オフライン時にもデータ収集可能な スマートフォン利用野外情報収集システムの開発

嘉山陽一*

Development of outdoor information collection system using smartphones, which collect data even when offline

Yoichi Kayama*

In 2019, I created a disaster information collection system using chatbots, and generalized the data sharing part of the system using Google Spread Sheet. However, the system using chatbots cannot collect data in places where there is no Internet connection, because the cell phone facilities in the Hitoyoshi area of Kumamoto prefecture were heavily damaged by the torrential rains in Kyushu in 2020. We prepared an execution environment for the system, but it could not be used. In order to collect information even when there is no Internet connection, I developed a prototype of an information collection system that runs as a native application for smartphones. In this system, it is possible to use a smart phone at a disaster site and store information such as text, still images, video images, and voice in the smart phone with location and time information. When the smart phone is connected to the Internet, the stored information is uploaded to a server and stored in a Google Spread Sheet.

Keywords: スマートフォン（smart phone），野外情報収集（field information gathering），オフライン（off-line）

１．はじめに

2019 年に LINE Chatbot を利用した災害情報収集システムを開発し,国内各地で発生した災害現場の情報をデジタルでクラウド上に集め,位置情報を電子地図にすることができた.これら情報は被災地へのボランティア派遣計画策定やゴミ撤去作業の計画策定に活用された.2020 年には Google Spread Sheet を空間情報ストレージとした情報収集と利活用フレームワークとなるようにそのシステムの改良を行った.しかし 2020 年 7 月の熊本県を中心とした豪雨では携帯電話基地局が被害を受け,災害現地ではしばらく LINE を利用することができなかった.被災状況調査のために LINE 利用災害情報収集システムを用意したが現地ではスマートフォンからインターネット接続を利用することができず災害情報収集システムを利用できなかった.

このような状況を考慮するとオフライン状態でもスマートフォンを利用して調査情報をデジタルデータとして取得,蓄積できるシステムの必要性が高くなったと言えよう.

２．災害情報収集システムの利用と機能拡張

2.1．2019 年災害情報収集システムの開発と利用

当初のシステムは LINE Chatbot からテキストや写真,動画,音声を登録するとそれらが Google Spread Sheet に登録されるという仕組みであった[1].さらに Google Spread Sheet のデータを読んで WEB 上で地図表示をするモジュールが付加された.

このシステムは 2019 年 8 月 28 日からの佐賀豪雨で初めて利用された後に 9 月 12 日からの台風 15 号,10 月 12 日からの台風 19 号,10 月 25 日からの台風 21 号関連豪雨での災害状況調査で利用された.とりわけ台風 19 号での被災で街中に大量のごみが積み上げられた長野市では国や自治体の諸機関とボランティアが一体となってごみ撤去計画を作成するための事前調査に本システムが利用された.

* 正会員 朝日航洋株式会社（Aero asahi corporation）
〒**350-1165** 埼玉県川越市南台南台 3-14-4 E-mail：**youichi-kayama@aeroasahi.co.jp**

2.2. 2020 年システム機能拡張

2019 年版システムでは地図データ利用プログラムが Google Spread Sheet をアクセスする API を直接利用していた.そのため地図描画プログラムと Google Spread Sheet の関係が密結合になっていた.よって開発したプログラム以外の地図利用クライアントからは Google Spread Sheet のデータを利用することはできなかった.そこで Google Spread Sheet を直接アクセスするのではなく Google Spread Sheet に格納されている情報を GeoJSON で返す API を作成した.この機能により GeoJSON を扱うことができるクライアントプログラムなら簡単に本システムのデータを利用できるようになった[2].

3．オフライン情報収集システムの検討

2020 年熊本豪雨での事例のようにインターネット接続可能であることを前提としていると災害情報収集システムの利用ができないことがありうる.スマートフォンの場合利用している携帯電話プロバイダの基地局が近隣に無い場合や,既存基地局が災害等で機能を停止してしまった場合 LINE が利用できないので災害情報収集システムが使えない.そのような場合に備えてスマートフォンのネイティブアプリケーションとして情報収集システムを構築し,ネット接続が無くてもスマートフォンのローカルストレージに情報を蓄積するシステムがあると有用だと思われる.近年のスマートフォンはカメラ(動画,静止画),GNSS レシーバ,マイク,加速度センサ等多様なセンサや入力手段を保有していてデータ保存や通信機能をもつ高度なデジタル情報の利活用システムであるといえよう.スマートフォンは災害時の現地情報を位置や時刻情報つきのデジタル情報として収集するためにはとても有用なツールといえる.

3.1. オフライン情報収集システムの機能

オフライン時でも動作可能なスマートフォンネイティブアプリケーションはスマートフォンにインストールして実行することになる.

機能としてはユーザが入力した災害状況情報（写真,動画,テキスト,音声等）を時刻,位置情報付きでスマートフォンのローカルストレージに記録することが必要になる.またネットワーク接続ができるようになった時点で蓄積された災害状況情報をサーバに一括でアップロードできると有用である.

そこでアップロード先を従来の災害情報収集システムと同じ Google Spread Sheet にすればそれ以降のデータ利活用部分は従来システムが利用できる.

図 1　システム構成図

近年の国内におけるスマートフォンは iPhone と Android が稼働するものが普及していると思われる.Android と iPhone の国内シェアは半数ぐらいのところで拮抗している(2021 年 6 月 Android 55.1% iPhone 44.9%)[3].この普及状況を考えると可能であるならば両方のプラットフォームで動作するシステムが作成できることが望ましい.

3.2. 開発フレームワーク検討

iPhone と Android ではそれぞれのネイティブアプリケーション作成のための様々なコンピュータ言語やフレームワークが用意されている.また iPhone と Android 双方むけのネイティブアプリケーションを作成するためのフレームワークとしては React Native, Flutter, Apache Cordova, Xamarin 等がある.今回は新しいフレームワークである Flutter を利用してみた.

Flutter とは 2018 年に Google 社が公開したモバイルアプリケーション開発のためのフレームワークである. Dart という言語を利用して同一ソースコー

ドで Android と iPhone のアプリケーションを作成することができる（最新版 Flutter では WEB, Windows, macOS, Linux のアプリケーションも作成できるようになってきている）.

Flutter でのプログラム作成はたくさん用意されている画面構成用のパーツを組み合わせて画面を作成していくことを基本作業とする.この時 MVVM（Model, View, View Model）という形式のプログラムモデルを採用している.従来の GUI 系プログラムは MVC（Model, View, Controller）というモデルを採用することが多かった. MVC の Model はデータとその処理を担い,View は表示や出力,Controller はユーザ処理のモデルへの伝達を担っていた.

MVVM の場合 Model と View は MVC での定義と変わらないが ViewModel という新しい概念が作られた.ViewModel とは Model と View を結び付けたものである. ViewModel は表示するためデータの状態と表示用の View を結び付けるものである.ViewModel の状態が変化すると View に表示変化のためのメッセージが自動的に送られる. Model は表示の変化には関与せず,ViewModel の状態更新だけを行えばよいことになる.従来の Controller の部分は ViewModel に包含される.システムのビジネスロジック的なデータと操作は Model が担当し,画面表示部分を ViewModel と View が担当することになる.これによってシステムが業務で必要とするデータ構造やロジックの部分とそれらの表示を行い,操作を実行する部分を別モジュールにすることができる.このような分割によってシステム構築時のビジネスロジック開発担当者と画面（フロントエンド）のデザインや開発を分担しやすくなる.またビジネスロジックと表示部分を切り離すことでビジネスロジックのフレームワークや OS への依存性を下げ,再利用可能性を向上させることができる.

４．オフラインシステム開発

4.1. スマートフォンネイティブアプリケーションの機能

本システムでは従来 LINE で行っていた災害情報収集の部分をオフライン時でも可能にすることをシステムの目的とする.そのために収集情報（テキスト,写真,動画）を直接クラウドに送るのではなくスマートフォンの中に記録する機能が必要である. またスマートフォン内に蓄積した調査データを通信ができるようになった段階で Google Spread Sheet にアップロードする機能が必要とされる.今回はスマートフォンローカルへのデータ保存として SQLite3 データベースを利用した.

調査情報はユーザが任意の名前のデータ格納フォルダ（ワークスペース）を作成してそこにテキスト,写真,動画データを格納するようにした.さらに選択されているワークスペースを Google Spread Sheet にアップロードする機能を実装した.

ワークスペースに格納した調査データはレコード毎に投稿日時と位置情報（緯度,経度）を持たせている.位置情報は投稿したときのスマートホンセンサ（GNSS 等）で計測された位置を記録している.旧来の LINE システムで投稿したデータでは投稿データレコード毎に位置情報をもたせることはできない. そのため LINE 投稿システムでは最初に位置情報データの投稿を行い,その後にテキストや画像の投稿を行うことで先に投稿した位置情報をテキストや画像にリンクして位置としていた. 今回作成したシステムではテキスト,画像等の各投稿データに位置を付加するので LINE から投稿したデータとは位置の扱いが異なる.そのため Google Spread Sheet のデータを利用する場合 LINE 投稿データと本システム投稿データでは扱いが異なることになる.Google Spread Sheet 上では LINE からの投稿データと本システムからの投稿データを混在した状態であつかうことが想定される.そのためシートに登録するデータ項目とシートデータ利用プログラムの改修が必要となる.

またワークスペースに登録した個別レコードはレコード単位に削除ができるようにした.またテキストのレコードについては登録テキストの修正を可能にした.

ワークスペースを Google Spread Sheet にアップロードを行う時のユーザ認証を LINE の認証プロバイダを利用して実装した. LINE の認証プロバイダとは

LINE のデベロッパーコンソールを利用して作成できるユーザ認証サービスである.ここで作成したプロバイダは LINE のユーザ ID とパスワードを使ってユーザ認証を行うことができる.また認証プロバイダにリンクする LINE 公式アカウントを指定できる.ここで従来から利用している災害情報登録用 LINE Chatbot をリンクする公式アカウントとして指定可能である.公式アカウントリンクが指定されている認証プロバイダを利用したユーザの認証が成功した場合,当該ユーザがリンク公式アカウントと友達になっているかどうかのステータスがクライアントプログラムにかえされる.そこで本プログラムではリンクされた Chatbot と友達になっていないアカウントのデータアップロードは許さない仕様にした.

またアップロードしたワークスペースは手元では不要になるのでワークスペースを削除する機能を実装した.

ユーザの設定可能項目としては LINE 認証チャンネルの ID とデータアップロードプログラムの URL を設定できるようにした.

図２　画面遷移

4.2. シートの改良

アップロードされたデータを格納する Google Spread Sheet では本システムからアップロードされたデータと LINE Chatbot からアップロードされたデータが混在することになる.前記のように LINE Chatbot からアップロードされたデータとネイティブアプリからアップロードされたデータでは座標の扱いが異なる.よって同じシートに混在するデータでもどちらのシステムからアップロードしたデータなのか識別できる必要がある.従来のシートの項目に「プログラム名」(投稿を行ったプログラムの名前)とトランザクション ID という項目を追加した.トランザクション ID とは 1 回のアップロード作業でアップロードされたデータレコードに共通に降られるユニーク ID である.アップロード後のデータ内でデータアップロード作業が同じデータを識別する場合に利用できる.この ID を利用してアップロード作業単位でデータの取り消しや修正を行うことが可能になる.プログラム名の項目は LINE で投稿したデータには「LINE」,本システムで投稿したデータには「ReportPost」という名称がカラムに記入されるものとした.

4.3. サーバシステムの改良

今回は作成したネイティブアプリケーションからバッチ処理でアップロードされたデータを Google Spread Sheet に書き込むサーバプログラムを新規に作成した.また地図データについては GeoJSON 形式でデータを配信するモジュールが実装されている.この地図データ作成部分のプログラムについて今回のネイティブアプリケーションで作成したデータにも対応できるように改良を行った.

アップロード機能については LINE のユーザ認証を利用した認証機能を利用し,アップロードデータを Google Spread Sheet に記録するプログラムを作成した.

GeoJSON データ配信機能については LINE で投稿したデータとネイティブアプリケーションで投稿したデータでは座標の付与の仕方が異なるので各データについてアップロードしたアプリケーション名を判別して別の処理を行った.LINE から投稿されたデータはテキストや画像の投稿レコードには座標はついていない.LINE には位置情報の投稿という機能があるので最初に位置情報を投稿した後でテキストや画像を投稿することにしている.このデータを GeoJSON に変換する場合は投稿ユーザ名と投稿時

刻でソートしたデータでテキストや画像レコードは同一ユーザが投稿した直前の位置情報レコードにリンクさせることで座標を作成し,GeoJSON の地物オブジェクトにしている.しかしスマートホンネイティブアプリケーションで作成したデータレコードはテキストや画像の各レコードに位置座標が格納されている.そのため単独のレコードをそのまま GeoJSON の地物オブジェクトにすることができる.

調査データが格納された Google Spread Sheet にはこの2種類のレコードが混在することが想定される.今回のシート形式の改良でシート内にアップロードプログラムの名称を記録するようにしたため各レコードは LINE から登録したレコードなのかネイティブアプリケーションで登録したレコードなのか識別をできるようになっている.よって Google Spread Sheet のデータを GeoJSON に変換する場合は各レコードを投稿したソフトウェア名称を判定して,その名称別に異なるロジックで GeoJSON オブジェクトを生成するようにした.

５．システム設定と操作

本システムの設定と操作は以下のような手順になる.

5.1. Google Spread Sheet とサーバの準備

最初にデータを格納する Google Spread Sheet を用意する.Google Spread Sheet では通常先頭のシートにデータ書き込みが実行される.またデータ登録時は関係ないがデータ利用時に表示する地図等を定義する「config」という名前のシートを用意して,そこにオーバーレイ表示する地図の名前や URL を定義が可能である.データ登録用サーバプログラムではこの Spread Sheet の ID と当該 Google Spread Sheet にアクセスを可能とするための Google サービスアカウントキーが記入された JSON テキストが必要となる.ここでは API 実行用の Google サービスアカウントが必要で,そのアカウントに対して Spread Sheet の共有の許可の設定を行う.サービスアカウントキーの JSON ファイルは Google API コンソールから取得できる[4].

データアップロード用のサーバシステムは PHP 言語で記述を行った.前記 Spread Sheet ID とサービスアカウント JSON の情報はサーバ実行環境の環境変数として設定を行った.

5.2. LINE 認証プロバイダの準備

アップロード作業用サーバはインターネット上に公開されて運用することになるのでなんらかの認証が必要になる.LINE ではユーザ認証用のプロバイダを作成して公開することで LINE のユーザ認証を別プログラムでも利用できるような仕組みを用意している.今回はユーザ認証としてこの仕組みを利用した.LINE の認証プロバイダでは LINE のユーザ名,パスワードを利用して認証を行うことができる.これだけだと本システムに関係が無い人でも LINE のアカウントを持っていれば認証を通ることができる.認証プロバイダでは既存の LINE 公式チャンネル（LINE Chatbot 等）との連携を定義できる.この定義を行った上で認証されたユーザについてはプログラム中で連携 LINE 公式アカウントと友達になっているかどうかの状態を取得できる.よって本システムでは現状で LINE の投稿用 Chatbot を用意して利用ユーザにはその Chatbot と友達になってもらい,ユーザ名,パスワードで認証を行った上でその Chatbot と友達になっているユーザに作業の認可を行うことにした.

LINE の認証プロバイダは LINE Developers console という WEB ページにアカウントを作成してログインすると作成することができる[5].Android のネイティブアプリケーションからはプロバイダの ID を指定して利用することになる.データ収集のプロジェクト毎にプロバイダを用意することになる.ネイティブアプリ側としてはどのプロジェクトの認証プロバイダを利用するかプロバイダの ID を設定する機能が必要になる.

5.3. スマートフォンアプリケーションインストールと設定

現状では Android 用 apk ファイルを作成して,そのファイルを Android 端末に転送しインストール許可の設定を行った Android 端末へのアプリケーション

インストールが可能である.また手続きをすることによって Android 用アプリケーションは Google Play[6], iPhone 用アプリケーションは App Store[7]で公開することが可能であるが,この公開作業はまだ行っていない.

インストールしたアプリケーションでは LINE 認証プロバイダの ID とアップロードサービスの URL を設定画面で設定できる.この設定を行うことでワークスペースのアップロードが可能になる.

5.4. データ収集

正しいアップロード URL と認証プロバイダ ID が設定されていると本システムで収集した情報（テキスト,画像,動画像）を指定した Google Spread Sheet にアップロードすることが可能になる.

データ収集を行う場合は最初にデータを格納するワークスペースを作成する.ワークスペースとは任意の名前で登録可能な収集データを格納するフォルダである.収集したデータはワークスペース単位で Google Spread Sheet にアップロードすることが可能である.

データ登録用ワークスペースを登録した後にリストでワークスペースを選択すると投稿データのリスト画面が表示される.この画面で下部のボタンをクリックすると選択ワークスペースに対するテキストの登録,写真の登録,動画の登録が可能である.

写真と動画像についてはカメラを起動してその場で撮影したものの登録とスマートフォン内に記録されている既存の写真や動画像ファイルを選択することを可能にした.

各投稿データは投稿ボタンを押したときにスマートフォンで位置情報を取得してデータレコードに記録して Google Spread Sheet にアップロードするデータとして記録している.

投稿データのリストでは各投稿データの削除ができる.またテキストの投稿データについては投稿データの修正が可能である.

投稿データリスト画面の下部にあるアップロードボタンをクリックすると選択されているワークスペースのデータをすべて Google Spread Sheet に対するアップロードが行われる.

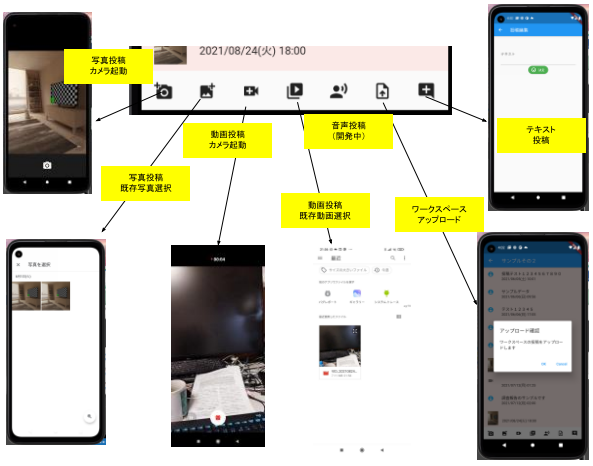

図３ ワークスペース画面メニュー

5.5. 収集データ利活用

収集したデータは Google Spread Sheet に保存して利用している.Google Spread Sheet に格納された情報の利活用方法は昨年まで開発したシステムと変わりが無い.

基本的には Google Spread Sheet にデータが蓄積されるので,そのまま表計算ソフトウェアとして利用可能である.検索,ソーティング,フィルタリング等表計算ソフトウェアの機能はそろっている.

またデータを地図表示するために指定シートを GeoJSON で出力する WebAPI を実装している.この GeoJSON データを WEB 地図に表示して利用するための WEB 地図標準クライアントを用意した.しかし,この機能は GeoJSON という標準的なインターフェースでデータを提供しているので GeoJSON 読み込み可能な空間情報処理ソフトウェアなら利用が可能である.

さらに本データを QGIS で利用するためのプラグインを複数開発した.Google Spread Sheet に定義した地図セットを QGIS プロジェクトとして取得する sheetmapimporter [8]というプラグインを開発した.またインターネットに直接接続できない環境で Google Spread Sheet に格納したデータを利用するために本システムで蓄積したデータを Microsoft Excel のシートに保存するプログラムと,その Excel ファイルをメールで送信し,受信した Excel ファイルを読ん

で QGIS のベクトルレイヤに追加するプラグインを開発した[9].

６．課題

本システムでは Google Spread Sheet にアップロードしたデータを WEB 地図等で地図化する機能は用意してある.ただしこれら機能はあくまでオンライン状態の場合でなければ利用できない.入力データの位置確認等のためオフライン状態でも利用できる地図表示機能の実装が必要だと思われる.

また現状で全データの投稿時に座標が取得できることを前提としてデータ収集を行っているが場所によっては座標取得ができないことが想定される.そのような場合の対処方法を想定して実装しておく必要がある.

データアップロード時のユーザ認証方法については現状で LINE ログイン機能のみ用意してある.その他の方法も利用も検討したい.

ネイティブアプリケーションについては現状で Android 版の apk ファイルをビルドして手動でインストールする方法が用意してある.ユーザがアプリケーションをインストールしやすくするためには公式のアプリケーションマーケットで作成したアプリケーションを公開できるといい.また iPhone 向けアプリケーションのビルドは Flutter で用意されているがまだ行っていない.iPhone 用アプリケーションマーケットで公開できるとアプリケーションの普及が進むと思われる.

７．まとめ

本スマートフォンネイティブアプリケーションの開発によって携帯電話の電波をとらえることができないオフライン環境での情報収集と蓄積が可能になった.後日データを Google Spread Sheet にアップロードすることによって LINE 収集データとのデータ統合とそちらで用意されていたデータ利活用環境が利用できる.

近年のスマートフォンは様々なセンサを搭載し持ち運びできるため野外情報を電子的に集めるための最良の機器といえる.スマートフォンを利用した情報収集,収集情報のクラウド上での共有,共有収集情報の利活用というような仕組みがモダンな空間情報利活用では必要とされるフレームワークであろう.本システムの開発がそのようなフレームワーク利活用を進めるパーツの一つになれば幸いである.

参考文献

[1]嘉山陽一・畑山満則・宮川祥子・佐藤 大（2019）チャットボットを利用した災害時情報収集システムの開発.「地理情報システム学会第 28 回研究発表大会論文集」.

[2]嘉山陽一（2020）Google Spread Sheet を利用した地理空間情報フレームワークの開発,「地理情報システム学会第 29 回研究発表大会論文集」.

[3]Smartphone OS sales market share evolution ,https://www.kantarworldpanel.com/global/smartphone-os-market-share/,(参照 2021-08-19)

[4]Google Cloud 認証の概要, https://cloud.google.com/docs/authentication/?hl=ja, (参照 2021-08-19)

[5]LINE login, https://developers.line.biz/ja/docs/line-login/ , (参照 2021-08-19)

[6]Google Play 公開前チェックリスト, https://developer.android.com/distribute/best-practices/launch/launch-checklist?hl=ja, (参照 2021-08-19)

[7]iOS App と iPadOS App を App Store に提出する, https://developer.apple.com/jp/ios/submit/ , (参照 2021-08-19)

[8]sheetmapimporter, https://github.com/yoichigmf/sheetmapimporter

[9]ReadReport, https://github.com/yoichigmf/ReadReport