Web API利用のためのプログラムライブラリ自動生成

高井正成[1]　阪口哲男[2]
筑波大学大学院図書館情報メディア研究科[1]
筑波大学図書館情報メディア系[2]
〒 305-8550 茨城県つくば市春日1-2

概要

近年，Webアプリケーションの提供者はその機能を他のアプリケーションから利用してもらうため，Web APIを提供する事例が増加している．Web APIではHTTPなどの通信プロトコルによるリクエスト・レスポンスにより通信を行うという大きな枠組みは共通しているものの，細かな仕様は各Web API提供者によって定められる．そのため各Web APIに互換性がなく，Webアプリケーション開発者はWeb APIを利用する場合，仕様を確認しそれぞれのWeb APIの仕様に沿ったプログラムライブラリを作成する必要がある．そこで本研究では開発者がプログラムライブラリを自分で作成する手間を省くため，Web APIの仕様文書から情報を抽出し，プログラムライブラリを自動生成する手法を提案する．

キーワード

Web API，Webアプリケーション，情報抽出

Automatic Generation of Program Libraries for Accessing Web APIs

Masanari TAKAI[1]　Tetsuo SAKAGUCHI[2]

Graduate School of Library, Information and Media Studies, University of Tsukuba[1]
Faculty of Library, Information and Media Science, University of Tsukuba[2]
1-2, Kasuga, Tsukuba, Ibaraki, 305-8550, JAPAN

Abstract

The case is increasing that Web Application providers publish Web APIs which are used by other applications with Web protocol. They are designed based on common simple protocol but detail specification of each of them is decided by its provider. So Web APIs don't have compatibility with each other. Web application developers have to read documents of Web APIs and make program libraries for them if they use Web APIs. This paper proposes a method to make program libraries automatically by means of information extraction from Web API documents in order to developers avoids the trouble of making program libraries.

Keywords

Web API，Web Application, Information Extraction

1. はじめに

近年，Webアプリケーションが自身の機能をWeb APIとして提供する事例が増えている．Web APIによってWebアプリケーション開発者は他の任意のWebアプリケーションの機能を利用することができる．Web APIリポジトリであるProgrammable Web[1]には現在6000件以上のWeb APIが登録されており，Web APIの登録数は年々増加している（図1）．

Webアプリケーション開発者はWeb APIを利用する場合，各Web APIの仕様を提供されるオンラインドキュメントなどで確認する．プログラミングの際はWeb APIを利用する処理はプログラムライブラリとして他の処理と分けて記述する場合が多い．Web APIではHTTPなどの通信プロトコルによるリクエスト・レスポンスにより通信を行うという大きな枠組みは共通しているものの，細かな仕様は各Web API提供者によって定められる．そのため各Web APIに互換性がなく，Webアプリケーション開発者はWeb APIを利用する場合，それぞれのWeb APIの仕様に沿ったプログラムライブラリを作成する必要がある．新たなWeb APIが公開された場合複数の開発者がそのWeb APIを利用しようとして，複数の同様の機能を持ったプログラムライブラリが作成されてしまう．実際にマイクロブログサービスtwitterのWeb APIではプログラムライブラリが複数作成，公開されており，公式サイトで紹介されているものだけでも43件のプログラムライブラリが存在し，Rubyという一つのプログラミング言語に限った場合でも5件のライブラリが存在する[2]．このように複数の開発者が別々にWeb APIのプログラムライブラリを作成することは非効率である．Web APIの仕様が変更された場合は，プログラムライブラリを修正しなければならないが，複数あるライブラリ全ての修正が必要となり，修正漏れがあったプログラムライブラリは動作不能になったり，機能が不足してしまったりすることもある．以上のような理由からWeb APIのプログラムライブラリを開発者が作成するのは開発において非効率であるといえる．そこで本研究では開発者がプログラムライブラリを自分で作成する手間を省くため，Web APIのオンラインドキュメントから情報を抽出し，プログラムライブラリを自動生成する手法を提案する．

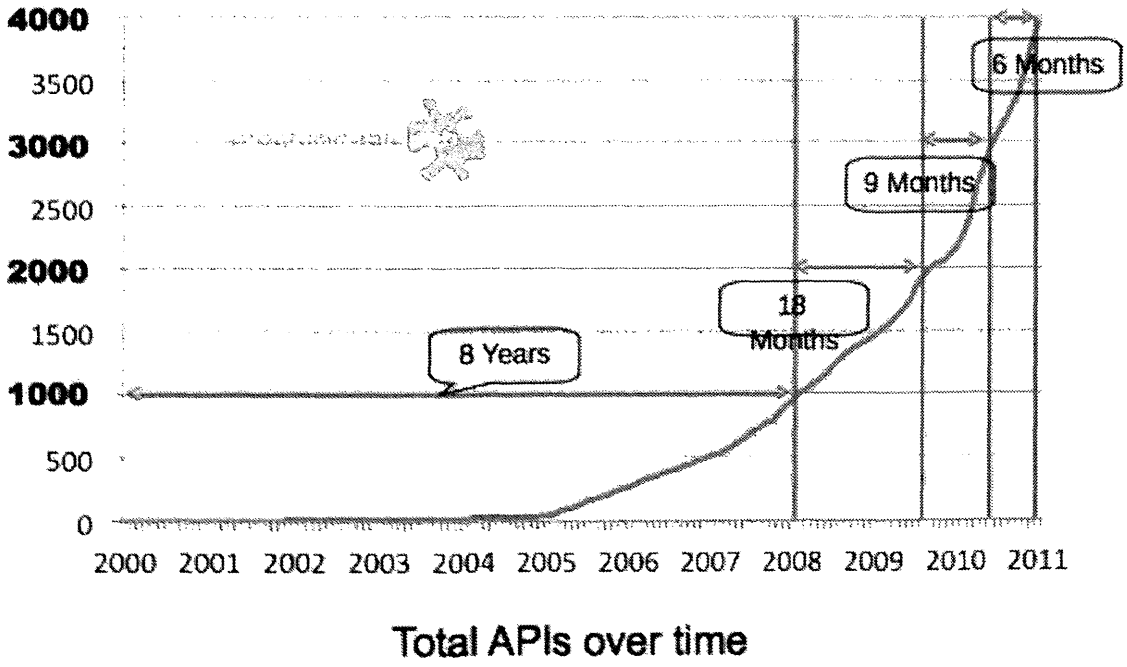

図1. Programmable Webにおける年ごとのWeb API登録数の推移([1]から引用)

2. Web APIのプログラムライブラリ生成手法

2.1. 概要

本章ではWeb APIのプログラムライブラリの生成手法について述べる．本手法では各Web APIのオンラインドキュメントを利用し，そこからプログラムライブラリの生成に必要な情報を抽出する（図2）．抽出した情報からプログラムライブラリを生成する．ここでいうオンラインドキュメントとは，Web APIを利用する開発者のためにWeb APIの仕様を記述したもので，リクエストのベースURLや利用できるパラメータ，レスポンスの形式などが記述されている．

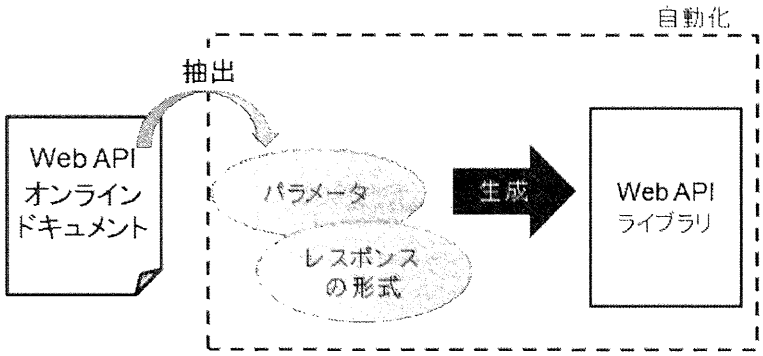

図2. Web APIプログラムライブラリ生成手法概要図

2.2. Web APIオンラインドキュメントからの情報抽出

プログラムライブラリの生成に必要な情報は各Web APIのオンラインドキュメントから抽出する．抽出にはWeb APIのオンラインドキュメントに見られる規則性を利用する．オンラインドキュメントではWeb APIの情報が提供元によって様々な記述形式で記されているがその中には幾つかの共通点がある．例としてtwitter APIのオンラインドキュメントとYahoo! Answers APIのオンラインドキュメントをそれぞれ図3，図4に示す．

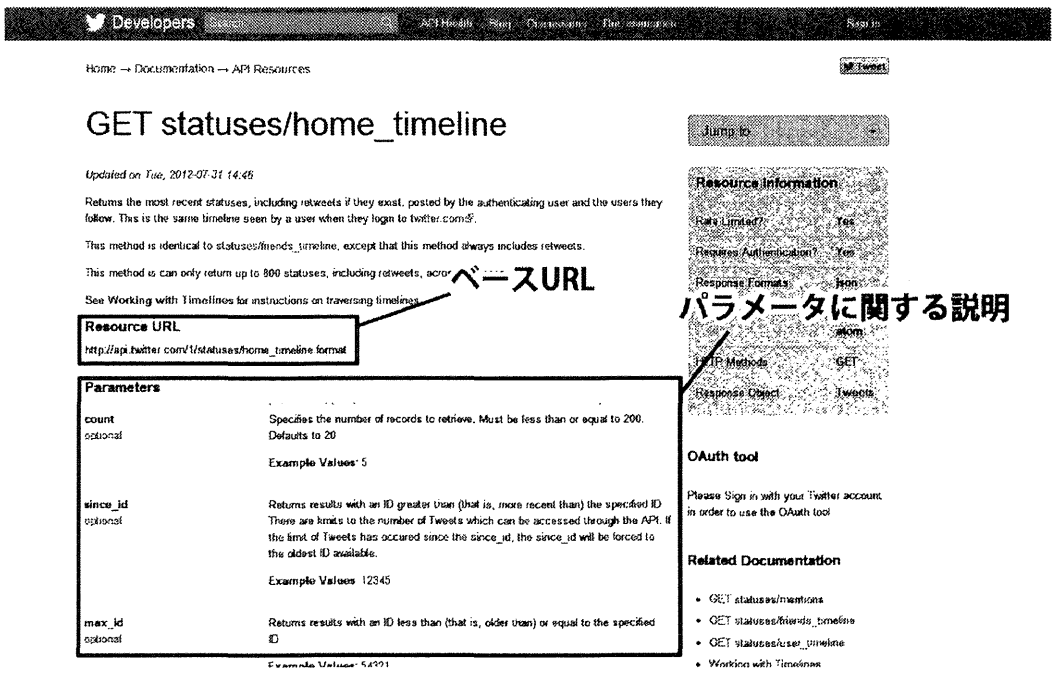

図3. Twitter APIのオンラインドキュメント（一部）

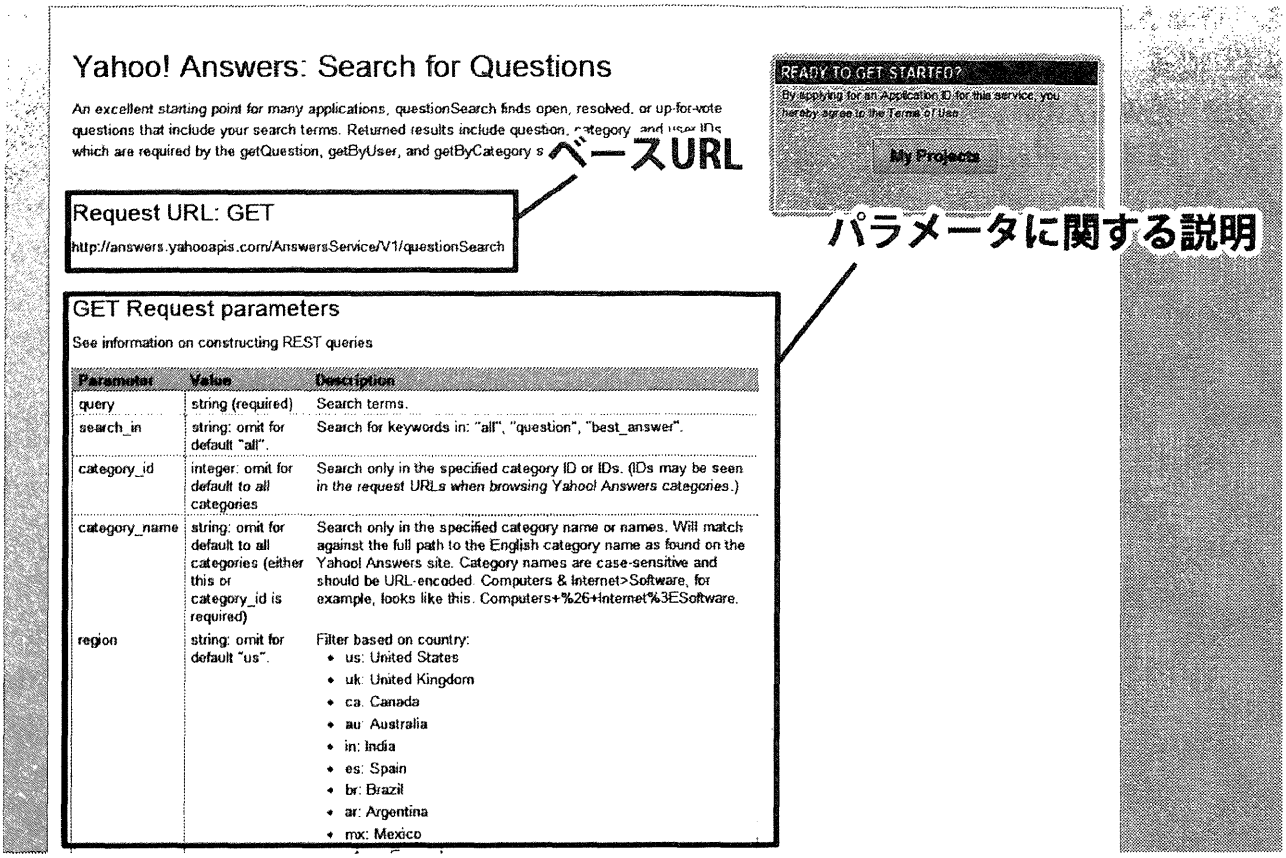

図4. Yahoo! Answers APIのオンラインドキュメント（一部）

Web APIの中で，特にREST型と呼ばれるものは，HTTPのリクエストを送りそのリクエストに応じたレスポンスが返される．図2，図3ではドキュメント中にResource URLもしくはRequest URLとしてベースとなるURL（ベースURL）が記述されている．その下にベースURLと一緒に送るリクエストパラメータについて，そのパラメータ名とパラメータの型や説明が記述されている．このように多くのオンラインドキュメントではベースURLの記述からパラメータに関する説明など共通の項目が説明されており，各情報は箇条書きや表によって整理されていたり，大文字によるタイトルの後に記述されていたりと記述の方法にある程度の規則性が見られる．本手法ではオンラインドキュメントのHTMLのタグ構造とキーワードに見られる規則性を利用し，情報の抽出を行う．

2.3. 抽出した情報の記述とプログラムライブラリへの変換

本研究では抽出した情報の記述にWebアプリケーション仕様記述言語であるWADL(Web Application Description Language)[3]を用いる．WADLはREST型のWeb APIの仕様を記述する場合に用いられる記述形式で，XMLで記述される．WADLを利用することで抽出した情報を機械的に解釈しやすい形で記述できる．現在WADLを利用することで既存の開発環境においてプログラムライブラリを生成することが可能となっている．例えば統合開発環境のひとつであるNetBeansではWADLを読み込み，プログラムライブラリを生成することができる．

WADLの生成にあたり，本研究では生成に必要とする情報を以下の7項目とした．本手法ではオンラインドキュメントからこれらの情報を抽出し，WADLを作成する．

- ベースURL…HTTPリクエストを送る際に指定するURL
- パラメータ名…リクエストで指定できるパラメータの名前
- パラメータの型…文字列や数字など各パラメータで指定できる型
- パラメータのデフォルト値…リクエストで指定しなかった場合の省略時値
- 必須パラメータか否か…必ず指定しなければならないパラメータであるか否か
- リクエストの形式…リクエストの送信方法に関する規定（GETメソッドかPOSTメソッドか，など）
- レスポンスの形式…レスポンス自体の形式に関する規定（XMLかJSONか，など）

3. 抽出手法の評価

本研究では抽出手法を，実際のWeb APIオンラインドキュメントに対して適用し評価する．

抽出精度は，人出による抽出との比較によって求める．対象はProgrammable Webに登録されているWeb APIのオンラインドキュメント（HTMLソースコード）とする．対象から，まず人手で2.3に示した7項目の情報を抽出する．次に抽出手法を実装したプログラムを対象に適用し，人手で得られた結果とプログラムの結果を比較する．

本手法のプログラムは，オンラインドキュメントの規則性に対応する複数の抽出ルールを実装したものとする．抽出ルールとは例えば「tableタグがあり，かつ“パラメータ”という単語がtableタグより上の行にある場合は，tableタグの第1要素をパラメータ名とみなす」などである．この例はパラメータ名に関する抽出ルールのひとつである．単純に抽出精度を求める他に，どのルールがより情報抽出に役立つか，いくつのルールがあれば情報抽出が行えるか実験を行い，それぞれの抽出ルールに対しての評価も行う．実験対象と，人手で抽出した情報と比較する際に，プログラム内で適用する抽出ルールを実験ごとに変更し，どのルールが抽出精度に影響を与えるか確認する．

4. 予備実験

本研究の手法の有効性を確かめるため，予備実験を行った．本章ではその予備実験の方法と結果，およびそれによって得られた知見について述べる．

4.1. 実験方法と結果

本手法を実装したプログラムを実際のWeb APIオンラインドキュメントに対して適用し，情報が抽出できるかどうか試行した．対象としたオンラインドキュメントを以下に示す．

(1) はてなキーワードAPI（旧文書）
http://d.hatena.ne.jp/keyword/%A4%CF%A4%C6%A4%CA%A5%AD%A1%BC%A5%EF%A1%BC%A5%C9API（参照 2012-08-20）
(2) はてなキーワードAPI（新文書）
http://developer.hatena.ne.jp/ja/documents/keyword/apis/rest（参照 2012-08-20）
(3) CiNii API（論文検索）
http://ci.nii.ac.jp/info/ja/api/a_opensearch.html（参照 2012-08-20）
(4) 郵便番号検索API
http://groovetechnology.jp/webservice/zipsearch/index.html（参照 2012-08-20）

(5) 顔検出 API
http://kaolabo.com/webapi/spec（参照 2012-08-20）

はてなキーワードAPIに関しては，旧文書と新文書の2件が存在したので，その両方を利用した．予備実験では抽出する情報は，2章で示した7項目のうち，ベースURLとパラメータ名のみとした．実験では対象から人手で情報を抽出し，それを正解とする．その後本手法を実装したプログラムを以上の5件のオンラインドキュメントに対して適用し，抽出した情報を正解と比較した．今回適用したルールを以下に示す．

(ルール1)「http://」で始まるURLの記法に則っており，そのURLの記法に則った文字列以前の行に「ベースURL」や「Request URL」など，ベースURLに関係するキーワードが多くある場合，その文字列をベースURLとみなす．
(ルール2)「http://」で始まり，途中「?」で区切られる，Web APIのベースURLとしてよく見られるURLの記法に則っており，そのURLの記法に則った文字列以前の行に「ベースURL」や「Request URL」など，ベースURLに関係するキーワードが多くある場合，その文字列をベース URLとみなす．このルールによって発見されたベースURLはルール1より優先されることとする．
(ルール3) HTMLのtableタグで囲まれた表の中の要素の一つであり，かつそのtableタグ以前の行もしくは表内で「パラメータ」や「Parameter」などパラメータに関係するキーワードが多くある，かつ要素自体は英数字とURLとして利用可能な記号で構成されている文字列がある場合，その文字列をパラメータ名とみなす．
(ルール4) HTMLのdtタグで囲まれており，かつそれ自体は英数字とURLとして利用可能な記号で構成されている文字列である場合，その文字列をパラメータ名とみなす．

実験の結果として，3.1で示したすべてのオンラインドキュメントから正解と等しい情報を抽出することができた．実験におけるベースURLの抽出に関する結果を表1に示す．ベースURLは，複数の抽出候補に対して優先順位をつけて抽出している．表1では抽出候補の上位3件を優先度順に上位から示している．

4.2. 予備実験により得られた知見

今回の予備実験によりベースURLとパラメータ名が本手法によって取得可能であることがわかった．今回は情報を2項目しか抽出していないが，パラメータ名に関しては表形式で記述されているもの（HTMLのtableタグを用いているもの）はその表の中の要素を抽出することに成功しているので，パラメータに関する他の情報もパラメータ名と同じように抽出できる可能性が高い．問題としては，今回対象としたオンラインドキュメントは日本語であったため，パラメータ名やベースURLとの区別がつきやすかったが，英語のオンラインドキュメントは区別がつきにくい可能性があることが挙げられる．ベースURLに関しては，正解のURL以外のURLも誤って優先度が高いとみなされて抽出されたものもあったため，ベースURL候補の優先順位の付け方を検討する必要がある．また今回は対象としたオンラインドキュメントの件数が少なかったが，より対象件数を増やした場合には表記のゆれなど今回出なかった問題が発生する可能性がある．

表1. 予備実験結果（ベースURLの抽出）

抽出対象	人手による抽出(正解)	提案手法による抽出
(1)	http://search.hatena.ne.jp/keyword	http://search.hatena.ne.jp/keyword
		http://red3.hatena.ne.jp/adframe
		http://red.st-hatena.com/adframe
(2)	http://search.hatena.ne.jp/keyword	http://search.hatena.ne.jp/keyword
		http://d.hatena.ne.jp/keyword
		http://hatenadeveloper.g.hatena.ne.jp/keyword/ja/keyword/apis/rest
(3)	http://ci.nii.ac.jp/opensearch/search	http://ci.nii.ac.jp/opensearch/search
		https://register-ci.nii.ac.jp/auth/action/login
		https://register-ci.nii.ac.jp/userregist/userTypeSel.do
(4)	http://api.postalcode.jp/v1/zipsearch	http://api.postalcode.jp/v1/zipsearch
		http://groovetechnology.jp/sitemap.html
		http://uchiai.com/
(5)	https://kaolabo.com/api/detect	https://kaolabo.com/xmlrpc.php
		https://kaolabo.com/api/detect
		http://kaolabo.com/feed

5. 関連研究

Web API を用いたアプリケーション開発効率向上に関しては井上らの研究[4]がある．この研究ではREST 型と呼ばれる Web API においてのサーバ内でのデータ管理について，アプリケーションに求められる共通機能をそなえたデータベース・システムを開発している．この研究は Web アプリケーションの提供者側のサーバ内の処理機能の効率化であり，開発者側のプログラムライブラリについては言及していない．この他に開発者側の開発効率化として Riabov らの研究[5]や Lu ら[6]の研究がある．彼らの研究では複数の Web API を利用したマッシュアップの効率化について述べられており，Web API の仕様は既にプログラムとして実装されている前提での研究となっている．以上の研究とは異なり，本研究では Web API の機能を利用するためのプログラムライブラリを生成する．

HTML からの情報抽出という観点では片山らの研究[7]や池田ら[8]の研究がある．これらの研究ではHTML のタグ構造や類似度を利用し HTML の解析を行なっているが，対象となる HTML は Web API のオンラインドキュメントではなく，広範囲の HTML を対象としている．本研究の手法は Web API のオンラインドキュメントのみを対象とし，それに見られる規則性を利用して，特定の情報の抽出を行なうものである．

6. おわりに

本研究では，Web APIのオンラインドキュメントから仕様の情報を抽出し，Web APIのプログラムライブラリを自動で生成する手法を提案した．これによりWebアプリケーション開発者はプログラムライブラリの作成に関する開発上の手間がなくなり，開発効率が向上する．

今後は英語のWeb APIオンラインドキュメントから情報を抽出できるようにし，実際にどの程度の情

報が抽出可能であるのか評価する.

参考文献

[1] "ProgrammableWeb - Mashups, APIs, and the Web as Platform". Programmable Web.com. http://www.programmableweb.com/, (参照2012-06-27).

[2] "Twitter Libraries | Twitter Developers". Twitter. https://dev.twitter.com/docs/twitter-libraries, (参照2012-08-08).

[3] Sun Microsystems. "Web Application Description Language". World Wide Web Consortium (W3C). http://www.w3.org/Submission/wadl/, (参照 2012-07-23).

[4] 井上武, 朝倉浩志, 植松幸生, 佐藤浩史, 高橋紀之. 迅速なアプリケーション開発のためのWeb API データベースシステム. 電子情報通信学会技術研究報告. IN, 情報ネットワーク, 2010-01-28, 109(411), p. 97-102.

[5] Anton V. Riabov ほか. Wishful Search: Interactive Composition of Data Mashups. WWW. 2008, p. 775-784.

[6] Bin Lu ほか. sMash: Semantic-based Mashup Navigationfor Data API Network. WWW. 2009, p.1133-1134.

[7] 片山太一ほか. スプログ検出における HTML 構造の類似性の有効性の評価. 情報処理学会研究報告. データベース・システム研究会報告. 2009-11-13, 2009-DBS-149(19), p. 1-8.

[8] 池田 彰吾ほか. 繰り返し構造を考慮した Web ページの見出しの階層構造の解析. 情報処理学会研究報告. 情報学基礎研究会報告. 2008-03-28, 2008(34), p. 31-38.