Vol.2011-IS-118 No.7
2011/12/5

エンドユーザ向け 情報ポータル作成システムの提案

新野朝丈[†1] 平山雅樹[†1]
松澤芳昭[†2] 太田剛[†2]

本稿では，プログラミングの知識を持たない「エンドユーザ」が，複数の Web ページから特定の目的に適う情報を収集し，新たな価値を提供するサイト「情報ポータル」を作成するためのシステムを提案する．この「情報ポータル」を作成するための手段として，ウェブマッシュアップと呼ばれる技術がある．しかし，様々な要素技術が提案されており，それらが実現できる機能には差がある．そこで，本研究では実際に「情報ポータル」開発プロジェクトを分析することでシステムに要求される要件を整理し，そのすべてを実現した．本システムのエンドユーザへの適合可能性を現在評価中である．

Proposal of System to Create "Information Portal" for End-user

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

In this paper, we propose the system to make "Information Portal" which provides integrated valuable information gathered from multi web pages. This system is also for "end-user" who has no knowledge of programming. Web mashup is one of the technologies to create "Information Portal". But each system of web mashup differs in abilities to create it. We analyzed a project to create "Information Portal", defined requirements of "Information Portal" creating system, and implemented a system which meets the requirement. We are evaluating the proposed system to be able to create "Information Portal" by "end-user".

1. はじめに

インターネットの発達に伴って多くの情報が氾濫するようになり，インターネットの利用者が特定の目的に適う情報を収集するための時間と作業負担が増えてきた．例えば，ニュースを収集する場合には，MSN 産経ニュース (http://sankei.jp.msn.com/) や日本経済新聞 (http://www.nikkei.com/) など，多数の Web ページが存在し，一つ一つのサイトを毎日チェックするためには手間が掛かる．

この対策として，複数の Web ページを 1 つのページに集約した「情報ポータル」が登場した．例えば google ニュース (http://news.google.co.jp/) が挙げられる．一つの情報ポータルを利用するだけで，多くのニュースページに記載されているニュースを知ることができる．

ウェブマッシュアップと呼ばれる技術を用いて，情報ポータルを作成することが可能である．Albinola[4] によると，ウェブマッシュアップとは，複数の Web ページ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創りだすための技術である．

ウェブマッシュアップを実施するためのフレームワークは多数提案されているが，その多くはプログラミングの専門的知識が無ければ使うことができない．Web ページを閲覧する者の多くはプログラミングの専門的知識を持たないため，多くのウェブマッシュアップフレームワークを使うことができない．

逆に，プログラミングの専門的知識を必要としないフレームワークでは，機能に制限がある．プログラミングによらず情報ポータルを作るためには，プログラミングを必要としないウェブマッシュアップ技術を組み合わせることによって，情報ポータルを作るために必要な機能を全てフレームワークの仕様者が実装する必要がある．

そこで，本稿ではプログラミングの専門的知識を持たない者 (以下,「エンドユーザ」と表記) 向けに，情報ポータルを作るために必要な機能を考察し，その機能を実現するためのウェブマッシュアップフレームワークを提案する．

†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

2. 要件の抽出

情報ポータル開発に必要な機能を考察するため，学会セミナ情報システム[12)] を分析した．

2.1 学会セミナ情報収集システム

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．

2.2 要　　件

学会セミナ情報収集システムの開発が完了した後に，情報ポータルにどのような機能がなければいけないのかという観点から見直しを行った．抽出された 6 つの要件をまとめて「情報ポータル要件」と呼び，以下に述べる．

要件 1: 抽出すべき情報を指定する作業負担を減らしたい

ウェブマッシュアップでは，取得したい情報を Web ページから抽出するために，抽出規則を作成する．学会セミナ情報収集システムを例にとるならば，XPath によって抽出規則を記述している．抽出規則を手作業で記述するためには，当然抽出規則の形式についての知識を持っている必要がある．XPath の記述についても，抽出したい文字列までの HTML タグのパスを，学会セミナ情報収集システムの使用者自ら調べなければならない．エンドユーザにこのような作業負担を課すことはできない．

要件 2: 抽出情報を検索，ソートしたい

タイトル，開催日時，開催場所のように，Web ページから取り出した文字列を「抽出情報」と呼ぶ．　複数の学協会からセミナや講習会の情報を集めると，その数は 300 を超える．この中からエンドユーザが自分の閲覧したい情報を見つけるためには，一覧表として表示される抽出情報を絞り込むことが有用である．学会セミナ情報収集システムでは，絞り込む方法として検索とソートを提供した．検索によって，特定の言葉を含むセミナ情報を探すことができ，ソートによってセミナを開催日時順に見ることが可能である．多くの Web ページで検索，ソートの機能を提供しており，情報ポータルでもこの機能が必要になると考える．

要件 3: 新しい情報をシステムが自動的に追加して欲しい

学協会の Web ページに新しいセミナ情報が追加された場合，そのセミナ情報は学会セミナ情報収集システムにも自動追加して欲しい．この要件が満たされないと，エンドユーザは新しい情報が追加されたかどうかを調べるために学協会の Web ページを巡回することになり，本来の目的である巡回の手間の軽減を実現できない．

要件 4: 情報ポータルを他の人と共有したい

特定の目的のために Web ページを統合した情報ポータルは，統合作業をしたエンドユーザ本人だけでなく，同じ目的を持つ他の者にとっても有用である．そこで，情報ポータルを他の者も閲覧できるようにすると有用である．

要件 5: 抽出した文字列を部分的に切り出したい

例として，セミナの開催日時が，ある学協会では「10 月 1 日」と記載され，別の学協会では「10/1」と記載されていた場合を考える．どちらも同じ日付を指しているが，その表記が異なる．これらの抽出情報に対してソートを実施すると，日付順には並ばないため，具合が悪い．このとき，文字列の中から，ソートに使いたい部分だけを抽出することができれば，この問題は解消される．日付の例であれば，文字列から「10」と「1」だけを抽出することがそれにあたる．

同様に，開催場所についても，ある学協会では「浜松市」と抽出され，別の学協会では「静岡県浜松市」と抽出される場合がある．もしも情報ポータルの閲覧者が市だけでソートをしたいと考えている場合,「静岡県浜松市」から「浜松市」だけを取り出したいと考えるはずである．

よって，文字列の抽出を行う際には，取り出した文字列から一部を切り出して利用する機能が必要になると考える．

要件 6: 抽出規則をシステムが自動的に修正して欲しい

学協会の Web ページの HTML タグの構造が変化した場合，それに伴って XPath も変更する必要がある．学会セミナ情報収集システム開発時に，15 箇所の学協会 Web ページからセミナ，講習会の情報の収集を試みたところ，1ヶ月に 2 回以上の頻度で，どこかしらの HTML タグの構造が変化した．HTML タグの構造が変わるたびに，エンドユーザは XPath を作り直す必要があり，不便である．

3. 先 行 研 究

本章では，Beemer[1)] により既存のウェブマッシュアップフレームワークを紹介すると共に，2 章にて述べた各要件を満たしているかを整理する．

Rattapoom[5)] によると，抽出情報を取り出す方法には，DOM（Document Object Model）型と Widget 型に分かれる．DOM 型では，HTML（HyperText Markup Language）の

DOM ツリーを XPath によって辿り，抽出情報を取り出す．この方法では，取り出せる抽出情報は，文字列に限定される．Widget 型では，抽出情報の取り出しにアプリケーションを用いる．

Widget 型のフレームワークの例としては，iGoogle[15] がある．iGoogle は，プログラマがウィジェットを作成し，エンドユーザは提供されているウィジェットの中から自分の目的に合ったものを使用する．

Yahoo Pipes[16] をはじめとした 5 つのフレームワーク[4),9),11),13)] では，プログラムのソースコードを書くこと無く，ウィジェットを作成する方法を提案している．これらの手法は，ブロック図の編集や表の編集によって，抽出情報の取り出し方と，取り出した抽出情報の配置方法を指定する Widget 型のフレームワークを提案している．しかし，Rattapoom[5] は，ブロック図や表を編集するためにもプログラミングの詳細知識は必要であると指摘している．

ウィジェットのプログラミングを補助するためのフレームワークも 3 つほど提案されている[6)–8)]．しかし，これらのフレームワークでも，プログラミングの知識が必要である．

Widget 型のウェブマッシュアップでは，目的とする機能をもつウィジェットが提供されていない場合，フレームワークの使用者は自らの手でウィジェットを作成しなければならず，2 章の要件 1 を満たさない．そこで，本稿では DOM 型のフレームワークを対象として，システムを提案する．DOM 型のフレームワークを**表 1** にまとめる．

表 1 DOM 型のウェブマッシュアップフレームワークの比較

ツール名	対象形式	2.2 節の要件を満たすか						フレームワーク使用者に要求される作業		
		1	2	3	4	5	6	属性付加	抽出情報選択	抽出情報編集
Piggy Bank	RDF	○	○		○			○		
Potluck	RDF	○	○			◎		○		○
Dapper	HTML	○		○	○			○	○	
Karma	HTML	○	○			◎		○	○	○
本システム	HTML	○	○	○	○	○	○	○	○	

「対象型式」は，ウェブマッシュアップを実施できる Web ページの形式を指す．通常は HTML 形式であるが，一部のツールは RDF（Resource Description Framework）と呼ばれる形式のみを対象としている．RDF とは，セマンティック・ウェブを実現するためにメタ情報が付加されたページである．

「2.2 節の要件を満たすか」は，2.2 で挙げた要件 1〜6 を満たす機能が備わっているかどうかを示している．「○」が記述されていれば，その要件を満たしていることを意味する．また，「◎」は，我々が要件を満たすのに必要と考えていた以上の機能が実装されていることを意味する．

「フレームワーク使用者に要求される作業」は，ウェブマッシュアップを実施するためにエンドユーザが実施する必要のある作業を示す．「属性付加」は抽出情報に属性名をつけることである．例えば開催日を表す抽出情報には「日付」，開催場所を表す抽出情報には「開催場所」という属性をつける．これにより，Web ページから抽出した一つ一つの抽出情報について，同じ意味を持つ抽出情報を一つにまとめて識別することが可能になる．「抽出情報選択」は，ウェブマッシュアップで収集したい抽出情報を選択することである．選択の方法はマウスクリックや範囲選択があるが，大きな手間の差は無い．「抽出情報編集」は，ウェブマッシュアップによって取り出した情報の内，文字列形式の情報を対象とした機能であり，取り出した文字列を自動的に編集するための規則を作成する機能である．例えば，「A 月 B 日」「C/D」という日付の文字列を，システムが自動的に「A-B」,「C-D」に編集し，形式を統一してくれる．規則の作成方法はツールによって異なるため，個別のツール紹介にて述べる．エンドユーザに要求される作業量は少ないことが望ましい．

Piggy Bank[2] では，解析対象を RDF 形式のみに限定している．抽出情報の収集は，RDF ファイルのメタ情報に基づいて自動的に実施されるため，フレームワーク使用者は RDF ファイルを提示するだけでウェブマッシュアップを実施できる (要件 1)．また，収集した抽出情報を検索可能な Web ページとして配信する (要件 2)．さらに，この Web ページをサーバにアップロードすることで，誰でもこの Web ページを訪れることができる (要件 4)．Piggy Bank では，RDF ファイルに書かれているタグを，抽出情報の属性名とみなしており，別途属性名を付けたい場合には，タグを書き換える必要がある．

Potluck[3] も Piggy Bank と同じく，RDF 形式に対象を限定することで，抽出情報を自動的に抽出し，検索可能な Web ページとして出力する (要件 1，2)．こちらは収集した文字列を編集する機能をサポートしている．ウェブマッシュアップ後の抽出情報の一部を手作業で編集すると，同じ属性の抽出情報はシステムにより自動的に同じように編集される．この機能を用いて，文字列の中から数値だけを抜き出すことも可能である（要件 5）．

Dapper[14] は Piggy Bank を改良し，RDF 以外の形式に対応できるようにしたツールである．Dapper では，フレームワーク使用者が Web ページ上で収集したい箇所をクリックすることで，抽出情報の収集のためのルールが生成される (要件 1)．そのルールに基づいて抽出情報を自動的に収集し，XML（Extensible Markup Language）あるいは RSS（RDF Site Summary）フィードを Dapper のサーバにアップロードする (要件 4)．Dapper の Web

サイトを訪れた者は，RSS リーダを用いることで，対象となる Web ページに新しく追加された抽出情報を取得することができる (要件 3)．

Karma[5] では，Web ページから収集したい箇所の文字列をドラッグアンドドロップすることで，XPath を生成する (要件 1)．集めた抽出情報は表にまとめられ，XPath を生成した者は表の検索とソートが可能である (要件 2)．Karma では，Potluck と同じく，収集した文字列を自動的に編集する機能を備えている (要件 5)．

本稿で提案するシステムは，対象型式を HTML とし，更に要件 1〜6 の解決を目指す．対象型式を HTML とする理由は，情報ポータルには RDF が提供されていないものも存在するためである．表 1 より，6 つの要件の内，要件 1〜5 は既存の技術の組み合わせで実施できることが分かった．そこで，本システムでは要件 1〜5 を全て実施できるように技術を組み合わせる．

この場合，エンドユーザに「属性付加」,「抽出情報選択」,「抽出情報編集」の全てを課すことになるが，本システムでは「抽出情報編集」について，要件 5 を満たすために，文字列を取り出すことしか求めておらず，Potluck や Karma のように複雑な編集をする機能は必要ないと判断したからである．そのため，エンドユーザが「抽出情報編集」作業を行う必要はない．

また，表 1 に示したように，要件 6 を満たすシステムはまだ無い．そこで，本件では要件 6 を実施するための方法も提案する．

4. システム設計

システムの設計にあたっては，先行研究を比較した上で，本システムの使用者，すなわちエンドユーザの負担を減らすことを優先する．

4.1 システム構成

本システムは，情報ポータルを格納するサーバだけではなく，エンドユーザ側に抽出情報を収集するための規則を生成するプログラムが必要となる．本システムでは，これをブラウザアドオンとして実装する．その理由は，本システムの使用者が Web ページ上で抽出したい文字列を選択するためには，Web ページを閲覧している，つまりブラウザを使用する必要があるためである．ブラウザが立ち上がっていれば，ブラウザアドオンをすぐに使用することができる．

本システムの構成図を**図 1** に示す．ブラウザアドオン，情報ポータル格納用サーバが今回の実装範囲である．次節以降で，各機能について説明する．

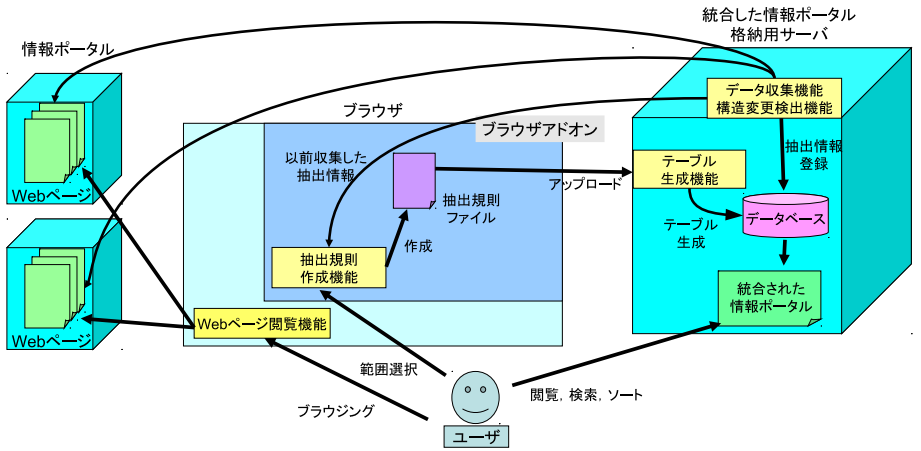

図 1 システム構成図

4.2 抽出規則作成機能

閲覧している Web ページに対して，抽出したい文字列を，エンドユーザがマウスの範囲選択によって示し，例示された文字列に対する HTML タグのパスから，システムが抽出規則を作成する．抽出規則は，収集したい抽出情報の種類，すなわち「属性」の数だけ作成する必要がある．例えば，情報処理学会のセミナー情報からタイトルと日付を収集したい場合,「タイトル」の属性と「日付」の属性，計 2 つの抽出規則を生成する．作成した複数の抽出規則はシステムにより 1 つのファイルにまとめられる．これを抽出規則ファイルと呼ぶ．抽出規則ファイルは，システムによって情報ポータル格納用サーバへアップロードされる．この機能により，要件 1 を満たすことができる．

4.2.1 抽出対象の指定

Rattapoom[5] によると，一つの属性につき一つ以上，収集したい抽出情報の例を示せば，抽出情報の抽出規則は生成可能である．抽出したい箇所を示す方法は複数ある．Dapper では，抽出したい箇所をクリックすることで抽出対象を指定する．このとき，抽出されるのは HTML タグで囲まれた文字列の全体に限る．もう一つの方法として，Karma で採用されている，Web ページ上で抽出したい箇所をマウスによって範囲選択する方法がある．

我々は，マウスによる範囲選択によって抽出対象の指定を行う方式を採用する．この方法を採用した理由は，マウスによる範囲選択により，要件 5 を満たすことが可能になるからで

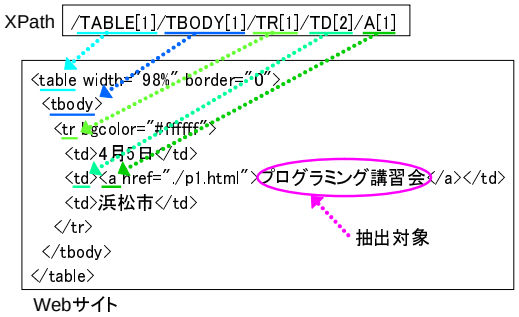

図 2 XPath による抽出対象指定の様子

ある．

Potluck，Karma では，システムが収集した文字列を自動的に編集する規則を，エンドユーザが作成することが可能であった．これでも要件 5 は実現可能である．しかし，本システムで求めているのは文字列の中からソートに使いたい部分を取り出すことだけである．これを実施するだけであれば，文字列を編集するための規則は作成しなくて良い．抽出規則作成の段階で，抽出情報として取り出したい箇所が範囲選択されているため，その選択範囲から文字列全体が必要なのか一部だけを取り出したいのかが分かるからである．Rattapoom[5] によれば一部の部分文字列を取り出したい場合には，部分文字列の前後にある文字列を調べることによって自動的に編集する規則は生成できる．

4.2.2 抽出規則の形式

このシステムでは，Web ページから情報を取り出すために XPath(XML Path Language) を使用する．XPath とは，XML ドキュメントあるいは HTML ドキュメントにおいて，特定の部分を指定するための構文である．XPath の例を**図 2** に示す．XPath を”/TABLE[1]/TBODY[1]/TR[1]/TD[2]/A[1]”と記述することにより，「プログラミング講習会」と記述されている箇所を指定することができる．

4.3 テーブル生成機能

抽出規則ファイルに記述された一つ一つの抽出情報に対する属性名，および抽出規則ファイル名より，収集した抽出情報を保存するためのデータベーステーブルをサーバ上に自動で作成する．

4.4 抽出情報収集機能

サーバ上のシステムはアップロードされた抽出規則ファイルに従って，抽出対象となる Web ページを定期的に巡回し，抽出情報を収集する．収集した抽出情報はデータベースに保管される．この機能により，要件 3 を満たすことができる．

4.5 構造変更検出機能

抽出情報収集機能を実行すると同時に，以前の巡回時に収集した抽出情報と同じものが 1 つでも抽出されるかを調べる．以前収集した抽出情報が全く抽出できない場合，システムは Web ページの構造が変更されたと判断し，抽出規則作成機能を実行する．抽出規則作成機能が自動的に実行されることによって，抽出規則ファイルが自動的に変更される．

4.5.1 抽出規則の再生成

抽出規則の再生成を行うためには，Web ページ上の抽出したい箇所の選択をもう一度実施する必要がある．4.4 節で言及したとおり，本システムでは収集した抽出情報をサーバ上のデータベースに保存しており，Web ページの構造が変わる以前の定期巡回において取得した抽出情報もサーバ上に残っている．抽出対象となる Web ページの HTML の構造が変化してしまっても，記載されている抽出情報そのものが全て変わるとは考えにくい．つまり，以前抽出したことがある抽出情報の一部は，構造変化後の Web ページの中に残っている可能性が高い．そこで，この残った抽出情報を用いることで抽出規則再生成を自動化する．

抽出規則を再生成する手順としては，構造が変化した Web ページに対して，サーバに残っている抽出情報によりテキストマッチングを行う．これにより，Web ページ上でエンドユーザが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．この技術により，要件 6 を満たすことができる．

4.6 情報ポータル

データベースに収集された抽出情報を，Web ページとして出力する．この Web ページでは，データベース内の抽出情報を表形式でまとめ，表の検索，ソートを行う機能も備える．これにより，要件 2 を満たすことができる

4.6.1 収集した抽出情報の提示

収集した抽出情報を提示する方法としては，CSV ファイルとして出力する方法，RSS フィードとして配信する方法，Web ページとして出力する方法がある．

CSV ファイルとして出力する場合，表計算ソフトやデータベースソフトを利用することで，抽出情報を検索，ソートすることが可能である．しかし，巡回のたびに新しく CSV ファ

イルが生成され，さらに出力された CSV ファイルの管理はエンドユーザ自身が行う必要があるため，継続的に利用する場合には，収集した情報の管理が煩雑になってしまう．

RSS フィードとして配信する場合，RSS リーダを用意することでフィードを受信することができ，エンドユーザ自身が能動的に行動を起こさなくても，定期的に新しく追加された抽出情報を得ることができる．しかし，RSS は抽出情報に属性名をつけることができず，また，RSS 自体が情報を蓄積する目的の技術ではないため，抽出情報を蓄積し，過去に保存したものも含めて，検索やソートにより抽出情報を利用することには不向きである．

そこで，本稿では，Web ページとして出力する方法を採用する．この方法では，サーバ上にデータベースを用意し，収集した抽出情報をデータベースに保存することで，蓄積された全ての抽出情報を対象として，検索，ソートを行うことを可能にする．また，抽出情報の収集，蓄積が全てサーバ上で自動的に実施されるため，エンドユーザが自らの手で，定期的な抽出情報の収集，および収集した抽出情報の管理を実施する必要もなくなる．

さらに，Web ページとして収集した抽出情報を公開することで，情報ポータルの統合を実施した者以外の人も，収集した抽出情報を閲覧することが可能となり，他の人との共有が可能になる．これにより，要件 4 を満たすことができる．

5. 実装システム

本章では，実装したシステムについて述べる．**図 3** にブラウザアドオンのインターフェースを，**図 4** に情報ポータルのインターフェースを示す．

5.1 ブラウザアドオンインターフェース

エンドユーザは，ブラウザアドオンインターフェースを用いて，Web ページから抽出情報を収集する規則を作成し，その規則をサーバにアップロードできる．ウィンドウ左下部にあるブラウザウィンドウから，収集したい抽出情報をマウスによって範囲選択し，「抽出規則生成」ボタンをクリックすることで抽出規則が生成される．図 3 では，情報処理学会 IPSJ カレンダー (`http://www.ipsj.or.jp/cgi-bin/ipsj_calendar.cgi`) からイベント情報の抽出規則を作成している．

ウィンドウ右下部には，抽出規則ファイルが表示され，生成した抽出規則の確認や修正を行うことができる．

「抽出対象箇所の表示」ボタンをクリックすることにより，本システム使用者は現在作った抽出規則で抽出される抽出情報を確認することができる．

ウィンドウ下部の「ファイルのアップロード」ボタンをクリックすることにより，ファイルアップロード用の確認ウィンドウが表示される．このウィンドウにおいて，抽出規則ファイルのファイル名を入力し，アップロードを指示する．これにより，右下の抽出規則ファイル表示ウィンドウに表示されているテキストが，抽出規則ファイルとしてサーバにアップロードされる．この時，サーバではデータベーステーブルの作成を実施する．

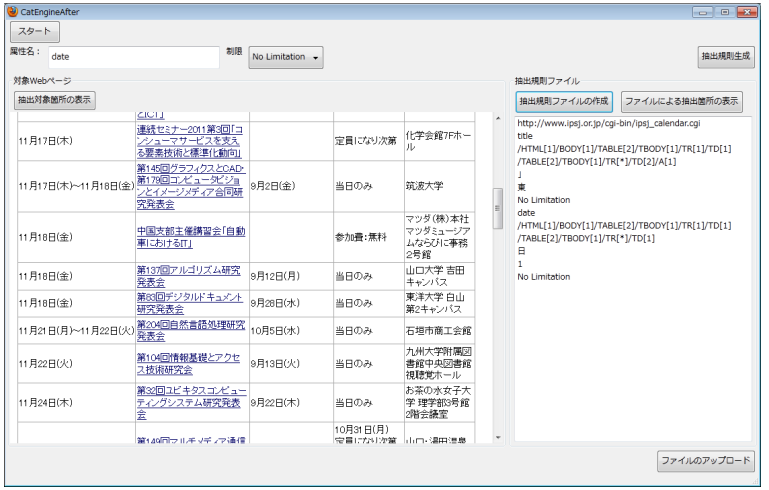

図 3 ブラウザアドオンインターフェース

ファイルのアップロードにおいて，既にサーバにアップロードしている抽出規則ファイルと同じファイル名を入力することで，情報ポータルに新しく抽出規則を追加することができる．これにより，複数の Web ページに記載されている抽出情報を横断的ひとつの情報ポータルにまとめ，新たな情報を作り出すことが可能となる．

5.2 情報ポータルのインターフェース

本システム使用者は，情報ポータルを用いて，データベースに蓄積されている抽出情報を閲覧することができる．図 4 は，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を抽出した情報ポータルに対し，イベント名称に「講習会」，日付に「11 月」を含む抽出情報を絞り込み検索したものである．入力フォームに検索したい言葉を入力することで属性ごとに絞り込み検索を実施することが，表の見出しをクリックすることでソートを実施することができる．

図 4 情報ポータルのインターフェース

6. 評　　価

本章では，下記の二点を評価する．

エンドユーザへの評価実験中である．ここでは，エンドユーザへの使用可能性と情報抽出精度について，既存システムとの比較により，本システムを評価する．

6.1 エンドユーザ使用可能性の評価

システム使用者の手間を測定するために，ステップ数[5] と呼ばれる指標を比較する．これは，システムを動かすためにシステム使用者が行った動作にステップ数と呼ばれる点数を付け，点数の大きさによって手間の優劣を評価するものである．下記の動作 1 つにつき 1 ステップ数が割り当てられている．

- テキストボックスに文字列を入力する
- ボタンをクリックする
- ドロップダウンリストから，項目を選択する
- テキストのをドラッグアンドドロップを行う

本システムの手間は Dapper と同じであると考える．何故ならば，本システムと Dapper の動作の違いは，欲しい情報を範囲選択するかクリックするかでしかないからである．

Dapper[14] は，エンドユーザが使用できるシステムである．本システムはその Dapper と同じ手間で使用することが可能である．従って，本システムはエンドユーザが使用することができるシステムであると考える．

6.2 情報抽出精度の評価

本システムを用いて欲しい情報を抽出できることを調べるために，実際に稼動している情報ポータルの再現を試みた．我々が選択した情報ポータルは，CPD(Continuing Professional Development) ポータル (http://jfes.heteml.jp/cpd/programs/publicindex) である．こちらでは，CPD プログラムと呼ばれる講義や講習会のタイトル，日付，開催場所を収集している．CPD ポータルは多数の Web ページから講義や講習会の情報を収集しているという点，そしてその収集先が明らかであるという点から，抽出できるデータの種類の比較に理想的だと考えた．CPD ポータルで抽出されている情報を目視で確認したところ，我々は CPD ポータルで抽出されている情報と同じ物を取り出すことが可能であると判断できた．従って，本システムで実装した機能により，欲しい情報を抽出することが可能であると考えた．

7. ま と め

インターネットから情報を取得する手間を削減する方法として,「情報ポータル」というものがあり，Web ページの閲覧者自身が欲しい情報だけを得るためには，閲覧者が情報ポータルを作る必要がある．そこで，エンドユーザが使うことができるウェブマッシュアップ技術を組み合わせることによって，情報ポータルを作成できるためのシステムを提案した．エンドユーザが情報ポータルを作るために必要な以下の 6 つの要件を洗い出した．

- 抽出すべき情報を指定する作業負担を減らしたい
- 抽出情報を検索，ソートしたい
- 新しい情報をシステムが自動的に追加して欲しい
- 情報ポータルを他の人と共有したい
- 抽出した文字列を部分的に切り出したい
- 抽出規則をシステムが自動的に修正して欲しい

6 つの要件の内，5 つは既存のマッシュアップ技術を組み合わせることで解決できることがわかったが，抽出規則を自動的に修正するという要件は，既存のマッシュアップ技術で解決することはできなかった．そこで，我々は抽出規則再生成機能を実装することによってこの問題を解決した．

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 Huynh, Stefano Mazzocchi, David Karger : Piggy Bank: Experience the Semantic Web Inside Your Web Browser, 4th International Semantic Web Conference, pp.413-430 (2005).
3) 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).
4) 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).
5) Rattapoom Tuchinda, Pedro Szekely, and Craig A. Knoblock : Building Mashups By Example, 13th International Conference on Intelligent User Interfaces, pp.139-148 (2008).
6) 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).
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) Rob Ennals, David Gay : User-Friendly Functional Programming for Web Mashups, 12th ACM SIGPLAN International Conference on Functional Programming, pp.223-234 (2007).
9) M. Cameron Jones, Elizabeth F. Churchill, Michael B. Twidale : Mashing up Visual Languages and Web Mash-ups, Hawaii International Conference on System Sciences, pp.143-146 (2008).
10) Woralak Kongdenfha, Boualem Benatallah, Julien Vayssiere, Regis Saint-Paul, Fabio Casati : Rapid Development of Spreadsheet-Based Web Mashups, 18th International World Wide Web Conference, pp.851-860 (2009).
11) Guiling Wang, Shaohua Yang, Yanbo Han : Mashroom: End-User Mashup Programming Using Nested Tables. 18th International World Wide Web Conference, pp.861-870 (2009).
12) 平山 雅樹, 新野 朝丈, 児玉 公信, 松澤 芳昭, 太田 剛 : 学生プロジェクトが直面した問題事例とアジャイルによる対処可能性の考察, 情報処理学会研究報告 IPSJ SIG Technical Report,Vol.2011-IS-115, No.3 (2011).
13) Eric Griffin : Foundations of Popfly: Rapid Mashup Development, Apress(2008).
14) Dapper: The Data Mapper (online), available from ⟨http://open.dapper.net/⟩ (accessed 2011-10-24).
15) iGoogle (online), available from ⟨http://www.google.co.jp/ig⟩ (accessed 2011-10-27).
16) Yahoo pipes (online), available from ⟨http://pipes.yahoo.com/pipes/⟩ (accessed 2011-10-27).