社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－SE－144 (23)
2004／3／19

フォーム自動記入のためのサービス統合フレームワーク実現方式

藤原克哉[†] 中所武司[††] 玉本英夫[†]

我々は，Webフォームへの記入を支援する自動記入エージェントの研究を行っている．これまでの自動記入方式では，記入する内容をあらかじめ登録できる範囲に記入対象が限られていた．本稿では，記入内容を外部Webサービスから自動抽出することで記入対象を広げたフォーム自動記入方式について述べる．記入内容の自動抽出には，可用性と網羅性を高めるために複数のサービスを利用する．本システムでは，複数サービスに問い合わせその結果を統合し利用するための，サービス統合フレームワークを実現した．本フレームワークはWebサービス変換機能，Webアプリケーション変換機能，サービス複合機能の3機能からなる．まずはじめに，異なるインタフェースを持つWebサービス群を透過的に扱うために，共通インタフェースへラッピングするWebサービス変換機能を実現した．次に，現在普及しているWebアプリケーションをWebサービスにラッピングするWebアプリケーション変換機能を実現した．さらに，これらの共通インタフェースを用いて複数サービスに問い合わせ，その結果を統合するサービス複合機能を実現した．最後にこのフレームワークを用いた記入内容自動抽出による自動記入エージェントを実現し，従来方式では自動記入できなかった書籍情報が自動記入できることを確認した．

Method of Web Services Integration for Filling Automatically in a Form

KATSUYA FUJIWARA,[†] TAKESHI CHUSHO[††] and HIDEO TAMAMOTO[†]

This paper describes the agent for filling in a form. However, values for filling in a form must be defined manually. Then, we developed the new system which extract the values automatically from external Web services. The system was built with the Web services integration framework which we developed. The framework includes Web services wrapper, Web applications wrapper, and dynamic service compounder. The Web services wrapper transforms various interfaces of services into a common interface. The Web applications wrapper transforms a HTML-based Web application into a Web service. The dynamic service compounder integrates these services.

1. はじめに

近年，ネットワーク接続されたパソコンの普及と共にオフィスの内外でエンドユーザが増加し，エンドユーザ主導によるシステム構築の必要性が高まっている．我々は，これまでインターネットにおける窓口業務システム構築を例題に，エンドユーザ主導型アプリケーション開発技法の研究を行ってきた．分散システム構築の分野では，プラットフォーム非依存のシステム間接続を実現する，SOAP，WSDL，UDDIの3つの標準技術からなるWebサービス技術が注目されている．Webサービスは，主に企業間取引において実用化が始まっておりセキュリティやトランザクションなどの関連技術が活発に検討されている．しかしながら，現状のWebサービスの利用形態は，UDDIを用いない静的結合モデルか，プライベートUDDIを用いた閉じた結合モデルであり，当初Webサービスに期待されていたオープンで動的なサービス連携という疎結合モデルの実現には至っていない．この主な原因の1つとして，不統一のサービスインタフェースが挙げられる．例えば，旅行予約システムが，ホテル予約サービスや航空券予約サービスを利用する場合，そのホテル予約サービスや航空券予約サービスが指定したインタフェースに合わせてシステムを構築することになる．現状ではこれらのインタフェースが同分野のサービスでも統一されておらず，インタフェース毎に個別開発する必要がある．最近ではWebサービスの普及に伴い分野毎にインタフェースの標準化が進んでおり，問

† 秋田大学工学資源学部情報工学科
Department of Computer Science and Engineering, Faculty of Engineering and Resource Science, Akita University
†† 明治大学理工学部情報科学科
Department of Computer Science, Faculty of Science and Technology, Meiji University

表 1　自動記入方式の特徴
Table 1　Features of methods for filling automatically in a form

自動記入方式	記入ルール	記入内容	適用範囲	記入精度
A: 過去の記入内容の再記入	固定の 1 つ	手動定義 (利用者が 1 回目に記入)	○ルールは任意のフォームに適用可 △同じフォームの 2 回目以降のみ	高
B: パターンマッチによる意味推論	手動定義 (専門家があらかじめ定義)	手動定義 (利用者があらかじめ登録)	○ルールは任意のフォームに適用可 △あらかじめ登録できる内容のみ	低
C: フォーム毎記入ルールの定義と利用	手動定義 (専門家があらかじめ定義)	手動定義 (利用者があらかじめ登録)	△ルール定義済みフォームのみ △あらかじめ登録できる内容のみ	高

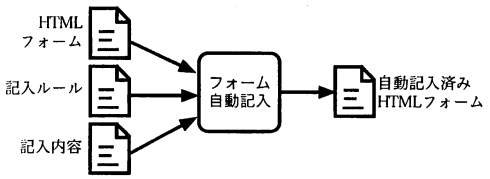

図 1　フォーム自動記入機能
Fig. 1　A model of filling automatically in a form

題の改善が期待されるが，異なるインタフェースで提供されるケースはなくならない．動的なサービス連携の実現のためには，まずこのインタフェースミスマッチの解消が必要である．本稿では，インタフェースの差異を吸収し透過的に扱うサービス統合フレームワークを実現することでこの問題の解決を目指す．また，サービス統合フレームワークを利用する例題として，記入内容を外部サービスから自動抽出するフォーム自動記入システムを構築し，その結果からサービス統合フレームワークの評価を行う．

2. フォーム自動記入技術

現在，インターネットにおいて WWW を利用したオンラインショッピングや銀行・証券取引，旅行予約等の窓口業務システムが既に広く実用化されている．これらの既存の WWW システムにおいて，窓口利用者は，氏名やメールアドレス等の同じような項目の入力を求められることが多い．フォーム自動記入は，このようなフォームにある決まりきった内容の記入を自動化することで，利用者のフォーム記入の手間を軽減することを目的とする．また，サーバ側のエンドユーザである業務の専門家にとっては，利用者による人為的な記入ミスを防ぐメリットがある．このフォーム自動記入機能は，現在，いくつかの方式が実現されている．フォーム自動記入機能の基本的なアーキテクチャを図 1 に示す．図のように自動記入機能は，電子フォームと，「どの入力項目にどの内容を入力するのか」という記入ルール，「その内容の具体的な値」である記入内容の 3 つの入力を元に，フォームの入力項目に自動記入し，記入済みフォームを出力する．現在利用されている自動記入方式は，表 1 に示す 3 種類に分類できる．

方式 A は，自動記入機能が，フォームの提出時に記入内容を保存し，次に同じフォームを利用する際に，前回の記入内容を自動記入する．汎用的な方式であり多くのフォームに適用できる．同じフォームには再度同じ内容を記入することが多いので有効であるが，初めて記入するフォームには適用できない．この方式は，多くの Web ブラウザで標準機能として実用化されている．

方式 B は，フォーム中の文脈から，何を記入する場所なのかを推論する．例えば，記入項目の近くに「名前」という単語がある（パターン）なら氏名を記入するというルールを定義しておくと，あるフォームの記入項目の前に「お名前：」と記されている場合，「名前」という単語とこのルールから氏名を自動記入できる．この方式は，汎用のルールを用いるため適用可能なフォーム数は多いが，推論の精度に限界があり記入間違いが避けられない [3]．記入できるフォームの種類は，利用者の氏名や住所などの記入内容があらかじめ用意できる範囲に限られる．

方式 C は，フォームのメタデータとしてそのフォームに何を記入すれば良いかという記入ルールを定義する．例えば，フォーム X の項目 Y には「氏名」を記入するというルールをフォーム毎に定義する．自動記入機能はそのルールに基づき氏名を自動記入する．記入ルールを専門家がフォーム毎に明示的に作成するため正確な記入ができるが，人手で作成するために対応するフォーム数が限られる．また，方式 B と同様に記入内容があらかじめ用意できる範囲に限定される．

各方式の実用例とその比較については文献 2) に詳しい．

3. 本研究の自動記入方式

先述のように，方式 A，C は記入精度が優れているが，適用できるフォーム数が少ない．方式 B は適用範囲が広いが，記入精度に劣る．自動記入機能には，記入精度が高く，適用範囲の広い方式が求められてい

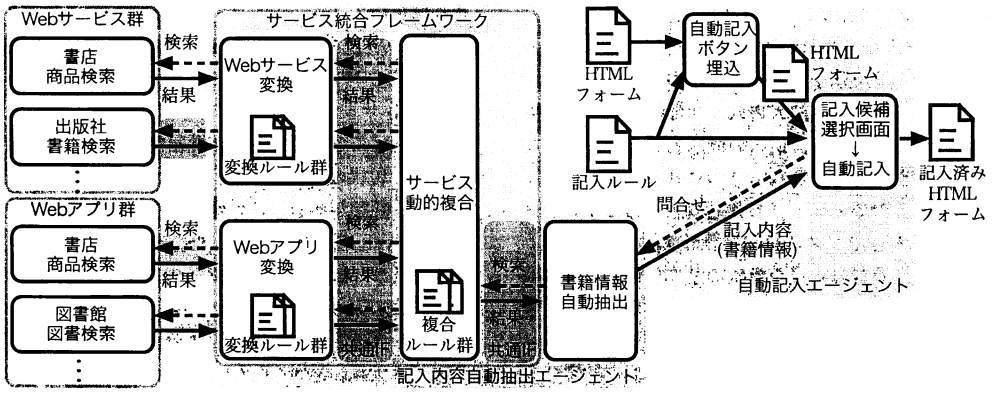

図 2　記入内容自動抽出のアーキテクチャ
Fig. 2　Architecture of extracting values for filling in a form

る．そこで我々は，3 方式を組み合わせる方式と，それぞれの方式を改良する方式について研究を行ってきた [1)~3)]．

本稿では，方式 C を改良した自動記入方式について述べる．方式 C は，記入精度は高いが適用範囲が限られていた．この方式の問題点は以下の 2 つである．

(1) フォーム毎に記入ルールを手動定義する手間がかかり，ルール定義済みフォームが限られる．
(2) 記入内容を利用者があらかじめ登録できる内容に限られる．

これまでに (1) の問題を解決するために，Semantic Web 技術 [4),5)] をベースとしたフォーム毎の記入ルール定義を行うルール記述言語を設計し，ルール定義を容易にするツール群を開発した [1)]．さらに，記入ルールを記入履歴から自動抽出する方式を実現することで手動定義を不要にして，適用可能なフォーム数を増加させた [2)]．

本稿では (2) の適用範囲があらかじめ記入内容を用意できる項目に限られる問題を解決するために，新たに外部の Web サービスから記入内容を自動抽出する自動記入方式を実現する．

4. 記入内容自動抽出による自動記入方式

4.1 実現方式の概要

本方式による自動記入の応用例として研究室の図書管理 Web システムの図書登録フォームに，書籍情報を自動記入するシナリオを設定した．これまでは，利用者がタイトルや著者名などを自ら書き写す必要があった．そこで以下のような自動記入方式により，利用者の手動記入の手間を解消する．まず利用者は，書籍の ISBN 番号のみを入力し自動記入ボタンを押す．次に自動抽出機能がその内容に該当する書籍情報を書店や図書館が提供している書籍検索機能に問い合わせ，結果を記入候補一覧として表示する．利用者が一致する内容を選択し確定ボタンを押すと自動記入される．最初に利用者が入力する情報は ISBN 以外のタイトルや著者の一部でも良い．

4.2 自動記入システムの構成

このような自動記入機能を実現するために，本システムは図 **2** に示すように 2 種類のエージェントからなる構成とした．自動記入エージェントは，フォーム，記入ルール，記入内容の 3 つの入力から，記入済みフォームを生成する．まず，自動記入ボタン埋込機能が，記入ルール定義に基づいて図書登録フォームに自動記入機能の呼出しボタンを埋め込む．利用者が，ISBN 番号等の書籍情報の一部を入力し自動記入ボタンを押すと，記入内容自動抽出エージェントに書籍情報を問い合わせ，記入候補選択画面を表示する．利用者が記入候補を選択すると，その内容を記入ルールに基づき図書登録フォームに自動記入し，記入済みフォームを表示する．記入内容自動抽出エージェントは，ISBN 番号等をキーワードに，外部サービス群に書籍情報を問い合わせ，その結果を記入内容候補として出力する．Web サービス変換と Web アプリケーション変換機能は，個々の外部サービスのインタフェースの違いを吸収する．サービス動的複合機能では，状況に応じてサービスを選択して問い合わせ，検索結果を複合する．図の Web サービス群は，書籍情報が検索できる SOAP ベースの Web サービスである．Web アプリケーション群は，書店などのホームページにある書籍検索機能で，Web ブラウザ向けに構築された一般の Web アプリケーションである．

4.3 対象ドメインのオントロジ設計

図2の記入ルールは，どのフォーム項目への記入ルールかを示す識別子，どのような内容を記入するかを示す情報の概念名，記入書式の詳細の3要素から構成される．記入内容は，情報の概念名とその具体的な値の2要素からなる．これら2つに共通する情報の概念名は，記入対象ドメインの共通概念であるオントロジ（語彙）としてドメイン毎に設計する．

共通オントロジは，最小単位の基本項目と，基本項目の組み合わせによる合成項目の2種類からなる．今回は対象とする書籍情報について，記入対象のフォーム項目と，記入内容を抽出するサービス群の検索結果の形式から，共通する特徴を抽出し，著者，訳者，タイトル，サブタイトル，シリーズ名，出版社，出版年月日，ISBN番号，言語，ページ数，定価の11の基本項目からなるオントロジを定義した．さらに，著者と訳者を組合わせた項目などよく利用される約30の合成項目をあらかじめ定義した．

4.4 サービス統合による記入内容自動抽出方式

本システムの記入内容自動抽出機能は，記入内容の書籍情報を書店や図書館などの書籍検索機能を提供しているWebサービスやWebアプリケーションから自動取得する．このような外部サービスへの問い合わせを実現するために，以下の3つの特徴を持つサービス統合フレームワークを構築する．

(1) 複数サービスの動的複合
(2) Webサービス変換機能
(3) Webアプリケーション変換機能

まず，本システムの記入内容抽出方式は，複数の外部サービスを実行時に選択し組み合わせて利用することとした．(1)の動的複合機能により複数のサービスに書籍情報を問い合わせ，結果を統合する．本システムの記入精度は，記入内容の網羅性と記入ルールの正確さにより決まる．網羅性は，正しい記入内容が取得できる比率で表される．複数のサービスを組み合わせることで網羅性が向上するメリットがある．例えば，新刊の書籍は図書館に登録されておらず，書店には登録される傾向があるなど，各サービスに特色の違いがあるためである．また，複数サービスを利用することで，耐故障性の向上やサービス提供側への負荷分散のメリットもある．

(2)のWebサービス変換機能では，個別に異なるWebサービスのインタフェースを，共通インタフェースに変換する機能を実現することでインタフェースの違いを吸収する．サービスの動的複合機能からは共通インタフェースでWebサービス変換機能に問い合わせれば良い．

(3)のWebアプリケーション変換機能では，現在普及しているHTMLベースのWebアプリケーションを，Webサービスへラッピングする．本方式では多くのサービスを利用するほど網羅性が高まると考えられるが，現状のWebサービスは数が少なく選択肢が限られる問題がある．この変換機能により既に普及しているWebアプリケーションに対応することで，利用可能なサービス数を増やせるメリットがある．

サービス統合フレームワークは，図2に示すように，これらの3機能に対応する，サービス動的複合，Webサービス変換，Webアプリケーション変換の3つのサブフレームワークからなる構成とした．

5. サービス統合フレームワークの構築

5.1 共通インタフェースの開発

まずはじめに，書籍検索機能の共通インタフェースを構築した．共通インタフェースは，RDFの問い合わせ言語であるRDQL(RDF Data Query Language)[6]をベースに開発した．RDQLからの主な変更点は，多くの検索サービスで用いられている文字列の部分一致検索機能の追加である．共通インタフェースは，要求メッセージと応答メッセージからなる．メッセージ内の書籍情報は4.3節で定義したオントロジで表現する．

要求メッセージは，「ISBNが〜である書籍」といった検索条件と「タイトル，著者名，ISBN番号を検索」など結果として得たい書籍情報の項目を指定して問い合わせる．要求メッセージの検索条件の記述例を図**3**に示す．応答メッセージは，RDF形式で「候補1の書籍のタイトルは〜，著者は〜」という書籍情報の項目の具体的な値が，検索条件に一致した数だけ返される．

次に，共通インタフェースから利用できる各サービスの機能の詳細を定義するために，RDFベースのサービスメタデータ記述言語を開発した．定義するメタデータは以下の3つである．

(1) 利用するオントロジ
(2) 利用できる検索条件，検索できる対象項目
(3) 変換ルール記述

図**4**にサービスメタデータ定義例の一部を示す．(1)は4.3節で定義した書籍情報のオントロジを指定する．オントロジを入れ替えることでこのフレームワークは他の分野に対応できる．(2)は変換元のWebサービス，Webアプリケーションによって異なる，利用できる検索条件と検索できる対象項目の2つの機能を定義する．例えば，書店では検索結果に価格が含まれるが，図書館の検索結果には価格が含まれないなど，各

```
SELECT ?x, ?y, ?z
 WHERE (?b book:title ?x)
       (?b book:author ?y)
       (?b book:ISBN ?z)
 AND ?z CONTAINS "1-234-56789-X"
 USING book FOR <http://wwhww.org/ontology/book/1.1#>
```

図 3 検索条件の記述例
Fig. 3 An example of the search condition

```
<rdf:RDF xmlns="http://wwhww.org/1.1#"
 xmlns:book="http://wwhww.org/ontology/book/1.1#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <QueryService rdf:about="http://www.lib.akita-u.ac.jp/">
  <targetType
rdf:resource="http://wwhww.org/ontology/book/1.1#Book"/>
  <conditionTemplate>
   <ContainsCondition>
    <targetProperty
rdf:resource="http://wwhww.org/ontology/book/1.1#ISBN"/>
   </ContainsCondition>
  </conditionTemplate>
  <resultTemplate>
   <book:Book>
    <book:title/><book:author/><book:ISBN/>
   </book:Book>
  </resultTemplate>
  <rule>
    ... 中略 ...
  </rule>
 </QueryService>
</rdf:RDF>
```

図 4 サービスメタデータ記述例
Fig. 4 An example of the service metadata description

サービスには機能に違いがある．(3) は Web サービス変換ルールと Web アプリケーション変換ルールの記述部である．それぞれのルール定義方式の詳細は，後述する．

5.2 サービスの動的複合方式

サービス複合の目的は，サービスを組み合わせることで，全体としての網羅性や可用性を高めることである．サービスの動的複合では，(a) 利用するサービスを選択し，(b) 選択したサービスに同時に問い合わせ，(c) 結果を複合する，3 つの処理を行う．

(b) における複数サービスへの問合せは，逐次実行と並列実行の 2 方式が考えられる．逐次実行の場合，必要な結果が得られなかった場合にのみ次のサービスに問い合わせれば良いのでサービス提供側やネットワークなどへの負荷が小さい．ただし，利用者の待ち時間が長くなり実用面で劣る．並列実行の場合，負荷が大きいが，待ち時間は最大でも最も遅かったサービスの待ち時間で済む．本方式では，利用者の待ち時間が少ない後者の並列方式を用いる．さらに，応答時間を短縮するために，応答に 30 秒以上かかっているサービスは待たずに，先に応答のあった結果のみを利用することにした．30 秒の時点で必要な結果が得られていないの場合にのみさらに応答を待つ．

この並列方式では，単純に全てのサービスに問い合わせる方式も考えられるが，網羅性は高い反面，サービス数が増えるとネットワークや処理の負荷が高くなる問題がある．そのため，(a) においてあらかじめ利用すべきサービスを選択する．最小限のサービスの組み合わせで高い網羅性が実現できるサービス選択方式が求められる．

本システムは，各サービスの以下の 3 つの特徴から最適なサービスを選択することとした．

(1) サービスの検索機能
(2) サービスの網羅性
(3) サービスの平均応答時間

1 番目に，各サービスの検索機能の違いからサービスを選択する．図 2 の書籍情報自動抽出機能からは，自動記入対象のフォームにより異なる検索条件で問い合わせが行われる．例えば，タイトルと著者，ISBN 番号の記入欄があり，タイトルと著者の一部を入力して自動記入ボタンが押された場合，入力されたタイトルと著者のそれぞれに部分一致する条件で書籍のタイトルと著書，ISBN 番号を検索する．一方，前節で述べたように，書籍検索の各サービスは検索機能に違いがある．そこで本システムでは検索条件に対応したサービスを，必要なサービスが少なくなるように以下の手順でサービス選択の優先順を決定する．

(a) まず必要な項目を全てまとめて検索できるサービスを選ぶ
(b) 必要な項目が全て検索できるようなサービスの組み合わせを選ぶ

例えば，タイトルと価格，ISBN 番号の記入欄があり，ISBN 番号を入力して自動記入ボタンが押された場合，まず ISBN 番号からタイトルと価格を検索できるサービスを選択すれば良い．それでも不足の場合は，ISBN 番号からタイトルを検索できるサービスとを ISBN 番号から価格を検索できるサービスの組み合わせを用いる．

2 番目に，各サービスの網羅性の違いから，組み合わせた網羅性が高くなるようにサービスを選択する．まず，あらかじめ各サービスの網羅性を調べておく．基準となる書籍 50 冊の標本集合を用意して，それぞれのサービスでその書籍を検索した結果，書籍情報が正しく得られたものをヒット集合として記録しておく．そして，各サービスのヒット集合の和集合が最も大きくなる組み合わせで，なるべくサービス数の少なくな

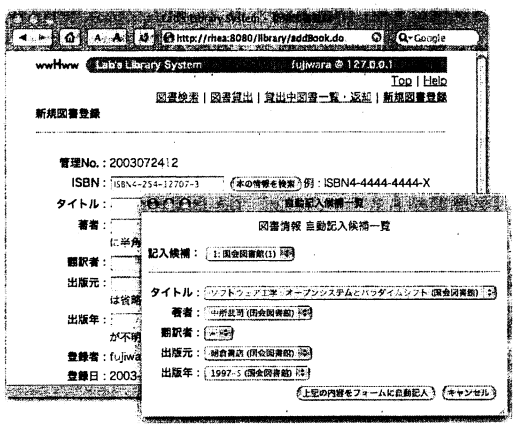

図 5 自動記入候補の選択画面例

Fig. 5 A window for selecting values for filling in a form

るものを選択する．なお，ヒット集合の和集合は，最大の場合に標本集合と一致する．同条件のサービスの組み合わせ候補が複数ある場合は，次の (3) により利用する組み合わせを選択する．

3 番目は各サービスの平均応答時間の違いからサービスを選択する．まず，最も早いサービスを単純に選ぶ方式が考えられるが，応答時間が短いものの，そのサービスのみに負荷が集中する問題がある．そこで，平均応答時間の早いサービスほど選択回数が多く，遅いサービスほど少なくなるように，各サービスの平均応答時間の逆数を全ての比較対象サービスの平均応答時間の逆数の総和で割った値の頻度で選択することとした．

5.3 Web サービス変換方式

Web サービス変換は，Web サービス発見，要求メッセージ変換，応答メッセージ変換の 3 つの機能からなる．Web サービス発見では，変換ルールの対応している WSDL を持つサービスを UDDI から検索し接続の準備をする．要求メッセージ変換では，書籍情報の検索条件を共通形式からサービス固有形式の要求メッセージへ変換する．要求メッセージのひな形と，ひな形への検索条件の埋め込みルールを定義しておく．応答メッセージ変換では，サービス固有形式の書籍情報から共通形式へ変換する．応答メッセージからの書籍情報の切り出しルールを定義しておく．

これらのメッセージの変換ルールの定義には，XML 文書変換言語の XSLT を用いる．

5.4 Web アプリケーション変換方式

Web アプリケーション変換では，Web アプリケーションを共通インタフェースの Web サービスとしてラッピングする．本システムでは，Web アプリケーションへの要求を，人が Web ブラウザを操作するのと同じ手順で行う．まず，Web アプリケーション変換のためのルールを，リンクやボタンのクリック，フォーム項目への記入などからなる Web ブラウザの操作手順と，Web ページからの情報の切り出し手順により定義する．そして，GUI を持たない仮想 Web ブラウザを用いてこれらのルールを実行する．変換ルールの定義には，Web アプリケーションの操作手順は手続き的な要素が大きいため，スクリプト言語の JavaScript を用いて記述することとした．以下に図書館図書検索の問い合わせ手順例を示す．

(1) メニューページを開く
(2) 検索フォームへのリンクをクリック
(3) 検索フォームに検索条件を入力
(4) フォームの送信ボタンをクリック
(5) 結果一覧の 1 つ目の結果へのリンクをクリック
(6) 結果ページの切り出し
(7) 次の結果について (5) から繰り返し

応答時間の短縮のため，(1) のメニューページは利用する検索フォームへのリンク部分が常に同じなのでキャッシュしておいたものを用いる．また，(5) から先は並列に実行する．なお，この例では (2) の検索フォームはセッション管理用の毎回異なる ID が埋め込まれているためキャッシュを用いることはできなかった．

6. 応用システムの実装

図 2 に示すように，サービス統合フレームワークを用いた記入内容自動抽出エージェントと，自動記入エージェントからなる自動記入システムを構築した．図の自動記入エージェントは HTTP プロキシとして動作する．本システムは，利用者端末または組織内のプロキシサーバで稼働させることを想定している．主な利用手順は以下の通りである．

(1) Web ブラウザが取り寄せている Web ページに対応する記入ルールがある場合，フォームに自動記入ボタンを埋め込み，Web ブラウザに渡す．
図 5 左上の新規図書登録フォームにある ISBN 記入欄の右の「本の情報を検索」ボタンが埋め込まれた自動記入ボタンである．
(2) 利用者がフォームの一部を入力し自動記入ボタンをクリックすると，自動記入エージェントが呼び出される．
(3) 自動記入エージェントは，記入内容自動抽出エージェントに一部入力された内容を渡し，書籍情報を問い合わせる．
(4) 記入内容自動抽出エージェントは複数のサービス

から書籍情報を検索し結果一覧を返す.

(5) 自動記入エージェントは，図5右下に示すような自動記入候補の選択画面を，第1候補を選択した状態で表示する.

(6) 利用者が記入候補から適切なものを選択し記入ボタンを押すと，自動記入済みのフォームがWebブラウザに表示される.

一連の手順で実際に研究室の図書登録フォームで自動記入できることを確認した．また，新たに図書館の図書リクエストフォームの記入ルールを作成し，自動記入できることを確認した.

本システムの実装には，Java 1.4.2と，SOAPに対応したWebサービス用フレームワークであるApache WSIF(WebService Invocation Framework) 2.0を用いた．また，Webアプリケーション変換の仮想Webブラウザ機能の実現にはWebアプリケーションテスト用フレームワークHttpUnit 1.5.3を用いた.

7. 実験と結果

7.1 実験方法

記入内容自動抽出から自動記入までの一連の流れを確認するために実験を行った.

書籍情報を検索する外部サービスには，実際に公開されていて利用可能な以下の4つを選んだ.

- 書店A： オンライン書店の商品検索機能
- 書籍検索B： 書籍データベースの検索機能
- 図書館C： 国会図書館の図書検索機能
- 図書館D： 大学図書館の図書検索機能

AはWebサービスである．B,C,DはWebアプリケーションである.

まず，実験のための準備として，これら4つを対象に変換ルールの作成を行った．さらに，各サービスの網羅性を調べるために，標本集合として，なるべくジャンル，出版年が重ならない50冊の和文書籍を選んだ．そして，各サービスについて標本集合の書籍を検索し，ヒット集合を求めた．また同時に，正しい結果が得られた時の応答時間を計測し，サービスの平均応答時間を求めた．なお，曜日や時間帯による混雑などの影響で外部サービスの応答時間が変化する可能性があるため，曜日，時間帯をずらして5回に分けて計測した.

そして，このシステムを用いて実際に書籍情報がフォームに自動記入できることを確認する．確認のために研究室の図書管理システムの図書登録フォームに50冊の図書情報を自動記入した．記入対象の50冊の書籍は，研究室の図書から無作為に選んだ．記入対象の図書登録フォームには，タイトル，著者，訳者，出版元，出版年，ISBN番号の6つの記入欄がある．実験では，対象の書籍のISBN番号のみを記入して、自動記入ボタンを押し、その結果、正しい記入内容が抽出されて残りの記入欄に自動記入できるかを確認した.

表2 各書籍検索サービスの網羅率
Table 2 Cover Rates of Services for Searching Books

サービス提供元	網羅率	検索できない書籍の内訳
書店A	86%	出版社変更，古い版，絶版など
書籍検索B	98%	最近出版された書籍
図書館C	98%	最近出版された書籍
図書館D	10%	専門書など多数

7.2 実験結果

結果として，記入対象の研究室の50冊全てについて，正しい書籍情報を自動抽出し，自動記入することが出来た．5.2節の選択方式で選ばれたサービスの組み合わせは，A,Bの組み合わせが34回，A,Cの組み合わせが16回であった.

8. 考察

8.1 網羅性によるサービス選択方式

実験で求めた各サービスのヒット集合の内訳を表2に示す．各サービスの網羅率は，標本集合に対するヒット集合の書籍数の割合である．結果として，書店Aで検索できなかった14%は全て古い版など販売されていない本であった．書籍検索Bと図書館Cで検索できなかった2%は出版されたばかりの新しい本であった．このように，サービス毎に網羅性に違いがあるため，サービスを組み合わせることで網羅性を向上できる．今回検索できなかったAの14%とB,Cの2%は互いに重複しないため，AとBや，AとCの組み合わせの網羅率は100%となる．一方，BとCの2%は同じ書籍であり，BとCの組み合わせでは網羅性の向上は期待できない．実験では，5.2節で述べた網羅性による選択方式により，網羅率が100%となる，AとBまたは，AとCの組み合わせが選択されることが確認できた.

なお，今回のサービスでは網羅性の違いが書籍の出版日に関連していることから，正確な結果を得るためにヒット集合の調査は同時期に行う必要があると言える.

8.2 応答時間によるサービス選択方式

実験で求めた各サービスの変換処理の平均応答時間を表3に示す．表の外部問合せが，Webサービスや Webアプリケーションからの応答待ち時間である．結果として，平均応答時間ではAのWebサービスが，

表 3 変換処理の平均応答時間
Table 3 Average Response Time of Transforming

サービス提供元	全体 (T1+T2)	外部問合せ (T1)	変換処理 (T2)
書店 A	2.1 秒	1.8 秒	0.3 秒
書籍検索 B	4.6 秒	1.4 秒	3.2 秒
図書館 C	23.2 秒	18.9 秒	4.3 秒
図書館 D	12.5 秒	10.3 秒	2.2 秒

全ての Web アプリケーションよりも優れていた．理由としては，1 回の検索処理が，Web サービスの方は 1 回の要求と応答で完了するのに対して，Web アプリケーションはいずれも 3 回から 4 回の要求／応答を行っていることが挙げられる．Web アプリケーション変換については，5.4 節で述べたように，検索フォームなどの固定のデータはキャッシュを利用することで要求／応答回数を減らす工夫をしている．さらに利用者の待ち時間を少なくするには，平均応答速度の速い Web サービスや Web アプリケーションを多く用意することと，現在は 30 秒としている最大応答待ち時間をさらに短くすることが考えられる．

8.3 記入候補の絞り込み

本方式では，利用者が書籍情報の一部を入力してから，その入力内容を検索条件として記入内容の書籍情報を検索する．書籍毎に固有の ISBN 番号と，それ以外のタイトルや著者名による検索では，結果が大きく異なった．著者名やタイトルの一部による検索では，同じ検索語を含む書籍が多数存在する場合もあり，利用者は記入候補として表示された不要な結果を含む検索結果の中から目的の書籍を探し出す必要がある．一方，ISBN 番号を入力して検索した場合は，各サービスでの検索結果が 1 件に絞り込まれるため，実験では第 1 候補に目的以外の書籍情報が選ばれることはなかった．なお，同じ ISBN 番号で複数の検索結果が一致するものもあるが，今回の実験に利用した書籍では，図書館で同じ書籍の古い版と新しい版が個別に登録されていた 1 件のみであった．

検索条件を増やしたり，工夫することで検索結果を絞り込むことが出来るが，記入すべき内容が増えると，自動記入のメリットを損ねることになる．今回の書籍情報の自動記入では，ISBN 番号を入力して自動記入することを推奨することとした．

8.4 変換可能サービスの動的発見

図 2 の Web サービス群はサービス変換ルールが対応している WSDL に一致するサービスを，UDDI により動的に発見し利用することができる．ただし，先述のように現状では共通のインタフェースを持つ Web サービスは少数で，動的発見のメリットが少ない．今回の実験で用いた Web サービスも固有の WSDL を用いており，UDDI で検索しても 1 つのサービスしか発見されない．今後は，さらに変換ルールも登録・公開できるようにした，UDDI ベースのディレクトリサーバを実現することで，変換可能なサービスも動的に発見し利用できる方式を検討していく．

9. ま と め

本研究では，異なるインタフェースを持つ Web サービスや Web アプリケーションを統合して透過的に扱うサービス統合フレームワークを構築した．さらに，そのフレームワークを用いて外部サービスから記入内容を自動抽出する自動記入方式を確立した．Web サービスの普及に伴いインタフェースの標準化は進むが，今後も同分野のサービスが異なるインタフェースで提供されるケースがなくなることはない．本研究の Web サービス統合フレームワークの基本概念は，フォーム自動記入に限らず広く Web サービスを利用したアプリケーション開発に適用可能と考えられる．また，Web アプリケーションのラッピング機能は，既存のシステムを変更することなく Web サービス化できるため，コストがかかる等の理由で Web サービス化されない分野に有効である．

参 考 文 献

1) 藤原克哉, 中所武司: 窓口業務アプリケーションフレームワーク wwHww におけるフォームナビゲーション機能の XML による実現方式, 情報処理学会論文誌, 43, 3, pp.793-803 (2002).
2) 藤原克哉, 中所武司: 窓口業務アプリケーションフレームワーク wwHww におけるルール生成を自動化した自動記入エージェントの実現方式, 情報処理学会論文誌,43,6,pp.1653-1662 (2002).
3) Takeshi CHUSHO, Katsuya FUJIWARA and Keiji MINAMITANI: Automatic Filling in a Form by an Agent for Web Applications, APSEC2002, IEEE Computer Society, 239-247 (2002)
4) T. Berners-Lee, J. Hendler and O. Lassila: The Semantic Web, Scientific American, (2001).
5) D. Fensel, I. Horrocks, F. Harmelen, D. L. McGuinness and P. F. Patel-Schneider: OIL: An Ontology Infrastructure for the Semantic Web, IEEE Intelligent Systems, 16, 2, 38-45 (2001).
6) A. Seaborne: RDQL - A Query Language for RDF, W3C Member Submission, W3C (2004).