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

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共に、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。また、今日の情報システム構築においては、フレームワークやデザインパターン、コンポーネントウェアなどの構成要素からアプリケーションを再帰的に構築していく技法が追求されている。本研究では、業務の専門家が自らのアプリケーションの構築に利用できるような窓口業務のアプリケーションフレームワークを開発した。窓口業務の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個々の窓口に依存する部分を明確に分離することにより、窓口業務アプリケーションフレームワークを抽出した。さらに、そのフレームワークを利用して業務の専門家がアプリケーションを構築する方法を確立し、実際に別のシステムに適用し、その評価を行った。

Development and Application Evaluation of A Distributed Application Framework for End-users

Katsuya Fujiwara† and Takeshi Chusho†

The number of end-users increases on the inside and outside of offices. This paper describes an application framework for window work in banks, city offices, travel agents, mail-order companies, etc. based on the philosophy of "All routine work both at office and at home should be carried out by computers." We developed the application framework of the window work which the business experts were able to use for building application. The window work application framework has been extracted from the library system which was developed as an example of the window work. Then, the framework was applied to another system, and was evaluated.

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、限られた人達が利用してきた。しかし、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共に、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、従来の構造化技法に代表されるトップダウン技法からコンポーネントウェアの組み合わせ（モデリング）というボトムアップ技法へ移行しつつある。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、コンポーネントウェア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が追求されている。

特にオブジェクト指向フレームワーク [1)~3)] に関して、GUI やネットワークなどのシステムレベルのフレームワークがすでに実用になっている。最近では、特定の問題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、利用されはじめており、企業の基幹業務を対象としたエンタプライズアプリケーションフレームワークが注目されている [4)~6)]。

しかしながら、フレームワークのカスタマイズ方式をはじめとした具体的なフレームワークを用いたアプリケーション構築技法は未だ確立されていない。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、業務の専門家が自らアプリケーションを構築する必要性の高い窓口業務のアプリケーションフレームワークを取り上げる。

本稿では、その例題システムとして図書管理システ

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

☆ 本研究の一部は EAGL 事業推進機構の支援を受けて実施したものである。

ムを構築し、窓口業務に共通の部分と個々の窓口に依存する部分を明確に分離することにより、窓口業務アプリケーションフレームワークを抽出する。

さらに、そのフレームワークを利用して業務の専門家がアプリケーションを構築する方法を確立し、実際に別のシステムに適用し、その評価を行う。

2. 応用システムの概要

窓口業務のアプリケーションは、WWW を利用したシステムが既にインターネット上に次々と実用化されているが、個別に開発されているため開発コストに見合った対象に限定される。依頼者にとっては、アプリケーションの増加が便利さを招く反面、ユーザインタフェースは多組織間で統一されておらず、必ずしも使い勝手の良いものではなく、個別に操作方法を習得する必要がある。

本研究では、業務の専門家自身が窓口の受付業務を自動化するためのアプリケーションフレームワーク [7),8)] を開発することにより、上記の問題を解決することを目的とする。

2.1 シナリオ

本報告で取り上げるオフィスの窓口業務について、そのイメージを明確にしておく。

例えば転居の場合、その前後で依頼者の立場でどのくらいの窓口業務が発生するだろうか。オフィスおよび家庭での窓口を思いつくままにあげると以下のようなものがある。

オフィス 勤労課、経理課、庶務課、……

家庭 区役所、水道局、NTT、郵便局、警察署、陸運支局、地方法務局、学校、電力会社、ガス会社、NHK、新聞販売店、銀行、保険会社、証券会社、クレジット会社、学会、出版社、同窓会、……

この中には電話で済むものもあるが、何らかの書類を提出するものも多い。さらにその多くは郵送ではなく、直接窓口へ行く必要がある。これらの処理がすべて自分のオフィスあるいは家庭のコンピュータから行えれば、時間の節約だけでも大きい。

一方、これらの窓口業務の担当者も毎日同じような質問を受け、同じような説明を繰り返し、提出された書類の記入内容のチェックをし、ほぼルーチンワーク化した判断処理をしている。これらの処理がコンピュータ化されれば、その分サービスの向上や費用の低減につながる。

2.2 応用システム

本研究で対象とする、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 MOON (a Multi-Organizational Office Network system) のシステム構成の例を図 1 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。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

図の依頼者端末のユーザインタフェースでは、以下のような共通のインタフェースを提供する。

- 受付窓口の問い合わせ (検索)
- 受付窓口からフォーム (書式) の取り寄せ
- フォームの表示とフォームへの記入
- 記入済書類の受付窓口への提出

2.3 2 種類のエンドユーザの利点

本システムでは、窓口への依頼者と窓口業務の専門家という 2 種類のエンドユーザを対象とし、依頼者にとっては、一つのブラウザから「依頼先」、「依頼項目」、「依頼内容」の 3 項目を基本とした同一のインタフェースですべての窓口への依頼を済ませられること、および窓口業務の専門家にとっては、このアプリケーションフレームワークを利用して自らの業務の電子化ができることを実現する。

双方のエンドユーザが享受する具体的な利点を以下に示す。

(1) 窓口への依頼者の利点
  - (a) 依頼方法 (手続き) の容易化
    誰に / 何を / どのように頼めばよいか、あるいはどのように記入すればよいかなどについて、容易に知ることができる。
  - (b) 書式 (書類) の入手/提出の容易化
    窓口へ出向く必要がない。
  - (c) 結果の迅速な応答
  - (d) 処理状況のフォローの容易化
    依頼業務の状況確認が簡単にできる。
  - (e) その他

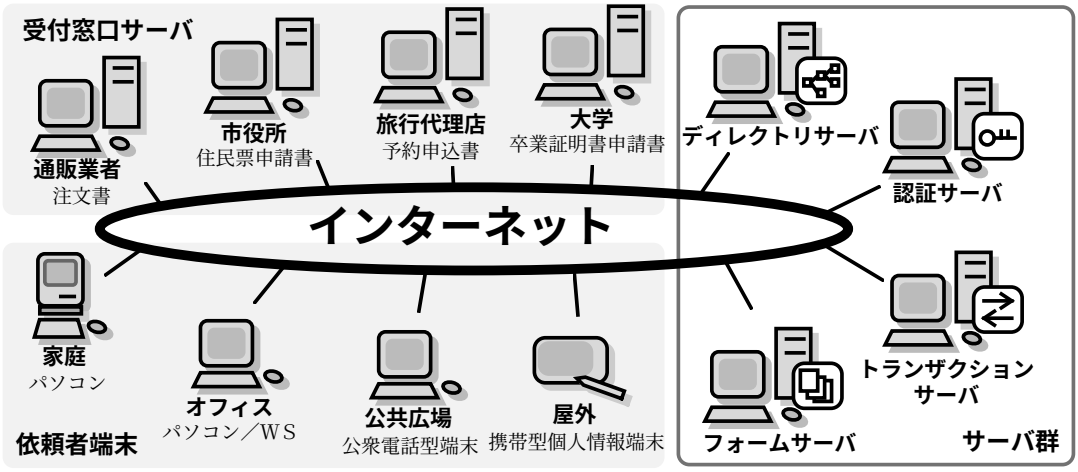

図 1 多組織間ネットワーク上の分散オフィスシステム MOON の構成例

窓口のたらい回し、窓口での順番待ちがなくなる。住所、氏名、電話番号などの手書き記入が不要になる。等々……

(2) 窓口業務の担当者および組織の利点

(a) ルーティンワークからの解放
手続き方法に関するヘルプ機能 (問い合わせ、誘導、例示) を自動化できる。

(b) 業務知識 (ノウハウ) の共有財産化
マニュアル化がしづらい属人的な断片的ノウハウのコンピュータ化が容易である。

(c) より創造的な業務の達成
業務の効率化にともない、業務の質の向上により多くの時間をかけられる。

3. システムモデル

3.1 対話インタフェース

今回の応用システムのシステムモデルについて述べておく。

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わ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する。

(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。

このメッセージ形式にちなんで、本システムを wwHww (the Who-What-How with WWW system) と名付ける。

3.2 機能仕様

簡単に各パラメータの仕様を示す。

Who メッセージの送信先
定数 依頼先
変数 同上またはその問い合わせ

What メソッド名
定数 依頼業務種別または提出書類名称
変数 同上またはその問い合わせ

How メソッドの実引数
定数 依頼業務内容または提出書類の書式
変数 同上またはその問い合わせ

Which メッセージ識別番号
定数 依頼または提出書類の識別番号
変数 同上またはその問い合わせ

上記のメッセージ形式は基本的には内部仕様であり、実際のシステムとユーザの間のインタフェースは別に定める。

実世界の窓口利用者と受付窓口との対話は、書類のフォーム (書式) の取り寄せ、フォームへの記入、書類の提出という書類を媒介とした対話と考えられる。本システムでは、フォームを電子フォームに置き換え、書類の表示・記入のための、統一されたインタフェースを受付窓口のクライアントとして実現する。

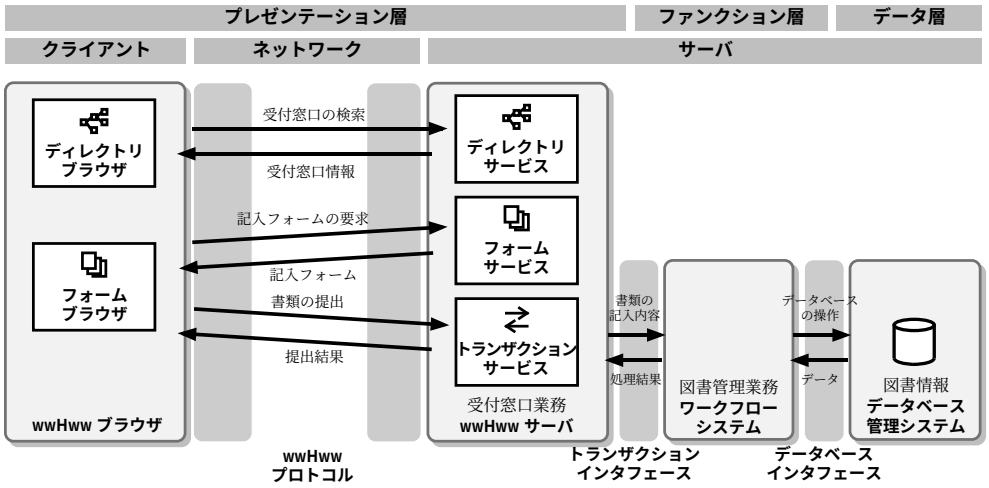

図 2 図書管理システムの構成

3.3 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。?は値の問い合わせを意味する。

(1) 業務依頼の例

(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。入力形式は依頼者側のエージェントに依存する (以下同様)。

(2) 依頼先、業務種別、書式の問い合わせ例

(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。表示形式は依頼者側のエージェントに依存する (以下同様) 。

(a, ?x,,) 窓口 a が担当する処理一覧が表示される。

(?x, b,,) 処理 b を担当する窓口がすべて表示される。

(?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)。

(3) 業務内容の説明要求の例

(?a,,,) 窓口 a の業務内容が説明される。

(a, ?b,,) 窓口 a が担当する処理 b の内容が説明される。

4. 例題システムの構築

このような窓口の受付業務の自動化のためのアプリケーションフレームワークを実現するために、まず、研究室内の図書管理業務を取り上げ、その一連の受付窓口業務を自動化した図書管理システム [8] を開発した。

4.1 システム構成

図書管理システムの全体構成を図 2 に示す。図書管理システムはネットワーク上での利用を考え、窓口利用者側と受付窓口側に分割したクライアント／サーバ型のシステム構成とした。両者はインターネットを通した共通プロトコルにより対話を行う。ネットワークから利用できることにより、自宅や外出先で、図書リストの最新情報の確認や、購入した図書の登録が可能となる。

本システムは、マルチプラットホームに対応した Java を利用して構築している。現在、Solaris2、Windows95、WindowsNT のプラットホーム上で動作することを確認済である。

4.2 サーバ側システム構成

4.2.1 概 要

図書管理システムのサーバ側システム構成は、窓口業務の基本フレームワークに対応する wwHww サーバ

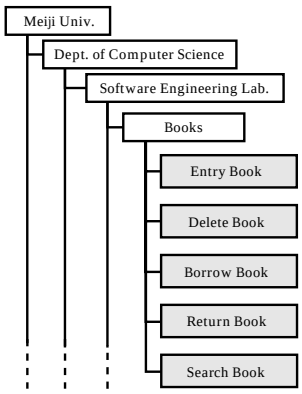

図 3 ツリー構造のディレクトリ構成の例

とそのバックエンドに位置する業務ワークフローシステムおよびデータベース管理システムで構成した。3 層クライアントサーバシステムの観点では、各々、プレゼンテーション層、ファンクション層、データ層に相当する。wwHww サーバをプレゼンテーション層に分類したのは、図書管理システムとしては、wwHww サーバと wwHww ブラウザの連携により窓口利用者との対話を行っているからである。

業務ワークフローシステムおよびデータベース管理システムは、窓口業務のアプリケーションフレームワークの観点では本来取り扱わない部分であり、窓口業務単体のシステムも構築できる。しかし、実用上は今回の例のように連携して利用されることが多いと思われる。なお、データベース管理システムには Oracle を利用している。

wwHww サーバは窓口利用者側から主に次の 3 種類の要求を受け付ける。

(1) 受付窓口と記入フォーム (書式) の検索
(2) 記入フォームの取り寄せ
(3) 書類の提出

そこで、図 2 に示すように、それぞれの要求に対応する次の 3 つからなる構成とした。

- ディレクトリサービス
  フォームの入手先、書類の提出先などを含む受付窓口情報をディレクトリ構造で管理し、窓口利用者からの多様な検索に対応
- フォームサービス
  記入フォームの書式データを管理し、受付窓口利用者からの要求に応じて記入フォームを提供
- トランザクションサービス
  窓口利用者から提出された書類を受け取り、業務ワークフローシステムと連携してトランザクション管理機能を提供

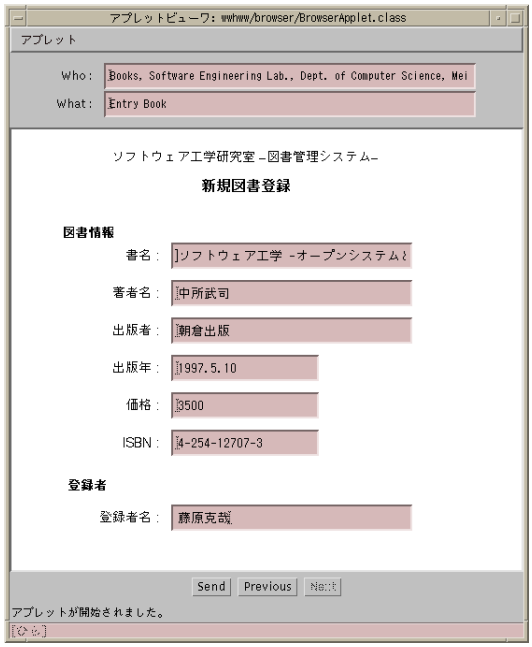

図 4 フォームブラウザの使用例

4.2.2 サービス管理方式

受付窓口の処理方式は、利用者に分かりやすく、業務の担当者に管理しやすくする必要がある。このような機能を実現するために、本システムでは受付窓口の名前空間を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。ディレクトリでは、受付窓口名、検索用のキーワード、サービスの説明、サービスの位置情報（物理アドレス）を管理する。

ディレクトリ構成の例を、図 3 に示す。図の「Meiji Univ.」以下の中間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。このようなディレクトリモデルは、組織内の階層構造をそのまま反映できるイントラネットに適している。

しかし、多組織間にわたるインターネット上の名前空間を扱う場合は更に概念的なサービスのカテゴリ分けを考慮する必要がある。

4.3 クライアント側システム構成

図書管理システムと受付窓口利用者とのインタフェースとなる wwHww ブラウザは、図 2 に示すようにディレクトリブラウザとフォームブラウザからなる。受付窓口利用者は、ディレクトリブラウザを用いて目的の受付窓口とその記入フォームを見つけ、フォームブラウザを用いて記入フォームを取り寄せて記入し、提出する。

現在は、フォームブラウザとして Java アプレット

図 5 受付窓口業務アプリケーションフレームワーク

版と Java アプリケーション版を開発した段階である。図 4 にフォームブラウザの使用例を示す。アプレット版はプログラムをネットワークからロードして実行するため、Java 対応の WWW ブラウザがあればネットワーク上のどこからでも利用可能であり、外出先での利用に便利である。しかし、プログラムのロードに時間がかかる欠点がある。一方、オフィス内や自宅から利用する場合には、アプリケーション版をあらかじめ用意しておくことによって応答性が向上する。

4.4 共通プロトコル

インターネット上で 3. のシステムモデルで定義された対話インタフェースを実現するために wwHww プロトコルを設定した。

wwHww プロトコル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 2 のようになる。まず受付窓口のサービスを検索し、そのサービスの記入フォームを取り寄せ、記入した書類を提出する。

今回のシステムでは、wwHww プロトコルは Java の分散オブジェクト環境である RMI(Java Remote Method Invocation) 上に構築している。

5. 窓口業務フレームワーク

5.1 フレームワークの構成

図 2 の図書管理システムにおいて窓口業務アプリケーションに共通の部分はクライアント側の wwHww ブラウザ、ネットワークの wwHww プロトコル、サーバ側の wwHww サーバである。あらかじめこれらの共通部分を受付窓口業務におけるアプリケーションフレームワークとして提供することで、業務の専門家が自ら受付窓口業務を構築できると考えられる。

図書管理システムのフレームワーク部分の構成を図 5 に示す。wwHww ブラウザは、ディレクトリブラウザとフォームブラウザと両者の共通部分である wwHww ブラウザ基本部からなる。wwHww サーバは、3 種類のサービスモジュールとそれらの管理を行う wwHww サーバ基本部からなる。トランザクションサービスでは、用意されたインタフェースを用いて業務ワークフローシステムとの連携が可能である。

なお、図 2 の業務ワークフローシステムとデータベース管理システムは、図書管理業務に固有な部分なので図 5 のアプリケーションフレームワークには含まれない。

今回、Java を使用して開発したアプリケーションのコードサイズは、受付窓口業務のフレームワークが 19 クラスで約 1500 ステップ、図書管理システム固有の部分が 7 クラスで約 600 ステップとなっている。

5.2 フレームワークの評価

フレームワークは、ある問題領域のソフトウェアの設計と実装の再利用を目的としている。協調動作するクラス群として与えられる問題領域に共通なコードをフレームワークとして用意しておくことで、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機能のみを開発すればよい。

フレームワークは、問題領域内の個々のアプリケーションに依存する部分（ホットスポット）と共通の部分（フローズンスポット）に明確に区別される [1)]。一般に、問題領域を抽象化し柔軟性を持たせたフレームワークは、適用範囲が広くなるが、アプリケーション開発者が作成すべきホットスポットのコードが多くなる。一方、特定の問題領域に特化するほど適用範囲は狭まるが、アプリケーションとしての完成度は高いためアプリケーション開発者が作成すべきコードが少なくてすむ。

本システムでは、フレームワークの利用を容易にするために、以下の 2 種類の支援機能を実現している。その概要を図 6 に示す。

(1) **差分コンポーネントのプラグイン**

アプリケーション開発者は、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必

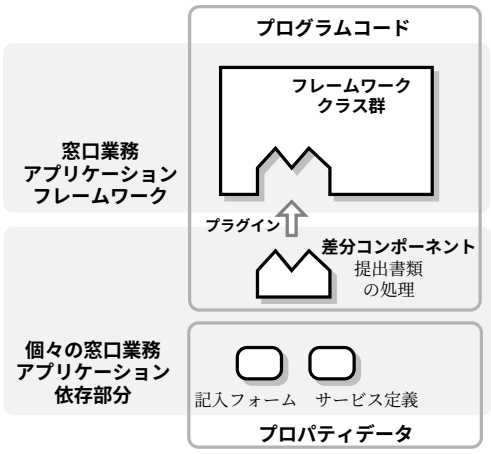

図 6 窓口業務フレームワークの 2 種類のカスタマイズ方法

要がある。窓口業務フレームワークの場合、提出された書類の処理方式は個々の窓口で異なるので、業務専門家（アプリケーション開発者）が定義しなければならない。

本システムでは、良く使われるデータベース蓄積方式と紙に印刷する方式をコンポーネントとして用意しておくことで、柔軟性を保ちつつアプリケーション開発者の労力を軽減している。

(2) **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の作成方法としては、上記の差分コンポーネントの他に、固有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れば良い場合がある。例えば、窓口業務フレームワークでは、業務の専門家は受付窓口の説明などの記述や記入フォーム等はプロパティデータとして用意すれば良い。

本システムでは、この方式によりアプリケーション開発者の用意するコードを軽減している。

5.3 業務の専門家による構築手順

業務の専門家がアプリケーションフレームワークを利用して、今回の図書管理システムのような窓口業務アプリケーションを構築する場合の具体的な手順は次のようになる。

(1) サービスの定義

まず、受付窓口で提供するサービスについて定義する。図書管理システムでは、5 つのサービスを提供しており、それぞれについて窓口の名前と窓口の検索の際に利用するキーワードやカテゴリ分けなどの情報を定義する。

(2) フォームの作成

それぞれのサービスに必要な記入項目を定義し、その記入を支援するナビゲーション情報の設定を行う。記入フォームは、GUI によるフォーム作成ツールを利用して作成する。図書管理システムでは、5 つのサービスに対応した 5 種類の記入フォームが必要となる。ナビゲーション機能には、記入内容チェック、オンラインヘルプ、エージェントによる自動記入などがある。

(3) 書類の処理方式の設定

提出された書類の処理方式を設定する。書類の処理方式には、ワークフローシステムなどの業務アプリケーションに受け渡す方法、データベースへ蓄積する方法、紙に印刷する方法などが考えられる。業務の専門家による業務アプリケーションの開発方法としては、エンドユーザ向け開発環境を利用した開発を想定している。

(4) サーバへの登録

最後に上記の 1 から 3 で定義した内容を wwHww サーバのディレクトリサービス、フォームサービス、トランザクションサービスに登録する。

5.4 他の例題への適用

この窓口業務フレームワークを利用して、業務の専門家が実際に窓口業務アプリケーションが作れることを確認するために、研究室内の備品管理システムを構築した。備品管理システムは、図 2 に示す図書管理システムと同様に、窓口業務システムと業務ワークフローシステム、データベース管理システムの 3 つのサブシステムから構成される。構築は前述の業務の専門家による構築手順に沿って以下のように行った。

(1) サービスの定義

備品管理の受付窓口では、備品の登録・削除・貸出・返却・検索の 5 つのサービスを提供する。ここでは各々のサービスについて、サービスの名前や提供するサービスの説明等の窓口業務フレームワークが指定した項目を埋めていく形でプロパティデータを作成した。

(2) フォームの作成

次に、サービス毎の記入フォームについて、窓口利用者が記入する項目の設定と、その項目を配置したフォーム全体のレイアウトの定義を行った。今回はフォーム用の記述言語でフォームデータを作成したが、現在、業務の専門家のためのビジュアルなフォーム作成ツールを開発中である。

(3) 書類の処理方式の設定

備品管理システムでは、提出された書類の処理をワークフローシステムで行う。そこで、フレームワークの指定したインタフェースに沿ってそのワークフローシステムを呼出すコードを定義した。

今回開発した備品管理システムのコードサイズは、備品管理システム固有の部分が 7 クラスで約 600 ステップと、図書管理システムとほぼ同程度である。備品管理システムに固有のコードのうち 6 クラスが業務ワークフローシステム部分である。残りの 1 クラスがその業務ワークフローシステムを呼び出す、窓口業務フレームワークにプラグインするコードである。

(4) サーバへの登録

備品管理の受付窓口は、図書管理システムと同じ受付窓口サーバに追加する形で登録した。具体的には、1 から 3 で作成したプロパティデータのファイルをサーバにコピーし、サーバの起動スクリプトにそのファイルを追加した。登録までの一連の手順は統一された構築ツールで支援する予定である。

備品管理システムの構築に当たって、窓口業務フレームワークの部分はそのまま利用し手を加えなかった。

今回は、業務ワークフローシステムを独自に開発したため、システム構築の時間の大半をこの業務ワークフローシステムの開発で費やした。このような、業務の専門家によるこれらのワークフローシステムなどの業務アプリケーションの開発方法としては、M-base[9] 等のエンドユーザ向け開発環境を利用した開発を想定している。

サンフランシスコフレームワーク [5),6)] は、小・中規模の業務アプリケーションを対象とし、基本的なビジネス・ロジックおよび分散オブジェクト等を提供する業務アプリケーションフレームワークである。このフレームワークのサポートするドメインの典型的なアプリケーションでは、全体の約 40 %をフレームワークがカバーし、アプリケーション開発者は残りの 60 %を開発している。

備品管理システム全体では、フレームワークの占める割合は約 70 %であるが、本システムのサポートする窓口業務アプリケーションでは、フレームワークが約 90 %、備品管理システム固有の差分コンポーネントが約 10 %を占めている。フレームワークの比率が多い理由として、窓口業務は個々のアプリケーションに依存しない共通部分が多いことがあげられる。業務によっては記入内容チェックの比率が多くなることがあるが、5.2 (2) で述べたようにプロパティデータの設定により業務の専門家自らが記入フォームに埋め込み可能である。したがって、窓口業務はフレームワークを用いてエンドユーザ主導でアプリケーションを開発するのに適した分野と言える。

6. おわりに

今回、例題として研究室内での図書管理システムを開発し、窓口業務アプリケーションフレームワークを抽出した。そして、そのフレームワークを実際に他の例題システムに適用し、業務の専門家が窓口業務アプリケーションを構築できることを確認した。

今後は、業務の専門家および一般の窓口利用者という 2 種類のエンドユーザの視点での使い勝手を向上させるツール群の開発、およびフレームワークの他の例題への適用について検討していく。

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致します。

参考文献

1) Wolfgang Pree: Design Patterns for Object-Oriented Software Development, ACM Press, (1995). 邦訳: 佐藤, 金澤: デザインパターンプログラミング, トッパン (1996).
2) Steve Sparks, Kevin Benner, and Chris Faris : Managing Object-Oriented Framework Reuse, IEEE Computer 29, 9, 52-61 (1996).
3) Mohamed Fayad, and Douglas C. Schmidt: Object-oriented application frameworks, Communications of the ACM, 40, 10, 32-38 (1997).
4) Ivar Jacobson, Maria Ericsson, and Agneta Jacobson: The Object Advantage, Bussiness Process Reengineering with Object Technology, ACM Press (1995). 邦訳: 本位田: ビジネスオブジェクト ユースケースによる企業変革, トッパン (1996).
5) IBM Shareable Frameworks, The San Francisco Project, IBM Corp, http://www.ibm.com/java/Sanfrancisco/
6) 吉田武稔, 志村功, 田中一郎: ビジネスオブジェクトとその実現のための基盤技術, 情報処理, 39, 2, 81-85 (1998)
7) 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).
8) 藤原克哉, 斉藤裕樹, 中所武司: 多組織間オフィスネットワークにおける分散アプリケーションフレームワーク wwHww のアーキテクチャと例題への適用, 情報処理学会ソフトウェア工学研究会資料, 97-SE-115-9, 65-72 (1997).
9) Takeshi CHUSHO, Yuji KONISHI and Masao YOSHIOKA : M-base : An Application Development Environment for End-users Computing based on Message Flow, APSEC'96, IEEE Computer Society, (1996).