

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

榎本 真佑^{†a)} 下條 彰[†] 鎌田 早織[†] 中村 匡秀[†]

Common Data Model and Mashup API for Integrating Heterogeneous Lifelogs

Shinsuke MATSUMOTO^{†a)}, Akira SHIMOJO[†], Saori KAMADA[†],
and Masahide NAKAMURA[†]

あらまし 様々なサービスを通じて記録された各種ライフログを、ネットワーク越しに統合・集約することで、より高い付加価値をもつサービスへ発展させることが可能である。本論文では、異種ライフログ統合のための標準データモデル (LLCDM)、及び標準データモデルにアクセスするためのマッシュアップ API (LLAPI) を提案する。各種ライフログの記録項目を 5W1H の観点から分類、整理することで、ライフログの種類に強く依存しない中立的なデータモデルを構築する。整理された標準データに対し、汎用的なマッシュアップ API を設けることで、効率的かつ柔軟なライフログの統合を支援する。LLCDM と LLAPI の有効性を確かめる評価実験として、二つのライフログを統合するアプリケーションを題材とした開発実験を行う。

キーワード ライフログ、マッシュアップ、標準データモデル、マッシュアップ API、Web サービス

1. ま え が き

ライフログとは、人間の行動が記録されたデジタルデータのことであり、その記録対象は日記やつぶやきから、体重・心拍数などの生体情報まで多岐にわたる。長期にわたるライフログの蓄積は、自己の振り返りや日々の生活改善、あるいは新たな自己の発見を促す。また複数人でライフログを共有することで、コミュニティ内での共通の話題の提供や、個人の嗜好に応じた推薦サービスなどが享受できる。

計算機技術の発展に伴い、現在では多種多様なライフログサービスがインターネット上に公開されている。これらのサービスを利用することで、ユーザは好みのライフログを Web 上で記録・共有することができる。具体的なライフログサービスの例としては、個人の日記や考えをテキストとして記録するブログから始まり、つぶやきを共有する Twitter [1]、画像や写真を共有する Flickr [2]、毎日の食事を記録する FoodLog [3]、移動経路情報（緯度と経度）を記録する GARMIN Connect [4] などが存在する。

一般にこれらのライフログサービスでは、ライフロ

グの「記録」と「利活用」は単一のサービス内に閉じた形で行われている。一方で、様々なサービスによって記録された複数のライフログを統合・集約（マッシュアップと呼ぶ）することで、より高い付加価値をもつサービスへと発展させることができる。例えば、体重ログと食事の写真ログをマッシュアップすることで、食生活の改善を支援するサービスが実現できる。既存のライフログサービスの中には、マッシュアップ用の API を用意しているものも存在しており、外部のアプリケーションから各種ログデータを操作することができる。しかし、これら API の仕様はサービスごとに異なっており、サービスの組合せごとに異なる処理ロジックが必要となる。

ライフログの高度かつ柔軟なマッシュアップを支援するために、本論文ではライフログマッシュアップのための標準データモデル (LLCDM)、及び標準データモデルにアクセスするためのマッシュアップ API (LLAPI) を提案する^(注1)。まず、標準的なライフログに必要なデータ項目を、5W1H の観点から分析し各種ライフログに独立なものと依存するものに分類する。この分類結果に基づき、ライフログの種類に依存しにくい標準データモデルを構築する。更に、このデータ

[†] 神戸大学大学院システム情報学研究所, 神戸市
Graduate School of System Informatics, Kobe University,
Kobe-shi, 657-8501 Japan

a) E-mail: shinsuke@cs.kobe-u.ac.jp

(注1)：本論文は筆者らの先行研究 [5]～[7] をもとに、システムの改善と再実験を行い、論文としてまとめたものである。

モデルを操作するための汎用的なマッシュアップ API を設計・実装する。

LLCDM 及び LLAPI の有効性を確かめるために、マッシュアップアプリケーションの開発実験を行った。実験の結果、LLCDM と LLAPI を利用することで開発工数を約 1/3 程度に削減できること、及びプログラムの信頼性を確保できることが明らかとなった。

2. 準備

2.1 マッシュアップ

一般的にマッシュアップとは、ネットワーク上で点在する様々な情報資源を何らかの観点で統合・連携し、付加価値を創造する行為を指す。このマッシュアップは、統合対象となるリソースの種類やその目的によって以下の三つに分類される [8]。コンシューマ・マッシュアップは、Web プログラミング上で広く知られるものであり、Web 上で提供されている各種サービスの API を利用し、一つサービスとして統合することを指す。データ・マッシュアップは複数のサービスのもつデータを統合し、新たな統合済みデータを作り出すことを指す。ビジネス・マッシュアップは、企業がもつ独自のサービスや情報と、公開されている外部の各リソースを統合し、ビジネス目的に役立てる行為を指す。

本論文では、マッシュアップという言葉でデータ・マッシュアップの意味で用いる。更に厳密な定義として、マッシュアップを「異なる二つのデータをある属性の一致、あるいは類似によって一つのデータに統合する行為」と定義する。

2.2 ライフログマッシュアップ

ライフログの記録と共有を支援するサービスを、本論文ではライフログサービスと呼ぶ。1. でも述べたとおり、現在では様々なライフログサービスが公開されている。エンドユーザは好みのライフログを、好みのサービスを利用して蓄積・共有することができる。

各サービスにより提供される複数のライフログデータを、ネットワーク越しにマッシュアップすることで、ライフログを単独で利用する以上の付加価値を生み出せる可能性がある。本論文では、複数のライフログを統合することにより得られる付加価値創造を、ライフログのマッシュアップと呼ぶ。例えば、体重ログと食事の写真ログをマッシュアップすることで、体重の変化と食事内容を同時に振り返り、食生活の改善に活用することができる。また、つぶやきログと睡眠ログのマッシュアップにより、悩みごとと睡眠時間の関係に

対する気づきを与えることができるかもしれない。

2.3 課題

一般にライフログの記録と利用は、同一のライフログサービス内に閉じた形で行われている。そのためライフログのデータは、サービス提供者によって異なる形で記録・管理されている。ライフログサービスの中にはマッシュアップを想定して、サービス API やログパーツなどのアクセス手段を公開しているものも存在する。外部のサービスやアプリケーションは、これらのアクセス手段を利用することで、ライフログを再利用、あるいはマッシュアップすることが可能となる。しかしながら、ライフログサービスの API やデータ構造には統一的な標準がなく、各サービスごとに異なっているのが現状である。

各ライフログサービスが提供するデータ構造の違いの一例を図 1 に示す。図 1 (a) は Twitter, 図 1 (b) は Flickr の API 呼出しにより得られるデータを指す。それぞれ、ユーザ shimojo が 2010 年 5 月 22 日にユー

```
<status>
  <created_at>Sat May 22 16:31:55 +0000 2010
  </created_at>
  <id>14503299100</id>
  <text>@masa-n SATC2 was wonderful!!</text>
  <source>web</source>
  <in_reply_to_screen_name>masa-n
  </in_reply_to_screen_name>
  <user>
    <id>68573479</id>
    <screen_name>shimojo</screen_name>
  </user>
  <place>
    <geo>1-1, Nada, Kobe, Hyogo, Japan</geo>
  </place>
  <time_zone>Osaka</time_zone>
  <lang>ja</lang>
  ...
</status>
```

(a) Twitter API のデータ構造

```
<photo id="4270257663"
  owner="42626851@N05"
  title="lunch @kobe univ."
  datetaken="2010-01-13 14:14:04"
  ownername="shimojo"
  lastupdate="1263530688"
  latitude="34.72631"
  longitude="135.23532"
  media="photo"
  url_s="http://farm4.static.flickr.com/3343/
    4617769019_9c195d8772_m.jpg"
  height_s="240"
  width_s="144"
  model="RICOH CX3"
  ...>
</photo>
```

(b) Flickr API のデータ構造

図 1 ライフログデータ構造の違い

Fig. 1 Differences of lifelog data structure.

ザ masa-n への返答としてつぶやいた内容と、ユーザー shimojo が 2010 年 01 月 13 日に撮影した昼食写真のメタデータが記録されている。どちらのデータも XML 形式ではあるが、データ全体の構造やタグの名前、値の書式などが異なる。例えば日付を表すタグは、Twitter では<created_at>、Flickr では<datetaken>であり、またその値の書式も異なる。

図 2 に現在のライフログのマッシュアップ形態を示す。現状、マッシュアップアプリケーションは、個々のライフログサービスに強く依存した形で実現されている。まず、ライフログデータを取得する場合、サービスごとに異なる呼出し方法が必要である。具体的には、サポートしている API プロトコルの種類、パラメータの指定方法や書式などがサービスごとに異なる。更に、取得したデータのマッシュアップ処理は、取得データの構造や書式に依存するため、統合させるサービスの組合せによって異なる処理ロジックが必要である。このようなサービスに強く依存した従来のマッシュアップ形態は、マッシュアップアプリケーションの開発効率と再利用性を低下させ、更には信頼性を低下させる原因になる。

2.4 目的とアプローチ

本論文の目的は、前節で述べたライフログマッシュ

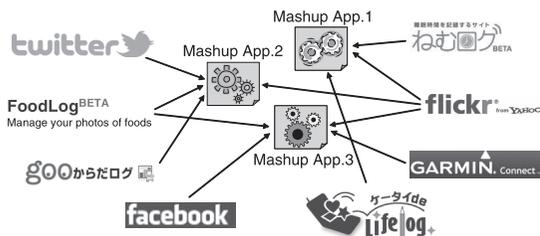


図 2 従来のライフログサービスのマッシュアップ
Fig. 2 Conventional approach of lifelog mashup.

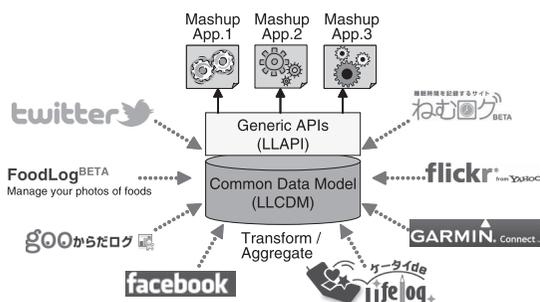


図 3 提案手法によるライフログサービスのマッシュアップ
Fig. 3 Proposed approach of lifelog mashup.

アップに対する課題を解決し、ライフログサービスに依存しないマッシュアップ方法を提供することである。本論文では、ライフログのための標準データモデル、及びマッシュアップ API を設計・実装する。以降では、標準データモデルとマッシュアップ API をそれぞれ、LLCDM (LifLog Common Data Model) と LLAPI (LifeLog Application Program Interface) と呼ぶ。

図 3 に提案手法を用いたライフログのマッシュアップ形態を示す。各ライフログサービスのデータは、何らかの形で、ライフログサービスに依存しない標準的なデータモデル (LLCDM) に変換される。この標準データモデルは汎用的なマッシュアップ API (LLAPI) から呼び出すことが可能である。ライフログのデータ構造、及びアクセス手段が各ライフログサービスに強く依存しなくなるため、容易かつ柔軟なマッシュアップの実現が可能となる。

3. LLCDM : 標準データモデル

3.1 概要

LLCDM のアーキテクチャを図 4 に示す。図中左側の円柱が、LLCDM に従ったライフログデータ保持のためのリポジトリを表す。円柱内下側、角の丸い四角が LLCDM に変換された個々のライフログデータを表す。ライフログデータは、ライフログサービスに非依存なデータ (標準データ) と依存するデータ (固有データ) をもつ。

ライフログの種類に非依存なデータ項目は、LLCDM のスキーマ (標準スキーマ) に組み入れる。LLAPI は標準スキーマを利用して、ライフログサービスに依存

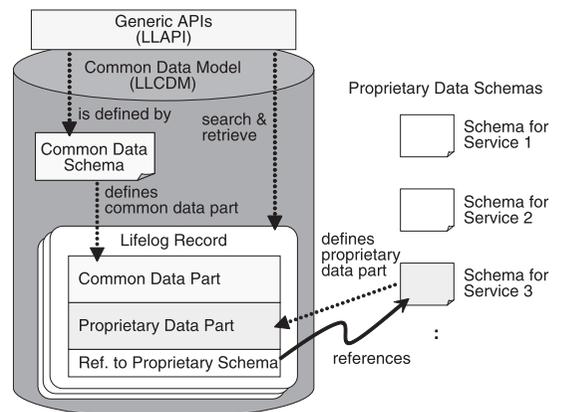


図 4 標準データモデル (LLCDM) のアーキテクチャ
Fig. 4 Architecture of lifelog common data model.

表 1 LLCDM の標準データスキーマ
Table 1 Common data schema of LLCDM.

分類の観点	スキーマ名	概要	例
What	<content>	オリジナルのデータ全てを含む構造データ	<photo id=".." owner=".." title=".."
	<ref_schema>	各サービスの固有スキーマへの参照	http://www.flickr.com/services/api/
Why	n/a	n/a	n/a
Where	<address>	ログが生成された住所	1-1, Nada, Kobe
	<latitude>	ログが生成された緯度	34.72631
	<longitude>	ログが生成された経度	135.23532
When	<date>	ログが生成された日付	2010-03-05
	<time>	ログが生成された時刻	12:00:00
Who	<user>	ログを生成したユーザ	shimojo
	<party>	ログ生成時に共に行動していたユーザ	saorin
	<object>	ログ生成の対象となったユーザ	masa-n
	<application>	ログ記録に用いられたアプリ, またはサービス	Flickr
How	<device>	ログ記録に用いられたデバイス	RICOH CX3

しないデータ操作を規定する。一方、ライフログサービス固有のデータ項目に関しては、単なる構造化されたデータとし、LLCDM 内ではその意味を定義・解釈しない。その代わりに、ライフログサービスごとのスキーマ（固有スキーマ）を別定義し、そこへの参照を保持する。

良いデータモデルと API を構築するには、ライフログに非依存なデータ項目をどれだけ洗い出せるかがかぎとなる。標準的なデータモデルを構築するために、ライフログを構成するデータ項目を 5W1H (What, Why, Where, When, Who & How) の観点から分析・抽出し、ライフログサービスに強く依存する項目と、非依存な項目に分類する。

5W1H からスキーマを整理する理由は、ログ間の共通項目の洗い出しやすさによるものである。ライフログとは人が主体となる履歴データのことであり、ほとんどのライフログデータは 5W1H の共通属性をもつと考えられる。例えば、「誰の、いつ、どこで、どのようにして」取得したログかといった情報は、ライフログの種類を問わず共通して保持されやすい。

3.2 5W1H からの標準データスキーマの分析

標準データ項目を 5W1H の観点から整理する。最終的に決定された LLCDM のデータスキーマを表 1 に示す。以降では、5W1H の各観点からの整理の指針を説明する。

- **What** : ライフログにおいて「何を」記録するかは多くのバリエーションがあるため、標準スキーマとして規定しにくい。例えば Twitter であればつぶやき<text>, Flickr であれば写真<photo>が What に該当する。したがって、What に相当するデータ項目は、それぞれのライフログサービスに特化した固有

データとなる。LLCDM においては、これらを特に区別・解釈せず、<content>というタグのついた単なる構造付きのデータとして扱う。更に<content>の解釈のために、各ライフログサービスの固有スキーマへの参照<ref_schema>を組み入れる。

- **Why** : そのライフログが「なぜ」記録されたかを表す項目としては、記録の目的<purpose>や理由<reason>, 契機<trigger>などが考えられる。しかし、これらの項目は明示的に記述されない限り外部から知ることはできない。ライフログサービスによっては、ログ蓄積の目的を説明できることもあるが（例えば、体重データの蓄積は痩せるため）、理由なく記録されたログもある。このように Why に関する情報はデータそのものではなく、データを利用するサービスに依存することから、標準スキーマからは除外する。

- **Where** : そのライフログが「どこ」で記録されたかは、ライフログサービスに依存しない標準的なデータ項目である。写真ログの場合は撮影した場所、体重の場合は計測した場所など、データに紐づく位置情報<location>は、データの種別を問わず説明できることが多い。したがって<location>を標準スキーマに組み入れる。なお、位置情報を表すデータ形式としては、緯度情報<latitude>と経度情報<longitude>, 及び住所<address>を用いる。

- **When** : そのライフログが「いつ」記録されたかは、ライフログサービスに依存しない標準的なデータ項目である。具体的な内容は、ログを記録した日付<date>と時刻<time>の二つである。ライフログサービスによっては、記録した地域のローカル時間を利用しているケースもあるが、このタイムゾーンの違いはサービスへの依存性を生む。標準スキーマでは時間

情報を全て協定世界時 (UTC +00:00) に変換し保持する。

- **Who**: ライフログが人間の行動の記録であることから、全てのライフログには「誰が」その行動を起こしたか、主体となるユーザ<user>がひもづけられると考えてよい。したがって<user>を標準スキーマに組み入れる。更に、その行動の主体が集団であった場合、誰と行動したかを表すデータ<party>や、行動の対象を表すデータ<object>も、ライフログサービスに依存しにくい標準データ項目である。これらは複数のユーザに関連づける有用なデータであり標準スキーマに組み入れる。

- **How**: そのライフログを「どのように」記録したのかは、理想的にはライフログサービスに依存すべきではない。例えば、どの写真共有サービス (例えば Flickr と Picasa) を利用しても、写真の記録内容に差異はない。また、同じ Flickr への投稿でも、PC を利用するか携帯を利用するかは写真の内容とは独立である。すなわち記録手段の相違によって、ログの内容や性質は変わらない。ゆえに、How に関するデータ項目は本来ライフログサービスに依存するものではないが、現実にはライフログサービスごとに提供される記録手段が異なるため、サービスへの依存関係が生まれている。便宜上 LLCDM では、どのサービスで記録されたか<application>、及びどのデバイスで記録されたか<device>の二つの項目を標準スキーマに組み入れる。

4. LLAPI: マッシュアップ API

4.1 概要

LLCDM に格納されたライフログのマッシュアップを支援する API としては、様々なものが考えられるが、本節では `getLifeLog()` と `getLifeLogNearestTime()` の二つについて説明する。本論文では 2.1 において、マッシュアップという言葉で、「ある属性が一致する異種ログ同士を紐付ける行為」と定義した。`getLifeLog()` はこの属性の一致を取る API であり、人やデバイスなどのラベル的な属性をキーとしたマッシュアップは、この API で対応することが可能である。一方、時刻や緯度経度といった定量的な属性は、その完全一致を取ることが難しいため、類似や区間をとるといったマッシュアップ方法が必要となる。本論文では時刻に関するマッシュアップ方法の一つとして、時刻の類似をとる API: `getLifeLogNearestTime()` について紹介する。

4.2 getLifeLog()

指定クエリにマッチするライフログを取得する API である。API のインタフェース、クエリ、返り値などの API の詳細は以下のとおりである。

インタフェース:

```
LifeLog[] getLifeLog(date, time, user,
                    party, object, location,
                    application, select);
```

引数:

```
date:      <date>に関するクエリ
time:      <time>に関するクエリ
user:      <user>に関するクエリ
party:     <party>に関するクエリ
object:    <object>に関するクエリ
location:  <location>に関するクエリ
application: <application>に関するクエリ
select:    抽出データに関するクエリ
```

クエリ:

リテラル (" "で囲まれた文字列)、論理和 (+)、ワイルドカード (*) から構成される検索式。

返り値:

クエリに合致するライフログデータのリスト (配列)。各ライフログデータは、表 1 の標準データ項目をメンバとしてもつ構造体 (ハッシュ) で表現される。

使用例:

ユーザ shimojo が 2010 年 03 月 01 日に Twitter と Flickr で記録したライフログデータの中から、日付、時間、ユーザ名、つぶやき、写真 URL、写真タイトルを抽出し、取得する。

```
getLifeLog({
    date      => "2010-03-01",
    user      => "shimojo",
    application => "Twitter+Flickr",
    select    => "date+time+user+
                text+url_s+title"})
```

4.3 getLifeLogNearestTime()

時刻などの定量的な属性に対しては、上記 `getLifeLog()` のみで支援することは難しい。ここでは時刻に関するマッシュアップを支援するために、時刻の類似を取る API: `getLifeLogNearestTime()` について紹介する。

インタフェース:

```
LifeLog getLifeLogNearestTime(date, time,
                               user, application);
```

引数：

date: <date>に関するクエリ
time: <time>に関するクエリ
user: <user>に関するクエリ
application: <application>に関するクエリ

返り値：

クエリに合致するライフログデータのうち、日時が最も近いライフログデータ 1 件。

使用例：

ユーザ shimojo が GARMIN で記録したライフログデータの中で、2010 年 03 月 01 日 12 時 34 分 56 秒に最も近いライフログデータを取得する。

```

getLifeLogNearestTime({
    date      => "2010-03-01",
    time      => "12:34:56",
    user      => "shimojo",
    application => "GARMIN"})
  
```

5. 実装

5.1 LLCDM の実装

ライフログサービス固有のオリジナルデータ形式から標準データへの変換には、オフライン方式とオンライン方式の 2 通りの実装方式が考えられる。それぞれ、あらかじめ変換済みの標準データを DB 中に蓄積しておく方式と、汎用 API からのリクエストが来るたびに、目的のライフログサービスにアクセスしデータを取得・変換して返す方式である。オフライン方式では最新データへのアクセスが制限されるが、高速で軽量のデータ検索・取得が可能である。逆に、オンライン方式では高いリアルタイム性をもつが、データの検索・取得のオーバーヘッドがかさむ。どちらの実装方式を採用するかは、支援するマッシュアップの目的や用途に合わせて選択するべきである。本論文ではデータの検索・取得速度を優先し、オフライン方式を採用した。

各ライフログサービスのオリジナルデータから、LLCDM への変換手順は以下のとおりである。

Step1. データマッピングの定義：オリジナルデータの各項目と LLCDM のデータ項目との対応を定義する。例えば、前述の図 1 (b) に示す Twitter データの場合、日付を表す<created_at>を LLCDM の<date>と<time>に割り当てるといった定義を決定する。

Step2. データ値の形式変換：オリジナルデータの値を LLCDM のデータ形式に統一する。先ほどの<created_at>の場合、"Sat May 22 16:31:55

+0000 2010"というオリジナル形式を"2010-05-22"と"14:14:04"に変換し、それぞれ LLCDM の<date>と<time>に割り当てる。表示形式の統一みならず、世界協定時への変換といったデータ内容の統一化も同時に行う。

Step3. <content>データの保存：変換済みデータの処理とは別に、LLCDM の What に該当する項目を保存する。具体的には、オリジナルデータの全てを LLCDM の<content>に保存し、オリジナルデータの外部スキーマへの参照先を<ref_scheme>に保存する。これは 3.2 で述べたように、What に該当するデータ項目を構造データとして扱うための処理である。

LLCDM のデータベース設計図を図 5 に示す。図中の +—E は親子関係を表し、+—... は参照関係を表す。各テーブルの下部にはインスタンスを併記している。UserInfo テーブルと Location テーブルは、それぞれユーザに関する情報と位置に関する情報をもつ。主となるライフログデータは LifeLog テーブルに保存され、各ライフログサービスの詳細と外部スキーマの定義は Application テーブルに保存される。

各ライフログサービスの変換プログラムは Perl を用いて作成し、LLCDM のデータベースには MySQL を用いた。データベースへの蓄積はオフライン方式として、cron によるバッチ処理呼出しを利用しており、1 日 1 回データの変換と DB への蓄積を行っている。現在では、Flickr, Twitter, BodyLogService, GARMIN Connect の四つのライフログが蓄積されている。

なお、各変換プログラムの作成時間は、API の解析を含め約 5 時間であった。従来のマッシュアップ形態の場合、マッシュアップサービスを作成するたびに同様のスクリプトを作成する必要がある。一方提案手法の場合、一度 LLCDM へ落とし込めば他のマッシュアップサービスにも再利用が可能であるという点で、LLCDM への変換工数は実用的な範囲であるといえる。

5.2 LLAPI の実装

全てのライフログデータが、標準スキーマに従ってデータベース中に蓄積されていれば、LLAPI は単なるデータベースの間合せプログラムとして実装できる。例えば 4.2 に示した getLifelog() は、図 5 に示したデータベースに対して、以下の SQL クエリを発行し検索結果をオブジェクトとして返せばよい。

```
SELECT * FROM LifeLog
```

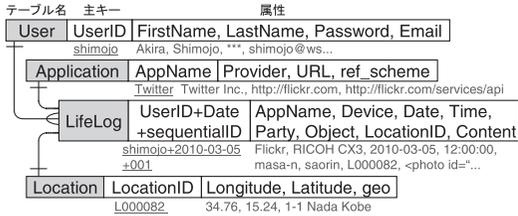


図 5 LLCDM データベースの ER 図
Fig. 5 ER Diagram of LLCDM.

```
WHERE Date LIKE $dateq
AND UserID LIKE $useridq
AND Application LIKE $applicationq
AND Device LIKE $deviceq
```

LLAPI の実装には Java 言語を用い、SQL クエリとオブジェクトのマッピングには O/R マッパーである iBATIS を用いた。O/R マッパーを利用することで、プログラムの処理ロジックと SQL クエリを分離することが可能である。実装された LLAPI は Apache Axis2 Web サービスフレームワークを用いて、Web サービスとしてデプロイした。これにより、LLAPI は SOAP/REST による HTTP プロトコルを利用した呼出しが可能となる。Web サービスの呼出し結果は、XML 形式で取得できるため、様々なクライアントから LLAPI を再利用することが可能となる。

6. 評価実験

6.1 概要

LLCDM 及び LLAPI の有効性を評価するために、具体的なマッシュアップアプリケーション TabetaLog を題材とした開発実験を行う。実験では、LLCDM と LLAPI の両方を用いる場合と用いない場合の 2 通りで開発を行い、開発効率と正確性の観点から提案手法の有効性を確かめる。

6.2 TabetaLog

TabetaLog とは、二つの異なるライフログサービスから体重ログと食事写真ログを取得し、一つの画面に可視化するマッシュアップアプリケーションである。TabetaLog を利用することで、体重の変化と食事内容を同時に振り返るといった体調管理の支援が可能である。TabetaLog のスクリーンショットを図 6 に示す。この画面はユーザ shimojo の最近 180 日の体重データをグラフで表したものである。グラフをクリックすると、その日の食事写真が画面左下に表示される。左上のユーザ名をクリックすることで、他ユーザのマ



図 6 TabetaLog のスクリーンショット
Fig. 6 Screenshot of TabetaLog.

ッシュアップデータを確認することができる。

マッシュアップ対象となる二つのライフログサービスは以下のとおりである。

- **BodyLogService**: 筆者らの研究室で開発した体重管理サービスである。ユーザは体重計を使用して体重と体脂肪率を測定し、その値を記録する。現在、BodyLogService は 5 種類の API が実装されており、日付やユーザなどの条件から各データを取得できる。

- **Flickr**: Yahoo! Inc. が提供しているオンライン写真アルバムサービスである。撮影した写真や画像をウェブ上で整理・分類・展示できるほか、他ユーザの写真を閲覧・コメントすることができる。現在、Flickr は約 180 種類の API を提供しており、これらの API を利用して様々な条件から写真を検索・取得できる。

TabetaLog の処理工程は以下に示す四つのステップから構成される。

Step1. ライフログデータの取得: 体重データと写真データを、Flickr API と BodyLogService API の両方を利用、あるいは LLAPI のみを利用して取得する。

Step2. 必要なデータ項目の抽出: 取得した 2 種類のライフログデータから必要なデータ項目のみを抽出する。体重データからは日付とユーザ、体重の値を、写真データからは日付と時刻、写真 URL を抽出する。

Step3. 日付とユーザ名でマッシュアップ: 抽出された二つのライフログデータを、日付とユーザ名をキーとしてマッシュアップする。マッシュアップ結果は JSON^(注2)形式のテキストデータとして出力する。

Step4. 可視化: マッシュアップした JSON データを ActionScript を用いて可視化する。体重データは

(注2): 構造化されたデータを記述するためのテキストベースのデータ記述言語。各種プログラミング言語に非依存な書式で記述される。

```

{
  number: 56,
  user: "shimojo",
  start: "2010-05-18",
  end: "2011-05-17",
  - data: [
    - {
      date: "2010-05-18",
      weight: 63.4,
      - foods: [
        - {
          URL: "http://farm5.static.flickr.com/...jpg",
          time: "12:30:28",
        },
        - {
          URL: "http://farm5.static.flickr.com/...jpg",
          time: "19:12:02",
        }
      ]
    },
    - {
      date: "2010-05-19",
      weight: 63.5,
      - foods: ...
    },
    ...
  ]
}

```

図 7 JSON ファイルの書式
Fig. 7 JSON file format.

折れ線グラフで表示され、食事の写真データは、ひもづけられた日付の体重データ（グラフ上の点）をクリックすることで画面左端に表示される。

6.3 実験内容

前節で示した TabetaLog の処理工程のうち、ライフログデータの取得・抽出・マッシュアップの三つの工程 (Step1 から Step3) を処理するプログラムの開発を行ってもらう。Step4 の可視化工程に関してはマッシュアップに非依存な工程であるため、実験からは対象外とした。実験では LLCMD と LLAPI を用いる提案法 (P_{prop}) と、各ライフログサービスの API をそれぞれ利用する従来法 (P_{conv}) の二つを比較し、提案手法の有効性を確かめる。

取得するデータは、2名のユーザ (shimojo と tokunaga) の体重データと写真データである。取得期間は 2010 年 5 月 18 日からの 1 年分とした。出力結果となる JSON ファイルの形式や構造についてはこちらで指定した。出力形式は図 7 のとおりである。

本実験の被験者は、情報系の修士学生 3 名と教員 2 名の計 5 名である。被験者には P_{prop} と P_{conv} の二つの開発方式を実施してもらうが、実験による慣れの効果を排除するために、実施する実験の順序はこちらで指定した。それぞれ P_{prop} 先行が 2 名、 P_{conv} 先行が 3 名である。なお、使用する言語と開発環境は被験者の自由とした。また P_{conv} 方式において各ライフログサービスの提供する API のうち、どれを利用するか

表 2 評価対象と尺度
Table 2 Evaluation perspectives and metrics.

評価対象	評価尺度
ソースコード	総行数、ファイル数
開発プロセス	工数 (分)
出力 JSON ファイル	取得体重ログ数、取得写真ログ数

についても自由とした。

6.4 評価尺度

実験の成果物及び開発プロセスを評価対象とする。評価対象と評価尺度を表 2 に示す。ここでの成果物とは、開発されたソースコードファイルと出力結果である JSON ファイルの二つを指す。これら二つの成果物は P_{prop} と P_{conv} で、それぞれで別に作成される。

ソースコードファイルからは、プログラムの規模を表す尺度として広く用いられる総行数とファイル数を計測する。開発プロセスからは開発に要した時間 (工数) を計測する。これらの評価尺度の計測は開発効率の定量化が目的である。更に、プログラムの正確性 (処理ロジックの誤りや、取得データの漏れなど) を確かめるために、出力結果となる JSON ファイルから、体重と写真の二つの取得データ数を計測する。実験の最後に LLAPI の利用に関するアンケートを実施する。

6.5 結果

開発実験の結果を表 3 に示す。取得ログ数の行に記載されている「正答」とは、正しいマッシュアップ処理により得られる正解の値のことを指し、“✓” は正答との一致を意味する。

まず総行数に着目する。5 人中 4 人の被験者が提案手法を利用することで少ない規模で開発を終えていた。平均では P_{conv} と比べて 0.72 倍であり、特に Perl プログラマ (被験者 A と B) は 0.46 倍と半分程度の規模で実装していた。また、Perl プログラマはファイル数も大きく減少していた。一方で Java プログラマにはファイル数の減少は見られなかった。

開発に要した工数に着目すると、実験の順序にかかわらず 5 人中全ての被験者で、 P_{prop} の方が開発時間が短い。平均での削減工数は約 1/3 であり、最大で約 1/4 の短縮化であった。特に P_{conv} で 8 時間以上かかった被験者 D は、 P_{prop} では 2 時間強と大きく減少していた。

プログラムの正確性を表す各ライフログデータの取得数を確認すると、全てのデータを正しく取得できたのは被験者 A と B の P_{prop} のみであった。しかしながら、被験者 A と B の P_{conv} 、及び他の被験者の両

表 3 TabetaLog 開発実験の結果
Table 3 Result of TabetaLog development experiment.

被験者名 開発言語 実験順序	A		B		C		D		E	
	Perl		Perl		Java		Java		Java	
	P_{prop}	先行	P_{conv}	先行	P_{prop}	P_{conv}	P_{prop}	P_{conv}	P_{prop}	先行
総行数	115	365	227	379	480	612	423	397	50	181
ファイル数	1	4	1	3	7	7	5	5	2	2
工数 (分)	114	196	54	205	96	252	147	514	132	397
取得体重ログ数 <shimojo> (正答 53)	✓ 53	54	✓ 53	54	32 ✓ 53	✓ 53	✓ 53	54	52	52
取得体重ログ数 <tokunaga> (正答 102)	✓ 102	101	✓ 102	101	52	103	103	104	103	115
取得写真ログ数 <shimojo> (正答 8)	✓ 8	9	✓ 8	9	✓ 8	9	✓ 8	9	✓ 8	9
取得写真ログ数 <tokunaga> (正答 85)	✓ 85	86	✓ 85	86	60	87	✓ 85	44	65	✓ 85

開発方式では一つ以上の誤りが含まれていた。

アンケートの結果から「マッシュアップ対象のサービスが増えた場合、 P_{conv} では更に多くの開発工数が必要だろう」、「(オフライン方式であるため) LLAPI では最新データの取得が難しい」、「Flickr では豊富な API が提供されているが、その分必要とする API の発見に時間を要した」などの LLAPI に対する意見が得られた。また LLCDM に対しては、「データの書式と構造が共通化されており、同じ処理ロジックを全てのライフログデータに適用できた」という意見が得られた。

7. 議 論

7.1 マッシュアップアプリケーション開発への効果

実験結果から LLCDM 及び LLAPI を利用することで効率的、かつ少ないコード行数でマッシュアップアプリケーションを開発できることが明らかになった。LLCDM により各種ログの書式と構造を統一化し、LLAPI によりデータ取得を一つの API 操作に集約できていた点が、開発工数の削減に繋がった要因であるといえる。本実験では二つのライフログサービスのマッシュアップを行ったが、更に多くのサービスをマッシュアップする場合、より提案手法の効果が現れると考えられる。この点について、三つのライフログ(移動履歴とつぶやき, 写真)を対象としたマッシュアップアプリケーションの開発実験を行った。結果、開発工数は 1/9 倍、総行数は 1/5 倍とそれぞれ大幅な削減が実現できた。実験の結果や詳細については [5] を参照にされたい。

次に TabetaLog 開発に対する LLCDM の効果について考察する。全てのデータを正しくマッシュアップ処理できていたのは、被験者 A と B の P_{prop} 方式のみであった。この 2 名の P_{conv} の誤りについては、各

表 4 LLCDM スキーマと他のライフログサービスの対応

Table 4 Feasibility of LLCDM to various lifelog services.

観点	What	Where	When	Who	How			
	<content>	<location>	<date>	<time>	<user>	<party>	<object>	<application>
Twitter [1]	つぶやき	○	○ ○	○ △ ○	○ ○ △			
Flickr [2]	写真	○	○ ○ ○	○ ○ ○	○ ○ △			
FoodLog [3]	食事履歴		○ ○ ○	○ ○ ○	○ ○ ○			
GARMIN [4]	運動記録	○	○ ○ ○	○ ○ ○	○ ○ △			
からだログ [9]	運動記録		○ ○ ○	○ ○ ○	○ ○ ○			
Life Pod [10]	行動履歴	○	○ ○ ○	○ △ ○	○ ○ △			
Life-X [11]	思い出	○	○ ○ ○	○ ○ ○	○ ○ ○			
Coolo [12]	機器操作	○	○ ○ ○	○ ○ ○	○ ○ ○			

ライフログデータのタイムゾーン変換処理の抜けが原因であった。 P_{prop} では各データの日付<date>が同じ世界協定時に統一されているが、 P_{conv} では Flickr は UTC +00:00, BodyLogService は UTC +09:00 で記録されており、タイムゾーンの変換が必要であった。LLCDM において各ライフログデータ固有の記述内容を標準化できていた点が、マッシュアップロジックの正確性向上に寄与していたといえる。

7.2 他のライフログサービスへの適用可能性

LLCDM が他のライフログサービスに依存しない、中立的なモデルであることを確かめるための補足的実験として、他のライフログサービスに対する LLCDM の適用可能性を評価する。LLCDM のデータスキーマと、8 種類の既存ライフログサービスの記録データの対応を表 4 に示す。“○”は各データスキーマに該当するデータを含む(あるいは含んでいると推測できる)ことを意味し、“△”は明確にデータ項目に存在はしていないものの、他のデータ項目から読み取れる可能性があることを意味する。例えば Twitter の場

合<user>と<object>は、それぞれ<screen_name>と<in_reply_to_screen_name>に対応する(図1(a)参照)。共に行動したユーザを意味する<party>はデータ中に明記されていないが、つぶやき内容に記録されたハッシュタグから取得できる可能性がある。

表より、ほとんどのライフログサービスが What, When, Who, Where, How のうちの一つ以上のスキーマに該当している。特に<date>や<user>は全てのライフログサービスが保持している。これらの結果から、LLCDMで規定したデータモデルは、ライフログサービスに依存しにくい中立なデータモデルであると考えることができる。

7.3 適用限界

本論文では、ラベル的な属性の一致を取る API: getLifeLog() と、時刻の類似を取る API: getLifeLogNearestTime() の二つについて紹介したが、これらの API で全てのデータ・マッシュアップに対応できるわけではない。まず、時刻や緯度経度といった定量的な属性に対して、類似や区間を取るための汎用 API が必要である。また、各ライフログの本質データとなる、What 項目の中身に基づいたマッシュアップや、セマンティックな情報を利用した広義な意味でのデータ・マッシュアップへの対応も限界点の一つである。

8. 関連研究

様々な種類のライフログデータを、自動的かつ統合的に収集する方法に関する研究が行われている [13]~[17]。MyLifeBits [14], [15] は、個人の生活に関するあらゆる情報を収集するシステムである。このシステムはライフログを蓄積するにとどまらず、データの意味解析やリンク構造の構築といった、蓄積データの検索・活用・管理のための機能をもつ。Aizawa [13] はウェアラブル技術を利用したライフログの計測システムを提案している。このシステムでは映像や音、脳波など様々なログを自動的に収集できるほか、抽象的なコンテキスト情報に基づいたログ検索機能をもつ。これらの研究は複数種類のライフログを計測・活用するシステムの提案であるが、本論文の提案方法は、ネット上に点在する異種のライフログを再利用しつつ、統合・集約する方法であるという点で異なる。

従来、あらゆる情報やデータの統合を目的として、オントロジーを用いた手法の研究が数多く実施されている [18]~[21]。Lee ら [21] は、ライフログデータを対象としたデータ統合システム LifeLogOn を提案

している。これらオントロジベースの手法は、オントロジ自体の構築が困難ではあるものの、データ表現の厳密さと柔軟さをもつ。我々の標準データモデルでは 5W1H という抽象的な観点から標準スキーマを規定したが、What に該当する<contents>を標準化するには、このようなオントロジベースの方法を利用できると考えられる。

9. むすび

本論文では、異なるライフログサービスのマッシュアップを支援するために、ライフログデータの標準データモデルと、標準データにアクセスするための汎用的な API を提案・実装した。実験では、体重ログと食事写真ログをマッシュアップするアプリケーション、TabetaLog を題材として評価を行った。実験の結果、提案手法を利用することで開発工数を約 1/3 に削減できること、またマッシュアップロジックの正確性を確保できることが確認できた。

今後の課題として、ライフログデータ共有の際に発生するプライバシーの問題に対処する必要がある。現在、LLCDM に蓄積されたデータにはアクセス制限が設けられていないため、ユーザ認証などの機能の追加が必要である。また、ライフログサービスの提供者とマッシュアップアプリケーションの提供者の間で、どのような収益モデルを構築するかといった、ビジネスモデルについても今後検討していきたい。

謝辞 この研究の一部は、科学技術研究費(基盤研究 B 23300009, 若手研究 B 21700077, 研究活動スタート支援 22800042), 及び、ひょうご科学技術協会の助成を受けて行われている。

文 献

- [1] “Twitter”, <http://twitter.com/>
- [2] “Flickr”, <http://www.flickr.com/>
- [3] K. Kitamura, T. Yamasaki, and K. Aizawa, “Food-Log: Capture, analysis and retrieval of personal food images via web,” Proc. Int’l Symposium on Applications and the Internet Workshops, pp.23–30, 2009.
- [4] “GARMIN Connect”, <http://connect.garmin.com/>
- [5] 鎌田早織, 坂本寛幸, 井垣 宏, 中村匡秀, “マッシュアップ API を用いた異なるライフログサービスの連携,” 信学技報, LOIS2009-94, 2010.
- [6] 下條 彰, 福田将之, 井垣 宏, 中村匡秀, “異なるライフログをマッシュアップするためのデータ変換・集約アクセス API の実装,” 信学技報, LOIS2009-93, 2010.
- [7] 下條 彰, 杉本真佑, 中村匡秀, “ライフログのためのマッシュアップ API の DB 実装と Web サービス化,” 信学技報, LOIS2010-43, 2010.

- [8] S. Mohan, E. Choi, and D. Min, "Conceptual modeling of enterprise application system using social networking and web 2.0 "social CRM system"," Proc. Int'l Conf. Convergence and Hybrid Information Technology, pp.237-244, 2008.
- [9] "goo からだログ", <http://karada.goo.ne.jp/>
- [10] A. Minamikawa, N. Kotsuka, M. Honjo, D. Morikawa, S. Nishiyama, and M. Ohashi, "RFID supplement for mobile-based life log system," Proc. Int'l Symposium on Applications and the Internet Workshops, vol.0, p.50, 2007.
- [11] "Life-x", <http://life-x.jp/>
- [12] M. Abe, Y. Morinishi, A. Maeda, M. Aoki, and H. Inagaki, "A life log collector integrated with a remote-controller for enabling user centric services," IEEE Trans. Consum. Electron., vol.55, no.1, pp.295-302, 2009.
- [13] K. Aizawa, "Digitizing personal experiences: Capture and retrieval of life log," Proc. Int'l Multimedia Modelling Conference, pp.10-15, 2005.
- [14] J. Gemmell, G. Bell, R. Lueder, S. Drucker, and C. Wong, "MyLifeBits: Fulfilling the memex vision," Proc. ACM Int'l Conf. Multimedia, pp.235-238, 2002.
- [15] J. Gemmell, G. Bell, and R. Lueder, "MyLifeBits: A personal database for everything," Commun. ACM, vol.49, pp.88-95, Jan. 2006.
- [16] I.J. Kim, S.C. Ahn, H. Ko, and H.G. Kim, "Automatic lifelog media annotation based on heterogeneous sensor fusion," Proc. Int'l Conf. Multisensor Fusion and Integration for Intelligent Systems, pp.703-708, 2008.
- [17] M. Memmel and A. Dengel, "Sharing contextualized attention metadata to support personalized information retrieval," Int'l Workshop on Contextualized Attention Metadata: Personalized Access to Digital Resources, pp.19-26, 2007.
- [18] H. Wache, T. Vogeles, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hubner, "Ontology-based integration of information - a survey of existing approaches," IJCAI Workshop Ontologies and Information Sharing, pp.108-117, 2001.
- [19] M. Lenzerini, "Data integration: A theoretical perspective," Proc. Symp. Principles of Database Systems, pp.233-246, 2002.
- [20] N.F. Noy, "Semantic integration: A survey of ontology-based approaches," SIGMOD Record, vol.33, pp.65-70, 2004.
- [21] S. Lee, G. Gong, I. Hwang, and S.G. Lee, "LifeLogOn: A practical lifelog system for building and exploiting lifelog ontology," Proc. Int'l Conf. Sensor Networks, Ubiquitous, and Trustworthy Computing, vol.0, pp.367-373, 2010.

(平成 23 年 7 月 4 日受付, 10 月 26 日再受付)



裕本 真佑 (正員)

平 18 京都産業大・理卒. 平 21 日本学術振興会特別研究員 (DC2). 平 22 奈良先端科学技術大学院大学博士後期課程了. 同年神戸大大学院システム情報学研究科特命助教. 博士 (工学). マイニングソフトウェアリポジトリの研究に従事.



下條 彰

平 22 神戸大・工・情報知能卒. 平 24 同大大学院博士前期課程了予定. ライフログのマッシュアップに関する研究に興味をもつ.



鎌田 早織

平 22 神戸大・工・情報知能卒. ライフログ及び Web サービスの研究に興味をもつ. P2P 技術を利用した Windows クライアントアプリケーションの開発及びテスト業務. M2M プラットホームにおけるマルチデバイスの管理サービス開発業務に従事.



中村 匡秀 (正員)

平 6 阪大・基礎工・情報卒. 平 11 同大大学院博士後期課程了. 平 12 同大サイバーメディアセンター助手. 平 14 奈良先端科学技術大学院大学情報科学研究科助手. 平 19 神戸大大学院工学研究科准教授. 平 22 同大大学院システム情報学研究科准教授. 博士 (工学). サービス・クラウドコンピューティング, ライフログ, ホームネットワーク, ソフトウェア工学の研究に従事. IEEE, ACM, 情報処理学会各会員.