目次

[第1章　序論 1](#_Toc378722085)

[1.1 研究の背景 2](#_Toc378722086)

[1.2 研究目的 3](#_Toc378722087)

[1.3研究方法 3](#_Toc378722088)

[1.4プロジェクトマネジメントとの関連性 3](#_Toc378722089)

[第2章 ソフトウェア開発 5](#_Toc378722090)

[2.1 アジャイル型開発のプロセス 6](#_Toc378722091)

[2.2各開発手法のメリット，デメリット 7](#_Toc378722092)

[2.3 アジャイル型開発で必要とされるツール 10](#_Toc378722093)

[2.4 GitHub 11](#_Toc378722094)

[2.4.1 バージョン管理とは 11](#_Toc378722095)

[2.4.2 GitHubの機能概要 15](#_Toc378722096)

[2.5 GitHubを用いた手法 18](#_Toc378722097)

[2.5.1 統計 20](#_Toc378722098)

[第3章 教育について 22](#_Toc378722099)

[3.1 授業形態の変化 23](#_Toc378722100)

[3.2 eラーニング 24](#_Toc378722101)

[3.3 LMSとは 25](#_Toc378722102)

[3.4 LMS活用 26](#_Toc378722103)

[3.4.1 学習ログ 26](#_Toc378722104)

[3.4.2 ポートフォリオ 27](#_Toc378722105)

[3.5 Moodleについて 28](#_Toc378722106)

[3.5.1 Moodleを利用する利点 29](#_Toc378722107)

[3.5.2 Moodleでできること 30](#_Toc378722108)

[3.5.3 Moodleの操作 31](#_Toc378722109)

[3.5.4 Moodleを利用することによって起きる変化 33](#_Toc378722110)

[3.5.5 Moodleのインストール 34](#_Toc378722111)

[第4章 マネジメント 38](#_Toc378722112)

[4.1 コミュニケーション・マネジメントとは 39](#_Toc378722113)

[4.2 プロセス概要 42](#_Toc378722114)

[第5章 開発 45](#_Toc378722115)

[5.1 方法の明確化 46](#_Toc378722116)

[5.1.1 GitHubのAPI 46](#_Toc378722117)

[5.1.2 GitHubのAPIの使い方 48](#_Toc378722118)

[5.1.3データの解析 55](#_Toc378722119)

[5.2 Moodleのカスタマイズ 58](#_Toc378722120)

[5.3運用方法 59](#_Toc378722121)

[5.4修正方法（コード内容） 59](#_Toc378722122)

[5.4.1コード詳細 60](#_Toc378722123)

[5.5動作原理 64](#_Toc378722124)

[第6章 結論 66](#_Toc378722125)

[6.1研究の結果 67](#_Toc378722126)

[6.2問題点 68](#_Toc378722127)

[6.3今後の課題 69](#_Toc378722128)

[6.4 結論 70](#_Toc378722129)

# 

# 第1章　序論

# 1.1 研究の背景

現在，ソフトウェア開発はウォーターフォール型ではなくアジャイル型あるいは反復型の開発手法が注目されている．Forrester Researchが行った調査によれば，ソフトウェア開発プロジェクトの約半数で非ウォーターフォール型開発手法（アジャイル型開発手法，もしくはそれ以外の反復型開発手法）が採用されていることが報告されています[1]．日本でも，楽天やNTTデータをはじめ大手企業がアジャイル型開発の取り組み始めてきており，今後の少しずつ普及していくことが見込まれます[3]．注目されている理由の一つとして開発の手順がことなることである．ウォーターフォール型とアジャイル型の開発の大きな違いは，最初に決めた要求を忠実に行い，顧客に会うのは最初と最後のみであることに対し，提起的に顧客と進捗を確認し，随時要求に応じて変更を行うかである．本来は，最初に要求定義をしっかり固めるため，仕様書通りに行えばよいのですが，実際は顧客のイメージしている成果物と食い違いが発生することが多くあります．上流から準じて行う開発工程の場合だと，その食い違いを訂正することに大きな手戻りになってしまい，予算やスケジュールの立て直しを行うので，時間がかかってしまいます．しかし，アジャイル型で行うと，顧客と定期的にミーティングを行えること，現状のシステムを触ってみられることでフィードバックをもらえるので，食い違いが少なく開発することができる．

アジャイル型開発を行う際には，さまざまな準備が必要であり，そのための知識もなくては行えない．必要となるのは，クラウドサービスとツールの活用ができることが必要となってくる．アジャイルの「機敏な」「俊敏な」という特徴を守るうえで，スピードが大事であり，時間や場所を限られないクラウドサービスは必須である．そのクラウド上で開発を行うときに利用するツールがアジャイル型開発を支えているので，どちらも知っていなければならない．なかでも，もっとも重要なものはバージョン管理ツールである[4]．変更を行うアジャイル型開発には，バージョン管理が必要不可欠なものであり，それを中心として開発が行われている．バージョン管理で常に最新のデータ・過去の変更履歴などを管理することで，バグを早期発見でき，大きな手戻りが発生することも防ぐことができる．このような情勢に対応して，ソフトウェア開発の教育現場でも，バージョン管理システムの導入が求められる．

しかし教育現場にITを導入する試みにおいて，その中心となることが多い学習管理システム（LMS）は，バージョン管理システムを用いる教育には対応していない．現状のLMSは，Web上で提示される教材を学習したり，ネットワーク上の掲示板システムを用いて複数の学習者同士が共同で，問題解決や論議を行ったりする授業形態である．一般的な授業向きであるのでソフトウェア開発に向いてはいない．教師が自ら作成すればよいのだが，サーバを用意する手間があるので教育現場に導入しにくくなってしまうので，LMSと一緒に行うことが望ましい．LMSの利点の１つに学習者の学習状況を細かく把握できるということがあるが，既存のLMSはSubversionやGitのような一般によく使われるバージョン管理システムとは連携していないため，バージョン管理システム上での学習者の学習状況を，LMS上で把握することはできない．そのため，学習者の行動を把握するにはLMSとバージョン管理システムをそれぞれ個別に確認しなければならず，教師への負担が大きくなることが懸念される．

## 1.2 研究目的

本研究では，ソフトウェア開発教育にバージョン管理システムを導入し，それを利用した学習の状況を，LMS上で把握できるようなシステムの開発を行う．LMS上でのログを解析することによって，表面化しにくいソフトウェア開発教育の理解度や，理解しにくいポイントを表面化することができると考える．なおかつ，実際に使われるツールを利用することで，現在のソフトウェア開発についていくことのできる知識を得ることを目的とする．

## 1.3研究方法

　ウォーターフォール型とアジャイル型を調査し，異なる点を洗い出し，必要な知識を集める．実際に現在利用されているバージョン管理ツールであるGitHubを学習者にアカウント作成し，利用してもらう．そこから，学習者のログを解析し活用頻度などを調査．LMSと統合するために，LMSを調査し，GitHubの特定のアカウントの活動を自動的に取得するプログラムを開発する．LMS上で，LMSのアカウントとGitHubのアカウントを統合するデータベースを構築し，学習者のGitHub上での活動を，LMS上で確認できるようにする．

## 1.4プロジェクトマネジメントとの関連性

ソフトウェア開発のプロジェクト・マネジャーには，開発プロセスについての深い理解が不可欠である．本研究は，そのようなプロジェクト・マネジャーを育成するための教育プログラムの提案である．

参考文献

[1] VersionOne, Inc. State of Agile Development Survey Results. 2011. http://www.versionone.com/state\_of\_agile\_development\_survey/11/.

[2] IPA. 非ウォーターフォール型開発の普及要因と適用領域の拡大に関する調査. 2012-06-24. http://sec.ipa.go.jp/reports/20120611/reports\_20120611-2.pdf.

[3] 日経コンピューターReport. 楽天がアジャイル開発を全社へ展開. 2012-11-27. http://itpro.nikkeibp.co.jp/article/COLUMN/20121116/437986/?k3.

[4] 高畠勇人, 渡辺裕, 株式会社テクノロジックアート. アジャイル型開発マネジメント クイックガイド. 株式会社技術評論社, 2013.

# 第2章 ソフトウェア開発

# 2.1 アジャイル型開発のプロセス

　アジャイル型開発は，非常に短いサイクルで開発を何度も繰り返し行う，短期反復開発である．顧客の要求にしたがって，優先度の高い機能から順に，要求・設計・開発・テストを短い期間で繰り返し行いながら，システムの全体を構築していく．原則として，事前に開発全体の詳細な計画は作らずに，一定の短い周期で動作可能なソフトを作り上げることになっている．徐々に作ることで，明確になる顧客ニーズや要件をシステムに反映し，リスクマネジメントの早期低減，顧客側と開発側のギャップを解消することができる．

テスト

開発

設計

要求

ウォーターフォール型

アジャイル型

時間

図1.　プロセスの違い

しかし，細かなアジャイル型プロセスは，決まったものがないので一概には言うことはできない．Scrumやextreme programming（以下 XP），フィーチャ駆動型開発（Feature Driven Development：FDD）などといった手法があり，個別の事情に合わせて選択され利用されている．VersionOne社が行った調査によれば，ソフトウェア開発企業や個人から6,042件の回答の内，Scrumを採用したプロジェクトが全体の52%，ScrumとXPの組み合わせを採用したものが全体の14%を占めていることから，Scrumがアジャイル型開発のベースとなっていることがわかる[1]． Scrumはマネジメントにフォーカスしたものであり，チーム力を最大に発揮することを目指しているものである． 各メンバーで協力し作業を進行していき，必要があれば他のメンバーの力を借りたり助けたりする，プロジェクト管理的な開発手法である．原則として，コミットすること・集中すること・オープンであること・他のメンバーを尊敬すること・勇気を持つことの5つがありコミュニケーションが重視されています．一方の XPはアジャイル型開発の代表的なものであり，エンジニアリングにフォーカスしたものである．包括的なドキュメントよりも，ソースコードを重視する，反復的な開発手法である．メンバーみんなで協力して行うというよりも，高い生産効率や状況変化に対応したものである．原則として，素早いフィードバック・単純さの採用・インクメンタルな変更・変化を取り込む・質の高い作業の5つがある[2]．

図2. 利用している手法

## 2.2各開発手法のメリット，デメリット

注目されている開発ではあるが，ウォーターフォール型がよくないわけではない．いらなくなるわけではない．さまざまな開発手法にはそれぞれ，メリットデメリットがある．開発するものや期間などに応じて，選ぶことが必要である．

現在，もっとも良く使われている ウォーターフォール型開発では最初の段階で要求を明確にすることができ，完了時のゴールが明確である．ゴールが明確になっていることで，開発工程のスケジュールをしっかりと計画することができる．上流工程は，多くの人を必要としないので，下流工程である開発などに人員を増やすことができたり，そのことにより品質向上につなげたりすることが可能である．残工程が明確であることから，納期遅延を回避することもできる．これらのことが，ウォーターフォール型のメリットと言える．しかし，最初の段階で要求を明確にするため顧客側に大きな負担がある．プロジェクトが進んでいくと変更を簡単には行うことができなくなってしまうので，上流工程が一番大事となる．さらに進捗していく中でプロセスの規模が明確になるため，スケジュールがずれる恐れがある．そのため，ウォーターフォール型開発に向いているのは，要求が明確であり，作るべきソフトウェアが固まっている場合に適切だといえる．

試作品を作ってから進めるプロトタイプモデルがある．要求に応じた試作品であるプロトタイプを提示してから，細かいシステムの機能や操作画面などを決めるため，要求のずれが応じにくい．最初にプロトタイプを作成するので，小規模なプロジェクト向きである．

プロトタイプ作成

要件定義

プロトタイプ修正

プロトタイプ評価

システム設計

プログラミング

テスト

検証

運用・保守

図3. プロトタイプモデル

ウォーターフォール型とプロトタイプ合わせたようなもので，スパイラルモデルがある．設計とプロトタイプを繰り返して開発していくものである．一度，要求と決めてからプロトタイプ作成する形なので，プログラムの規模やスケジュールなどの予測をすることができる．プロトタイプなので，仕様書の変更に対応することができる．開発工程はしっかり計画するので，設計工程が伸びて実装に費やせる期間が短くなるということが起きなく進捗管理することができる．スパイラルモデルに向いているのは，大規模なプロジェクトである．

目標の設定

代替手段検討

次のフェーズの計画

リスク分析

代替手段評価

開発と検証

図4. スパイラルモデル

アジャイル型開発は，先ほど説明した通り，反復型の開発手法である．反復型ではすでにスパイラルモデルがあったが，似ているようで少し異なる．アジャイル型は，プロジェクト開始の時点で，ほぼ何も決まってはいない．スケジュール計画など，細かく決められるものがなく，スパイラルモデルよりも柔軟である．そのため，どんな変更にも対応することができ，全貌がみえてなくてもプロジェクトを進めることが可能である．要求が明確でない場合は，プロトタイプで回していくことにより明確にすることができる．しかし，変化に柔軟に対応するため，時間がかかってしまうことがある．そのため，ビジネスニーズが常に変化する新規の新しいシステムなど作成する小規模プロジェクトに向いているといえる．

## 2.3 アジャイル型開発で必要とされるツール

アジャイル型開発プロセスはさまざまなものがあるが，基本的なツールについては変わらない．適切なツールを選び，利用することができるかによって，そのプロジェクトの進捗や生産性に影響がでると考えられます．以下の表が，アジャイル型開発で必要となるツールである[4]．

表1. 主要なツール

|  |  |
| --- | --- |
| ツール | 内容 |
| ソースコード | ソースコードを開発し実行可能なモジュールに変換 |
| コードインスペクション | 開発されたソースコードが，コーディング等に則しているか検査 |
| タスクボード | 未作業・作業中・作業済が一目でわかるボード |
| バーンダウンチャート | 縦軸に残作業，横軸に時間を表したチャート．これにより，どれだけ作業が行ったか，どれだけ作業が残っているかがわかる |
| ビルド，デプロイ | 実行可能なファイルやリソースファイルをひとまとめにパッケージングし，実行環境（検証環境や本番環境）にデプロイ |
| テスト | 開発したモジュールが正しく動作するか確認する作業（テスト）をサポート．特に，繰り返しテストをする場合の作業負荷を軽減 |
| 常時結合 | バージョン管理ツールやビルド，テストツールと連携し，開発したアプリケーションのビルド・テストを常に動かすための仕組みを提供 |
| バージョン管理 | 成果物の変更履歴を管理し，ある時点の状態に復元することが可能．特にソースコードの変更履歴管理に利用され，並列開発にも対応 |
| プロジェクトマネジメント | 進捗・品質・課題の管理を中心としたプロジェクト運営支援．進捗や品質，課題の見える化とマネージャやメンバーとの共有が図りやすいものが望ましい |

## 2.4 GitHub

　SubversionやGitのほかに，バージョン管理ツールでGitHubというものがあります．今一番利用されていて，人気があります．世界中の開発者が利用しており，2012年5月の時点で170万人を超えていて，2013年9月には530万人と1年ちょっとで3倍以上伸びています．GitHubはGitリポジトリ環境をそのまま提供しているだけではなく，エンジニア同士がつながって共同で開発できる「ソーシャルコーディング」を実現していることが大きな特徴になっている．たとえば，別のエンジニアが進めているプロジェクトのソースコードにバグがあり，それを修正してオリジナルのリポジトリに反映したいと考えた場合，forkと呼ばれる機能を使ってソースコードを取り込んで修正し，その内容を反映するようにオリジナルの開発者にリクエストを送ること（Pull Request）ができる．このようにエンジニア同士がソースコードを介してつながり，ソーシャルコーディングを実現できることがGitHubの人気の理由の最大のポイントである．その他にも，オープンソースということやサーバを建てる必要がないということで，誰でもすぐに利用することができる．開発用の周辺機能であるWikiやファイルアップロード・タスク管理の機能が一緒についているなどといったことも理由としてあげられている[5]．

### 2.4.1 バージョン管理とは

バージョン管理は，アジャイル型開発を支えるもっとも重要な技術である．要求が最初の段階で明確になっていなく，変更を伴いながら開発を行うアジャイル型開発には，バージョン管理が必要不可欠なものである．ソースコードやモジュールなど，修正履歴を保持し，任意の時点の内容に復元する仕組みは，どの開発プロジェクトでも必要である．しかし，変更運用ルール等を徹底しにくく，複数人が同一ファイルに修正を行う事故を招きやすく，バージョン管理ツールで変更履歴を管理することが一般的である．変更を一つのファイル，もしくは時間を通じたファイルの集合に記録するシステムなので，後で特定のバージョンを呼び出すことが可能である．ファイルを以前の状態まで戻したり，プロジェクト丸ごとを以前の状態に戻したりできる．過去の変更を見直したり，誰が何を修正したか，いつ何を導入したかを確認したりさまざまなことが行える．バージョン管理システムでドキュメントなどを管理することにより，万が一データを紛失してしまっても，簡単に復元することができる．一般的なものだと，SubversionやGitがよく利用されている．リポジトリといえる，ファイルサーバの使い方によって管理方法が3つあるといえる[6]．

1. 個別バージョン管理システム

　初期のオープンソースソースソフトウェアとしてのバージョン管理システムと言えばRCSが挙げられるだろう．RCSはファイル単位での管理しかできないため，バージョン管理については限定的なことしかできなかった．個人での利用であれば問題ないかもしれないが，複数人が利用するプロジェクトでは何らかの運用上の工夫が必要になることが多くなるため，集中型バージョン管理システムであるCVS等へと発展していくことになった．

リポジトリ



リポジトリ



個人

複数人

図5. 個別バージョン管理システム

1. 集中型バージョン管理システム

　集中型はリポジトリが1つしかなく，毎回そこからアクセスすることから集中型と呼ばれている．変更履歴などは，すべて中心のリポジトリに記録されており，そこから確認することができるシンプルさが集中型のメリットといえる．しかし，サーバであるリポジトリにすべてが管理されているので，なにかサーバで不具合が起きたりするとデータが消えてしまう恐れがある．さらに，リポジトリにアクセスすることができないと利用することができないのでネットワークがない環境では作業を進めることができないことがデメリットといえる．

1. 分散型バージョン管理システム

分散型は，集中型と異なりリポジトリをクローンとして利用することができる．クローンとは，中心のサーバではなく自分の環境にコピーできるということである．分散型はそのクローンされた環境から，変更を行うことが可能である．さらに，クローンでも変更履歴が管理されるので，大本のリポジトリにわざわざアクセスする必要がない．変更してから，自分でリポジトリにあげることができるので自分のペースで更新することができることがメリットである．その反面，自分で行う作業が多く少し複雑なことがデメリットである．

リポジトリ

集中型

分散型

コミット

チェックアウト

図6. 集中型と分散型バージョン管理システム

近年では，オープンソースプロジェクトでGit，Subversionなどバージョン管理システムが採用されている．分散型バージョン管理システムでは，複数のリポジトリを持ち，連携できることから，多く利用されている．

表2. オープンソースプロジェクトで利用されているバージョン管理システム

|  |  |  |  |
| --- | --- | --- | --- |
|  | プロジェクト | 概要 | バージョン管理システム |
| 分散型 | Linux | カーネル | Git |
| Android | OS | Git |
| Perl | 言語 | Git |
| Vim | エディタ | Mercurial |
| Mozilla Firefox | ブラウザ | Mercurial |
| Ruby on Rails | Webアプリケーションフレームワーク | Git |
| X.Org | グラフィカルウィンドウシステム | Git |
| Samba | Windowsとのファイル共有 | Git |
| Emacs | エディタ | Bazaar |
| 集中型 | Chromium | ブラウザ | Subversion |
| FreeBSD | OS | Subversion |
| Ruby | 言語 | Subversion |

### 2.4.2 GitHubの機能概要

　さまざまな機能があり，従来のウォーターフォール型では利用しない機能やこの先に必要となる用語についての説明を行う．

1. リポジトリについて

　リポジトリとは，データやファイルを蓄積する場所である．GitHubの場合，公開リポジトリと非公開リポジトリがあり，有料アカウントと無料アカウントで区別されている．公開リポジトリは，すべてのファイルやソースコードを確認することができるようになっており，GitHubのアカウントを持っていれば変更も可能である．非公開リポジトリでは，非公開リポジトリがあるアカウントに許可してもらうと利用することができる．そのため，学校や企業で利用する際に，プライベートな空間として利用することができるので，使い分けることができる．

1. 共通リポジトリ

　共通リポジトリとは，チームメンバーで共有するリポジトリで，ソースコードのメインバージョンが保存されている．

1. ローカルリポジトリ

　作業者のコンピュータにあるリポジトリである．

1. アドオン（add）

　ソフトウェアに追加される拡張機能のことである．

1. 作業ツリー

　ローカルリポジトリ上にある現在の作業ファイルである．作業ツリーの変更点はaddによるインデックスに追加される．

1. インデックス

　ローカルリポジトリへ反映する変更を一時的に保存しておく場所である．インデックスの内容はコミットによりローカルリポジトリへ反映される．

1. クローン（clone）

　リポジトリをローカルリポジトリに複製することである．クローンしたローカルリポジトリは変更履歴も複製されているので，元のリポジトリとまったく同じように履歴の参照やコミットをすることができる．複製をすることによって，自分の手元で作業を行うことができるようになる．

1. フォーク（forking）

　1つのプロジェクトが複数に分散していくことである．誰かのリポジトリをほかの人がコピーし改変していくことである．

1. ライト（write）

　フォークしたときにコピーしたオリジナルのリポジトリのデータに書き込みをすることである．

(10) コミット（commit）

　コミットとは，ファイルやディレクトリの追加・変更をリポジトリに記録する作業である．コミットを行うことで，誰が変更をいつ行ったのかが記録される．さらに，コミットをする際には，かならずコミットメッセージが求められるので，変更した点や理由などがわかる．

(11) イシュー（Issue）

　名詞であるように，問題という意味をもつ．GitHubであげられているソースコードやファイルについてのバグの報告・機能の追加・方向性の論議などに使うことができる．設定しているメールアドレスに直接届くことで，伝えたい相手にすぐに伝わることができる．自分または相手が発行されたイシューを完了することができたらクローズすることができ，作業があとどれだけあるか把握することができる．さらにイシューを発行する際には，期日も決めることができるので，スケジュールを管理することもできる．

(12) Wiki

　Web上から簡単に内容を書き換えることのできるwebサイト管理システムのことである．HTMLを意識することなく，ブラウザから誰でも変更が可能なWebサイトを作成することができる．複数人で共同の記事や文書を作成するのに用いられる．複数人で利用するため，サポート機能として変更履歴機能と差分機能がついている．変更履歴機能では，記事をいつ誰がどのように変更したかについて確認することができる．また，変更した内容に誤りがあった場合には記事を変更前に戻すこともできる．差分機能は，時間軸における2点間での記事の変更箇所をハイライトで表示することが可能である．

(13) organization

　グループページの作成．個人でもつ公開リポジトリのほかに，会社や団体でorganizationアカウントを作成することができる．もう1つのリポジトリを作成できることで，公開リポジトリの整理，分類がすることができる．さらに，グループページはアカウントや権限の管理を一括して行えたり，有料の場合だと支払いを統合できたりするなど，自分で構築することが可能である．

(14) Pull Request

　Pull Requestは，GitHubのリポジトリにpushした変更や機能追加をほかの人のリポジトリに取り込んでもらうための要求を出す機能である．Pull Requestが送信されると，相手先にPull Requestの内容や含まれているコード変更などを伝えることができ，相手のリポジトリに自分が行った変更を取り込んでもらうことができる．そこから，ソースコードの差分などについてディスカッションすることも可能である．

(15) マージ（merge）

　プルリクエストした人がその人のリポジトリに対して行われた変更を自分のリポジトリにも取り入れることである．

(16) ブランチ

　履歴の流れを分岐して，記録していくものである．複数のメンバーが同時に機能追加を行ったり，バグ修正を行ったりする場合に利用する機能である．メインのブランチから自分の作業専用のブランチを作成し，作業が終わりしだいメインのブランチに自分のブランチの変更を取り組みます．分岐したブランチは他のブランチ，他のメンバーの作業に影響を受けないため，同じリポジトリ中で複数の変更を同時に進めていくことができる．

(17) ウォッチ機能

他の人のデータを見ることができる．他の人の進捗状況やプロジェクト内容を観覧できる．

(18) グループ機能

　特定のアカウントユーザで構成し，特定のユーザ同市でリポジトリを共有したりすることができる．グループ機能を公開状態にしておけば，グループ以外のユーザも観覧することは可能である．しかし，非公開状態にするとグループで決められたユーザ以外は観覧やリポジトリを操作することができない．

## 2.5 GitHubを用いた手法

　GitHubを用いた授業の一例を紹介する．作業は以下の順で行う．

教師の場合

1. GitHubのアカウント作成する
2. 授業で利用するためのプライベートリポジトリを作成する．
3. 学習者にアカウント報告を指示する．
4. プライベートリポジトリへのアクセスを許可する．

GitHub環境

公開

リポジトリ

非公開

リポジトリ

学習者

図7. GitHub教師環境イメージ

学習者の場合

1. GitHubのアカウントを作成する．
2. アカウントを教師に報告する．
3. GitHub for windowsをインストールする．
4. リポジトリクローンを作成する
5. ローカルでファイルを作成する
6. ローカルリポジトリでコミットする
7. ローカル環境からsyncすること
8. プライベートリポジトリにコミットする．

非公開

リポジトリ

学生のファイル

学生のファイル

学生のファイル

図8. 学習者環境イメージ

このような形でGitHubを用いることで学習者に一通り利用させてみることが可能である．GitHubの非公開リポジトリで行うことによって，学習者以外から見ることができないので安心して利用することができる．

### 2.5.1 統計

　学習者の行動を2013年7月7日から2013年9月29日の約2か月半の間，調査を行った．

現在，この効果について調査中である．

　授業でアカウントを作成した学習者は139人であり，9月29日までのGitHubのコミット数は262commitsであった．コミット数だけ確認すると，一見全員がコミットし，GitHubを活用したように考えられる．しかし，授業内での課題であるファイル提出まで行った学習者は95人であったため，7割の学習者が利用したことになる．

表3. 統計

|  |  |
| --- | --- |
| 学習者数（アカウント数） | 139人 |
| GitHubのコミット数 | 262commits |
| GitHubにファイル提出を行った人 | 95人 |

参考文献

[1] VersionOne, Inc. State of Agile Development Survey Results. 2011. http://www.versionone.com/state\_of\_agile\_development\_survey/11/.

[2] 山田正樹. 実践アジャイル ソフトウェア開発法とプロジェクト管理. 日経印刷株式会社, 2005.

[3] システム開発の工程とソフトウェア開発モデル. 2009-10-05. http://gihyo.jp/lifestyle/serial/01/ipa-terminology/0012.

[4] 高畠勇人, 渡辺裕, 株式会社テクノロジックアート. アジャイル型開発マネジメント クイックガイド. 株式会社技術評論社, 2013.

[5] WEB+DB PRESS Vol.69. 2012-07-25.

[6] 岩松信洋, 上川純一, まえだこうへい. Gitによるバージョン管理. 株式会社オーム社, 2011.

# 第3章 教育について

## 3.1 授業形態の変化

現在，塾や学校での授業でインターネットなどを用いたeラーニングと呼ばれる授業形態が行われるようになってきている．パソコン・タブレットの普及率とともにインターネットの利用者が増えていることから学習形態が変化してきているのだ．総務省情報通信政策局が行った通信利用動向調査報告書によれば，図のような結果がわかる．タブレットが普及されるようになってから，パソコンの普及は少々減少したがインターネット自体の利用は減ってはいない[1]．そこで教育分門のIT促進を図るためにeラーニングを利用しようとする企業や大学が増えてきているのである．eラーニング白書よれば，eラーニングを「導入している」企業は54.6%，「導入を検討している」企業は13.5%となっており，そのうち，従業員「5000人以上」では82.8%，「2000～4999人」では62.9%，「1000～1999人」では36.4%となっており，総じて企業規模が大きいほど導入率が高いといえる[2]．このように普及してきている理由の一つとして，学習者にとって低コストであり，迅速に知識を学べるところにある．社会人や多くの知識を得たい人は，大学や専門学校に行くよりもコストがかからずに，働きながらでも専門知識やキャリアを高められるのである．

図9. パソコン・タブレットの普及率

## 3.2 eラーニング

パソコンや携帯端末，コンピュータネットワークなど，情報技術を用いて行う学習方法のことである．eラーニングの「e」はelectronic電子的なという意味をもち，CD・DVDなどの利用，webページでの情報収集においても，eラーニングと呼ぶことができる．学習者がそれぞれパソコンと向き合って学習することで，個々のペースで学習を進めることができる．主に学習教材は，マルチメディアやハイパーリンクを活用した資料や便覧などであるので，復習や予習といった反復学習が容易であることといった利点がある．また，学習者の進捗や習熟度などのデータを容易に集計・管理できるので教育環境の改善に役立ちます．広い範囲で利用することができるので，eラーニングの活用方法や発想によって分類されてきている．場所に限らず，どこでも学習することができることをユビキタスという意味から，uラーニング．モバイル端末を用いた学習形態を行う場合は，mラーニングと分類されることがある．広義ではeラーニングは，すべてのインターネットを用いた学習と定義されてはいるが，狭義の定義で，LMSにマネジメントされたweb上での学習とされている．

## 3.3 LMSとは

LMSとは，学習管理システムのことであり，eラーニングの実施に必要な，学習教材の配信や成績などを統合して管理するシステムのことである．パソコンのオペレーションシステムのように，必要な機能をあらかじめ搭載し，簡単に操作できるものになっている．学習管理システムなので，学習者の成績や進捗管理も行え，指導の指針となるデータとしても管理・取得できるようになっている．大人数の受講者の成果を把握しやすくするために用いられるものであるので，個人向けのeラーニングシステムとは異なり，学校や企業で多く活用されている．さまざまな企業がLMSを作成しているが，オープンソースやフリーのものもあり，種類が豊富である．各国さまざまなLMSがあるが，互換性を保つためにSCORMという規格が策定されている．SCORMは，アメリカのADL（Advanced Distributed Learning）という標準化推進団体により定義・公開されており，eラーニング共通化規格として，事実上の世界標準である．そのため，企業や国が異なるLMSや教材であっても高い互換性を保持できるようになっている．

表4. SCORM対応 LMS一覧

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 開発言語 | 国 | ライセンス | 日本語対応 |
| Moodle | PHP  MySQL | オーストラリア | GPL | ○ |
| ILIAS | PHP  MySQL | ドイツ | GPL | ○ |
| Canvas | Ruby  MySQL | アメリカ | AGPLv3 | × |
| Sakai CLE | Java  MySQL | アメリカ | ECL | × |
| ATutor | PHP  MySQL | カナダ | GPL | × |
| ELECOA | PHP | 日本 | 三条項BSD  ライセンス | ○ |
| SCORM Server4 | Java | 日本 | フリーソフト | ○ |
| Chamilo | PHP  MySQL | ベルギー | GPL | × |
| eFront | PHP  MySQL | アメリカ | CPAL | × |
| Open Elms | ASP  SQL Server | イギリス | GPL | × |
| Ganesha LMS | PHP  MySQL  PostgreSQL | フランス | GPL | × |

## 3.4 LMS活用

### 3.4.1 学習ログ

LMSが導入された環境は，サーバ上で学習者の行動つまりログを管理し，確認することができる．このことから，学習ログの解析を行う研究が近年多くなされている．

LMSを用いて，授業を行うと教師と学習者のコミュニケーションではなく，パソコンと学習者という教育の形になりがちである．そのため，教師は学習者の学習状況を把握することができない．このことを改善するために，多くのLMSでは，テストスコアや教材へのアクセス時刻といった，学習者ごとの学習履歴を学習ログとしてサーバで，記録・保存を行っている．直接，学習者の行動を見ているわけではないため，学習ログの解析や利用が進んでいる．

篭谷の研究の，Moodleを利用した授業展開と利用履歴の解析によれば，毎週の授業と連動し教員がさまざまな教材資料を提示でき，レポートや小テストによって学習者の学習を支援していくシステムとしてオープンソースであるMoodleの導入を開始した．さまざまな講義科目・演習科目で学習支援ツールとして使用した．小テストや記事の投稿など科目に合わせた使いかたを行った．その結果，一定の導入効果が得られるといっている．ログによると授業時間外に資料の観覧，記事の投稿などを多くの学習者が行っている．自宅などからの深夜にアクセスしている学習者もいたことから，LMSの特徴でもある時間・空間に縛られないことがよくわかる．

### 3.4.2 ポートフォリオ

　ポートフォリオとは，書類入れやファイルを意味する言葉である．総合的な学習の評価方法として，近年注目されている外来語である．ポートフォリオ評価は，たとえば「学習活動において生徒が作成した作文，レポート，作品，テスト，活動の様子が分かる写真やＶＴＲなどをファイルに入れて保存する方法」と定義されている．

つまり，学習者が作成した成果物を蓄積した学習記録ファイルです．たとえば，次のようなものを蓄積させます．

・提出し添削を受けた模範課題

・授業に関連して学習者が自分で調べたこと，コメントしたことのすべて

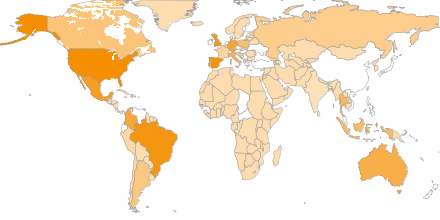
・中間試験の答案，その模範解答

こうしたポートフォリオの利用は，学習者にとっても教師にとっても，メリットがあります．学習者は，自分がコースを通じてどのように進歩してきたかを常に一覧することができます．どれだけの課題を提出したか，あとどれだけの課題が控えているのかなどが自己管理できます．教師にとっては，期末にポートフォリオを提出させることで，それぞれの課題をそのつど採点したり，各学習者の課題の提出ぐあいをそのつど記録にとどめたりすることができます．

LMSを利用することでこれらの，記録もすべてサーバで管理することができるのでわざわざ，ポートフォリオを作成する手間がかかりません．ポートフォリオ評価についても，現在少しずつ注目があり，研究が行われていたりします．学習者と教師にメリットだけではなく，外の人からもメリットがある．その人が，どういう学習をしてきたのかとか，どのような知識があるのかなどがわかるため，就活などにも役立つからである．

## 3.5 Moodleについて

Moodleとは，無償で利用することができるラーニングマネジメントシステムである．元々はプログラマや教育理論屋向けでありModular Object-Oriented Dynamic Learning Environmentの略称である．また，動詞としての語moodleは，ものぐさに徘徊する，思いついたことをするといった意味を持ち，このように楽しみながらの作業が洞察や創造に結びつくことから，いわれている．このことはMoodleが開発された経緯と，学習者・教師がオンライン授業で学ぶ・教えることの両方について言えることである．そして，Moodleを使う人をMoodlerと言う．公式サイトへのMoodler数は80万人以上であり，さらにMoodleサイトが約4万サイトある．そして，ライセンスフリーのため一日あたり約2000回ダウンロードされており，多くの人に利用されている．言語も豊富で80言語に対応していることから，200カ国以上で使用実績がある．しかし色の濃いところが，ユーザが多い国になるのだが，日本はまだ色がついてないので，これから今後普及する可能性が大いにある[5]．

図10. Moodle利用の度合い

MoodleはPHPで開発されているため， PHPの動作するUNIX，Linux，FreeBSD，Windows，Mac OS XなどほとんどのOSで動作することができ，またPHPをサポートしたwebホスティングサービスを行うプロバイダーで使用可能である．さらに，データは単一のデータベースに格納され，主にMySQLとPostgreSQLがサポート対象であるが，ADOdbを使用しているためADOおよびODBC経由で商用データベースも使用することが可能である．最近では，モバイル端末（スマートフォンやタブレット）にも対応してきており，これからさらに利用できる幅が広がることが考えられる．

### 3.5.1 Moodleを利用する利点

　さまざまなLMSがある中でMoodleを利用する利点は次の通りである．

1. 無償である．

　無償であるため，手軽に利用してみることができる．

1. オープンソースである．

　オープンソースライセンスで配布されていることにより，改変・拡張が可能である．各国の法律や運用ルールに合わせて設定を変更したり，文字コードの変換処理を調整したりすることが可能である．

1. コミュニティがある．

　非常に活発なユーザコミュニティがあります．そのため，活用方法や運用方法についての議論が行われたり，問題が起きた時に相談をしたりすることができる．共同開発もできる．責任を持って不具合に対応する機関が存在しないため，コミュニティ大事である．

1. どこでも利用可能．

　インターネット環境があれば，どこからでも利用することができる．パソコンからだけではなく，スマートフォン・タブレットからでも利用可能である．

1. 日本語対応．

　日本語対応しているオープンソースはまだまだ少なく，学習者に利用してもらうのならば，やはり母国語でないと活用が難しくなってくる．

1. PHPで開発がされている．

　PHPで開発が行われていることにより，ほとんどのオペレーションシステムで動作することができる．

### 3.5.2 Moodleでできること

Moodleは，社会的構成主義という学習スタイルを支援するように作られています．この学習スタイルはインタラクティブなものである．社会的構成主義の考え方によれば，人間がもっとも効果的に学ぶためには，教育材料とのやりとりがあり，ほかの人のために新たな教材を作り，その教材についてほかの学習者とやりとりをするのがよいことであると考えます．伝統的な授業のやり方と社会的構成主義に沿ったやり方の違いは，講義を重視するか，ディスカッションを重視するかということである．

　教師からの一方的ではなく，ディスカッションや情報共有を行うことが強みであり，学習者同士のやりとり，学習者と教師とのやりとりが活発に行われる環境を作ることができるのである．その環境作りをしてくれる具体的な機能は次のようなものになる[6]．

表5. Moodle機能一覧

|  |  |
| --- | --- |
| 機能名 | 内容 |
| SCORM | 両規格に準拠して作成されたパッケージの掲載． |
| Wiki | 通常のWiki機能． |
| チャット | 通常のチャット機能． |
| データベース | 任意の情報を収集し，データベース化して格納可能． |
| フォーラム | 教師や学習者あるいは学習者同士の情報交換． |
| レッスン | 説明やテストなどを組み合わせた一連の学習の流れ． |
| 課題 | 学習者に課題を提示する機能．課題の提出にファイルアップロード機能を使うことも可能． |
| ワークショップ | 課題と似ているが，他の学習者も提出物を評価することが可能． |
| 外部ツール | 他のウェブサイト上の学習リソースや活動にアクセスする機能． |
| 小テスト | 理解度をチェックするクイズや簡単なアンケートを作成． |
| 調査 | システムで準備されている質問を使い，学習者に対する調査事項． |
| 投票 | １つのテーマに対して，選択肢から回答となるものを選択． |
| 用語集 | 説明を必要とする用語を登録． |
| 日誌 | 教師と学習者との情報交換． |
| LAMS | 学習活動の手順（シーケンス）を作成．管理・配布する． |
| Hot Potatoes | Hot Potatoesで作成した問題をMoodle上に掲載． |
| エクササイズ | 課題と似ているが，提出前に自己評価が必要． |
| カレンダー | ワークショップ，課題，小テスト，およびイベントの表示． |

### 3.5.3 Moodleの操作

　Moodle操作の基本の流れは図11の通りである．Moodleのさまざまな機能を活用することができるのは，コースのホームページの画面である．コースのホームページ画面には，表5のような機能を活用することができ，各コースごとに設定が可能である．ホームページには，そのコースの最新情報が表示されており，新しい教材の配布や教師からの連絡事項などが確認することができる．

開始

WebブラウザでMoodleの

トップページへアクセス

コース一覧から受講（編集）したいコースを選びコースのHPにとびます

コース上のHP上で学習を行ったり，

編集を行ったりする

コース上での学習（編集）が終了すると，

ログアウトまたは他コースへ移動

終了

認証ページでIDとパスワードを

入力し，ログインします

図11. MoodleのTOP画面

　LMSの特徴である行動履歴は，図12のように表示され管理することができる．行動履歴が見れる画面は教師（管理者）と学習者では異なるが，学習者自身も行動履歴を確認できる．図12は教師側であり管理者画面である．学習者が一覧になっており，そこから，最終アクセスなど学習者の状況を確認することができる．学習者の氏名をクリックすることで，その学習者がMoodle上で受講している教科やログを見ることができる．ログを見ることで学習者がいつアクセスしているのか，なにを確認しているのか把握することができる．メールアドレスを登録していれば，そこから個人へとすぐにメールを送ることができ，学習者に対して細かい教育ができるようになっている．

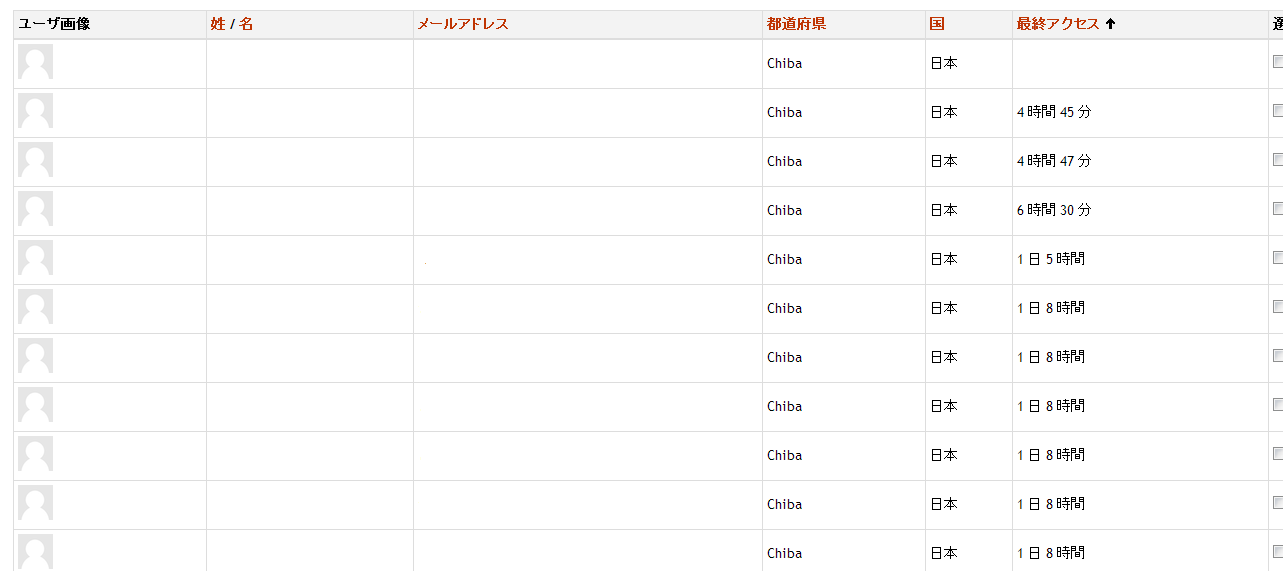


図 12. Moodleの管理者画面の行動履歴

Moodleを利用する際には，リソースと活動の両方をうまく使うことが重要である．リソースとは，成績に反映しない教材であり，閲覧や配布できる，テキストやファイルリンクのことである．また活動とは，成績に反映される教材であり，学習者の作成データが反映されるものである．課題や小テストなどのことである．この2点の両方を使うことで，授業中だけではなくMoodleが活発に利用されるようになる．リソースは，教師側からのインプットのみになるが，HTMLやPDF，wordなど授業には必要なものである．そのため，利用頻度は高いといえる．活動の使い方が，Moodleを上手に使えているかを左右する．機能一覧にあげている機能のほとんどが活動である．これらの機能をうまく利用することで，時間があるときにカレンダーで予定確認や，チャットで会話など活発に利用されるようになる．

### 3.5.4 Moodleを利用することによって起きる変化

　Webを使うことにより，配布資料を印刷する手間と時間が短縮することができる．学習者側では，欠席してしまっても資料がもらうことができ，どちらにとってもメリットである．講義に対する変化も期待できると考えられる．学習者の反応がないことが多く，一方的な講義になりがちなのが現状である．そのため，学習者がどこまで理解してきているのか，ちゃんと講義に参加しているのかがわからないのである．Moodleで教材配布することになれば，講義前に確認をし，学習者自らがダウンロードすることになる．自分でダウンロードするという動作をすることで，講義の内容を確認・把握することができ予習となる．

### 3.5.5 Moodleのインストール

　Linuxの場合

　Ctrl + Alt + T で端末を起動し，以下の手順でインストールを行う．一般的なMoodle利用の場合はsudo apt-get install moodle で環境が作ることができるが今回は開発をする際に変更などができるように一つずつインストールを行います．

1. 環境を作るためにサーバ，データベース，phpをインストールします

sudo apt-get install apache2 mysql-server mysql-client php5

※sudo：ユーザの権限レベルでプログラムを実行するためのコマンドである

※apt-get：パッケージを取得してインストール・アップデートする

途中でMySQLのroot（管理者）のパスワードを聞かれるから,「pass」のような簡単なものを設定しておきます．

1. データベースの準備

mysql –uroot -ppass

Mysqlに入りmoodleを利用するベータベースを作成します．

※-uroot：-u(管理者)

※-ppass：-p(パスワード)

Moodleというデータベース作成し，moodleが直接変更できるように権限を与えます．

create database moodle;

grant all on moodle.\* to moodleuser@localhost identified by ‘passwd’;

exit

※create database：新しいデータベース作成

※grant：管理者がアカウントを作成したり，アカウントに権限を与えたりする

※all on：後ろの構文のデータベース権限を割り当てる

※exit：mysqlを閉じる

データベースができたのでhttp://download.moodle.org/ からMoodleをダウンロードし，保存する．そのディレクトリにApacheが書き込めるようにしておきます．

1. ファイルの展開

※cd：ディレクトリ移動

cd /var/www

sudo tar zxf ~/ダウンロード/moodle-latest-26.tgz

sudo chown www-data:www-data moodle

ダウンロードしたファイルをファイルシステムのwwwの中に展開します．

※tar：ファイルで利用

※zxf：tarのコマンドオプションである

-z：アーカイブをgzipにフィルターする

-x：アーカイブからファイルを取り出す

-f：fileテープの代わりに指定したファイルをアーカイブする

※chown：ファイルやディレクトリの所有者を変更

1. セットアップ

http://localhost/moodle/install.php にアクセスするとセットアップが始まる．言語を選択したら，環境チェックされその際に「cURL PHP拡張モジュールが必要」など表示された場合はsudo apt-cache search curl | grep phpなどとして探し，必要なものをインストールします．

※apt-cache search：パッケージの検索を行う

※grep：文字列を検索する

sudo apt-get install php5-curl

sudo service apache2 restart

さらに，ディレクトリの作成が必要になるので，作成し所有権の変更を行います．

sudo mkdir /var/moodledata

sudo chown www-data:www-data /var/moodledata

※mkdir：ディレクトリを作成する

また他に必要なものをsudo apt-cache search curl | grep phpで確認し，以下の2つをインストールし，実行します．これでMoodleの環境が作成されました．

sudo apt-get install php5-mysqlnd

sudo service apache2 restart

sudo apt-get install php5-gd php5-xmlrpc php5-intl

sudo service apache2 restart

参考文献

[1] 社会実情データ図録. http://www2.ttcn.ne.jp/honkawa/6200.html.

[2] eラーニング白書. 株式会社オーム社, 2012.

[3] 植野真臣. 知識社会におけるeラーニング. 初版, 培風館, 2007.

[4] 野良ウサギのSCORM. SCORM対応フリーウェア、オープンソースLMS一覧. 2012-04-27. http://scorm.jpn.org/id/1317.

[5] Moodle. https://moodle.org/?lang=ja.

[6] William H. Rice IV. Moodleによるeラーニングシステムの構築と運用. 株式会社技術評論社, 2009.

[7] Moodleを利用した授業展開と利用履歴の解析.2005-01-20. http://crf.flib.u-fukui.ac.jp/dspace/bitstream/10461/5038/4/vol.37\_p13-20.pdf.

# 第4章 マネジメント

4. コミュニケーション・マネジメント

　本章では，本研究と関係のあるマネジメントの定義について述べる．

## 4.1 コミュニケーション・マネジメントとは

　プロジェクト情報の生成，収集，配布，保管，検索，最終的な廃棄等を適宜，適切かつ確実に行うために必要なプロセスからなる．プロジェクト・マネジャーは，チーム・メンバー，組織内のあらゆる階層のステークホルダー，組織外のステークホルダー等とのコミュニケーションに大半の時間を費やす．効果的なコミュニケーションは，プロジェクトに関与する広範なステークホルダー間の橋渡しの役割を果たす．効果的なコミュニケーションはまた，多様な文化的背景と組織的背景，レベルの異なる専門技術，プロジェクトの実施や成果についてさまざまな視点と利点等を相互に結び付ける．

　コミュニケーション・マネジメント・プロセスには以下の5つのプロセスがある．

1. ステークホルダー特定

インプット

・プロジェクト憲章

・調達文書

・組織体の環境要因

・組織のプロセス資産

ツールと技法

・ステークホルダー分析

アウトプット

・ステークホルダー登録簿

・ステークホルダー・マネジメント戦略

1. コミュニケーション計画

インプット

・ステークホルダー登録簿

・ステークホルダー・マネジメント戦略

・組織体の環境要因

・組織のプロセス資産

ツールと技法

　・コミュニケーションに対する要求事項の分析

　・コミュニケーション技術

　・コミュニケーション・モデル

　・コミュニケーション手段

アウトプット

　・コミュニケーション・マネジメント計画書

・プロジェクト文章更新版

1. 情報配布

インプット

　・プロジェクトマネジメント計画書

　・実績報告書

　・組織のプロセス資産

ツールと技法

　・コミュニケーション手段

　・情報配布ツール

アウトプット

・組織のプロセス資産更新版

1. ステークホルダーの期待のマネジメント

インプット

　・ステークホルダー登録簿

　・ステークホルダー・マネジメント戦略

　・プロジェクトマネジメント計画書

　・課題ログ

　・変更ログ

　・組織のプロセス資産

ツールと技法

　・コミュニケーション手段

　・人間関係のスキル

　・マネジメント・スキル

アウトプット

　・組織のプロセス資産更新版

　・変更要求

　・プロジェクトマネジメント計画書更新版

・プロジェクト文書更新版

1. 実績報告

インプット

　・プロジェクトマネジメント計画書

　・作業パフォーマンス情報

　・作業パフォーマンス測定結果

　・予算の予測

　・組織のプロセス資産

ツールと技法

　・差異分析

　・予測手法

　・コミュニケーション手段

　・報告システム

アウトプット

　・実績報告書

　・組織のプロセス資産更新版

　・変更要求

　上記のプロセスは相互に関連するだけではなく，他の知識エリアのプロセスとも関連する．各プロセスはどのプロジェクトでも少なくとも1回は現れる．プロジェクトがフェーズに分けられる場合には，1つ以上のプロジェクト・フェーズに現れる．本章では，各プロセスを明確に定義したインターフェースを持つ個々の構成要素として示しているが，実際には，ここでは示していない重なり合いや相互作用もある．

　ほとんごのコミュニケーション・スキルは，一般のマネジメントとプロジェクトマネジメントで共通している．たとえば，以下のようなものがあるが，これらに限定されるものではない．

・積極的かつ効果的に傾聴すること

・確実に理解するためにアイデアや状況を質問し，正しく調べること

・情報を特定し，または確認するために実態を調査すること

・期待を設定し，マネジメントすること

・行動を起こすように，個人または組織を説得すること

・当事者が相互に受け入れできる合意に達するように交渉すること

・深刻な影響が生じないように，コンフリクトを解消すること

・要点をまとめ，整理し直し，次の手順を特定すること

## 4.2 プロセス概要

1. ステークホルダー特定

プロジェクトにより影響を受けるすべての人および，すべての組織を特定し，それらがプロジェクトの成功に対して有する利害，関与，影響に関する適切な情報を文書化する．プロジェクトを成功させるためには，プロジェクトの初期段階においてステークホルダーを特定し，それぞれの関心，期待．重要性，影響のレベルを分析することが不可欠である．ステークホルダーのプラスの影響を最大化し，潜伏的なマイナスの影響を談話させるために，ステークホルダーが関与する度合や時期を判断し，核ステークホルダーと取り組む戦略を制定する．プロジェクト実行中に起こりうる変化に対応するために，定期的に評価を行い，それに対応した戦略をレビューしなければならない．

　ほとんどのプロジェクトには，多数のステークホルダーが存在する．プロジェクト・マネジャーの時間は限られているので，できるだけ効率的に時間を使うために，プロジェクトへの関心，影響，関与の度合いに応じてステークホルダーを分類する．

1. コミュニケーション計画

プロジェクトのステークホルダーが求める情報ニーズを定め，コミュニケーションへの取り組み方を定義する．たとえば，誰が，いつ，どのような情報を必要とするか，またその情報を誰が，どのように提供するかなどのステークホルダーの情報ニーズやコミュニケーション・ニーズに対応する．ステークホルダーの情報ニーズを特定し，それらのニーズを満たすのにふさわしい方法を決定することは，プロジェクトを成功させるための重要な要素である．

不適切なコミュニケーション計画を制定することは，メッセージの伝達を遅らせる，注意が必要な情報を間違った人に伝達する，必要なステークホルダーの一部に意思を伝えないなどの問題につながる．コミュニケーション・マネジメント計画書には以下のような内容を記述する．

* コミュニケーションに関するステークホルダーの要求事項
* 伝達すべき情報．言語，書式，内容，詳細度など．
* その情報を配布する理由
* 必要な情報を配布する時期と頻度
* 情報伝達の責任者
* 機密情報の伝達を認可する責任者
* 情報を受信する人またはグループ
* 情報伝達の手段や，技術
* コミュニケーション活動に割り当てる資源
* エスカレーション・プロセス．下位レベルの要因では解決できない課題をどのくらいの期間で，どのマネジメントの経路を通して，上位者へもち上げていくかを明確にする．
* プロジェクトの進捗，進展に伴って，コミュニケーション・マネジメント計画書を更新し，洗練する方法
* 共通用語集
* プロジェクトにおける情報の流れを表したフローチャート，想定した認可手順のワークフロー，報告書リスト，会議の予定など
* コミュニケーションの制約条件．

1. 情報配布

計画に従って，適切な情報をプロジェクトのステークホルダーに提供する．このプロセスはプロジェクト・ライフサイクルを通して，すべてのマネジメント・プロセスで実行される．このプロセスの焦点は主に実行プロセスの活動であり，コミュニケーション・マネジメント計画書の実施，および予期しない情報の請求への対応することなどがある．

* ステークホルダーへの通知

解消済みの課題，承認済みの変更，プロジェクトの全体的な状況等についての情オフをステークホルダーに提供する．

* プロジェクト報告書

公式や非公式のプロジェクト報告書には，プロジェクトの状況を記述する．

* プロジェクトのプレゼンテーション

プロジェクト・チームは，プロジェクト・ステークホルダーの一部または全員に対して，公式や非公式に情報を提供する．情報と提供の方法は，聞き手のニーズに合致したものにすべきである．

* プロジェクト記録

プロジェクト記録には，コレスポンデンス，メモ，議事録，プロジェクトについて記述したその他の文書等がある．この情報は，実行可能かつ適切な方法で，体系的に維持しなければならない．プロジェクト・チーム・メンバーも神ベースまたは電子媒体を使ってプロジェクト・ノートや登録簿に記録を残す．

* ステークホルダーからのフィードバック

プロジェクトの運営に関するステークホルダーからの情報を配布し，またプロジェクトの将来のパォーマンスを修正したり，改善したりするためにそれを利用する．

* 教訓の文書化

課題の原因，是正処置の選択理由，情報配布についてのその他の教訓を文書化する．プロジェクトの教訓は，当該プロジェクトと母体組織双方の履歴データベースの一部となるように，文書化し，配布する．

1. ステークホルダーの期待のマネジメント

ステークホルダーのニーズに応えるためにステークホルダーと話し合い，協力し．さらに発生した問題への対処を行う．ステークホルダーの期待のマネジメントは，プロジェクト・ステークホルダーの期待に影響を与え，その懸案事項に対処し，課題を解決するために，プロジェクト・ステークホルダーに向けた以下のようなコミュニケーション活動である．

* プロジェクト目標を達成し，その状態を継続したいというステークホルダーの要望に対して調整したり，影響を与えたりすることによりプロジェクトの受け入れの可能性を高めるために，ステークホルダーの期待を積極的にマネジメントする．
* 現在はまだ問題となっていないが，今後問題になると予期される懸案事項に対処する．これらの懸案事項をおおい隠すことなく議論し，リスクを評価する必要がある．
* 特定した課題を明確にし，解決する．解決策の結果として変更要求が出されたり，プロジェクト外で対処したりすることがある．たとえば，解決策を別のプロジェクトやフェーズにもち越したり，他の部署に移管したりすることがある．

1. 実績報告

状況報告，進捗測定，予測等のパォーマンス情報を収集し，配布するプロセスである．プロジェクトの進捗とパォーマンス，さらにプロジェクトの結果の予測の理解と伝達を行うために，実績報告プロセスではベースラインと実績データとの比較を定期的に収集し，分析する．

実績報告書では，受け手ごとに適したレベルの情報を提供する必要がある．形式は，単純な状況報告書から詳細な報告書まで幅がある．単純な報告書では，出来高パーセントや分野別の状況表示（スコープ，スケジュール，コスト，品質）などのパォーマンス情報を表示する．詳細な状況報告書には，以下のものが含まれる．

* 過去のパォーマンスの分析
* リスクと課題の現状
* 報告期間内に完了した作業
* 次の報告期間に完了予定の作業
* 報告期間内に承認された変更の要約
* 差異分析の結果
* プロジェクトの完了時予測値（タイムとコストを含む）
* レビューや検討すべきその他の関連情報

# 第5章 開発

## 5.1 方法の明確化

LMSであるMoodleにGitHubでの行動の表示を目的とする．オープンソースであるMoodleのカスタマイズ方法やGitHubからのデータの取り出し方を探し明確にする．

GitHubから情報を取得するためにはAPIが必要である．GitHubDeveloperにさまざまなAPIが記載されている．その為，どのAPIで必要な情報が得られ，どのような情報が取れるのかを確認する．

### 5.1.1 GitHubのAPI

　APIとはアプリケーション・プログラミング・インタフェース（Application Programming Interface）のことである．OSやアプリケーションソフト，あるいはウェブアプリケーションが自ら持つ機能の一部を外部のアプリケーション（ソフトやウェブサービス）から簡単に利用できるようにするインタフェースである．機能の呼び出し手順や記述方法などを定めた仕様のことである．APIが提供されている機能は独自に開発する必要がないため，プログラムの開発を効率的に行うことが可能となっている．

・APIを公開するメリット

　Web製作者にとっては，他社の膨大なデータベースや機能を無料で利用できるため，Webサイトの開発のコストを大幅に削減できるので，効率的に制作できる．このため，個人や小資本の会社でも，人気のWebサイトを低コストで効率的に作成されることが可能になったのである．

　また，API提供会社にとっては，自社のみでは考え付かないようなWebサービスや自社のみでは実現できないようなWebサービスを外部の人間が作成してくれるため，API利用サイトから自社のサイトへユーザが流入してくるのを期待できる．さらに，自社のデータベースや機能を間接的に自社のサイトを利用してくれるユーザが増加し，それが結果としてデータを集積するスピードが速まり，競争優位に立つことを期待できるのである．

GitHubのAPIは機能ごとに設定されている．今回は，GitHubでの行動を把握する情報が必要である．そのため，eventsについてのAPIを調査した．

表6. EventsのAPI

|  |  |  |
| --- | --- | --- |
| APIの種類 | API | 意味 |
| GET | /users/{sername}/events | ユーザが実行したイベントの一覧を取得する． |
| GET | /networks/{sername}/{repoName}/events | ネットワーク内でユーザが実行したリポジトリの共有イベントの一覧を取得する． |
| GET | /orgs/{org}/events | 組織の共有イベントの一覧を取得する． |
| GET | /repos/{sername}/{repoName}/events | ユーザが実行したリポジトリのイベントを取得する． |
| GET | /events | 共有イベント名を取得する． |
| GET | /users/{sername}/events/orgs/{org} | 組織でユーザが実行したイベントの一覧を取得する． |
| GET | /repos/{sername}/{repoName}/issues/events | リポジトリでユーザが実行したイベントの一覧を取得する． |
| GET | /users/{sername}/received\_events | ユーザが受信したイベントの一覧を取得する． |
| GET | /users/{sername}/events/public | ユーザの共有イベント名の一覧を取得する． |
| GET | /users/{sername}/received\_events/public | ユーザが受信した共有イベントの一覧を取得する． |

### 5.1.2 GitHubのAPIの使い方

　APIを使うことでどんなデータが取れるのか，どのように見ることができるのかMoodleに組み込む前に調査が必要である．そのため，APIを使うことのできる環境を作り実装を行う．（Moodleのインストールの際と同様に，Linuxを利用した場合の方法である．）

1. 準備

sudo apt-get install curl

Ctrl + Alt + Tで端末起動後，curlのインストールを行う．

　cURLとは， URLシンタックスを用いてファイルを送信または受信するコマンドラインツールである．cURLはフリーで使いやすいクライアントサイドURL転送ライブラリのlibcurlを使うため，幅広いインターネットプロトコルをサポートする．HTTPやFTPでのダウンロードやアップロードするためのさまざまな機能を持っている．

1. GitHubのログイン情報を保存する

echo ‘ユーザ名:パスワード’ > GitHub.passwd

chmod 600 GitHub.passwd

GitHubのAPIを操作する上で，毎回ログイン情報を入力するのは手間になる．そこで，ログイン情報をパソコンに保存させる．Echoを利用すること，ログイン情報の出力先を指定することができ，今回の場合は’ユーザ名:パスワード’をGitHub.passwdに出力指定を行っている．Chmodはファイルやディレクトリの所有者を変更するコマンドだが，次に続く3桁の8進数を使用して権限を細かく指定している．

表7. よく利用されるパーミッション指定一覧

|  |  |
| --- | --- |
| コマンド | 意　　味 |
| chmod 644 | 所有者は読み書き可能．所有者以外は読み取りのみ可能． |
| Chmod 600 | 所有者は読み書き可能．所有者以外はファイルに対する操作は一切不可． |
| Chmod 755 | 所有者はディレクトリに対してあらゆる操作が可能．所有者以外は書き込み不可． Cd，lsは可能． |
| Chmod 700 | 所有者はディレクトリに対してあらゆる操作が可能．所有者以外は，ディレクトリに対する操作は一切不可．(ディレクトリ内を見ることはできない) |

③API呼び出し

cat GitHub.passwd

ログイン情報がちゃんと保存されているかGitHub.passwdを表示するようcatで指定し確認します．そこまで完了したら，APIの呼び出しに進むことができます．

　EventsのAPIの表から実際に利用してみると，このAPIはユーザが実行したリポジトリのイベントを取得することができます．

curl -s -u $(cat GitHub.passwd) "https://api.GitHub.com/repos/Diogenesthecynic/FullScreenMario/events"

図13. データのイメージ

④複数ページにわたるデータを取得する方法

　GitHubのAPIは用途に応じてさまざまなものがあるが，ここではデータを取得しファイルに保存することができるプログラムapi.pyと，ファイルからデータを読み出して処理するプログラムjqの2つの部分に分けて操作することにする．こうすることで何回もGitHubにアクセスすることを避けることが可能であり，シェルスクリプトなどを活用することができるからである．

(1)api.py

　データを取得しファイルに保存するプログラム．準備として必要なことはPythonとHTTPアクセスのためのrequestsをインストールすること．作業ディレクトリにapi.pyというファイルを作成することである．

sudo apt-get install python python-setuptools

sudo easy\_install requests

api.py

#!/usr/bin/python

# coding: UTF-8

import sys, json, requests

tmp = open('GitHub.passwd').readline().rstrip('\n').split(':');

username = tmp[0]

password = tmp[1]

#print >> sys.stderr, username,password

url = sys.argv[1]

count = 0

while (url is not None):

#print url

r = requests.get(url, auth=(username, password))

print >> sys.stderr, r.headers['status'],

for item in r.json():

count = count + 1

print json.dumps(item)

if (r.links.has\_key('next')):

url = r.links['next']['url']

else:

url = None

print >> sys.stderr, count

【api.pyの利用】

例1

https://GitHub.com/Diogenesthecynic/FullScreenMario のオープンなissuesをすべて取得する．

今読んでいるファイルのあるディレクトリにあるapi.pyを，作業ディレクトリにダウンロードしておく．

http://developer.GitHub.com/v3/issues/#list-issues で紹介されているGET /repos/:owner/:repo/issuesで試す．issueがたくさんある場合でも，api.pyを使えばすべて取得できる．

python api.py "https://api.GitHub.com/repos/Diogenesthecynic/FullScreenMario/issues?per\_page=100" > openissues.txt

?per\_page=100は無くても動くが，一度に取得するデータを最大にしておくことで，API利用可能回数の消費を抑えたほうがよい．APIの利用回数には制限があり，GitHubAPIは時間あたりの呼び出し回数の上限がある．上限を超えてしまうとしばらく操作ができなくなったりするため，注意する必要がある．

APIから持ってきたデータをopenissues.txt に保存している． APIでデータを取得するのには時間がかかるため，一度にたくさん指示をするよりも，いったんファイルに保存して置くのがよい．つまり，テキストではなくpython api.py URI | ./jq...のように，次の処理に直接渡さないほうがよい．

(2)jq

ファイルからデータを読み出して処理するプログラム．取得したデータは，データが1行に1件JSON形式で書かれている．シェル上つまりOSの外側を貝殻のように包んだ層を経由し，間接的にOSを操作する場合はJSON形式のデータを利用する．そのため JSON形式のデータを扱いたいときには，jqを使うのが便利である．

作業ディレクトリにjqのバイナリをコピーする．

jqを実行できるように，chmodで権限を指定する．chmodは3つの部分からなっており，ReferencesとOperatorとModeからなる文字列で表現される．これで，jqに実行権限を付与することができる．

chmod +x jq

表8. References

|  |  |  |
| --- | --- | --- |
| References | クラス | 説明 |
| U | ユーザ | ファイルの所有者 |
| G | グループ | 所有者が属するグループ |
| O | その他 | グループ以外の全ユーザ |
| A | すべて | 上記3つのすべて． |
| なし | すべて | ugoやaと同じくすべて． |

表9. Operator

|  |  |
| --- | --- |
| Operator | 説明 |
| + | 指定されたモードを指定されたクラスに追加する． |
| - | 指定されたモードを指定されたクラスから削除する． |
| = | 指定されたモードが指定されたクラスの正確な内容となる．  つまり，指定されなかったモードは削除され，指定されたモードだけが付与される． |

表10. Mode

|  |  |  |
| --- | --- | --- |
| Mode | 名称 | 説明 |
| R | リード | ファイルを読み出し可能．  ディレクトリ内容を参照可能． |
| W | ライト | ファイルやディレクトリに書き込み可能． |
| X | 実行 | ファイルを実行可能，  ディレクトリに移動可能． |

【jqの利用】

　jqのプログラムはJSON形式のデータを見やすくしたり，整理をしたりするためのフィルタの役割を果たします．手順としては入力データを受け取り，出力データを生成します．さまざまな形で，生成するためにたくさんのコマンドがあります．必要に応じて，コマンドや順番を変えることによって，データの整形・絞り込みを行うことができます．

jq

フィルタ

入

力

出

力



図14. jqの処理

例1

先ほどのapi.pyの利用で保存されたissuesのデータであるopenissues.txtを用います．現在のままのデータでは，見にくくデータを読み取りにくい．そのためjqを用いて整形を行う．

./jq '.' openissues.txt | less

lessとは，テキスト・ファイルの内容を観覧するコマンドである．観覧中に行える交差が限られているため注意する必要がある．jqを通して，openissues.txtを表示すると整形されて表示されていることがわかる．

表11. 観覧中に行える操作

|  |  |
| --- | --- |
| 操作 | 説明 |
| :e[file] | 新規にファイルを読み出す．  ファイル名を指定しない場合は，現在のファイルを再度読み出す． |
| :n | 次のファイルを観覧する． |
| :p | 前のファイルを観覧する． |
| :x | 先頭のファイルを観覧する． |
| /pattem | 現在位置からファイルの末尾方向へpatternを検索し，移動する． |
| ?pattern | 現在位置からファイルの先頭方向へpatternを検索し，移動する． |
| b, ESC-v | 前の画面へ戻る．  コマンドを入力する前に数字を入力すると，指定した画面数ずつ戻る． |
| d | 半画面進む．  コマンドを入力する前に数字を入力すると，指定した半画面数ずつ進む． |
| e, j | 1行進める．  コマンドを入力する前に数字を入力すると，指定した行数ずつ進む． |
| f, SPACE | 次の画面へ進む．  コマンドを入力する前に数字を入力すると，指定した画面数ずつ進む． |
| g, < | ファイルの先頭へ移動する． |
| G, > | ファイルの末尾に移動する． |
| h, H | lessの操作ヘルプを表示する． |
| n | 検索を再度行う． |
| N | 前回の検索を逆方向に行う． |
| u | 半画面戻る．  コマンドを入力する前に数字を入力すると，指定した反画面ずつ戻る． |
| q, Q | 終了する |
| y, k | 1行戻る．  コマンドを入力する前に数字を入力すると，指定した行数ずつ戻る． |

　さらに整形したまま，表示を変えることが可能である．

./jq '.' -C openissues.txt | less -R

　lessのあとに-Rを足すことによってカラーで表示され，より見やすくすることができる．

### 5.1.3データの解析

　APIでのデータの呼び出し方や整形方法を理解したので，eventsのデータを実際に取得し，データの中身についての解析を行う．

　利用するGitHubAPIはeventsの一覧の中からユーザが実行したイベントの一覧を取得するGET /users/:user/eventsを使います．

python api.py "https://api.GitHub.com/users/Genki966/events" > events.txt

　ユーザの活動履歴を取得する．GitHub DeveloperによればeventsはAPIで取得できるページが30ページで固定されているためトータル300件しか取れない．そのため今回はper\_page=を設定する必要がない．

　テキストを作成することができたら，データを整形し必要なデータを突き止めます．まず，整形カラーにすることで必要なタイトルが何であるか探します．

./jq '.' -C events.txt | less -R



図15. Eventsの構成

ユーザの活動を知りたいので，そのために必要な情報は日時を表示しているcreated\_atと，どのような活動を行っているのかはtypeを見ればわかる．どこで何をしているかの情報はrepoの下位にあるnameでリポジトリ名がわかるので，それらの情報を利用する．

調べた情報をまとめ保存するためにも，さらに整形を行う．

./jq '.created\_at,.repo.name,.type' events.txt

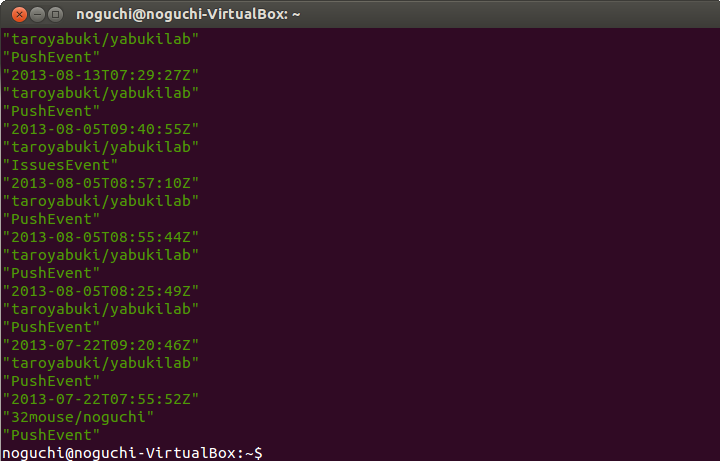


図16. 整形後

　ほしいデータを一覧表示するために，個人の活動の日時とタイプ，リポジトリ名をjqの中に入力します．これをすることにより，必要な情報だけを一覧で見ることができます．

　さらに，この結果を奇数行目だけ改行するように整形を行うと，Excelなどの表計算ソフトで管理がしやすくなる．カスタマイズしたときに，確認が行えるようにcsv形式での保存を行う．

./jq '.created\_at,.repo.name,.type' events.txt | awk '{ printf("%s", $0); if (NR % 3 == 0) printf("\n"); else printf(","); }' > events.csv

　そのために用いられるコマンドはawkである．オークと読み，UNIX上で開発されたプログラム言語である．効率的にテキストファイルを処理することを目的としており，csvやtsv形式のファイルを簡易に処理する目的で広く使われている．基本動作は，テキストを1行読み込んで，awk以降の文を実行しテキストが終わるまで行う動作である．1行とは，個人の活動の日時で1行であるため，その後のリポジトリ名とタイプまで読み込むと3行である．今回の場合，3行目つまり奇数目の時にだけ改行をするように整形を行うと処理がしやすくなる．行数を3で割ったあまりがゼロ，つまり奇数行なら改行，偶数行ならカンマを付ければいい．そうすることで，csv形式にしたときに，表に入れることが可能である．

## 5.2 Moodleのカスタマイズ

　Moodleにはさまざまなレポート機能が備わっている．独自のレポートが必要な時には，そのためのプラグインを作成する．今回のカスタマイズでは，既存のレポートを利用する．既存のレポート画面に，GitHubでの活動履歴が表示されるように修正を行う．

　対象とするレポートは，Moodleユーザのプロフィール画面に表示される「活動レポート」を修正する．修正後のイメージは図17であり，すべてのログのページにGitHubでの活動が表示されている．



図17. Moodleカスタム後

## 5.3運用方法

　Moodleのユーザのプロフィールの編集を行う．プロフィールの任意のところで[IDナンバー]の項目にGitHubのアカウントを入力して保存します．

## 5.4修正方法（コード内容）

　Crtl + Alt + Tで端末を起動しsudo gedit /var/www/moodle/report/log/user.phpを入力して，user.phpを開く．その中でecho $OUTPUT->header();の後に以下のコードを挿入する．

001 //print\_r($user);

002 $GitHub = $user->idnumber;

003 echo <<<EOF

004 <div id="GitHubevents"></div>

005 <script>

006 var url = 'https://api.GitHub.com/users/{$GitHub}/events';

007 var xhr = new XMLHttpRequest();

008 var html = '<table><caption>Activities on GitHub</caption>'

009 + '<tr><th>created\_at</th><th>repo</th><th>type</th></tr>';

010 xhr.open('GET', url);

011 xhr.onreadystatechange = function(){

012 if (xhr.readyState === 4 && xhr.status === 200){

013 var events = eval(xhr.responseText);

014 console.log(events);

015 for (var i = 0; i < events.length; ++i) {

016 var event = events[i];

017 var repo = 'https://GitHub.com/' + event.repo.name;

018 html += '<tr><td>' + event.created\_at + '</td>'

019 + '<td><a href="' + repo + '">' + event.repo.name + '</a></td>'

020 + '<td>' + event.type + '</td></tr>';

021 }

022 html += '</table>';

023 document.getElementById('GitHubevents').innerHTML = html;

024 }

025 };

026 xhr.send(null);

027 </script>

028 EOF;

### 5.4.1コード詳細

　コードについての解説を行う．

001 //print\_r($user);

002 $GitHub = $user->idnumber;

003 echo <<<EOF

004 <div id="GitHubevents"></div>

　001～004は主にこれからプログラムを動かすための準備段階である．ここで行っていることは2つである．Moodleのプロフィール（$user）に入力してあるIDナンバー（idnumber）を（$GitHub）という箱に入れる．データを表示するために（GitHubevents）という場所を用意している．

005 <script>

006 var url = 'https://api.GitHub.com/users/{$GitHub}/events';

007 var xhr = new XMLHttpRequest();

008 var html = '<table><caption>Activities on GitHub</caption>'

009 + '<tr><th>created\_at</th><th>repo</th><th>type</th></tr>';

　005からスクリプトが行われている．006はデータを取ってくるURLであり，GitHubのAPIである．URLにアカウントが挿入されることによって，その人の行動であるイベントにたどり着くことができる．次の行のXMLHttpRequestは，ブラウザ上でサーバーとHTTP通信を行うためのAPIである．このAPIがあることによってデータを取ってくることができる．

　008・009は表として表示させるための，表の見出し（Activities on GitHub）と日時・場所・種類の項目（created\_at・repo・type）である．

010 xhr.open('GET', url);

011 xhr.onreadystatechange = function(){

012 if (xhr.readyState === 4 && xhr.status === 200){

010は先ほどのURLのGitHubへ通信を開始することを意味します．011はコールバック関数であり，HTTP通信状態が変化したタイミングで内部的に呼び出される関数です．これがあることによって通信が完了したのかなどがわかることができます．012がコールバック関数に対してのif文です．つまり，readyStateプロパティの戻り値が4であり，かつ，statusプロパティの戻り値が200である場合という意味になります．

表12. readyStateプロパティの戻り値

|  |  |
| --- | --- |
| 戻り値 | 概　　要 |
| 0 | 初期化されていない（openメソッドがまだ呼び出しされていない） |
| 1 | ロード中（openメソッドは呼び出されたが，sendメソッドはまだ呼び出されていない） |
| 2 | ロード済み（sendメソッドは呼び出されたが，応答は取得していない．ステータス/ヘッダー情報はまだ利用できない） |
| 3 | 一部の応答を取得（応答ステータス/ヘッダーは取得できるが，本題は取得できない） |
| 4 | 全データを取得済み |

表13. HTTPステータスコード

|  |  |
| --- | --- |
| HTTPステータス | 意味 |
| 200 OK | 成功 |
| 401 Unauthorized | 認証を要求 |
| 403 Forbidden | アクセスを拒否 |
| 404 Not Found | リソースが見つからない |
| 500 Intemal Server Error | 内部サーバー・エラー |
| 503 Service Unavailable | サーバーが利用不可 |

013 var events = eval(xhr.responseText);

014 console.log(events);

その場合プログラムの以下の処理を行います．013は，取得したデータがまだ整理されていないので扱いやすい形に変換を行います．テキストの形であるため，（eval）を利用してオブジェクトに変換を行います．次に（console.log）でeventsを選択することによってeventの中身を見ることができます．

‍‍‍‍object, object, object, object, object, object, object, object, object,

元のデータ

►object

►object

►object

►object

►object

オブジェクト

変換したデータ

►object

►events

►created\_at

►repo

►type

変数の中の

データ

図18. データ整理の流れ

015 for (var i = 0; i < events.length; ++i) {

016 var event = events[i];

017 var repo = 'https://GitHub.com/' + event.repo.name;

018 html += '<tr><td>' + event.created\_at + '</td>'

019 + '<td><a href="' + repo + '">' + event.repo.name + '</a></td>'

020 + '<td>' + event.type + '</td></tr>';

021 }

　015からはfor文なので，iが0から始まり1ずつ足しながらデータ数までループします．表示される時にその場所へリンクできるようにrepoにURLを用意して，書き込んでいきます．

022 html += '</table>';

023 document.getElementById('GitHubevents').innerHTML = html;

024 }

025 };

026 xhr.send(null);

027 </script>

028 EOF;

023で一番最初に準備した（GitHubevents）という場所にすべてのプログラムでの処理を入れることで，活動レポートに表示することが可能となっている．

## 5.5動作原理

　Moodleのレポート機能を修正している．レポートを表示しようとすると，その表示画面にGitHubでの行動履歴を表示するためのスクリプトを埋め込む．そのスプリクトにはMoodleのデータベースの「IDナンバー」に登録されたGitHubのアカウントが埋め込まれており，GitHubのAIPで指定されたGitHubのアカウントの行動履歴を読み取り，その結果を解析して，レポートページに表示を行う．

Moodle

GitHub

IDナンバー

アカウント

events

レポート

表示

埋め込み

図19. システム概念図

参考文献

[1] 矢吹太朗. Webアプリケーション構築入門. 第2版, 森北出版株式会社, 2012.

[2] GitHub Developer. http://developer.github.com/libraries/.

[3] jq. http://stedolan.github.io/jq/manual/.

[4] JavaScript プログラミング講座. 2002-2013. http://hakuhin.jp/js.html.

[5] なるべく書かないawkの使い方. 2013-12-09. http://d.hatena.ne.jp/zariganitosh/20131209/minimum\_awk.

# 

# 第6章 結論

## 6.1研究の結果

　図1は，Moodle上で表示されたGitHubの行動履歴である．学習者が，いつ・GitHub上のどのプロジェクトで・何をしたのかが表示されている．教師は，GitHubを利用する学習者の学習状況を，GitHubにアクセスせずに確認することができる．

　さらに，リポジトリをリンクしていることにより，作業したリポジトリへMoodleから直接とぶことができるように

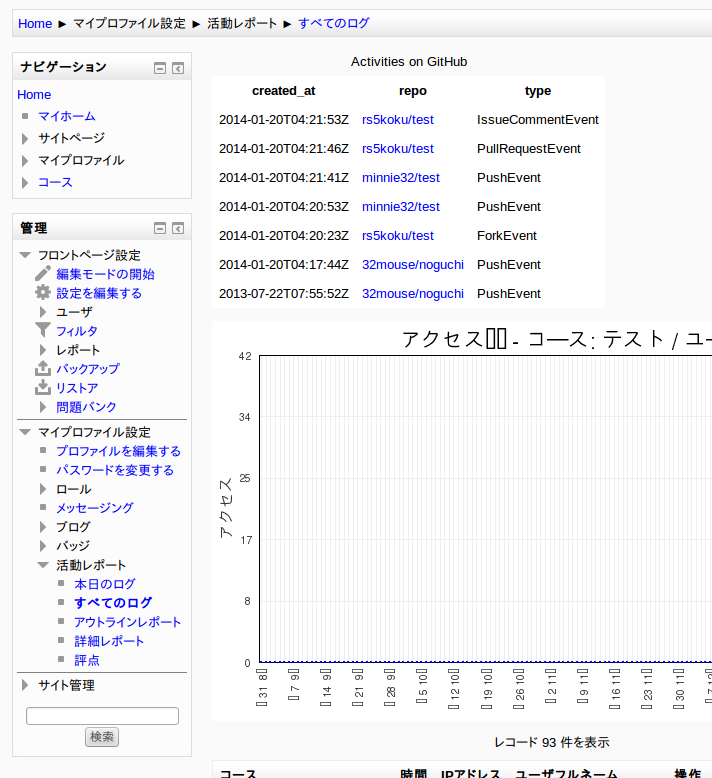
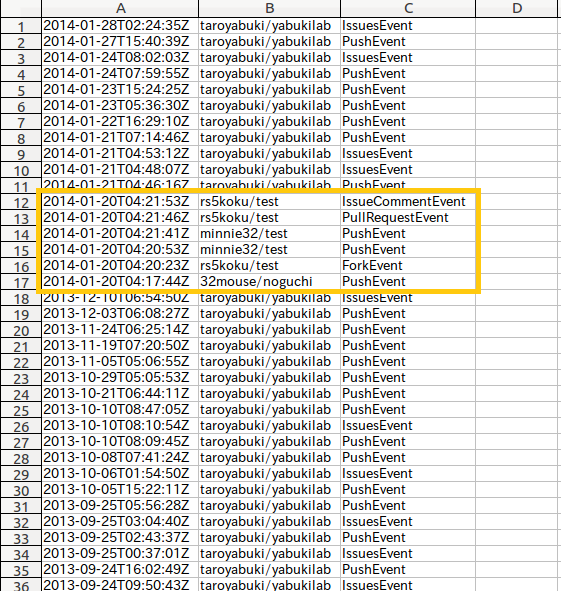


図20. Moodle表示画面

## 6.2問題点

　今回の研究から，次の問題点あがった．現在，Moodle上に表示されるGitHubの行動履歴は，events.csvで取得できたデータの一部（黄色の枠）である．その他の表示されてない行動履歴は，GitHubでの非公開リポジトリでの行動履歴である．現状のシステムでは，非公開リポジトリが表示されないため，公開リポジトリでの利用にかぎる．講義内などでのプライベートで利用したい場合は，さらに改善を行う必要がある．



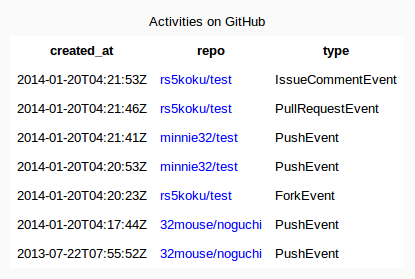


図21. 行動履歴の表示

## 6.3今後の課題

　現状のシステムだと，Moodleの活動レポートにすべてのGitHubでの行動を表示してしまう．そのことにより，授業とは別で利用している活動などももってきてしまうので，関係のない活動を外すことが必要になる．さらに，非公開での活動を今回，もってくることができなかったのでそこの改善も必要であると考える．

Moodle

GitHub

個人の

活動

授業

授業

図22. イメージ図

## 6.4 結論

本研究の目的はオープンソースのLMSであるMoodle上で，学習者のGitHubでの行動履歴を確認できるシステムの開発である．このシステムが動作することにより，eラーニングであるLMSの活用の幅が広がることが考えられる．今回学習者は，LMSでGitHubを導入することで，ソフトウェア開発の現場でよく使われているツールに親しむことができた．このことを応用すると，LMSでさまざまなサービスを導入しても，学習者の学習状況を，教師は詳細に把握できるようになると考えられるであろう．