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

2005－IS－91（16）
2005／3／16

フレームワークを利用した Web型ビジネスアプリケーションの開発

山本瑛安　黒田充紀　小嶋聡史　佐野哲平　寺地亮人　平山欣央　松永賢次　本江渉
（専修大学　ネットワーク情報学部）

10 名の学生による 1 年間のプロジェクト学習活動により，Web 型ビジネスアプリケーション開発を行った実例を報告する。財務会計システムと販売管理システムから構成されるアプリケーションを，基本設計の策定から実装・テストまでのサイクルを二周行って作成した。オープンソースで提供されている Web アプリケーションサーバで動作するフレームワークとデータベースを利用したシステムになっている。これらのツールを利用することで，学部学生でも実用的なアプリケーションを開発できることを示し，ツールの学習コストも開発効率の向上に見合うことを示した。

Developing a Web-based Business Application with Framework Technology

Eian Yamamoto, Mitsunori Kuroda, Satoshi Kojima, Teppei Sano, Yoshito Terachi,
Yoshio Hirayama, Kenji Matsunaga and Wataru Motoe
Department of Network and Information, Senshu University

This paper demonstrates a Web business application that 10 university students developed through a project-based learning activity. The application, which is comprised of a financial management system and a sales management system, has been developed through two cycles of designing, coding and testing process. Frameworks on a Web application server and database system provided as open source software are utilized in a part of the application. With these open source software, university students can develop a practical application and learning-cost of open source software is below efficiency improvement.

1. はじめに

現在，WEB 上では数多くのオープンソースが提供され，システム開発の際は欠かせない資源となっている。これらオープンソースは頻繁に機能や性能の強化・変更がなされるため，開発者は情報収集などによる迅速な対応が不可欠となる。書籍による情報のみならず，インターネットによる最新かつ精度の高い情報収集が，開発工程やシステムの品質に大きく影響する。

我々は，大学の授業の一環で，「ERP のシステム開発」を研究目的とした活動を一年間行った。所属する専修大学ネットワーク情報学部は，経済・経営・商学と情報科学とが一体となった文理融合型の学部である。1，2 年次では，情報分野に関する基本的な知識やスキルを習得する。3 年次の必修科目として「プロジェクト１」という授業がある。この授業は，学生および教員が様々なテーマを提案する。担当教員 1 名（もしくは 2 名）とそのテーマに興味・関心を持った学生 5 人～10 人前後でグループを編成する。そして，1 年間の期間の中で，各グループがテーマに沿った研究や問題解決，作品製作に取り組む。従来の講義形式の授業とは異なり，あらかじめ具体的な学習内容が定められていない。学生と担当教員で，相互に意見やアイデアを出し合い，成果物を作り上げていくものである。

本稿では，このプロジェクトを通して行った活動について，開発過程や経緯，開発方法，周辺技術の学習方法とそのコストについて述べる。

2. 開発体制

我々のプロジェクトは「ERP のシステム開発」を研究目的とする。メンバーは，会計学に精通した担当教員 1 名，大学 3 年次生 10 名で構成される。学生は，Java を得意とする者，アルゴリズムを得意とする者，ネットワークを得意とする者，ユーザインタフェースやデザインを得意とする者，会計に詳しい者，また技術以外の面では，コミュニケーション能力に優れた者，モチベーションの向上を図る者など，個性豊かな学生が揃った。学生は，情報分野に関する基礎知識はもちろん，Java による開発を 1 年以上経験した者である。

表 1：開発スケジュール

期間	概要
1 月	テーマ決定
2～4 月	春期休暇（各自，会計や ERP について学習）
4～6 月	システムに盛り込むべき機能，システムの画面遷移デザイン，開発手法や手順の設計，および Struts フレームワークの学習
6～7 月	プログラムの実装，中間報告
8～9 月	夏期休暇（各自、自習）
9～10 月	Hibernate の組み込み，再度設計
10～12 月	Hibernate の組み込み，実装，デバッグ，デモ会
翌年 1 月	最終発表

プロジェクトは，財務会計システムと販売管理システムを開発する 2 グループに別れた。それぞれ，設計／デザイン担当，プログラム担当，その他技術系（Struts, Hibernate, Tomcat など）の担当と分担し開発を行った。財務会計システムと販売管理システムは，ひとつの ERP のシステムであるため，両グループとも，相互に不整合が生じないようにコミュニケーションを意識し，共通のパッケージ開発に取り組んだ。

システム開発では，メンバー間の意見交換，プロジェクトに対するモチベーションの持続，時間の有効活用が不可欠である。そのため受講する講義時間が終わり次第，コンピュータ室に皆が集まり，雑談を交えながら互いにコミュニケーションを取っては開発を行った。しかしシステムを完成させるためには，大学にいる時間だけでは足りない。メンバー各自が自宅などでも作業を行う必要がある。その際に，掲示板や MSN メッセンジャー，Skype といった各種コミュニケーションツールを利用した。掲示板は，スケジュール管理やファイル管理，また議論の場として利用した。各個人の仕事に対しての連絡事項の通達や，早急なレスポンスを求めるときは MSN メッセンジャーを利用した。複数人で音声による討論をしたい場合には Skype を利用した。これらのツールを活用することで，プロジェクトにおける活動範囲を自宅にも広げ，スケジュールをより円滑に進行できた。

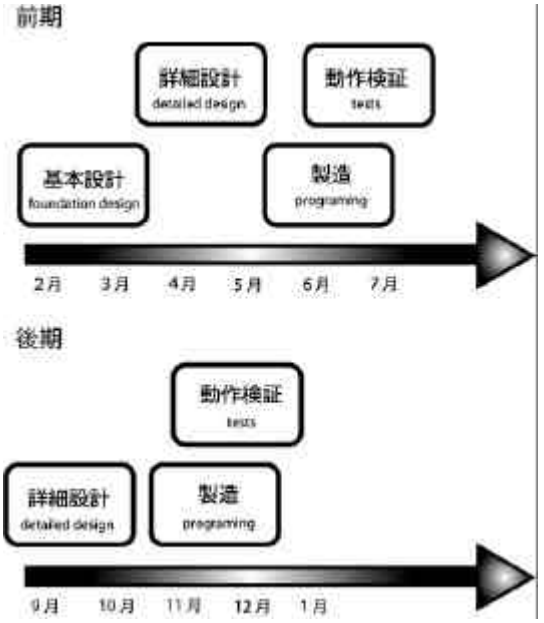

図 1：開発サイクル

3. スケジュール

プロジェクトの活動期間はおよそ 1 年間である。開発スケジュールは，表 1 の通りである。大学の学事暦に合わせ前期（4 月～7 月）と後期（9 月～12 月）の半期を区切りとし，設計，実装，テストを 1 サイクルとして，2 回のサイクルでスパイラルモデルに基づく開発を行った（図 1）。前期では，春季休暇中の勉強を踏まえ，システムの設計や開発技術に関する基礎勉強に励んだ。これらの知識をもとに実践を積み，勉強と実践を繰り返しながら，システムの完成を目指した。後期は，再度設計のやり直しと開発技術の見直しを行い，開発に努めた。

また，年間を通して週に一度担当教員を含めて話し合いを行った。ここでは，両グループの活動報告や今後の活動や進捗状況について教員からの指示を仰ぎ，フィードバックしながら開発を進めた。

4. 要求仕様

4.1 開発の目的

大学の講義の一環であるプロジェクトでは教員による数多くの提案テーマが掲げられた。我々は「ERP のシステム開発」のテーマに賛同しプロジェクトが結成された。このテーマの下に集まった理由は人それぞれである。開発する中で自分のスキルアップを図りたい者，会計の知識を生かしたい者，など様々な考えを持ったメンバーが集まった。その考えの中心にあるのは企業情報システムを開発することで，業務処理や経営戦略にも触れられ，その経験を今後に生かせるということである。情報システム，コ

ンピュータサイエンスのみならず，幅広く学べるチャンスであった。テーマを先に決まった上でメンバーが集まったので，開発目的の基幹にあるものは最初からひとつに統一されていた。

4.2 開発したシステムの概要

プロジェクトの期間は 1 年と限られている。そのため業務で本格的に利用できる ERP パッケージの開発は困難である。そこで，我々のプロジェクトでは，財務会計システムと販売管理システムの開発に絞り活動を行うこととした。

当システムは，スーパーマーケットでの業務を想定し，主として食料品や日用品を扱い仕入・販売を行うものである。LAN に繋がる PC 端末からユーザ認証することで，サーバ上で動作する当システムを利用できる。端末の WEB ブラウザを介してシステムにログインし，業務を行う。ユーザは，経理を担当する者，販売を担当する者など業務に携わるものを想定し，端末のあるあらゆる場所での業務を可能とする。

財務会計システムは，企業における経済活動を把握することが主となる。そのためには，資金がどのように使われたかを入力すること，それを財務諸表で出力することが必要である。

一方，販売管理システムは，商品の発注，売上情報などを管理することが主となる。そのためには発注の入力機能や売上情報を時系列で管理するような機能が必要である。

双方のシステムは独立したものではなく，1 つのパッケージソフトウェアとして動作するために，販売管理システムで売上を計上した時点で財務会計システムに売上データを渡すなどのシステム間の連携が必要である。両システムの連携を図ることで，企業は売上のデータをわざわざ両システムで入力する手間を省くことができ，これにより両システムでのデータの不整合を防ぐこともできる。このように ERP の中でも財務会計システムと販売管理システムに焦点を当て，それらを連携するシステムを開発する。

4.3 機能的要件

市販された財務会計／販売管理プログラムのパンフレットから機能を洗い出した。システムの機能的要求は図 2 の通りである。利用者はシステムにログインするとメニュー画面より財務会計，販売管理のいずれかを選択する。

財務会計システムでは，伝票形式や出納帳形式などの形式から仕分けデータの入力を行う。そして，必要に応じて集計，決算し財務諸表を出力することができる。また，会計期間や勘定科目の追加といった設定も行うことができる。

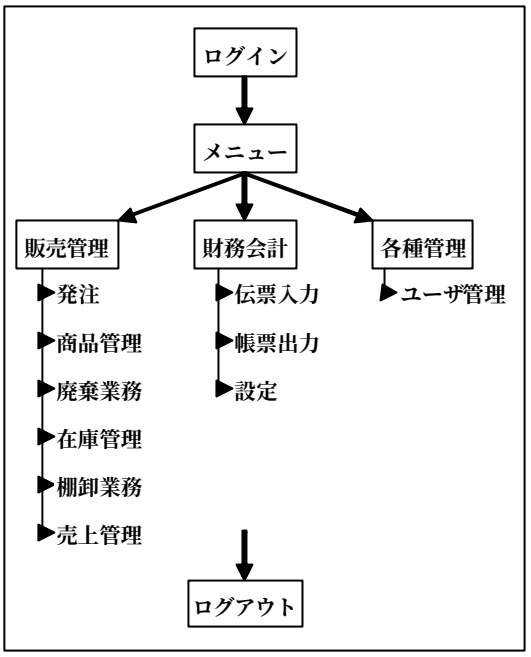

図 2：機能的要求

販売管理システムでは発注，商品管理，廃棄，在庫管理，棚卸業務，売上業務，そして取引先管理といった販売に関わる一連の操作を行うことができる。

また，販売管理システム，財務会計システムともに，出力される帳票や発注書などは PDF に変換して生成する機能や，売上データをグラフにして参照することができる機能をもつ。

4.4 非機能的要件

業務システムは，多くの場合，業務上 24 時間常に稼動し続けることが要求される。システムに障害が起こった場合，業務に支障をきたし多大な損害が生じる事態になりかねない。そのような万が一の事態を避けるためには，システムが停止することなく，それまでと変わりなく業務が続けられることが必要となる。

また，複数人での利用が前提にあるため，ある程度の負荷に耐えうるサーバのスペック，効率のよいプログラムの記述が求められる。

4.5 システムのアーキテクチャ

本システムのアーキテクチャを図 3 に示す。アーキテクチャ決定の経緯は以下の通りである。

アプリケーションをクライアント部分に焦点を当てて考えると 2 つの方法が考えられる。WEB ブラウザ型クライアント（以下 WEB 型）と端末機に専用アプリケーションのインストールされたクライアントである。この 2 つを検討した結果，次のようなメリットから WEB

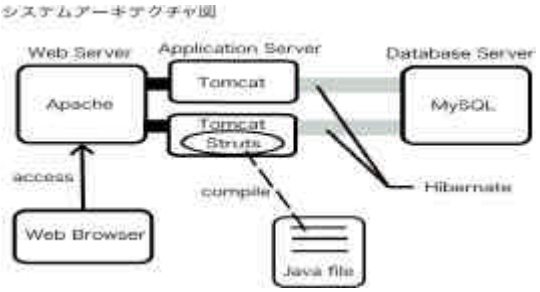

図 3：システムアーキテクチャ図

型を採用することとした。

- プラットフォームを問わない。
- 環境構築が容易である。
- 開発経験のある Java 言語の利用が可能である。
- システムの管理・保守が容易である。

しかし，ユーザインタフェースにおける表現の限界，負荷集中によるリスク，セキュリティのコストなど，いくつかのデメリットもある。

WEB 型では更に.NET，PHP，Java の実装技術の選択肢があったが，扱いなれているという点や，フリーのアーキテクチャで構成できるといった点から Java を選択した。アプリケーションサーバ，データベースサーバも同じような理由から Tomcat，MySQL に決定した。プロジェクトでは予算は少なく限られており，フリーで利用できるということがシステムアーキテクチャ決定の大きな要因となっている。アプリケーションフレームワークとして Struts，O/R マッピングフレームワークとして Hibernate を採用した。PDF の生成には iText，グラフの生成には JFreeChart を利用した。開発環境として Eclipse.org が開発した Eclipse を利用し，Tomcat，Struts などのプラグインをインストールして利用した。

前述した非機能的要件を満たすため，Tomcat は 2 台のサーバを使用して負荷分散を行っている。これにより，アベイラビリティの高いシステム構築を実現し，業務の遂行を安定して行うことを目指した。

5. オープンソースの利用について

前節で述べとおり，本システムはオープンソースのソフトウェアを利用して作成されている。本節では利用したオープンソースソフトウェアである Struts([1, 2, 3, 4])，Hibernate([5, 6, 7, 8, 9, 10]，Tomcat([1, 11])についての概要と使用した評価を示す。

5.1 Struts

5.1.1 概要

Struts は Apache Software Foundation によって開発・管理されているオープンソース WEB アプリケーションフレームワークである。フレークワークはプログラムの主たる動作の定型化をするものである。これによりプログラマごとのスキルの違いを吸収することができ，プログラムの品質を保つことができる。

また Struts は，入力値の検証やエラー処理機構などの WEB アプリケーションに必要な定型的機能を簡単に扱える仕組みが用意されている。

5.1.2 MVC モデルアーキテクチャ

Struts は MVC モデルアーキテクチャ（以下 MVC モデル）に基づいて設計されている。MVC モデルとはシステム開発において開発者の分業とシステムの保守性を向上するための理論である。M（モデル）は業務ロジックやデータベースアクセスを司る部分であり，V（ビュー）は目に見える部分のデザイン，そしてモデル，ビューの流れを規定しているのが C（コントローラ）である。MVC に分離することにより，デザイナは内部処理を気にすることなく，デザインだけに集中することができる。同様にプログラマは内部処理に集中することができる。

また M，V，C のうちいずれか一つの変更が他の要素に影響を与えないということがシステム開発の保守性を向上させている。これはモジュールの独立性とも呼ばれる。たとえば，データベースの構造に変更が生じた場合でも，モデルの部分だけの変更ですむといった利点がある。

5.1.3 必要な学習に関する評価

表 2 は Struts を利用しての評価である。現在 Struts は多くの書籍が出版され，また WEB でも多くの情報が公開されているため，それに関する情報を取得するのは容易である。

本プロジェクトでは Struts を理解し，ある程度使えるようになるまでに多くの時間を費やした。特にハードルが高かったのは Struts の処理フローを理解することである。MVC モデルでは前述したように，他のモジュールを意識しないでもモジュールを作成できる。しかし，アプリケーションフローが複雑なため，それを理解していない場合，プログラムの一部分だけ取って理解することは難しい。Struts は MVC モデルによりファイルが分散化される。個々のモジュール理解の難易度はそれほど高くはない。しかし，個々のモジュールにはそれぞれの役割があり，一連の流れの中で複雑に絡み合っている。

表 2：Struts の学習についての評価

Struts	難易度	動員人数	学習時間
概要の理解	★★	10 人	約 60 日間
処理フローの理解	★★★★★	10 人	
Java プログラム	★★★	4 人	
設定ファイル	★★★	4 人	
カスタムタグ	★★★★	4 人	
その他の機能	★★	2 人	

★の数が多いほど難易度が高い

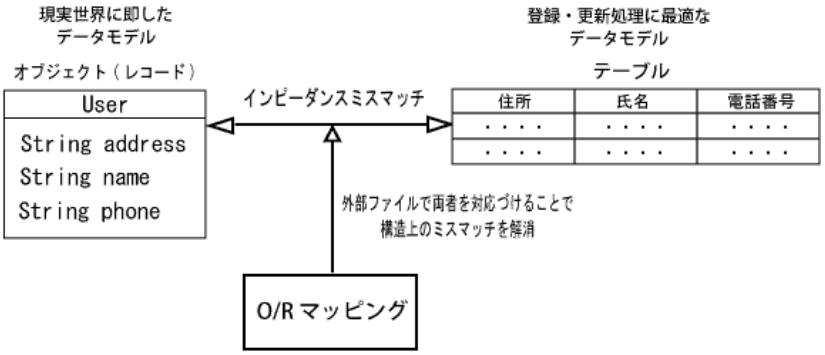

図 4：O/R マッピング

ある一連の処理に必要なファイルを見極め，それらがどのような順序で呼び出されるのかを理解する必要がある。書籍や WEB 上の資料の説明を読むだけでなく，実際にコードを書いて，動かしてみて，自分の目で処理フローを確認することが Struts 理解への一番の近道であった。

5.2 O/R マッピングフレームワーク Hibernate

5.2.1 概要

今回開発したシステムを含め，大量のデータの保持を必要とするような WEB アプリケーションでは，データベースの利用は必須である。使用している Java 言語のデータ設計は，オブジェクト指向に基づく現実世界に即したものであり，MySQL はデータをレコードなどの単位でとらえる関係データ設計がなされている。これらは各々が最適な役割を果たすために必要な概念であるが，両者には構造上の差異が生じる。Java のオブジェクト指向と SQL の非オブジェクト指向の手続きが混在することになり，互いの長所を阻害する。これらの差異はインピーダンスミスマッチと呼ばれ，これを解消するにはオブジェクトを分解して SQL を発行し，オブジェクトを再構築するといった，コーディングの工程で非常に煩雑な作業を必要とする（図 4）。これらの問題を回避・軽減するため，Object /Relational（以下 O/R）マッピングという概念およびフレームワークを利用することにした。

5.2.2 フレームワークの選択

前述した 2 つの概念に存在する構造上の差異を埋め，データベースをオブジェクト指向で扱いやすくするために考えられた概念が O/R マッピングである。O/R マッピングはオブジェクトとデータベースの対応づけ（マッピング）を行い，SQL を意識しないコーディングを実現し，データアクセスを容易にするものである。

この概念を実現するためには，O/R マッピングフレームワークが必要となる。オープンソースによるツールはいくつか存在しているが，それらの中で比較的豊富なドキュメント群を扱い，使い勝手の良さがあるなどという面を考慮し，今回 hibernate.org によって開発・配布されている Hibernate を採用した。

5.2.3 必要な学習に関する評価

表 3 は Hibernate を利用しての評価である。Hibernate は比較的新しい技術であるために書籍，特に和書が非常に少ない。そのため WEB での情報収集が主となる。数こそ少ないが，概要の解説やサンプルアプリケーションといった

表 3：Hibernate の学習についての評価

Hibernate	難易度	動員人数	学習時間
O/R マッピングの概念	★★★	6 人	約 45 日間
テーブルのリレーション	★★★★★	3 人	
コーディング	★★	4 人	

表 4：Tomcat の学習についての評価

環境構築	難易度	動員人数	学習時間
Tomcat のインストール	★	1 人	約 5 日間
Tomcat と Apache の連携	★★	1 人	
チューニング	★★★★★	1 人	

必要最低限の資料を集めることができた。Hibernate を利用するにあたっては，O/R マッピングの概念と利用目的を理解しなければならない。また O/R マッピングの概念を理解するには，オブジェクトと非オブジェクトを扱うデータモデルの概念を理解しなくてはならなかった。

次にテーブルのリレーションを設定する。この作業が Hibernate で一番苦労した点であった。

Hibernate では，Java オブジェクトとデータベースの構造を結びつける設定ファイルを記述しなければならない。テーブルのフィールドの定義やテーブル間のリレーションの定義など，多くの要素，属性があるため，記述内容が複雑になる。この設定ファイルは関連する補助ツールにより自動生成できる。ただし，ツールの利用によって作業の効率を上げることは可能だが，内容を理解した上での利用が前提となる。

例えば，システムを利用するユーザの情報を取得する場合（図 4），Hibernate では，初めからオブジェクトを結果として返すため，結果を組み立て直す手間が無くなる。また，テーブル間の関連の定義もされているため，SQL も簡単なものになる。また，データの登録，更新，削除といった単純な作業であれば，値を設定したオブジェクトを save（）や delete（）といったメソッドに渡すだけで行うことができる。

Java プログラミングをある程度理解していれば，Hibernate を利用したデータベースアクセスを組み込むのは比較的容易である。また，コーディングをすることによって，利用方法がより鮮明となり，理解を深めることができる。

5.3 アプリケーションサーバ Tomcat

Tomcat は，Jakarta プロジェクトより提供されているオープンソースである。Tomcat により，JSP や Java サーブレットの処理を行う。

Tomcat ベースのアプリケーションサーバは，非機能的要件に基づく環境の構築を前提とする。

5.3.1 概要

当システムでは，アベイラビリティの高いシステムの完成を目指す必要がある。Tomcat 5.0 ベースのアプリケーションサーバを 2 台用い，あるサーバに障害が起こった場合は，別のサーバに業務が引き継がれ業務を続けられることとなる（図 3）。分散処理には，主に DNS サーバを利用いた方法や Apache のモジュールを利用した方法などがある。後者の Apache のモジュールである JK2 コネクタを利用した。

JK2 コネクタにより，Apache から JK2 コネクタを介して Tomcat へと接続することとなる。これにより負荷分散機能とフォルトトレランス機能が有効となりアベイラビリティを高めることができる。

5.3.2 環境の構築

JK2 コネクタの導入には，Apache と Tomcat による環境が整っていることが前提となる。環境構築には JK2 コネクタモジュールを入れ，次に Tomcat 及び Apache の設定ファイルの一部を書き換え，コンテキストパス，サーバ名，そしてポート番号を適応させパスを通す。この一連の作業により，分散処理を行うことができる。

5.3.3 必要な学習に関する評価

表 4 は Tomcat を利用しての評価である。構築にあたっての設定は，WEB 上のマニュアルや参考本を読めば誰でもできるといってもよいほど，それらに従えばできる単純な作業である。ただし Linux の基本操作ができることを前提とする。Jakarta[1]等のリファレンスに従えば導入は比較的容易である。資料は主に，Jakarta から提供されるものとなるが，日本語に翻訳済みのものも多い。Tomcat のバージョンにより，設定ファイルの書き方が異なるなど，やや紛らわしいこともある。これらは，オープンソースがゆえに，WEB 上にある多くの資料に救われるが，それらの精度は不確かである。開発経験

によりその良し悪しを見分ける作業がプロジェクト成功への近道となった。

環境が整ったら，パフォーマンスチューニングを行い，リソースを最大限に生かす必要がある。チューニングは主として JVM（Java 仮想マシン）のメモリ割当てや，プログラミングそのものであるため，Tomcat のアプリケーションの知識だけでは，最適なパフォーマンスチューニングの実現は困難であった。プログラミングやシステムに関して幅広い知識が必要となるため，パフォーマンスチューニングの詳細についてはここでは省略する。

書籍や WEB 上などに十分な資料があるので，Tomcat に関する情報を入手すること自体は容易であった。技術情報が多ければ，様々なパターンの問題において，解決方法を短時間で見出すことができる。オープンソースの提供元からのリファレンスの公開は，一連の作業の流れと，その作業について定義される。これにより，WEB 上の情報の精度は明確になり，リファレンスを基にした様々な情報が生まれ，様々な解決方法が公開される。自分が探している情報，つまり構築に当たる環境を熟知し，それに適した情報さえ間違えなく探し出すことができれば，大きな時間コストの削減とつながる。

6. 完成したシステム

6.1 システムの外部評価

システム全体としては，機能的要件を満たし，非機能的要件は Tomcat を二重化することによって，システムの信頼性を向上させ，システムを完成することができた。完成後，専修大学ネットワーク情報学部で主催された外部公開のプロジェクトデモ会でシステムのデモを行った。その場で 24 あるプロジェクトの中から優秀賞に入賞することができ，神奈川県情報サービス産業協会の方から表彰をいただくなど，第三者からの評価を得ることができた。

6.2 数値からみるシステムの規模

作成したシステムのファイル数／ステップ数の表を表 5，6 にまとめた。この数値を Java ソースコードの部分だけで，それに関ったプログラマ一人当たりに換算すると，5250 ステップ，57 クラスになる。はじめに述べたような学生が 10 人程度集まり 1 年を通して開発に取り組めば，この程度のシステムを開発することができる。

表 5：構築したシステムのファイル数

種類（拡張子）	ファイル数
.java	228
.jsp	127
.xml	59
.wave	33
.png	41
.gif	267
.css	64
.html	20
.js	32

表 6：構築したシステムのステップ数

	ステップ数
Java ソース	20,803
JSP ソース	10,694

※　空行を除いたステップ数
　　コメント行は含んで計算

7. 考察

7.1 フレームワークの利用について

本システムの開発においては，Struts は有効であったといえる。学習に時間的コストはかかったものの，後期に入り，仕組みを理解し，使い方を慣れていくうちに有効だったと感じることが多かったからである。本システムの開発に携わったプログラマはサーバサイド Java のプログラミング経験／知識がなかった。また複数人による開発の経験も少なかったため，他人のソースを読む，他人のプログラムと結合させるといったことも当然少なかった。このような初心者プログラマに対して，定型化を強制することによって，ある一定のレベルを保証する Struts は品質の面から，そしてプログラムの結合や他人のソースを理解することなどにかかる時間の面からも有効であった。

Hibernate も Struts 同様に学習面での時間的コストは多少かかったが，コーディング面では有効であったと言える。最もプログラマにとって有効だったと感じられるのは，データそのものをオブジェクトとして扱える部分である。これによりデータの追加，更新，削除に関する処理部分のコーディングが Hibernate 導入以前よりも簡略化された。

フレームワークを利用することの意義は，あらかじめ完成しているものを利用することで，必要な機能を一から作る手間を省き，効率良く開発を行うことにある。注意しなければならないのは，必要な機能を提供しているかどうかという点や，学習にどれくらいの時間がかかるかという点である。特に学習のためのコストは重

要で，極端に時間がかかってしまい，フレームワークを活用せず自分で作った方が早かったといった事態になりかねない。このような事態を避けるには，各自の技術的なレベルや理解力を把握し，開発期間などの制約，条件を考慮したうえで，利用するかどうかを判断すべきである。

7.2 各技術の学習方法について

詳細な情報や専門性の高い情報が必要な場合は，関連する書籍，文献などから調査する必要がある。概要などの比較的簡単な情報を知りたい場合には，インターネットを利用した情報収集が良い。手軽に利用できる点や，情報の即時性などから考えても，非常に重要である。

近年のインターネット利用者の増加や，誰でも手軽に情報発信を可能にした，WEB ログなどのサービスの普及により，これまで以上に多種多様な情報を容易かつ迅速に得ることが可能になってきている。

一方で，すべての情報が容易に入手できるとは限らない。比較的情報量の少ない技術も扱ってきた。このような場合には，手元にある情報や，知識・経験を頼りに手探りで実践し，理解を試みるべきであろう。この例に限らず，活動を通じて獲得した様々な知識や情報をいかに活用していくかがカギとなる。

7.3 メンバーの自主性とコミュニケーション

システム開発においては，プログラムのバグやシステムの設計に矛盾が生じるなど，様々な問題に直面する可能性が高い。本稿の冒頭でも述べたように，定められた期間の中で効率良く開発を進めるためには，各メンバーが積極的に行動を起こし，メンバー間でコミュニケーションをとることが重要となる。

注意すべき点は，相手に伝えたいことや，その意図を明確に伝えようと意識することである。本来の意味や意図とは異なって伝わってしまうと，手戻りや仕様変更など，開発工程に影響する可能性もある。特に掲示板やメッセンジャーなどの文章によるやり取りの場合だと，直接会話をするよりも感情などが伝わりにくいため，このような事態が起こりやすい。メンバー同士が直接的かつ活発なやり取りができる機会を増やすことで，このような誤解を極力低減し，モチベーションの維持・向上を図ることができる。

8. おわりに

本稿では，10 名の学生から構成されるプロジェクトにより，Web 型ビジネスアプリケーションという 20K ステップ程度の情報システムを構築する試みにおいて，どのようにコミュニケーションをとりながら開発を進めていったのか，どのようにオープンソースで提供されるフレームワーク技術を学習し利用していったのかについて述べた。

この試みによる知見は，1 年程度の研修を受けた企業での技術者がオープンソース技術を活用した情報システムを構築する際，あるいは物理的位置の離れた技術者がインターネットを活用してコミュニケーションをとりながら情報システムを開発する際にも適用できると考えている。

参考文献

[1] The Apache Jakarta Project, http://Jakarta.Apache.org/

[2] チャック･カバンス，ブライアン･キートン著，長瀬嘉秀訳：プログラミング Jakarta Struts，オライリー・ジャパン，2003.

[3] テッド・ハスティッド他著，株式会社クイープ訳：STRUTS・イン・アクション，ソフトバンクパブリッシング，2003.

[4] @IT：連載 Struts を使う Web アプリケーション構築術（第1回） http://www.atmarkit.co.jp/fjava/rensai3/struts01/struts01_1.html

[5] 中嶋睦月，木村貴由：最強 O/R マッピングツール Hibernate 実践入門，オープンソース Java プロダクツ，pp.164-200，技術評論社，2004.

[6] @IT 連載記事：Hibernate で理解する O/R マッピング（第 1 回）， http://www.atmarkit.co.jp/fjava/rensai3/ormap01/ormap01.html

[7] hibernate.org, http://www.hibernate.org/

[8] Hibernate リファレンス（日本語訳）， http://www.hibernate.org/hib_docs/reference/ja/html/

[9] Hibernate's Wiki, http://nekop.programmers.jp/wiki/Hibernate/?FrontPage

[10] Hibernate Pad, http://wiki.bmedianode.com/Hibernate/

[11] ジェイソン･ブリテン，イアン･ダーウィン著，村上雅章訳：Tomcat ハンドブック，オライリー・ジャパン，2003.