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

藤原克哉† 中所武司†

近年，ワークステーションやパソコンの普及およびそれらをつなぐネットワークの普及とともに，業務の専門家が自ら情報システムを構築する必要性が高まっている．本研究では，すべての日常的仕事はコンピュータが代行すべきであるという観点から，エンドユーザが自ら作り，自ら利用できるような窓口業務を例題としたマルチエージェントフレームワークを開発した．具体的には，XMLベースの窓口/フォームメタデータ定義言語を開発し，窓口側でこれらの定義を構築・管理するエキスパートエージェントと，依頼者側で窓口検索とフォーム記入を支援するユーザエージェントと，両者を仲介するブローカエージェントからなるマルチエージェントフレームワークと，これらのエージェント間のコミュニケーションを実現するFACLを開発した．

Development of XML-based Navigation Services of an Application Framework of Window Work for End-users

Katsuya Fujiwara† and Takeshi Chusho†

The number of end-users increases on the inside and outside of offices. Enduser-initiative development of applications has become important for automation of their own task. As the solution based on the philosophy: "All routine work both at office and at home should be carried out by computers," this paper describes a multi-agent framework and an agent communication language (ACL) for distributed systems including window work in electronic commerce. The multi-agent framework which we developed is a Java application framework and includes a XML-based metadata definition language, a form-based ACL (FACL) as a common protocol for passing application forms, and the three kinds of agents.

1. はじめに

情報システムは，従来，情報処理の専門家が開発し，限られた人たちが利用してきた．しかし，近年，パソコンやそれらをつなぐネットワークの普及とともに，オフィスの内外でエンドユーザが増加し，業務の専門家が自ら情報システムを構築する必要性が高まっている．

今日の情報システム構築においては，オープンなアーキテクチャとアプリケーションフレームワーク，デザインパターン，コンポーネントなどの構成要素から，ビジュアルツールを用いてアプリケーションを再帰的に構築していく技法[1)~4)]が追求されている．特にあらかじめ用意されたテスト済みコード（クラスライブラリ）を再利用するオブジェクト指向フレームワーク[5)~7)]に関して，GUIやネットワークなどのフレームワークがすでに実用になっており，特定領域に特化したアプリケーションフレームワークが開発，利用され始めている．しかしながら，これらのフレームワークを用いたアプリケーション構築技法はシステムエンジニア主導によるアプリケーション構築を前提としており，その具体的な開発プロセスにはプログラミング言語を用いたソフトウェアのコーディングが含まれるので，エンドユーザによる構築に適さない．

本研究では，エンドユーザ主導で利用できるフレームワークが今後重要になると考え，その具体的なアプリケーション構築技法を確立するために，業務の専門家が自ら構築する必要性の高い窓口業務を例題としたアプリケーションフレームワークwwHww[20)~22)]を開発した．これまでに，例題システムとして図書管理システムを構築し，窓口業務に共通の部分と図書管理業務に固有の部分を明確に分離することにより，窓口業務のアプリケーションフレームワークを抽出した．さらに，差分コンポーネントのプラグインとコンポーネントのプロパティデータ設定の2種類のホットス

† 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
Computer Science Course, Major in Sciences, Graduate School of Science and Technology, Meiji University

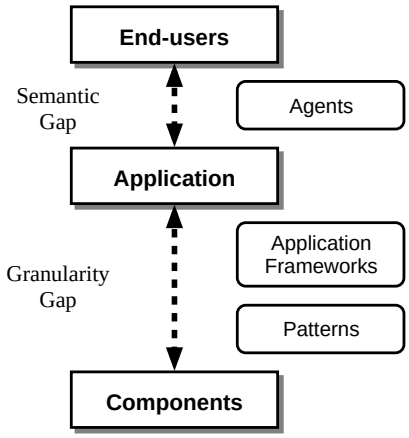

図 1 粒度的ギャップと意味的ギャップを埋める技術
Fig. 1 Technologies for bridging a semantic gap and a granularity gap between end-users and components.

ポットカスタマイズ方式により，業務の専門家がこのフレームワークを利用してアプリケーションを構築する方式を確立し，実際に別の 3 つのシステムに適用し評価を行った．

しかしながら，業務の専門家によるアプリケーション構築においては，ホットスポットのプログラムサイズよりもホットスポット構築の作業量を少なくすることが重要であり，カスタマイズを容易にするための方式が課題となっていた．そこで，本論文では窓口業務システムにマルチエージェントモデルを適用し，エンドユーザによるアプリケーション構築・利用を支援する知的ナビゲーション機能を実現することでこの問題の解決を目指す．**図 1** に示すように，アプリケーションフレームワークやパターンは，アプリケーションとコンポーネントの粒度的ギャップを解消する技術であるのに対し，エージェントはエンドユーザとアプリケーションの間の意味的なギャップを解消するために有用である．

2. 研究の目的と対象

本研究の対象とする窓口業務アプリケーションの典型的な利用手順は以下のようなものである．

(1) 窓口とフォームの検索
(2) フォームへの記入
(3) 書類の提出と処理状況の確認

窓口業務のアプリケーションは，WWW（World Wide Web）を利用したオンラインショッピングや銀行・証券取引，旅行予約などのシステムがすでにインターネット上に次々と実用化されている．また最近では，行政サービスの受付などを電子化する電子政府の実現が注目されている．しかし，現状の WWW システムには利用手順の (1)，(2) に関していくつかの問題点がある．

第 1 に，(1) の窓口検索において，現状のアプリケーションでは，窓口依頼者がインターネットを経由して申込書を提出する場合，その窓口を呼び出すために，ハイパーリンクによる検索や，全文検索を行うことになる．

ハイパーリンクを利用する場合，リンク構造が複雑で目的のところに簡単にたどり着けないことが多い．全文検索システムでは，その文書に含まれる語句で検索するため，検索条件を工夫しなければ結果件数が多くなり，目的の窓口を見つけるのが難しくなる．

たとえば，引っ越しの際に公共サービスの住所変更の窓口を探すようなエンドユーザの意図を理解した検索の自動化のためには，自動処理可能な意味定義方式が必要である．既存の WWW システムでは，窓口サービス提供者は文書や記入フォームを HTML（Hyper Text Markup Language）を用いて作成し，提供している．HTML は，人間が見るための文書を画面に配置・表示するための形式であり，コンピュータがその文書に書かれている意味を理解するのは難しい．

このような情報の意味による検索を可能にするために，HTML 文書に対してその文書を説明する情報（メタデータ）をコンピュータに処理しやすい形式で定義する方式[10)~12)] が提案されている．これらは，意味情報の定義/交換のための外部形式と，対象ドメインの語彙からなるスキーマを定義することで，知識表現とその自動処理を可能にする．しかしながら，外部形式については HTML に META タグを追加する方式[10)] や HTTP（Hypertext Transfer Protocol）を拡張する方式[13)] などがあるが，これらは特定の言語やプロトコルに依存するものである．

本システムでは RDF（Resource Description Framework)[12)] をベースとした窓口サービスのメタデータ定義言語を開発した．RDF は外部記述形式に XML（Extensible Markup Language）を用いた汎用のメタデータ定義方式であり，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の拡張や再利用が可能になっている．窓口業務に共通のスキーマを用いてメタデータを定義することで，カテゴリー分けやサービス内容による，多組織にわたる窓口を縦断した検索が可能になる．

第 2 に，手順 (2) のフォームへの記入において，窓口依頼者は氏名や住所など同じような項目の入力を求められることが多い．また，記入フォームを提出した際

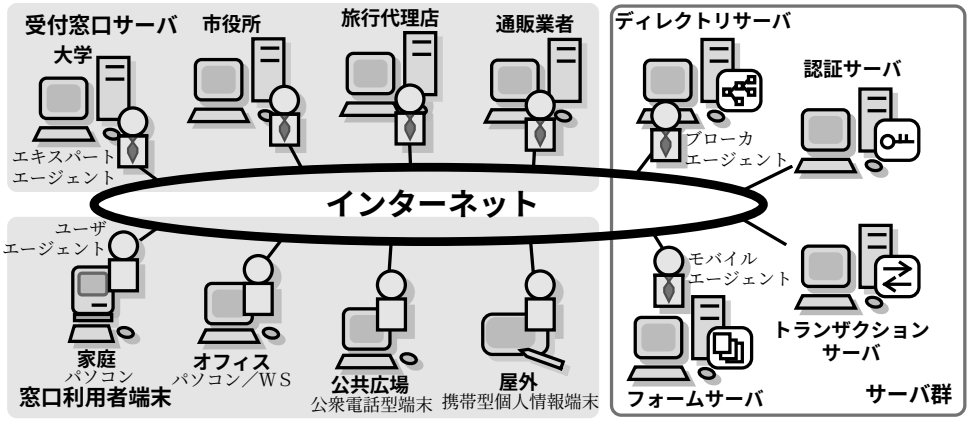

図 2 多組織間ネットワーク上の分散オフィスシステム MOON の構成例
Fig. 2 A MOON (Multiagent-Oriented Office Network) system.

に記入の不備を指摘されて記入し直す場面も多い．これらの問題を解決するために，フォームへの自動記入機能と，記入内容の自動チェック機能が必要である．自動記入については，いくつかの WWW ブラウザに独自拡張機能として実現されているものがあるが，メールアドレスを書くべき所に住所を自動記入するなどの記入ミスをしたり，記入できない場合が多いなど実用性に問題がある．クライアント側記入チェックについては，スクリプト言語 JavaScript（ECMAScript）を用いて個別に実装できるが，WWW ブラウザによる実行動作に違いがありデバッグが難しいことからあまり利用されていない．

これらの電子フォームの問題点を解決するために，XML を用いた新しいフォーム定義言語[14)~16)] がいくつか提案されているが，いずれも HTML を置き換えるものであり，現在の WWW ブラウザでは利用できない．

本システムでは，フォームの表示レイアウト定義には HTML をそのまま用い，自動記入や記入チェックからなるフォーム記入支援機能の定義を新たに HTML フォームのメタデータとして RDF ベースの記述言語を用いて定義する．HTML を利用することで，従来の WWW ブラウザでは知的ナビゲーション機能を除いた一般の電子フォームとして動作する．メタデータ定義言語では，自動記入のための各記入項目に何を記入するかを説明する意味情報を定義する．業務の専門家が定義することで，従来方式の自動記入のミスや漏れの問題が解決される．また，よく使う記入チェック機能をコンポーネント化しておくことで，スクリプトを直接記述せずに簡易に定義できるようにする．また，コンポーネントのスクリプトコードを実行環境に用意することで，ブラウザやスクリプト言語のバージョンの違いに柔軟に対応できる．

本論文では，このような窓口検索とフォーム記入を支援するナビゲーション機能のための 2 種類のメタデータ定義言語を設計し，これらを窓口業務システムに適用したマルチエージェントフレームワークを開発し評価を行う．

3. 応用システムの概要

本研究では，窓口業務を主体とした多組織間ネットワーク上の分散オフィスシステムを対象としている．マルチエージェントモデルをベースとした MOON（Multiagent-Oriented Office Network）システムの構成を**図 2** に示す．図の受付窓口サーバは，窓口業務の担当者の端末である．この業務の専門家は従来のエンドユーザコンピューティングにおけるエンドユーザである．窓口依頼者端末は，従来は窓口へサービスの依頼に訪れる一般の人の端末である．この窓口依頼者もエンドユーザである．

図の枠内に示すシステム共通のサーバは，以下のようなものである．

- ディレクトリサーバ：受付窓口のアドレスと業務（サービス）のディレクトリを管理
- フォームサーバ：各種の提出書類のフォーム（書式）を管理
- トランザクションサーバ：提出された書類とその識別番号を管理
- 認証サーバ：書類の提出者の認証の管理

図 2 には以下の 4 種類のソフトウェアエージェントが示されている．

- エキスパートエージェント：受付窓口サーバに存在

し，業務の専門家のノウハウを学習して，フォームへの記入の誘導，ヘルプメッセージの表示，記入内容のチェックなどを行う．

- ユーザエージェント：依頼者端末に存在し，個々のユーザに固有の情報を学習し，電子フォームの名前や住所などの欄に自動記入する[23]．
- モバイルエージェント：フォームサーバに存在し，クライアント側からの検索に応じて依頼者端末へ移動し，エキスパートエージェントの機能ら果たす．実際はエキスパートエージェントとモバイルエージェントは同一であり，電子フォームの中に組み込まれている．
- ブローカエージェント：ディレクトリサーバに存在し，クライアント側からの検索に応じて適切と思われるフォームを提示する．

3.1 対話言語

このようなマルチエージェントモデルでは，エージェント間コミュニケーション言語（ACL; Agent Communication Language）が必要である．その代表的なものとして FIPA ACL[19] があるが，我々は FACL（a Form-based ACL）[24] を開発した．主な相違点は，FIPA ACL があらかじめ 23 個の限定された対話行為（Communicative Act）を用意しておくのに対して，FACL では，「1 サービス = 1 フォーム」という考えから，対話行為数を限定しないフォーム単位の対話を実現した．

そして，オブジェクト指向のメッセージ駆動型の分散協調モデルをベースにした分かりやすい対話インタフェースとして，「誰に 何を どのように 頼む」というメッセージにその識別番号（どれ）を加えた 4 項目のパラメータを有する以下の基本形式を設定した．

(Who, What, How, Which)
パラメータの説明
Who: メッセージの送信先
What: メソッド名
How: メソッドの実引数
Which: メッセージ識別番号

Who は，窓口すなわち依頼先あるいは書類の提出先である．What は，依頼業務の種別あるいは提出書類の名称である．How は，依頼業務の内容あるいは提出書類の書式である．Which は，依頼または提出書類を識別するための受付番号またはコードである．

3.2 対話言語の使用例

システムの使用例を示すことで内容を説明する．特にここでは wwHww で設定した「誰に」（Who），「何を」（What），「どのように」（How）頼むという 3 種類のパラメータと「受付番号」（Which）パラメータを用いて，受付窓口業務に関連したほとんどの依頼内容を表現できることを示す．

なお，簡潔に説明するために，外部仕様の表示形式ではなく，(Who, What, How, Which) の基本形式を用いる．パラメータの a，b は定数または値のバインドされた変数を意味する．x，y は値が未定義の変数を意味する．?は値の問合せを意味する．

A．業務依頼の例

（1） (a, b, c, x) 窓口 a に処理 b を依頼するために書類 c を提出し，受け取った受付番号を x に記入する．入力形式は窓口依頼者端末に依存する（以下同様）．

（2） (a, b, , ?f) 窓口 a に依頼済みの処理 b（受付番号 f）の状況が表示される．

（3） (a, b, , -f) 窓口 a に依頼済みの処理 b（受付番号 f）に対して取消しを依頼する．

B．依頼先，業務種別，書式の問合せ例

（1） (a, b, ?x,) 窓口 a に処理 b を依頼するための書式が表示され，入力が誘導される．表示形式は窓口依頼者端末に依存する（以下同様）．

（2） (a, ?x,,) 窓口 a が担当する処理一覧が表示される．

（3） (?x, b,,) 処理 b を担当する窓口がすべて表示される．

（4） (?x, ?y=“k”,,) キーワード “k” に関連した処理を担当する窓口とその処理をすべて表示する．キーワード “k” が処理の名称または説明の中に含まれているものを検索して表示する（具体例：駐車許可を得るために “駐車” というキーワードでその担当部署と手続きを調べる）．

（5） (?a,,,) 窓口 a の業務内容が説明される．

（6） (a, ?b,,) 窓口 a が担当する処理 b の内容が説明される．

4. 知的ナビゲーション機能の実現方式

4.1 例題システムの構築

窓口業務の知的ナビゲーションシステムを実現するために図書管理システムを構築した．例題システムの全体構成を**図 3** に示す．ユーザエージェントは，検索，記入，提出の 3 つの利用手順を支援する 3 つのナビゲーション機能からなる構成とした．

- ディレクトリナビゲーション
- フォームナビゲーション
- トランザクションナビゲーション

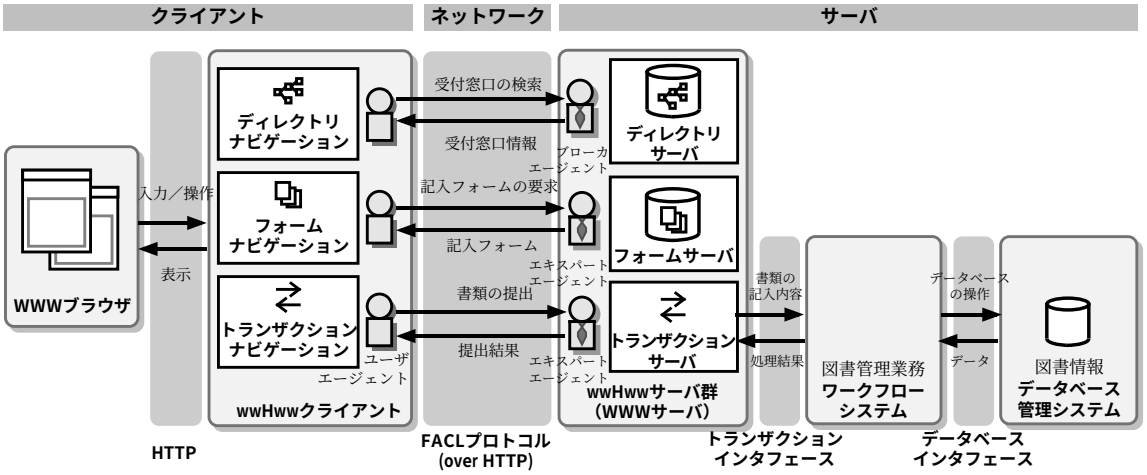

図 3 例題システムの構成
Fig. 3 Structure of library system.

インターネット上で 3.1 節で定義した対話インタフェースを実現するために FACL プロトコルを開発した．本システムでは，既存の WWW システムとの相互運用のために，メッセージのネットワーク転送プロトコルに HTTP を用いる．また，書類提出のメッセージについては，HTTP のほか，非同期メッセージングに適した SMTP と，XML メッセージングの標準技術の SOAP に対応する．窓口の利用手順に対応する FACL を用いたクライアントサーバ間での典型的な対話の流れは図 3 のようになる．

4.2 FACL プロトコルの実装

FACL の外部形式として開発した RDF ベースのメッセージ記述言語は以下のようなものである．

FACL では，メッセージを<Message>要素を用いて定義する．<Message>要素には，Who，What，How，Which の各パラメータに対応する，<who>，<what>，<how>，<which>要素が含まれる．各パラメータの値は以下の 4 種類である．

(1) **定数** a，b：値をパラメータ要素内に記述する．例：<what>図書貸出</what>

(2) **値のバインドされた変数（値の問合せ）** ?a，?b：値の問合せを示す<Inquiry>と入力値をパラメータ要素内に記述する．例：<what><Inquiry/>図書貸出</what>

(3) **未定義の変数（値の問合せ）** ?x，?y，?y="k"：値の問合せを示す<Inquiry>をパラメータ要素内に記述する．
?x の例：<what><Inquiry/></what>
?y="図書"の例：<what><Inquiry keyword="図書"/></what>

(4) **空**：パラメータ要素を省略する．

FACL は，メッセージが他の XML 文書に含まれる形式と，独立した 1 つの XML 文書とする形式の 2 つの形式を用いる．前者は 3.2 節の A(1) の処理依頼のメッセージの場合である．後者はそれ以外のメッセージの場合である．

処理依頼のメッセージにおける処理依頼内容の How パラメータのデータ形式（文書型）は窓口に固有である．FACL のメッセージ記述は文書中の任意の場所に記述すればよいので，XML 文書型は業務の専門家がバックエンドのシステムで処理しやすい形式を自由に指定できる．たとえば SOAP の場合，FACL のメッセージ記述はヘッダ部分の<Header>要素内に埋め込む．「図書管理」窓口に「図書貸出」の申請書類を提出するメッセージ例は以下のように記述できる．

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Header>
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:w="http://wwhww.org/1.0/">
    <w:Message rdf:about="">
     <w:who>/明治大学/理工学部/情報科学科/ソフトウェア工学研究室/図書管理</w:who>
     <w:what>図書貸出</w:what>
     <w:how rdf:resource="" />
     <w:which/>
    </w:Message>
  </rdf:RDF>
 </SOAP-ENV:Header>
 <SOAP-ENV:Body>
  <takeout
```

```
    xmlns="http://wwhww.org/schemas/lifecycle/takeout">
   <itemid>19991220</itemid>
   <user>fujiwara</user>
  </takeout>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

RDF 形式を示す<rdf:RDF>要素内の<w:Message>要素が FACL のメッセージ記述部分である．<w:how rdf:resource=""/>は How パラメータがこの文書全体であることを示す記述である．

依頼処理以外のメッセージはルート要素が<RDF>の XML 文書形式とする．以下に，3.2 節の B(2) の(a, ?x,,) に対応する「明治大学ソフトウェア工学研究室の図書管理窓口」の担当する処理一覧を要求するメッセージの記述例を示す．

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=
 "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:w="http://wwhww.org/1.0/">
 <w:Message rdf:about="">
  <w:who>/明治大学/理工学部/情報科学科/ソフトウェア工学研究室/図書管理</w:who>
  <w:what><w:Inquiry/></w:what>
 </w:Message>
</rdf:RDF>
```

4.3 窓口メタデータ定義言語の実装

2 章で示した窓口の意味情報を定義するために，RDF ベースの窓口メタデータ定義言語を開発した．窓口メタデータ定義言語では，窓口情報を<Agent>要素を用いて定義する．<Agent>要素は，以下のプロパティとサービス（フォーム）定義からなる．

(1) **窓口名** <name>：窓口の名前空間は，実社会の組織の構造に則したツリー構造のディレクトリとした．ディレクトリの構成例を**図 4** に示す．図の明治大学以下のノードは組織の階層構造を，末端のノードはその部署で行われている窓口のサービスを表している．

(2) **窓口の説明** <subject>（簡易表示用），<description>（詳細表示用）：窓口依頼者による検索の際に手がかりとなるようなキーワード，説明文を記述する．

(3) **窓口に関連するリンク** <link>：窓口に関連するリソース（Web ページ）へのリンクを記述する．この窓口の連絡先メールアドレスや，オンラインヘルプ集などの URL を定義しておくことで，**図 5** 下に示すようにディレクトリブラウザから依頼者が参照できる．

(4) **窓口サービスの定義** <service>：窓口の提供するサービス（フォーム）のメタデータを<Form>要素を用いて定義する．<Form>要素内の，rdf:about 属性はフォームの物理アドレス（URL），<title>要素はフォーム名を定義する．<description>，<subject>の定義は窓口の説明と同様である．

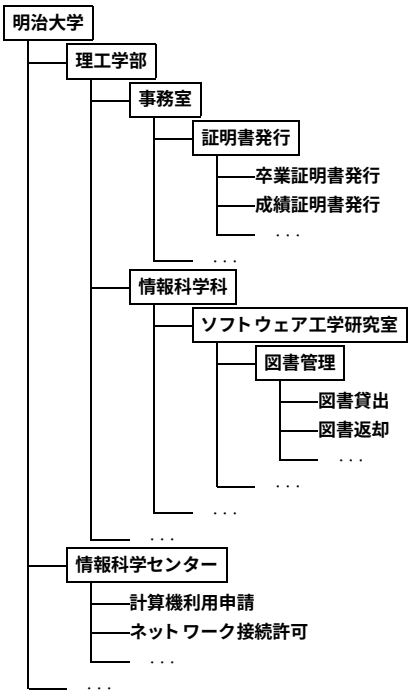

図 4 組織の階層構造に基づくディレクトリの構成例
Fig. 4 Example of the directory model.

図書管理窓口のメタデータの記述例の一部を以下に示す．

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=
 "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns="http://wwhww.org/1.0/">
 <Agent
 rdf:about="http://www.se.cs.meiji.ac.jp/library/">
  <name>/明治大学/情報科学科/ソフトウェア工学研究室/図書管理</name>
  <subject>ソフトウェア工学研究室の図書管理システム</subject>
  <description>ソフトウェア工学研究室の図書管理システム．&lt;br&gt; 図書の研究室外への持ち出し／返却手続き等を行う（自己申告制）．</description>
  <service>
   <Form rdf:about=
    "http://www.se.cs.meiji.ac.jp/library/takeout">
    <title>図書貸出</title>
    <subject>図書の研究室外への持ち出しのための貸出し手続き</subject>
   </Form>
  </service>
 </Agent>
</rdf:RDF>
```

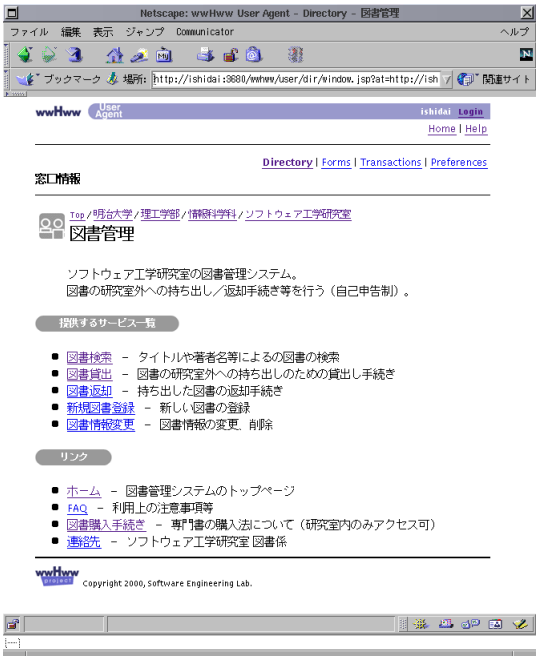

図 5 ディレクトリ検索結果の表示画面例
Fig. 5 Example of the browser at a client terminal.

業務の担当者は，担当する窓口の，メタデータ定義を受付窓口サーバに登録する．登録されたデータは，ディレクトリサーバが定期的に収集し分散管理するとともに，ユーザエージェントからの問合せに対応する．

ユーザエージェントによる「図書管理」窓口の詳細表示の例を図 5 に示す．画面は業務内容の説明文，サービス（フォーム）一覧，窓口に関連するページへのリンクからなる．

4.4 フォームメタデータ定義言語の実装

4.4.1 概　　要

2 章で示した，自動記入や記入チェックなどからなるフォームナビゲーション機能の詳細について述べる．本システムでは，フォームの表示レイアウト定義には従来の HTML を用い，記入フォームのナビゲーション機能の定義は HTML フォームのメタデータとして，RDF 形式の記述言語で定義する．

フォームメタデータは，<Form>要素を用いて定義する．rdf:about 属性はメタデータの定義対象の HTML フォームの URL を記述する．<input>要素内の<FormItem>に各フォーム部品のナビゲーション機能を定義する．1 つの<FormItem>要素は，HTML フォームの 1 フォーム部品に対応する．<name>要素にフォーム部品を識別する名前を指定する．フォーム部品の名前とは，HTML の NAME 属性で定義されたものである．たとえば，テキスト入力部品 <INPUT type="TEXT" name="namae_yomi"> の名前は namae_yomi である．ラジオボタンについては同じ名前の複数の<INPUT>要素が 1 グループとして動作するため，同じ名前のグループをまとめて 1 フォーム部品として数える．

A. フォームへの自動記入

本システムで提供するフォームへの自動記入機能は，以下の 3 つに分類できる．

(1) 窓口依頼者（クライアント）側の情報の自動記入
住所，氏名などの窓口依頼者のプロファイル情報をユーザエージェントに学習させて，自動記入を行う．住所，氏名などよく使うプロファイル名はフレームワークで定義しておく．それ以外はアプリケーションごとに定義する．
なお，自動記入では個人情報を扱うためプライバシの保護に配慮する必要がある．本システムでは，記入する内容をユーザエージェントがあらかじめ窓口依頼者に明示し，確認する．
「依頼者の名のふりがな」の自動記入定義例：

```
<value>
 <UserProfile
 name="user/name/firstname_ja_kana"/>
</value>
```

(2) 窓口（サーバ）側の情報の自動記入
サーバ上にある情報を取り寄せて記入する．

(3) 他の記入内容と連動した自動記入
フォーム上の他のフォーム部品への記入内容に基づいて計算などを行い記入する．
記入部品 total と tax の値の合計を自動記入する例：

```
<value>
  <Script template="sum">
    <param rdf:resource="total"/>
    <param rdf:resource="tax"/>
  </Script>
</value>
```

B. 記入内容の自動チェック

フォームの記入内容チェックには，書類の提出前のクライアント側でのチェックと，提出後のサーバ側でのチェックと 2 種類考えられるが，ここで述べているものは，クライアント側での事前チェックである．

クライアント側記入内容チェックは大きく以下の 3 つに分類できる．

(1) 個々の部品の記入内容のチェック
書式や値が正しいかチェックする．
よく使うチェック機能はあらかじめフレームワークにデータ型として用意しておき，<datatype>要素でデータ型名を指定する．ひらがなのみ記

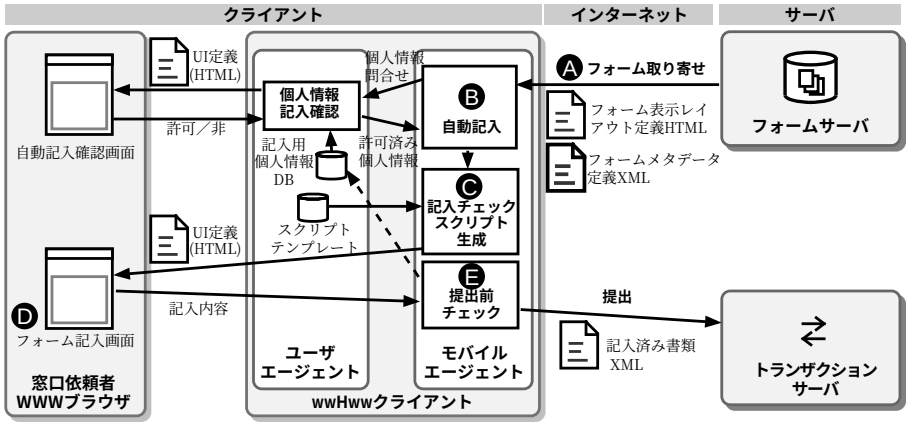

図 6 フォームナビゲーションの処理方式
Fig. 6 Example of the form navigation.

入可能な部品の定義例：<datatype rdf:resource="http://wwhww.org/datatype/ja/1.0/#Hiragana"/>

(2) 複数部品の記入内容のチェック
記入内容によって他の記入項目への自動記入（値の計算など）や表示/非表示の変更を行う．
自動記入の定義は上記 A 項の (3) に示すとおりである．

(3) 送信時のチェック
記入漏れがないか．全体として記入の矛盾がないかチェックする．
<datatype>要素で定義された個々の部品のチェックを送信ボタンを押した時点で行う．

なお，フレームワークに用意されていない特殊なチェック機能が必要な場合は，スクリプトを書く必要がある．

C. オンラインヘルプ

個々の部品の記入例などを記述した文書を HTML 形式で定義する．ヘルプは，ヘルプボタンによって呼び出され，ポップアップウインドウ上に表示する．

<FormItem>要素内の<help>要素にオンラインヘルプの URL を定義する．

例：<help rdf:resource="http://www.se.cs.meiji.ac.jp/library/takeout/help.html#name"/>

4.4.2 フォームメタデータ定義言語の記述例

図書管理システムの図書貸出フォームのメタデータの記述例の一部を以下に示す．

```
<rdf:RDF xmlns:rdf=
 "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns="http://wwhww.org/1.0/">
<Form rdf:about="http://www.se.cs.meiji.ac.jp/library
/takeout/">
 <title>図書貸出</title>
 <subject>図書の研究室外への持ち出しのための貸出し手続き</subject>
 <input>
  <FormItem>
   <name>namae_yomi</name>
   <datatype rdf:resource="http://wwhww.org/datatype/ja
/1.0/#Hiragana"/>
   <value>
    <UserProfile name="user/name/firstname_ja_kana"/>
   </value>
   <help rdf:resource="http://www.se.cs.meiji.ac.jp/lib
rary/takeout/help.html#usr"/>
  </FormItem>
 </input>
</Form>
</rdf:RDF>
```

w:Form 項目内が，図書管理フォームのメタデータ定義である．w:FormItem 項目内は，フォームの記入部品のメタデータ定義である．ここではこの記入部品について，w:datatype 項目では記入するデータの型が「ひらがなのみの文字列」であることを，w:value 項目では「依頼者の名のふりがな」をユーザエージェントに問い合わせて自動記入することを定義している．

4.4.3 フォームナビゲーションの実行環境

これらのフォームナビゲーション機能の実行手順を**図 6** と**図 7** に示す．その詳細は以下のようなものである．

A. フォーム取り寄せ ユーザエージェントは，フォームサーバより表示レイアウト定義（HTML）とフォームメタデータ定義（XML）を取り寄せる．

B. 個人情報自動記入 フォームメタデータ定義の自動記入定義より，自動記入する項目の候補リストを作成する．
ユーザエージェントは，リストにある項目の実際

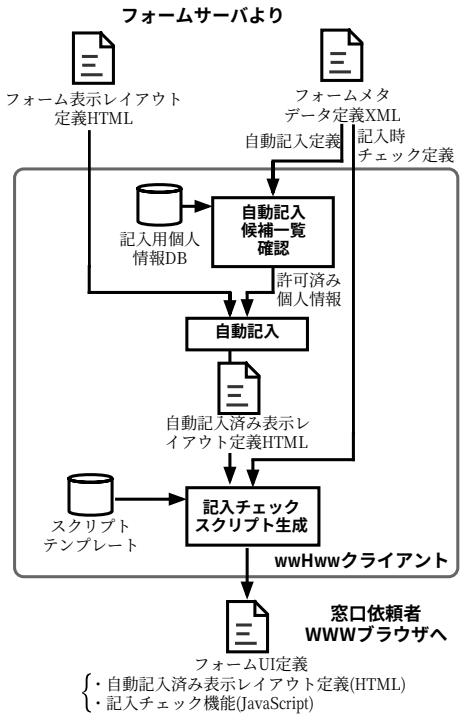

図 7 フォーム UI 生成の処理方式
Fig. 7 Example of generating form.

のデータを個人情報 DB から読み込む．依頼者の個人情報は，住所や氏名などよく記入するものについて，ユーザエージェントがあらかじめ依頼者にインタビューを行い，ユーザエージェントの個人情報 DB に登録しておく．

窓口依頼者に記入候補リストの個人情報を自動記入してよいか確認する．

依頼者が許可した項目について，個人情報を HTML フォームに埋め込む．

C. 記入時用スクリプト生成 フォームメタデータ定義の記入時の記入チェック機能定義について，対応するスクリプトを HTML フォームに埋め込む．記入チェック機能を実現するスクリプトのテンプレートは，WWW ブラウザに依存するため，クライアント側に用意しておく．

D. フォームの表示と記入 HTML フォームを窓口依頼者の WWW ブラウザに表示し，窓口利用者が手動で記入する．HTML に埋め込んだスクリプトは記入時に WWW ブラウザ上で実行される．

E. 提出前チェックと提出 WWW ブラウザから記入内容を受け取り，送信時記入チェック定義により，記入内容をチェックする．問題のある個所を指摘するメッセージを HTML に加えて，手順 C に戻る．

問題がなければ，窓口のトランザクションサーバに提出する．最後に記入内容を履歴として個人情報 DB に登録する．

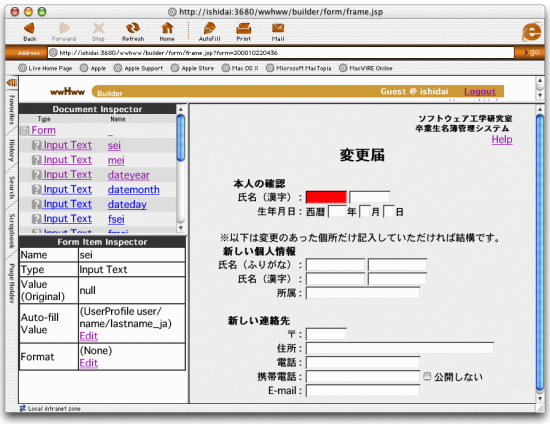

図 8 フォームナビゲーション機能の構築例
Fig. 8 Example of the builder for form navigation definitions.

すでにインターネット上にあるフォームに，フォームの意味定義を追加することで上記のフォームナビゲーション機能が実現できる．

メタデータの定義は，**図 8** に示す業務の専門家のためのビジュアルな構築ツールを用いて構築する．

これらのナビゲーション機能により，依頼者の労力が軽減されるだけでなく，業務の専門家にとっては自動記入により人為的な記入ミスがなくなり，より正確な情報が得られるメリットがある．

4.5 システムの実装環境

本システムの実装には，Java 言語と JSP（Java Server Pages）を用いた．プログラムの大きさは，Java 言語で開発した部分は 45 クラスで 6200 ステップで，JSP 部分は 46 ページで JSP から自動生成された Java コードが約 9300 ステップである．

5. 評　　価

5.1 ホットスポットのカスタマイズ方式

アプリケーション開発者は，個々のアプリケーションに固有の処理をホットスポットに定義することで，フレームワークをカスタマイズしてアプリケーションを構築する．フレームワークのカスタマイズ方式は以下の 2 種類に分類できる．

(A) プロパティデータの設定 アプリケーションに固有の処理を定義するために，そのプロパティデータをフレームワークの指定した形式で用意する．

(B) 差分コンポーネントのプラグイン アプリケー

ションに固有の機能をフレームワークの指定したインタフェースに沿って用意し，フレームワークにプラグインする．

(A) は業務の専門家による構築に適している．(B) の場合は，必要なコンポーネントが用意されていることが望ましいが，新たな開発が必要な場合もある．

本システムのナビゲーション機能において，業務の専門家が構築するホットスポットは，以下のようなものである．

(1) 窓口メタデータ定義
ディレクトリナビゲーションとトランザクションナビゲーションのための，窓口名，窓口の説明，関連する情報へのリンク，窓口サービスの定義

(2) フォーム表示レイアウト定義
HTML によるフォームの画面表示定義

(3) フォームメタデータ定義
フォームナビゲーションのための，自動記入機能，記入チェック機能，オンラインヘルプの 3 つの機能を定義

(1)，(2) と (3) のオンラインヘルプは，(A) 方式のプロパティデータとして用意すればよい．(3) の自動記入機能と記入チェック機能にはビジネスロジックが含まれるため，(B) 方式で定義する必要がある．これらの自動記入や記入チェックについては，典型的処理が多く，ほとんどのものはあらかじめ用意しておくことができる．

本システムでは，これらのプロパティデータとコンポーネントの接続情報を 4 章で述べたような XML 形式のメタデータとして定義する方式とした．その結果，プロパティ設定中心の作業によるフレームワークのカスタマイズが可能になり，エンドユーザによる構築が容易となった．本作業は，図 8 で示したようなビジュアルツールを用いて簡単に行える．

5.2 適用範囲に関する考察

最近では，企業間（B2B）電子商取引の分野で eCo Framework[8] や BizTalk[17]，UDDI[18] に代表される，インターネットを対象としデータやメッセージ記述言語に XML を用いることで相互運用性を高めたインターネット EDI[8] が注目されている．本研究の対象とする窓口業務は，受付窓口側の企業と窓口依頼者側の顧客との間の企業対消費者（B2C）電子商取引を主な対象としているが，B2B についても企業がサービスを提供するという点で共通であり，B2B に適用可能である．本システムでは，メッセージ記述形式に XML を用いるとともに，トランザクション処理を部品として用意し，容易に交換できるようにすることで他の XML を用いた B2B 電子商取引技術との相互運用を考慮している．なお B2C 電子商取引では特に，窓口依頼者の立場での使い勝手が重要であり，依頼者とのユーザインタフェースとなるフォームをはじめとした，依頼者の窓口利用を支援するナビゲーションシステムが重要になると思われる．

6. おわりに

本論文では，窓口業務を例題としたエンドユーザ向きアプリケーションフレームワーク wwHww について述べた．特に，知的ナビゲーションに必要な窓口とフォームのメタデータを定義する 2 種類の記述言語を開発し，窓口側でこれらの定義を構築・管理するエキスパートエージェントと，依頼者側でナビゲーション機能を実行するユーザエージェントと，両者を仲介するブローカエージェントからなるマルチエージェントフレームワークを実現し，これらのエージェント間のコミュニケーションを実現する FACL を開発した．

結果として，2 種類のホットスポットのカスタマイズ方式と業務の専門家によるカスタマイズを支援するツール群により，本研究の目的であるエンドユーザによるアプリケーション構築の作業量が軽減されることを確認した．また，フォームナビゲーション機能により，依頼者は窓口の多様な検索が可能になり，フォーム記入の労力が軽減されることを示した．

参考文献

1) Pree, W.: *Design Patterns for Object-Oriented Software Development*, ACM Press, (1995). 佐藤，金澤（訳）：デザインパターンプログラミング，トッパン (1996).
2) Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: *Design Patterns, Elements of Reusable Object-Oriented Software*, Addison-Wesley (1995).
3) Johnson, R.E.: Frameworks = (Components + Patterns), *Comm. ACM*, Vol.40, No.10, pp.39–42 (1997).
4) Mellor, S.J. and Johnson, R.: Why Explore Object Methods, Patterns, and Architectures?, *IEEE Software*, Vol.14, No.1, pp.27–30 (1997).
5) Sparks, S., Benner, K. and Faris, C.: Managing Object-Oriented Framework Reuse, *IEEE Computer*, Vol.29, No.9, pp.52–61 (1996).
6) Islam, N.: Customizing system software using OO framework, *IEEE Computer*, Vol.30, No.2, pp.69–78 (1997).
7) Fayad, M. and Schmidt, D.C.: Object-oriented

application frameworks, *Comm. ACM*, Vol.40, No.10, pp.32–38 (1997).
8) Glushko, R.J., Tenenbaum, J.M. and Meltzer, B.: An XML Framework for Agent-based E-commerce, *Comm. ACM*, Vol.42, No.3, pp.106–114 (1999).
9) Simple Object Access Protocol (SOAP) 1.1, *W3C Note*, World Wide Web Consortium (2000).
10) Kunze, J.: Encoding Dublin Core Metadata in HTML, RFC 2731, IETF (1999).
11) 浦本直彦，武田浩一：インターネットでの情報の記述と交換方式の最近の動向，人工知能学会誌，Vol.13, No.4, pp.519–527 (1998).
12) Lassila, O. and Swick, R.R.: Resource Description Framework (RDF) Model and Syntax, *W3C Recommendation*, World Wide Web Consortium (1999).
13) Nielsen, H., Leach, P. and Lawrence, S.: An HTTP Extension Framework, RFC 2774, IETF (2000).
14) Boyer, J., Bray, T. and Gordon, M.: *XFDL Specification v4.1 for Windows*, PureEdge Solutions (2000).
15) McKenzie, G.F.: *XFA Specification*, JetForm Corporation (1999).
http://www.xfa.org/specifications.html
16) Dubinko, M., Dietl, J., Merrick, R., Raggett, D., Raman, T.V. and Welsh, L.B.: XForms 1.0, *W3C Working Draft 19 December 2000*, World Web Consortium (2000).
17) BizTalk: *BizTalk Framework Specifications*, Microsoft Corp. (1999).
http://www.biztalk.org/Resources/specs.asp
18) UDDI: Universal Description, Discovery, and Integration.
http://www.uddi.org/specification.html
19) FIPA: Foundation for Intelligent Physical Agent, The Specification.
20) 中所武司：wwHww：分散オフィスシステムのためのエンドユーザコンピューティング向きオブジェクト指向モデル，情報処理学会ソフトウェア工学研究会資料 94-SE-97-5 (1994).
21) 藤原克哉，中所武司：窓口業務を例題としたエンドユーザ向き分散アプリケーションフレームワーク wwHww の開発と適用評価，情報処理学会論文誌，Vol.41, No.4, pp.1202–1211 (2000).
22) 藤原克哉，中所武司：エンドユーザ向き分散アプリケーションフレームワーク wwHww—知的ナビゲーション機能の XML による実現方式，情報処理学会ソフトウェア工学研究会資料 2000-SE-128-1, pp.1–8 (2000).
23) 南谷圭持，藤原克哉，中所武司：エンドユーザ向き分散アプリケーションフレームワーク wwHww—自動記入エージェント実現方式，情報処理学会第 61 回大会講演論文集（1）1W-5, pp.273–274 (2000).
24) Chusho, T. and Fujiwara, K.: FACL: A Form-based Agent Communication Language for Enduser-Initiative Agent-Based Application Development, *COMPSAC2000*, pp.139–148, IEEE CS (2000).

(平成 13 年 2 月 26 日受付)
(平成 13 年 12 月 18 日採録)

藤原 克哉（学生会員）

1974 年生．1999 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前期課程修了．現在同大学院博士後期課程在学中．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．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．