Webアプリケーション Webサービス化 ラッパシステムの実装と評価

中野雄介[†1] 山登庸次[†1]
武本充治[†1] 須永宏[†1]

近年，インターネットをはじめとする分散環境で，様々なコンポーネントを連携させ，サービスを提供する試みがさかんになっている．これにともない，ネットワーク上でのコンポーネントの提供が増えている．しかし，このようなコンポーネントの数は十分とはいえず，連携サービスの作成者が意図したサービスを自由に作れるほどコンポーネントはそろっていない．そこで，Web アプリケーションが生成する HTML ドキュメントの中から，Web アプリケーションの処理結果の部分を抽出する手法を，Web アプリケーションの Web サービス化のためのラッパ生成に応用することを提案する．本手法は HTML ドキュメント内の各タグのネストの回数（深度）の変化に規則的なパターンがある部分を結果部分として抽出する．本手法を評価するための評価プログラムを実装し，既存の Web アプリケーションが生成する HTML ドキュメントから，結果部分の抽出を試みた．100 ドキュメント中 76 ドキュメントから結果部分を抽出することに成功した．加えて，本提案はラッパ作成者に対して十分なユーザビリティを提供できることを確認した．

Implementation and Evaluation of Wrapper System that Creates Web Services from Web Applications

Yusuke Nakano,[†1] Yoji Yamato,[†1] Michiharu Takemoto[†1] and Hiroshi Sunaga[†1]

This paper proposes a novel method to create Web Services by transforming existing Web applications in the Internet or enterprise networks. Recent technical trends in service coordination using various kinds of components in distributed service execution environments necessitate the provision of elemental service components, which typically take the form of Web Services. However, the number of Web Services is still small although that of Web applications is enormous, and so there should be a method that makes good use of such Web applications. Our approach is to convert HTML documents a Web application creates into a Web Service interface by extracting the operational result parts of the Web application. In this method, the depth of HTML tags, i.e., the number of nests of a given HTML document is analyzed to identify a regular patter and extract this part as an output. Through the evaluation of the wrapper system including this method, it is shown that correct result segments can be extracted from 76% of documents among a hundred. Also, it is clarified that this method has good usability for wrapper creators.

1. はじめに

近年，ネットワーク上のコンポーネントを組み合わせることで，新たなアプリケーションを作成する，Service Oriented Architecture（SOA）と呼ばれるコンセプトが注目を集めている．SOA では，アプリケーションを構成するコンポーネントのインタフェースとして Web Service[1] を，インタフェースの記述には Web Service Description Language（WSDL）[2] を用い，各コンポーネントは Simple Object Access Protocol（SOAP）[3] により連携することで，1 つのアプリケーションを実現する．これにより，アプリケーションを柔軟・低コストに作成・提供することが可能となる．

SOA のコンセプトをユビキタスコンピューティング環境におけるユーザへのサービス提供に応用しようとする研究がさかんになってきている．ユーザの状態により，コンポーネントを動的に組み合わせる，ユ

†1 日本電信電話株式会社 NTT ネットワークサービスシステム研究所
NTT Network Service Systems Laboratories, NTT Corporation

ビキタスサービスプラットフォームやそれらの上でのアプリケーションが実装されている[4)–8)]．しかし，ユビキタスサービスはユーザの状態に応じて最適なコンポーネントの組合せを動的に決定するため，多様なコンポーネントが必要となる．

一方，マッシュアップと呼ばれる簡易なアプリケーションの作成手法が注目を集めている．マッシュアップは Web2.0 のコンセプトに含まれ[9)]，近年のユーザによるアプリケーション作成を促進している．マッシュアップによるアプリケーション作成では，作成者は Web サービスなどのプログラムコンポーネントを発見し，それらをつなぎ合わせる．これにより，容易にアプリケーションを作成することができる．しかし，多くの作成者の意図するアプリケーションを実現するためには膨大なコンポーネントが必要となる．

以上のように，膨大で多様なコンポーネントが求められている．そこで，我々は既存の Web アプリケーション，特に検索系の Web アプリケーションをコンポーネントとして利用するためのラッパに関する研究を行ってきた．ラッパは Web アプリケーションのプロトコルとコンポーネントのプロトコルとを相互変換することで，Web アプリケーションをコンポーネントとして利用可能とする．Web アプリケーションとは Web ブラウザの利用者から HTTP でリクエストを受け，これに対する結果を HTML ドキュメントの形で Web ブラウザに返すことで働く Web 上のアプリケーションである．たとえば，ホテル検索や路線検索サイトなどが Web アプリケーションといえる．多くの Web アプリケーションは最終的な情報を表示する前に，ユーザに検索をさせ，候補を提示する．ユーザはこれらの候補の中から所望の候補を選択し，最終的な情報を閲覧する．そこで，ラッパは多くの Web アプリケーションをラップするために，このような検索系の Web アプリケーションを対象とする．

Web アプリケーション個別にラッパをコーディングするためには，多大な労力が必要である．このため，ラッパは Web アプリケーションごとに用意されたコンフィグファイルに沿って変換の処理を行う．たとえば，ホテル検索 Web アプリケーションをコンポーネント化する場合は，ホテル検索用のコンフィグファイルを記述すればよい．しかし，多くの Web アプリケーションをコンポーネント化するためには膨大なコンフィグファイルを記述する必要があり，この作業の削減が求められていた．

コンフィグファイルの記述において，最も時間のかかる記述が，Web アプリケーションのプロトコルからコンポーネントのプロトコルへの変換ルールである．このルールは Web アプリケーションが生成する HTML ドキュメント内のコンポーネントの戻り値にあたる部分を指定する．つまり，Web アプリケーションが生成する HTML ドキュメント内の，Web アプリケーションの処理結果の部分を自動抽出できれば，コンフィグファイルの記述コストを大幅に削減できる．

これまで，我々は Web アプリケーションが生成する HTML ドキュメントからの処理結果の部分の自動抽出に関して研究を行ってきた[10)]．しかし，このような技術のラッパ生成への応用に関しては十分に研究してこなかった．そこで，本稿では Web アプリケーションが返す HTML ドキュメントから Web アプリケーションの処理結果の部分を自動抽出するための手法を基に，Web アプリケーションの Web サービス化を実現するラッパシステムを提案する．その後，ラッパシステムの適用性とユーザビリティの評価結果を報告し，提案システムにより，ラッパ生成にかかる手間を軽減できることを示す．

2. 関 連 研 究

様々な Web アプリケーションが異なる形式の HTML ドキュメントを生成する．このため，すべての形式の HTML ドキュメントから結果部分を抽出するラッパの実現は不可能である．しかし，膨大な Web アプリケーションそれぞれにラッパを作成するためには多大なコストがかかる．そこで近年，Web ラッパの生成に関する研究がさかんになっている．

文献 11) で Web ラッパの生成に関する研究がまとめられている．Web ラッパの生成は教師つき学習と教師なし学習との 2 つに分類でき，教師つき学習である文献 12) からラッパ生成に関する研究がさかんになった．本稿で提案する手法は教師なしの学習に分類されるため，教師なし学習によるラッパ生成に関する関連研究をあげる．

文献 13) は Web アプリケーションのソースコードからラッパを生成する．成功率は高いが，ソースコードが必要となるため，市中にある多くの Web アプリケーションの Web サービス化はできない．

文献 14) は文字列の最大反復を発見することにより，結果部分を抽出するラッパを提案している．この手法は結果部分にイレギュラな結果（宿検索の結果ページ内の，お勧めの宿の結果だけタグの構造が違うなど）が含まれた場合，抽出に失敗すると考えられる．

また，近年実際に Web アプリケーションをコンポーネント化するためのラッパ生成ツールの提供が始まっ

ており[15]，このようなラッパに対して注目が集まっている．

3. ラッパシステム

我々のラッパは多くの Web アプリケーションをラップするために，コンフィグファイルを必要とする．コンフィグファイルを手で記述することができるが，多大な労力を必要とする．この課題を解決するためのラッパシステムを提案する．

3.1 ラッパシステムの概要

ラッパシステムの概要を**図 1** に示す．まず，ラッパ管理者はラップ対象の Web アプリケーションの処理結果の Web ページの URL をラッパ生成ツールに入力する．ラッパ生成ツールは入力された URL を保持すると同時に，この URL に対する HTML ドキュメントを取得，保持する．その後，ラッパ管理者はラッパ生成ツールを操作することで，保持された URL と HTML ドキュメントからコンフィグファイルを生成する．生成されたコンフィグファイルによってラッパは対象の Web アプリケーションをラップできる．このようにして，Web アプリケーションをコンポーネント化でき，様々なユビキタスサービスやマッシュアップアプリケーションで利用可能となる．

3.2 ラッパ

ラッパは Web アプリケーションのプロトコルと，コンポーネント（ここでは Web サービスとする）のプロトコル（ここでは SOAP とする）とを相互変換することで Web アプリケーションをコンポーネント化する．**図 2** にラッパの動作を示す．まず，ラッパはユビキタスサービスなどの Web サービスクライアントから SOAP でリクエストを取得する．その後，ラッパは SOAP で取得したリクエストを Web アプリケーションのリクエストに変換し，変換後のリクエストを Web アプリケーションへ送信する．リクエストを受け取った Web アプリケーションはそれに対するレスポンスを HTML ドキュメントの形でラッパへ送信する．HTML ドキュメントを受け取ったラッパは HTML ドキュメントから Web アプリケーションの処理結果の部分を抽出し，これを戻り値の形に成形する（ここでは XML ドキュメントの形に成形する）．その後，この XML ドキュメントを Web サービスの戻り値としてクライアントに返す．このようにして，ラッパは Web アプリケーションを Web サービス化する．以上の処理はラッパに設定されたそれぞれの Web アプリケーション用のコンフィグファイルに沿って行われる．

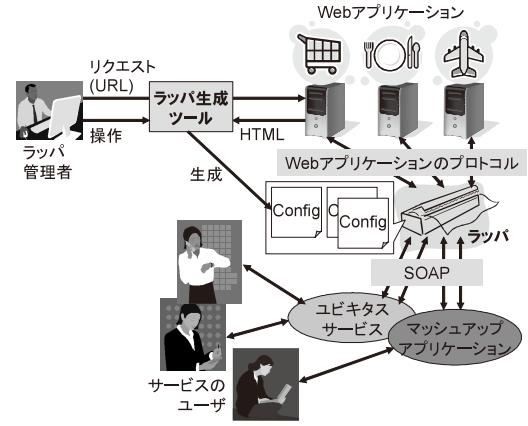

図 1 ラッパシステム概要
Fig. 1 Wrapper system

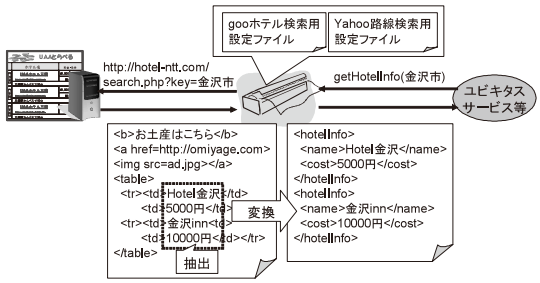

図 2 ラッパ動作
Fig. 2 Wrapper.

3.3 ラッパ生成ツール

ラッパ生成ツールはラッパ管理者とのインタラクションにより，ラッパのコンフィグファイルを生成する．検索系の Web アプリケーションは検索結果の HTML ドキュメントをデータベースなどから自動生成する場合が多い．このような HTML ドキュメントにはタグの規則的な繰返しパターンがあると考えられる．ラッパ生成ツールはこの繰返しパターンを手がかりとして，HTML ドキュメントから Web アプリケーションの処理結果の部分（検索結果など）を自動抽出することで，Web アプリケーションのプロトコルからコンポーネントのプロトコルへの変換ルールの生成を支援する．これにより，この変換ルールを含むコンフィグファイルの生成を支援できる．

4. 抽出手法

コンフィグファイルの生成のために，Web アプリケーションが生成する HTML ドキュメントから，Web アプリケーションの処理の結果の部分のみを抽出する手法の研究を行ってきた．本手法により，Web アプリ

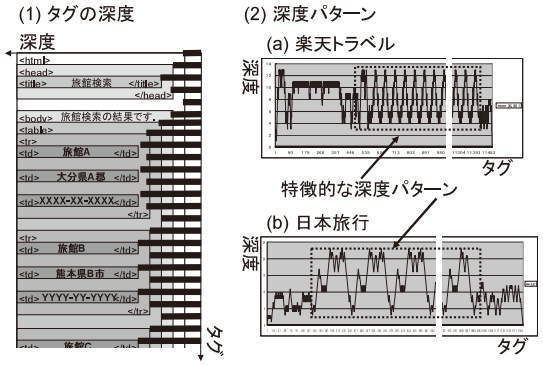

図 3 タグ深度とパターン
Fig. 3 Depth of tag.

ケーションのプロトコルからコンポーネントのプロトコルへの変換ルールの生成を支援できる．

4.1 抽出手法の概要

本手法は Web アプリケーションが生成する HTML ドキュメントに含まれるタグの深度変化の特徴を手がかりとし，結果部分を抽出する．タグの深度とはタグの入れ子の回数である（**図 3** (1)）．Web アプリケーションは機械的に HTML ドキュメントを生成する．このため，生成された HTML ドキュメントには繰返しのパターンが現れることが多い（図 3 (2)）．このような Web アプリケーションが生成する HTML ドキュメントの特徴を利用する．

本手法は人が Web アプリケーションの結果ページから結果部分を見つける方法と類似している．ある人が海外のホテル検索サイトの結果ページからホテルの検索結果を探し出すとする．この人は結果のページから繰り返し同じようなパターンが連続している範囲を探す．この方法はうまく機能し，この人はホテルの検索結果を見つけることができる．このような，人が自然に身に付けている方法を用いることで，様々な Web アプリケーションに適用可能な，柔軟な抽出を実現できる．

4.2 抽出手法の詳細

本手法は HTML ドキュメントに含まれるタグの深度が周期的に変化する部分を結果部分であると推定する．これを実現するために，深度データを波と考え，この波をスペクトル分析することで，周期的に変化している部分を発見する．発見された部分は波全体における大まかな位置を示すため，推定された位置と HTML 内のタグの情報とを用いて，HTML 上の正確な結果部分の位置を推定する．

HTML ドキュメントから結果部分を抽出するための手法を**図 4** に示す．本手法は 9 つのステップから構成される．各ステップについて説明する．

Step1. HTML ドキュメント内の各タグの入れ子の回数をカウントすることで深度を算出し，深度データを生成する．

Step2. HTML ドキュメント内の各部の深度特徴を抽出するために，算出された深度データを等分する．

Step3. FFT により，等分された深度データを周波数成分に展開し，周波数特性を各部の深度特徴とする．

Step4. HTML 内で，深度特徴が類似する部分が連続している区間（連続類似深度特徴区間）を発見する．

Step5. 発見された連続類似深度特徴区間で長さが最長となる区間に対応する HTML ドキュメント内の部分を推定結果部分として発見する．

Step6. 推定結果部分を補正し，正しい結果部分として抽出する．

Step7. 結果部分を検索結果 1 件ごとに分割する．

Step8. 分割後の検索結果が過分割されている場合，適切な分割を行うために再結合を行う．

Step9. 抽出された検索結果に，表の見出しのような結果以外の不要部分が含まれる場合，これらを除去する．

次からは各ステップに関して述べる．

4.2.1 タグ深度算出

Web アプリケーションが生成する HTML ドキュメント内の各タグの入れ子回数をカウントする．このとき，カウントされるタグは開始タグと終了タグのペアからなるタグである．つまり，
 や <img> など，終了タグをともなわないタグを無視する．これは，終了タグをともなわないタグは不規則に出現する傾向があるからである．たとえば，宿検索サイトの結果ページに，各宿に関するコメントが記述されていたとする．このようなコメントには，
 タグが多数含まれ，かつ，それぞれの宿のコメントによって，含まれる
 タグの数は異なると予想できる．このような，不規則に用いられるタグは，周期的な深度パターンを崩す可能性がある．

4.2.2 深度データ分割

HTML ドキュメントの各部の深度特徴を算出するために，深度データを等分する．これにより，部分ごとの深度特徴を調べることができ，波全体から周期的に深度が変化している部分を発見することができる．

分割のサイズは HTML ドキュメントの深度特徴により，可変とする必要がある．これは，**図 5** に示すように，HTML ドキュメントが，波長の長いパターンを持つ場合は分割サイズを広くし，波長の短いパターンを持つ場合は分割サイズを狭くする必要があるため

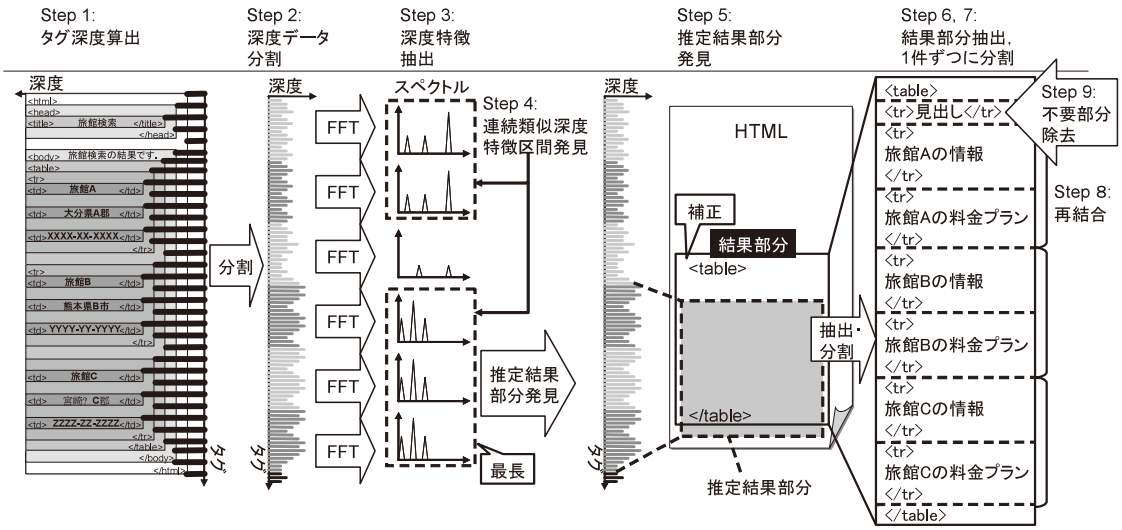

図 4 結果部分抽出手法
Fig. 4 Overview of method.

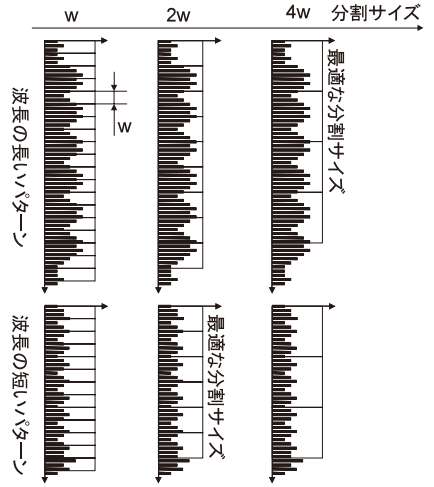

図 5 深度データ分割サイズ
Fig. 5 Division of depth data.

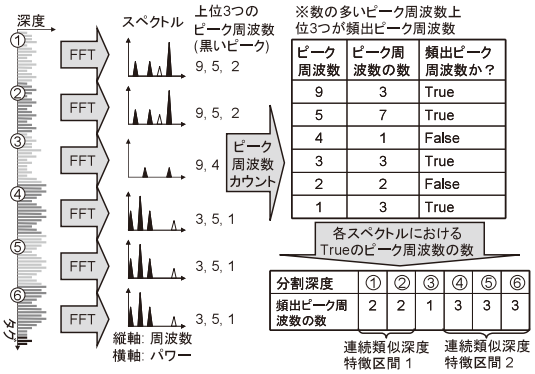

図 6 結果部分の推定
Fig. 6 Spectra classification.

である．

分割サイズは基準となるサイズを決めておき，まずそのサイズで分割を行い，結果部分の抽出を試みる．その後，基準サイズの 2 倍のサイズで分割し，再度抽出を試みる．このようにして，基準のサイズの 2 乗倍のサイズで分割を行い，抽出を試み，さらに 2 倍のサイズで分割し，抽出するという動作を繰り返す．分割サイズがあらかじめ定められた閾値を超えると，サイズの拡大を停止し，それまでに抽出された結果部分から最長のものを正しい結果部分と判断する．

このようにして，様々な Web アプリケーションに最適なサイズの分割を行うことで，適切に結果部分を抽出することができる．

4.2.3 深度特徴抽出

FFT により，分割された深度データそれぞれを周波数成分に展開し，各分割深度の深度特徴を抽出する．これは，FFT により抽出された深度特徴が連続して類似している部分，つまり，周期的な深度パターンがある部分を結果部分として抽出するためである．深度特徴は，FFT によって算出されたスペクトルから，高いピーク周波数の上位 3 つとする．**図 6** を用いて説明すると，分割深度 ① のスペクトルにはピークが 4 つあり，そのうち上位 3 つ（黒く塗りつぶされたピーク）の周波数は 9，5，2 であった．つまり，部分深度 ① の深度特徴は［9，5，2］となる．

4.2.4 連続類似深度特徴区間発見

深度特徴が類似している分割深度が連続している部分を発見する．この部分を連続類似深度特徴区間と呼ぶ．連続類似深度特徴区間の発見手法を図 6 で説明

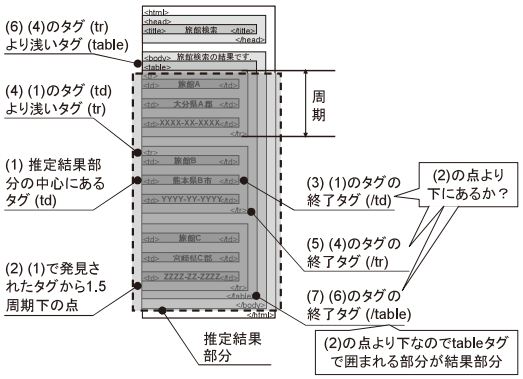

図 7 結果部分の抽出
Fig. 7 Finding result.

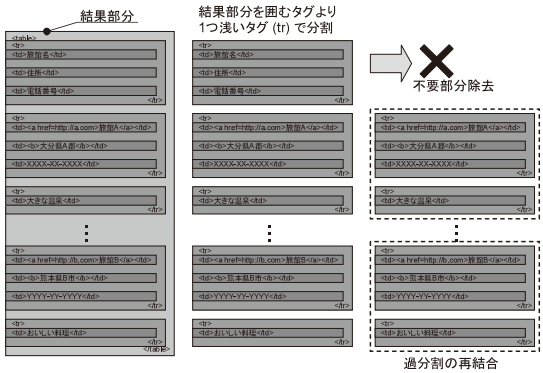

図 8 結果部分の分割
Fig. 8 Divide search result segment.

する．まず，全分割深度の深度特徴に含まれるピーク周波数を周波数ごとにカウントする（周波数 3 は ④，⑤，⑥ の深度特徴に含まれるので，3 回とカウントされる）．次に，それぞれのピーク周波数が頻出するかどうかを確認する．本手法では，カウント回数の多い上位 3 つのピーク周波数を頻出ピーク周波数とする（1 位は 7 回の周波数 5．2 位・3 位は 3 回の周波数 9，3，1）．その後，各分割深度に含まれる頻出ピーク周波数の数をカウントする（分割深度 ① の場合，ピーク周波数 9，5 が頻出ピーク周波数であるため，2 とカウントされる）．各分割深度の深度特徴に含まれるピーク周波数中，閾値以上の周波数が頻出ピーク周波数となる分割深度を連続類似深度特徴区間とする（閾値が 2 の場合，分割深度 ①，② は頻出ピーク周波数を 2 つ含むため，連続類似深度特徴区間となる）．

4.2.5 推定結果部分発見

複数の連続類似深度特徴区間が発見された場合，最長の区間を推定結果部分とする．これは，結果部分は Web アプリケーションが生成する HTML ドキュメントの広い部分を占めると仮定したためである．このようにして，発見された推定結果部分は HTML ドキュメント中での大まかな結果の位置を示す（図 6 の場合，分割深度 ④〜⑥ にあたる部分が推定結果部分となる）．

推定結果部分の発見と同時に，結果部分の波の周期を算出する．これは推定結果部分の深度データのスペクトル中，最も高いピーク周波数の逆数を求めることで算出される．この周期は結果部分に含まれる結果 1 つに含まれるタグの数と等しい（**図 7** の周期）．

4.2.6 結果部分抽出

推定結果部分は結果部分の大まかな位置を示すため，正確な結果部分を抽出する必要がある．これには，推定結果部分内にあるタグの情報を用いる．これにより，推定結果部分を補正し，結果部分を抽出することができる．

図 7 に結果部分抽出のための手法を示す．はじめに，推定結果部分の中心にあるタグを発見する（図 7 では td）．次に，発見されたタグから 1.5 周期下の点を発見する．その後，最初に発見されたタグの終了タグを発見する（図 7 では /td）．このとき，発見された終了タグが，先の点より上にあるとき，開始タグより浅いタグを発見し，新たな開始タグとする．そして，新たな開始タグに対する終了タグを新たな終了タグとし，新たな終了タグが先の点より下にあるかどうかを再度確認する．この処理を，終了タグが先の点より下になるまで繰り返す．終了タグが先の点より下になった場合，その終了タグ（図 7 では /table）とこれと対になる開始タグ（図 7 では table）とに囲まれる区間を結果部分とする．

4.2.7 結果部分分割

抽出された結果部分を結果 1 件ごとに分割する．たとえば，宿検索の Web アプリケーションの場合，宿 1 軒分の情報ごとに分割する必要がある．これにより，結果 1 件ごとを抽出できる抽出ルールを生成可能となる．

分割は結果部分を囲むタグ（**図 8** の場合 table タグ）よりも 1 つ浅いタグ（図 8 の場合 tr タグ）がある箇所で行われる．しかし，このような分割では，タイトルのような結果以外の不要部分が混入していたり，1 件分の情報が複数に過分割されたりするため（図 8 では宿の基本的な情報と，宿に対するコメントとで分割されている），これらの課題の解決が必要となる．

4.2.8 過分割の再結合

先のような課題を解決するために，過分割の再結合を行う．過分割の解決のために，再結合数を算出する（図 8 の場合，1 回再結合することで旅館の情報とコメントとを結合でき，宿 1 軒ごとの結果に分割できる）．

図 9 に再結合数の算出手法を示す．まず，各分割結果をグラフと考える．次に，各グラフどうしでルートからリーフまでの一致回数をカウントする（A，B 間では tr-td-TEXT が一致しているので 1 回）．各グラフで一致回数があらかじめ定められた閾値以上になっている類似分割結果（図 9 の灰色に塗りつぶされた部分）の数をカウントする（図 9 の類似数）．その後，類似数を縦軸，分割結果グラフを横軸とし，各分割結果グラフの類似数をプロットする．これを用い，プロットされた点から一次関数的に減少する点列を発見する．このとき，発見された点列の各点の間にある点の数を結合数とする（図 9 では各点の間に 1 つの点があるので，再結合数 1）．

4.2.9 不要部分の除去

不要部分が混入する課題を解決するために，抽出ルールで抽出できない分割結果を不要部分と判定する．抽出ルールは抽出すべき結果であると推定される 1 つのグラフとする．このグラフと類似する分割結果を抽出すべき結果であると判定し，それ以外を不要部分と判定する．

抽出ルールは先で発見された点列のメディアンとその近傍の結合数分のグラフとなる（図 9 では F，G のグラフが抽出ルールとなる）．これは，メディアンが抽出すべき結果である可能性が高いと仮定したためである．また，結合数分の抽出ルールが必要となるので，メディアンに近い，つまり，抽出すべき結果である可能性の高いグラフも抽出ルールとする．このようにして生成された抽出ルールを結果部分全体に適用することで，不要部分を除去することができる．

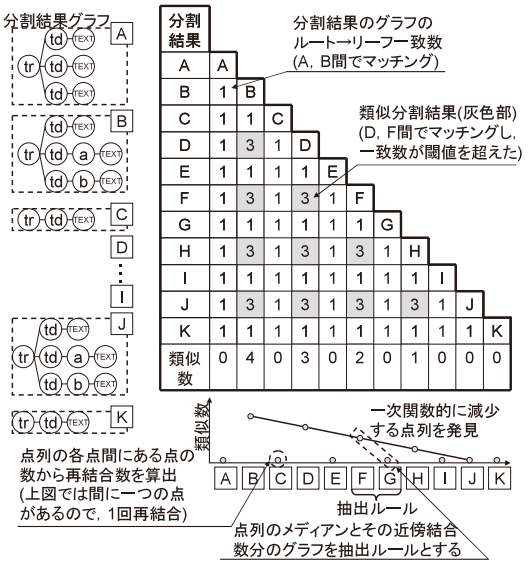

図 9 過分割の再結合
Fig. 9 Unite search results.

5. ラッパシステムの実装

本抽出手法を用い，ラッパシステムの実装を行った．ラッパシステムはラッパ生成ツールとラッパからなる．ラッパ生成ツールはラッパのためのコンフィグファイルを抽出手法に従って生成する．ラッパは生成されたコンフィグファイルに従って，Web アプリケーションを Web サービスとして利用可能とする．ラッパシステムにより，ラッパ管理者は簡易に Web アプリケーションを Web サービスとして利用可能とするラッパを提供することができる．次からはラッパシステムの実装に関して述べる．

5.1 ラッパ生成ツールの実装

ラッパ生成ツールを Web アプリケーションとして実装した．ラッパ管理者は Web ブラウザから本ツールを利用し，ラッパのためのコンフィグファイルを生成する．このとき，ラッパ管理者はウィザード形式の GUI に従って，本ツールを操作することでコンフィグファイルを取得する．以下に**図 10** を用い，ウィザードの各ステップにおけるユーザの操作を述べる．

① ラップ対象 Web アプリケーションの結果ページの URL を URL 入力フィールドに入力し，送信ボタンを押下する．

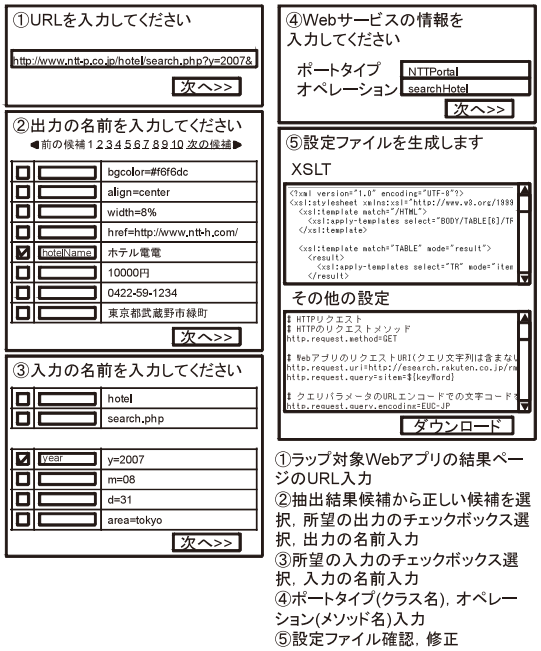

図 10 ラッパ生成ツール
Fig. 10 Wrapper tool.

② Web アプリケーションの処理結果 1 件が抽出結果として表示される（宿検索の結果ページ中の宿情報 1 件分など）．ラッパ管理者は正しい部分が抽出されていることを確認する．誤った抽出結果が表示された場合は，次の抽出結果の候補を確認する．正しい抽出結果を表示後，Web サービスの出力に含めたい文字列のチェックボックスを選択し，出力の名前を入力する（宿名が表示されている行のチェックボックスを選択し，出力名として hotelName を入力する）．

③ Web アプリケーションの結果ページの URL を構成する要素が表示される（URL のパスに含まれるディレクトリ名とクエリ文字列に含まれる各引数）．ラッパ管理者は Web サービスの入力としたい要素のチェックボックスを選択し，入力の名前を入力する（都道府県名が表示されている行のチェックボックスを選択し，入力名として area を入力する）．

④ Web サービスのポートタイプ（クラス名）とオペレーション（メソッド名）とを入力する．

⑤ 生成される XSLT ファイルとラッパ独自設定ファイルとの，2 種類のコンフィグファイルを確認し，これらのコンフィグファイルを書庫ファイルとしてまとめてダウンロードする．

5.2 ラッパの実装

ラッパをサーブレットとして実装した．ラッパサーブレットはクライアントから SOAP のメッセージを受け取ると，そのメッセージがどの Web サービス宛なのかを判断し，判断結果の Web サービスに対応するコンフィグファイルに従って，SOAP メッセージから Web アプリケーションのリクエスト文を生成する．その後，そのリクエストに対する HTML ドキュメントからコンフィグファイルに従って結果部分を SOAP メッセージに変換し，クライアントに返信する．

クライアントは Web サービスを用いるために WSDL を要求する．これにこたえるために，ラッパは WSDL を生成し，提供する．ラッパ管理者がコンフィグファイルをラッパの特定のディレクトリに保存すると，ラッパはこのコンフィグファイルを用いて WSDL を生成し，特定の Web ページから取得できる状態にする．

6. 評　　価

我々は抽出手法の抽出成功率を評価するために，測定プログラムを実装し，本プログラムを用い，既存の Web アプリケーションが生成する HTML ドキュメントから結果部分の抽出を試みた．この抽出結果を確認することで，本手法の抽出成功率を評価した．同時に，本手法を用いたラッパ生成ツールのユーザビリティ評価のための実証実験を行った．

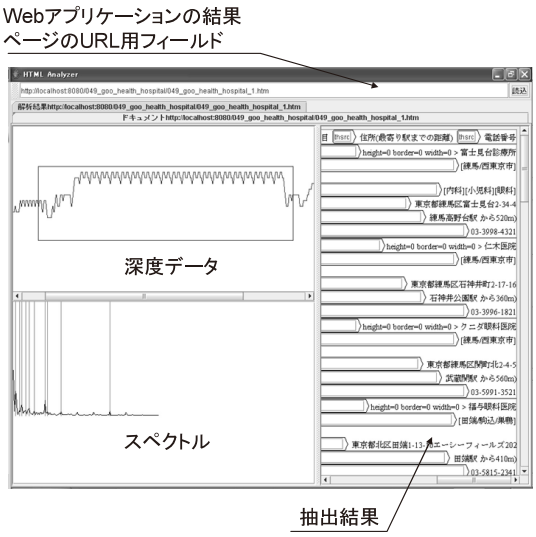

図 11 測定プログラム
Fig. 11 Experimental program.

6.1 抽出成功率評価手法

測定プログラムのユーザインタフェースを**図 11** に示す．評価者は Web アプリケーションが生成する HTML ドキュメントの URL を URL 入力用のフィールドに入力し，送信ボタンを押下する．すると，本プログラムが抽出対象となる HTML ドキュメントを取得し，先に説明した手法によって結果部分の抽出を試みる．その後，深度データ，スペクトルデータの可視化結果と，抽出結果とを表示する．

評価用の HTML ドキュメントは既存の Web アプリケーション 100 個が生成するドキュメントとした．対象とする Web アプリケーションは旅館検索などの検索サービスを提供するものであり，繰返しパターンがない商品の購入サイトなどを対象外とした．また，本手法は HTML ドキュメントの規則的な深度パターンを手がかりとするため，検索結果が 5 件以上となるドキュメントを収集した．**表 1** に各サイトからの収集ドキュメント数，抽出成功数，サイトに含まれる Web アプリケーションの例，その Web アプリケーションの URL，その Web アプリケーションからの抽出の成否をまとめた．

評価において，正しい抽出ルールが生成できる場合，抽出成功とした．つまり，抽出結果に Web アプリケーションの処理の結果が，1 件ごとに結合された状態で含まれている場合，この抽出結果を成功とした．

表 1 Web アプリケーション例
Table 1 Target web applications.

サイト名 (成功数/収集数)	サイトに含まれる Web アプリ	URL	抽出成否
楽天 (10/10)	楽天市場	http://www.rakuten.co.jp/	OK
	楽天トラベル	http://travel.rakuten.co.jp/kaigai/	OK
Yahoo (24/32)	Yahoo ニュース	http://headlines.yahoo.co.jp/hl	推定結果部分発見失敗
	Yahoo 中古車	http://autos.yahoo.co.jp/ucar/search/joken.html	OK
	Yahoo アパート検索	http://realestate.yahoo.co.jp/	OK
goo (22/33)	goo 車・バイクカタログ	http://autos.goo.ne.jp/catalog/index.html	OK
	goo 転職	http://job.goo.ne.jp/	OK
	goo アニメ	http://anime.goo.ne.jp/tvanime/index.html	OK
MSN (8/12)	MSN travel blog	http://4travel.travel.msn.co.jp/e/msn/travelogue/overseas/	推定結果部分発見失敗
Infoseek (6/7)	Infoseek ニュース	http://news.www.infoseek.co.jp/	OK
その他 (6/6)	スラッシュドット ジャパン	http://slashdot.jp/search/	OK

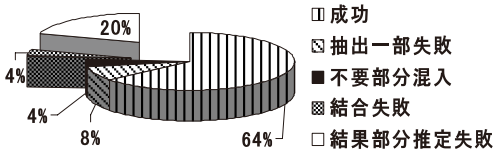

図 12 抽出成功率評価結果
Fig. 12 Success rate of extraction.

6.2 抽出成功率評価結果

以上の評価手法による抽出成功率評価結果を**図 12** に示す．64%の HTML ドキュメントに対して，過不足なく結果部分の抽出に成功している．8%のドキュメントは抽出すべき一部の結果が不要部分と誤認されている．また，4%のドキュメントは不要部分を含んだ．しかし，抽出ルールの生成には抽出結果のメディアンのみを用いるため，これら 12%から抽出ルールを正しく生成できると判断し，合計 76%（図 12 の一体になっている部分）のドキュメントに対して有効であると確認した．

結合に失敗している 4%の原因は，結合の順番を誤って判定してしまったことであった．これは，旅館情報の下に旅館のコメントを結合するべきだが，旅館のコメントの下に旅館の情報を結合してしまった，などを例としてあげることができる．これでは，正しい抽出ルールを生成できない．

また，結果部分の推定に失敗している 20%に関しては，HMTL ドキュメント中に複数の周期的な深度パターンが現れることが主な原因であった．たとえば，結果ページに大量のリンク集があった場合，この部分

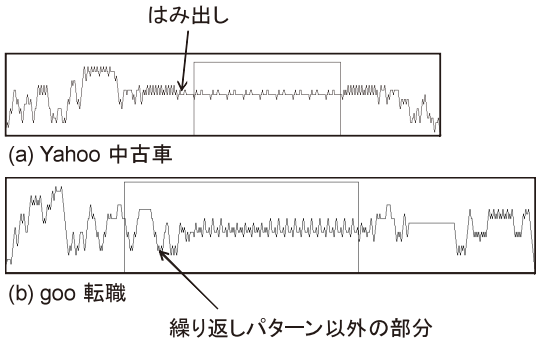

図 13 HTML ドキュメントからの抽出例
Fig. 13 Example of extraction.

に周期的な深度パターンが現れる．提案手法では，リンク集と Web アプリケーションの処理結果の部分とを見分けることはできないため，誤ってリンク集の部分を抽出してしまうことがある．

本手法が HTML ドキュメントの繰返しパターンを用いることで，結果部分を抽出できていることを示すために，実際に HTML ドキュメントのどの部分を抽出しているのかを調べた．測定プログラムは HTML ドキュメント内で，繰返しパターンが現れていると推定される区間を矩形で囲み，可視化する．可視化結果を見ると，**図 13** (a) の例では，繰返しパターンの一部が推定された区間からはみ出している．また，図 13 (b) の例では，繰返しパターン以外の部分も推定された区間に入っている．しかし，表 1 を参照すると，(a)，(b) ともに正しく抽出されており，本手法に含まれる補正の仕組みが正しく働いていることが分かる．

以上の抽出成功率の検証結果，本手法は既存の大半の Web アプリケーションからのラッパ生成が可能であるといえる．

6.3 ユーザビリティ評価手法

ラッパ生成ツールと既存のツールである Dapper とのユーザビリティを評価，比較することで，提案システムによるラッパ生成ツールの有効性を検証した．

Dapper はインターネット上の Web アプリケーションとしてラッパ生成サービスを提供する．多くの部分が JavaScript で記述され，主にクライアントのリソースを用いてラッパを生成する．Dapper の操作を以下に説明する．

1. ラップ対象 Web アプリケーションの検索ページの URL を Dapper に入力する．
2. Dapper が提供するブラウザに表示される検索ページ内の検索キーワード用テキストフィールドをマウスで指定し，Web サービスの引数名を入力する．
3. Dapper ブラウザから適当なキーワードを入力，結果ページを表示することで，ラッパ生成に用いる検索結果ページのドキュメントを 2 つ以上 Dapper に保持させる．
4. Dapper ブラウザに表示される検索結果ページ内の Web サービスの戻り値となる文字列をクリックし，戻り値名を入力する．

ラッパ生成ツールも Web アプリケーションとしてラッパ生成サービスを提供する．今回はサーバとして PentiumM 1.10 GHz，メモリ 752 MB のラップトップ PC を用い，100 Mbps の LAN 内でサービスを提供した．本ツールは多くの処理をサーバ側で行い，クライアント側はユーザとのインタラクションのみを行う．

ユーザビリティの評価には，System Usability Scale (SUS)[16] によるアンケート結果，キーロガーによるユーザの操作履歴を用いた．

被験者は Web サービス連携のためのプラットフォームの研究開発にかかわる研究者 10 名とし，あらかじめ決められた 2 つの Web アプリケーションのラッパを生成するという課題をさせた．まず，被験者には片方のツールを利用し，練習用の Web アプリケーションのラッパ生成をさせ，そのツールの利用に慣れさせた．次に，そのツールを用いて課題に取り組ませ，キーロガーによる操作履歴を収集した．その後，被験者は SUS によるアンケートへの記入を行った．このとき，先に利用したツールの利用経験が次のツールの評価に影響することを避けるため，1 つのツールの利用から次のツールの利用までに 1 日以上の時間を空けた．さらに，被験者を 2 つのグループに分け，片方のグループはラッパ生成ツールを先に利用し，もう片方は Dapper を先に利用した．

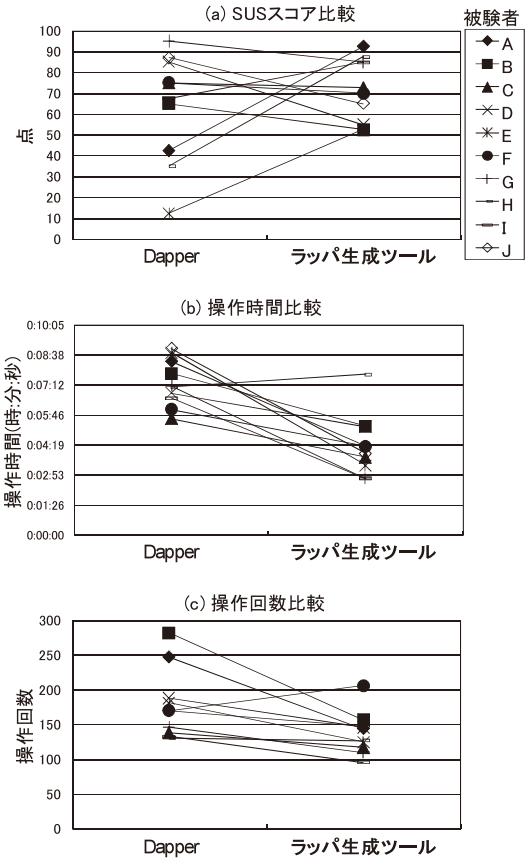

図 14 ユーザビリティ比較
Fig. 14 Comparison of usability.

6.4 ユーザビリティ評価結果

上記の評価手法によるユーザビリティの評価結果を**図 14** に示す．

6.4.1 SUS による評価結果

SUS による評価の結果，平均点は Dapper が 64，ラッパ生成ツールが 72 であった．しかし，図 14 (a) に示すとおり，被験者によって点数にばらつきがあり，平均点が高いラッパ生成ツールのユーザビリティが高いとはいえず，ラッパ生成ツールは Dapper と同程度のユーザビリティを提供すると考えられる．

6.4.2 操作履歴による評価結果

操作履歴を集計した結果，操作にかかった時間の平均は Dapper が 7 分 17 秒，ラッパ生成ツールが 4 分 17 秒であった．図 14 (b) に示すとおり，ほぼすべてのユーザがラッパ生成ツールの利用において，早く操作を完了している．これは，以下の要因が考えられる．

1. インターネット上で公開される Dapper と LAN

内で公開されるラッパ生成ツールとの通信時間や異なるユーザ数によるサーバ負荷などの差
2. 複数のドキュメントを必要とする Dapper と 1 つのドキュメントで済むラッパ生成ツールとの操作回数の差

1 に関しては，実装と運用の差であるため，本手法の優位性を示すことはできない．2 に関しては，自動抽出という，本手法の優位性を示している．図 14 (c) に示すとおり，ラッパ生成ツールの操作回数が少ないことが分かる．なお，操作回数にはマウスのクリック回数とキーのタイプ回数が含まれる．

以上のユーザビリティ評価の結果，ラッパ生成ツールは既存ツールと同程度のユーザビリティを持ち，ユーザにかかる負担を軽減させることができることを確認した．

7. おわりに

本稿では，Web アプリケーションを Web サービスとして利用可能とするラッパシステムに関して述べた．本システムに含まれるラッパ生成ツールはラッパを動作させるコンフィグファイルの生成支援を実現する．ラッパ生成ツールで用いられる，Web アプリケーションが生成する HTML ドキュメントからの結果部分抽出手法の評価の結果，既存の Web アプリケーションの 76%から結果部分を抽出でき，正しいコンフィグファイルを生成できることを確認した．加えて，ユーザビリティの評価の結果，提案手法は既存のラッパ生成サービスである Dapper と同程度のユーザビリティを提供すること，ラッパ生成にかかる操作を低減させることができることを確認した．

今後は，Web アプリケーション側のデザイン変更があった場合，自動的に設定ファイルを更新するラッパに関して研究を進め，ラッパのメンテナンスにかかるコストの軽減を目指す．

謝辞 本研究の一部は，平成 18 年度総務省「ユビキタスネットワーク認証・エージェント技術の研究開発」の研究助成によるものである．

参考文献

1) Web services web site. http://www.webservices.org/
2) WSDL: W3C Note, Web Services Description Language (WSDL) 1.1 (Mar. 2001). http://www.w3.org/TR/wsdl/
3) SOAP: W3C Note, Simple Object Access Protocol (SOAP) 1.1 (May 2000). http://www.w3.org/TR/soap/
4) Takemoto, M., Sunaga, H., Tanaka, K., Matsumura, H. and Shinohara, E.: The Ubiquitous Service-Oriented Network (USON) An Approach for a Ubiquitous World Based on P2P Technology, *Proc. P2P2002*, pp.17–21 (Sep. 2002).
5) Takemoto, M., et al.: Service Elements and Service Templates for Adaptive Service Composition in a Ubiquitous Computing Environment, *Proc. Asia Pacific Conference on Communications* (*APCC*), Vol.1, pp.335–338 (Sep. 2003).
6) Takemoto, M., et al.: A Service-Composition and Service-Emergence Framework for Ubiquitous Computing Environments, *SAINT2004 Workshop* (Jan. 2004).
7) Takemoto, M., et al.: A Context-Aware Content-Provision Service Based on a Ubiquitous Service-Oriented Network Framework, *Proc. SAINT Workshop* (Feb. 2005).
8) Sunaga, H., Takemoto, M., Yamato, Y., Yokohata, Y., Nakano, Y. and Hamada, M.: Ubiquitous Life Creation through Service Composition Technologies, *WTC2006* (May 2006).
9) O'Reilly, T.: What Is Web 2.0 (Oct. 2005). http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html
10) 中野雄介，山登庸次，武本充治，須永 宏：Web アプリケーションの結果ページからの結果部分抽出手法，*DEWS2007* (Mar. 2007).
11) 山田泰寛，池田大輔，坂本比呂志，有村博紀：WWW からの情報抽出—Web ラッパーの自動構築，人工知能学会誌，Vol.19, No.3, pp.302–309 (2004).
12) Kushumerick, N.: Wrapper Induction: Efficiency and Expressiveness, *Artificial Intelligence*, Vol.118, No.1-2, pp.15–68 (2000).
13) Huy, H.P.: Web Service Gateway—A step forward to e-business, *Proc. ICWS '04*, pp.648–655 (June 2000).
14) Chang, C.H. and Lui, S.C.: IEPAD: Information Extraction Based on Pattern Discovery, *Proc. 10th International Conference of World Wide Web*, pp.4–15 (2001).
15) Dapper. http://www.dappit.com/index.php
16) 山岡俊樹：人間工学講義，武蔵野美術大学出版局 (2002).

(平成 19 年 5 月 17 日受付)
(平成 19 年 11 月 6 日採録)

中野　雄介（正会員）

2005 年和歌山大学大学院システム工学研究科修了．同年日本電信電話株式会社入社．同社，NTT ネットワークサービスシステム研究所にて Web マイニング，ユビキタスコンピューティング，グループウェア研究に従事．2004 年情報処理学会第 66 回全国大会学生奨励賞受賞．電子情報通信学会会員．

山登　庸次（正会員）

2000 年東京大学理学部卒業．2002 年同大学大学院理学系研究科修了．2002 年日本電信電話株式会社入社．同社にて，Peer-to-Peer ネットワーク，ユビキタスコンピューティング，Service Delivery Platform 研究開発に従事．2005 年 4 月から 2007 年 3 月電子情報通信学会次世代ネットワークソフトウェア時限研究専門委員会幹事補佐．2007 年 2 月電子情報通信学会次世代ネットワークソフトウェア研究奨励賞受賞．2007 年 9 月電子情報通信学会通信ソサイエティ功労賞受賞．電子情報通信学会会員．

武本　充治（正会員）

1992 年東京大学理学部情報科学科卒業．1994 年同大学大学院理学系研究科情報科学専攻修士課程修了．同年日本電信電話株式会社入社．以来，NTT ネットワークサービスシステム研究所，NTT 未来ねっと研究所等において，分散コンピューティングシステムの研究に従事．1999〜2000 年 Massachusetts Institute of Technology，Laboratory for Computer Science において Visiting Scientist として研究に従事．2003 年より早稲田大学大学院情報生産システム研究科博士後期課程に在学中．1998 年電子情報通信学会学術奨励賞受賞．IEEE，電子情報通信学会各会員．

須永　　宏

1981 年東京工業大学工学部卒業．1983 年同大学大学院理工学研究科修了，博士（東北大学・情報科学）．NTT 研究所にて交換 OS，電話・パケット交換，VoIP，P2P，ユビキタス等長年情報通信システムの研究開発に携わる．ITU にてラポータ（SG7），TTC にて専門委員長（IP 電話）も務める．1999 年 ITU 協会賞著作賞受賞．電子情報通信学会，IEEE 各会員．