# マッシュアップツールとしての SQL の利用: 拡張可能関係データベースを用いた Web サービス統合システムの設計と実装

松井　勇樹† 　市川　哲彦‡ 　田中　稔†

†山口大学大学院理工学研究科　〒755-8611 宇部市常盤台 2 丁目 16-1

‡山口大学メディア基盤センター　〒755-8505 宇部市南小串 1-1-1

**あらまし**　近年，XML で記述されたデータ利用が普及するに従い，XML でのデータ管理や Web サービスの提供がなされるようになった．また，それらを組み合わせたメディエータやマッシュアップと呼ばれるアプリケーションを組み込む事も行われている．本研究ではそのような Web サービスの統合利用を容易にするために，拡張可能データベース管理システムから Web サービスの利用を可能とするためのツールの開発を行った．このツールは，Web サービスへアクセスするためのユーザ定義関数の XML による仕様記述から，関数定義を与える SQL 文を生成することができる．これらの関数は Web サービスが返した値を関係として返す集合値関数なので，SELECT 文の FROM 節に通常の関係と同様に指定することができる．従って，SQL の標準的な問合せ記述能力を用いて各種 Web サービスや関係データベースを統合利用することが可能となる．これにより，短時間で各種 Web サービスを組み合わせたアプリケーションが開発可能である．これを示すために，本ツールを用いて開発したアプリケーションについても報告を行う．

**キーワード**　Web サービス，拡張可能データベース管理システム，統合利用

# SQL as a Mashup Tool: Design and Implementation of a Web Service Integration Approach based on the Concept of Extensible Relational Database Management Systems

Yuki MATSUI† 　Yoshihiko ICHIKAWA‡ 　and 　Minoru TANAKA‡

† Graduate school of Science and Engineering, Yamaguchi University　2-16-2 Tokiwadai, Ube-shi, Yamaguchi, 755-8611 Japan

‡ Media and Information Technology Center, Yamaguchi University　1-1-1 Minamikogushi, Ube-shi, Yamaguchi, 755-8505 Japan

**Abstract**　Recently Web services based on the HTTP and XML technology have become widely used, and application programs or mashups integrating such services have also proliferated. In order to support the integration process, we have developed a tool to convert user-defined function specifications written in XML to the corresponding user-defined functions loadable to PostgreSQL, an extensible relational database management system. With this conversion layer provided by the tool, multiple Web services and relational databases can be integrated in SQL, and therefore, integrated applications or mashups can be built quickly without being bothered by variety of Web service interfaces defining the calling sequences and result data types. Moreover, even if one of the Web service providers a particular mashup depends on changes its service specification, the application does not need to be modified accordingly, when the corresponding function's calling interface remains intact by changing the specification of the function body. This property which we refer to as Web service independence is quite important since Web services interfaces may change without any previous notices.

**Keyword**　Web Service, Extensible Relational Database Management System, Integration

## 1. 研究の背景と目的

Extensible Markup Language(XML)で記述されたデータの利用が普及するに従い，XML データベースの構築や XML データベースへの問合せの利用がなされるようになり，それらを基盤とした Web サービスの提供もなされるようになった[1]．またこれらを組み合わせて付加価値をつけるメディエータやマッシュアップと呼ばれるアプリケーションを組むことも一般的になりつつある．

業務用に管理されている各種データを Web サービス

で入手可能な各種 XML データと統合して利用するための方法としては，関係 DBMS の XML 抽象データ型を使って関係モデルの世界で統合する方法[2] や，逆にラッパを用いて XML データ管理の世界で XML データとして統合する方法[3]，アプリケーションの独自ロジックで関係データと XML データを組み合わせる方法などが挙げられる．しかしいずれの手法も従来の SQL 等の古典的な関係問合せ言語の範囲を超えているため，問合せ構成にしてもアプリケーション構成にしても新たな技術の習得・利用が必要となる．

実際に Web アプリケーションで利用されるデータは検索サービスに代表されるようにあまり複雑ではなく比較的フラットな構造を持ったものが多い．従って記述能力として制限はされるものの，Web サービスに対してのラッパを拡張可能関係 DBMS に内蔵することが可能なケースも多々あると考えられる．その場合 SQL 等の一般的な問合せ言語を用いて関係データを Web サービスのデータと同時に利用する事が可能になる．また複数の Web サービスがあった場合でも，各サービスにアクセスするための関係モデル内でのスキーマを理解するだけで SQL 等の問合せ言語によってそれらを統合利用することが可能になる．

そこで本研究では拡張関係 DBMS の一つである PostgreSQL[4]をプラットフォームとして利用し，各種 Web サービスを拡張 SQL から利用するためのシステムの開発を行った．本稿ではシステムの概要と有用性を示すために作成したいくつかのアプリケーションについて概略を述べる[1].

## 2. 関連研究

Transact-SQL[6]のユーザ定義関数を用いて Web サービスにアクセスするシステムとして，EaRDB[7]が報告されている．これは Web サービスの統合利用を SQL のレベルで行おうという点では本研究と類似しているが，ユーザ定義関数の自動生成機能は有していない．

また，Web サービスに対して SQL 的な構文でアクセスする事を可能とする YQL[8]が公開されている．PostgreSQL のビューに対して各種ルールを記述し，それらのアクションで Web サービスへのアクセス機能を呼び出すようなイメージに近い．また，機能の呼び出しに必要なパラメータは WHERE 節で指定している．データソースの XML による記述を与える点や SQL ライクな構文を用いる点では本研究と類似している．しかし，YQL はデータソースとして FROM 節で指定できるのは単一のテーブル（実質的にはデータ取得モジュール）であり複数のテーブルを指定することができないため、複数のサービスを利用する際には外部のアプリケーションを利用するか、複数のソースにアクセスするためのデータ取得モジュールをテーブルとして記述しなければならない点が異なる．

## 3. 対象とする Web サービス

Web サービスは HTTP を用いた情報交換によってサービスを提供する枠組みである．通常は XML によるデータのやり取りが行われる．W3C の定義では SOAP が基本となるようであるが，GoogleMaps API[9]の登場以降，REST 型サービスが主流となった．また，この API は近年の Web サービスマッシュアップブームの火付け役でもある．Web アプリケーションの統合をする上で SOAP, REST など各種プロトコルをサポートすることが必要となるが，本研究では REST 型サービスを前提としたシステムの開発を行っている．

## 4. ユーザ定義関数自動生成システム

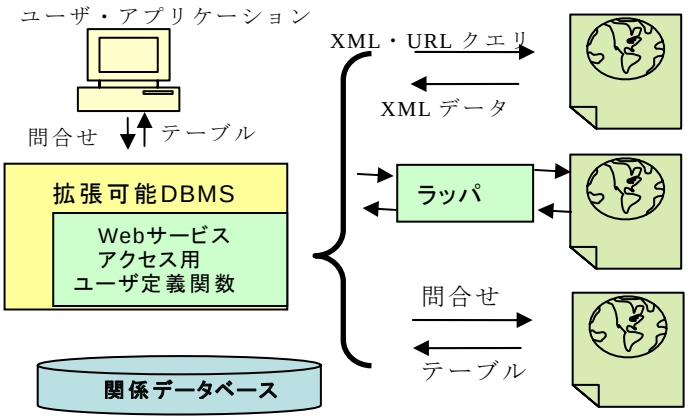


**図 1. システム概念図**

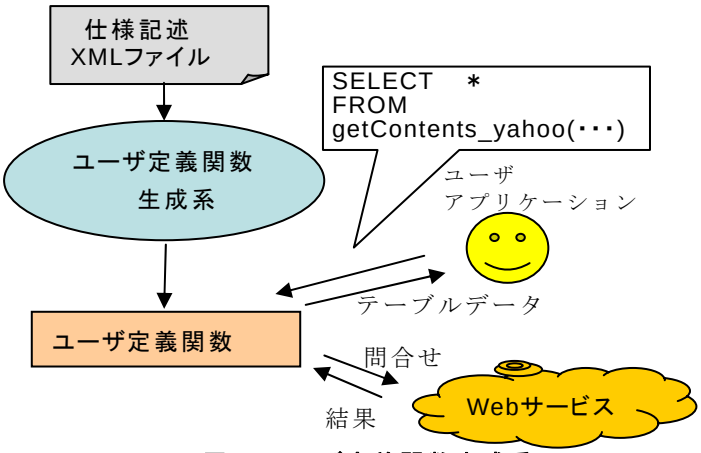


**図 2. ユーザ定義関数生成系**

本研究で開発したシステムの概念図を図 1 に示す．ユーザやアプリケーションは拡張可能 DBMS に設けた Web サービスに対するインターフェース（ユーザ定義

[1] 本研究の一部は文献[5]にて公表済みである．

関数）を利用する事により，SQL の世界で Web サービスを利用可能となる．

通常ユーザ自身がユーザ定義関数を直接作成するのであるが，本システムではユーザの負担を軽減し，ソフトウェアの信頼性を高め，かつ汎用性を持たせるために，ユーザ定義関数の定義は仕様を記述した XML 形式のファイルから自動的に生成することとした．ユーザ定義関数生成系の概念図を図 2 に示す．

本研究では，ユーザ定義関数の仕様記述に用いる XML ベースの言語を PostgreSQL Web-service Access Function markup Language (Pg/WAFL)と呼んでいる．この例を図 6 に示す． これは Yahoo! Web サービスに対して問合せ要求を出すためのものである．以下ではまず Web サービスアクセス用ユーザ定義関数の利用例を説明し，続いて Pg/WAFL による記述内容について説明を行う．

### 4.1. SQL での Web サービスのアクセス方法

Web サービスアクセス用に生成された関数の例として，Yahoo! Web 検索の例を考える．図 6 の記述からは getcontents_yahoo という関数が生成されるので，これ用いると，例えばキーワード Yahoo での検索要求は，次のように SQL で記述することができる:

```
SELECT *
FROM getcontents_yahoo($${}$$,'Yahoo','any')
```

関数の第 2 引数が問合せ，第 3 引数が問合せ対象のデータ形式の指定になっている．第 1 引数はオプションパラメータを渡すための二次元配列であり，この例では何も与えていないが，例えば次のように用いることができる:

```
SELECT *
FROM
getcontents_yahoo($${{'numberperpage','50'},{'start
number','1'},{'reqnumber','1000'}}$$,'Yahoo','any')
```

この例ではページングに関してのパラメータを指定しており，50 件ずつ，1 件目から 1000 件検索することが指示されている．第 1 引数は，このようにオプションパラメータを指定する際に主に利用される．

### 4.2. Pg/WAFL による記述

Pg/WAFL による記述は大きく三つのパートに分けられ，それぞれ次のような事項について指定を行う:

- Part I: 関数ヘッダ部
  関数のパラメータと返却値の型の指定を指定する;
- Part II: Web サービス記述
  Web サービスの呼び出しパラメータが関数パラメータからどのように計算されるかを指定する;
- Part III: 変換部
  Web サービスから返された結果をどの様にテーブルに変換するかの指定する．

Part I はユーザ定義関数のヘッダ部,すなわち関数名，引数，戻り値の型を指定しており，それぞれ FuncName 要素，FuncArgs 要素，および RetType 要素により記述されている．FuncArgs の子要素である FuncArg 要素が各引数の記述を与えており，argname 属性が引数の名前を,また要素のコンテントが引数の型を与えている．引数には引数リスト内の位置で識別されるポジショナル引数と，引数に関連付けられたキーワードによって識別されるキーワード引数の二種類があり，前者は第二引数以降に指定され，後者は第一引数の二次元配列で指定される．FuncArg 要素では，これらの区別は argtype 属性の値で指定されている．また，required 属性の値によって，引数が必須か省略可能かが指定されている．

RetType 要素は返却値の型を指定する．スカラー値を指定することも可能ではあるが，基本的には関係返すのでで，srf 属性を Yes にし，RowType 要素を用いて関係のタプルの型指定する．ここで，srf は Set Returning Function の略である．タプルの型の名前は RowType の name 要素で指定し，タプル要素は SrfElement 要素で指定する．図 3 の例では次のような SQL 文によりタプルの型が作成される:

```
CREATE TYPE contents_yahoo AS
  (rank integer, title text, summary text, url text)
```

属性 rank は Pg/WAFL では特殊な型 rank が指定されている．これは各タプルが，Web サービスが返したリスト中の何番目のレコードに対応しているかを表す値である．値は自動的に付与されるので，Part III のデータ変換部では明示的に指定する必要は無い．最終的に生成される関数は次のようになる:

```
CREATE FUNCTION
    getcontents_yahoo(text[][], text, text)
  RETURNS SET OF contents_yahoo
  AS 関数本体の記述
  LANGUAGE 'plphp'
```

Part II では，Web サービスへのアクセス方法が記述されている．baseurl には Web サービスの URL が，param_paging にはページングにかかわる 3 つのパラメ

ータが，param にはその他のパラメータが記述されている．ページングを行うためには，ページ単位で Web サービスにアクセスしてデータを取得し，最終的な結果とする必要があるが，この処理はすべて自動生成されるため，ここではどのようなパラメータを使うのかだけを指定すれば良い[2]．各パラメータには source が存在するが，これは Part I で記述したどの引数が使用されるかについて記述している．例えば，source="query"では，Part I で“query”という名前が付けられた引数を利用する．必須パラメータか否かは required 属性で指定しており，オプショナルなものについては param 要素のコンテントでデフォルト値を記述している．

Part III では，Web サービスから返されたデータを，XML で記述された表に変換する処理が記述されている．記述言語は様々なものが考えられるが，現在は XSLT を用いている．変換後の XML データをさらに内部形式に変換する処理が必要とされるが，その処理は生成されたプログラム内に自動的に埋め込まれる．なお，現在は変換部の正当性チェックは行っていない．型チェックは今後の（難しい）課題の一つである．

Pg/WAFL のような Web サービスの記述レイヤーを導入することで，Web サービスの変更があっても，Web サービスからユーザ定義関数へのマッピングを変更することでアプリケーションの変更を避ける事が可能となる．このことから，Pg/WAFL は物理データ独立性や論理データ独立性と同様な「Web アプリケーション独立性」を与える機能を果たしていると言える．

現在のツールは，PostgreSQL8.2.6，Pl/PHP1.3.3[10]，PHP5.2.4 を用いて実装した．

## 5. アプリケーション

本システムの有用性を示すために，いくつかアプリケーションを作成した．一つ目は簡易 RSS リーダで，これは Web サービスの SQL による利用のもっとも単純な例である．次がメタサーチエンジンであり，これは複数の Web サービスを統合利用する例である．三番目がニュースキーワード抽出記録システムであり，Web サービスとローカルデータベース，自然言語処理を行うローカルアプリケーションを SQL で統合利用する例である．

### 5.1. 簡易 RSS リーダ

簡易 RSS リーダは，RSS データを取得してきて表示するだけの簡単なアプリケーションで，特定の単語を含むタイトルの記事を検索する機能が付与されている．このアプリケーションは SQL 文を実行し，出力されたテーブルを整形するだけのシンプルな作りになっている．タイトル検索の部分も標準の SQL 演算である“LIKE”を用いて，以下のような問い合わせにより行っている:

```
SELECT * FROM
getcont_yahoorss($${}$$,'itmedia_ep.xml')
WHERE title LIKE  '%検索単語%'
```

### 5.2. メタサーチエンジン

メタサーチエンジンランキングの尺度などに関しては開発者のロジックで固定となり，また Web 検索エンジンのみを対象としているものが多く，近年の多様化している様々なエンジンへの対応は行われていない．そこで，これらのエンジンへのアクセスをユーザ定義関数にて行い，SQL の記述により自由な組み合わせやランキングを可能とするアプリケーションの開発が可能となる．

Web サービスアクセス用の関数は各 Web サービスから得たデータをテーブル形式で返すため，以下のような SQL 文により複数エンジンで検索した結果の結合操作が簡単に行える:

```
SELECT
ysearch.rank, ysearch.title, ysearch.summary,
ysearch.url, yimg.title, yimg.summary, yimg.url
FROM getcont_yahoo($${}$$,'Okinawa','any')
  AS ysearch
LEFT JOIN
getcont_yahoogazou($${}$$,'Okinawa','any')
  AS yimg
ON url_compare(ysearch.url,yimg.url)>=1
ORDER BY ysearch.rank ASC
```

この例では Yahoo!ウェブ検索サービス・Yahoo!画像検索サービスを組み合わせ，前者に後者の情報を付加する処理を行っている．テーブルの結合条件はユーザ定義関数 url_compare で与えているが他の条件に変更することで組み合わせ方は自由に変更できる．上記の SQL を用い作成したメタサーチエンジンのスクリーンショットを図 4 に示す．図 4 は‘羽生善治’というキーワードを用いて検索した結果が出力されている．

### 5.3. ニュースキーワード抽出記録システム

このシステムはニュースサイトにアクセスし，RSS データを基に記事を取得した上で単語の出現頻度を求め，ローカルデータベースに定期的に格納する簡単なプログラムである．システム構成を図 3 に示す．この

[2]現在はサポートしているページングモデルが一種類だけであるが，今後は追加を検討している．

アプリケーションを構築するために，あらかじめ形態素解析ツール茶筌[11]の機能をユーザ定義関数として利用できるようにしてある．この関数は URL をパラメータとして与えると，データを取得した後に形態素解析を行い，さらに単語の出現頻度を求めて表形式にて返却する．従ってこのアプリケーションでは，データ取得，形態素解析，集計結果の保存は全て SQL で記述しており，アプリケーション本体は PHP による 25 行ほどのスクリプトで実現している．ユーザインターフェースとしてはタグクラウドを用いた Web アプリケーションを作成している．このアプリケーションは，外部システムの利用も含め，様々な情報源や処理を統合利用する例の一つである．

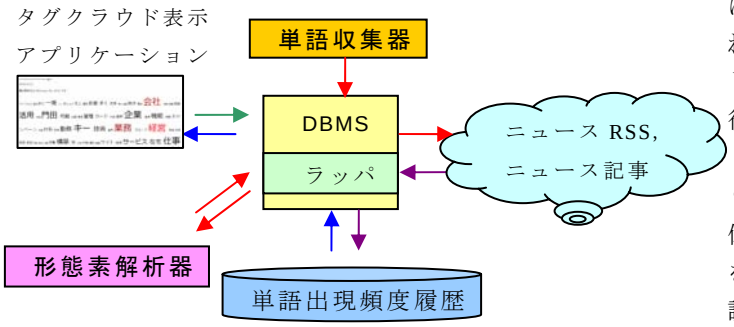


**図 3. システム構成**

単語収集器は毎日決まった時刻に呼び出され，ラッパを呼び出して，出現単語とその頻度のリストを取得する．取得したリストは単語出現頻度履歴に保存する．それぞれは以下の機能を持つ．ラッパ部は，Web サービスにアクセスするために Pg/WAFL での記述から自動生成されたユーザ定義関数と，茶筌呼出し用の関数を組み合わせて作られており，ニュース記事配信サービスと形態素解析サービスを組み合わせて得られるメディエータの役割を果たしている．そのため SQL のプログラマから見ると，一つの Web サービスであるかのように見える．このように，SQL レベルでの関数の組合せで容易にメディエータが構成できるのが本システムの利点である．

上記のプログラムにより作成された単語出現頻度履歴から，タグクラウド生成プログラムを作成した．アプリケーション画面を図 5 に示す．ここでは，頻出する単語のフォントサイズを大きく，更に頻出度が高い単語は色を黒から赤に変えていく，という表現を行っている．

また，スライダーバーにより日付を指定し，単語出現頻度履歴から過去のデータを表示することが可能となっている．

## 6. まとめと今後の課題

本研究では様々な Web サービスに対するインターフェースを，XML ベースの言語 Pg/WAFL による仕様定義を与えるだけで簡単に作成可能なシステムの開発を行った．これにより様々な情報源を統合利用するための枠組みとして SQL を活用することが容易に可能となる．

今後の課題としては，システムにより作成されたユーザ定義関数を用いたアプリケーションを更に開発すると共に，ユーザ定義関数の作成も含めたトータルのアプリケーション開発効率について，開発性，保守性，拡張性などの指標により評価を行う事が挙げられる．また，ページングや Web サービスパラメータの渡し方にも現在サポートしていないスタイルがあるため，それらについても検討が必要である．さらに，これは難しい問題ではあるが，変換部の正当性のチェックも今後の検討課題として考えている．

この研究のように外部システムへのアクセスを行う場合，トランザクションの隔離レベル[12]が極端に低くなる点については注意が必要である．外部情報源を用いた際のトランザクションレベルの分析も今後の課題である，

## 文　献


[1] 横田一正監訳，國島丈生訳，“XML データベース入門,” 共立出版, 2006. (S. Abiteboul, P. Buneman and D. Suciu, Data on the Web: From Relations to Semistructured Data and XML, Elsevier, 2000.)

[2] A. Eisenberg, et al., “SQL2003 has been published,” SIGMOD RECORD 33(1), March 2004, pp. 119–126.

[3] M. F. Fernandez, et al, “Catching the boat with Strudel: Experiences with a web-site management system,” Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, Washington, 1998, pp. 549-552.

[4] 日本 PostgreSQL ユーザ会，http://www.postgresql.jp/

[5] 松井 勇樹，市川 哲彦，田中稔，``拡張可能関係データベース管理システムを用いた Web サービス統合,'' 第 11 回 IEEE 広島学生シンポジウム (HISS09), Nov. 21 - 22, 2009, D-08.

[6] Microsoft Inc,. “Transact-SQL リファレンス (Transact-SQL), http://msdn.microsoft.com/ja-jp/library/ms189826(SQL.90).aspx

[7] 大島 裕明，小山 聡，田中 克己，“EaRDB: Web 集約質問処理のためのプラットフォーム,” DBSJ Letters Vol.6, No.2, September, pp. 49-52, 2007.

[8] Yahoo! Query Language - Yahoo! Developer Network, http://developer.yahoo.com/yql/

[9] Google Maps API，http://code.google.com/intl/ja/apis/map

[10] Command Prompt Inc., “PL/PHP” http://projects.commandprompt.com/public/plphp

[11] 茶筌，奈良先端科学技術大学院大学情報科学研

究科自然言語処理学講座,　　*Concepts and Techniques*, Morgan Kaufmann, 1993.
[12]J. Gray and A. Reuter, *Transaction Processing:*


**図 4. メタサーチエンジンの出力例. Yahoo!Web 検索と Yahoo!画像検索の結果を結合し，Web 検索の結果に対して，そのサイトにある関連画像の一部を同時を提示している.**

**利用している画像の URL は，本論分執筆時において次の通りである．28 河出書房新社,http://www.kawade.co.jp/，29 寄せの構造 with 将棋 棋書ミシュラン，http://snow.freespace.jp/Rocky-and-Hopper/index.htm，30 将棋順位戦データベース,http://www.ne.jp/asahi/yaston/shogi/index.html，31 日本将棋連盟,http://www.shogi.or.jp/，32 ides,http://www.idesnet.co.jp/**

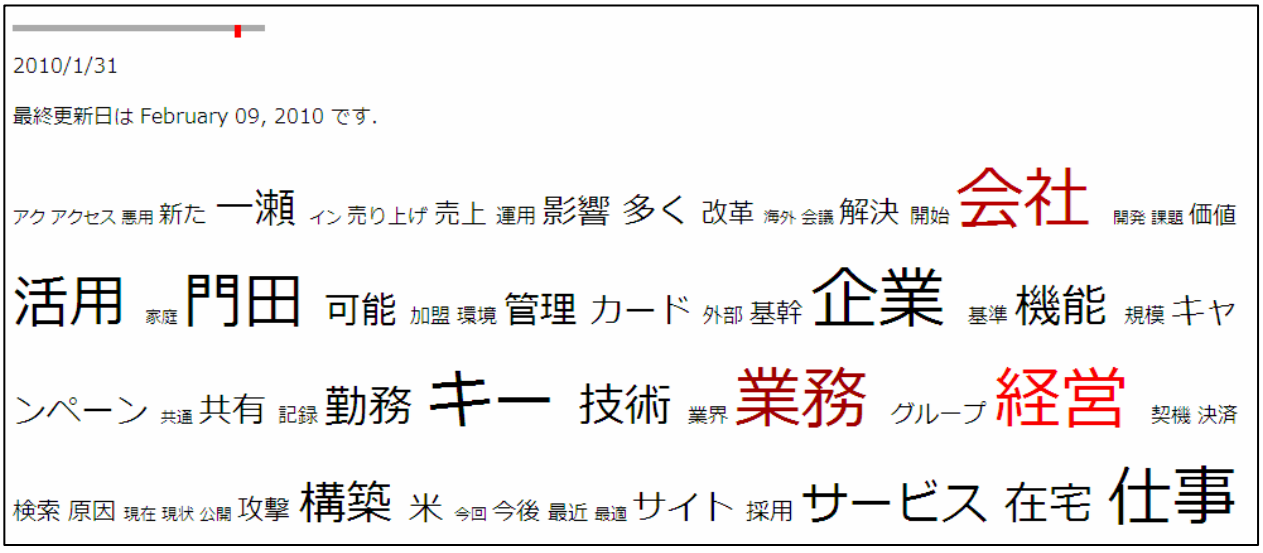


**図 5. ニュースキーワード抽出記録システムのユーザインタフェース: タグクラウドによる標示がなされている．また，スライドバーより閲覧日を選択する事も可能である.**

```
<?xml version="1.0" encoding="UTF-8"?>
<FuncSpec>
 <!-- Part I: function header -->
 <FuncName>getcont_yahoo</FuncName>
 <FuncArgs>
   <FuncArg argname='reqnumber' argtype='keyword'
            required="true">integer</FuncArg>
   <FuncArg argname='numberperpage' argtype='keyword'
            required="false">integer</FuncArg>
   <FuncArg argname='query' argtype='positional'>text</FuncArg>
   <FuncArg argname='format' argtype='positional'>text</FuncArg>
   <FuncArg argname='appid' argtype='keyword'>text</FuncArg>
   <FuncArg argname='startnumber' argtype='keyword'
            required="false">integer</FuncArg>
 </FuncArgs>
 <RetType srf='Yes'>
   <RowType name='contents_yahoo'>
     <SrfElement name='rank' type='rank' />
     <SrfElement name='title' type='text' />
     <SrfElement name='summary' type='text' />
     <SrfElement name='url' type='text'/>
   </RowType>
 </RetType>
 <!-- Part II: Web service binding -->
 <Request type='xml'>
   <url>
     <baseurl>http://search.yahooapis.jp/WebSearchService/V1/webSearch</baseurl>
     <param_paging start='start' size='results'
                   source_start='start' source_total='reqnumber' source_size='numperpage'/>
     <param name="appid" type="xsd:string" source="appid"
            required="false">############</param>
     <param name="query" type="xsd:string" source="query"
            required="true" />
     <param name="format" type="xsd:string" source="format" required="true" />
   </url>
 </Request>
 <!-- Part III: conversion of responce data to table data -->
 <Returning lang='xslt'>
   <xsl:stylesheet version="1.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:yahoo="urn:yahoo:jp:srch">
     <xsl:output method="xml" indent="yes"/>
     <xsl:template match="/">
       <table>
        <xsl:apply-templates/>
       </table>
     </xsl:template>
     <xsl:template match="yahoo:Result">
       <record>
         <title><xsl:value-of select="yahoo:Title"/></title>
         <summary><xsl:value-of select="yahoo:Summary"/></summary>
         <url><xsl:value-of select="yahoo:Url"/></url>
      </record>
     </xsl:template>
   </xsl:stylesheet>
 </Returning>
</FuncSpec>
```

**図 6.　Pg/WAFL 記述例**