Vol.2012-IS-119 No.9
2012/3/15

情報ポータル運営者のための軽量マッシュアップ作成ツールの提案と評価

平山雅樹[†1] 新野朝丈[†1]
松澤芳昭[†2] 太田剛[†2]

我々はかつて学会セミナ情報収集システムの開発を行い，複数の学協会の Web ページからイベント情報を収集するシステムを作成した．このシステムは学協会からイベント情報を抽出する規則をユーザが作成する必要があり，抽出規則の作成，管理に大きなコストを要した．そこで，本システムでは「エンドユーザ」が「情報ポータル」の作成，管理をすることに特化したツールの開発を目的とする．エンドユーザはプログラミングの知識を持たないユーザを想定し，情報ポータルは学会セミナ情報収集システムのように Web ページから特定の目的に適う情報を収集，集積するサイトを想定する．情報ポータルを作成できる技術としてマッシュアップがある．マッシュアップツールで有名な Karma では必要な要素技術を 5 つ挙げ実装している．我々は目的を達成するために 6 つ目の要素技術を提案し，それにもとづいて実装を行った．本システムをエンドユーザに使用してもらったところ，いずれも情報ポータルを短時間で作成することができた．

Proposal and Evaluation of Lightweight Mashup Tool to Create Information Portal

Masaki Hirayama,[†1] Tomotake Niino,[†1]
Yoshiaki Matsuzawa[†2] and Tsuyoshi Ohta[†2]

In recent years, PBD(Programming By Demonstration)-based mashup creation tools which can be used for ”end-user” with no programming skills has been proposed. In this paper, we proposed PBD-based mashup creation tool for creating ”information portal” which collect information from multiple web pages as end-user desire. There is a favorite tool named ”Karma” to create information portal. Karma needs five tasks to use. We developed tool to create information portal which needs only two tasks. Our experiment shows that end-user could create information portal easily according to his interest with the proposed system.

1. はじめに

インターネットの発展により，インターネットのユーザは膨大な量の Web ページにアクセスすることが可能になった．しかしながら，ユーザが求める情報は複数の Web ページに独自の形式で散在しており，ユーザはそれらを見て回るだけでも手間を要する．

そこで，近年では複数の Web ページの内容を集約，整理する行為自体にも価値が見出されている．実際に複数の Web ページの情報を収集するコンテンツ,「情報ポータル」は多数存在している．例えば google ニュース (http://news.google.co.jp/) では，複数の Web ページからニュースの記事を収集し，提供している．

情報ポータルが増加すれば，散在した情報が集約されて Web ページの巡回が容易になると考える．しかし，情報ポータルの作成には Web システム開発の知識が要求されるため，情報ポータルの数は少ない．Web システム開発の知識を持たないユーザ (以下，エンドユーザ) でも情報ポータルを作成することができれば，この問題を解決することができる．

本プロジェクトの目的は，エンドユーザに情報ポータルの開発を可能にすることである．

我々はかつて実施した情報ポータルの開発プロジェクトを分析し，エンドユーザ向けの情報ポータル作成システムに求められる要件を調査した．

本システムの要件を実現するための要素技術として，我々はマッシュアップに注目した．マッシュアップとは，複数の情報源から欲しいデータのみを取り出し，組み合わせることによって新たなデータを提供する機能のことである[9)]．

マッシュアップの目的は多数あり，それぞれの目的に応じたツールが多数提案されている．これらのツールの要素技術を組み合わせることにより，要件を満たすシステムを作ることができると考える．

2. 要求分析

我々は本システムの要求を得るために，かつて実施した学会セミナ情報収集システム[⋆1]を

†1 静岡大学大学院情報学研究科
Graduate School of Informatics, Shizuoka University
†2 静岡大学情報学部
Faculty of Informatics, Shizuoka University
⋆1 成果物はこちらよりアクセスできる：http://cat-engine.inf.shizuoka.ac.jp/test/cakephp/toppages

分析した．

2.1 学会セミナ情報収集システム概要

このシステムは，日本工学会に所属する複数の学協会のイベント情報ページより，セミナや講習会のタイトル，開催日時，開催場所を定期的に収集して提示するシステムである．複数の学協会のイベント情報ページよりセミナや講習会の情報を収集するので，典型的な「情報ポータル」の例といえる．学会セミナ情報収集システムは以下の機能を持つ．

機能 1 学協会のイベント情報を提供する者「提供者」は，システムがイベント情報を収集するために必要な情報「抽出規則」を提示することができる．

機能 2 システムは抽出規則を使って学協会のイベントカレンダーページをクローリングし，イベント情報を収集することができる．

機能 3 システムはイベント情報をデータベースに登録することができる．

機能 4 イベント情報を閲覧する者「閲覧者」はシステムの持つデータベースからイベント情報を閲覧，検索，ソートすることができる．

学会セミナ情報収集システムでは抽出規則を提供者が記述することが難しいため，予めシステム側が用意した抽出規則を使用する．

2.2 学会セミナ情報収集システムの問題

学協会の Web ページは毎日のように内容が変更されるため，既存の抽出規則ではセミナ報を抽出できなくなることがある．この時，提供者が自ら抽出規則を修正できることが望ましい．このことから，以下の要件が得られる．

要件 1 抽出規則の定義ファイルを作成から管理まで実施できること

予めシステム側で用意された抽出規則では，全てのセミナページから情報を抽出することはできない．また，セミナページの構造が変更されたときに抽出規則が適用できなくなる可能性がある．セミナ情報の提供者が自ら抽出規則を作成し，必要に応じて変更することができれば，この問題に対処できる．

要件 2 エンドユーザが本システムを使用することができること

本システムで対象とするユーザは学協会の管理者をはじめとするエンドユーザである．本システムはエンドユーザが持つ知識で使用することができなければならない．

要件 3 ユーザがサーバの管理を実施する必要が無いこと

情報ポータルを公開するためには Web サーバが必要となるが，エンドユーザにサーバ管理の負担を課すことはできない．

3. 先 行 研 究

本章では，情報ポータル作成に関係する先行研究について述べる．

3.1 マッシュアップ

Albinola の定義によると，マッシュアップとは複数の情報源から欲しい情報のみを取り出し，組み合わせることによって新たな情報を提供する機能のことである[9]．

Web ページから情報を収集することは本システムでも同様であり，本システムもマッシュアップの一部である．

マッシュアップを作成するためのツールは多数提案されており，Beemer は複数のマッシュアップツールのレビュー論文を発表されている[1]．Beemer によるとマッシュアップツールには 7 種類のカテゴリが存在する．我々はその中でも”End User Programming”のカテゴリに属するツールを調査した．

Marmite[16],Yahoo’s Pipes[21],Microsoft Popfly[4] はいずれも Widget 型のマッシュアップツールと呼ばれている．ユーザは Widget と呼ばれる，ブロック型のモジュールをブロック図のようにつなぎ合わせることによってプログラミングを実施することができる．Tuchinda は Widget を組み合わせることとプログラミングを書くことは基本的に同じであると指摘している[11]．そのため，エンドユーザが Widget 型のマッシュアップツールを使用することは難しい．

Marmite が最初に提案された Widget 型のツールであり，Yahoo’s Pipes は Widget 型のツールを Web サービスとして提供した．Microsoft Popfly は他の 2 つのツールに比べて Widget の種類が最も多く，また使用可能な Widget を推測し，ユーザに提示するという特徴がある．

Potluck[2]，Piggy Bank[3] は共に，Resource Description Framework(RDF) 形式の Web ページのみを対象としたマッシュアップツールである．RDF では Web ページの中にテキストと属性名がタグに示されており，システムはそのタグを使ってテキストを抽出する．そのため，ユーザは抽出規則を作成する必要が無い．しかし，これらのシステムでは RDF 形式ではないページからテキストを取り出すことができない．

Potluck では Potter’s Wheel[15] を用い，RDF ファイルから取り出したテキストを自動的に編集する規則を作成することができる．

Dapper[18]，Karma[12]，Vegemite[6] はいずれも，PBD によって抽出規則を作成する．ユーザが Web ページから取り出したいテキストや画像を選択すると，システムはその抽出規

則を生成する．Dapper は Web サービスとして誰でも無料で使うことができ，出力形式を RDF，表形式，Web ページ形式で選択できるという特徴がある．Karma では抽出規則の作成だけでなく，データベースの設計も PBD で実施するという特徴があり，Vegemite は PBD によって生成された抽出規則をスクリプトで出力し，ユーザが自由に編集できるという特徴がある．PBD ではユーザが抽出規則作成のためにプログラミングを実施する必要が無いため，本システムの目的に適している．

iGoogle[19] は，ユーザ自身がオリジナルの Web ページを作成するための Web サービスである．iGoogle では Gadget と呼ばれているマッシュアップを複数提供しており，ユーザは自分の目的にあった Gadget を選択し，自身の Web ページに配置する．Gadget の作成を作成するためにはプログラミングを行う必要がある．

マッシュアップを Widget や PBD によって作成するのではなく，直接プログラミングするための補助ツールも提案されている[7),8),13)]．これらのツールは当然ながらユーザがプログラミングを行う必要がある．

3.2 マッシュアップフレームワーク

Tuchinda はマッシュアップツールで実施する作業工程として，5 つのものを定義した．すなわち，Data Retrieval(抽出規則の定義)，Source Modeling(データベーススキーマの定義)，Data Cleaning(Web ページから取り出したテキストの編集規則の定義)，Data Integration(複数のデータベーステーブルの結合方法の定義)，Data Display(データベースに保管されているデータの表示方法の定義) の 5 つである[11)]．

しかし，これらの作業項目の中には一度作成した抽出規則を修正する工程が含まれていない．そこで，本稿では 6 つ目の作業項目として Rule Fixing を定義する．

3.1 節で述べたツールの特性を**表 1** にまとめる．Tuchinda も同様の表をまとめられているが，我々はプログラミングの知識が必要か，必要でないかの観点から特性を再定義している．

Data Retrieval について，PBD は欲しいテキストを範囲選択することによって抽出規則を作成できるもの，RDF は抽出規則の定義を実施せず，RDF ファイルのタグを抽出規則として用いるものである．Programming は，抽出規則の作成にプログラミングの知識を要するものである．

本稿では Widget を含め，プログラミングが必要な工程を全て Programming とする．Tuchinda はプログラミングが必要なものについて更に細かく定義していたが，本稿ではそれらの違いを重視しない．

表 1 マッシュアップツールの特徴

ツール名	Data Retrieval	Source Modeling	Data Cleaning	Data Integration	Data Display	Rule Fixing
Marmite[16)]	Programming	Manual	Manual	Manual	Web Page,RDF	Remake
Yahoo's Pipes[21)]	Programming	Manual	Manual	Union	Web Page,RDF	Remake
Microsoft Popfly[4)]	Programming	Manual	Manual	Manual	Web Page,RDF	Remake
Piggybank[3)]	RDF	Manual	N/A	Manual	Web Page	Remake
Potluck[2)]	RDF	Manual	PBD	Manual	Web Page	Remake
Dapper[18)]	PBD	Manual	Manual	Manual	RDF,Table	Remake
Karma[12)]	PBD	Database	PBD	PBD	Table	Remake
Vegemite[6)]	PBD	Manual	N/A	Union	Table	Script
iGoogle[19)]	Programming	Manual	Manual	Manual	Web Page	Remake

Source Modeling では，ユーザが属性名を 1 つずつ入力してスキーマを定義するものを Manual とした．Karma も原則として Manual であるが，システムはユーザが過去に作成したテーブルを自動的に探索し，スキーマを推測することができる．

Data Cleaning では，ユーザがテキストを編集する規則を直接記述するものを Manual，Potter's Wheel を用いて編集規則を作成するものを PBD，Cleaning を実施しないものを N/A としている．

Data Integration では，ユーザが結合方法を提示するものを Manual，PBD によって結合方法を定義できるものを PBD としている．また，Union は Integration の中でも Join を実施しない代わりに自動的に Union を実施できるものを指す．

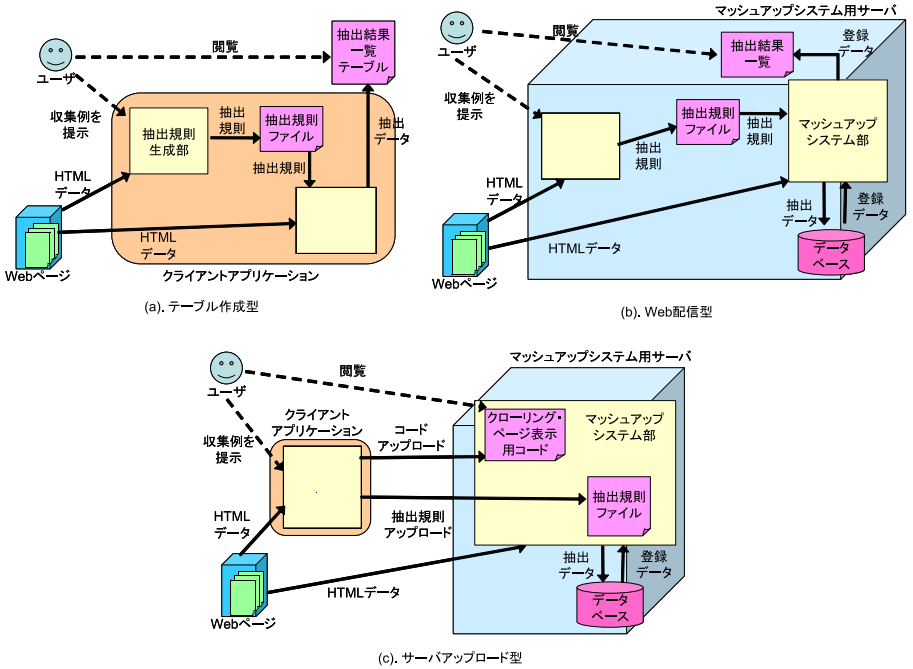
Data Display では，収集したテキストが Web ページとして提示されるものを Web Page，RDF ファイルとして出力されるものを RDF，表としてローカルに保存されるものを Table としている．ユーザが結合方法を提示するものを Manual，PBD によって結合方法を定義できるものを PBD としている．

Rule Fixing では，抽出規則を作りなおす時に Data Retrieval の再実行が必要となるものを Remake，スクリプトを編集することで作りなおすことができるものを Script としている．

4. システム設計

4.1 アーキテクチャ

本節では，情報ポータル作成システムのアーキテクチャとして，3 つのアーキテクチャを考え，本システムで採用するアーキテクチャを検討する．



(a). テーブル作成型　(b). Web配信型　(c). サーバアップロード型

図 1　検討するアーキテクチャ

4.1.1　テーブル作成型アーキテクチャ

テーブル生成型アーキテクチャを図 1 の (a) に示す．このアーキテクチャは，抽出規則生成部とマッシュアップシステム部で構成されている．抽出規則生成部では，使用者の指示に基づき，抽出規則を生成する機能を持っている．マッシュアップシステム部では，生成された抽出規則を基に，Web ページからテキストの抽出を行う．抽出したテキストは，テーブルの形にまとめられ，使用者へと提示される．このアーキテクチャをもつツールとしては，Karma[12] が挙げられる．

4.1.2　Web 配信型アーキテクチャ

Web 配信型アーキテクチャを図 1 の (b) に示す．このアーキテクチャも，抽出規則生成部とマッシュアップシステム部で構成されている．抽出規則生成部では，使用者の指示に基づき，抽出規則を生成する機能を持っている．マッシュアップシステム部では，定期的に Web ページを巡回し，Web ページごとに生成された抽出規則を基に，Web ページからテキストの抽出を行い，抽出したテキストをデータベースへと保存する．抽出したテキストは，フィードとして配信する，あるいは Web ページとして表示することで，使用者へと提示される．このアーキテクチャを持つツールとしては，Dapper[18] が挙げられる．

4.1.3　サーバアップロード型アーキテクチャ

サーバアップロード型アーキテクチャを図 1 の (c) に示す．このアーキテクチャは，抽出規則・コード生成部とマッシュアップシステム部で構成されている．抽出規則・コード生成部は，クライアントアプリケーションとなっており，使用者の指示に基づき，抽出規則を生成すると共に，マッシュアップシステム部において，クローリングおよび抽出結果の提示に使用されるソースコードを生成し，マッシュアップシステム用サーバへとアップロードする．マッシュアップシステム部では，アップロードされたソースコードを使用して，定期的に Web ページを巡回して，Web ページごとに生成された抽出規則を基に，Web ページからテキストの抽出を行い，抽出したテキストをデータベースへと保存する．抽出したテキストは，アップロードされたソースコードのレイアウトの形で Web ページとして使用者へと提示される．また，抽出結果を提示する際のレイアウトを使用者の手により改良することが可能である．

4.1.4　アーキテクチャの検討

3 つのアーキテクチャについて検討した結果を表 2 に示す．Web 配信型とサーバアップロード型は，サーバ上にマッシュアップシステムを配置し，使用者への提示を行なっているため，Web 上での共有が可能であり，またクローリングを実施している．一方，テーブル作成型は，個人使用を目的としているため，Web 上にてマッシュアップシステムを公開する機能は備えておらず，クローリングも行なっていない．

即興性については，テーブル型は，クライアントアプリケーション上で即座に規則の作成からテキスト抽出結果の提示までを行うことができる．サーバアップロード型も，クライアントアプリケーション上で抽出規則の作成を行うため，即時に規則を作成することができ，即興性に優れていると考える．

レイアウトの変更は，サーバアップロード型のみが実施することができ，テーブル作成型，Web 配信型では，提示の方法をいくつかの選択肢の中から選択することはできるが，使用者自身が変更することはできない．システムの複雑さについては，サーバアップロード型は，クライアントアプリケーションとサーバ上のシステムで構成されるという構造上，他の二つのアーキテクチャよりも，システムが複雑なものになってしまう．

表 2 アーキテクチャの検討

検討事項	テーブル作成型	Web 配信型	サーバアップロード型
Web 公開	×	○	○
クローリング	×	○	○
即興性	○	×	○
レイアウトの変更	×	×	○
システムの複雑さ	○	○	×

”○”：検討事項を満たす　”×”：検討事項を満たさない

表 3 ツールの特徴と要件

作業工程	特徴	要件 1	要件 2	要件 3
Data Retrieval	Programming	○	×	−
	PBD	×	○	−
	RDF	×	○	−
Source Modelng	**Manual**	○	○	−
	Database	○	○	−
Data Cleaning	Manual	○	×	−
	PBD	×	○	−
	N/A	×	○	−
Data Integration	Manual	○	×	−
	Union	○	○	−
	PBD	○	○	−
Data Display	**Web Page**	○	−	○
	RDF	×	−	○
	Table	×	−	×
Rule Fixing	Remake	×	○	−
	Script	○	×	−

”○”：検討事項を満たす　”×”：検討事項を満たさない　”-”：要件と関連性なし

サーバアップロード型は，システムが他の二つのアーキテクチャに比べて複雑になってしまうが，それ以外の事項をすべて満たすことができる．本システムでは，サーバアップロード型アーキテクチャを採用する．

4.2 要素技術の組み合わせ

本システムの設計においては，各作業工程毎に利用できる要素技術を吟味した．次に，要件と照らし合わせて費用対効果の高い組み合わせを考える．

4.2.1 Data Retrieval

本システムでは，Data Retrieval の実装方針として PBD を選択した．Manual ではプログラミングの専門知識がなければ実施できず，RDF では様々な抽出規則を作成できないからである．

しかし，PBD による抽出規則の作成にも問題はある．PBD では，抽出規則として XPath を使用し，XPath に基づいてテキストを抽出する．しかし，XPath よって抽出されるテキストは，HTML タグの中に存在するテキスト全てであり，テキストの一部だけを取り出したい場合に，対応することができない．タグの中のテキストの一部を切り出して抽出することもあるため，タグの中のテキストの一部を切り出すアルゴリズムが必要となる．

これを実現するための方法として，XPath にテキストマッチングを加えた複合型の Data Retrieval アルゴリズムを使用する．抽出規則を生成する際に，抽出対象のテキストの前後にあるテキストを切り出しパターンとして記録しておき，XPath によりテキストを抽出した後に，切り出しパターンとのマッチングを行う．切り出しパターンがマッチした場合には，その位置でテキストを切り出すことで，タグの中の一部のテキストを取得することができる．

4.2.2 Source Modeling

本システムでは，Source Modeling の実装方針として Manual を選択した．Modeling の Database では，一度作ったスキーマと同じものを作ろうとした場合には効果を発揮するが，それ以外の場合では Manual よりも遅くなってしまうからである[12)]．

4.2.3 Data Cleaning

本システムでは，Data Cleaning の実装方針として N/A を選択した．Manual ではプログラミングの専門知識がなければ実施できないからである．

PBD では一見，全てを満たすことができそうに見える．しかし，Data Cleaning の PBD 手法である Potter’s Wheel は英語を前提とした技術であり，日本語に対応した技術はない．

そこで，本システムでは，日付，時間，数値，文字列という 4 つの型に限定し，それぞれの型ごとに，変換形式を事前に定義する事前定義型アルゴリズムを開発した．この変換方式では，ユーザの任意の形式に変更することはできない．しかしながら，ポンシステムの要件内では，Data Cleaning は，ソートを行うためのみに要求されているものであり，可用性は担保される．

4.2.4 Data Integration

本システムでは，Data Integration の実装方針として Union を選択する．Union が一番ユーザに時間的負担を要求しないからである．この場合，情報の Join が実施できないが，これは要件に含まれていない．

4.2.5 Data Display

Data Display の実現方法として，CakePHP による Web ページの表示を使用する．検索

やソートを行うために，サーバ上のデータベースへ抽出されたテキストを登録して，アクセスに応じて抽出結果を表示する Web ページ型が適していると考える．さらに，Web ページのレイアウトは CakePHP によって実現されるため，使用者自身が CakePHP のレイアウトを変更することで，Web ページのレイアウトの変更や機能の追加が可能となる．

4.2.6 Rule Fixing

Remake では，抽出規則を一から作り直さなければならず，Script では，プログラミングの知識を用いずに実施することはできない．

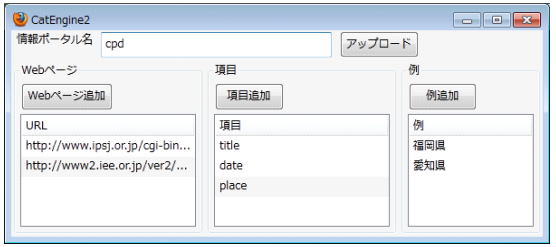
本システムでは，抽出規則を一から作り直す必要性を減らすために，抽出規則の再生成を自動化する．手順としては，構造が変化した Web ページに対して，サーバに残っているテキストによりテキストマッチングを行う．これにより，使用者が Web ページ上で抽出したいと考えている箇所を特定することができる．抽出したいと考えるであろう箇所が特定できれば，その後の処理は抽出規則生成時と同様であるため，抽出規則の再生成が可能となる．

4.3 インタフェース

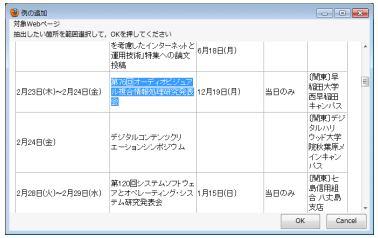
図 2 に本システムのインタフェースを示す．図 2 の (a) は抽出規則管理ウィンドウ，(b) は例追加ウィンドウ，(c) は情報ポータルのインタフェースである．本システムを起動すると，まずクライアントアプリケーションのメインのインタフェースである抽出規則管理ウィンドウが表示される．抽出規則管理ウィンドウにおいて,「Web ページ追加」ボタンを押下することで，現在ブラウザ上に開いている Web ページが抽出対象ページとして登録される．

次に,「項目追加」ボタンを押下することで，Web ページから抽出する項目の登録を行うことができる．項目の登録時に，項目の型の登録も行う．Web ページと項目の登録が完了したら，例の登録を行う．例の登録には，まず,「例追加」ボタンを押下し，図 2 の (b) に示す例追加ウィンドウを表示させる，例追加ウィンドウ上で，例として登録したいテキストを選択し,「OK」ボタンを押下することで，例の登録を行うことができる．例を登録した後は，URL，項目，例のそれぞれのリストにおいて，右クリックメニューから，その段階での抽出規則で抽出することができるテキストのプレビューを表示することができる．

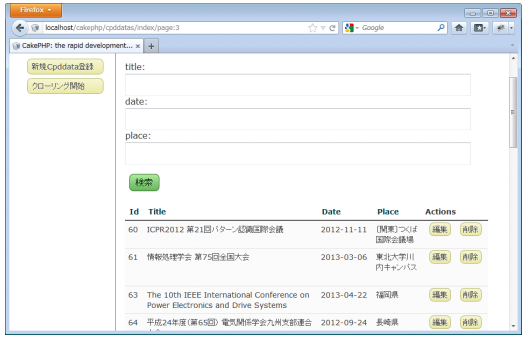
情報ポータルとして必要な Web ページ，項目，例の登録が完了したら,「情報ポータル名」に情報ポータルの名前を入力し,「アップロードボタン」を押下することで，サーバに抽出規則と，クローリングおよび抽出結果の提示に使用されるソースコードをアップロードすることができる．情報ポータルのインタフェースを図 2 の (c) に示す．情報処理学会 IPSJ カレンダーと電気学会の学会イベントカレンダーから，イベント名称と日付，開催場所を抽出した情報ポータルである．



(a). 抽出規則管理ウィンドウ



(b). 例追加ウィンドウ



(c). 情報ポータルのインタフェース

図 2 インターフェース

表 4 課題の実施時間

被験者 1	被験者 2	被験者 3	被験者 4
3 分 42 秒	7 分 08 秒	8 分 37 秒	9 分 07 秒

5. 評価実験

本章では，システム評価実験について述べる．

5.1 被験者による情報ポータル作成実験

5.1.1 実験方法

被験者実験では，被験者を招き，本システムを使って情報ポータルの課題を実施してもらった．これにより要件 2 が満たされているかをテストする．被験者は静岡大学の学生 4 名であり，2 名がプログラミングを履修したばかりである学部 1 年，2 名は高いプログラミングの知識を持つ研究室所属者である．実験では最初に 12 分の動画の動画によって覚えてもらい，本システムの使用方法を覚えてもらった．次に，制限時間 20 分の課題を解いてもらった．

実施する課題は学会セミナ情報収集システムを簡略化したものである．学会セミナ情報収集システムでセミナ情報提供学協会の中からリスト形式のものと表形式のものを 1 つずつ選択し，それらからセミナ情報を収集する．

5.1.2 実験結果

被験者ごとの課題の完了時間を**表 4** に示す．いずれの 10 分以内に課題を終えることができている．この点から，全員が情報ポータルを作成することができたと言える．このことから本システムはエンドユーザが使用可能であることが分かる．

5.2 既存システムとの比較実験

5.2.1 実験方法

既存システムとの比較実験では，本システムと既存システムの作業時間を比較することにより，本システムが技術選択によって軽量化されていることを検証する．比較の対象としては Karma を選択した．

実験方法としては，Karma で実施されている課題を被験者に実施してもらい，時間を比較する．被験者は 5.1.1 節と同一である．Karma での被験者はプログラマ 20 名 (情報系研究室の修士，博士 20 名)，非プログラマ 3 名 (学部生，事務員 3 名) である．

実施する課題は Karma の Task2 と同一の内容であり，ロサンゼルスの物件情報を提供する Web ページから，2 通りの検索結果を収集するというものである[⋆1]．Karma では時間の測定にあたって Retrieval，Modeling，Cleaning，Integration で実際にユーザがインターフェースを操作していた時間のみを測定したため，この実験でも同様の測定方法を取る．

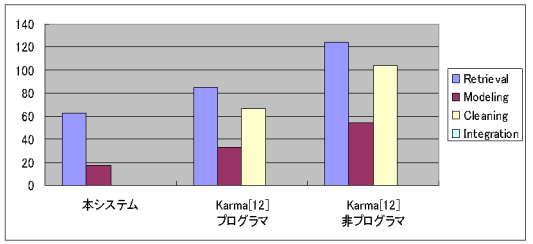


図 3 Karma との比較結果

5.2.2 実験結果

Karma との比較結果を**図 3** に示す．作業時間は全体的に本システムの方が短く，特に Data Cleaning は本システムでは Cleaning の所要時間が 0 となっており，Cleaning が自動化されていることが分かる．このことから，本システムでは要素技術の選択によってユーザの作業が軽量化されていることが分かる．

5.3 実システム作成実験

5.3.1 実験方法

実システム作成実験では，4.2 節で示したアルゴリズムの有用性を検証した．実験では情報ポータルに関するテーマを設け，本システムでそのテーマを実施できるかを調査した．実施したテーマを次に示す．

テーマ 1　放射線量収集ページ

放射線量の測定値をリアルタイムで提示する Web ページ 2 箇所より，測定値，測定時間，測定量を取得する．

テーマ 2　ソフトウェア更新情報収集ページ

フリーソフトの更新情報を提示する Web ページ 4 箇所より，最新の更新履歴を収集する．

テーマ 3　セミナー情報収集システムの再現

⋆1 http://losangeles.craigslist.org/search/

表 5 チーム内実験の収集可能情報数

テーマ名	記載情報数	取得可能情報数	取得不可能情報数
テーマ 1	26	26	0
テーマ 2	4	4	0
テーマ 3	469	469	0

セミナー情報収集システムの開発で用いた Web ページ 7 箇所より，セミナの日程，開催場所，セミナタイトルを収集する．

これらのテーマに対して，必要な情報が収集できるか否かを調査する．

5.3.2 実験結果

テーマごとに，収集が必要な情報の数と，実際に収集できた情報の数を**表 5** に示す．いずれも必要としていた情報を全て収集することができた．

6. 考　　察

本稿ではエンドユーザ向けの情報ポータル作成システムとして 3 つの要件を提案した．

要件 1　抽出規則の定義ファイルを作成から管理まで実施できること

要件 2　エンドユーザが本システムを使用することができること

要件 3　ユーザがサーバの管理を実施する必要が無いこと

事前の知識調査で被験者 2,3,4 は Web システムの開発の経験が無いことが分かったことから，この 3 名はエンドユーザであると言える．被験者 2,3,4 を含めた全員の被験者が情報ポータル作成実験で実施することができた．このことから要件 1,2 が満たされていることが分かる．

要件 3 については実験は実施していないが，本システムはサーバが抽出規則に記載されている情報の収集を自動で実施する．そのため，要件 3 が満たされていると言える．

以上より要件 1〜3 が満たされており，本システムはエンドユーザが情報ポータルを作成，管理するという目的が達成されていることが分かる．

本システムでは表形式，リスト形式の Web ページに対象を絞り，更に Cleaning を 4 パターン，Integration を Union に限定した．それでも，実システム作成実験では 3 つのテーマをひと通り実施することができた．このことから，本システムは学会セミナ情報収集システムの再現以外にも，ユーザが決めたテーマに沿って情報ポータルを作成できることが分かる．

7. ま と め

本稿では，情報ポータル作成システムについて，我々が以前に開発した学会セミナ情報収集システムを振り返ることにより，以下の 3 つの要件を提示した．

要件 1　抽出規則の定義ファイルを作成から管理まで実施できること

要件 2　エンドユーザが本システムを使用することができること

要件 3　ユーザがサーバの管理を実施する必要が無いこと

そして，要件を実現するために，クローリングと抽出規則の自動変更という新しい工程を追加することで，Karma のフレームワークを拡張し，拡張フレームワークをもとに改めて先行研究の分類を実施した．我々は 3 つの要件を満たす新しいアーキテクチャとして，サーバアップロード型アーキテクチャを提案した．サーバアップロード型アーキテクチャは，抽出規則と共に，マッシュアップシステム部において，クローリングおよび抽出結果の提示に使用されるソースコードをクライアントアプリケーションにおいて生成して，サーバ上へとアップロードし，サーバ上でマッシュアップシステムを動作させるものである．

評価実験の結果，本システムを用いることによってエンドユーザは情報ポータルを作成，管理することができるという結論になった．このことから，本システムを用いることによって学会セミナ情報収集システムの管理者はベンダーに頼らずに自らシステムを保守することができる．また，これに類似したシステムについて，自らシステムを構築して管理することができる．

参 考 文 献

1) Brandon Beemer, Dawn Gregg : Mashups: A Literature Review and Classification Framework, Future Internet, Vol.1, Issue 1, pp.59-87 (2009).
2) David F. Huynh, Robert C. Miller and David R. Karger : Potluck : Data Mash-up Tool for Casual Users, 16th International Conference on World Wide Web, pp.737-746 (2007).
3) David Huynh, Stefano Mazzocchi, David Karger : Piggy Bank: Experience the Semantic Web Inside Your Web Browser, 4th International Semantic Web Conference, pp.413-430 (2005).
4) Eric Griffin : Foundations of Popfly: Rapid Mashup Development, Apress(2008).
5) James Clark ,Steve DeRose : XML Path Language(XPath) version 1.0 w3c recommendation, Technical Report REC-xpath-19991116, World Wide Web Consortium(1999).

6) James Lin,Jeffrey Wong,Jeffrey Nichols,Allen Cypher,Tessa A. Lau : End-User Programming of Mashups with Vegemite, In Proc. IUI, pp.97-106(2009).
7) Joel Brandt, Scott R. Klemmer : Lash-Ups: A Toolkit for Location-Aware Mash-Ups, Proceedings of the 19th annual ACM Symposium on User Interface Software and Technology, pp.79-80 (2006).
8) Junichi Tatemura, Arsany Sawires, Oliver Po, Songting Chen, K. Selcuk Candan, Divyakant Agrawal, Maria Goveas : Mashup Feeds : Continuous Queries over Web Services, ACM SIGMOD, pp.1128-1120 (2007).
9) Matteo Albinola, Luciano Baresi, Matteo Carcano, and Sam Guinea : Mashlight: a Lightweight Mashup Framework for Everyone, 18th International World Wide Web Conference, pp.10-49 (2009).
10) Nevill-Manning, C. G. : Programming by demonstration, New Zealand Journal of Computing 4(2):pp.15-24.(1993).
11) Rattapoom Tuchinda, Pedro Szekely, and Craig A. Knoblock : Building Mashups By Example, 13th International Conference on Intelligent User Interfaces, pp.139-148 (2008).
12) Rattapoom Tuchinda,Craig A. Knoblock,and Pedro Szekely, : Building Mashups by Demonstration, ACM Transactions on the Web (TWEB) (2011).
13) Rob Ennals, David Gay : User-Friendly Functional Programming for Web Mashups, 12th ACM SIGPLAN International Conference on Functional Programming, pp.223-234 (2007).
14) Venkatesh, Viswanath, Bala, and Hillol : Technology Acceptance Model 3 and a Research Agenda on Interventions, Decision Sciences, vol. 39, no. 2, pp. 273-315, (2008).
15) V. Raman and J. M. Hellerstein : Potter's wheel: An interactive data cleaning system, In VLDB Journal,pp.381-390(2001).
16) Wong, J. and J. Hong : Making mashups with Marmite: towards end-user programming for the web, In Proc.CHI 2007. ACM Press, pp.1435-1444(2007).
17) 佐々木俊尚：キュレーションの時代 「つながり」の情報革命が始まる, ちくま新書 (2011).
18) Dapper: The Data Mapper (online) , available from (`http://open.dapper.net/`), (accessed 2011-10-24).
19) iGoogle (online), available from (`http://www.google.co.jp/ig`) (accessed 2011-10-27).
20) Microsoft's Popfly: Mashup creation for the masses(online), available from (`http://news.cnet.com/8301-17939_109-9720583-2.html`), (accessed 2012-2-14).
21) Yahoo pipes (online), available from (`http://pipes.yahoo.com/pipes/`) (accessed 2011-10-27).