社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

ライフログ可視化フレームワークMashMapの実装と評価

高橋　昂平†　　松本　真佑†　　中村　匡秀†

† 神戸大学　〒 657–8501 神戸市灘区六甲台町 1–1
E-mail: †koupe@ws.cs.kobe-u.ac.jp, ††{shinsuke,masa-n}@cs.kobe-u.ac.jp

あらまし　本稿では，位置情報が含まれる様々なデータをマッシュアップし，地図ベースのライフログサービスを容易に開発するための **MashMap フレームワーク**の実装と評価を行う．MashMap は地図ベースのライフログを汎用的に扱い，地図上への可視化を容易にするための機能を持つ．これらの機能を Java を用いて実装し，API を RESTWeb サービスとして公開する．提供される様々な API を GUI を介して気軽に利用できるフロントエンドとして MashMap Builder の実装を行う．また，実装した MashMap フレームワークを用いて地図を用いたライフログ振り返りサービスを開発する実験を行う．提案フレームワークを利用した場合と，利用しない場合とで，開発工数やプログラムのサイズの違いを比較し，提案フレームワークの有効性を評価する．

キーワード　ライフログ，マッシュアップ，位置情報，API，フレームワーク，地図作成

Implementation and Evaluation of MashMap Framework for Visualizing Location-based Lifelog

Kohei TAKAHASHI†, Shinsuke MATSUMOTO†, and Masahide NAKAMURA†

† Kobe University　Rokko-dai-cho 1–1,　Nada-ku,　Kobe,　Hyogo,　657–8501 Japan
E-mail: †koupe@ws.cs.kobe-u.ac.jp, ††{shinsuke,masa-n}@cs.kobe-u.ac.jp

Abstract　This paper presents the implementation and evaluation of *Mashmap framework*. It treats location-based lifelogs in general and has functions to facilitate the visualization of location-based lifelogs. We have implemented it using Java language and published its APIs as the REST Web service. Also, We implement *MashMap Builder* as a GUI frontend in order to feel free to call the APIs. And, we conduct an experimental evaluation, where subjects develop a service visualizing various lifelogs which is created in two different processes with and without the MashMap framework. Between the processes, we evaluate the effectiveness of proposed framework by conparing man-month, lines of source code and so on.

Key words　lifelog, mashup, location data, API, framework, map creation

1. はじめに

位置情報を含む様々なライフログには，地図を用いた可視化が必要不可欠である．この可視化の行程はライフログの種類に依らない共通した処理であるため，我々は先行研究 [1] において，様々な種類の位置付きライフログデータを地図上に可視化するための汎用的なアプリケーションフレームワーク **MashMap フレームワーク**を提案している．MashMap フレームワークでは，様々な形式の位置付きライフログを，我々の研究グループで提案しているライフログ標準データモデル (LLCDM) 形式 [2] [3] に変換してデータベース (DB) に蓄積する．アプリケーションの開発者は，必要なデータを DB から選別するためのフィルタと，そのデータの表示方法を定義して，データソースを作成する．次に，定義したデータソースを一つあるいは複数選択して，MashMap を定義する．MashMap とは，データソースを指定された表示方法で同一地図上に重ねて表示 (マッシュアップ) するデータオブジェクトである．MashMap オブジェクトは最終的に MashMap Renderer を利用し，Google Map 上に可視化される．提案フレームワークを用いることで，開発者は地図を用いた様々なライフログサービスを迅速かつ容易に開発することが可能となる．

現在，提案フレームワークには各オブジェクトを操作・取得する様々な API が用意されているが，これらの API の内 MashMap オブジェクトを作成する過程において使用する API はほとんど同じである．さらに，既に作成したオブジェクトを再利用すれば効率的な開発が行える．しかし，たくさんの API

が備えられているが故に，開発者は最初に何をすればいいのか混乱してしまうかもしれない．開発者がこれらの API を気軽に使い，試せるような仕組みがあれば，位置付きライフログのわかりやすい可視化を促進すると共に，より付加価値のあるライフログマッシュアップの一助になると我々は考える．

一方，先行研究 [1] では，MashMap フレームワークの実装を行った．また，ケーススタディとして MashMap オブジェクトを可視化する MashMap Viewer を開発し，実際に位置付きライフログを可視化する過程から開発工数削減の見通しを得ているものの，実際に開発を行った検証をしておらず，どの程度の効果を得られるのか評価がされていない．

そこで本稿では，MashMapAPI を介して提案フレームワーク上に既に作成したオブジェクトを一覧しながら，各オブジェクトを画面上の簡単な操作で作成できる GUI フロントエンドとして MashMap Builder を実装する．さらに，提案フレームワークの有用性を確かめるため，複数の種類のライフログを地図上に可視化する評価実験を行う．

2. 準　　備

2.1 位置付きライフログ

位置情報が含まれたライフログを，本稿では**位置付きライフログ**と呼ぶ．位置付きライフログはライフログの一種であるため，基本的には人間の何らかの行動やイベントの記録を表している．各記録には，その行動が発生した日付や時刻，関係するユーザが紐付けられる．加えて位置付きライフログは，その行動が発生した位置 (場所) の情報も含んでいる．具体的には，その場所の [**緯度, 経度, 高度**] **の組**，もしくは**住所**で表現される．位置情報は記録する行動の内容に強く依存しない汎用的なデータであり [4]，写真やつぶやき，運動記録など様々なライフログに付加することができる．

近年のスマートフォンの普及などにより，ライフログに位置情報を付加することも一般的になってきているため，様々な分野でのサービス応用が期待される．

2.2 位置付きライフログを用いたサービス例

位置付きライフログを用いたサービスをいくつか紹介する．図 1(a) は，ウェザーニュース社のウェザーリポートチャンネル [5] の画面である．このサービスでは，ユーザがレポートしたその地点のピンポイントな天気を同一の地図上に可視化し共有できるサービスである．ユーザは天気の他，コメントや写真などもレポートとして記録でき，それらを共有できる．

図 1(b) は，みんなでつくる放射線量マップ [6] の画面である．福島原発事故以来開発されたこのサービスは，参加者がそれぞれ自ら放射線量を測定し，測定地点の場所情報と共にサーバに送信する．各地で測定された放射線量は，同一の地図上に集約されて描画され，広範囲の放射線量マップが作成される．

これらの他にも，GARMIN [7] や foursquare [8]，Twitter [9]，Facebook [10] など，様々な位置付きライフログサービスが存在する．図 1 で示したように，位置付きライフログを用いたサービスでは，表示するデータの内容や表示形式はそれぞれ異なるものの，地図上にデータを可視化することは共通している．

a. ウェザーリポート　　b. みんなでつくる放射線量マップ

図 1　位置付きライフログを用いたサービスの例

この共通した処理は，現在各サービス提供者が独自に開発しており，再利用されていない．また，位置情報を用いた機能も各サービスでばらばらで，位置付きライフログを必ずしもフル活用できていない．

これらの課題に対処するため，様々な形式の位置付きライフログを可視化するアプリケーションフレームワークとして，我々は先行研究 [1] において MashMap フレームワークを提案している．その概要を 3.1 節で述べる．

3. 先行研究:MashMap フレームワーク [1]

3.1 概　　要

MashMap フレームワークとは，様々な形式の位置付きライフログを地図上に可視化するためのアプリケーションフレームワークである．MashMap とは，様々な位置付きログをマッシュアップ (Mashup) して地図 (Map) に表示するという意味を込めた造語である．フレームワークにおいて開発者は，位置付きライフログから自分の必要なデータを選んでデータの源泉 (**データソース**, data source) とし，複数のデータソースをマッシュアップした地図，すなわち MashMap を作成する．

図 2 にフレームワークの全体像を示す．図の最上部は，ユーザが様々なサービスで記録した位置付きライフログを表している．これら様々な形式の位置付きライフログデータを，提案フレームワークのデータベース (DB) へ随時インポートしておく．インポートの際に，我々の研究グループが提案している [2] [3] **ライフログのための標準データモデル (Life Log Common Data Model, LLCDM)** 形式に適宜変換する．LLCDM は 5W1H の観点からライフログが備えるべきデータ項目を整理し，特定のサービス・アプリケーションに依存しないデータストアを定義する．これによりライフログへの統一的なアクセスが可能となる．また，アプリケーションに依存するログの内容に関しては自ら解釈せず，外部のスキーマに任せるという構造をとっている．

インポートされたライフログは，LLCDM 形式に変換されたライフログデータにアクセスするための標準的なインタフェースとして提供される**マッシュアップ API (Life Log API, LLAPI)** を介してアクセスできる．

アプリケーションの開発者は，DB に蓄積されたライフログデータから自分が必要なものを選別して，データソースを作成する．データソース作成においては，データの選別に用いる

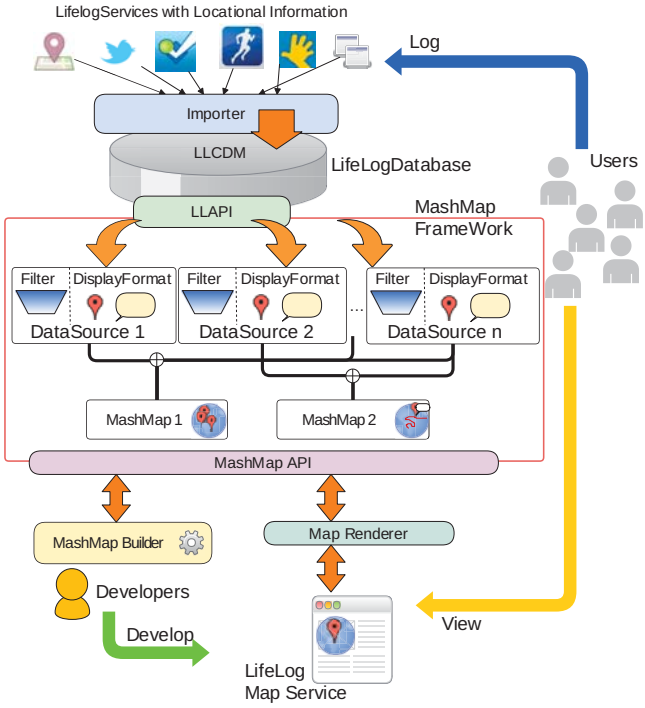

図 2　MashMap フレームワークの全体図

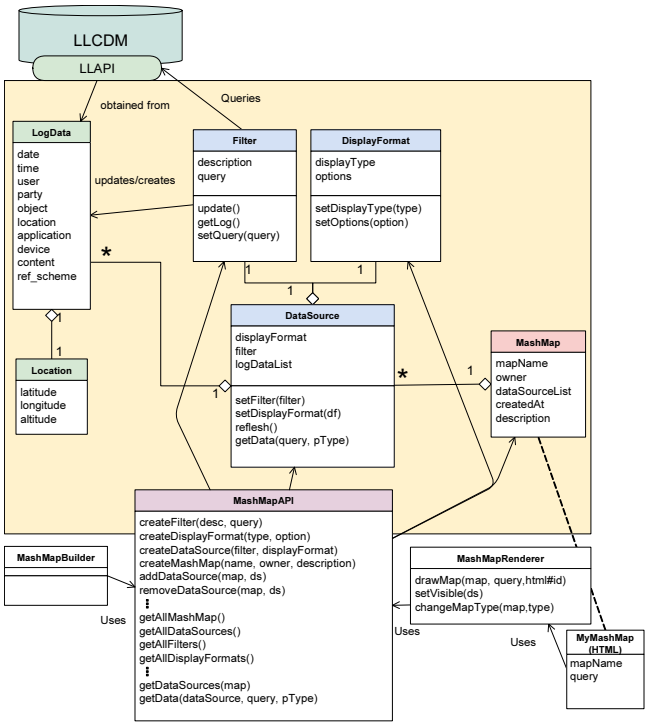

図 3　MashMap フレームワークのクラス図

フィルタ (filter) と，選別したデータをどのように地図上に表示するかを定義する**表示形式** (display format) を指定する．

次に開発者は，MashMap オブジェクトを作成し，作成済みのデータソースに接続する．異なるデータソースを複数組み合わせて接続することが可能である．こうして作成された MashMap オブジェクトは，MashMap Renderer によって地図形式に可視化される．地図上では，MashMap 内の複数のデータソースが，それぞれ指定された表示形式で描画され，複数の位置付きライフログがマッシュアップされた地図ができ上がる．

MashMap の作成やデータソースの設定などは全て MashMapAPI を通じて行われる．MashMapAPI は MashMap フレームワークの各種オブジェクトを操作するためのインタフェースを提供する．詳しくは次節で述べる．フレームワークのコアの部分 (図 2 中央四角部分) をクラウド上のサーバに配置して，MashMap フレームワークをサービスとして使う (Software as a Service, SaaS) ことも可能である．

我々は，MashMap フレームワークを GoogleAppEngine 上に Java 言語で実装した．API 実装には JAX-RS (jersey: JAX-RS (JRS311) Reference Implementation for building RESTful Web service) を利用し，RESTful な API を提供する．

3.2　MashMapAPI

図 3 は，提案フレームワークのクラス図である．LLAPI からのライフログデータの取得には Filter のみが関連し，データの表示方法の定義には DisplayFormat のみを用いる．それらの組み合わせて 1 つの DataSource として扱い，MashMap オブジェクトは，1 つまたは複数の DataSource をそれぞれ可視化する仕組みになっている．

図の中央下部に表されている MashMapAPI は MashMap フレームワークの各種オブジェクトを操作するためのインタフェースを提供するファサードクラスである．大きく分けて 3 種類に分類できる．1 つ目は，Filter, DisplayFormat, DataSource 及び MashMap の各種オブジェクトの生成・消去・更新を行う API である．例えば 2012 年 9 月 13 日にあるユーザ (koupe) が GARMIN を利用して取得した GPS ログを抽出するためのフィルタを作成するには以下のような API 呼び出しを行う．

```
createFilter("沖縄旅行 3 日目の GARMIN ログ",{user:
  koupe,application:GARMIN,s_date:2012-09-13,...);
```

2 つ目は，各種オブジェクトの一覧を取得する API である．これは，既に作成済みであるそれぞれのオブジェクトを再利用する際や参照が必要なときに利用する．Filter の場合は以下の API を実行する．

```
getAllFilters();
```

3 つ目は，主に MashMap オブジェクトを実際に可視化する際に必要となる API で，MashMap オブジェクト自体に付随する情報や，紐付けられている DataSource のリストを取得するものと，各 DataSource の定義に従ってログデータの系列を取得したりすることができる API がある．ある MashMap を取得して，それに紐付けされた各 DataSource についてログデータの系列を取得するには以下のようにする．

```
dataSourceList = getDataSources(mashMapId);
foreach(ds:dataSourceList){
  getData(ds.id, "*", "json");
}
```

3.3　現状の課題

3.2 節で触れたように，提案フレームワークには各オブジェクトを操作・取得する様々な API が用意されているが，これらの API の内 MashMap オブジェクトを作成する過程において

使用する API はほとんど同じである．さらに，既に作成したオブジェクトを再利用すれば効率的な開発が行える．しかし，たくさんの API が備えられているが故に，開発者は最初に何をすればいいのか混乱してしまうかもしれない．開発者がこれらの API を気軽に使い，試せるような仕組みがあれば，位置付きライフログのわかりやすい可視化を促進すると共に，より付加価値のあるライフログマッシュアップの一助になると我々は考える．

一方，先行研究 [1] では，MashMap フレームワークの実装を行った．また，ケーススタディとして MashMap オブジェクトを可視化する MashMap Viewer を開発し，実際に位置付きライフログを可視化する過程から開発工数削減の見通しを得ているものの，実際に開発を行った検証をしておらず，どの程度の効果を得られるのか評価がされていない．

そこで本稿では，MashMapAPI を介して提案フレームワーク上に既に作成したオブジェクトを一覧しながら，各オブジェクトを画面上の簡単な操作で作成できる GUI フロントエンドとして MashMap Builder を実装する．さらに，提案フレームワークの有用性を確かめるため，複数の種類のライフログを地図上に可視化する評価実験を行う．

4. MashMap Builder

4.1 機 能 概 要

MashMap Builder は，MashMapAPI の開発者向け GUI フロントエンドである．MashMap オブジェクトを画面上の簡単な操作で作成することが可能である．画面上で `Filter`, `DisplayFormat`, `DataSource` をそれぞれ定義・選択し，新たな MashMap オブジェクトを簡単かつ迅速に作成することができる．次節で具体的な利用例と共に説明する．

4.2 利用シナリオ

シナリオとして,「あるユーザ (koupe) が，2012 年 9 月 13 日に取得した GARMIN の GPS ログを，赤色の線で地図上に可視化したい」場合を設定する．

まず，Filter を作成する．図 4 に実際の画面を示す．Filter 作成用のフォームと，既に作成された Filter のリストが表示されている．このように，MashMap Builder には各オブジェクト毎の作成画面が用意されている．開発者は Filter の説明を表す description の他，LLAPI のクエリに対応する start date, end date, start time, end time, user, application の項目をフォームを埋め，下部の create ボタンを押すことでオブジェクトが作成される．記述の仕方は以下のような具合である．

`description` : 沖縄旅行 3 日目の GARMIN ログ
`start date` : 2012-09-13
`end date` : 2012-09-14
`start time` : 00:00:00
`end time` : 00:00:00
`user` : koupe
`application` : GARMIN

次に，DisplayFormat を作成する．Filter と同様にフォームを埋める形式である．以下に例を示す．ここで，DisplayType は Marker, Line, InfoWindow の 3 択になっている．

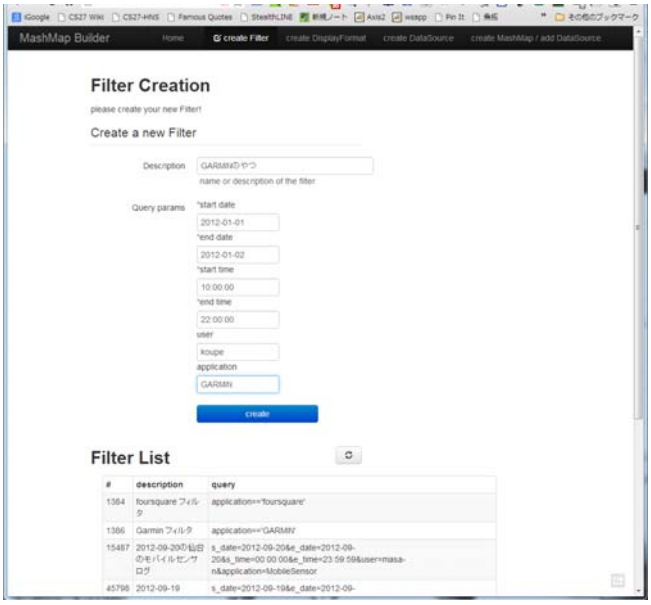

図 4　Filter 作成画面

`DisplayType` : Line
`Options` : strokeColor:”red”

作成した 2 つのオブジェクトから，DataSource を作成する．Filter 及び DisplayFormat それぞれの ID を入力することで DataSource は定義される．同画面上には既に作成された DataSource だけでなく Filter，DisplayFormat のリストも表示されているため，そのリストからそれぞれ選んで入力する．

最後に MashMap オブジェクトを作成する．まず，MashMap オブジェクト自体の情報として name や description を入力して MashMap オブジェクトを作成し，その後 DataSource を追加する．MashMap オブジェクト作成時のフォーム入力例は以下のようになる．

`name` : 沖縄旅行 3 日目
`description` : 2012 年 9 月 13 日分の沖縄旅行時ログまとめ

DataSource の追加には，追加する DataSource の ID と，追加先の MashMap の ID を同画面のリストより選んで入力する．複数の DataSource を追加していくことで複数の種類のライフログを同一地図上に可視化することができる．また，作成済みの MashMap オブジェクトのリストをクリックすると，その MashMap オブジェクトを実際に可視化した結果を確認することができる．その画面が図 5 である．MashMap 表示画面では，地図上に可視化されている DataSource 毎に表示/非表示を切り替えて確認することができる．

4.3 実　　装

MashMap Builder は HTML 及び JavaScript によって実装された Web アプリケーションであり，JavaScript ライブラリとして提案フレームワークに含まれる MashMap Renderer に加え，jQuery (v1.8.2), さらに，CSS フレームワークとして

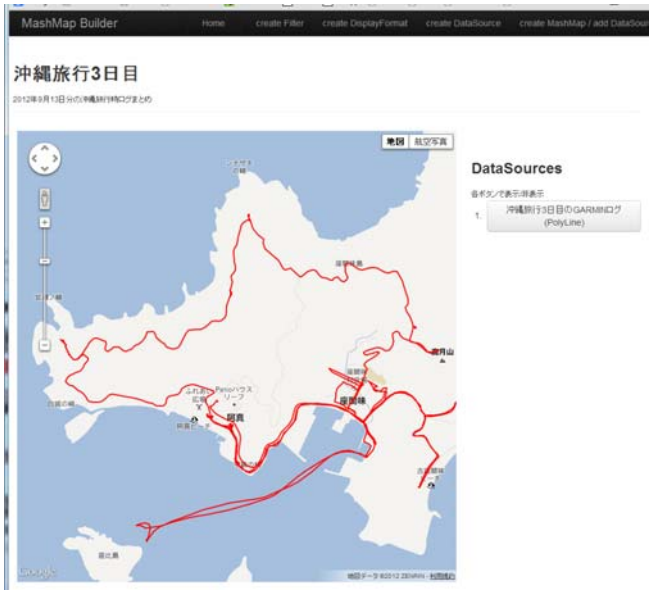

図 5　MashMap 表示画面

Twitter Bootstrap (v2.1.1) を利用した．プログラムの有効行数は，HTML ファイル 7 ファイルで 761 行，JavaScript は 334 行であった．

5.「旅行振り返りマップ」開発実験

5.1「旅行振り返りマップ」

この節では，評価実験において被験者に作成してもらう「旅行振り返りマップ」について説明する．旅行振り返りマップでは，旅行中にユーザの収集したライフログを同一地図上に可視化することにより，旅行の思い出を実データと共にわかりやすく振り返ることができるアプリである．実験では，ある旅行中に GARMIN により収集した GPS ログと，Twitter にポストしたツイートログを同一地図上に可視化する．画面のサンプルを図 6 に示す．あるユーザが 2012 年 9 月に沖縄旅行をした内，2012 年 9 月 13 日の 1 日分のライフログを可視化している．可視化の方法はそれぞれ，GPS ログは赤色の線で表示されている．ツイートログは，まずツイートした地点に青色のフキダシアイコンを表示し，アイコンをクリックするとフキダシが表示される．フキダシの中身はツイッターのアカウント名とアイコン画像，及び発言内容となっている．

この旅行振り返りマップを作成する作業行程はそれぞれ以下のようになる．

- **従来手法**

Step1: LLAPI を介して，可視化するライフログのデータを取得する．

Step2: GoogleMaps JavaScriptAPI を利用してライフログデータを地図上に可視化する．

- **提案手法**

Step1: MashMap Builder を利用して必要な Filter, DisplayFormat, DataSource, MashMap の各オブジェクトを作成する．

Step2: JavaScript を利用して MashMap オブジェクトを地図上に可視化する．その際，MashMap Renderer (JavaScript ライブラリ) を用いる．

図 6　旅行振り返りマップの画面

5.2 実験概要

MashMap フレームワークの有用性を確かめるために，ライフログを地図上に可視化する一連の処理を実装する評価実験を行う．実験では，複数の種類のライフログを地図上にマッシュアップして可視化する「旅行振り返りマップ」を作成してもらう．提案フレームワークを利用した場合（提案手法）と，利用しない場合（従来手法）における開発工数と，実験後に実施したインタビューの評価項目に基づき，提案フレームワークの有効性について評価を行う．被験者はシステム開発経験はあるが，GoogleMaps API については経験のない 40 代男性 1 名で，実施の順番は提案手法による開発を先に行い，従来手法による開発を後から実施した．

また，4. 節で述べたように，MashMap Builder には MashMap オブジェクトを作成した時点でその MashMap を表示する機能があるが，この機能は確認用途のみとし，ソースコードを表示したり，コピーするといったことは禁止した上で実験を実施した．本実験では，まず，提案手法・従来手法どちらにも依存しない要素のみが記述された HTML ファイルに編集を加えてくという形で開発に着手してもらった．

5.3 評価項目

本実験では，提案手法・従来手法それぞれにおいて以下の項目を測定する．

A: 学習に要した時間

B: 開発に要した時間

C: ソースコード行数

D: 可視化されたデータ（GPS ログ及びツイートログ）のレコード数の正確性（作成レコード数/正解数 [%]）

項目 A により，各手法における学習コストの違いを明確にする．この時間は 2 度目以降の開発において大幅に短縮可能な時間である．項目 B では学習以外にかかった時間を測定しており，2 度目以降の開発において項目 A 程の短縮を見込めない時間と思われる．また，各被験者は 1 人で開発を行うため，項目 A の時間と項目 B の時間の合計が全体の工数となる．この他，一般的に製品の大きさを特徴付ける指標として項目 C を利用する．測定するのは JavaScript のコード行数である．項目 D では，可視化されるべきデータが全て表示されているかどうかを確かめ，成果物の完成度を評価する．正解数は，LLAPI から取得できる該当期間のデータの数をカウントした．

開発終了後，被験者の主観的な意見やフィードバックを収集するため，アンケートを行った．アンケートの内容は，以下の 3 つである．

Q1: 提案手法により開発が楽になりましたか？(5 段階)

Q2: 従来手法と比べて，提案手法で楽であった点があれば教えてください．

Q3: 従来手法と比べて，提案手法で苦労した点があれば教えてください．

5.4 結　　果

実験に際し測定した各項目の結果を表 1 にまとめた．各項目において提案手法による工数削減が見られる．

実験後に行ったアンケートでは Q1 では 5(とても楽になった) の評価を得られた．Q2 では「GoogleMapsAPI をほとんど覚えなくてよかった」点,「LLAPI からのデータ取得に際し，RPC の方法を覚えなくてよい」点や,「作成した地図をクリックで確認できる」点に加え,「従来法では JavaScript と GoogleMapsAPI についてきちんと理解していないと非常に時間がかかりバグ取りも大変だったが，提案フレームワークではそれが要らない」点が挙げられた．Q3 においては「MashMap Builder で作成したオブジェクトが ID 番号で管理されており，番号を直接入力しなければならず混乱があった」点,「過去に作ったオブジェクトの手直しができず，作り直さなければならなかった」点など MashMap Builder の操作性についての問題が挙げられた．

5.5 考　　察

実験結果からもわかるように，学習コストを含め開発工数を大幅に削減できている他，コードの記述量も従来手法に比べてきわめて少なくなっており，提案手法により開発者の支援に貢献したといえる．また，データのレコード数はどちらも問題がなかった．

開発時間は提案手法では従来手法の半分以下に抑えられており，効率的な開発を支援できたと考える．学習時間は，従来手法の 2/3 程度の時間を必要としたが，提案フレームワークについての学習以外に学習の必要な部分が必要最低限に抑えられたためだと思われる．具体的には表示形式の定義の際に GoogleMapsAPI の知識が必要となるが，これはどちらにも共通に必要な知識であり，提案手法による開発を先に実施したにも関わらずこのような結果となった事は成果であると思われる．さらに,「MashMap Builder の操作への慣れで更に工数が削減できそうだ」という意見も得られた．

その一方で MashMap Builder の操作性により，開発者に逆に不便を強いてしまった場合があった．各オブジェクトの編集・削除といった機能や，オブジェクトの ID を手入力ではなく，リストから選択するように実装することで解消できると考える．

表 1　実 験 結 果

項目	提案法	従来法
A	57 分 43 秒	92 分 57 秒
B	26 分 53 秒	64 分 37 秒
A+B	84 分 37 秒	157 分 34 秒
C	8 行	139 行
D	100%	100%

6. おわりに

本稿では，MashMapAPI を介して画面上の簡単な操作で提案フレームワーク上のオブジェクトを一覧・作成できる開発者向け GUI フロントエンドとして MashMap Builder の実装を行った．また，開発実験を行い，提案フレームワークの有効性を評価した．実験では GARMIN で取得した GPS ログと，Twitter にポストしたツイートログを同一地図上に可視化するウェブアプリケーション「旅行振り返りマップ」の開発を行った．その結果，提案フレームワークを利用することにより開発工数が削減できることが示された．また，同一地図上に可視化するデータソースが増えれば増えるほど，更なる工数削減が期待される．

今後の課題としては，実験後のアンケートで指摘されているように，MashMap Builder の操作性を向上させることがある．それだけでなく，MashMap フレームワークを強化し，位置付きライフログと位置付きでないライフログをマッシュアップさせるような機能を実現することなどが挙げられる．

謝辞　この研究の一部は，科学技術研究費（基盤研究 C 24500079, 基盤研究 B 23300009），及び，関西エネルギー・リサイクル科学研究振興財団の助成を受けて行われている．

文　　献

[1] 高橋昂平，下條　彰，松本真佑，中村匡秀，“位置情報を含むライフログの可視化サービス開発支援フレームワーク，” 信学技報，vol.111，no.470，pp.183–188，March 2012.
[2] まつ本真佑，下條　彰，鎌田早織，中村匡秀，“異種ライフログ統合のための標準データモデルとマッシュアップ api，” 電子情報通信学会論文誌，vol.J95-D，no.4，pp.758–768，April 2012.
[3] A. Shimojo, S. Matsumoto, and M. Nakamura, “Implementing and evaluating life-log mashup platform using rdb and web services，” The 13th International Conference on Information Integration and Web-based Applications & Services (iiWAS2011), pp.503–506, Dec. 2011.
[4] 中村匡秀，下條　彰，井垣　宏，“異なるライフログを集約するための標準データモデルの考察，” 信学技報，vol.109，no.272，pp.35–40，2009.
[5] ウェザーリポート Ch. http://weathernews.jp.
[6] みんなでつくる放射線量マップ．http://minnade-map.net/.
[7] GARMIN Connect. http://connect.garmin.com.
[8] foursquare. http://foursquare.com/.
[9] Twitter. http://twitter.com/.
[10] Facebook Places. http://facebook.com/places/.