論文番号 fm2018-03

LTIに準拠したネットワーク自己学習機能の提案と実装

15RD093 菅原 良太, 15RD150 沼田 悠貴

指導: 藤本 衡 准教授

提出日: 2018 年 12 月 25 日

概 要

近年、インターネットの普及が進むにつれ、情報技術者にとってネットワーク技術への理解は必要不可欠なものであると同時に、座学などを用いて知識としてネットワーク技術を学習しても、実際のネットワークと学習したネットワーク技術の知識が繋がりづらい分野である。実際にネットワークを構築し、機器情報などを追加することで、正しいネットワークを形成する演習を行うことが実際のネットワークに関する知識を深めるのに効果的だと考えられる。しかし、教育機関や学習者である個人が、ネットワーク構築の演習に必要な機器をすべて揃え、それらを用いてネットワークの構築を行うのは、あまり現実的ではない。解決策として、e ラーニングを用いた自己学習が挙げられる。

また、多くの企業や教育機関において LMS を用いての e ラーニング学習が行われている。しかし、LMS が行うのは学習の管理である。より高度な学習を LMS 上で行うには、学習したい内容に合わせた学習支援ツールを LMS に導入しなければならない。

そこで、これらの問題を解決するため、本研究では、LTI に準拠した学習支援ツールとして、ネットワーク自己学習機能を保持した Web アプリケーションの実装を提案する。学習支援ツールは独立した Web アプリケーションとして機能しているので、様々な LMS から呼び出すことが可能である。本研究では LTI に準拠した LMS として Canvas と Moodle をそれぞれ導入し、LTI に準拠した学習支援ツールとしてネットワーク自己学習機能を持った Web アプリケーションを導入した。

目 次

1 はじめに

近年、インターネットの普及が進むにつれ、情報技術者にとってネットワーク技術への理解は必要不可欠なものであると同時に、座学などを用いて知識としてネットワーク技術を学習しても、実際のネットワークと学習したネットワーク技術の知識が繋がりづらい分野である。実際にネットワークを構築し、機器情報などを追加することで、自らの手で正しいネットワークを形成する演習を行うことが実際のネットワークとそれに付随する知識を深めるのに効果的だと考えられる。しかし、教育機関や学習者である個人が、ネットワーク構築の演習に必要な機器をすべて揃え、それらを用いてネットワークの構築を行うのは、あまり現実的ではない。解決策として、e ラーニングを用いた自己学習が挙げられる。プログラミング学習において、e ラーニングを用いた自己学習を行うことができる Web サイトが近年普及しつつある。情報技術者にとってネットワーク技術への理解がプログラミングの知識同様、必要不可欠なものになっている現在、ネットワーク技術においても e ラーニングを用いた自己学習の機会を提供することが求められている。

また、多くの企業や教育機関において LMS(Learning Management System) を用いての e ラーニング学習が行われている。しかし、LMS が行うのは学習の管理であり、教材や資料の配布、簡単なテストや課題の実施、それに対する評価を行うのが主な機能である。より高度な学習を LMS 上で行うには、学習したい内容に合わせた学習支援ツールを LMS に導入しなければならない。

ネットワーク自己学習機能の先行研究として、魚本ら [1] は、特定の LMS のプラグインとしてネットワーク自己学習機能を導入した。これは、特定の LMS 上で

の動作を想定して設計されており、同一の LMS 上でしか動作できず、また、導入した LMS に強く依存しているため、LMS 側に変更があった場合、それに合わせてプラグイン側も変更しなければならず、プラグインとして動作するための細かな設定を行わなければならない。

北澤ら [2] は、仮想 Linux である User Mode Linux を利用してサーバ上に複数の仮想マシンを生成し、それらを仮想ネットワーク機器として扱うことでネットワークシミュレートを行うシステムを開発した。このシステムは Web アプリケーションとして動作し、学習者はアプリケーションを操作することによって仮想ネットワークを構築する。これは、独立したネットワーク自己学習機能として動作しているため、LMS との連携を行うことができず、採点機能などを利用しようとした場合、LMS 側に手動で行わなければならない。

そこで、これらの問題を解決するため、本研究では、LTI(Learning Tools Interoperability) に準拠した学習支援ツールとして、ネットワーク自己学習機能を保持した Web アプリケーションの実装を提案する。LTI に準拠した学習支援ツールであれば、LTI に準拠した LMS から呼び出すことができる。これによって逐一インストールする必要がなく、学習支援ツールは独立した Web アプリケーションとして機能しているので LTI に準拠した LMS ならば、様々な LMS から呼び出すことが可能である。本研究では異なる仮想マシン上に LTI に準拠した LMS として Canvas と Moodle をそれぞれ導入し、LTI に準拠した学習支援ツールとしてネットワーク自己学習機能を持った Web アプリケーションを導入した。異なる LMS である Canvas と Moodle から LTI に準拠した学習支援ツールであるネットワーク自己学習機能を同じように使用し、Web アプリケーション側での動作に応じた得

点を LMS 側に反映することで LTI に準拠したネットワーク自己学習機能の実装とした。

以下、２節では、本研究で利用した LTI について説明する。３節では、本研究で提案したネットワーク自己学習システムに付いて説明する。４節では、実際に LTI を用いての実装実験について説明する。

また、本研究において、LMS 側とネットワーク自己学習機能においての LTI に関する部分を菅原が、ネットワーク自己学習機能のシステムに関する部分を沼田が担当した。

2 LTI

LTI(Learning Tools IterOperability) とは、異なるプラットフォーム間における学習支援ツールの相互運用を可能にするための規格 [3] であり、ツール間の通信プロトコルは HTTP 上でのメッセージ交換として実装されている。

LTI に準拠することの具体的なイメージとして、次のケースを想定することができる

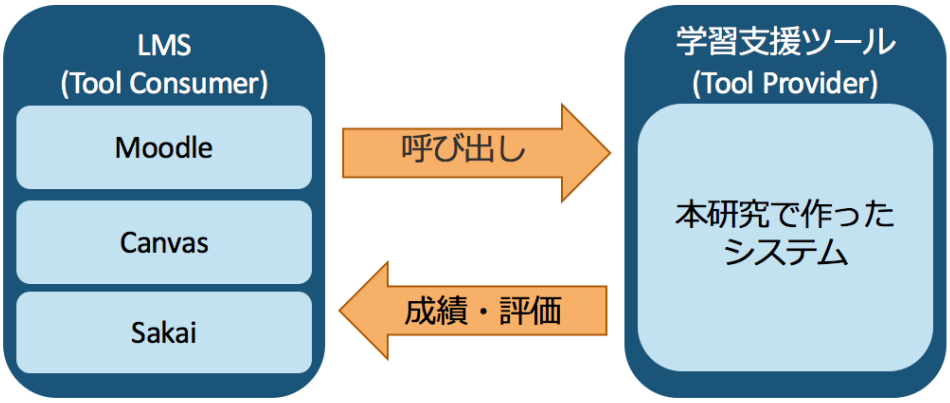

図 1: LTI 具体的なイメージ

LTI では、ユーザ情報や課題の進行状況を直接管理する LMS をツールコンシューマと呼ぶ。標準機能として LTI に対応している LMS として、Canvas, Moodle, Sakai, Blackboard などがある。一方、個別の具体的な問題や教材を提供するプラットフォームをツールプロバイダと呼ぶ。LTI に準拠したツールプロバイダを実装すれば、LTI に対応した複数の LMS からツールを実行することが可能となる。これはツールをプラグインとして実装するのに比べてソフトウェア開発効率の面で極めて有利である。

ユーザーとツールコンシューマ間では、ユーザー ID と PW を用いて認証を行う。しかし、ユーザーがツールコンシューマでツールプロバイダを使用する際は、ID と PW を再度入力せずに使用することができ、ユーザーからは LMS の機能の一部かのようにして見える。これが LTI の利点であり、この認証を省くために OAuth と呼ばれるプロトコルが使われている。

2.1 OAuth

OAuth(オーオース) とは、SNS や Web サービス間で「アクセス権限の認可」を行うためのプロトコルである。また、OAuth には 1.0 と 2.0 が存在しているが、本研究では LTI1.0 の実装にあたり OAuth1.0 を使用している。

2.2 LTI1.0 における OAuth1.0

OAuth1.0 実装では、第三者による不正なログインを防ぐための OAuth signature(署名) 及び key(暗号) の作成をする必要があった。しかし、現在 OAuth2.0 が主流の中、LTI1.0 では OAuth1.0 が採用されているため、署名および暗号を作成する関数を Ruby で自作した。

署名及び暗号の作成手順を以下に示す。

1.「キー」を作成

2.「文字列」の作成

3.「キー」と「文字列」用いて署名を作成

2.2.1 キーの作成

「oauth_consumer_secret」、「oauth_token_secret」を RFC 3986 に基づき URL エンコードし、&で繋げれば完成。

本研究では「oauth_consumer_secret」を設定し、「oauth_token_secret」は存在させなかった。また、各々を RFC 3986 に基づき URL エンコードし、「oauth token secret」を空白とし、&のみを繋げて Key を作成した。

2.2.2 文字列の作成

1. パラメータをアルファベット順に並べ、キー=値... の形で並べた上で,RFC 3986 に基づき URL エンコードする。

2. リクエストメソッド、リクエスト URL を RFC 3986 に基づき URL エンコードする。

3. リクエストメソッド、リクエスト URL、パラメータの順で&で繋げることで文字列を作成した。

2.2.3 署名の作成

1.LTI1.0 では HMAC-SHA1 方式を採用しているため、作成した「キー」と「署名」を用いて HMAC-SHA1 方式でハッシュ値を生成する。この時バイナリデータでハッシュ値を生成する必要がある。

2. 生成したハッシュ値を、base64 エンコードすることで作成。この手順で出てきた数値が署名となる。

2.3 成績反映

成績反映の手順を以下に示す。

ツールコンシューマから成績を返すパラメータ「lis_outcome_service_url」を設定し、特定のユーザーを一意的に示す、「SourcedId」をパラメータ「lis_result_sourcedid」から取得し、XML 内の「SourcedId」を書き変え、ツールプロバイダでまとめた点数を XML 内の「textString」に加えた上で送信する。

3　システム概要

3.1　システム

本研究では、プラグインとして LMS 上に新しい機能を提供するのではなく、LTI に準拠した Web アプリケーションを用いて、異なる LMS で同様の機能が提供でき、Web アプリケーション側での操作に対し LMS 側に特定の点数を返すことを目的とした。そこで、異なる仮想マシン上にそれぞれ LMS である Canvas、Moodle と、独立した Web アプリケーションとしてネットワーク自己学習機能を導入した。また、ネットワーク自己学習機能は Ruby on Rails を用いて実装した。これらは図 2 で表しているようにネットワーク自己学習機能は実際には独立した Web アプリケーションであるが、あたかも LMS 側にプラグインとして導入されているように機能を提供する。

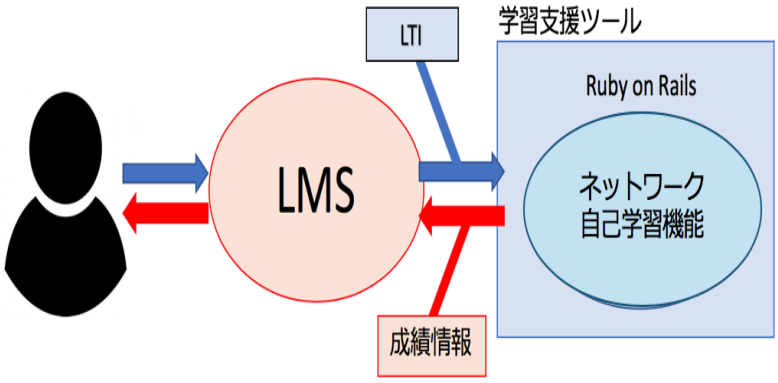

図 2: 本研究で提案するシステムの構成

また、Web アプリケーション側は LMS に呼び出された際、独立した Web アプリケーションとしてネットワーク自己学習機能を提供する。この機能にはネットワークを自由に構成し、機器情報を設定することのできる自由描画モードと、予め問題として構成されたネットワークに正しい機器情報を追加することで正しいネットワークの作成を目指す問題演習モードが有る。問題演習モードでの正誤によって得られた得点を LMS 側に返すことで LMS での学習者の評価を行う。

魚本ら [1] の制作したネットワーク自己学習機能は Moodle の独自プラグインとしてネットワーク自己学習機能を実装している。クライアントサイドである独自プラグインとしてのシミュレータ部分は HTML と JavaScript で、Moodle のプラグインとしての設定の部分は PHP で、シミュレータで作成されたネットワークの構成の正誤の判定プログラムは Ruby でそれぞれ記述されている。これは、様々なシステムを使用しているため、複数のシステム間でデータのなどの連携を行わなければならず、安定性にかけていた。

そこで、本研究ではすべてのシステムを Ruby on Rails の中で実装した。MVC アーキテクチャに基づいて設計することにより、魚本 [1] らのシステムをすべて Ruby on Rails 内で実現した。これにより、複数のシステム間でのデータの送受信などを行う必要性がなくなり、システムとしての安定性を実現した。

3.2 Ruby on Rails

本研究で提案したネットワーク自己学習機能は、Ruby on Rails を用いて実装されている。Ruby on Rails とは、Ruby で構築された、Web アプリケーションを開発するためのフレームワークである。特徴として MVC アーキテクチャの採用や

設定より規約という設計哲学などが挙げられる。

MVC とは「Model」,「View」,「Controller」の頭文字であり、MVC アーキテクチャとはアプリケーションの構成が以下の図 3 のように分類することに由来している。

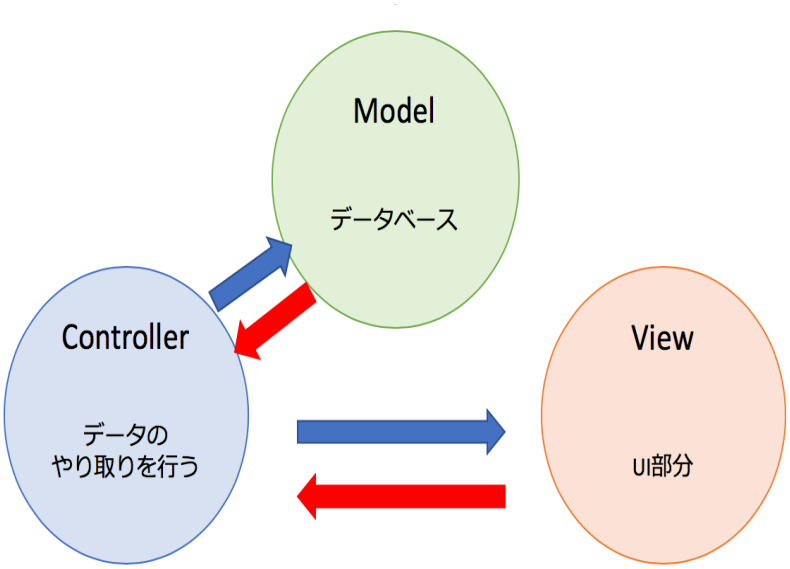

図 3: MVC アーキテクチャ

ここで「Model」とはデータベースに収めたデータやそのデータのルールなどを表す。「View」は、アプリケーションの UI の部分を指す。HTML や CSS を用いて配置やデザインを決定する。「Controller」は View と Model の間を取り持つ部分である。Model と View の間でデータの受け渡しなどを行う。

Ruby on Rails ではこれら「Model」,「View」,「Controller」が機能として独立しているため、それぞれの部分の開発を効率良く行うことができ、仕様変更や新たな機能の追加が容易に行える。また、「View」として UI 部分が独立している