Railsによる技術センター業務依頼システム構築

○森圭史朗，甲斐崇浩，相川勝，西岡祐介
宮崎大学工学部教育研究支援技術センター

1. はじめに

技術センター情報処理技術班では、情報処理技術の急速な進展に対応していくため、常に新たな技術習得が求められている。その中で、以前より教員から要望のあった業務依頼のシステム化に応えるため、班として開発業務に取り組むことになった。今まで技術センターに業務を依頼する場合、依頼者側は、技術センターHP 上にある複数の申請様式ファイルに従って紙ベースによる業務依頼書を作成し、技術センターに提出する必要があった。また、技術センター側も依頼を受けた業務に関しての書類やデータ等を作成する必要があるため、多くの時間と労力が費やされていた。これらをシステム化することで、ペーパレス化によるコスト削減、業務管理の労力削減、業務依頼申請が容易になることが期待できる。開発業務においては、システムの土台としてWebアプリケーションフレームワークであるRailsを採用し、班としての新たな共通技術習得にも取り組んだ。よって今回、技術センターの業務依頼システムを構築した内容について報告する。

2. Railsについて

Railsは、Ruby環境で利用可能なWebアプリケーションフレームワークで、MVC（Model View Controller）アーキテクチャを採用している（図1）。Rails公式のGemパッケージは、RubyGems（https://rubygems.org）により配布される。

Railsアプリケーション開発の特徴として、DB操作は、Railsコマンドから実行でき、アプリケーション動作は、Webサーバの代わりにRails標準のWEBrickにて確認できる。

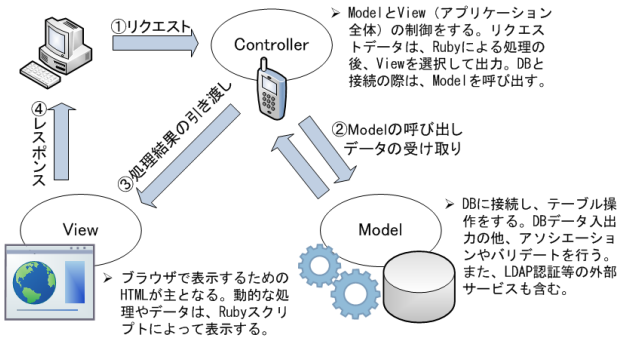

図1　MVC（Model View Controller）アーキテクチャ

3. システム設計及び開発

業務依頼システムを構築するに当たって、まず、表1のシステム開発環境の構築と業務依頼フローに従ったシステム設計（DBのER図、DBのテーブル定義書、システムフロー図、画面設計の作成）をそれぞれ分担して取り組んだ。DBサーバには、PostgreSQL、Rails開発環境には、rbenvを採用し、複数人の開発スタッフが個別にRailsアプリケーションを構築できるようにした。システム設計では、DBのテーブル名やフィールド名等の名称を統一し、ER図においてアソシエーション規則の取り決め、システムフロー図に伴う画面デザインの作成を行った。これらのシステム設計資料を基に、Bootstrapを用いたView作成担当とRailsアプリケーションのModel・Controller作成担当に分かれてコーディングを開始した。コーディング中は、定期的なミーティングを実施し、進捗状況や新たに取得した技術情報を共有しながら、最適な機能の実装を目指した。今回、構築したWebアプリケーションは、依頼者（Customer）、

表1　システム環境

分類	名称	概要
OS	CentOS6	サーバ用OS
サーバアプリケーション	ruby-2.1.6	Ruby本体
	node-v0.10.25	JavaScriptフレームワーク
	nginx-1.8.0	Webサーバ
	openssl-1.0.1p	通信の暗号化
	postgresql-9.4.4	DBサーバ
主なRubyパッケージ（Gems）	rails(4.1.12)	Rails本体
	passenger(5.0.14)	Webサーバ追加モジュール
	capistrano(2.15.5)	本番用環境構築（デプロイ）
	seed-fu(2.3.5)	DBデータ追加・編集
	activeldap(4.0.4)	LDAP認証
テンプレート	bootstrap-3.2.0	CSS、JavaScript

技術職員（Staff）、業務調整小委員会（Admin）の 3 つのネームスペースによる構成とした。

各ネームスペースの機能は、以下の通りである。

- Customer（依頼者）：
  依頼リスト及び内容閲覧、所属学科依頼リスト及び内容閲覧、新規申請・編集・削除、過去依頼引用申請、研究継続申請、代理依頼申請、完了報告書作成（ファイルアップロードを含む）
- Staff（技術職員）：
  承認リスト及び内容閲覧、所管技術職員の依頼リスト及び内容閲覧、受付内容閲覧、完了報告書作成
- Admin（業務調整小委員会）：
  業務依頼情報の条件検索及び閲覧、受付業務内容の調整・承認、承認メール送信（依頼者、Cc：業務担当者）、報告書確認（依頼者、技術職員）

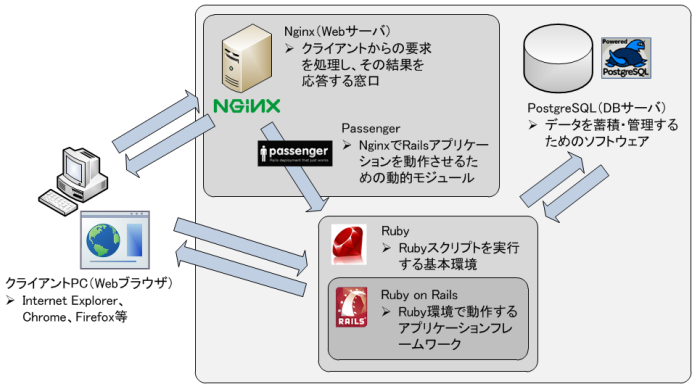

図 2　構築した本番用システム環境

最後に、各担当者が作成したアプリケーション機能を 1 つに統合し、Capistrano によって本番用システム環境へデプロイすることにより、Nginx+ Passenger+ Rails+ PostgreSQL（図 2）による業務依頼システムを完成させた。このデプロイによって構築される本番用システム環境は、CSS や JavaScript の最適化や圧縮が行われ、運用するシステムのパフォーマンスを向上させることができる。

4. システム運用開始

作成した Rails アプリケーションの不具合を抽出するため、テストシナリオを作成し、構築したシステムに対しテストを行って可能な限り不具合を修正した。このテストによる不具合を修正後、今年度分の業務依頼実データをシステムに入力、そして、学内統一認証を利用するための LDAP 認証を導入して 2015 年 8 月上旬にシステム運用を開始した（図 3）。

運用開始後、開発環境では問題なかったトラブルが 2 つ生じた。1 つ目は、システムへログイン不可となる現象である。これは、Google Chrome を使用し、かつシステムに favicon が無い場合、再読み込みが発生して session 処理に失敗することが原因であった。もう 1 つは、特定の Internet Explorer において View が崩れてしまう現象である。こちらは、互換性表示が設定されている場合に発生する現象と分かり、HTML のヘッダ情報を追加することで解決した。

図 3　新規業務依頼申請画面

5. おわりに

今回、業務依頼システムの構築を行った。業務依頼をシステム化することで、依頼者は技術センターへ出向いて業務依頼書を提出することなく、常時 Web 上から申請することが可能となった。また、技術センターの業務管理においても業務依頼書のペーパレス化や受付から承認までの手続きの簡素化、依頼情報の DB 化により必要な情報の迅速な取得が可能となった。班でシステム構築に取り組んだことにより、班全体としてシステム構築に関する新たな知識や方法を共有し習得できたため、システムに不具合が生じた際は、問題を一人で抱えることなく対処することが可能となった。

参考文献

山田祥寛.　Ruby On Rails4 アプリケーションプログラミング,　技術評論社,　2014