筑波大学大学院博士課程

システム情報工学研究科特定課題研究報告書

学内レポート管理ワークフローシステムの開発

～チケット駆動スクラム開発におけるプロトタイプ作成および開発環境構築～

森有司

(コンピュータサイエンス専攻)

指導教員　田中二郎

2011年3月

本プロジェクトでは、株式会社日立製作所ソフトウェア事業部の提案テーマに基づき、「学内レポート管理ワークフローの開発」を行った。システムの主目的は教員のレポート管理コストを低減し、また学生のレポート提出忘れのリスクを低減することである。

本プロジェクトは最小限かつ効果の大きなプロジェクト管理手法として、「チケット駆動ソフトウェア開発プロジェクト管理手法」を取った。チケット駆動ソフトウェア開発プロジェクト管理とは、チケットという単位により、要件・バグ・成果物・課題等、ソフトウェア開発に関わる様々な関心事を、その計画から終了まで管理する手法である。本プロジェクトにおいて、筆者は開発環境及び運用環境の構築、管理を行った。またプロジェクトの初期においてスクラムにおけるプロトタイプの作成を行い、その後のスプリントにおいてモデル部分の開発全体、また機能としてユーザ管理機能の実装を担当した。

本報告書ではプロジェクトの概要と採用した手法、全体の計画について述べたうえで、その中で筆者が担当した部分について詳しく報告を行う。

目 次

第 1 章　プロジェクト概要 **1**
1.1 プロジェクト目的 1
1.1.1 プロジェクト背景 1
1.1.2 レポート管理ワークフロー 1
1.1.3 システムが解決する問題 3
1.2 システム概要 4
1.2.1 主要機能の実現：レポート提出状態の遷移 4
1.2.2 システム構成 6
1.3 プロジェクト構成 6
1.3.1 チケット駆動スクラム開発 7
チケット駆動ソフトウェア開発プロジェクト管理 8
スクラム開発 8
1.3.2 全体スケジュール 9
1.3.3 プロジェクトの実施と役割分担 9

第 2 章　開発環境管理 **13**
2.1 プロジェクト管理環境 14
2.2 構成管理・動作確認環境 15
2.3 作業環境 16
2.4 評価実験運用環境 17

第 3 章　開発担当 **18**
3.1 プロトタイプ作成 18
3.2 開発担当分担 18
3.3 モデル開発 22
3.4 ユーザ管理機能 22

第 4 章　まとめ **33**

参考文献 **34**

図 目 次

1.1 関係概略 2
1.2 概略ユースケース図 5
1.3 レポート提出状態遷移図 6
1.4 プロジェクトの進行 9

2.1 プロジェクトの状況確認とプロダクトの動作確認 13
2.2 実装作業時のチェックアウト・コミット 14
2.3 Redmine 16

3.1 要件定義（開発構想書）、プロトタイプ、処理方式設計の関係 19
3.2 CakePHP の MVC モデル 20
3.3 開発担当分担のイメージ 21
3.4 ログイン画面 25
3.5 学生ユーザの登録 26
3.6 学生ユーザの一斉登録 27
3.7 学生ユーザの一斉登録結果 28
3.8 講師ユーザの登録 29
3.9 学生ユーザの一覧 30
3.10 ユーザ自身による学生情報の編集 31
3.11 管理者による学生情報の編集 32

第1章　プロジェクト概要

本プロジェクトの概要について、その目的と構成、計画を説明する。

1.1　プロジェクト目的

本研究開発プロジェクト「学内レポート管理ワークフローシステムの開発」は株式会社日立製作所の提案プロジェクトであり、その目的は次の点にある。

- 開発する学内レポート管理ワークフローシステムにより、学生のレポート提出ミス・提出忘れを防ぎ、講義を行う教員のレポート管理コストを低減すること
- プロジェクトの開発メンバがシステムの開発を通じて、ソフトウェアの設計・開発スキルやプロジェクトにおける品質管理や構成管理等の管理スキルを修得すること

1.1.1　プロジェクト背景

レポートの提出状況の管理に関し、次のことが考えられる。

- 提出状況を講師・TA 間で共有する必要がある

  図 1.1 は、科目周りの関係を示している。1 つの科目は複数の講師・TA により担当され得る。

- 学生の提出ミス・提出忘れが起こりうるが、これを防ぐべきである

現在、レポートの提出状況の管理方法としては、提出状況を記入したスプレッドシート（Excel、Google スプレッドシート等）の共有による方法や、moodle[8] のような LMS を利用する方法などが取られている。本プロジェクトにおける開発システムはこれらに比べ、複数の通知機能による学生の提出ミス・提出忘れを防止することを特徴とする。

1.1.2　レポート管理ワークフロー

大学にて開設されている個々の科目においてレポートの処理手順は様々であるが、その多くは次のワークフロー [1] を共通点として見出すことができる。

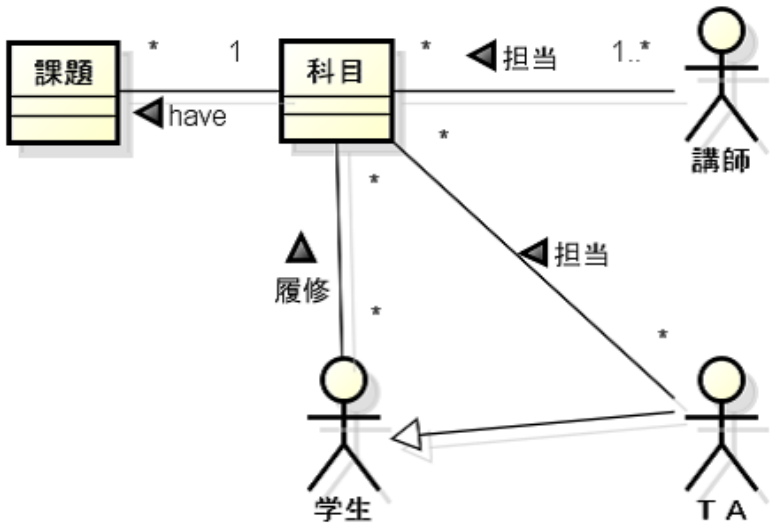

図 1.1: 関係概略

1. 講師が課題を提示する

   課題内容、期限、提出方法等が提示される。

2. 学生はレポートを作成し、期限内に提出する

   提出方法は講師・TA への手渡し、提出ポストへの投函、メールによる送付等、多岐に渡る。

3. 講師、もしくは TA がレポートを受け取る

   この際、レポートが提出基準を満たしていない等の理由により、差し戻しが行われることがある。差し戻しが行われた場合、学生はレポートを修正し再提出を行うことになる。

   また各学生の各課題に対する提出状況は何らかの方法により管理され、科目の講師間にて共有される。

4. 講師がレポートを採点する

   採点後、学生に対しフィードバックや模範解答の提示を行う場合がある。

本プロジェクトではこの一連の流れをレポート管理ワークフローと呼び、これをシステム化の範囲とする。

1.1.3 システムが解決する問題

上述したレポート管理ワークフローにおける問題について、本システムは以下の二点を中心に解決する。

問題 1 提出状況・採点の共有

レポート管理ワークフローにおいて、講師側は各学生の各課題の提出状況・採点について共有を行う。「レポートの受取を TA が行い、まとめた上で講師に送付する」等といった状況では、提出状況の記録・閲覧を複数人の間で行う必要がある。

問題 2 学生のレポート提出失念

レポート管理ワークフローの 2 において、学生がレポートの提出を失念することがある。主な原因としては次のことが考えられる。

原因 a レポート課題が出されたことを知らない

レポート課題が提示された講義に参加しなかった等の理由により、提出する必要のある課題が課せられていることを知らない。

原因 b レポート課題そのものや、提出期限を失念する

スケジュールの記録を間違える・怠る等の理由により、レポート課題が課せられていることや、その期限を忘れてしまう。

問題 3 学生のレポート提出ミス

例えばレポート提出方法がメールによる送付であった場合に送信先を誤る等、提出ミスに気づかないことがある。

本システムは次の機能を提供することにより、上述の問題を解決する。

- **提出状況の記録・閲覧機能**

  本システムは Web アプリケーションの形態を取り、ユーザが Web ブラウザを通じてレポート提出状況の記録・閲覧を行うための機能を提供し、問題 1 を解決する。

- **複数の通知機能**

  本システムは提出状況を監視し、学生に対してレポート提出に関する次の通知を行う。

  – 課題が提示されたことの通知

    学生が提出する必要のあるレポート課題が提示された際にメールによる通知を行う。これにより問題 2 の原因 2a の発生を防ぐ。

  – 課題の提出期限の通知

    提出期限が迫っているレポート課題がある場合、その旨を学生にメールにより通知する。また、提出期限を過ぎてもレポート提出を受け付ける課題においては、提

出期限を過ぎた際にその旨を学生に再度通知する。これにより問題 2 の原因 2b の発生を防ぐ。

– レポートの受取・差し戻しの通知

講師・TA がレポートを受取・差し戻しした時に、その旨を学生にメールにより通知する。これにより学生は提出したレポートが受け取られたことや差し戻されたことを知ることができるようになり、問題 3 の解決となる。

1.2 システム概要

本プロジェクトにおいて開発するシステムについて述べる。開発システムのシステム名は「threpo」である。

図 1.2 は本システムのユースケース概略を示したものである。本システムのユーザは、講師と学生、及びシステム管理者の 3 種類であり、また学生が TA である場合がある。また各ユーザにより、データ・操作に対する権限は異なるため、本システムはユーザのログインを必要とする。

システム管理者はユーザの管理のみを行う。講師は自身の担当科目、課題の管理を行い、講師と TA は課題に対する各学生のレポートの提出状態を操作を行う。講師（及び TA）、学生は自身が関わるレポートの提出状態を確認する。学生は、自分に課せられた課題のレポートに対し、その提出状態の変化と提出期限の接近を通知される。

各ユーザはシステムにメールアドレスを登録する（講師、学生はシステムのユーザ ID となる）。学生への通知はこのメールアドレスに対してメッセージを送信することで実行される。

本システムでは前述の主要機能を実現するため、レポートの提出状態を管理する。

1.2.1 主要機能の実現：レポート提出状態の遷移

本システムの各主要機能を、各課題に対する各学生のレポート提出状態を管理することで実現する。

図 1.3 は、本システムにおける、レポート提出状態の遷移を表した図である。図の状態は、1 人の学生に対する 1 つの課題のレポートが「講師側において」どのように判断されているかを表している。講師が担当科目において課題を提示したときに、この状態遷移が開始される。また各状態の意味は次の通りである。

未提出 レポートが受け取られていない状態を示す。

受取済 レポートが提出され、講師側に届いた状態を示す。

受理済 レポートが提出に必要な条件（記名、書式等）を満たしていることを講師側が確認し、採点するに足ると判断された状態を示す。

採点済 講師により、レポートが採点された状態を示す。

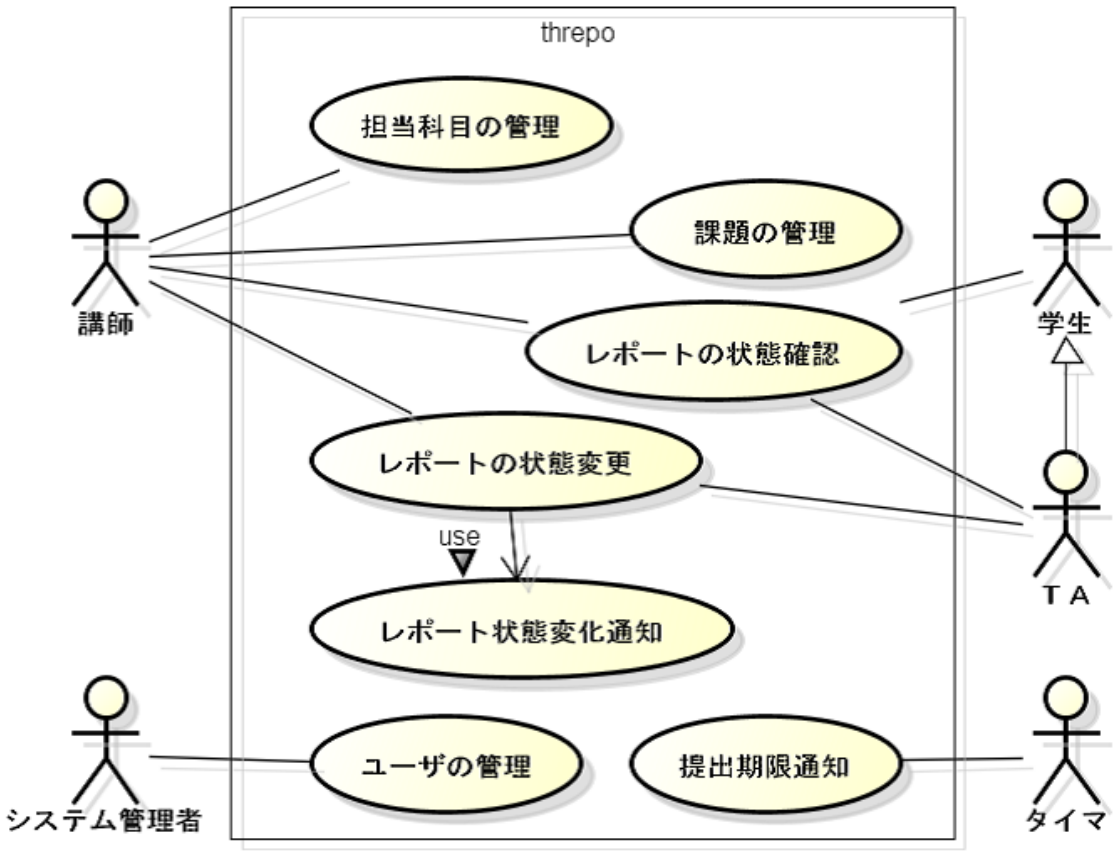

図 1.2: 概略ユースケース図

差し戻し レポートに何らかの問題が発見されたために、学生が再提出する必要がある状態を示す。

例えば学生がレポートを提出した後、講師側がシステムに受け取ったことを記録したとき、そのレポートの状態は「受取済」となる。

前述の本システムの各主要機能はこの状態遷移を元に実現する。「提出状況の記録・閲覧機能」はこの状態遷移の管理自体であり、講師、TA であれば担当科目の課題に対する各レポートの提出状態を記録・閲覧でき、学生であれば自身が提出者である各レポートの提出状態を閲覧できる。表 1.1 は各ユーザがレポート提出状況に対しどのような権限を持っているかまとめたものであり、表中にない権限は一切持たない。「課題が提示されたことの通知」「レポートの受取・差し戻しの通知」は状態遷移に対する入場動作として定義され、そのレポートの提出者である学生に対し、システムからメールを送信する。「課題の提出期限の通知」は、提出期限が接近した時に、提出状態が未提出・差し戻しである場合に実行される。

図 1.2 の「ユーザの管理」「担当科目の管理」「課題の管理」等は、レポートの提出状態に関

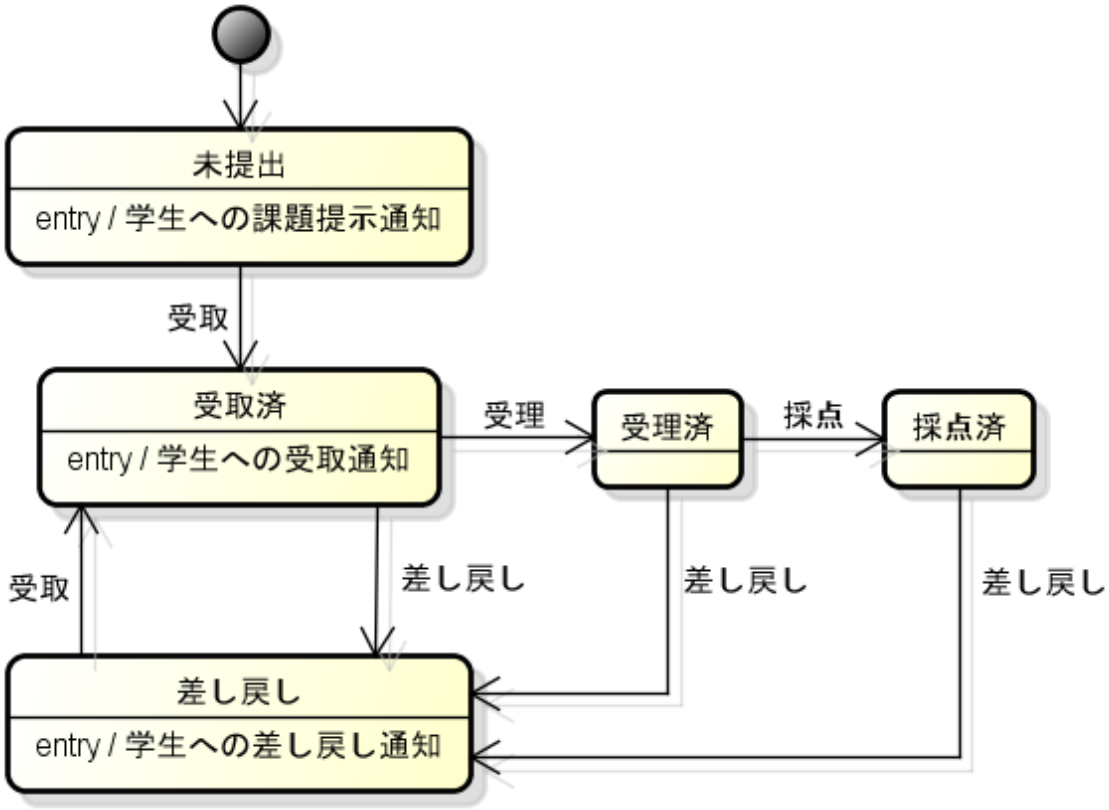

図 1.3: レポート提出状態遷移図

連する情報である、「どの学生が提出するのか」「どの科目、課題のレポートか」といった情報を構築するための機能である。

1.2.2 システム構成

本システムは Web アプリケーションの形態を取る。開発言語は PHP であり、フレームワークとして CakePHP[6] を使用する。

本システムは学内に設置した運用サーバに配置される。システムのユーザである講師、学生は PC、iPhone の Web ブラウザを通し、学内学外よりシステムを利用する。

また本システムの実行に必要な環境は表 1.2 の通りである。

1.3 プロジェクト構成

本節ではプロジェクトの構成について述べる。

表 1.1: レポート提出状況への権限

ユーザ	対象	権限
講師	担当科目の課題に対するレポートの提出状況	閲覧＋編集
学生	自身が提出者であるレポートの提出状況	閲覧
	TA として担当している科目の課題に対するレポートの提出状況	閲覧＋編集

表 1.2: 実行環境

分類		ソフトウェア
サーバ	OS	CentOS 5.5
	HTTP サーバ	Apache 2.2
	データベースサーバ	MySQL 5.0
	メールサーバ	何らかの SMTP サーバ
クライアント	ブラウザ（PC）	Firefox 3.0 以上、Internet Explorer 8 以上

- プロジェクトの関係者

  開発メンバである学生は筑波大学大学院システム情報工学研究科　コンピュータサイエンス専攻　博士課程 2 年に所属する（筆者である）森有司、河村翼、韓貞美、カンキスルン、の 4 名である。ここにテーマの提案とプロジェクト全体に関するアドバイスを行う、株式会社日立製作所の技術者である居駒幹夫氏、弥生隆明氏、公立はこだて未来大学の大場みち子教授、本プロジェクトテーマの委託元教員である天笠俊之准教の 4 名を含めた 8 名が主要な関係者である。また後述する評価実験において学生 17 名の協力を受けた。

- 開発手法

  開発手法として、チケット駆動スクラム開発を採用する。これは、アジャイル開発手法の一つであるスクラム開発をチケット駆動ソフトウェア開発プロジェクト管理の方法にて運用するものであり、以降にて詳しく説明する。

- 期間

  9 月 1 日より、翌年 1 月までの約 5 か月をプロジェクトの期間とする。ただしソフトウェア開発自体は評価実験のため、12 月 22 日までがその期限となる。

1.3.1　チケット駆動スクラム開発

本プロジェクトにて採用した開発手法である、チケット駆動スクラム開発について説明する。チケット駆動スクラム開発とは、アジャイル開発手法の一つであるスクラム開発を、チケット駆動ソフトウェア開発プロジェクト管理により運営するものである。これらについて以下に説明する。

チケット駆動ソフトウェア開発プロジェクト管理

本プロジェクトでは、最小限かつ効果の大きなプロジェクト管理手法として、「チケット駆動ソフトウェア開発プロジェクト管理手法」[5] を取る。チケット駆動ソフトウェア開発プロジェクト管理とは、チケットという単位により、要件・バグ・成果物・課題等、ソフトウェア開発に関わる様々な関心事を、その計画から終了まで管理する手法である。

チケットは「○○機能の実装」、「○○におけるバグの修正」等、目的をベースに発行され、これに次のような情報が付加される。

1. チケット分類・内容
2. 担当者
3. 開始日・期日
4. 工数
5. 優先度
6. 対象バージョン

チケットには、そのチケットに関する次のような情報を記録していくことができる。

1. 作業分類・内容
2. 作業時間
3. 進行状況
4. その他備考情報

本プロジェクトではこのチケットを Redmine*を使用し管理する。これにより開発メンバと日立の技術者・課題担当教員がプロジェクトの状況を Web を通しリアルタイムに把握することができる。

スクラム開発

本プロジェクトでは上述のチケット駆動のプロジェクト管理の中で、機能開発をスクラム [2][3] を用いて行う。

スクラムはアジャイル開発手法の一つであり、30 日単位のサイクル（スプリント）で動作するソフトウェアの開発を進める。スプリントの開始時にはそのスプリントでの開発目標と計画を立てる（SPM：スプリントプランミーティング）。スプリント中は開発メンバ間で日々

*Redmine 公式サイト `http://www.redmine.org/`

の進捗を報告しあう（デイリーミーティング）。スプリントの終了時にはそのスプリントを振りかえり、次のスプリントの計画を行う（SRM：スプリントリプランニングミーティング）。

SPM、SRM では、開発目標に対する計画がタスクの集合としてリストアップされる。本プロジェクトでは各タスクをチケット化し管理する。

デイリーミーティングではチケットの進行状況（前回までの作業内容、次回までの作業予定）と懸案事項（障害、心配事等）を確認し、必要ならば別途フォローアップを行う。

また、スプリント（1st スプリント）の開始前に必要となる準備工程（図中のスプリント前工程）として要件定義、各種規約作成、開発環境構築、プロトタイピング、概要設計を行う。

1.3.2 全体スケジュール

プロジェクト全体のスケジュールについて述べる。図 1.4 はプロジェクト進行の概略を示すものであり、日程は表 1.3 の通りである。

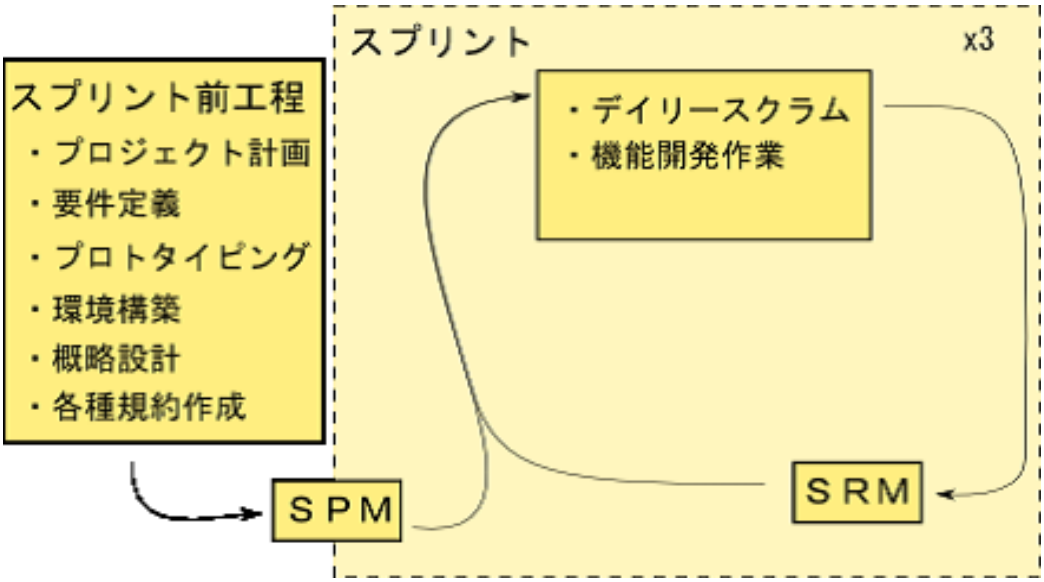

図 1.4: プロジェクトの進行

1.3.3 プロジェクトの実施と役割分担

プロジェクトの実施時における開発メンバの役割を述べる。
まず、スプリントを行う前工程として、9 月中は次のことを行った。

1. 要件定義

   開発システムの要件定義を行い、開発構想書を作成する。

表 1.3: 日程

工程	日程
スプリント前工程	9 月中
1st スプリント	2010 年 10 月中
2nd スプリント	2010 年 11 月中
3nd スプリント	2010 年 12 月 1 日〜12 月 22 日
評価	2010 年 12 月 22 日〜2011 年 1 月 8 日
ドキュメント作成	〜2011 年 1 月 31 日
報告書（本報告書）作成	〜2011 年 1 月 19 日

2. 各種規約作成

   プロジェクト運営の各種規約を作成する。

3. 開発環境構築

   プロジェクト運営に必要な環境を構築する。

4. プロトタイピング

   1st スプリント以降における機能追加の元となるプロトタイプを作成する。

5. 概要設計

   開発システムの基本的な分析・設計を行う。

6. SPM

   全体の大まかなスケジュールを計画し、1st スプリントの具体的なタスクのリストアップを行う。

筆者はスプリント前工程では、既に説明した開発環境の構築に並行して、プロトタイプの作成を行った。

スプリント前工程後の、各スプリントにおける開発目標は次のようになった。

1. 1st スプリント（10 月中）

   1st スプリントの開発主目的は、最もシステムにとって基本的な機能である「レポート提出状況の管理」が可能になることである。これに加え、その動作が確認できるだけの関連データ（講師、学生、科目、課題、及びそれらの関係）が入力可能である必要がある。また、レポート提出状況の管理機能の上に作られる機能である通知機能に関しては、その技術的懸案事項についての調査までを 1st スプリントの範囲とした。

   このスプリントにおいて筆者は機能開発のベースとなるプロトタイプ作成者として、モデルの開発を担当した。

結果として、最大目標である「レポート提出状況の管理」はデータの扱いが可能になった。動作確認に必要な関連データは（プロトタイプから使用している）CakePHP の機能を最大限利用し、仮画面／処理を通して可能になった。利用した機能は CakePHP のスキャフォールドという、モデル定義に対して CRUD を備えた単純な（アクセス制御等は含まない）画面／処理を作成する機能であり、シェルからのコマンドによりごく短時間で利用できる。

2. 2nd スプリント（11 月中）

   1st スプリントの積み残しに加え、12 月の評価実験を行う上で必須となる、関連データの入力／編集／一覧と通知機能が動作することを開発主目的とした。

   このスプリントにおいて筆者は引き続きモデルの開発を担当し、また新たにユーザ管理機能の実装を担当した。

3. 3rd スプリント（12 月 1 日〜12 月 22 日）

   2nd スプリントにて実装の遅れた機能と、ユーザビリティを意識した View の整備を開発目標とした。

   このスプリントにおいて筆者は引き続きモデルの開発とユーザ管理機能の実装を担当した。

プロジェクトの各工程における筆者の担当範囲のみをまとめたものが、表 1.4 である。筆者はプロジェクト全体を通しての役割として、開発環境（及び運用環境）の構築／管理を担当した。また、スプリント前行程においてはプロトタイプ作成を担当し、以降のスプリントにおいては、モデル開発とユーザ管理機能開発を担当した。評価実験においては運用環境の構築から管理までと、threpo の管理者としてユーザの登録と編集、また threpo に関する参加者からの質問対応受付を担当した。

次章以降はこれら筆者の担当範囲について報告する。

表 1.4: 担当範囲

工程	担当
スプリント前工程	開発環境構築
1st スプリント	モデル開発
	開発環境管理
2nd スプリント	モデル開発
	ユーザ管理機能開発
	開発環境管理
3nd スプリント	モデル開発
	ユーザ管理機能開発
	開発環境管理
評価	評価実験運用環境構築・管理
	評価実験時のシステム管理者担当
ドキュメント作成	保守書：システム運用方法の記述

第2章　開発環境管理

スプリント前工程において担当範囲の一つとして、開発環境の構築を行った。構築した環境の概要を図 2.1 に示す。この章では、これについて報告を行う。

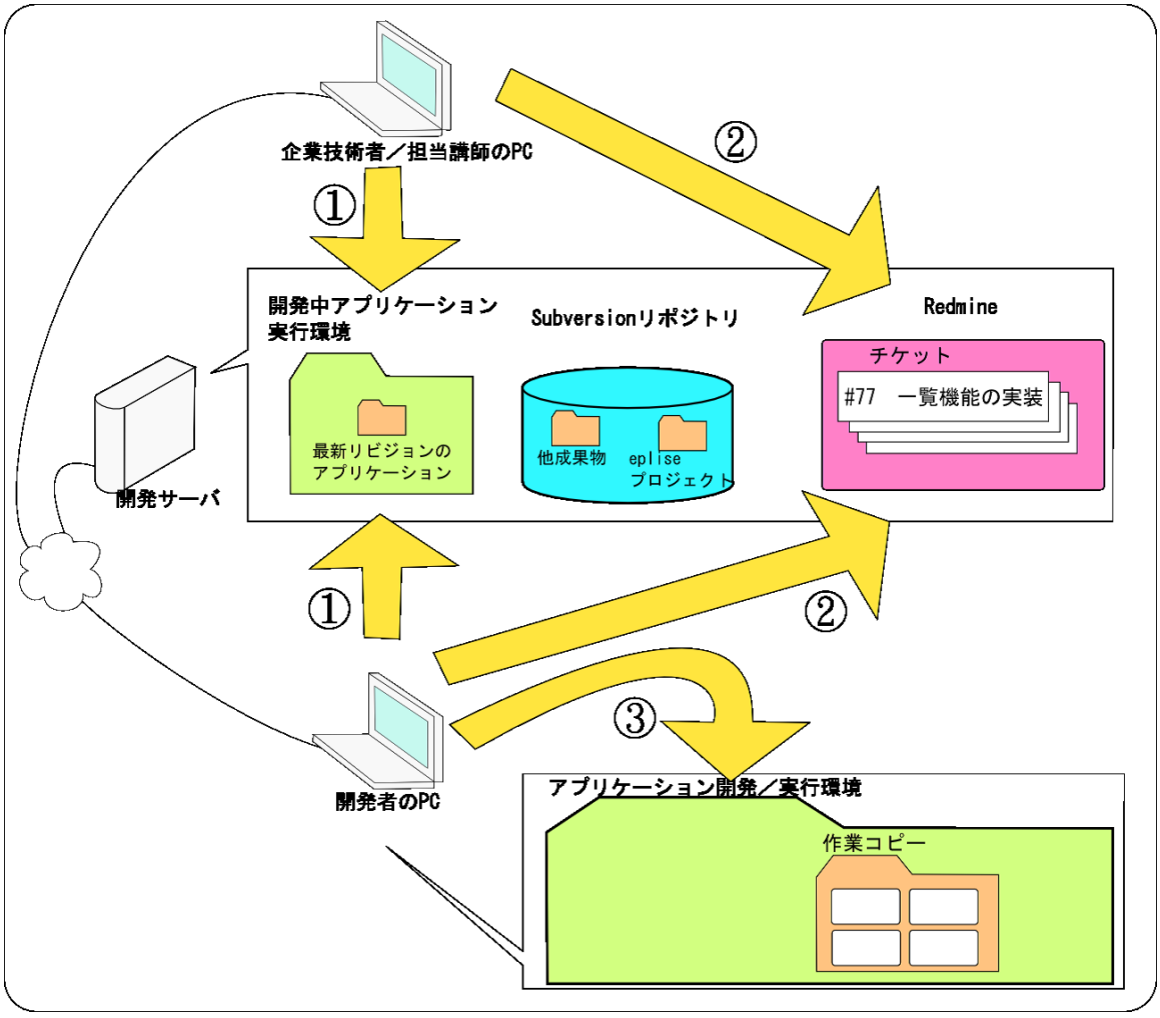

図 2.1: プロジェクトの状況確認とプロダクトの動作確認

2.1　プロジェクト管理環境

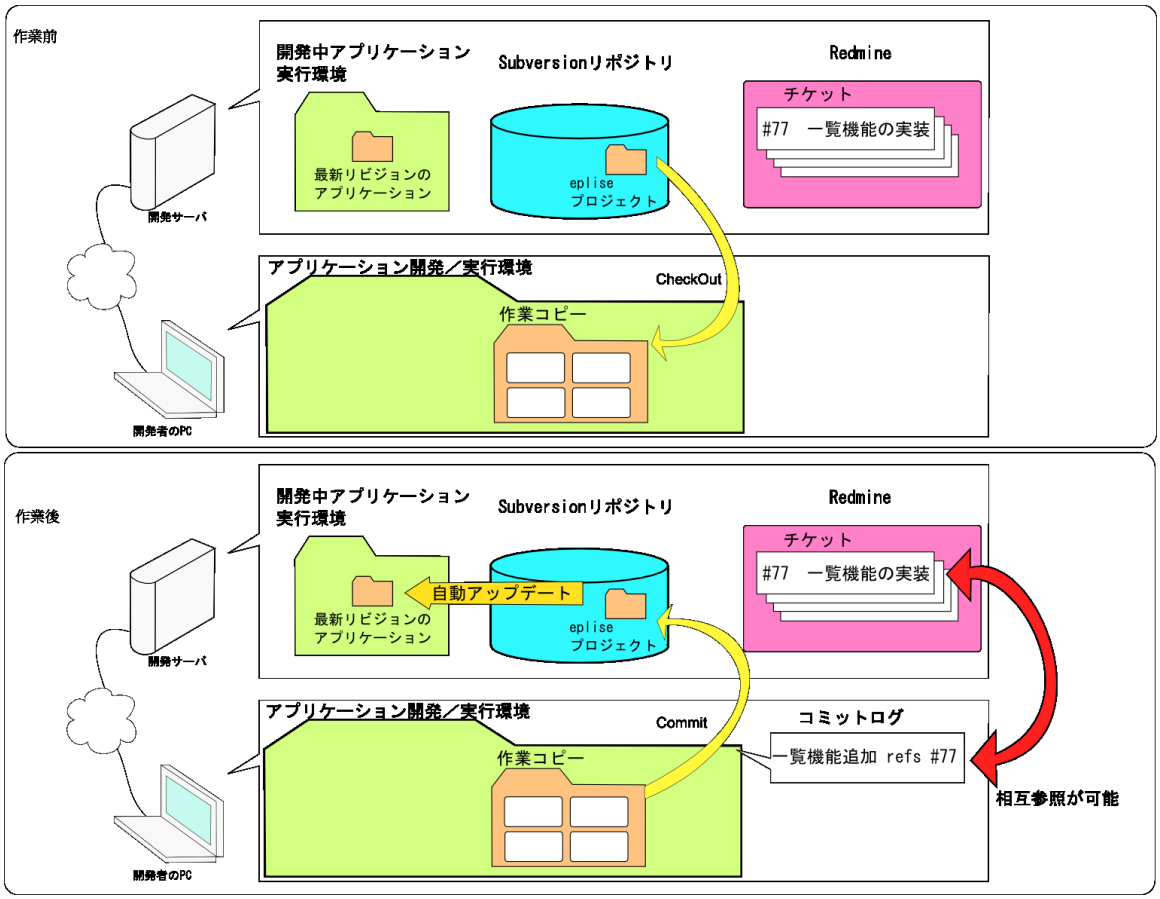

図 2.2: 実装作業時のチェックアウト・コミット

図 2.1 は、各環境からのプロジェクト進行の状況とプロダクトの状況を確認する手段を示している。各矢印は Web ブラウザを通して矢印の先にアクセス可能であることを示している。矢印① と矢印② は最新リビジョンのコードの動作確認とチケットの状況確認（及び編集）を、開発メンバ・課題担当教員・日立技術者が行えることを示している。なお、矢印① と矢印② のアクセスにはそれぞれ、Apache の Digest 認証、Redmine のユーザ認証を必要とする。矢印② は、開発メンバが担当部分の実装作業中に実装中のコードの動作をローカルの環境で確認できることを示している。

図 2.2 は実装作業のチェックアウト・コミットの様子を示している。リポジトリから（Eclipse プロジェクトである）作業コピーをチェックアウト先が、そのまま実行環境として設定されており、実装中の容易な動作確認を可能としている。また作業終了時にコミットを行うと設定されているフックスクリプトにより、必要なファイルをリポジトリの最新リビジョンから開発管理サーバの実行環境に自動デプロイされる。これによりプロジェクト関係者全員が最新

の開発状況を容易に確認できるようになっている（この時コミットログにチケットとの関連を記述することで相互に参照可能になる）。ただし、データベースのテーブルスキーマに更新が必要な場合は手動で作業する必要がある。

2.2 構成管理・動作確認環境

課題担当教員により用意されたインターネットに接続可能な開発管理サーバ（CentOS5.5）に、プロジェクトを進める上で必要となる環境を構築した。

1. Subversion

   構成管理に使用するため、Subversion のインストール・設定を行った。プロジェクト用リポジトリに、コードのほか、プロジェクト関連資料、議事録、メモをそれぞれディレクトリに分け管理する。

   リポジトリには開発メンバのみがアクセスできるよう、Apache の設定により、認証を必要とするようにした。

   リポジトリのバックアップは、開発メンバが高頻度にて最新作業コピーを有していることからその優先度を下げ、開発中の作業工数の問題から行わなかった。

2. Redmine

   Web ベースのプロジェクト管理ソフトウェアであり、チケットの管理を中心に、ニュース・文書・ファイル等の管理を行うことができる（図 2.3）。チケットの管理においては、Subversion と連携しチケットとリビジョンやファイルに関連を張る、チケットの更新時に関係者にメール通知を行う、などといった機能を備えている。この Redmine のインストールと Subversion 連携設定等の初期設定を行った [4]。バックアップは、Redmine の使用しているデータベースとアップロードされたファイルを 1 日 1 回（cron により）取得し、これを管理用リポジトリ（前述のプロジェクト用リポジトリと別のもの）に自動でコミットするように設定、これを複数人にて作業コピーに取得する形で行った。

   Web ベースのソフトウェアであるため、開発メンバはもとより、課題担当教員、遠隔地にいる日立技術者もプロジェクトの状況を把握し、チケットを通じてアドバイスを行うこともできる。Redmine 自体は Web 上どこからでもアクセス可能としたが、プロジェクトの情報自体は関係者以外に公開すべきでない情報を多分に含むため、プロジェクトの情報はログインしたユーザのみが取得できるよう設定した。

3. 動作確認用実行環境

   スプリントは動作するソフトウェアの機能追加という形で進められるため、統合されたコードは動作することになる。そこで最新のリビジョンのコードの動作を容易に確認するため、動作確認用実行環境を構築した。この実行環境への最新リビジョンからのデプ

ロイは、Subversion リポジトリへのコミットに対して自動的に行われるよう、Subversion のフックスクリプトによって設定した。

開発管理サーバにはインターネットからアクセス可能なため、（サーバの認証ユーザとして登録されている）プロジェクト関係者は全員が動作を確認できるようになっている。

図 2.3: Redmine

2.3 作業環境

開発メンバの普段使用している PC 等に、スプリントにおいて実装作業を行うための開発作業環境を構築する。方法としては次のソフトウェアを、文字コード等共有したい事柄の設定を行った上で共有するという形を取った。

1. EclipsePDT+Subvertive

   IDE としてメンバに使用者の多い Eclipse の PDT を、Eclipse 用 Subversion クライアントである Subvertive と合わせて開発作業環境として使用する。

   開発環境サーバのリポジトリには、Eclipse プロジェクト全体を登録する。これにより実装環境の一部の設定を共有する。

2. 動作確認用実行環境（XAMPP）

   動作確認用の実行環境として XAMPP を使用する。これにより開発メンバは自分の担当部分の実装を進めた後、リポジトリにコミットし共有する前に、（担当部分の独立した）動作確認を行うことができる。

2.4 評価実験運用環境

評価実験時には運用環境を構築、管理し、また threpo の管理者を担当した。運用中のバックアップは cron により定期的（4 時間毎）に作成した。threpo の管理者としては主にユーザの登録を行い、またパスワードを忘れた実験参加者へのパスワードの再発行や、実験参加者からの質問などへの対応を行った。この際の運用環境構築の手順を元に、threpo 運用書を作成した。

第3章　開発担当

本プロジェクトにおいて筆者が開発を担当した部分について述べる。

3.1　プロトタイプ作成

スクラム開発において各スプリントは、「実行可能なソフトウェアに対して機能追加をする」といった形に計画・実施される。よって最初のスプリント実施の前に、機能の追加先となる実行可能なソフトウェアが土台として必要になる。また使用する開発言語・フレームワークに対しての知識獲得も目的の一つとなる。

本プロジェクトでは、要件定義とプロトタイピング、処理方式設計を同時に実行した。その様子を図 3.1 に示す。開発構想書の担当者が要件定義を進める中で、プロジェクトメンバはシステムのイメージを話し合い、それぞれの担当者が実現するための処理方式の設計とプロトタイプの開発を進める。また、プロトタイプ開発によって得られたフレームワーク・開発環境等に関する知識は各担当者にフィードバックされ、それぞれの成果物に反映される。

3.2　開発担当分担

プロトタイプ開発者として、処理方式設計者等とミーティングを行い、1st スプリント以降の開発担当分担の決定に携わった。特にプロトタイプを開発する中で効果的に作業可能になるであろう分担を目指す立場としてミーティングに参加した。

開発システムは、フレームワークとして CakePHP を使用した。CakePHP は MVC モデルを採用しており、コンポーネント間の命名規則を元にした動的結合、データのスキーマ定義から簡易な CRUD を作成するスキャフォールド機能等を備えている [7]。

本プロジェクトの開発分担の方針として次の分担方法が採用された。これはプロトタイプ開発の経験から提案を行ったものであり、図 3.2 は開発メンバに向けて説明を行うために作成した CakePHP の MVC モデルを示したものである。

- モデルは単体での動作確認が容易であり、データのスキーマ定義の管理と密接な関係があるため、全てのモデルはプロトタイプ作成者が開発する。
- コントローラを中心に行われる命名規則による動的結合が多くあるため、IDE の入力支援が機能せず、結合されるコンポーネント間のすり合わせが頻繁に必要になる。そのた

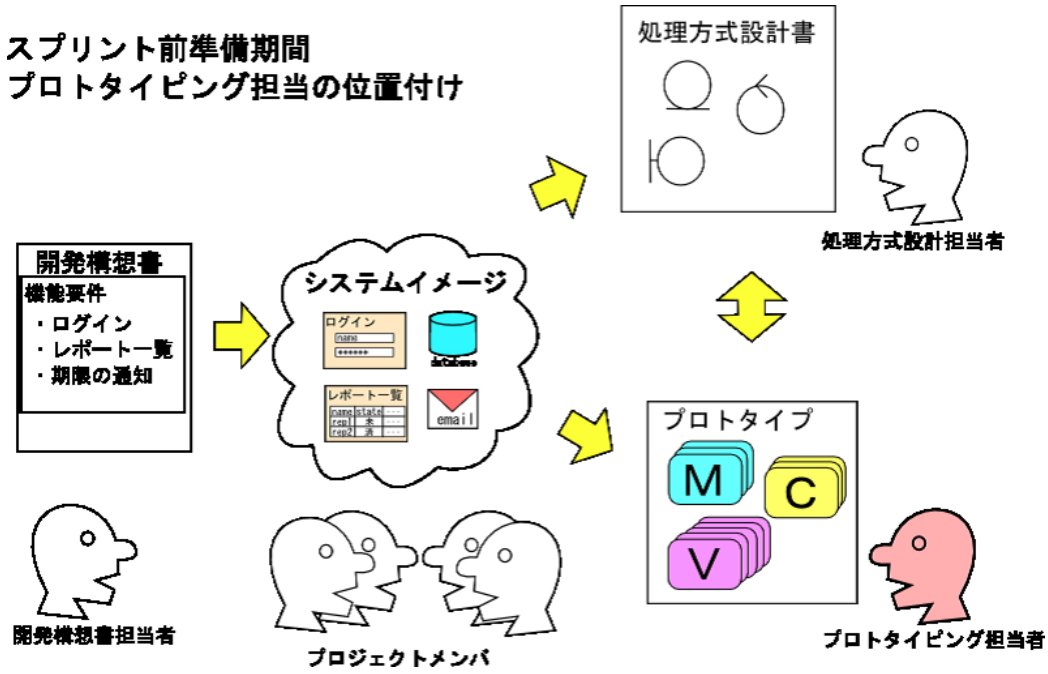

図 3.1: 要件定義（開発構想書）、プロトタイプ、処理方式設計の関係

め、コントローラと機能を対応させ、開発担当はコントローラ単位に割り振り、コントローラに対するビューは同一者が開発する。

- プロトタイプではスキャフォールド機能を利用し、動作確認に利用可能な CRUD コードを実装した。この自動生成されたコードを仮実装とし、置き換える形で開発を進める。

1st スプリントの開発主目的達成のため、次の通りに開発分担を行った。

- モデル部分＋フレームワーク関連部分（モデル担当者）

  前工程にて作成したプロトタイプに深くかかわる部分は、プロトタイプ担当者が引き継ぐ。主にモデル部分を担当し、他の開発メンバにデータへのアクセス手段を提供する。

- 技術的懸案事項の解決

  開発メンバ間にとって未知であること、もしくは知識面に不足であることなど、技術的懸案事項の解決を中心に担当する。特にシステムの特徴となる、メールによる通知に関する懸案事項の解決を優先的に解決する。開発メンバの中で、開発言語とフレームワークに関し一定の知識を持つメンバ 1 名がこの担当を行う。

- 開発主目的の機能（機能担当者）他 2 名は開発主目的である「レポート提出状況の管理」と「それのために必要なデータ管理のうちの入力部分まで」をそれぞれ担当する。

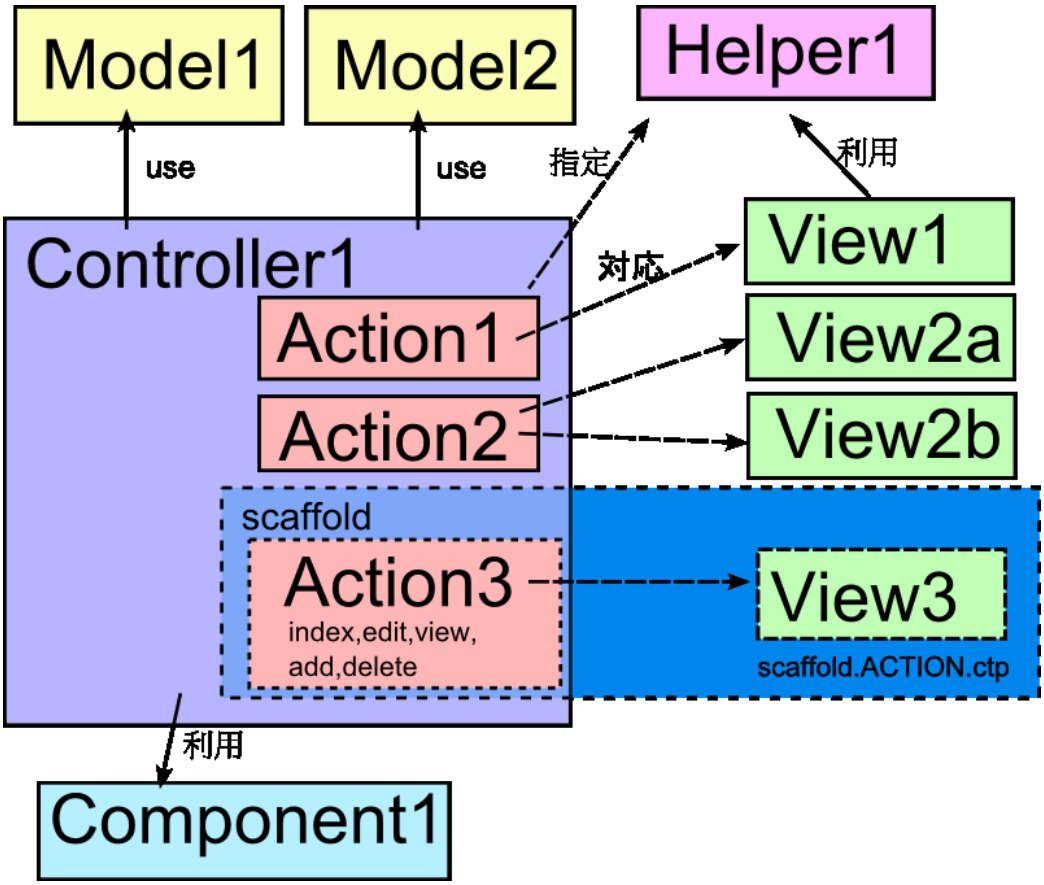

図 3.2: CakePHP の MVC モデル

図 3.3 は、モデル担当者と機能担当者の作業分担上の関係を示したものである。① と② はそれぞれ担当者間で連携が必要であることを示している。1st スプリントでは特に、① では密な連携が必要となる。

この分担により作業は次のように行われる。

1. モデル部分担当者と機能担当者により、必要なモデルインタフェース（図 3.3 ①）を決定する
2. モデルインタフェースの宣言のみ作りリポジトリへ登録する
3. 各担当者は担当部分の開発を行う
   - モデルインタフェースに機能担当者は動作確認用のダミーデータを出力するコードを仮実装し、次の手順で機能の開発を行う
     (a) 結合テストの計画
     (b) 実装

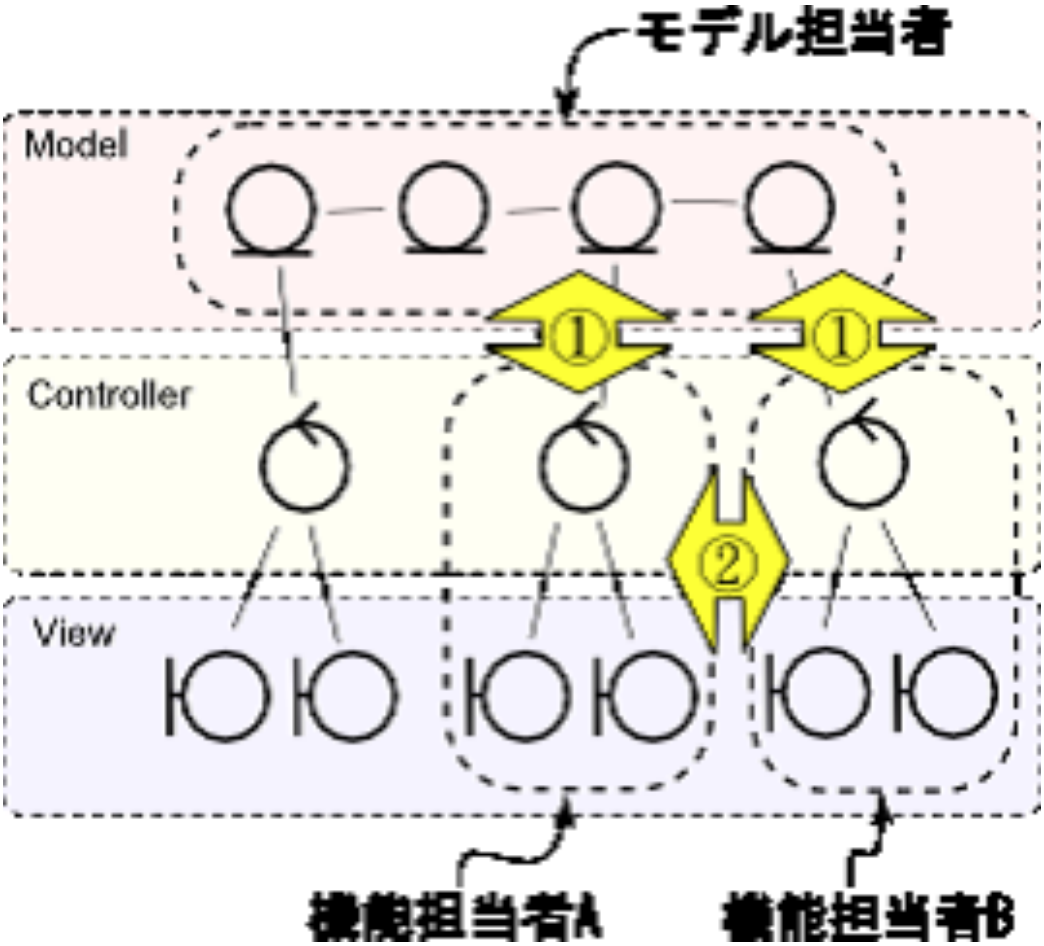

図 3.3: 開発担当分担のイメージ

(c) 結合テストの実施

- モデル担当者は各モデルインタフェースを開発する
  (a) 単体テストの simpletest による記述
  (b) 実装
  (c) simpletest による確認

4. モデル担当者と機能担当者の実装を結合し、テストを行う

1st スプリントの作業結果として、作業量上の問題から上記の結合テスト／ simpletest の実施が行えなかった。また作業環境上での動作確認が容易であり、かつ作業分担上、各メンバの編集による他メンバの担当部分に対する影響が少ない（編集ファイルの競合が少ない、統合したコードにおけるバグの切り分けが容易である）ことが 1st スプリントの実施により明ら

かになった。2nd スプリント以降においては作業工数上、計画したテストの完全な実施が難しいと判断し、以上の理由からこれらのテストの優先度を下げることに決定した。

2nd スプリント以降ではモデル担当と機能担当という分担が順調に動作したことから、これを引き継ぐ形の作業分担を行った。そのスプリント開始時に開発すべき機能を全て上げ、各メンバの作業量が同程度になるように分担した。

3.3 モデル開発

前述のとおり、モデル開発者として機能開発者とともにモデルインタフェースを決定した。この際の手順は次のように行った。

1. 機能担当者から実装上の計画を記述してもらう

   機能担当者が考えている実装方法をシーケンス図などを用いて図示し、その中でモデルに必要となる動作を示す。

2. モデル担当者と機能担当者がミーティングを行う

   上述の図を元に実際のインタフェースを決める。この際実装方法のレビュを行いながら、インタフェースを調整する。

3. 仮コードを作成する

1st スプリントにおいては、モデル担当と機能担当の作業分担に曖昧な部分があったが、ミーティングを行う中でモデル担当者を通して実装方法の方針統一を図ることができた。ただし決定したインタフェースの検証を実装前に行わなかったため、インタフェースの不備が 10 件前後あった。これらの修正自体は規模の小さなものであったことから、判明次第デイリーミーティングなどにおいて報告される度、すぐに修正する形にて対応することが可能であった。そのため 2nd スプリント以降も同様の作業手順を取ることとした。

最終的なモデルのコード量を表 3.1 に示す。

3.4 ユーザ管理機能

2nd スプリント以降においてユーザ管理機能の開発を行った。開発した機能は以下の通りである。

- ログイン（ログアウト）

  すべてのユーザが、ログイン画面（図 3.4）を通じて threpo にログインする。threpo のすべての機能はログインを必要とする。ログインにはログイン ID とパスワードを使用し、管理者以外のユーザのログイン ID はシステムに登録したメールアドレスとなる。ページ下部にはパスワードの忘却等への対応のため、システム管理者のメールアドレスが表示される。

表 3.1: モデルコード量

モデル名	説明	有効行	コメント行	合計
Assignment	課題	376	132	508
Registration	科目の履修関係	38	9	47
ReportAcceptingFlow	各レポートの状態	782	546	1328
StudentInfomation	学生情報	201	41	242
Subject	科目	619	280	899
SubjectProvider	科目の担当関係	43	15	58
TeacherInfomation	講師情報	174	56	230
TeachingAssistant	科目の TA 関係	50	8	58
User	ユーザ	277	67	344
WaitingEmail	送信待ちメール情報	45	34	79
計		2605	1188	3793

- ユーザの登録

  管理者のみが使用する。講師、学生ユーザを登録する。登録は一人づつ情報を入力する形式と、複数人を同時に入力する形式を用意し、複数人のユーザを登録する場合にはランダムパスワードの生成と、メールによるアカウント通知をオプションとして利用できる。

  図 3.8、図 3.5 はそれぞれ、講師、学生を一人登録する形式の画面である。1 人づつの登録を行う場合、パスワードは明示的に入力をする。図 3.6 は、学生を複数人同時に入力する形式の画面である。入力方法は csv の方式であり、登録時パスワードはランダムな文字列が与えられ、登録結果画面（図 3.7）に表示される。

- ユーザの一覧

  管理者のみが使用する。図 3.9 のような画面により、講師のみ、学生のみまたはすべてのユーザの一覧を表示し、ここから次のユーザ情報の編集が可能である（画面は学生ユーザの一覧）。

- ユーザ情報の編集

  各ユーザはページ上部のリンクから、自身のメールアドレスとパスワードを変更することができる（図 3.10）。管理者のみ、すべてのユーザのすべての情報を（一覧画面よりユーザを選択することで）編集可能である（図 3.11）。

表 3.2: ユーザ管理機能コード量

ファイル	説明	有効行	コメント行	合計
View(15 ファイル)				
add_ student.ctp	学生の追加	53	4	57
add_students.ctp	学生の一括追加	90	3	93
add_students_result.ctp	学生の一括追加結果	74	4	78
add_teacher.ctp	講師の追加	51	3	54
add_teachers.ctp	講師の一括追加	92	3	95
add_teachers_result.ctp	講師の一括追加結果	71	4	75
edit_student_profile.ctp	学生情報のユーザ自身による編集	32	3	35
edit_student_profile_by_admin.ctp	学生情報の管理者による編集	52	3	55
edit_teacher_profile.ctp	講師情報のユーザ自身による編集	32	3	35
edit_teacher_profile_by_admin.ctp	講師情報の管理者による編集	51	3	54
index.ctp	全ユーザの一覧	84	3	87
list_student.ctp	学生の一覧	85	3	88
list_teacher.ctp	講師の一覧	83	3	86
login.ctp	ログイン	31	1	32
save_admin.ctp	管理者情報の編集	72	4	76
Controller(1 ファイル)				
users_controller.php	ユーザ管理機能	573	155	728
計		1526	202	1728


threpo: report state management system :Users - Mozilla Firefox
Google
http://localhost/threpo/
threpo: repor...
threpo
ログインページ
threpoへようこそ！
ログインに必要な情報を入力してください。
ユーザ名*
ログイン名(メールアドレス)を入力してください。
パスワード*
パスワードを入力してください。
ログイン
パスワードを忘れた場合等は、管理者に連絡してください(ashiojin@gmail.com)
threpoに関してのご意見・ご質問先: 森有司(ashiojin@gmail.com)

図 3.4: ログイン画面

図 3.5: 学生ユーザの登録

図 3.6: 学生ユーザの一斉登録

図 3.7: 学生ユーザの一斉登録結果

図 3.8: 講師ユーザの登録

図 3.9: 学生ユーザの一覧

図 3.10: ユーザ自身による学生情報の編集

図 3.11: 管理者による学生情報の編集

第4章　まとめ

本プロジェクトでは、株式会社日立製作所ソフトウェア事業部の提案テーマに基づき、教員のレポート管理コストを低減し、また学生のレポート提出忘れのリスクを低減することを目的に、「学内レポート管理ワークフローの開発」を行った。本プロジェクトではチケット駆動のプロジェクト管理を行い、また開発手法としてスクラムを使用した。本稿ではこれらの概要とプロジェクト計画について述べたうえで、筆者がプロジェクト全体に渡って担当した開発環境管理と、実装を担当したシステムのモデル部分及びユーザ管理機能についての報告を行った。

謝辞

株式会社日立製作所ソフトウェア事業部の居駒幹夫氏、弥生隆明氏、公立はこだて未来大学の大場みち子教授には、本プロジェクトを進めるにあたり多くのご指導とご協力を頂いている。また課題担当教員の天笠俊之准教授、指導教員の田中二郎教授には、本報告書の執筆を含め、ご指導を頂いた。ここに感謝の意を表する。最後に、本プロジェクトを共に遂行した河村翼氏、韓貞美氏、カンキスルン氏に感謝の意を表する。

参考文献

[1] 戸田保一, 速水治夫, 飯島淳一, 堀内正博. ワークフロー―ビジネスプロセスの変革に向けて. 日科技連出版社. 1998 年

[2] 藤井拓. アジャイルモデリングへの道 2005 年 5 月号 第 2 回：スクラム組んで開発しよう!. http://www.ogis-ri.co.jp/otc/hiroba/technical/IntroASDooSquare/chapter2/IntroScrumCaseStudyMay2005.html. 2010 年 11 月 1 日参照

[3] Ken Schwaber, Mike Beedle. アジャイルソフトウェア開発スクラム. ピアソンエデュケーション. 2003 年 9 月

Ken,S, Mike,B. アジャイルソフトウェア開発スクラム (長瀬嘉秀, 今野 睦, スクラムエバンジェリストグループ訳). 株式会社ピアソンエデュケーション. 2003

[4] 前田剛. 入門 Redmine 第 2 版 Linux/Windows 対応. 秀和システム. 2010 年 8 月

[5] 小川 明彦, 阪井 誠. Redmine によるタスクマネジメント実践技法. 翔泳社. 2010 年 10 月

[6] 岸田健一郎，安藤祐介，新原雅司，CakePHP による実践 Web アプリケーション開発. 株式会社毎日コミュニケーションズ. 2009 年 4 月

[7] マニュアル :: 1.3 コレクション :: The Cookbook http://book.cakephp.org/ja/ 2010 年 09 月 1 日参照

[8] Moodle.org (http://moodle.org/) 2010 年 09 月 1 日参照

付録

開発構想書

レポート課題の提出ワークフロー管理システム

threpo

河村翼
カンキスルン
韓貞美
森有司

第 2.0 版

平成 22 年 12 月 17 日

改訂番号	日　付	作成者	説明
0.0	9月5日	河村	草案作成。
0.1	9月28日	河村	
0.2	10月4日	河村	
1.0	10月5日	河村	
1.1	10月28日	河村	外部資料から参照される部分にユニークなIDを追加した。
1.2	11月8日	河村	機能要件の追加、修正。また、優先度の変更。
1.3	12月2日	河村	機能要件、非機能要件の修正。
2.0	12月17日	河村	

目次

1　はじめに.....1
1.1　システムの概要.....1
1.2　参考文書.....2
2　ユーザに関する説明.....3
2.1　想定するユーザ.....3
2.2　ユーザ環境.....3
3　開発根拠.....4
3.1　ユーザヒアリング・アンケート.....4
3.1.1　教員.....4
3.1.2　非常勤講師.....4
3.1.3　TA.....5
3.1.4　学生.....5
3.2　現状業務.....5
3.3　開発根拠.....8
3.4　代替案および競合案.....8
3.4.1　現状維持.....8
3.4.2　Moodleの利用.....8
4　システム概要.....10
4.1　システム化の範囲.....10
4.2　ワークフロー.....11
4.3　システム構成.....12
4.3.1　前提事項.....13
4.3.2　制約事項.....13
4.4　システムの存在理由.....13
5　機能要件.....14
6　非機能要件.....19
7　システムの典型的な利用シーン.....21
8　ドキュメント要求.....23
9　用語辞典.....24

1　はじめに

本文書の目的は、プロジェクトの存在理由を明らかにすることである。対象とするユーザが抱える課題を明らかにし、この課題を解決するためにシステムに要求される事項を要件としてまとめる。この要件を満たすシステムを開発することが本プロジェクトの目的である。

システムの具体的な振る舞いについては保守書に記述することとし、本文書ではシステムの要件を中心にシステムの概要を述べる。

1.1　システムの概要

本システムは、筑波大学の授業におけるレポート課題の提出ワークフローを管理するシステムである。本システムが管理するワークフローについては 2 章以下で詳しく述べる。なお、レポート課題の媒体は紙、電子ファイルを問わない。また、授業中に行う演習課題の内、その授業内で解答が回収されるものについては、本システムが扱うレポート課題の範疇外とする。

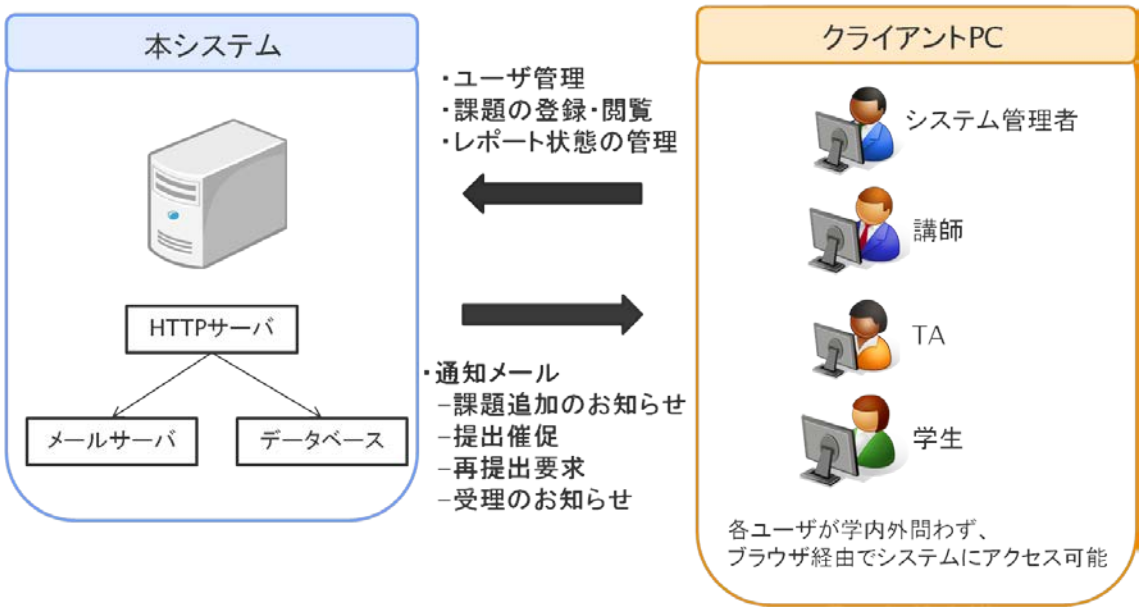

図 1　システム概要.

本システムは web アプリケーションであり、各ユーザはブラウザ経由でシステムを利用することが可能である。また、システムから規定されたタイミングで、通知メールが学生ユーザへ送られる。

レポート課題の提出ワークフロー管理システムを実現するために、本システムは次の機能を提供する。

表 1　システムが提供する機能群.

機能分類	説明、主な目的
ユーザ管理	ユーザをシステム上で識別すること。また、各ユーザが持つ権限を管理すること。
科目管理	科目ごとに課題やレポートの管理ができること。
レポート提出・受け付け	学生が電子ファイル形式のレポートを提出する際、また、講師がレポートを受け付ける際、この支援を行うこと。
レポート管理	ユーザがレポート提出状況を容易に把握できるよう支援すること。
通知	学生が期限内にレポートを提出できるよう支援を行うこと。例えば、レポート提出締め切り前に提出要求のメールをシステムが学生に送信することで、学生の提出忘れを防ぐこと。
レポートフィードバック	提出されたレポートに対するフィードバックがシステム上で行えること。

また、本システムは次の機能を提供しない。

表 2　システムが提供しない機能群.

機能分類	説明
コンテンツマネジメント	CMS(Content Management System)の実装は行わない。
システム連携	既存の CMS や、Twins(筑波大学の履修管理システム)などのシステムと連携しない。

1.2　参考文書

[1]Leffingwell,D.・Widrig,D.　石塚圭樹・荒川三枝子・日本ラショナルソフトウェア(2000)『ソフトウェア要求管理―新世代の統一アプローチ (Object Technology Series)』　ピアソンエデュケーション

[2]鶴保征城・駒谷昇一(2006)　『ずっと受けたかったソフトウェアエンジニアリングの授業 1』　翔泳社

2　ユーザに関する説明

本章では、本システムの利用者となるユーザを明らかにする。

2.1　想定するユーザ

筑波大学の講師、TA(Teaching Assistant)、学生の 3 種が本システムのエンドユーザとなる。講師は教員、非常勤講師の 2 種から成り、区別が必要な時はこちらの用語を用いてユーザの特定を行う。非常勤講師は主に一般企業に所属する者であると想定する。

ユーザすべてに使い勝手の良いシステムを提供することが望ましいが、今回は開発期間の都合上、「高度 IT コース(高度 IT コース人材育成のため実践的ソフトウェア開発専修プログラム)」に所属する学生、高度 IT コースの科目を担当する講師および TA をメインターゲットとし、こちらに焦点を当てたシステムを開発する。

高度 IT コースの科目は他の科目よりも非常勤講師が担当することが多いという特徴がある。今後、産学連携の活性化が進むことが予想されるため、非常勤講師を意識し上記のメインターゲットを設ける。

2.2　ユーザ環境

メインターゲットとなるユーザはコンピュータサイエンスに関わる者であるため、一般にスキルレベルは高いと想定する。

・使用マシン：学生はノートパソコンを 1 台は所持している。主に Windows®PC を使用。
・授業：学生は多くの科目を履修する必要がある。また、1 つの科目に対して複数人の講師が講義を行うことも少なくなく、各講師ごとにレポート課題を出すことも多い。この時、講師はレポート課題の採点結果を他の講師と共有する必要がある。1 科目における履修学生の数が非常に多くなる科目も存在する。

3　開発根拠

本章では、レポート管理の現状業務、課題の分析を行い、システムの開発根拠を明らかにする。

3.1　ユーザヒアリング・アンケート

ここでは、ユーザに対して行ったヒアリング・アンケート結果を分析することで、教員、非常勤講師、TA、学生の 4 種のユーザの視点から、それぞれが抱える課題を明らかにする。なお、ヒアリングは教員 4 名、非常勤講師 4 名、学生 12 名に対して行い、学生ヒアリングに基づき作成したアンケートを高度 IT コースの学生約 60 名に対して実施した（回答数は 29)。

ヒアリング・アンケートの結果、各ユーザが抱える課題として次のものが挙げられる。

3.1.1　教員

1. 提出されたレポートは一定期間保存しなければならず、特に紙媒体の場合、保存場所に困ること。
2. レポートの提出期限を守らない学生がいること。
3. LMS(Learning Management System)を利用することで、授業コンテンツがいつでもどこでも確認でき、授業に出席しなくなる学生が増えること。
4. TA がまとめたレポートを受け取る作業に時間がかかる場合があること。
5. 提出形式がメールである場合、学生が指定したサブジェクトで送ってくれないことがあり、教員が受信を見落とす危険性があること。
6. 締め切り後に提出されたレポートに対する減点処理などがめんどうであること。
7. 既存のレポート管理システム、ツールでは最終的に成績を付ける Twins との連携ができていないこと。

3.1.2　非常勤講師

1. レポートの受理確認が大変であること。
2. レポート期限を守らない学生への催促が大変であること。
3. 企業ではメールのドメインで受けとる、受け取らないの問題があるため、メールを使ったレポート提出に確実性がないこと。
4. 複数人に送ることもあり、その際、その中の一人だけの宛先が違っていたことがあり、その対処が大変であった。
5. 学生が同じレポートを複数人の講師に送る場合があり、すべての送り先へ正しく送ら

れない可能性があること。
6. 複数人の講師でレポート採点結果を共有する場合、最新のものがどれかわからなくなる危険性が増すこと。

3.1.3 TA

1. レポートの提出期限を守らない学生がいること。
2. 締め切り後に提出されたレポートに対する処理がめんどうであること。

3.1.4 学生

1. レポートの内容に関するフィードバックがないこと。あったとしても、いつのレポートに対するものであるか分かりづらいことがあること。
2. どのような課題が出ているのか、また、その詳細が分かりづらいことがあること。
3. レポートの締め切りが分かりづらいことがあること。
4. レポートの締め切りを忘れてしまうことがあること。
5. 正しくレポートが受理されていないにもかかわらず、レポートを送った学生本人は受理されていないことに気付かないことがあること。
6. LMS を使ってレポートを提出する際に用いるログイン ID, パスワードの管理が面倒であること。

3.2 現状業務

アンケート・ヒアリングから現行の業務フローを作成した。ただし、科目、課題、教員などにより業務フローは異なるということが判明したため、以下に載せる業務フローは一例である。以下の例では、担当者に TA を含むものと含まない時のある業務フローについて示している。図 2 は前者であり、図 3 は後者である。また、図 2、図 3 における業務フローで管理される成果物をそれぞれ表 3、表 4 に示す。

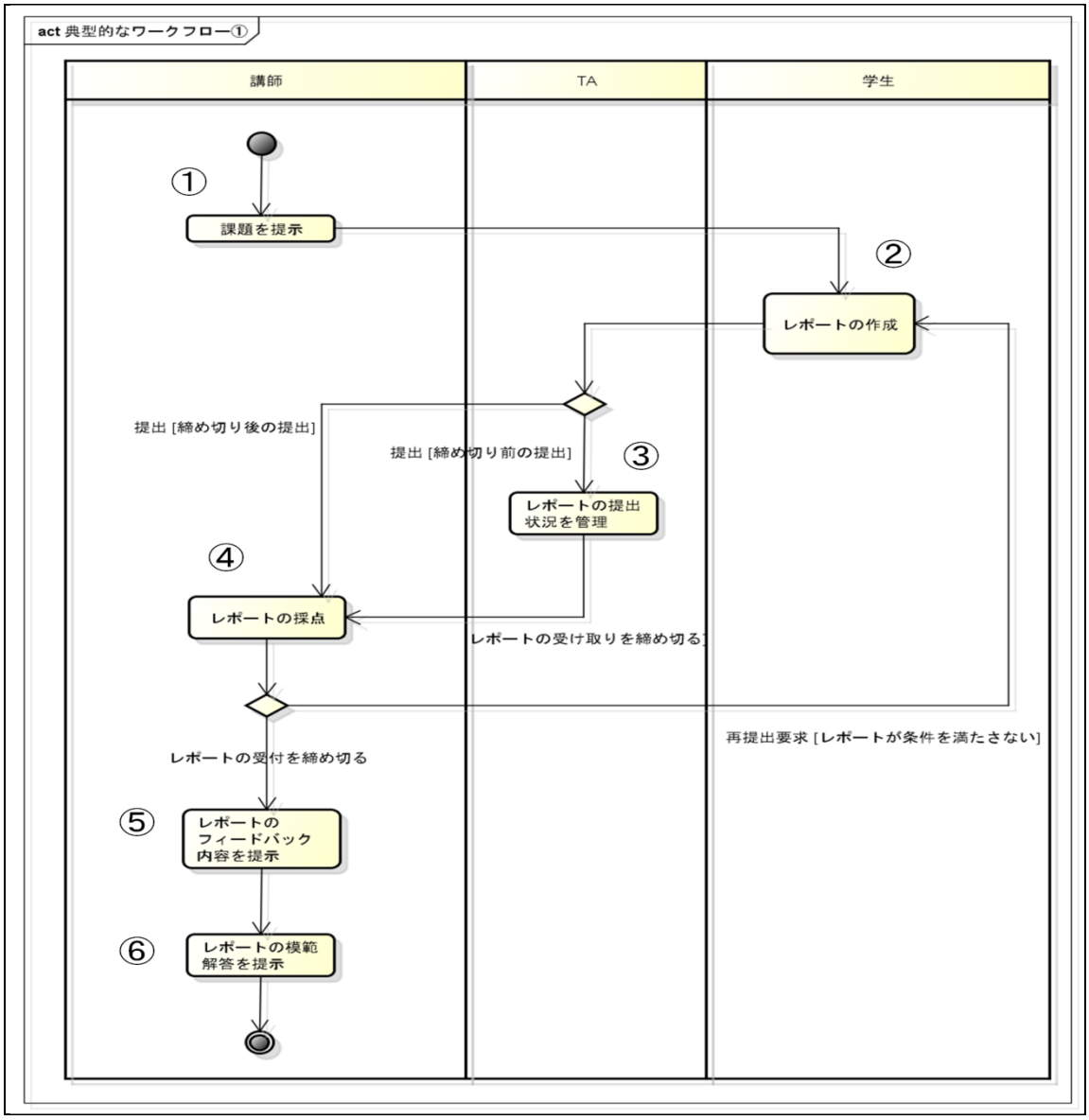

図 2　TA を含むレポート課題の提出業務フロー.

表 3　図 2 の各ステップにおける作業とフロー上を流れる成果物.

#	作業	情報
①	授業中に提示する	課題
②	電子ファイル形式で作成する	レポート
③	スプレッドシートを用いて管理する	レポート提出状況一覧表
④	提出日、レポートの内容などから採点する	レポート採点結果一覧表
⑤	レポートにコメントを付けて学生へメール送付する	レポートのフィードバック
⑥	模範解答を web 上に公開する	レポートの模範解答

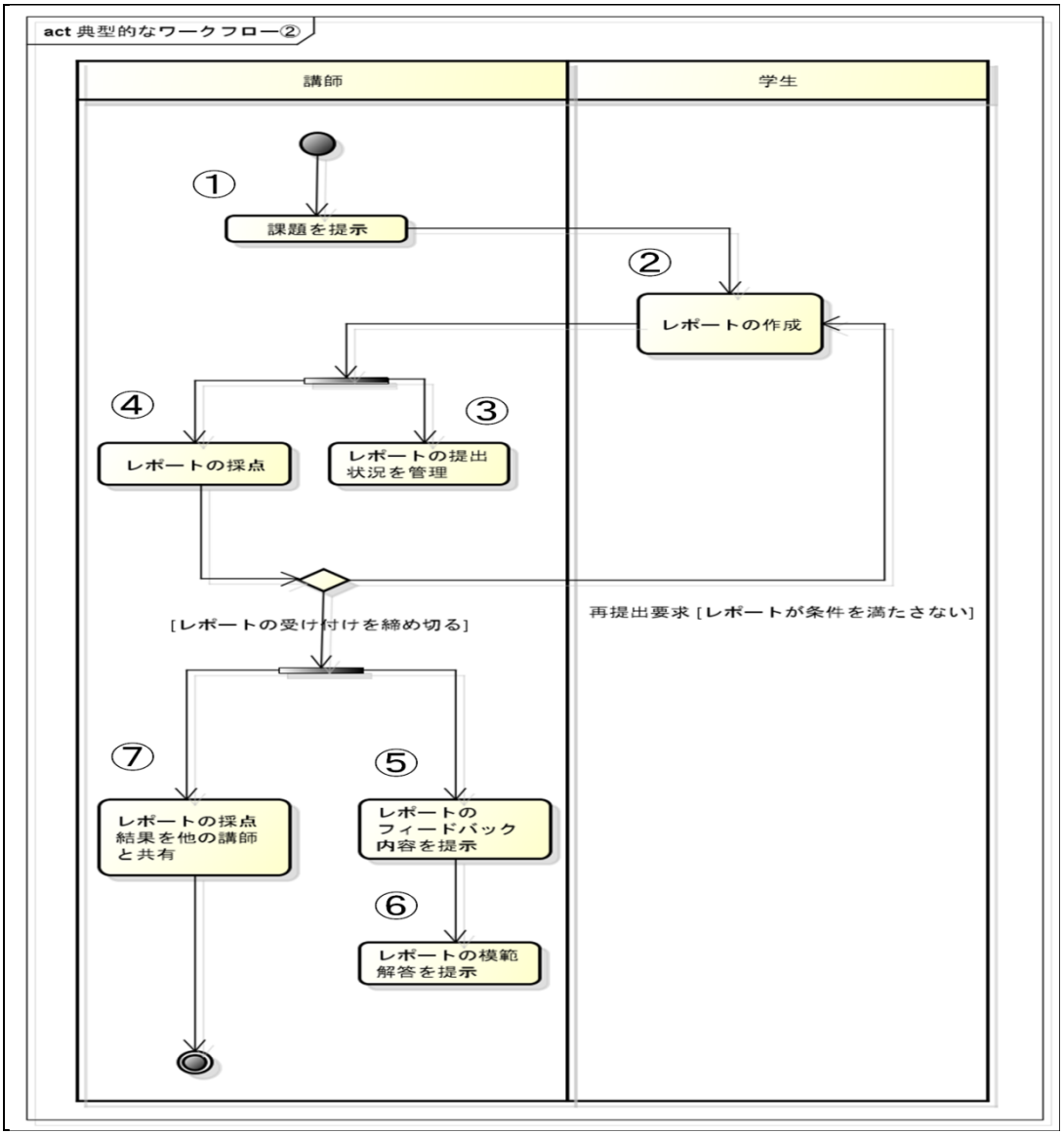

図 3　TA を含まないレポート課題の提出業務フロー.

表 4　図 3 の各ステップにおける作業とフロー上を流れる成果物.

#	作業	情報
①	授業中に提示する	課題
②	電子ファイル形式で作成する	レポート
③	スプレッドシートを用いて管理する	レポート提出状況一覧表
④	提出日、レポートの内容などから採点する	レポート採点結果一覧表
⑤	レポートにコメントを付けて学生へメール送付する	レポートのフィードバック
⑥	模範解答を授業中に提示する	レポートの模範解答
⑦	メールにて他の講師とレポート採点結果を共有する	レポート採点結果一覧表

表 3、4 で記述したレポート提出状況一覧表とは、履修生とレポートの提出状況の関係を表したものである。レポートの提出状況は、「レポートの提出締め切り日」、「提出日」、「レポート状態」の情報を持ち、レポート状態は、「未提出」、「提出済み」の状態を持つことが一般的である。この表はスプレッドシートの形で管理されることがほとんどである。レポート採点結果一覧表とは、学生と、この学生が提出した(提出しない場合も含む)レポートを採点した結果得られた点数の関係を表したものである。こちらの表もスプレッドシートの形で管理されることがほとんどであり、レポート提出状況一覧表と組み合わさり、1 つの成果物として管理されることも多くある。

3.3 開発根拠

ユーザヒアリング・アンケートから、各ユーザに共通してレポート提出期限に関わる事項に関心が高いことが判明した。特に非常勤講師に注目すると、学生の提出忘れへの対応は学内の教員に比べ、難しいことが想定される。本プロジェクトチームではこの点に着目し、学生のレポート提出忘れを防止するシステムに存在意義を見出す。また、ユーザヒアリング・アンケートから、学内のレポート管理業務フローは科目、課題、教員などの構成により異なることが判明した。そこで、ワークフローシステムの開発提案を行う前に、レポート課題提出業務フローのあるべき姿を提案し、その業務フローに対して上記で述べた存在意義を持つシステムを開発することとする。

3.4 代替案および競合案

ここでは、ユーザが利用できると認識している代替手段を明らかにする。それぞれの案の強み、弱みを知ることで、本システムの存在理由の指標とする。

3.4.1 現状維持

現状、多くの講師はスブレッドシートを利用し、レポート課題の管理を行っている。スプレッドシートの強みとしては利用者が自由にカスタマイズできるという点がある。履修生の数が少ない場合や、講師 1 人で科目を担当する場合において、スプレッドシートはレポート管理を行うのに十分足りるツールであることも事実である。だだ、学生に対するレポート提出を促す作業はこの範疇外であり、本システムと直接的に競合する案ではない。

3.4.2 Moodle の利用

既存の CMS である Moodle が筑波大学の授業支援のために導入されている。Moodle の強みは充実した授業支援コンテンツを持つことである。講師は Moodle を利用して課題を作成することも可能で、課題作成ツールとして利用できる。一方、今回のヒアリング・アン

ケートから、講師のレポート管理を困難にする大きな要因の一つに学生が提出締め切りを守らないという問題があることが判明した。Moodle はこの部分のサポートを提出リマインダメール機能でサポートしているが、学生に提出締め切りを守らせることに注力したシステムではなく、改善の余地が残ると判断する。本システムはこの部分で差別化を図る。

4　システム概要

4 章ではシステム化の範囲を明らかにすると共に、ユーザの抱えている課題を解決するシステムの概要を述べる。

4.1　システム化の範囲

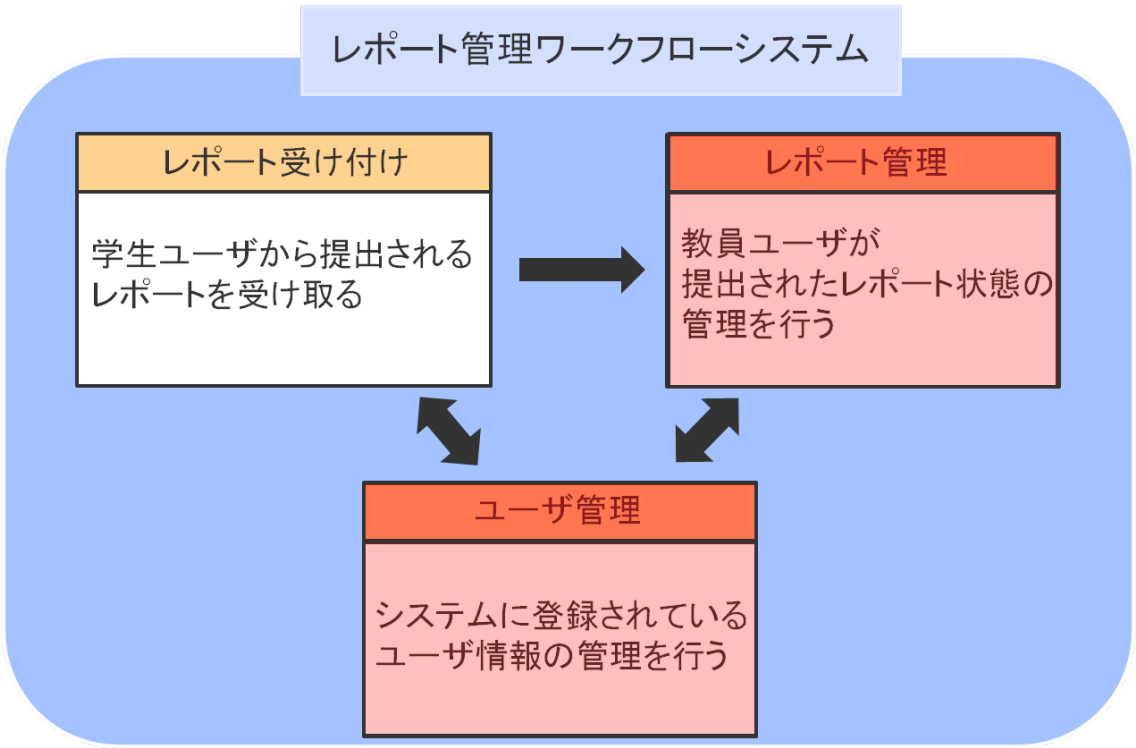

図 4　システム化の範囲.

図 4 は課題を解決するために提案するレポート管理ワークフローシステムである。レポート管理ワークフローシステムは「レポート受け付け」、「レポート管理」、「ユーザ管理」の三つのサブシステムから構成されるシステムである。今回は開発期間の都合と重要度を考慮した結果、「レポート管理」システムの開発のみ行う。このシステムの一部の機能として「ユーザ管理」を実現する。

4.2　ワークフロー

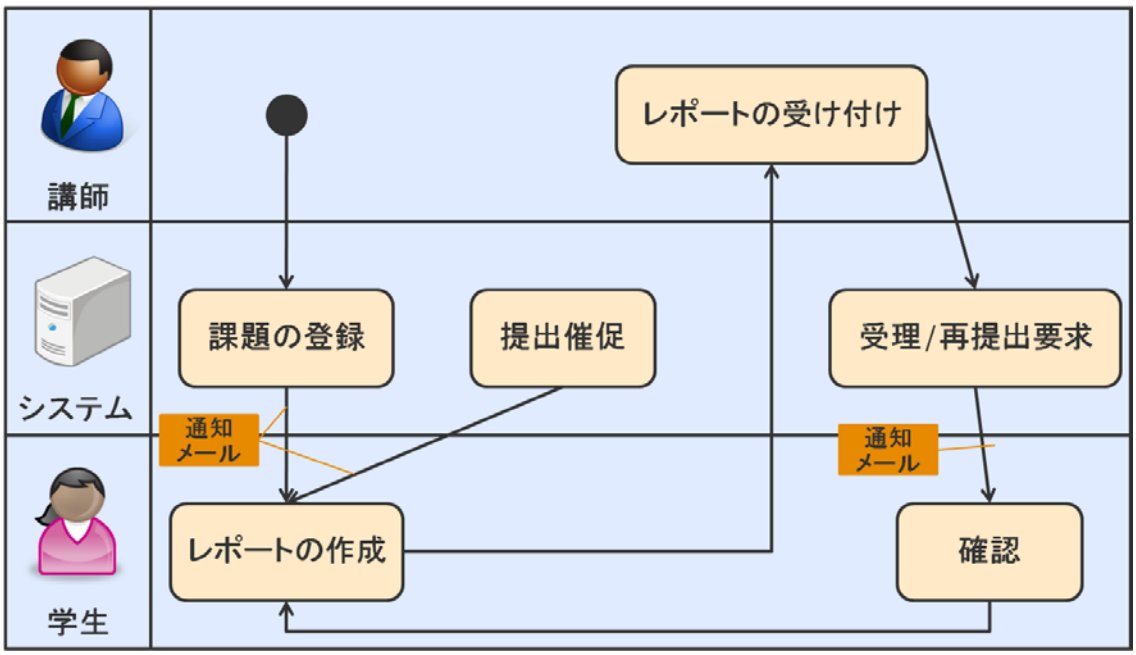

図 5　ワークフロー.

本システムはワークフローシステムである。ワークフローシステムは繰り返し行われる定型業務に向いている。現状、科目によってレポート課題の提出業務は様々であるため、すべての科目に共通する業務フローを定義することはできない。

そこで、ユーザヒアリング・アンケートを参考にしながら、各ユーザのニーズを反映した業務フロー（図 5 参照）を定義する。この業務フローの一部をシステムを利用することで自動化し、ワークフローとする。今回は図 5 における「課題の登録」から「レポートの作成」、「提出催促」から「レポートの作成」、「受理/再提出要求」から「確認」のフローをシステムがメール通知機能でサポートし、フローの自動化を行う。

4.3　システム構成

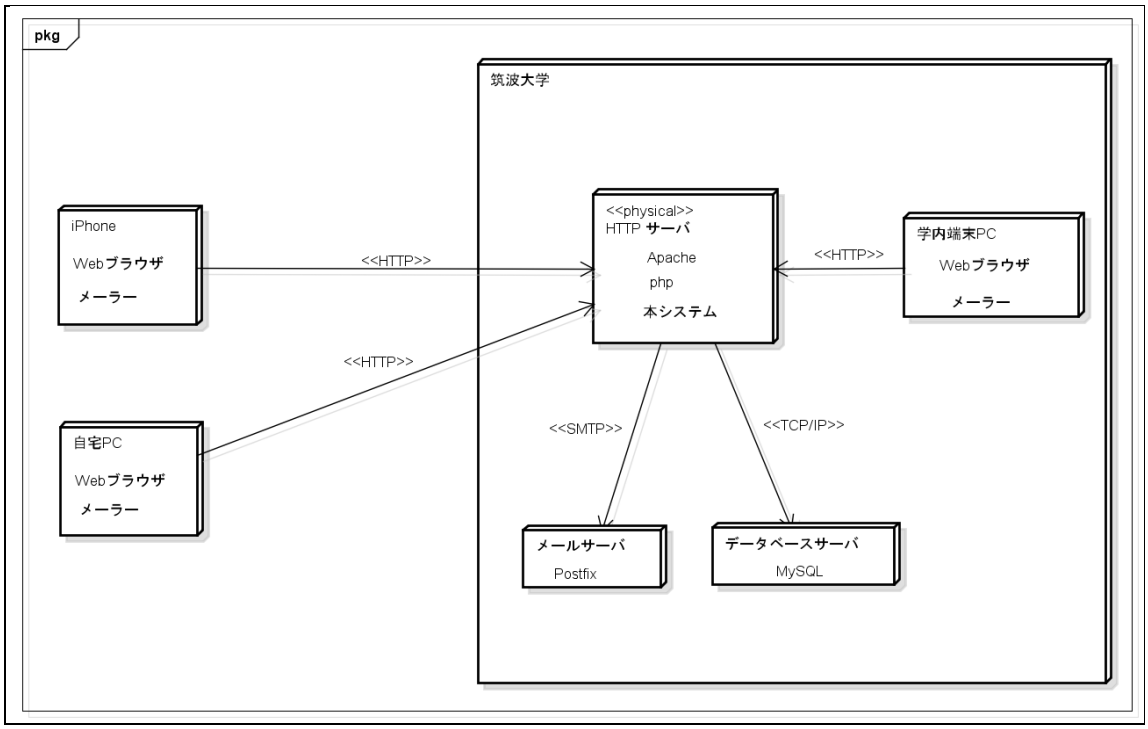

図 6　配置図.

図 6 に本システムの構成を示す。本システムは管理サーバ(HTTP サーバ、メールサーバ、データベースサーバ)、クライアント PC、iPhone から構成される。ユーザヒアリングから抽出した要望ではないが、今後利用が進むと予想される iPhone もクライアント端末として利用可能とする。

ユーザは学内外問わず、クライアント PC、iPhone から HTTP を使い管理サーバへアクセス可能である。また、本システムは、オープンソースである CakePHP を利用した Web アプリケーションである。

システムの実行環境は以下の通りである。

表 5　実行環境.

分類		名称
サーバ	OS	CentOS
	HTTP サーバ	Apache
	メールサーバ	Postfix
	データベースサーバ	MySQL
クライアント	端末	PC、iPhone
	ブラウザ	Firefox、Internet Explorer

4.3.1　前提事項

サーバ、クライアント共にインターネットに接続している必要がある。また、クライアント側は電子メールクライアントをインストールした端末を利用する必要がある。

4.3.2　制約事項

利用できるブラウザは一般的によく使われている Firefox、Internet Explorer とし、システム開発時の最新バージョンである Firefox3.6、Internet Explorer8 での動作を保障する。

4.4　システムの存在理由

本システムを利用することで、講師、TA のレポート管理コストを低減することが可能である。特に、システムを導入しない場合に比べ、提出期間内におけるレポート提出率の向上が見込まれる。また、締め切り後に提出されるレポートの管理も容易になる。一方、学生側から見ても、レポート提出忘れのリスク軽減が見込まれる。

他システムとの差別点としては、レポート状態を定義することにより、レポート状態の変更のタイミングでメール通知を行えるきめ細やかさと、レポート状態遷移の定義を変更することでメール通知タイミングを変更できる、通知機能における拡張性の高さである。

5　機能要件

システムの満たすべき機能要件を表 6 に示す。重要度を、高い順位に H、M、L の三段階設ける。重要度はユーザのニーズの高さ、システムの存在意義に関わるものほど高い。また、重要度を加味した上で、要件の関連、作りやすさの関係から優先度を重要度と同様に設定し、H の項目から実装していく。

システムユーザとして、講師、学生、TA のほかに管理者を追加する。管理者はシステム全体の管理を行うユーザであり、システム保守も行う。また、1 科目に対して、この科目を担当する講師の中から必ず 1 人、科目管理者を設定する(講師の中に権限を設ける)。科目管理者は履修生の成績を Twins へ登録する講師であるとする。

表 6　機能要件.

#	分類	機能要件	説明	重要度	優先度
FR1.3_1.	ユーザ管理	ユーザの 登録	管理者は本システムにログインできる講師(講師番号と名前)、学生(学籍番号と名前)、を登録できる。 なお、システム利用ユーザであるTAは学生としてユーザ登録する。	H	H
FR1.3_2.	ユーザ管理	ユーザの 削除	管理者は本システムに登録されているユーザ情報を削除できる。 ユーザ削除に伴い、科目や課題に関連しているユーザ固有の情報を削除すること。	H	L
FR1.3_3.	ユーザ管理	ユーザの 一覧	管理者は本システムに登録されているユーザを一覧できる。	H	H
FR1.3_4.	ユーザ管理	ログイン	利用者ごとに与える権限を変えるために、アカウント認証を行う。	H	H
FR1.3_5.	ユーザ管理	ユーザ情報の編集	各ユーザはシステム上に登録されている自身のユ	M	L

			ーザ情報を編集できる。ユーザ情報には学籍番号、氏名、E-Mailアドレスが含まれる。		
FR1.3_6.	科目管理	科目の登録	科目責任者は担当する科目を登録できる。	H	H
FR1.3_7.	科目管理	科目の削除	科目責任者担当する科目を削除できる。科目を削除する際、レポート提出状況一覧表、レポート採点結果一覧表、提出されたレポートがデータベース上に存在する場合は、明示的に削除されない限りこのデータはデータベースに保持する。	H	L
FR1.3_8.	科目管理	科目の変更	科目責任者は担当する科目情報を変更できる。科目情報には科目名が含まれる。	M	L
FR1.3_9.	科目管理	講師、TAの登録	科目責任者は担当する科目に講師、TAを登録することができる。	H	H
FR1.3_10.	科目管理	講師、TAの削除	科目責任者は科目に登録されている講師、TAを削除することができる。	M	L
FR1.3_11.	科目管理	履修生の登録	科目責任者は担当する科目を履修している学生を履修生として登録することができる。	H	H
FR1.3_12.	科目管理	履修生の削除	科目責任者は科目に登録されている履修生を削除することができる。	M	L
FR1.3_13.	科目管理	科目の一覧	講師、TAは担当する科目を一覧できる。また、学生は履修する科目を一覧	H	H

			できる。		
FR1.3_14.	科目管理	履修生の 一覧	講師、TAは履修生を一覧できる。	M	L
FR1.3_15.	科目管理	科目の 締め切り	講師は科目を締め切ることができる。 科目が締め切られると、科目に登録されている課題がすべて締め切られる。	H	L
FR1.3_16.	課題管理	課題の登録	講師は担当している科目の課題(課題概要、提出期限、提出方法)を登録できる。	H	H
FR1.3_17.	課題管理	課題の削除	講師は担当している科目の課題を削除できる。	M	L
FR1.3_18.	課題管理	課題の変更	講師は登録している科目の情報を変更できる。	L	L
FR1.3_19.	課題管理	レポートの提出期限の設定	講師は課題ごとにレポートの提出期限を設定することができる。	H	H
FR1.3_20.	課題管理	提出期限外の受け取り設定	講師は課題ごとにレポートの提出期限外にレポートを受け取るか否かの設定ができる。	H	H
FR1.3_21.	課題管理	課題の閲覧	科目に登録されている講師、学生、TAは登録されている課題をすべて閲覧することができる。	H	H
FR1.3_22.	課題管理	課題の提出締め切り	講師は課題を締め切ることができる。 提出期限外の受け取りが不可と設定されている場合(本表の項目番号19)、提出期限を過ぎた課題を自動的に締め切る。		
FR1.3_23.	レポート管理	レポート提出状	レポートが紙・電子ファ	H	H

		況の入力	イルであるかを問わず、講師は提出されたレポートを確認し、提出結果をシステムに入力する。(レポート提出状況一覧表が作成・更新されることに同意) レポート提出状況一覧表に含まれる情報については表4下の記述を参照		
FR1.3_24.	レポート管理	レポートの提出結果の一覧	学生は自身が履修している科目のレポートの提出結果を一覧できる。	H	H
FR1.3_25.	レポート管理	レポートの採点結果の入力	講師はレポートの採点結果をシステムに入力することができる。(レポート採点一覧表が作成・更新されることに同意)	M	M
FR1.3_26.	レポート管理	レポート採点結果一覧表の編集	講師はレポート採点結果一覧表を変更・閲覧することができる。	M	M
FR1.3_27.	通知	レポート提出要求メールの送信	講師が課題を登録する際に、履修生にレポート提出要求のメールを出すことが可能である。	H	H
FR1.3_28.	通知	レポート提出催促メールの送信(期限前)	システムは締め切り日時前の指定日に、自動的に履修生にレポート提出催促メールを送る。指定日は講師が設定する。また、学生は通知を受け取るか否かの選択が可能である。	H	H
FR1.3_29.	通知	メーラ立ち上げ	講師は、本システム上からメーラを起動すること	L	L

			が可能である。宛先は、対象の履修生全員、個々の履修生のいずれかを選択可能である。		
FR1.3_30.	通知	レポート提出催促メールの送信（期限後）	締め切り後にもレポート提出を受け付ける場合、システムは締め切りが過ぎても提出していない履修生に対してレポート提出催促メールを送る。 また、学生は通知を受け取るか否かの選択が可能である。	H	H
FR1.3_31.	通知	レポート提出結果メールの送信	講師がレポート状態を受理または差し戻しにした際に、自動的にその結果を学生にメールにて通知する。	H	H
FR1.3_32.	通知	レポート未提出の警告表示	締め切り日経過後であるにも関わらず、レポートが提出されていない学生にアクションを取らせる必要が出てきた際に、本システムは当該学生がアクセスした際に警告メッセージを表示する	H	H

6　非機能要件

システムの満たすべき非機能要件を表 7 に示す。

表 7　非機能要件.

#	分類	要件	説明	重要度	優先度
NR1.3_1.	性能	サービス提供時間	基本的に24時間365日とすること。 運用に関しては本表の項目番号12で記述。	M	M
NR1.3_2.	性能	データ量（メインターゲットユーザ）	最大登録数は次の通り。 講師情報：100件以上 （講師番号、名前、E-Mailアドレス） 学生情報：200件以上 （学籍番号、名前、E-Mailアドレス） 科目情報：200件以上（科目名、概要） 課題情報：1000件以上（課題名、概要）	H	H
NR1.3_3.	性能	データの保存	明示的にデータが削除されない場合、3年間以上保存可能であること。	H	M
NR1.3_4.	性能	アクセス環境	学内・学外からのアクセスが可能であること。	H	H
NR1.3_5.	性能	オンラインレスポンスタイム	全ての操作が3秒以内に終了することを目指す。ただし、利用者のネットワーク環境に左右される部分もあるため、ベストエフォートとする。	M	L
NR1.3_6.	信頼性	トランザクション数	少なくとも100,000トランザクションは連続処理可能であること。ここでのトランザクションとは、ユーザがブラウザを利用	H	H

			してシステムへアクセスし、応答が来るまでの処理である。		
NR1.3_7.	操作性	使いやすいユーザインタフェース	各画面のデザインに統一性を持たせること。	H	M
NR1.3_8.	操作性	ログインの容易さ	ユーザのログインが容易に行えること。	H	L
NR1.3_9.	保守性	MVCモデルに基づいた全体設計	システムの保守・管理は委託元に引き継ぐため、保守・管理が容易に行えるよう設計すること。	H	M
NR1.3_10.	保守性	文字コード	UTF-8を用いること。	H	H
NR1.3_11.	運用面	データのバックアップ	システム障害に備えるため、システム運用者は1日1度、データのバックアップを取ること	H	H
NR1.3_12.	運用面	メンテナンス	週に一度、30分程度メンテナンスのためサービスを停止する(メモリリーク対策のためのリブート作業を行う)。	H	H

7　システムの典型的な利用シーン

本章ではシステムの典型的な利用シーンを挙げる。

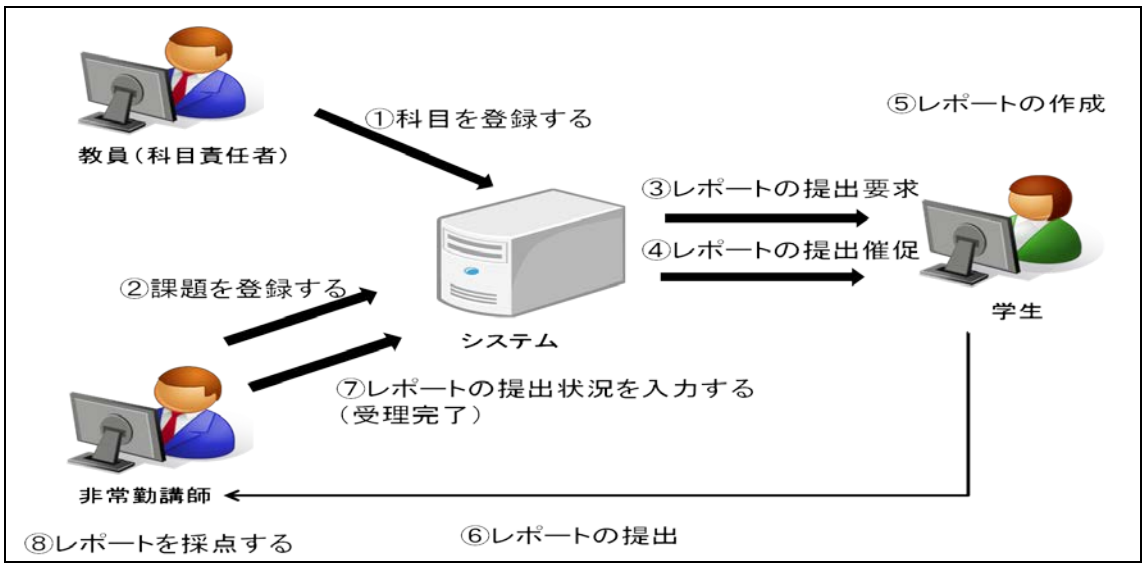

図 7　科目の登録からレポート受理までの利用シーン.

図 7 は教員(科目管理者)が科目を登録する時点から学生がレポートを提出し、受理されるまでのフローにおけるシステムの利用シーンを示している。非常勤講師がシステムに課題を登録(②)すると、自動的に学生に対してレポート提出要求のメールが届く(③)。また、レポートが未提出の学生に対して、提出催促メールを送る(④)。

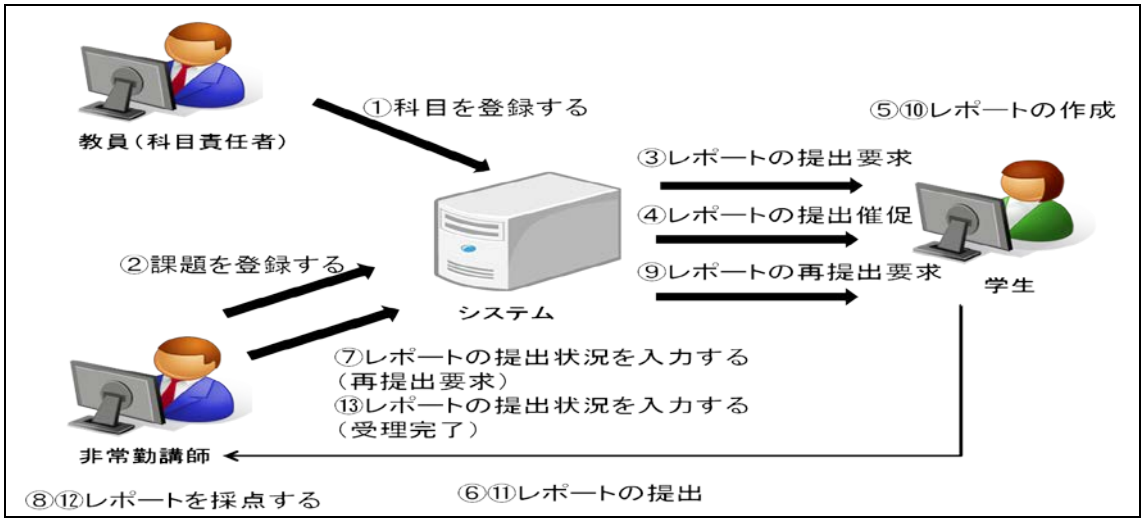

図 8　科目の登録からレポート受理までの利用シーン(再提出要求含む).

図 8 は上記の利用シーンの中と似ているが、レポートの再提出要求を行う点が異なる。講師が提出されたレポートを採点した結果、レポート内容が講師の課した条件を満たさない場合は再提出の要求が学生へ通知される(⑨)。

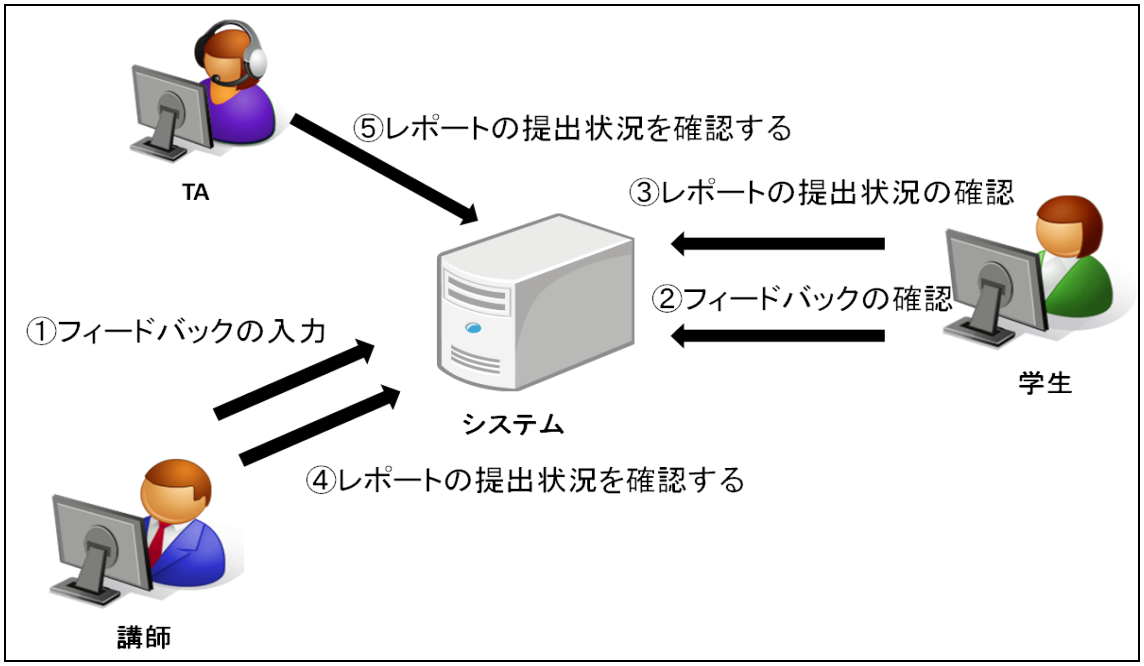

図 9　レポート受理後の利用シーン.

図 9 はレポート受理後の利用シーンを示している。講師、TA は履修学生全員のレポートの提出状況が一覧でき、学生は自身が出したレポートの提出状況が確認できる。また、講師はレポートに対するフィードバックの入力が可能であり、学生はそれをシステム上で確認できる。

8　ドキュメント要求

本プロジェクトでは以下の文章を成果物として生成する。

・開発構想書(本文書)

プロジェクトの目標設定を明文化したもの

・保守書

本システムを運用、再利用する人に向けた文書

・使用の手引

本システムを利用するエンドユーザ(講師、TA、学生)に向けた文書

9　用語辞典

本文書で用いる用語とそれに対する説明を次に示す。

表 8　本文書で用いる用語の説明.

番号	用語	説明
1.	業務フロー	業務の流れ
2.	ワークフロー	業務フロー全体または一部を自動化した業務フローであり、本システムが管理するフローである。
3.	課題	講師が履修生に出すレポート課題
4.	レポート	課題に対する解答
5.	CMS	デジタルコンテンツの管理を行うシステム。Wiki やブログ CMS にあたる。
6.	LMS	学習を支援するためのシステム
7.	講師	本システムユーザの 1 人。授業を提供する者。
8.	教員	講師の区分の 1 つ。正規に大学に雇用されている者。
9.	非常勤講師	講師の区分の 1 つ。企業講師であることが多い。
10.	TA	本システムユーザの 1 人。担当講師の指示のもと、授業の補助を行う者。学生からレポートを受け取る業務が多い。
11.	レポートの提出期限	正規なレポート提出が可能な期間。講師がこの期間外にレポート提出を認める場合もある。
12.	課題の締め切り	レポート提出の受け付けを終了すること。

threpo 運用書

第 1.0 版

平成 23 年 1 月 12 日

改訂番号	日　付	作成者	説明
1.0	2011年1 月12日	森	作成。

文書概要

threpo の運用に必要な情報として、
必要環境、インストール（設定を含む)、バックアップについて述べる。

必要環境

threpo の動作に必要な環境は次の通り。

- サーバ(CentOS 5.5)
  - Apache 2.2.3
  - PHP 5.2
    - ✧　CentOS5.5 では PHP5.2 のインストールが必要となることに注意
  - MySQL 5.1

- 他
  - 利用可能な SMTP サーバ

インストール

threpo のインストールには、以下のことを行う。

1. データベースの作成
2. コードのデプロイ
   (ア) threpo アプリケーションコードのデプロイ
   (イ) threpo アプリケーションドキュメントルートのデプロイ
   (ウ) CakePHP ライブラリのデプロイ
3. 設定項目の編集
4. 管理者ユーザの作成
5. cron の設定

以降では、次の位置に threpo をインストールする場合を説明する。

- threpo アプリケーションコード
  - /var/www/cakeapps/threpo
- threpo アプリケーションドキュメントルート
  - /var/www/html/threpo
- CakePHP ライブラリ
  - /var/www/cakeapps/cakephp-cakephp-efb6e08

また/var/www/html 以下には Apache により公開され、.htaccess の使用が認められている必要がある。

データベースの作成

threpo 用のデータベースを作成する。
データベースのエンコードを utf8 に設定すること。
またそのデータベースに接続するためのユーザを作成する。
作成したデータベースとユーザの情報は、後述のデータベースの設定に使用する。

コードのデプロイ

それぞれ threpo プロジェクトのリポジトリより取得し、配置する。
リポジトリ内のディレクトリは以下の通り：

- threpo アプリケーションコード
  - /svn/threpo/eclipse_projects/threpo/threpo/cakeApps/threpo
- threpo アプリケーションドキュメントルート
  - /svn/threpo/eclipse_projects/threpo/threpo/public_html/threpo
- CakePHP ライブラリ
  - /svn/threpo/eclipse_projects/threpo/threpo/cakephp-cakephp-efb6e08

設定項目の編集

以下の設定項目を編集する。

- データベース設定
- 配置情報
- cron shell 内設定

データベースの設定

作成したデータベースの設定を記述する。

編集対象：/var/www/cakeapps/threpo/config/database.php

例：

```
var $default = array(
        'driver' => 'mysql',
        'persistent' => false,
        'host' => 'localhost',
        'login' => 'threpo',
        'password' => '*******',
        'database' => 'threpo',
        'encoding' => 'utf8'
);
```

設定後データベースの初期化を行う。

```
php /var/www/cakeapps/cakephp-cakephp-efb6e08/cake/console/cake.php schema create Threpo –app /var/www/cakeapps/threpo_production
```

配置情報

配置した、threpo アプリケーションコード、threpo アプリケーションドキュメントルート、CakePHP ライブラリの位置関係を記述する。

記述方法：例の太字部分のように記述する。
　例えば、/xxx/yyy の場合、DS . 'xxx' . DS . 'yyy' と記述する。

編集対象：/var/www/html/threpo/index.php
例：

```
/**
 * The full path to the directory which holds "app", WITHOUT a trailing DS.
 *
 */
        if (!defined('ROOT')) {
                define('ROOT', DS . 'var' . DS . 'www' . DS . 'cakeapps');
        }
/**
 * The actual directory name for the "app".
 *
 */
        if (!defined('APP_DIR')) {
                define('APP_DIR', 'threpo');
        }
/**
 * The absolute path to the "cake" directory, WITHOUT a trailing DS.
 *
 */
        if (!defined('CAKE_CORE_INCLUDE_PATH')) {
                define('CAKE_CORE_INCLUDE_PATH', DS . 'var' . DS . 'www' . DS . 'cakeapps' . DS . 'cakephp-cakephp-efb6e08');
        }
```

cron shell 内設定

cron shell 内に、運用サーバのホスト名の指定を行う必要がある。

編集対象：/var/www/cakeapps/threpo/vendors/shells/mail_notification.php

例：

```
class MailNotificationShell extends Shell
{
        var $uses = array('ReportAcceptingFlow' , 'User');
        var $hostname = 'server.sample.jp';
```

管理者ユーザの作成

管理者ユーザがいない状態で、threpo にログインすると以下のように、管理者ユーザ登録画面が表示される。画面の指示に従い、管理者ユーザを作成する。

cron の設定

まず、次の二つのスクリプトファイルの内容を編集する。

編集対象：/var/www/cakeapps/threpo/vendors/shells/cron_waiting_mail_script.sh

例：

```
#!/bin/bash
cake_path=/var/www/cakeapps/cakephp-cakephp-efb6e08/cake/console
app_path=/var/www/cakeapps
app_name=threpo
cd $app_path
$cake_path/cake waiting_mail_notification -app $app_path/$app_name
```

編集対象：/var/www/cakeapps/threpo/vendors/shells/cron_mail_notification_script.sh
例：

```
#!/bin/bash
cake_path=/var/www/cakeapps/cakephp-cakephp-efb6e08/cake/console
app_path=/var/www/cakeapps
app_name=threpo
cd $app_path
$cake_path/cake mail_notification -app $app_path/$app_name
```

編集後それぞれのスクリプトを cron に登録する。
cron_waiting_mail_script.sh は高頻度の繰り返し（1-10 分間隔程度）に設定する。
cron_mail_notification_script.sh は毎日の最初（午前 0 時 1 分等推奨）に設定する。

crontab 設定例：

```
*/1 * * * * /var/www/cakeapps/threpo/vendors/shells/cron_waiting_mail_script.sh
1 0 * * * /var/www/cakeapps/threpo/vendors/shells/cron_mail_notification_script.sh
```

バックアップ

threpo のバックアップは threpo の使用しているデータベース自体をそのままバックアップすることで行うことができる。