北海道胆振地震をはじめとする災害時の行方不明者や災害弱者の逃げ遅れの状況から見える通りその機能が有効に活用されていないのが現状である。現在位置情報を利用したガイドシステムは存在するが、利用者同士のコミュニケーションや情報共有が念頭に置かれていない場合が多い。また、従来災害時に用いられている伝言板等のシステムでは携帯電話と PC との連携が取れず、避難時に災害時の状況を確認できる手段がないことから避難者の立場が考慮されていないことが問題視されている。

そこで、個人単位で位置情報を発信できる点に着目し、災害時における避難者の位置情報の共有の簡易化を図る Web アプリケーションを構築する。また、外部からの匿名性を保持した上で、設定したグループ内における個人間での情報の相互共有を可能にし、災害時に家族等の複数人のコミュニティ内で活用できるような機能も構築する。また、その機能を応用し、フィールドワーク等複数人の位置情報を相互に確認したいような場面でも利用できるようなシステムを構築する。(491 文字)

目 次

第1章　目的・テーマ設定

本研究の設定する目的とテーマ設定に至るまでの背景について記述する。

1.1　テーマ設定の背景

日本では近年に見られる熊本地震や北海道胆振東部地震をはじめとした、地震や大雨による洪水等、 多数の被害者が出るような災害が頻繁に起こっている [1]。その中でも高齢者をはじめとする災害弱者の逃げおくれによる被害が問題となっている [2]。他にも、電話や交通等のライフラインが混雑するため避難後の安否確認が困難になる [3]。そこで近年では避難経路の確保や把握等の日常的な活動を行うことが促進されている。しかし、事前の準備のみならず実際に災害時の状況に陥った際の対応策が必要であると考えた。そこで、各自が持っている位置情報を発信できる情報端末を利用することで被災地の人間の逃げ遅れの確認や、家族等複数人でのコミュニティ内での情報の相互共有をする手段の必要性を感じた。

また、その他にも野外でのフィールドワークを複数人で行う際や集団のツアー客の自由行動等、予め時間や集合場所を決めていたとしても集団の動向を一目で把握できないのは不便な状況が存在する。宗森らによると既存の位置情報を使ったガイドシステムでは、その場での交換やチャットなど、利用者同士のコミュニケーションや情報共有は念頭に置かれていない場合が多いとされている [4, 5]。また、Apple Store や Google Play から配布されているアプリケーションの中にも、位置情報を不正に取得している例が確認されている [6]。

さらに、Google Play で nekonotesoft から配信されている情報共有ソフト “いまどこ?位置検索” 等をはじめとした無料のアプリケーションでは共有できる数に制限があったり全ての機能を利用するには有料であったりといったケースが存在する。例として上述の “いまどこ?位置検索” では 14 日間の試用期間では「自動測位機能」および「見守り機能」を利用することができず、また 14 日目以降の利用にはライセンスの購入と月額料の支払いが必要になる [7]。

そこで、特別な外部のアカウントや電話番号等の登録情報を必要としない、 参加者の状態を個人単位で制限なくリアルタイムで相互に把握することができるアプリケーションが必要だと考えた。

1.2　過去の類似研究の例

本研究では災害時に複数人の位置情報を取り扱うことに重点を置くため、過去の研究の中から災害時の位置情報や複数人の位置情報を扱う取り組みが行われた文献について記述する。

松崎らは、スマートホームシステムのホームサーバ [1]を利用して地震災害時における被災状況の確認や被災者の救助支援に応用するシステムを提案している [8]。モバイル端末とホームサーバの通信によりユーザとホームサーバとの距離を観測することで、在宅情報から生き埋め等の

[1]家庭内に存在する家電製品や防犯システム等を一律に制御するシステムである。ホームサーバとはその各種センサを制御されているサーバのことである。

被災者を特定し、救助要請マップを作成してシミュレーション実験を行い、研究の有効性を示した。しかし、ここではホームサーバごとに管理しているため、人数に誤差が出ることや使用する多数の機器に対し電力を供給し続けなければならないことが問題となっている。

また、複数人の位置情報を用いた遠隔地での情報共有に関して、宗森らは PDA(携帯情報端末) と GPS と携帯電話を使用し、位置情報を用いて遠隔地間で行う鬼ごっこをはじめ複数種の電子鬼ごっこの実験を行っている [4]。位置情報を変換することで離れた場所でも問題なく電子鬼ごっこを行い、実験の結果では位置情報を柔軟に扱ったサービスが受け入れられている。

第2章　提案

現状抱えている問題点を踏まえて、本研究で提案するシステムについて記述する。

2.1　現状

災害時等を想定した際の離れている人同士の安否確認等の情報の相互確認についての現状を記述する。

2.1.1　災害時の逃げ遅れによる被害

渡邊らは、災害時に住民の孤立や逃げ遅れの二次災害による犠牲者の発生を問題とし、最短避難経路提示システムを開発した [9]。そこでは、従来の災害時の伝言板等のシステムでは携帯電話と PC との連携がとれず、避難者の立場が考慮されていないことが問題視されていた。そこで、避難所や交差点に ID とグループ番号を付加することで、携帯電話を活用し最短の避難経路に誘導するシステムを実現した。また、GPS 内蔵型携帯電話が使用できない状況では Wi-Fi 付きノート PC を用い、Google 社から Wi-Fi のアクセスポイントと IP アドレスを基に測位された現在地を利用することで実測値との誤差を 5%以下で実現した。ここから、災害時であっても正確に位置情報を発信・共有可能であることが示されている。

2.1.2　避難後の安否確認

能島によると 2016 年の熊本地震におけるライフラインの復旧に関して、阪神淡路大震災や東日本大震災と比べ大幅に短縮され、迅速さにおいてはほぼ飽和状態に達していると述べられている [3]。しかし、ライフラインの完全な復旧には 2 週間を要しているのが現状であり、その間は十分に外部との連絡を取れない人がいるのが事実である。

2.1.3　既存のアプリケーション

現在、Apple 社が提供するクラウドサービス “iCloud” の機能である “友だちを探す” や Google が提供している地図・ローカル検索サービス “Google マップ” の機能である “現在地を共有” 等の位置情報の共有を目的としたアプリケーションが存在する。しかし、これらのアプリケーションは利用する度に共有相手の設定や時間の指定等の設定の変更が必要になる。また、個人間の共有を目的にしているため、複数人のグループ等で共有をする際はグループ内の全員がお互いに個人間の設定をする必要がある。そのため、人数が増えるほど必要な設定数が増えていくことになる。具体的には、既存のシステムでは複数人と情報共有する際に利用者全員が自分以外の利用者全員と設定する必要があるため、利用者の数を n としたときに $n*(n-1)$ 回の設定が必要になる。それに対して、本システムでは複数人で情報共有をする際、予め作成したグループに利用

者がそれぞれ加入する設定を行うのみなため、利用者の数を n としたときに n 回の設定が必要になる。実際の数値で考えたときの既存のシステムと本システムの差は表 2.1 のようになる。

また、これらのアプリケーションではお互いに iCloud や Google アカウント等の外部の登録が必要な固有アカウントを把握している利用者の情報しか得ることができないため、情報の共有に制限がある。

2.2　提案するシステム

そこで、本研究では上述の問題点を踏まえて以下のような機能を導入したシステムを提案する。

1. システム利用者の位置情報を把握できる機能

   本システムの根本的な機能として、利用者がシステムに送信した位置情報のデータをブラウザ上のマップにユーザの設定する分単位で更新して表示する。

2. 一定範囲の地域内の分布を表示するマップ作成アプリケーション

   災害時に利用者の避難状況を把握できるようにするため、設定した地域内の利用者の分布をブラウザ上のマップに表示する。ここでは個人は特定せず、あくまで分布のみの表示とする。

3. 個別アカウントによるグループ管理

   利用者個人を判別するためのアカウントを用いて、複数人単位で相互の情報管理を可能にする。既存のアプリケーションにおける問題点を踏まえ、複数人での共有を簡潔な作業で行えるようにする。

4. 複数人グループで情報共有する利用者のマップ作成アプリケーション

   上記の複数人グループ内で、個人を判別した上で位置情報を相互に設定した数分単位で更新・管理できるマップをブラウザ上で実現する。

表 2.1: システム利用時に必要な設定数の差

利用人数	既存のシステム	本システム
2	2 回	2 回
3	6 回	3 回
4	12 回	4 回
n	n(n-1) 回	n 回

第3章　システムに必要な情報と構築環境

本研究で開発するシステムに必要なデータと構築環境について記述する。

3.1　必要なデータの考察

本システムを設計するにあたり必要なデータについて考察する。

3.1.1　ベースシステムにおける必要最低限のデータ

本システムの設計において、前段階として Web 上のマップ描画における位置情報の共有のみを目的としたシステムを考える。必要なデータは以下の通りである。

- ユーザ名

  個人を特定するために必要な固有のユーザ名が必要になる。

- グループ情報

  本システムにおける複数人で情報を共有する機能を利用する際に、共有するユーザ間に共通の情報を付与する必要がある。そこで、本システムではグループという括りの情報を付与し、共通のグループに所属するユーザ間で情報を共有する形式を取る。そのため、グループの作成等の設定に関する情報が必要になる。

- 位置情報

  Web 上のマップにユーザの位置を表示するために必要になる。

ここから本システムを設計していく上で必要なデータを考察しながら追加していく。

クッキー ID

Cookie とは、ユーザ側に各種情報を保持させるためにサーバから送られ、ユーザの端末に保存されるファイルである。本システムは Web 上の複数ページを跨いで利用するため、ページを移動した際にもユーザの情報を引き継ぐ必要がある。また、一度本システムにログインしたユーザが本システムから別のページに移動した後もログイン状態を保持する形式にする。そこで、本システムにログインした際にユーザの端末毎に固有の ID を設定し Cookie に保存し、クッキー ID というデータとして扱う。

パスワード

本システムを利用する際、先述の通り Cookie を取得していれば本人の認証を維持することは可能である。しかし、利用端末の故障等の理由から別の端末に変更した際や複数端末からの利用をする際に Cookie を引き継ぐことができない。そのため、 本システムのクッキー ID とは別に情報を引き継ぐための個別の値が必要になる。そこで、クッキー ID とは別にユーザごとのパスワードをユーザを認識するための情報として設定する。

グループマスタ

複数人数でのグループで情報を共有する際、本システムでは情報の更新頻度等の設定は個別で行う。しかし、この機能では位置情報とともにユーザ名も表示するため、作成したグループに全てのユーザが自由に加入できる設定にすると、他ユーザの名前と位置情報を自由に取得できることになり、プライバシやセキュリティ面の安全性に欠ける。そこで、ユーザの中にグループマスタという役職を割り当て、グループへの参加承認等に関する権限を付与する必要がある。

最終利用時刻

本システムは Web 上のマップ描画によるリアルタイムな情報共有を目的としている。ユーザ名と位置情報のみの表示ではマップに表示される他のユーザがいつ情報を送信したかを知る手段がないためリアルタイムに情報を更新する利点が失われる。そこで、本システムでユーザが情報を送信する際に利用した時刻も送信し、閲覧できるようにする必要がある。

地域名

本システムでは災害時のユーザの避難分布を表示する。その際、全ユーザの情報を表示すると範囲が広く、表示されるマーカの数も多くなるため、1 画面で確認するマップの視認性が著しく低下する。そこで、一定範囲ごとの地域を設定し、設定した地域内の利用者の分布を表示する。現段階では市町村を単位としているが、人口等を考慮しさらに明確な条件のもと範囲の単位を設定する必要がある (図 3.1)。

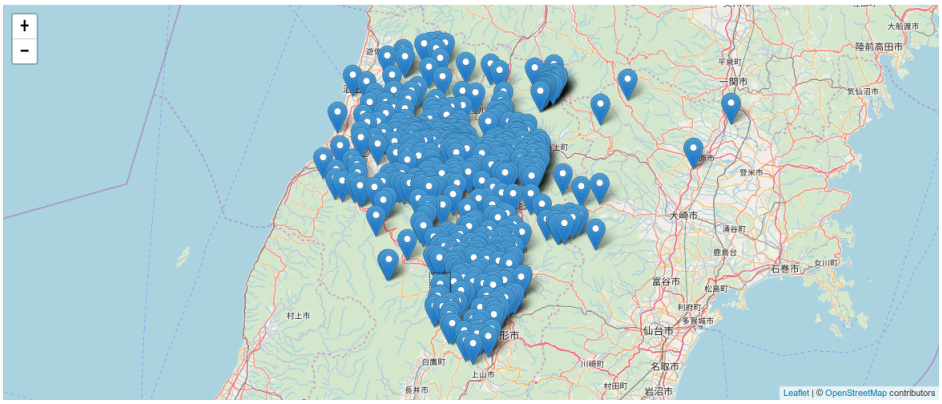

図 3.1: 全ユーザの情報表示時の例

3.1.2 本システムに実装するデータ

上記の点を踏まえて、本システムでは以下のデータをユーザの端末から取得し、管理する。

- ユーザ名
- パスワード
- クッキー ID
- 位置情報
- グループ加入情報
- グループ編集権限情報
- 利用時刻
- 地域名

3.2 データの正規化

本システムは関係データベース (RDBMS) でデータを管理するため、データを正規化する必要がある。正規化とは、データを効率良く扱うために特定のルールにしたがって整理することである。

例として、本システムで利用するデータの一部をまとめた表 3.1 がある。この状態では任意のデータを抽出したり参照したりする際に一度に複数のデータが当てはまるため効率的ではない。また、一つのデータを修正するために複数の箇所を変更する必要があり手間がかかる場合がある。そこで、本研究ではプログラムからデータの参照や抽出を行うため第 3 正規化まで行う。

以下に本システムで利用したデータの正規化の手順について記述する。

表 3.1: 生のデータ

太朗	hogehoge	鶴岡	A グループ 38.4, 138.5 12:00 B グループ 38.6, 138.1 13:00
花子	gehogeho	酒田	A グループ 38,1, 138.1 13:05
鈴木	pasuwa-do	鶴岡	B グループ 38.5, 138.5 14:30

非正規形

表 3.1 のように整理されていない生の状態のデータのことを非正規形という。データベースでデータを管理する際、このように一つの項目 (フィールド) 内に複数のデータが格納されていても複数のデータと認識することはできない。これらのデータを関連性を失わないようにデータベースが計算機で処理しやすい形式にすることが正規化である。

表 3.2: 繰り返しを排除した形

name	pass	group	local	lat	lng	time
太朗	hogehoge	A グループ	鶴岡	38.4	138.5	12:00
太朗	hogehoge	B グループ	鶴岡	38.6	138.1	13:00
花子	gehogeho	B グループ	酒田	38.1	138.1	13:05
鈴木	pasuwa-do	B グループ	鶴岡	38.5	138.5	14:30

第 1 正規化

非正規形において繰り返しや、一つのフィールド内に複数格納されているデータを別の行 (レコード) に分離し、繰り返しを排除する。

さらに、テーブルごとにデータを一意に識別するための項目を設定する。このような列 (カラム) のことを主キーと呼ぶ。主キーは複数カラムから構成されることもあり、その場合は複合主キーと呼ぶ。今回は name と group が決まれば他のデータが一意に決定するためこの二つを複合主キーとする。

第 2 正規化

表 3.2 の第 1 正規形から部分関数従属性を排除する。関数従属とは、ある属性 X を決めると他の属性 Y が一意に決まるような状態のことである。この時、X を決定項、Y を被決定項と呼ぶ。部分関数従属とは、このときの被決定項が複数の決定項を持っている状態のことである。表 3.2 の場合、name が決まれば pass が決まるが、他の項目が必ずしも決定しない。そこで、たとえば name を主キーとする pass だけをまとめたテーブルを新規に作成する。このように第 2 正規化をすることでテーブルごとの列の数を減らし、データの管理をより単純化する。

表 3.3: パスワードテーブル

name	pass
太朗	hogehoge
花子	gehogeho
鈴木	pasuwa-do

表 3.4: 地方テーブル

name	local
太朗	鶴岡
花子	酒田
鈴木	鶴岡

表 3.5: ユーザ情報テーブル

name	group	lat	lng	time
太朗	A グループ	38.4	138.5	12:00
太朗	B グループ	38.6	138.1	13:00
花子	B グループ	38.1	138.1	13:05
鈴木	A グループ	38.5	138.5	14:30

第 3 正規化

第 2 正規形から推移的関数従属性を排除する。推移的関数従属性とは、被決定項または決定項と被決定項の組み合わせによって他の被決定項が一意に決まることを指す。今回の場合、表 3.9 において、name と group が決まればその他のデータが決まるため、別のテーブルを新たに作成する。検索の効率を考慮して部分的に正規化の程度を敢えて落とした状態のままにする場合もある。最終的には表 3.6, 3.7, 3.8, 3.9 の状態になる。

3.3 作業環境

開発は以下の環境で行う。

- Ruby 2.3.0
- Leaflet.js 1.3.4
- SQLite3

表 3.6: パスワードテーブル

name	pass
太朗	hogehoge
花子	gehogeho
鈴木	pasuwa-do

表 3.7: 地方テーブル

name	local
太朗	鶴岡
花子	酒田
鈴木	鶴岡

表 3.8: 位置情報テーブル

name	group	lat	lng
太朗	A グループ	38.4	138.5
太朗	B グループ	38.6	138.1
花子	B グループ	38.1	138.1
鈴木	A グループ	38.5	138.5

表 3.9: 利用時刻テーブル

name	group	time
太朗	A グループ	12:00
太朗	B グループ	13:00
花子	B グループ	13:05
鈴木	A グループ	14:30

第4章　システムの設計

本システムの設計について記述する。

4.1　本システムの概観

本システムを利用する際の流れは以下の通りである。

1. アカウントの作成・ログイン

   本システムを利用するための情報を入力しアカウントを作成し、ログインする。

2. グループの作成・加入・選択

   指定したユーザ同士で構成されるグループ周辺の設定をする。

3. 利用端末から情報を送信・取得

   ユーザ自身の位置情報・利用時間等の情報を RDBMS に送信すると同時に最新の情報を取得する。

4. 取得したデータが反映されたマップの閲覧

   取得したデータを元に目的に応じて作成されたマップを閲覧する。

なお、目的がユーザ自身の情報の更新に限る場合グループを選択する必要はなく、自身の情報のみを更新する機能を利用することができる。システム全体の概念図は図 4.1 の通りである。

4.2　個別アカウントの管理

本システムはアカウント作成の際にユーザ名とパスワードと地域のみを設定する設計で、メイルアドレス等の外部のデータを必要としない。そのため、ユーザ名を個人を判別するためのユニークキーとする。

4.3　JavaScript を用いた動的な Web の作成

本システムでは主に JavaScript を用いて動的な Web ページを作成する。

4.3.1　用語解説

本章の話をするにあたって必要な用語の解説を記述する。

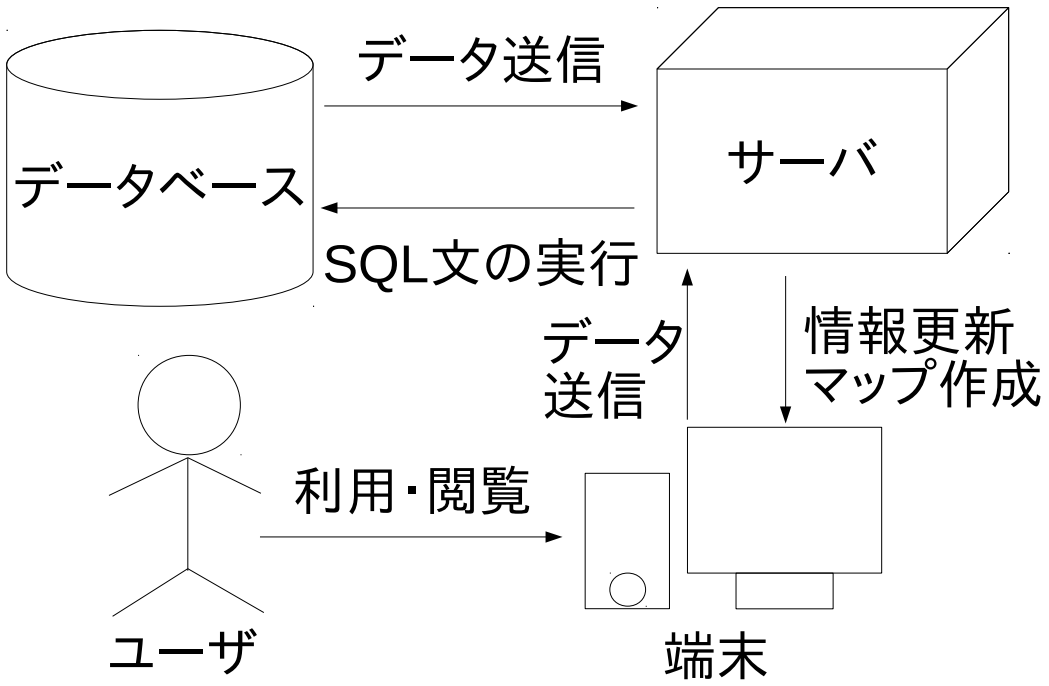

図 4.1: システム利用時の概念図

JSON

JavaScript Object Notation(JSON) とは、表形式では記述が困難なデータを人間に対しての可読性を残しつつ計算機に対して伝達する記法である。一般的に Web アプリケーションでデータを送信する際に使用される。以下に本システムで使うデータを用いた JSON 形式の例を記述する。

JSON 形式の例

```
[
 {
  "名前": "mituyuki",
  "緯度": "138",
  "経度": "38",
  "時間": "2018-12-28 15:49",
  "group" : [
       "Agroup",
       "Bgroup"
       ]
 }
]
```

GeoJSON

GeoJSON とは、JSON を用いて位置情報等の空間データをエンコードしたファイル形式である。GeoJSON 形式ではジオメトリ (形状)、フィーチャー (地物)、フィーチャーコレクション

(フィーチャーの集合) の 3 つのオブジェクトによって固有の値がそれぞれ決まっている。それぞれのオブジェクトの説明を記述する。

- フィーチャーコレクションオブジェクト

  features という名前のキーが必要である。value はフィーチャーオブジェクトを要素とする配列である。

- フィーチャーオブジェクト

  geometry というジオメトリオブジェクトを value とする key と、properties という任意の JSON オブジェクトを value とする key が必須である。

- ジオメトリオブジェクト

  位置やマーカの種類等の構成を設定する要素を記述するオブジェクトである。

以下に GeoJSON 形式の例を記述する。

GeoJSON 形式の例

```
[
 {
  "type": "FeatureCollection",
  "features": [
     {
      "type": "Feature",
      "properties": {
        "name": "mituyuki",
        "description": "あなたとの距離:300m"
       },
      "geometry": {
        "type": "Point",
        "coordinates": [
          138,
          38
         ]
       }
     }
   ]
 }
]
```

OpenStreetMap

OpenStreetMap(OSM)[1]とは、道路地図等の地理情報データを誰でも自由に参加・編集・利用できるようにし、フリーの地理情報データを作成することを目的にしたプロジェクトである。

[1]https://www.openstreetmap.org

図 4.2: OSM の地図表示例

Ajax

Asynchronous JavaScript + XML(Ajax) とは、JavaScript と XML を使って非同期にサーバとの通信を行い反映させることである。非同期な通信とは、Web ブラウザからサーバに情報を送信する際に、一部の情報のみを送信することでサーバからのレスポンスを待たずに他の処理を行うことができる形式の通信のことである。反対にサーバに情報を送信する際に全てのデータを送信し、サーバからのレスポンスがあるまでページ内で他の作業が行えない形式の通信を同期通信という。

4.3.2　Leaflet.js を用いたマップ作成

Leaflet.js とは Web 上で地図を作成・表示するためのオープンソースの JavaScript ライブラリである。GeoJSON からデータを読み込み、ポップアップ等の機能を利用したマップを作成することができる。本システムではこれを用いて OSM の地図をベースにしたマップを作成する。

4.3.3　Ajax による非同期通信

本システムでは利用時にページ内の全ての情報を更新する必要はないため、一部の情報のみを送受信する設計とする。また、ユーザが情報を送受信する間にも同ページ内のマップを閲覧する等の他の作業をできるようにする必要がある。特に、同期通信を行うと情報を送受信する度にマップが更新されるため、ユーザがスムーズにマップを閲覧することができない。そこで、非同期通信を用いて更新する必要がある情報のみをサーバにリクエストする。本システムでは以下のような流れで利用している。

1. ユーザ自身の位置情報を送信する。
2. CGI で SQL 文を実行し、データベースからその時の利用者に必要なデータを JSON 形式で返す。
3. 受け取ったデータを基にユーザの設定したグループを判別し、同グループのユーザの位置情報と時間を表示するマップを作成する。

実際に本システム内で Ajax 通信を用いた部分のプログラムを用いて例示する。
データの送信部分のコードは以下の通りである。

送信部分の JavaScript

```
function submit() {
    var conn = new XMLHttpRequest();
    conn.open('POST', './getgps.cgi');
    conn.setRequestHeader(
    'Content-Type', 'application/x-www-form-urlencoded');
    conn.send('gname='+encodeURIComponent(gname.value)+'&'+'name='+
        encodeURIComponent(name.value)+'&'+'lat='+encodeURIComponent(lat)
        +'&'+'lng='+encodeURIComponent(lng));
    conn.onreadystatechange = respond;
}
```

送信時の動作の流れは以下の通りである。

1. サーバに通信リクエストを送るため、XMLHttpRequest(XHR) オブジェクトを生成
2. サーバに情報を登録するためアクセスメソッドは POST を選択し、パスを指定
3. XHR のメソッドの setRequestHeader で送信ヘッダを設定
4. データの変数名と値を記述し送信
5. XHR のメソッドの onreadystatechange を respond という関数に代入

アクセス先では送られたデータを基に SQL 文を実行し、必要なデータを JSON 形式で返す。
データ受信部分のコードは以下の通りである。

受信部分の JavaScript

```
function respond(str) {
     if (this.readyState == 4) {
        var resp = JSON.parse(this.responseText);
        var MyName = resp[' 名前']; // 自分の名前
        var NameData =  resp['namedata']; // DB の名前一覧
        var LatData =  resp['latdata'];   // DB の Lat 一覧
        var LngData =  resp['lngdata'];   // DB の Lng 一覧
        var LaLn = [parseFloat(LatData[0]), parseFloat(LngData[0])];
    }
}
```

送信時の動作の流れは以下の通りである。

1. readyState が 4(XMR 通信完了後) の時に if 文を実行
2. JSON 形式の文字列を取得
3. それぞれの変数に代入

また、この時に受信したデータを基にプログラムを実行しマップを作成する。

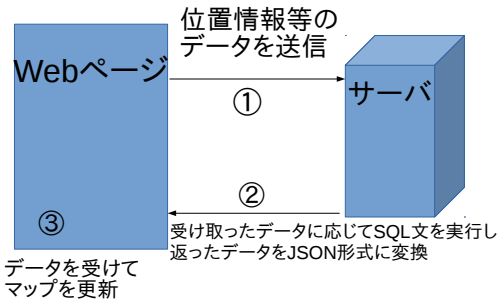

図 4.3: 本システムでの非同期通信の例

4.4　RDBMS によるデータ管理

前章で延べた通り本システムでは RDBMS で各データを管理する。そこから CGI より SQL 文を実行することで必要なデータを取り出す。

本システムでは前章の例に挙げたデータの他にグループのマスタ、クッキー ID のデータを加えて正規化を行った以下のテーブルでデータを管理する。

- ユーザ情報 user(ユーザ名, パスワード)
- 設定地域情報 local(ユーザ名, 設定した地域)
- クッキー ID 情報 cookie(クッキー ID, ユーザ名, 最終ログイン時刻)
- 利用時刻情報 lasttime(ユーザ名, グループ名, 最終利用時刻)
- 位置情報 latlng(ユーザ名, 取得した経度, 取得した緯度)
- グループ情報 ginfo(グループ名, マスタ名)
- グループ加入状況 gmember(ユーザ名, グループ名)

ここではデータ内容の説明をするため分かりやすいカラム名で表記した。実際のカラム名は表 4.1 の通りである。

表 4.1: 4.4 のカラム名と実際のカラム名の対応表

ユーザ名	user
パスワード	pass
設定した地域	local
クッキー ID	cookie
グループ名	group
最終ログイン時刻	login
最終利用時刻	time
取得した経度	lat
取得した緯度	lng
マスタ名	master

4.4.1 本システムの ER 図と解説

本システムで扱うデータの ER 図は図 4.4 の通りである。以下にそれぞれのテーブルの関係について解説する。

user テーブルと local テーブル

本システムではアカウント作成の際に名前とパスワードと地域を設定する。地域の設定は 3.1.1 で記述した通り市町村単位で選択する。地域の設定は 1 アカウントに一つであり作成時に必ず設定するため 1 対 1 の関係である。

user テーブルと cookie テーブル

本システムではログイン時にユーザごとの cookie を作成する。アカウントを作成したのみの状態や、一定時間が経過し cookie が削除された状態では cookie は存在しないため、1 対 0 または 1 の関係である。

user テーブルと lasttime テーブル

本システムのいずれかの情報共有の機能を利用し、情報を送信した際にユーザごとの最終利用時刻をデータベースに登録する。アカウントを登録したのみで情報共有の機能を利用していない際は time のデータは存在しないため、1 対 0 または 1 の関係である。

user テーブルと latlng テーブル

上述の lasttime テーブルと同様に、アカウントを登録したのみで情報共有の機能を利用していない際は lat と lng のデータは存在しないため、1 対 0 または 1 の関係である。

user テーブルと gmember テーブル

ユーザが複数人グループの情報共有機能を利用する際にグループへの登録が必要になる。同時に複数のグループに加入することが可能であり、またグループに全く加入していないユーザも存在するため、1 対多の関係である。

gmember と ginfo

一つのグループには一人以上のマスタが存在し、マスタのいるグループには一人以上のメンバーが所属しているため、多対多の関係である。

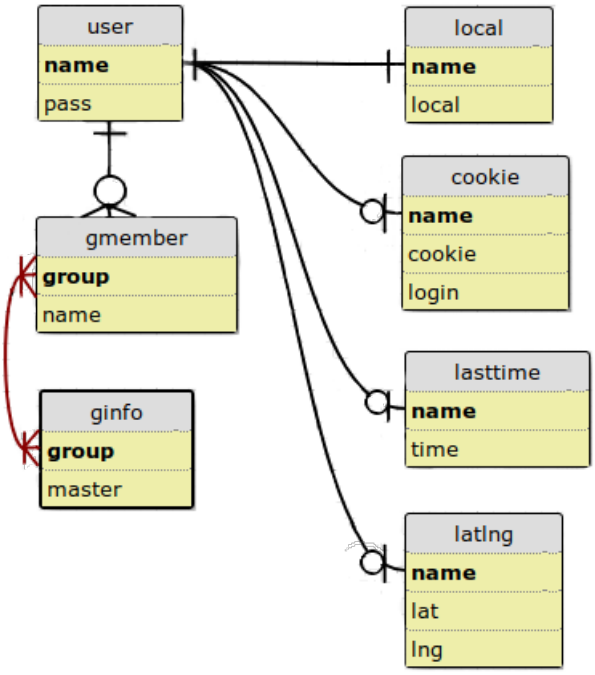

図 4.4: 本システムで扱うデータの ER 図

第5章　システムの実行

実際に構築したシステムを利用する際の動作とその時のシステムの動きについて記述する。利用時の流れは図 5.1 のようになる。

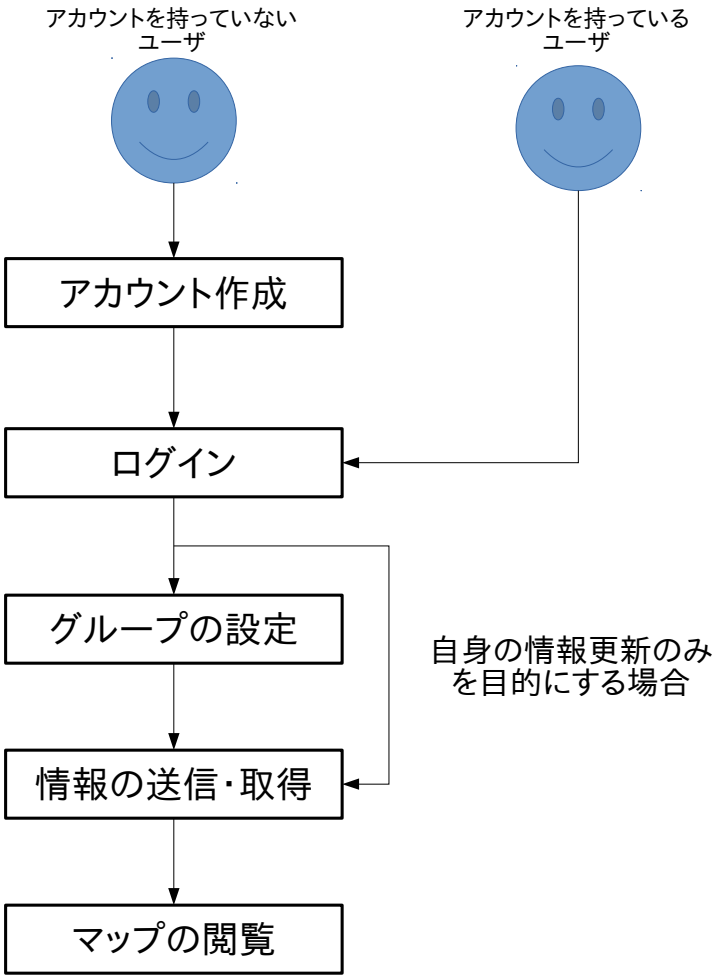

図 5.1: システム利用時の流れ

5.1　個人アカウントの管理

本システムを利用する際、利用者を特定するための個人アカウント周辺の機能に関して記述する。

5.1.1　個人アカウント作成

ユーザ名とパスワードと地域の情報を設定し、利用者個人を判別するためのアカウントを作成する機能である。アカウント作成時にはユーザ名、パスワード、地域の 3 項目を入力する。ユーザ名を個人を判別するためのデータとするため、ユーザ名が既存のアカウントと重複していた際にはエラーを表示する。

図 5.2: ログイン画面

5.1.2　ログイン

作成したアカウントのユーザ名とパスワードを入力し、システムにログインするための機能である。ユーザのログイン情報を保持するため、ログインと同時にクッキー ID を取得する。クッキー ID の有効期限は 24 時間に設定している。

5.2　位置情報を利用した情報共有

本システムのメインとなる機能である。利用者が意図的に情報を送信したり、送信情報をマップ上に表示して共有したりする機能と、そこで得た情報を元に地域ごとに利用者の分布を表示する機能の大きく 2 つに分かれる。

5.2.1　個人の位置情報の送信

本システムにログイン中のユーザが自分の利用情報をデータベースに送信する機能である。ここで得たデータを基に後述の地域ごとの分布マップを作成する (5.2.3)。

5.2.2 複数人グループでの利用

複数人単位のグループ内で個人を特定した上で相互に情報を共有するシステムである。特に、災害時に家族等複数人のコミュニティ内でお互いの情報を把握するための機能である。この機能は、災害時のみならず、グループワーク等複数人の位置情報を相互に把握したい状況で利用することも目的とする。この機能を利用する際に利用者が送信したデータも 5.2.3 のマップ作成に使用する。システム利用時の流れは図 5.3 の通りである。

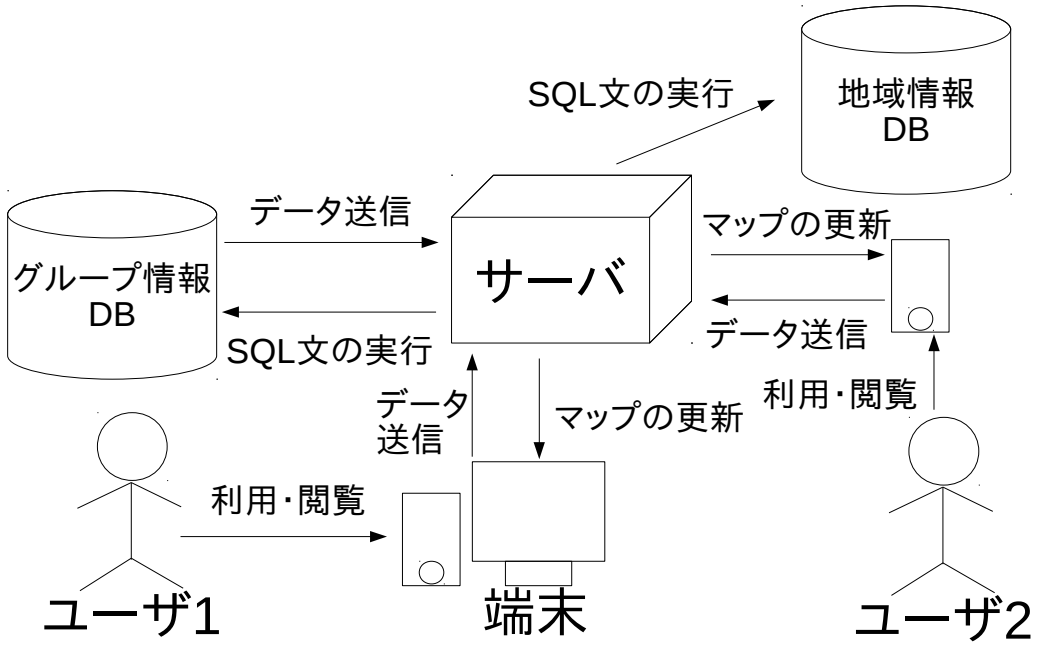

図 5.3: グループ利用時の流れ

グループ作成・管理

少人数グループでの利用をするためのグループを作成・管理する機能である。以下の 3 つの機能からなる。

- 新規グループの作成
- グループへの加入
- 加入済みグループからの脱退

グループ単位のマップの作成

5.2.2 の機能を用いて作成したグループ単位で利用するシステムである。グループメンバが最後に利用した位置情報、名前、時刻をマップ上のマーカで表示する。

図 5.4 は本システムの利用時の一例である。5.2.2 の機能を用いて作成した “みつゆきの仲間” というグループで mituyuki という名前のユーザが位置情報を共有し、test という同じグループのユーザの情報を表示している状態である。

図 5.4: グループ単位のマップの表示例

5.2.3　設定地域内の分布表示

災害時に利用者の避難状況を確認することを想定したシステムである。ユーザが 5.2.1 や 5.2.2 のシステムを利用した際に発信される情報を基に、各ユーザが設定した地域ごとの分布マップを作成する。ここではユーザのプライバシやセキュリティ面への考慮からユーザ名等は表示せず、地域内での分布のみを表示する。図 5.5 は山形県酒田市の人口密度をもとにマップを擬似的に作成したものである。

5.3　本システムの有効性

以上の点を踏まえ、本研究で提示した問題に対する本システムの有効性を記述する。

5.3.1　災害時の避難状況の視覚的な確認

1.1 や 2.1 で記述した通り、位置情報を発信できる情報端末が普及しているにも関わらず、それを用いた利用者同士のコミュニケーションや情報共有が有効に活用されていないことが問題視されていた。それに対し、本システムでは災害時を想定し、スマートフォン等の携帯端末のみ

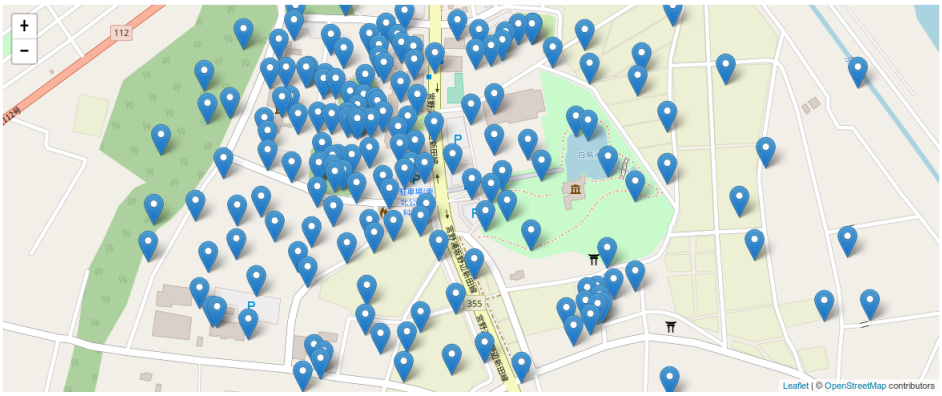

図 5.5: 地域の分布表示の例

を用いて利用者同士の情報を Web 上のマップに描画することで視覚的に確認できる設計を実現した。

5.3.2 完全に独立した設計

2.1.3 で記述した通り、既存の類似アプリケーションでは iCloud や Google アカウント、電話番号等の外部の情報を所持している必要があった。本システムでは専用のアカウントを作成し、またその際にも外部の情報を必要としない設計を実現した。

5.3.3 簡潔で誰でも使える設計

既存のシステムと比べた本システムの複数人数で情報を共有する際のグループ作成にかかる行程数の比較は 2.1.3 で記述した通りである。また、 1.1 で記述したような既存のアプリケーションと比べ、ライセンスの購入等の必要がないため誰でも本システムの機能を自由に利用可能である。

第6章　結論とシステムへの評価

本システムへの結論・評価としては以下のようなものが挙げられる。

6.1　結論

利用者から発信された情報を元にリアルタイムで Web 上のマップを描画するシステムを構築した。それにより、災害時における逃げ遅れ等の二次災害の原因として問題視されていた避難者との情報共有に関する問題を、指定した地域の避難状況を携帯端末から視覚的に確認可能にすることで解決した。また、外部の情報を必要とせず、誰でも制限なく利用できる簡潔なシステムを実現した。

6.2　システムへの評価と展望

本システムへの評価と展望について記述する。

6.2.1　有事の際の情報リセット

本システムは災害時のみならず、平常時からの利用も考慮している。それに伴い、地震等が発生する以前のデータが残っていると地震後に全てのユーザが情報を更新しなければ正確な避難情報を得ることができなくなるという事態に陥る。そこで、Web スクレイピング[1]を用いて常時気象庁からデータを取得し、災害発生時にその時点よりも前のデータを消去する等の処理を加えることで災害時に対応可能にする必要がある [10]。

6.2.2　グループ参加の有無と個人の特定

本システムを個人でのみ利用しているユーザに関して、個人を特定する方法が完全にない状態になっている。匿名性の保持という点では問題はないが、災害時を考えた際に個人の特定の必要性に関して懸念される。

6.2.3　グループ管理のセキュリティ観点

現段階の本システムは、ユーザであれば誰でも存在する全てのグループに自由に出入りできるようになっている。グループに ID を割り振る、特定のメンバーに承認周りの権限を付与する等してグループ管理周辺のセキュリティについて整えていく必要がある。

[1]Web サイトから Web ページの HTML データを収集して、特定のデータを抽出、整形し直すこと。

6.2.4　同時利用時のキャパシティ

5.2.3 から、個人の特定を目的としない場合の避難所から離れている利用者のばらつき等の視認性においては問題なく見えるが、災害時に多くの利用者が同時に利用した場合、また、より人口の多い場合などの動作に関してプログラムを用いて摸擬的に検証する必要がある。

参考文献

[1] 内閣府. 災害情報・防災情報のページ - 内閣府, 2018-11-10. http://www.bousai.go.jp/updates/.

[2] 金野達也, 戸羽開, 柴田義孝, 橋本浩二. 津波などの 2 次災害を考慮した災害時避難経路提示システム. 第 76 回全国大会講演論文集, Vol. 2014, No. 1, pp. 613–614, mar 2014.

[3] 能島暢呂. 熊本地震における供給系ライフラインの被害と復旧 : 震災から得られた教訓と残された課題 (特集 平成 28 年熊本地震 (2) 住民生活). 消防防災の科学, No. 127, pp. 30–34, 2017.

[4] 宗森純, 宮内絵美, 牟田智宏, 吉野孝, 湯ノ口万友. 電子鬼ごっこ支援グループウェアの開発と適用. 情報処理学会論文誌, Vol. 42, No. 11, pp. 2584–2594, nov 2001.

[5] 宗森純, 上坂大輔, タイミンチー, 吉野孝. 位置情報を用いた汎用双方向ガイドシステム xexplorer の開発と適用. 情報処理学会論文誌, Vol. 47, No. 1, pp. 28–40, jan 2006.

[6] 坪田大吾, 花田経子. ios アプリケーションにおける個人情報の取り扱いに関する調査と考察. 第 76 回全国大会講演論文集, Vol. 2014, No. 1, pp. 605–606, mar 2014.

[7] DailyTimer.net. いまどこ?オンラインマニュアル. http://file.blog.fc2.com/nekonotesoft/manual/imadoco-jp.html(参照 2018-12-20).

[8] 松崎頼人, 榎原博之. 地震時におけるスマートホームを利用したアドホックネットワーク―生き埋め被災者のための救助要請 map データの配信. 情報処理学会論文誌数理モデル化と応用（TOM）, Vol. 6, No. 1, pp. 64–78, mar 2013.

[9] 渡邊博之, 成田祐一, 大山勝徳, 加瀬澤正, 武内惇, 竹中豊文. モバイル端末を活用した災害時最短避難経路提示システムの開発. 情報処理学会論文誌, Vol. 53, No. 7, pp. 1768–1773, jul 2012.

[10] 小島一恭, 奥村高広. 2115 web スクレイピングによる温熱データと居住者の温冷感申告との関係性 (福祉機械・ヒューマンインターフェース (2)). 機素潤滑設計部門講演会講演論文集, Vol. 2014, pp. 129–130, 2014.