Webサービス統合による自動記入エージェントの実現方式

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，記入内容を実行時に外部 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．そこで，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．Web サービス変換機能は，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 アプリケーション変換機能は，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．サービス複合機能は，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．まずはじめに，このフレームワークを用いて記入内容自動抽出機能を構築した．さらに，この自動抽出機能を用いた自動記入エージェントを実現した．最後に，従来方式では自動記入できなかった書籍情報の記入を例題として実験し，実際にフォームに自動記入できることを確認した．

An Implementation Method of an Agent for Filling Automatically in a Form with Web Services Integration

Katsuya Fujiwara,[†] Takeshi Chusho[††] and Hideo Tamamoto[†]

This paper describes an agent that automatically fills in a Web form for users. In current systems, values to be filled in a Web form must be defined manually. Hence, we have developed a new system which extracts the values automatically from external Web services. The system is built with the Web services integration framework which we have proposed. The framework includes a Web services wrapper, a Web applications wrapper, and a dynamic service aggregator. 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 aggregator integrates these services. In this paper, we use publication search services as examples, and perform an experiment to verify whether our system with these services works well. Finally, we show that our developed agent can extract the values from external services and fill them in a form.

1. はじめに

近年，インターネットを利用したオンラインショッピングや銀行・証券取引，旅行予約などの Web システムが広く実用化されている．また，最近では行政サービスの窓口を電子化する電子政府の実現が注目されている．これらの既存の Web システムにおいて，利用者は，氏名や住所，メールアドレスなどの同じような項目の入力を求められることが多い．このような電子フォームへの記入作業を支援する自動記入機能がいくつか実現されているが，実際に自動記入できるフォームが少なかったり，記入を間違えたりするなどの問題がある．自動記入機能は，適用範囲の広く，記入精度の高い方式が求められる．そこで，我々はこれまでに，記入精度を向上させるメタデータ形式の記入ルール記述言語と適用範囲を広くする記入ルールの自動抽出方式を確立した[1),2)]．本稿では，あらかじめ利用者が記入内容を登録できる範囲に記入対象が限られていた問題を解決し，適用範囲をさらに広げるために，外部 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．また，複数の Web サービスを統合して記入内容の自動抽出に利用することにより，記入

† 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, School of Science and Technology, Meiji University

内容をより確実に抽出できる．

2. フォーム自動記入技術

2.1 現状と問題点

フォーム自動記入は，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．現在，いくつかのフォーム自動記入機能が実現されている．これらの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 つの入力をもとに，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 1 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．このルールは汎用的なので任意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 2 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．たとえば，記入欄の近くに「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欄の前に「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．

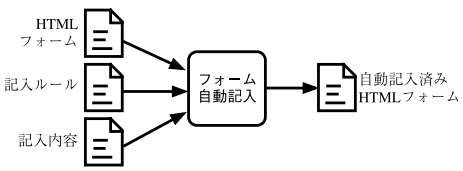

図 1 フォーム自動記入機能
Fig. 1 A model of filling automatically in a form.

方式 3 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よいかという記入ルールを定義する．たとえば，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ごとに定義する．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ごと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 2 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．

2.2 本研究の自動記入方式

先述のように，方式 1，3 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 2 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 3 を改良した自動記入方式について述べる．方式 3 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) フォームごと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ごと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という問題を解決するため

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
1：前回記入内容の再記入	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
2：記入欄周辺情報に基づく推論	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
3：フォーム対応記入ルール事前定義	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

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．

3. 記入内容自動抽出による自動記入方式

3.1 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 システムを事例システムとして取り上げ，そ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．図書登録フォームには，タイトルや著者，出版社，ISBN 番号など 6 項目の記入欄がある．これまでは，利用者が自らこれらの記入欄に書籍情報を書き写す必要があった．そこで，利用者が情報の一部を記入するだけで，残りの記入欄に記入すべき情報を外部 Web サービスから自動抽出して記入する自動記入システムを実現することで，手動記入の手間を軽減する．

本方式による自動記入の流れを**図 2** に示す．本方式は，これまでに開発した方式 3 の自動記入エージェント[1),2)] と，新たに開発した記入内容自動抽出エージェントの 2 種類のエージェントからなるシステム構成とした．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，最後に記入済みフォームを生成する．これまでの方式 3 の自動記入エージェントでは，記入内容はあらかじめ登録しておいた個人情報であった．本方式では，記入内容自動抽出エージェントが，記入内容を外部 Web サービスから抽出し自動記入エージェントに渡す．図のフォームは，情報の記入欄がある一般の HTML 形式のフォームである．記入ルールは，どのフォームへの記入ルールかを示す識別子，各記入欄についてどの内容を記入するかを示す情報の概念名，記入内容の取得方法の 3 要素から構成される．記入ルールは，フォームごとにあらかじめ定義しておく．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．

自動記入エージェントは，図のように自動記入ボタン埋込み，記入候補一覧，自動記入の 3 機能からなる．本方式の自動記入手順を以下に示す．

(1) 自動記入エージェントは，フォームとその記入ルールを取り寄せる．
(2) 自動記入エージェントは，フォームに自動記入ボタンを埋め込み，Web ブラウザに表示する．
(3) 利用者が，書籍の ISBN 番号などの一部を入力し，自動記入ボタンを押す．
(4) 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記入内容を送り，残りの記入内容を問い合わせる．

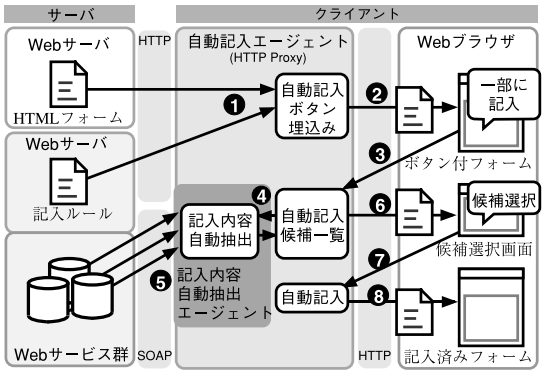

図 2 記入内容自動抽出による自動記入の流れ
Fig. 2 Flow of filling automatically in a form.

(5) 記入内容自動抽出エージェントは，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，結果を自動記入エージェントに渡す．
(6) 自動記入エージェントは，書籍情報の検索結果を記入候補一覧として表示する．
(7) 利用者が，候補一覧から一致する内容を選択し，確定ボタンを押す．
(8) 自動記入エージェントは，選択した内容を自動記入したフォームを生成し，Web ブラウザに表示する．

記入内容自動抽出エージェントは，利用者が入力した ISBN 番号など書籍情報の一部を自動記入エージェントから受け取り，それらを検索語として，外部 Web サービス群から書籍情報を検索する．その結果を記入内容候補として自動記入エージェントに出力する．

3.2 記入ルール記述言語

記入ルールの外部記述には，RDF（Resource Description Framework）ベースの記入ルール記述言語を用いる．RDF は Semantic Web 技術の 1 つで，外部記述形式に XML（Extensible Markup Language）を用いる Web 向きのメタデータ定義方式である．HTML フォームへの記入方法を定義する記入ルールのように，HTML 文書を説明するメタデータを定義するのに適している．また RDF は，オブジェクト指向に基づくクラスの継承機能を用いてスキーマ（オントロジ）の拡張や再利用が可能になっている．

本方式のルール記述言語は，これまでに開発した方式 3 の記入ルール記述言語[1)] をベースに，書籍情報のオントロジと記入内容自動抽出による自動記入の定義部を拡張して開発した．

記入ルールの記述例の一部を**図 3** に示す．記入ルール記述は，主に以下の 4 つの定義部からなる．

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://wwhww.org/1.1#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#
">
 <Form rdf:about="http://rhea.ie.akita-u.ac.jp/library/
addBook.do">
  <navigation rdf:parseType="Collection">
   <SearchExternalAndFill>
    <name>external/book[1]</name>
    <type rdf:resource="http://wwhww.org/ontology/book/
1.1#Book"/>
    <buttonLabel>本の情報を検索</buttonLabel>
    <insertButtonAt>isbn</insertButtonAt>
   </SearchExternalAndFill>
  </navigation>
  <input rdf:parseType="Collection">
   <FormItem>
    <name>title</name>
    <label>タイトル</label>
    <value>
     <Variable>
      <source>external/book[1]</source>
      <property rdf:resource="http://wwhww.org/ontology
/book/1.1#title"/>
     </Variable>
    </value>
   </FormItem>
  … 中略 …
  </input>
 </Form>
</rdf:RDF>
```

図 3 記入ルール記述例
Fig. 3 An example of the filling rule description.

A. フォームメタデータ定義

<Form> 要素内に HTML フォームに対応する記入ルールを定義する．

- rdf:about 属性：対象の HTML フォームの URL．図の例では，事例システムの図書登録フォームの URL を指定している．
- <navigation>：要素内に B のフォームの自動記入方式を定義．
- <input>：要素内に C のフォーム項目メタデータを定義．

B. 記入内容自動抽出による自動記入の定義

<SearchExternalAndFill> 要素内に記入内容自動抽出エージェントを用いた自動記入について定義する．

- <name>：自動記入エージェントのインスタンス名．
- <type>：検索対象の情報の型を示す概念名．例では書籍を指定．
- <buttonLabel>：自動記入ボタンに表示するラベル．
- <insertButtonAt>：どこに自動記入ボタンを埋め込むかを定義．例では記入欄 isbn の位置を指定．

C. フォーム項目メタデータ定義

<FormItem> 要素内に記入欄やボタンなどフォーム項目のメタデータ定義を行う．

- <name>：フォーム項目の識別子（HTML の NAME 属性で定義されたもの）．
- <label>：フォーム項目に表示されているラベル．
- <value>：要素内に D の記入内容の詳細を定義．

D. 記入内容の詳細定義

<Variable> 要素内に自動記入する記入内容の詳細を定義する．

- <source>：記入内容の値の取得元となる自動記入エージェント名．
  （<SearchExternalAndFill> 内の <name> で定義した名前）
- <property>：記入内容の情報の概念名．例では書籍のタイトルを指定．

<type> 要素と <property> 要素で用いる情報の概念名は，記入対象ドメインの共通概念であるオントロジ（語彙）としてドメインごとに設計したものである．

共通オントロジは，最小単位の基本項目と，基本項目の組合せによる合成項目の 2 種類からなる．まず，事例システムの対象とする書籍情報のオントロジを設計した．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．本オントロジの一部は，著作物などのメタデータを扱う標準オントロジの Dublin Core[6] のサブプロパティとして定義しており，Dublin Core と相互運用性がある．

合成項目は，基本項目の組合せと，書式の制約を加えたもので，著者と訳者を組み合わせた項目など，よく利用される約 130 項目をあらかじめ定義した．

4. サービス統合技術

4.1 Web サービス統合と関連研究

本方式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．

Web サービスは，近年，分散システム構築の分野

で注目されている，プラットフォーム非依存のシステム間接続技術である．Web サービス技術は SOAP，WSDL，UDDI の 3 つの標準技術からなる．Web サービスを利用した情報システム構築は，主に企業間取引の分野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[7]．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[8),9)]．この主な原因の 1 つとして，サービスインタフェースの不整合があげられる．

たとえば，本研究の自動記入方式では，フォームに記入する内容をインターネットに公開された Web サービスから取得する．書籍情報の記入欄であれば，書店の書籍検索サービスなどから書籍情報を問い合わせる．問合せはそのサービスが指定したインタフェースに合わせて行うが，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，インタフェースごとに問合せ機能を個別開発する必要がある．最近では Web サービスの普及にともない分野ごとにインタフェースの標準化が進んでおり，問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．

このような問題を解決する技術として，Semantic Web Services（SWS）がある．SWS は，Web サービスの意味情報を WSDL より詳細に記述することで，サービスを自動的に発見し利用するサービス統合技術である．Web サービスでは利用できるサービスが厳密にインタフェースの一致するものに限られるが，SWS では，サービスインタフェースのオントロジを定義することにより，厳密に一致するもののほかに，変換して結合可能なサービスも発見し利用できる．サービスの意味記述には，Semantic Web 技術を用いる．代表的な記述言語として，DAML プログラムにより提唱されている OWL-S[10)] がある．OWL-S は，RDF ベースの記述言語で，サービスの発見と選択のためのプロファイル定義，結合のためのインタフェースのオントロジ定義などの意味記述を行う．また，OWL-S の応用例として，社内の既存の資源管理，スケジュール管理，書類管理システムなどを統合して利用可能にするアプリケーションなど，主にイントラネットのインタフェース統合を目的とした研究[11)] が行われている．しかし，SWS には意味情報の記述に手間がかかる問題があり，意味記述を容易に行うためのオーサリングツール・自動生成ツールが求められる．

本稿では，SWS を用いて，インタフェースが不統一のサービス群から記入内容自動抽出を実現するサービス統合方式について述べる．さらに，自動生成によりサービスの意味記述を支援するツールを提案する．

4.2 Web アプリケーション変換

Web サービス統合に関連して，現状の Web サービスは，利用できるオープンなサービス数が少なく，動的結合の選択肢が限定される問題がある．広く普及している HTML ベースの Web アプリケーションでは実現されているが，Web サービスは存在しないものが多い．そこで，本稿では Web アプリケーションをラッピングし Web サービスとして利用する，Web アプリケーション変換機能を用いる．

Web アプリケーションのラッピングには，Screen Scraping 技術を用いる．Screen Scraping は，Web サーバに自動的にアクセスし HTML からデータを抽出する技術である．Web アプリケーションのテスト工程の自動化[12)] に利用されており，最近では，オンラインバンキングをラッピングすることで銀行の口座情報を単一 UI で管理するアグリゲータ[13)] が開発されている．しかし，これらは個別にラッパーをプログラミングする必要がある．また，Web アプリケーションの頻繁な UI の変更にそのつど対応する必要があり，より簡易なラッパーの構築法が求められる．本稿では，Web ブラウザの操作を記録し，その手順からラッパーを自動生成するツールを提案する．また，Web サービスと Web アプリケーションの両方の意味記述に対応するためのサービスメタデータ記述言語について述べる．

5. サービス統合フレームワーク

5.1 概　　要

本稿では，これらの 2 つの問題を解決する．まず，インタフェースミスマッチの解消のために，インタフェースの差異を吸収し透過的に扱うサービス変換機能を実現する．次に，サービス数不足問題の解決のために，現在普及している HTML ベースの Web アプリケーションを Web サービスにラッピングする，Web アプリケーション変換機能を実現する．

そして，これらの 2 つの変換機能に，変換した同分野のサービスの中から実行時に最適なサービスを複数選択し複合して利用するためのサービス複合機能を加えた，以下の 3 つの機能を持つサービス統合フレーム

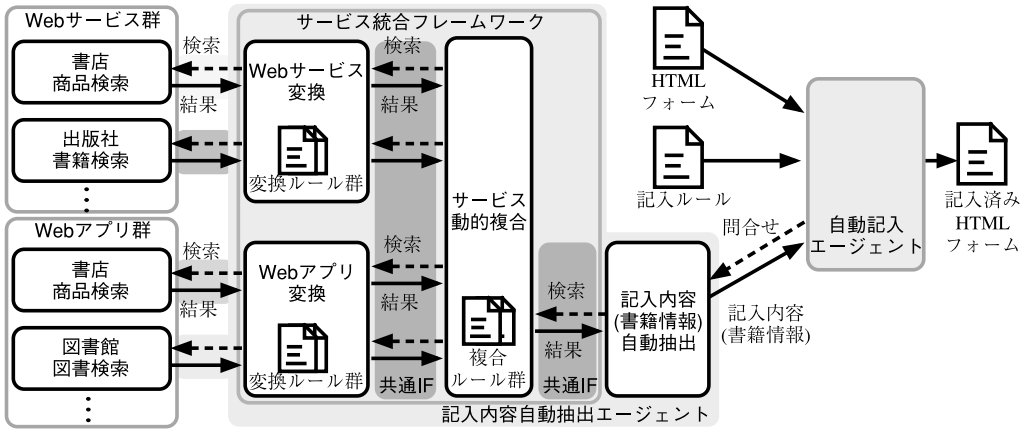

図 4 記入内容自動抽出のアーキテクチャ
Fig. 4 Architecture for value extraction for filling in a form.

ワークを構築する．

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

5.2 サービス統合による記入内容自動抽出方式

記入内容自動抽出のアーキテクチャを**図 4** に示す．図のように記入内容自動抽出エージェントは，Web サービス統合フレームワークを用いて構築する．本フレームワークは，前節で述べた 3 機能に対応する Web サービス変換，Web アプリケーション変換，サービス動的複合の 3 つのサブフレームワークから構成される．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．

まず，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．図のサービス動的複合機能により同種の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．本方式の記入精度は，記入内容の網羅性の高さ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．たとえば，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．これらの同種類のサービスの結果を組み合わせれば書籍情報が抽出できる可能性，つまり網羅性が向上する．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．動的複合機能では，各サービスの特色の違いから実行時に最適なサービスの組合せを求めて，問合せと結果の複合を行う．

図の Web サービス変換機能では，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，個別の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よい．

図の 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すでに普及している 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．

なお，このサービス統合機能は記入内容の自動抽出に限らず情報検索サービス一般に適用できる概念である．本フレームワークは，他の情報検索システムへの応用を考慮して汎用的なフレームワークとして構築した．

5.3 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 向けの問合せ言語である RDQL（RDF Data Query Language）[14] 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に特化した機能の追加である．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 3.2 節で定義したオントロジで表現する．

要求メッセージの基本形式を以下に示す．

```
SELECT [出力句] WHERE 句 [AND 句] [USING 句]
.....................................................
例：
SELECT ?x, ?y, ?z
 WHERE (?b book:title ?x)
       (?b book:author ?y)
       (?b book:ISBN ?z)
 AND ?x CONTAINS "Java" && ?y CONTAINS "Duke"
 USING book FOR <http://wwhww.org/ontology/book/1.1#>
```

先頭に ? を付けたものが変数を示す．上記の例では，?x，?y，?z，?b の 4 つの変数を使用している．出力句は検索対象の変数を定義する．例では ?x，?y，?z の 3 つを出力用変数として定義している．WHERE 句は，検索対象のプロパティを定義する．例では，ある ?b についてタイトルが ?x，著者が ?y，ISBN 番号が ?z であるものを検索することを示す．AND 句は，検索条件式を定義する．例は，?x に Java が含まれ，かつ ?y に Duke が含まれるものを検索することを示す．USING 句は，オントロジの名前空間を定義する．この例で定義した book は WHERE 句で用いている．

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〜」という出力変数に指定した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．

5.4 サービスの動的複合方式

5.4.1 概　　要

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) 選択した各サービスに問い合わせ，(c) 結果の一覧を返す，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よいので，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，待ち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場合にのみさらに応答を待つ．

この並列方式では，単純にすべ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合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．

5.4.2 サービス選択手順

サービス複合機能は，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．

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

選択肢がなくなるまでこの (1)，(2)，(3) の順にサービスの絞り込みを行う．

選択手順 (1) では，各サービスの検索機能の違いからサービスを選択する．各サービスの検索機能は，入力に設定できる検索条件と，出力として得られる情報の 2 つに違いがある．

まず，検索条件についてその対象項目の違いにより大きく以下の 3 種類に分類できる．

(a) 項目が 1 つの検索
　例：タイトルに “検索語” が含まれるものを検索．
(b) 複数項目の AND 検索
　例：タイトルに “検索語 1” が含まれかつ著者に “検索語 2” が含まれるものを検索．
(c) 複数項目の OR 検索
　例 1：タイトルか著者に “検索語” が含まれるものを検索．
　例 2：タイトルに “検索語 1” が含まれるかもしくは著者に “検索語 2” が含まれるものを検索．

本方式では，書籍情報の一部を記入し，その内容に部分一致する書籍を検索するため，1 項目に記入した場合は (a)，2 項目以上に記入した場合は (b) の検索方式に対応したサービスが必要である．さらに，各サービスはタイトルや著者などのどの項目を検索条件に利用できるかと，出力の検索結果として得られるかに違いがある．

図 4 の記入内容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合せが行われる．たとえば，タイトルと著者，ISBN 番号，価格の 4 つの記入欄があり，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，タイトルと著者の入力内容のそれぞれに部分一致する AND 条件で書籍のタイトルと著者，ISBN 番号，価格の 4 つを検索する．一方，前述のように，書籍検索の各サービスは検

索機能に違いがある．そこで本方式では，検索条件に対応したサービスの選択の優先順を以下の手順で決定することにより，必要なサービスが少なくなるようにした．

(a) まず必要な項目をすべてまとめて検索できるサービスを選ぶ．
(b) 必要な項目がすべて検索できるようなサービスの組合せを選ぶ．

先ほどの例では，タイトルと著者が入力されたので，まずタイトルと著者の部分一致の AND 検索ができるサービスの中で，(a) により，出力としてタイトル，著者，ISBN，価格の 4 つすべてを得られるサービスを選択する．それでも不足する場合は，(b) により，たとえば結果にタイトル，著者，ISBN が得られるサービスと，タイトル，著者，価格が得られるサービスの組合せを用いる．

選択手順 (2) では，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合せで，なるべくサービス数が少なくなるように以下の手順で選択する．

(a) 1 つのサービスでヒット集合が標本集合と一致するものを選択し終了する．
(b) 2 つのサービスの組合せのヒット集合が標本集合と一致するものを選択し終了する．組合せのヒット集合は，その各サービスのヒット集合の和集合である．
(c) 以下同様に N 個のサービスの組合せのヒット集合が標本集合と一致するものを選択し終了する．

一定数の組合せを探索した時点で条件に一致するものが見つからない場合は，それまでの組合せの中で最もヒット集合の大きいものを選択することとした．

同条件のサービスの組合せ候補が複数ある場合は，次の選択手順 (3) により利用する組合せを選択する．

選択手順 (3) では，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．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すべ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．

5.4.3 サービスメタデータ記述言語

サービス記述言語の WSDL では主にメッセージの型情報などを定義するが，前述のサービスの選択に必要などのような情報が検索できるのかなどのサービスの意味情報を定義することは想定されていない．そこで主に以下の 4 つのメタデータを定義する，RDF ベースのサービスメタデータ記述言語を開発した．**図 5** にサービスメタデータ記述例の一部を示す．

```
<rdf:RDF xmlns="http://wwhww.org/ws/query/1.0#"
 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#B
ook"/>
  <features>
   <Condition>
    <evaluationMethod rdf:resource="http://wwhww.org/ws/query/1.
0#And"/>
    <target rdf:parseType="Collection">
     <Condition>
      <evaluationMethod rdf:resource="http://wwhww.org/ws/query/
1.0#Contains"/>
      <targetProperty rdf:resource="http://wwhww.org/ontology/bo
ok/1.1#ISBN"/>
     </Condition>
     <Condition>
     …中略…
  <features>
   <Result>
    <targetProperty rdf:resource="http://wwhww.org/ontology/book
/1.1#ISBN"/>
    <targetProperty rdf:resource="http://wwhww.org/ontology/book
/1.1#author"/>
    …中略…
   </Result>
  </features>
  <performances>
   <CoverSet>
    <dateMeasured>2004-01-12T06:06:35Z</dateMeasured>
    <sample rdf:resource="http://wwhww.org/ex/book/sample50jp200
401"/>
    <inclusionBitmap>000010000000000000001000000000000010100000
0001000</inclusionBitmap>
   </CoverSet>
  </performances>
  <performances>
   <ResponseTime>
    <dateMeasured>2004-01-12T00:00:00Z</dateMeasured>
    <average>10300</average>
   </ResponseTime>
  </performances>
  <source>
   <WebAppWrapper>
    <script rdf:resource="../rules/akitaulib.js"/>
   </WebAppWrapper>
  </source>
 </QueryService>
</rdf:RDF>
```

図 5 サービスメタデータ記述例
Fig. 5 An example of the service metadata description.

A．検索機能の詳細定義

`<Condition>` と `<Result>` 要素内に，選択手順 (1) で用いる，各サービスの利用できる検索条件，検索できる対象項目などの検索機能の詳細を定義する．ここでは 3.2 節で定義した書籍情報のオントロジを用いる．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．

B. 標本集合の網羅性

`<CoverSet>` 要素内に，選択手順 (2) で用いる，標本集合の網羅性のデータを定義する．

C. 平均応答時間

`<ResponseTime>` 要素内に，選択手順 (3) で用いる平均応答時間を定義する．

D. 変換ルール記述

`<WebAppWrapper>` または `<WebServiceWrapper>` 要素内に Web サービス変換ルールまたは Web アプリケーション変換ルールを定義する．これらは，後述する Web サービス変換機能と，Web アプリケーション変換機能で用いる．

5.5 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．

5.6 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．本方式では，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合せ手順例を示す．

(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．

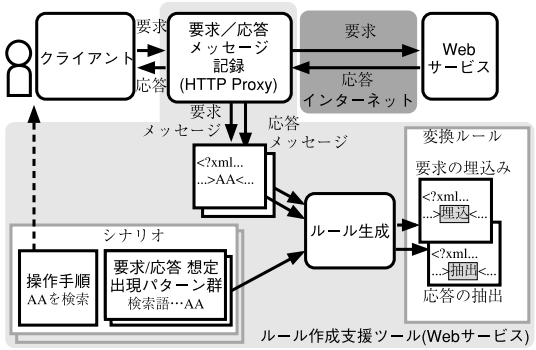

図 6 ルール作成支援ツールのアーキテクチャ
Fig. 6 An architecture of the rule builder.

(6) 結果ページの切り出し．
(7) 次の結果について (5) から繰り返す．

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つね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．

5.7 ルール作成支援ツール

これらのルール作成を支援するために，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自動生成を行う 2 つのルール作成支援ツールを構築した．

Web サービス変換ルールの作成支援ツールは，Web サービスとその Web サービス用の既存のクライアントアプリケーションとの通信内容を記録し，ルールを自動生成する．ツールは，**図 6** に示すように，Web サービスとクライアント間の HTTP Proxy として動作する．ルール作成者は，クライアントアプリケーションを実行して，たとえば「ISBN 番号に 1-2345-6789 を含む書籍を検索する」という具体的な指示からなる 8 つのシナリオのとおり操作するだけでよい．シナリオは，書籍情報の部分一致 AND 検索のルール生成を目的とし，要求メッセージのルール生成用と，応答メッセージ用の 2 種類がある．要求メッセージ用シナリオでは，タイトル，著者などのすべての項目の部分一致の AND 検索を行うシナリオを 2 つ用意した．応答メッセージ用シナリオでは，単数の結果，複数の結果，見つからない結果が得られる 3 つの場合について 2 つずつの 6 シナリオを用意した．ツールは，記録された要求，応答メッセージから，あらかじめ想定される検索条件と書籍情報の出現パターンを探し，出現部

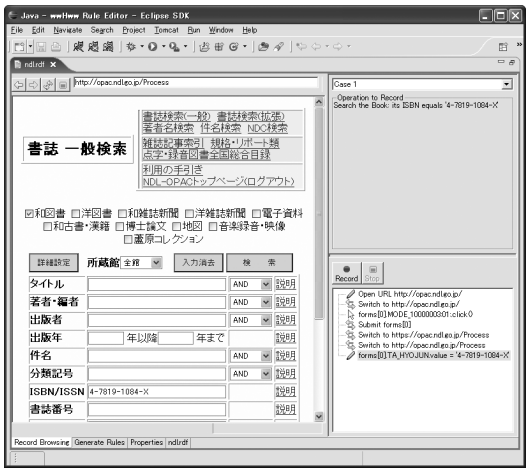

図 7 ルール作成支援ツール画面例
Fig. 7 An example of a window of the rule builder.

図 8 自動記入候補の選択画面例
Fig. 8 A window for value selection for filling in a form.

分を穴埋めするような要求メッセージ変換ルールと，出現部分を抽出する応答メッセージ変換ルールを自動生成する．書籍の商品検索ができる Amazon の Web サービスに適用し，このツールで変換ルールが生成できることを確認した．

Web アプリケーション変換ルールの作成支援ツールは，対象 Web アプリケーションの利用時の操作と Web ページを記録し，ルールを自動生成する．ツールは，**図 7** に示すように，画面左の操作の記録に対応した専用の Web ブラウザと，画面右のシナリオと操作記録の表示部からなる．ルール作成者は，このツールの専用ブラウザを用いて，Web サービス変換ルールの作成支援ツールと同じ 8 つのシナリオのとおりに操作すればよい．異なる 3 つの Web アプリケーションに適用した結果，ツールを用いて変換ルールの平均約 8 割を自動生成できたが，主に例外処理を行う約 2 割を手動定義する必要があった．

なお，ルール作成者の立場では，シナリオと手動定義部分は少ない方がよい．今後は，さらに他に適用した評価を行うとともに，より少ないシナリオでより多くの部分を正確に自動生成できる方式の検討が必要である．

6. 応用システムの実装

図 4 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．このシステムは，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．

図 8 に自動記入システムの利用画面例を示す．主な利用手順は以下のとおりである．

(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 8 左上のフォームは，研究室の図書管理システムの図書登録フォームである．フォームの ISBN 記入欄の右にある「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 8 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．
図の例では，選択画面の上部の記入候補メニューにある最初の “1:” は候補番号で第 1 候補であることを示す．続けて抽出元のサービス名とそのサービスの結果の中での候補番号を示す．このメニューから他の記入候補を選ぶと，下部の各項目がその記入候補に切り替わる．下部のメニューはタイトルだけ別の記入候補にするなど個別に選ぶこともできる．

(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自動記入できることを確認するために，研究室の図書登録フォームと，図書館の購入図書リクエストフォームの2種類のフォームに適用した．1つ目の図書登録フォームは8つの記入欄がある．そのうちのISBN，タイトルなど6つが自動記入対象の書籍情報の記入欄である．備考，キーワードの2つは自動記入の対象外である．2つ目の購入図書リクエストフォームは7つの記入欄があり，そのうちのタイトル，著者，出版年，出版者，ISBN，価格の6つが書籍情報である．リクエストの理由と利用者名は本方式での自動記入の対象外である．なお，利用者名については2.2節で述べた方式3により自動記入できる．

これらの2つのフォームに対して，著者やISBNなどの1項目に記入した場合と，著者とタイトルや出版年とタイトルの一部など複数項目に記入した場合に，残りの記入欄に正しく自動記入できることを確認した．

自動記入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（Web Services Invocation Framework）2.0[15]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[12]を用いた．

7. 実験と結果

7.1 実験方法

記入内容自動抽出による自動記入方式を評価するために実験を行った．

まず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．

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．なお，ジャンルは，文学書，地図，辞書，各種専門書などを書店と図書館におけるカテゴリ分けを参考に選んだ．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ときの応答時間を計測し，サービスの平均応答時間を求めた．

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．

7.2 実験結果

結果として，記入対象の研究室の50冊すべてについて，正しい書籍情報を自動抽出し，自動記入することができた．選ばれたサービスの組合せは，AとBの組合せが34回，AとCの組合せが16回であった．これらはともに，5.4節の選択方式(1)，(2)，(3)のすべてを用いて求められた．AとBあるいはAとCの組合せの両方のサービスで書籍情報が得られた回数が42回，片方のサービスのみで得られた回数が8回であった．

8. 考察

8.1 サービスの網羅性

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%はすべ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，サービスごと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させることができる．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，AとCの組合せの網羅率はともに100%となる．一方，BとCの2%は同じ書籍であり，BとCの組合せでは網羅性の向上は期待できない．実験では，5.4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の組合せが選択されることが確認できた．

表2 各書籍検索サービスの網羅率
Table 2 Coverage rates of services for book search.

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

表 3 変換処理の平均応答時間
Table 3 Average response time of transformation.

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 (1 回)	0.3 秒
書籍検索 B	4.6 秒	1.4 秒 (3 回)	3.2 秒
図書館 C	23.2 秒	18.9 秒 (4 回)	4.3 秒
図書館 D	12.5 秒	10.3 秒 (3 回)	2.2 秒

外部問合せの括弧内は要求応答回数

実験結果において，すべての書籍情報が正しく得られたのは，今回の実験で選択された A と B，A と C の組合せが，網羅性において適切な選択であったからと考えられる．

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いえる．

8.2 サービスの平均応答時間

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，B，C，D のすべての Web アプリケーションよりも優れていた．その理由としては，1 回の検索処理が，Web サービスの A では 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 回の要求と応答を行っていることがあげられる．この応答時間は利用者の待ち時間となるため短い方が望ましい．

応答時間を短縮するために，Web アプリケーション変換について，5.6 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が考えられる．

また，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方式が考えられる．たとえば，今回の実験では応答時間が 20 秒以上かかったものに図書館 C への問合せの 14 回があった．そのうちの 13 回はもう一方の結果が 20 秒以内に得られたため，最大応答待ち時間を 20 秒にしても自動記入できる．残りの 1 回は 20 秒を超えて結果を待つことになる．ただし，この方式では C の結果を含まないため組合せによる網羅性向上のメリットが失われる．そこで，時間内に結果が得られなかった C の検索もそのまま続行しておき，最初の記入候補に求める書籍情報が含まれない場合に残りの検索結果を表示するボタンを用意するなどの工夫が必要である．

実験結果について，まず応答時間による選択の前の，網羅性による絞り込みの時点で A と B，A と C の 2 つの組合せが選択されていた．それぞれの平均応答時間の逆数で表される性能は 1.27 回/秒，0.61 回/秒である．5.4.2 項で述べた選択手順 (3) のとおり，この性能の比率の 67.6%，32.4%に一致する，A と B の組合せが 34 回，A と C の組合せが 16 回が結果として選ばれている．

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．書籍ごとに固有の 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手間がかかる．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，工夫したりすることで検索結果を絞り込むことができるが，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 4 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 により実行時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書店 A の Web サービスも独自のインタフェースを用いている．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．

8.5 本自動記入方式の適用範囲

本稿では，書籍情報の自動記入を例題にシステムを構築し実験を行った．本システムでは，記入内容の意

味情報を定義するオントロジを RDF を用いて拡張可能である．他の分野のオントロジを構築して拡張することで，書籍情報に限らずあらかじめ用意できないものに記入可能なフォームの適用範囲を広げることができる．現在，備品登録フォームなどにある製品情報の記入欄や，出張届フォームなどにある鉄道・航空の経路情報の記入欄に自動記入する例題について検討している．

なお，あらかじめ用意しておく必要のある利用者の名前や住所，メールアドレスなどの記入内容は，これまでに開発した方式 3 の自動記入方式で記入すればよい．本稿の自動記入方式とこのような他の自動記入方式を組み合わせれば，たとえば書籍情報と利用者情報の両方が自動記入可能になり，全体として記入範囲の広い自動記入システムを実現できると考えられる．今後，これらの自動記入方式の最適な組合せ方式を検討する．

8.6 サービス統合フレームワークの応用

4.1 節で述べたように，現状の Web サービスはインタフェースが統一されていない問題があった．本研究では，異なるインタフェースを持つ複数の書籍検索サービスから記入内容を自動抽出するために，これら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，フレームワークで用いるサービス意味記述の容易化のためにルール作成支援ツールを構築した．

Web サービスの普及にともな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．たとえば，旅行予約システムがホテル予約サービスと航空券予約サービスを利用するサービス連携の例題を考える．システムは，ホテル予約サービスと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構築することになるが，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，インタフェースごとに個別開発する必要がある．サービス統合により，サービス連携に利用するこれらのホテル予約サービスや航空券予約サービスがそれぞれ統合できれば，上記の個別開発の必要がなくなるメリットがある．このような他分野への応用のためには，本稿では書籍情報の検索サービス向けに構築したフレームワークの共通インタフェースとそのオントロジを，その分野に合わせて構築すればよい．

また，本フレームワークの 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．この機能は，コストがかかるなどの理由で Web サービス化されない分野に有効である．

9. ま と め

自動記入機能は，記入精度が高く，適用範囲の広い方式が求められる．本稿では，これまでの自動記入方式があらかじめ用意できる内容に適用範囲が限られていた問題を解決するために，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実現した．

まず，外部サービスからの記入内容抽出のために，異なる Web サービスと Web アプリケーションをラッピングし共通のインタフェースで利用可能にするサービス統合フレームワークを構築した．そして，このフレームワークを用いて記入内容を自動抽出する自動記入システムを構築した．また，複数のサービスを網羅性などを考慮した組合せで利用することにより，より確実な記入内容自動抽出を可能にした．そして，書籍情報の自動記入を例題に実験を行い，実際のフォームに自動記入できることを確認した．結果として，本自動記入方式により，書籍情報などあらかじめ用意できない記入内容のフォームに適用範囲が広がることを確認した．

謝辞 本稿をまとめるにあたり，有益なアドバイスをいただいた査読者の方々に感謝いたします．

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，Vol.43, No.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，Vol.43, No.6, pp.1653–1662 (2002).
3) Chusho, T., Fujiwara, K. and Minamitani, K.: Automatic Filling in a Form by an Agent for Web Applications, *APSEC2002*, pp.239–247, IEEE Computer Society (2002).
4) Berners-Lee, T., Hendler, J. and Lassila, O.: The Semantic Web, *Scientific American*, (2001).
5) Fensel, D., Horrocks, I., Harmelen, F., McGuinness, D.L. and Patel-Schneider, P.F.: OIL: An Ontology Infrastructure for the Semantic Web, *IEEE Intelligent Systems*, Vol.16,

No.2, pp.38–45 (2001).
6) Kokkelink, S. and Schwänzl, R.: Expressing Qualified Dublin Core in RDF / XML, Dublin Core Metadata Initiative Proposed Recommendation (2002).
7) 青山幹雄：Web サービス技術と Web サービスネットワーク，電子情報通信学会技術研究報告，102(560) 情報ネットワーク，IN2002-163, pp.47–52 (2003).
8) 山本里枝子：Web サービス技術の動向，情報処理学会ソフトウェア工学研究会研究報告 2002-SE-137-12, pp.79–82 (2002).
9) 浦本直彦：Web における情報統合—セマンティック Web と Web サービス，情報処理，Vol.44, No.7, pp.707–712 (2003).
10) Martin, D. (Ed.): OWL-S: Semantic Markup for Web Services, DAML white paper (2004).
11) Tsai, T., Yu, H., Shih, H., Liao, P., Yang, R. and Chou, S.T.: Ontology-Mediated Integration of Intranet Web Services, *IEEE Computer*, Vol.36, No.10, pp.54–62 (2003).
12) Gold, R.: HttpUnit User Manual, SourceForge.Net (2003). http://httpunit.sourceforge.net/doc/manual/
13) MoneyLook, SBI Technology (2004). http://www.sbi-tech.jp/service/moneylook/index.html
14) Seaborne, A.: RDQL — A Query Language for RDF, W3C Member Submission, W3C (2004).
15) WSIF: Web Services Invocation Framework, Apache Software Foundation (2003). http://ws.apache.org/wsif/

(平成 17 年 4 月 4 日受付)
(平成 17 年 11 月 1 日採録)

藤原 克哉（正会員）

1974 年生．2002 年明治大学大学院理工学研究科基礎理工学専攻情報科学系博士後期課程修了，博士（工学）．同年 4 月より秋田大学工学資源学部情報工学科助手，現在に至る．オブジェクト指向分析・設計技法，アプリケーションフレームワーク，インターネット環境におけるアプリケーション開発技法に興味を持つ．著書『Java による Web アプリケーション入門』（サイエンス社，共著）．電子情報通信学会，日本ソフトウェア科学会，IEEE Computer Society 各会員．

中所 武司（正会員）

1946 年生．1969 年東京大学工学部電子工学科卒業．1971 年同大学院修士課程修了．同年（株）日立製作所入社．同社システム開発研究所主任研究員を経て，1993 年から明治大学理工学部情報科学科教授，現在に至る．ソフトウェア工学の研究に従事．コンポーネントベースのアプリケーション開発方法論に関心を持つ．工学博士（東京大学）．1982 年度情報処理学会論文賞，1986 年度大河内記念技術賞受賞．著書『ソフトウェア工学』（朝倉書店），『ソフトウェア危機とプログラミングパラダイム』（啓学出版），『プログラミングツール』，『人工知能』（昭晃堂，共著）等．電子情報通信学会，日本ソフトウェア科学会，人工知能学会，日本信頼性学会，IEEE Computer Society，ACM 各会員．

玉本 英夫（正会員）

1949 年生．1976 年東京大学大学院博士課程修了．同年秋田大学鉱山学部電子工学科講師．1980 年同助教授．1991 年同大学鉱山学部情報工学科助教授．1993 年同教授．1997 年同大学工学資源学部情報工学科教授，現在に至る．この間，論理回路の故障診断，画像計測，マルチメディア等の研究に従事．工学博士．電子情報通信学会，人工知能学会，計測自動制御学会，IEEE 各会員．著書『論理回路の故障診断』，訳書『フォールト・トレランス入門』（共訳）等．