ソフトウェア工学　139－7
(2002. 10. 11)

Webアプリケーションにおける予約業務フレームワークの抽出実験と再利用性の検証

津久井　浩†　　中所　武司†

† 明治大学理工学研究科基礎理工学専攻ソフトウェア工学研究室
E-mail: †{tsukui,chusho}@cs.meiji.ac.jp

あらまし　業務の専門家自らが情報システムを構築する必要性が高まっている。エンドユーザ主導のWebアプリケーション開発技法の研究の一環として、予約業務フレームワークをとりあげ、アプリケーションの具体例として開発した会議室予約システムと座席予約システムから共通部分を抽出した。両システムは予約単位が異なるものであるが、予約対象となる資源の情報と予約に必要な情報の内容に依存する箇所が多く、それらの箇所はこの2つの情報の定義から自動生成できることがわかった。最終的な再利用率は会議室予約システムが43％、座席予約システムが68％であった。

キーワード　フレームワーク, 予約業務, Webアプリケーション

Web Application Framework for Reservation Systems and its Reusability

Hiroshi TSUKUI† and Takeshi CHUSHO†

† Computer Science Cource, Major in Sciences,
Graduate School of Science and Technology, Meiji University
E-mail: †{tsukui,chusho}@cs.meiji.ac.jp

Abstract　The number of end-users using the Internet increases on the inside and outside of offices. End-duser-initiative development of web applications has become important for automation of their own tasks. For this purpose, an application framework approach using Struts, is tried. First, a room reservation system and a seat reservation system are developed. Then the framework is extracted and its reusability is evaluated. As a result, it is confirmed that 43% and 68% of codes are shared on these two reservation systems respectively.

Key words　framework, reusability, web application

1. はじめに

近年、インターネットやイントラネットに接続されたパソコンの普及と共に、オフィス業務の効率化という観点から、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[1]。たとえば、基幹業務システムのように投資に対する効果が明確なものは、従来のように情報処理の専門家が開発すればよいが、小さな部門あるいは個人の業務を対象とするものや頻繁に機能変更が発生するものは従来の開発方法はなじまない。このような分野では、Webアプリケーションが適切な場合が多いが、コーディング経験のない業務の専門家がゼロからWebアプリケーションを構築することは非常に難しい。

一方、最近の情報システム構築では、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ことが追求されている。特に問題領域に特化したアプリケーションフレームワークは、その領域のアプリケーション構築を行なう時の開発効率の向上が図れるので、実用化が進んでいる。

そこで本研究では、エンドユーザ主導のWebアプリケーション開発技法の研究の一環として、予約業務フレームワークをとりあげた。予約業務は、施設の予約やチケットの予約などが挙げられるが、紙や電話による従来の手続きをWebブラウザ上で実現することで、手間や人件費の削減など利用者と運営者の両方に高い利益をもたらす業務の1つである。フレームワーク構築のためのプロセスとして、最初にアプリケーションの具体例に2つ選択し、実際に構築することで共通部分の分析を行ない、フレームワークの抽出を行なった。具体例には、会議室予約システムと座席予約システムを選択した。2つの予約業務の予約単位は、会議室予約が空間（会議室）と時間（時限）に対

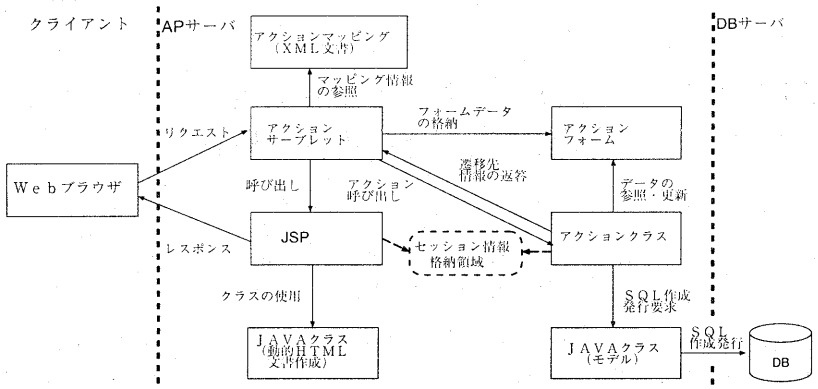

図 1　MVC モデル 2 に基づくシステムアーキテクチャ

し、座席予約は時間（イベント）と空間（座席）となる。このように予約単位の違う 2 つの業務をプログラム上で比較することで、会議室予約などを含めた施設予約以外の予約業務にもフレームワークの適用が行なえるように考えた。

本稿では、最初に 2 つのシステムに適用したシステムアーキテクチャについて述べ、次に会議室予約システムと座席予約システムの機能の違いと DB テーブルの違いの比較結果を述べる。そして、抽出されたフレームワークについて述べ、予約業務 Web アプリケーションの再利用性の検証について述べる。

2. システムアーキテクチャ

頻繁な機能変更に対応するためには、高い保守性を持ったシステムアーキテクチャが重要となってくるので、MVC モデルを Web に応用したもの（MVC モデル 2）を適用した。具体的には、Java クラスをモデル、サーブレットをコントローラ、JSP をビューとする Struts フレームワーク [3] を適用することで、図 1 のようなシステムアーキテクチャを構築した。システムを構成する各要素について述べる。

アクション・サーブレットは、アプリケーションの中に 1 つだけ存在し、すべての HTTP リクエストを管理する。送られてきたリクエストを最初に受け取り、その内容を解析し、その他のサブシステムへ指示を送る。通常は Struts で提供されたクラスを、そのまま利用する。アクション・フォームは、Web ページ上でのフォーム情報を扱うクラスで、Struts が提供したクラスを拡張することが前提となる。ユーザが入力したフォームの値を格納しておく。アクション・マッピングは、URL とアクションのマッピング情報を保持する XML 文書ファイルである。アクション・サーブレットは、このファイルを参照することでアプリケーションフローを制御する。アクション・クラスは、ビジネスロジックを呼び出したり、記述したりするクラスである。Java クラス（モデル）は、アクション・クラスからの処理依頼を受け、ビジネスロジックを実行するクラスである。SQL 文の作成や発行を行ない、必要に応じて結果をアクション・クラスに返す。Java クラス（動的 HTML 文書生成）は、表示情報の動的生成を伴う HTML 文書の作成を行なうクラスである。例えばカレンダー作成や個人の予約リスト作成を行なうクラスがある。

3. 比較実験

3.1 システム概要

会議室予約は、従来の事務室のスケジュール表への書き込みという方式を Web ブラウザ上で実現するために構築したものである [6]。概要は以下の通りである。

- 対象ユーザ

  教員（一般利用者）と事務員（運用管理者）
- 対象資源

  情報科学科で使用される会議室（現在 4 室）
- 予約単位

  1 日の中で区切られた 1 ～ 6 限までの枠単位

会議室予約では、最初にユーザが ID とパスワードによるログインを行なう。その後、利用する機能を選択する。例えば予約時には、該当する会議室を選択した後に図 2 の週間カレンダー画面から時間を指定して、予約を行なう。

図 2　予約時の画面（会議室予約）

座席予約は、以下のような状況を仮定して構築した。

- 対象ユーザ
  インターネットを利用する人
- 対象資源
  ある施設の中で行なわれる行事（以下イベントと呼ぶ）
- 予約単位
  3×3に区切られた座席

座席予約は、該当するイベントを選択した後に座席選択画面から座席を指定する。座席予約では、ログインによるユーザ認証処理を行なうのではなく、予約時にユーザ情報を同時に入力してもらうようにした。そして、予約完了時にランダムで一意な予約IDを発行し、それをユーザに保持してもらうことで、取消を行なえるようにした。

3.2 機能の比較

会議室予約システムが持つ機能と座席予約システムが持つ機能を一般利用者が利用するものについて比較した。

- 共通機能
  予約・取消・予約状況照会
- 会議室予約システム固有の機能
  ログイン・ユーザパスワードの変更
  個人予約状況照会
  定期予約
- 座席予約システム固有の機能
  なし

予約業務において、最低限必要となる機能は「予約」「取消」「照会」の3つである。共通機能の「予約状況照会」は、会議室予約システムにおける会議室の空き状況、座席予約システムにおけるあるイベントの座席空き状況の事を言う。会議室予約システムは、ユーザ情報を管理するサブシステムを有し、ログイン・パスワードの変更機能を実行する。また、ユーザ単位での管理を行なうことより、個人予約状況照会という機能が可能となる。定期予約は、指定された期間の指定された曜日・時限をまとめて予約するものである。

3.3 DBテーブルの比較

会議室予約システムと座席予約システムのDBテーブルの比較を図3に示す。

予約情報の管理は、予約単位を1レコードとして保存していく方式である。つまり、実世界における予約申し込み伝票をそのまま原簿として保存することに対応する。どちらの予約情報テーブルも予約の対象資源となる会議室・イベントに関するテーブルが存在し、このテーブルに存在する各々の資源の名前を予約情報テーブルは外部キーとする。また、予約情報テーブルでは、一意となる情報が必ず存在する。会議室予約は、会議室名と予約時間の2つの組み合わせが一意となる。座席予約は、イベントIDと座席名の組み合わせが一意となる。ただし座席予約は一意な予約IDを発行するので、これをキーとして処理が行なわれる。また会議室予約にのみ、ユーザ情報を管理するテーブルと大学の休日や祝日に関する情報を管理するテーブルがある。

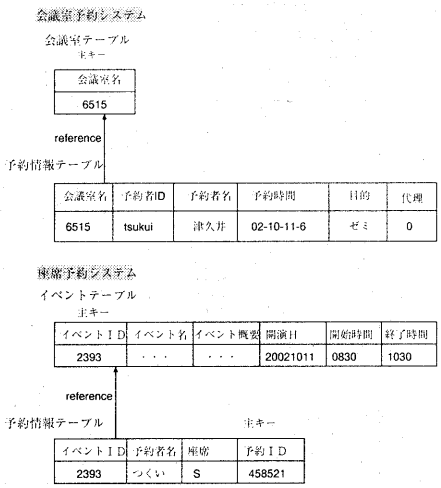

図3 DBテーブルの比較

4. 抽出された予約業務フレームワーク

抽出された予約業務フレームワークを図4に示す。
以下にその詳細について述べる。

4.1 DBテーブル

3.3節のDBテーブルの比較から、予約業務では予約の対象となる資源に関するテーブルと、予約のために必要な情報に関するテーブルが抽出された。ここで資源テーブルは、列に資源名が入る。予約情報テーブルは、対象資源・予約者名・時間名（空間名）の3つの列が入る。また対象資源は、資源テーブルの資源名を外部キーとして設定している。その他の列は、業務で固有に必要となるものである。

4.2 Javaクラス（モデル）

Javaクラス（モデル）では、以下のクラスが抽出された。

- 予約情報を定義したデータ型クラスReserve
- Reserveオブジェクトを利用して、予約処理などを行なうReserveManagerクラス
- 資源情報を定義したデータ型クラスResource
- Resourceオブジェクトを利用して新しい資源の追加や削除を行なうResourceManagerクラス
- SQL文を発行するDBHandlerクラス
- DBへのコネクションを管理するDBConnectionHandlerクラス

Resource・Reserveクラスは、DBテーブルで設計した資源テーブル・予約情報テーブルをプログラム上で扱うためのデータ型クラスなので、DBテーブルに完全に依存する。ReserveManagerは、予約・取消・照会処理を行なう。これらの処理は、SQL文作成が主となり、Reserveクラスが持つ変数に依存する。ResourceManagerは、資源の登録・削除・照会処理があり、SQL文作成が主となり、これらはResourceクラスが持つ変数に依存する。DBHandlerクラスは、DBへの更新・検索の通知処理があり、変更処理は完全に共通な処理として抽出さ

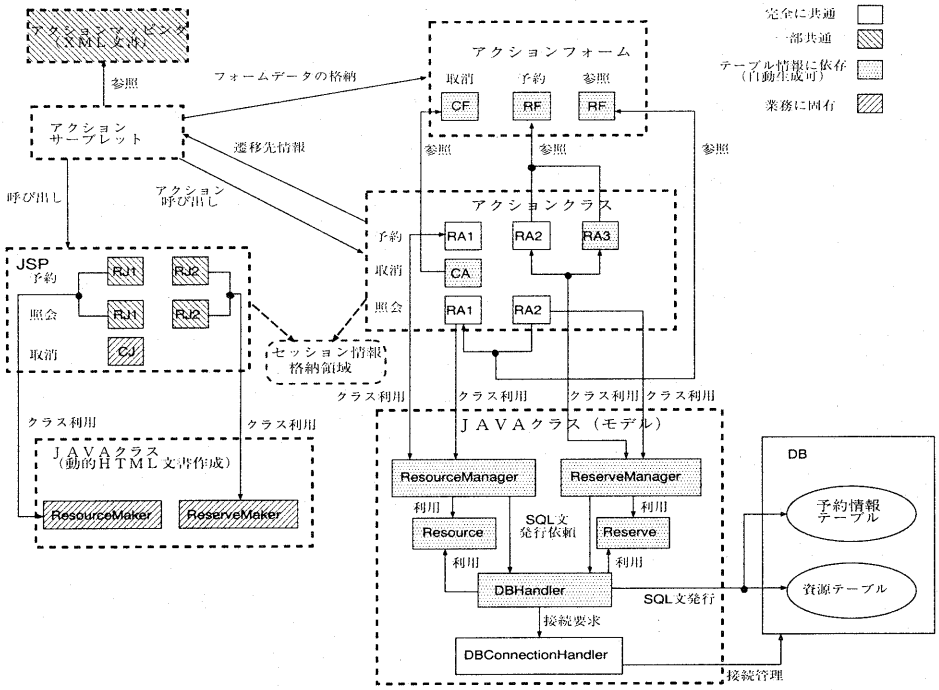

図4　抽出されたフレームワーク

れた。検索の場合には戻り値が必要となるが、この時の戻り値は、Reserve オブジェクトと Resource オブジェクトを格納した配列であるので、予約情報テーブルと資源テーブルに依存する。DBConnectionHandler クラスは、処理内容が完全に共通であったので、カスタマイズ箇所は存在しない。

4.3　Java クラス（動的 HTML 文書生成）

Java クラス（動的 HTML 文書生成）で抽出したクラスは、ResourceMaker と ReserveMaker の2クラスである。ResourceMaker は、資源の一覧を表示し選択させるクラスである。会議室予約ではセレクトタグによる選択、座席予約では一覧をリスト表示し、ラジオボタンによる選択を行なうためのクラスである。ReserveMaker は、カレンダーの作成や座席表の作成など予約状況を表示するためのクラスである。また ReserveMaker は、予約時のフォーム有りと照会時のフォーム無しのいずれかを指定できるようになっていた。2つのクラスは、JSP から必要な情報を受け取る Setter メソッドと作成した HTML 文書を返す Getter メソッドが抽出された。しかし、メソッドの処理内容に関しては、完全に業務に固有の記述となっていた。よって、メソッドのみを宣言したスケルトンクラスとなり、その処理内容は業務に固有となる。

4.4　アクション・クラスとアクション・フォームと JSP

アクション・クラスとアクション・フォームと JSP は、予約・取消・照会機能ごとに抽出結果を述べる。

4.4.1　予　　約

両システムの予約の手続きは、ユーザの視点からみて

（1）　予約の対象となる資源を選択する手続き

（2）　選択した資源の時間（空間）を選択する手続き

の2つの処理からなるが、システム内部の視点では

（1）　予約の対象となる資源を選択させる Web ページの作成

（2）　対象となった資源の予約状況を表示し枠を選択させる Web ページの作成

（3）　予約処理

の3つの処理に分かれた。

そこで、これらに対応する以下のアクション・クラスを抽出した。

(RA1)　ResourceManager から資源情報を取ってくる処理

(RA2)　ユーザが選択した資源をキーとしてその予約情報を ReserveManager から取ってくる処理

(RA3)　予約に必要な情報を集め Reserve オブジェクトを生成し、ReserveManager に予約依頼する処理

RA1・RA2は、全て共通した記述であった。RA3は、Reserve オブジェクトを生成するために必要な情報を集める箇所が業務に固有の記述となる。

JSP は、2つのクラスを抽出した。

(RJ1)　どの資源を予約するかを選択するための Web ページ

(RJ2)　対象資源の予約状況を表示し、枠を選択するための Web ページ

RJ1 は、セッション情報格納領域から資源情報（Resource オブジェクトの格納された配列）を受け取って、Java クラス（動的 HTML 文書生成）の ResourceMaker に渡す処理と、結果の HTML 文書を受け取る処理が共通の記述として抽出された。

RJ2は、セッション情報格納領域から予約情報（Reserveオブジェクトの格納された配列）を受け取って、Javaクラス（動的HTML文書生成）のReserveMakerに渡す処理と、結果のHTML文書を受け取る処理が共通の記述として抽出された。その他、ページのレイアウトや操作性向上のために設置したフォームなどは、業務に固有の箇所となった。

ユーザから入力されたフォームの値を格納するアクション・フォームは、両システムとも1クラスのみの構成であったので、そのクラスRFを抽出した。このRFは予約手続きが開始してから完了するまでの間にユーザから受け付けた入力情報を保持する。これは予約処理を行なうため、つまりReserveオブジェクトを生成するために必要な情報である。よって、このクラスはReserveクラスが持つ変数に依存する。

4.4.2 取　　消

取消手続きは、両システムで手順が異なっている。会議室予約では、ログインした時のIDを元に本人が予約している情報を一覧提示し、その中からユーザが選択する。座席予約では、予約IDを直接テキストに入力してもらう。よって、取消情報を取得するためのJSPクラスCJ1は、業務に固有の箇所となる。アクション・クラスは、Javaクラス（モデル）のReserveManagerに取消依頼を行なうCAを抽出した。取消は、予約情報の一意な項目をもとに処理を行なうので、CAは予約情報テーブルで設計した一意な列に依存する。

アクション・フォームは、予約と同じく両システムとも1クラスのみで構成されているので、そのクラスCFを抽出した。CFは、どの予約情報を取り消すかということを特定するための一意な情報を保持するので、CA同様に予約情報テーブルで設計した一意な列に依存する。

4.4.3 照　　会

照会は、何をキーとするかで機能が分かれる。ここで共通した照会機能は、会議室予約状況照会とイベント空き状況照会のように資源名をキーとしたものであった。資源名をキーとした場合のシステム処理手順は、4.4.1節の予約の処理手順の予約状況を表示するWebページ作成までは、使用したクラスが同じである。よってアクション・クラスは、予約で抽出したRA1とRA2を抽出した。JSPも同様にRJ1・RJ2を抽出した。ただし、RJ2でReserveMakerを使用する箇所は、フォーム無しを指定するように記述している。

また、アクション・フォームもRFを再利用した。

4.5 アクション・マッピング

アクション・マッピングは、ページフローを記述したファイルで、アプリケーションの中に1つだけ存在する。ファイルの中で、予約と照会のページフローの一部は同じであったので、この部分を抽出した。それ以外の箇所は業務に固有となる。

4.6 アクション・サーブレット

アクション・サーブレットは両システムともStrutsで提供された1クラスが抽出された。このクラスは、カスタマイズ不要である。

4.7 システムの流れ

抽出されたフレームワークを適用した際のシステムの流れを示す。例として予約手続きのなかで、ユーザが予約する対象の資源を選択してから、その資源の予約状況を表示するWebページの作成を行なうまでの手順を示す。

（1） 選択された資源がRFにセットされる
（2） RA2がJavaクラス（モデル）のReserveManagerに選択された資源の予約情報(Reserveオブジェクトの格納された配列)を全てとってくるように依頼する
（3） RA2は受け取った予約情報をセッション情報に格納する
（4） アクション・サーブレットがアクション・マッピングを参照し、RJ2にフォワードする
（5） RJ2はセッションに格納されている予約情報を受け取り、ReserveMakerに渡す
（6） ReserveMakerはその予約情報をもとにHTML文書を作成し、RJ2に返す
（7） RJ2は作成されたHTML文書を受け取りクライアントに返す

5. フレームワークの再利用性の検証

5.1 データの定義

最初にDBテーブルの設計によって資源テーブルと予約情報テーブルの各列の定義、一意となる情報、外部キーの設定を行なう。また、プログラム上で使用する各列の変数名を定義する。そして予約情報は、定義した各列の値をユーザがいつ入力するかを定義する。その入力方法は、以下の3つに分けられる。

- 予約時に入力
- 予約時以外に入力
- 不要

予約時以外に入力する情報とは、ログイン時など予約を行なう前に別の機能で既に入力されている情報をいう。不要は、システム内部の処理によって決定される情報をいう。例として会議室予約システムと座席予約システムで定義された予約情報を図5、図6に示す。

予約情報	会議室名	予約者ID	予約者名	予約時間	目的	代理
一意	○	←組み合わせ→		○		
外部キー	○					
変数名	roomName	userID	userName	reserveTime	purpose	proxy
入力方法	予約時	ログイン時	不要	予約時	予約時	不要

図5　会議室予約システムの予約情報

予約情報	イベントID	予約者名	座席	予約ID
一意				○
外部キー	○			
変数名	eventID	userName	seatName	reserveID
入力方法	予約時	予約時	予約時	不要

図6　座席予約システムの予約情報

	サブシステム	Javaクラス（モデル）	アクションクラス	アクションフォーム	JSP	Javaクラス（動的HTML文書生成）	アクションマッピング	合計
会議室予約	全体	740	370	170	300	640	100	2320
	フレームワーク	460	280	140	60	0	60	1000
	業務に固有	280	90	30	240	640	40	1320
	フレームワークが占める割合	62%	75%	82%	20%	0%	70%	43%
座席予約	全体	550	300	90	260	130	90	1420
	フレームワーク	480	270	90	60	0	60	960
	業務に固有	70	30	0	200	130	30	460
	フレームワークが占める割合	87%	90%	100%	23%	0%	67%	68%

図 7　ステップ数におけるフレームワークの割合

5.2　自動生成可能な箇所

Java クラス（モデル）で抽出された DBConnectionHandler クラスを除く 5 クラスは、4 節の抽出結果から資源情報と予約情報にのみ依存していることが分かったので、これらのクラスはすべて自動生成が可能となる。

予約時に使用するアクション・フォーム RF とアクション・クラス RA3 は、5.1 節で述べた予約情報の入力方法の定義から自動生成することが可能となる。具体的には、RF は予約時に入力すると定義された項目のみを変数として宣言する。RA3 は Reserve オブジェクトを生成するために必要な情報を以下のように取得する。予約時に入力すると定義された情報は RF から取得する。予約時以外で入力すると定義された情報は、事前に入力されている情報なのでセッション情報格納領域から受け取る。不要と定義された情報は、システム内部の処理によって決定されるので、Reserve オブジェクト生成前にエンドユーザがその処理を記述する。

取消時に使用するアクション・フォーム CF とアクション・クラス CA は、予約情報の一意な項目の定義より自動生成が可能となる。

また、アクション・フォームクラス内での変数名と、セッション情報格納領域に登録するときの変数名は、5.1 節の資源情報と予約情報で定義された変数名がそのまま使用される。

データの定義によって、Java クラス（モデル）とアクション・クラスとアクション・フォームの 3 つのサブシステムの、予約・取消・照会に必要な処理は自動的に構築できることがわかった。

5.3　ステップ数によるフレームワークの割合

今回抽出したフレームワークが、既存に開発した 2 つのシステムにおいて、どの程度の割合を占めるかをステップ数によって検証した。比較は、予約・取消・照会機能で使用したクラスのステップ数の合計を、各サブシステムごとに計算した。ただし、クラスの中で予約・取消・照会処理に直接関係しない処理は、ステップ数に含めていない。結果を図 7 に示す。

会議室予約では、Java クラス（動的 HTML 文書生成）のカレンダーを作成する ReserveMaker と、Java クラス（モデル）の休日管理や時間の計算処理などのステップ数が大きかったため、フレームワークの占める割合が全体として低くなってしまった。また、予約手続きにおいて、操作性向上のための処理や、予約確認画面の挿入に伴う処理などがアクション・クラスに記述されていたために、アクション・クラスの割合も少し低めとなった。一方座席予約では、Java クラス（モデル）とアクション・クラスとアクション・フォームの部分のみでは、平均して 90 %近い部分がフレームワークによって構築されている。

全体の再利用性を大きく下げてしまったのは、ビューの役割を果たす JSP と Java クラス（動的 HTML 文書生成）である。JSP は、エンドユーザのページデザインなどにも依存するので、この箇所は、エンドユーザ自身に構築してもらうのが理想となる。

もし、カレンダーや座席の指定をテキスト形式で入力するのならば、Java クラス（動的 HTML 文書生成）は、必要ない。しかし、予約業務では通常カレンダーや座席の画面を表示し、枠を選択して予約を行なうものである。テキスト形式の予約は、利用者にとって使い勝手の悪いアプリケーションとなる。よって、Java クラス（動的 HTML 文書生成）の構築をエンドユーザにどのように支援するかが今後の課題である。

6.　おわりに

本稿では、会議室予約システムをベースに作成した座席予約システムとの比較実験を行ない、フレームワークの抽出及び再利用性の検証を行なった。今後は Java クラス（動的 HTML 文書生成）の支援方法を検討する。そしてフレームワークを構築し、他の予約業務への適用実験と評価を行なう。

文　献

[1] Takeshi Chusho, Katsuya Fujiwara, Hisashi Ishigure and Kei Shimada : A Form-based Approach for Web Services by Enduser-Initiative Application Development, SAINT2002 Workshop (Web Service Engineering), IEEE Computer Society, pp.196-203 (Feb. 2002).

[2] 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,41,4,1202-1211(2000)

[3] The Ja-JakartaProject http://www.ingrid.org/jajakarta/struts/

[4] ジム・コナレン：UML による Web アプリケーション開発 ピアソン・エデュケーション (2000)

[5] 原田洋子:Java サーバサイドプログラミング 技術評論社 (2001)

[6] 津久井浩: 3 層 Web アプリケーションの実用実験による変更容易性の評価 FIT(2002)