一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
LOIS2024-72(2025-03)

Moodle 用動画配信システムにおける 視聴履歴の可視化と動画間の視聴状況の共有

渡邉 康太† 藤本 茂雄†† 檜垣 泰彦†††

†千葉大学大学院融合理工学府 〒263-8522 千葉市稲毛区弥生町 1-33
††千葉大学国際未来教育基幹 〒263-8522 千葉市稲毛区弥生町 1-33
†††千葉大学アカデミック・リンク・センター 〒263-8522 千葉市稲毛区弥生町 1-33

E-mail: †††higaki.yasuhiko@faculty.chiba-u.jp

あらまし 千葉大学では，SCORM を利用して Moodle と連携した動画配信システムを構築している．しかし現行のシステムでは，学生の視聴行動を詳細に分析することや動画の多言語対応時に各言語間で視聴履歴を共有することが難しい．これらの課題の解決のために，視聴行動の可視化機能と動画間における視聴状況の共有機能を開発した．視聴行動の可視化では，受講生の動画視聴中の操作を記録し，動画の再生位置と経過時間を軸とした折れ線グラフを用いて可視化した．また，動画間の視聴状況の共有では，同一の SCORM パッケージ内に複数の講義動画の情報を統合し，講義動画間で視聴状況が共有されるようにした．これらの対応により，教員は学生の視聴状況をより詳細に把握することが可能となった．

キーワード Moodle，動画配信システム，視聴ログ，可視化，多言語化，SCORM

Visualization of Viewing Logs and Sharing Viewing Status between Videos in a Moodle-based Video Distribution System

Kota WATANABE† Shigeo FUJIMOTO†† and Yasuhiko HIGAKI†††

†Graduate School of Science and Engineering, Chiba University 1-33 Yayoi-cho, Inage-ku, Chiba-shi, 263-8522 Japan
††Institute for Excellence in Educational Innovation, Chiba University 1-33 Yayoi-cho, Inage-ku, Chiba-shi, 263-8522 Japan
†††Chiba University Academic Link Center 1-33 Yayoi-cho, Inage-ku, Chiba-shi, 263-8522 Japan

E-mail: †††higaki.yasuhiko@faculty.chiba-u.jp

Abstract At Chiba University, a Moodle-based video distribution system using SCORM has developed for the purpose of media instruction. However, the current system makes it difficult to analyze students' viewing behaviors in detail and sharing viewing status between multilingual video contents. To address these issues, we have developed functions for visualization of students' viewing behaviors and sharing viewing status between lecture videos. In tracking students' viewing history, students' viewing behaviors while viewing videos were recorded and visualizes them using line graphs with the video playback position and elapsed time as axes. Furthermore, the sharing viewing status function integrated information on multiple lecture videos within the same SCORM package, so that viewing status was shared between lecture videos. It has confirmed that instructors can obtain more detailed knowledges of students' viewing behaviors.

Keywords Moodle, Video Delivery Systems, Viewing Logs, Visualization, Multilingual Support, SCORM

1. はじめに

千葉大学では 2020 年度からの「全員留学」プログラムに備え，Moodle 上でのメディア授業を目的として動画配信システムを開発した [1]．本動画配信システムは，AES 暗号化を施したマルチビットレートの HTTP Live Streaming（HLS）を採用した動画配信を実現している．また，Moodle との連携には，e ラーニング教材向けに設計された標準規格である SCORM を利用している．動画配信サーバーは，クライアントからのリクエストに応じてプレイリストファイルやセグメントファイルを提供しており，この構成によってプレイリストの URL を指定するだけで HLS 形式の動画をネイティブに再生可能となる．

動画教材を SCORM として実装することで，SCORM

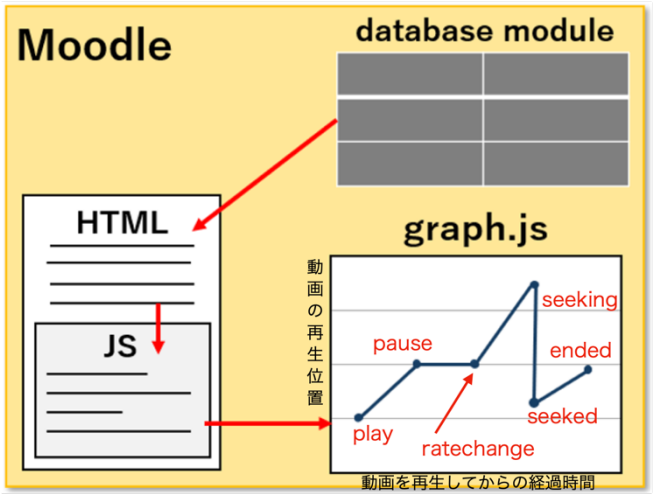

図 1. Moodle のデータベースモジュールに保存された視聴ログから可視化グラフを作成する流れの概要.

の LMS API 機能を利用して，教材側から Moodle へ視聴情報（スコア＝動画の何%を視聴済みか）を渡すことができるが，ここには「どこまでを視聴したか」が記録されており，「どのように視聴したか」を把握することはできない．そこで，学生の講義動画視聴中の動画プレイヤーに対する操作を記録してグラフとして可視化し（可視化グラフ），教師が学生の視聴行動を視覚的に把握できる機能（視聴行動の可視化機能）の追加を行った[2]．動画視聴時の視聴ログは，4 つの情報を 1 セットとして Moodle のデータベースモジュールに保存している．保存されているデータはそれぞれ「Moodle のアカウント」「当該動画にアクセスした時刻」「動画の再生位置」「動画を再生してからの経過時間」である．また，これらの情報を取得するタイミングは，受講者の操作に関係付けられており，「play」「pause」「seeking」「seeked」「ended」「ratechange」のイベントの発生時に視聴時のログとして保存される．さらに，これらの保存された視聴ログに対して，データベースモジュールの標準機能であるテンプレートを活用し，「動画の再生位置」と「動画を再生してからの経過時間」を軸とする折れ線グラフを作成することで視聴時の行動を可視化した（図 1）．

なお，データベースモジュールに保存された視聴ログを分析したところ，一部の視聴ログが操作イベントの発生時刻とは異なる順序で保存されている場合があることが確認された．この現象は，学生がシークバーを動かした際に操作イベントが極めて短い時間の間に複数回発生し，処理順序が前後することに起因すると考えられる．そのため，可視化グラフを作成する際には，保存された視聴ログを操作イベントが発生した順序に並べ替える処理を行っている（図 2）．

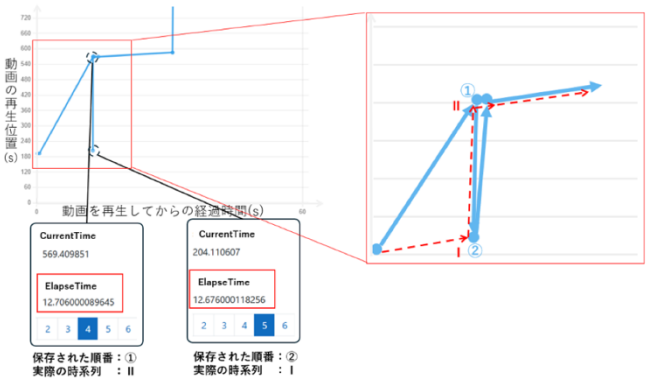

図 2. 視聴ログの並び替えによる可視化グラフの修正（青実線から赤破線）.

表 1. 検証から得られたデータ

	動画 A	動画 B
視聴人数（人）	23	29
動画の長さ(秒)	1, 019	732
実習の有無	無	有
総視聴時間(秒)	10, 896	63, 984
視聴ログの保存件数(件)	330	1, 254

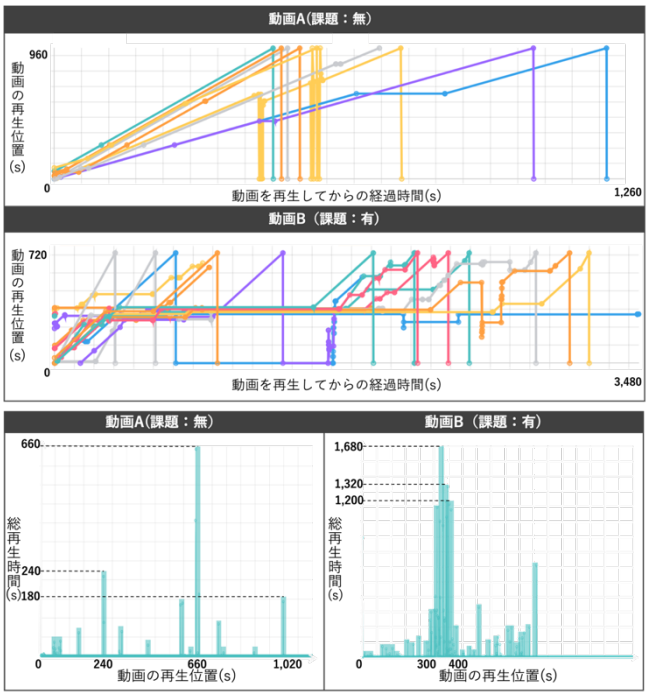

図 3. 検証から得られた可視化グラフ.

2. 視聴行動の可視化機能の検証

オンデマンド授業として提供する 2 つの講義動画に対して，本システムを適用し，動作の検証を行った（表 1）．また，検証結果から得られた可視化グラフを図 3 に示す．

図 3 より，実習が伴わない動画 A では，再生中に動画を一時停止する傾向がほとんど見られず，多くの学生が再生速度を 1.5 倍から 2 倍に設定して視聴していることが確認された．一方で，視聴後に講義内容と関連した HTML ファイルの作成が実習として課される動画 B では，動画の 300～400 秒（停止位置①）および 500～600 秒（停止位置②）の部分で，一時停止の多数発生が見られる．これは，それぞれの停止位置において，実習内容の説明が行われていることが要因と考えられる．また，各停止位置の停止時間を比較した結果，停止位置①の停止時間が長いことが確認された．この要因として，停止位置①では実習内容に関する技術的な説明が行われており，学生が理解を深めるためにより多くの時間を要したことが考えられる．一方，停止位置②では実習に関する確認事項が述べられており，技術的な解説と比較して短時間での理解が可能であったことが，この差を与えたと推察される．

以上の検証の結果，千葉大学の Moodle 運用環境下において，本システムは動画プレイヤーによる学生の視聴行動の検出，視聴ログのデータベースモジュールへの保存，さらに視聴ログの可視化について正常に動作することが確認された．

3. 視聴行動の可視化機能の改善

以上の動作検証を通じて，「データベースモジュールにおける一覧表示数の不足」と「視聴ログの重複」の 2 つの課題も確認されており，下記ではこれらの改善について説明する．

1 つ目の「データベースモジュールにおける一覧表示数の不足」では，多数の学生による動画視聴や同一ユーザによる複数回の動画視聴，さらに想定以上の操作イベントの発生により，データベースモジュールの一覧表示数を優に超える数の視聴ログが保存された．これにより，検証段階ではすべての視聴ログを可視化グラフに反映することが不可能であったため，グラフのスケーリングや動画プレイヤーによる視聴ログのフィルタリング機能を実装するなど，データ許容量の増加や，効率的に視聴ログを保存する工夫が求められる．

データベースモジュールの初期設定では，最大で 1,000 件までの保存データを一覧表示することができるが，受講者による様々な操作イベントによって 1,000 件を超える視聴ログがデータベースに保存されることが想定される．本研究で用いた HTML から取得する視聴ログにより可視化グラフを作成する手法では，1,000 件を超えて保存された視聴ログも含めすべての視聴ログを HTML として表示する必要がある．先の報告[2]では「moodle/mod/data/lib.php」ファイル内に記述されている一覧表示するデータ数の設定を変更することで，

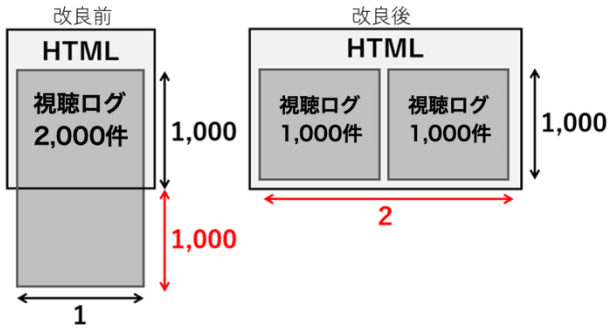

図 4. 改良前後における視聴ログの HTML ページへの表示方法の変化．

一覧表示するデータの件数を増やしていた．この方法は Moodle 本体に変更を加えるものであったため，Moodle 本体に変更を加えない方法として，データベースモジュールの 1 つのエントリに対して N 件の視聴ログを保存できるように，1 つのエントリに対して $4N$ 倍のフィールドを用意する方法を提案する．

また 2 つ目の「視聴ログの重複」では，保存された視聴ログに重複したデータが確認された．これは，動画プレイヤーが同一のアクションに対して，不要に繰り返し同じ視聴ログを保存していることに起因すると考えられるため，動画プレイヤーに重複するデータの保存を防ぐため仕組みを実装する．

3.1. データベースモジュールへの対応

1 つ目の課題である「データベースモジュールにおける一覧表示数の不足」に対しては，データベースモジュールのフィールド数を拡張することで対応する．フィールド数を拡張することで，1 つのエントリに格納できる視聴ログが増え，結果としてデータベース全体のエントリ数を削減できる．具体的には，フィールド数を 1 つ追加すると，1 つのエントリに保存できる視聴ログが 1 件増加する．そのため，図 4 に示すように改良前は 1,000 件までしか HTML に表示できなかった視聴ログが，改良後には 2,000 件まで表示が可能となる．この手法により，検証時に比べて少ないエントリ数で同量の視聴ログを保存・可視化することが可能となり，システムの柔軟性が大幅に向上する．

また，フィールド数には上限がないため，将来的なデータの拡張にも柔軟に対応できる．視聴ログを HTML として表示する際のページ読み込み時間は，フィールド数ではなく，保存されている視聴ログの総数に依存する．そのため，フィールド数を増やしてもページの表示速度には直接的な影響を与えない．一方で，エントリ数の削減によってデータベースの検索効率が向上し，結果的に読み込み時間の短縮が期待できる．

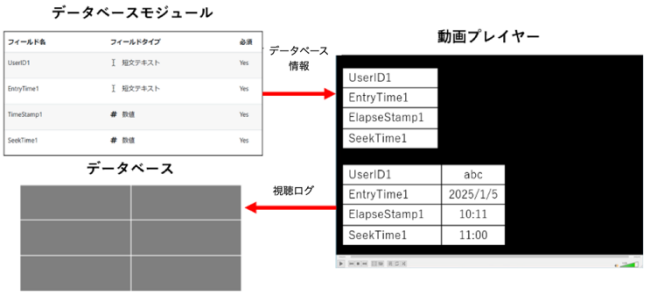

図 5. API を利用した動画プレーヤーとデータベースモジュールの関係.

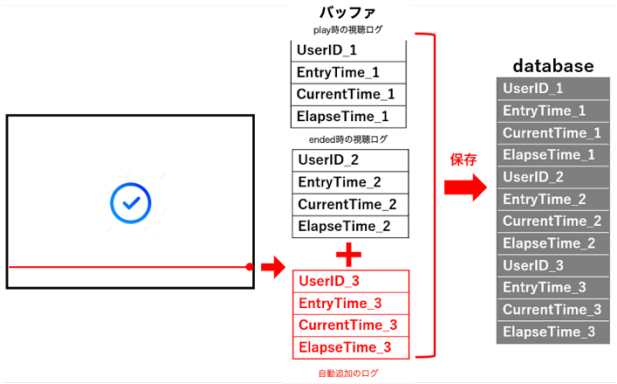

図 6. 視聴ログの自動追加の概要.

3.2. 動画プレーヤーへの対応

データベースモジュールのフィールド数を拡張するにあたり，視聴ログの保存が適切に行われるよう，動画プレイヤーの機能も改良する．具体的には，フィールド数の増減に柔軟に対応できるデータ保存方式を導入し，将来的な拡張にも適応可能な設計とする．この機能を動画プレイヤーに実装するために，Moodle API である「mod_data_view_database」を使用する．「mod_data_view_database」は，データベース ID を基にデータベースモジュールのフィールド ID を取得することができる．図 5 で示すように，この API を活用して，視聴ログの保存先であるデータベースのフィールド ID を動的に取得する．この仕組みにより，動画プレイヤーはデータベースのフィールド数の増減に柔軟に対応し，視聴ログの保存が可能となる．

なお，この動画プレイヤーの改善にあたっては，慎重に考慮すべき点がある．それは，操作イベントが発生しているにもかかわらず，学生の視聴ログが正常に保存されない可能性があることである．具体的には，改良後の動画プレイヤーでは，視聴ログはデータベースのフィールド数に応じて，一定量に達した時点で一括してデータベースに保存する必要がある．そのため，すべてのフィールドが完全に満たされていない不完全なエントリは送信されない．例えば，図 6 が示すように，フィールド数を 3 倍にした際に，学生が 1 度も動画プレイヤーに操作を行わなかった場合，視聴ログは動画開始時の「play」と動画終了時に「ended」の 2 つのイベントのみがバッファに記録されたままとなり不完全なエントリとなってしまうため，データベースには送信されない．この問題に対処するため，操作イベントを自動的に一定量発生させる機能を追加する．具体的には，図 6 のように動画の終了時およびページのアンロード時に ended イベントと unload イベントを，データベースモジュールのフィールド数に応じて発生させることで，未送信のバッファに残った視聴ログを確実に保存できるようにする．この対応により，動画プレイヤーで明示的な操作が行われなかった場合でも，最低限の視聴ログが記録され，可視化グラフの作成に必要な情報が確保される．結果として，視聴ログの欠損を防ぎ，より正確で信頼性の高いデータ分析が可能となる．

加えて，動画プレイヤーに重複する視聴ログの保存を防止する機能を実装した．この機能は，同一の操作イベントが連続して記録された場合，これを検出して視聴ログの送信データから排除する仕組みである．この改良により，冗長な視聴ログの保存を抑制し，データの蓄積を効率化することが可能になる．

4. 視聴状況の共有機能

4.1. 要件・設計

千葉大学では Moodle を用いて講義動画を提供する際，SCORM を活用して受講生の視聴状況を Moodle に保存しているが，その視聴状況は動画ごとに管理されており，一般に他の動画の視聴状況とは無関係である．

一方，動画間で視聴状況が共有されることが望ましい場合もある．その一例としては，同じ内容の動画を複数のビットレートで提供する場合が挙げられる．実際，現在千葉大学で利用している動画配信システムでは，通常画質モードと低画質モードの 2 つのビットレートでの動画を提供しているが，これら 2 つのビットレートを切り替えた際も視聴状況は共有されており，切り替えにあたって，視聴状況が元に戻ってしまうことはない．

同様に，同一内容の動画を複数の言語で提供する場合についても，各言語間で視聴状況が共有されることが望ましいと考えられる．同一内容の動画を複数の言語で提供する場合も，受講者は提供される言語のいずれかの動画を視聴すればよく，視聴状況のログが複数の動画に分散することを防ぐためにも，視聴状況は各言語の動画ごとに管理するのではなく各言語の動画間で共有されていない状況が望ましい．

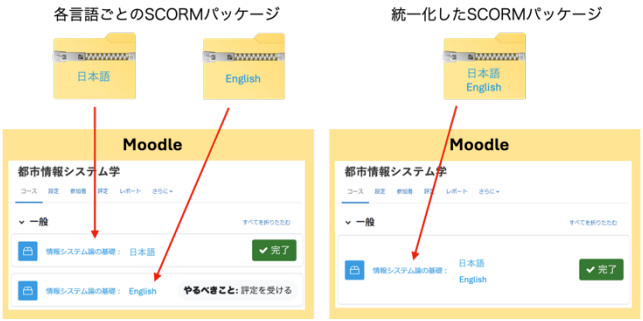

図 7. Moodle のコース画面での視聴完了（活動完了）の確認の様子. 各言語の動画ごとに SCORM パッケージを利用した場合（左図）と，言語間での視聴状況を統一的に管理可能とした SCORM パッケージを利用した場合（右図）.

そこで本研究では，各言語用に用意された複数の動画から言語を選択して再生する際，複数の動画間で視聴状況を共有して，１つの SCORM パッケージを通じて Moodle に視聴状況を保存できるようにする(図 7).

本機能の設計において，以下の要件を満たす必要がある.

要件 1：自由な言語選択を可能とすること
要件 2：視聴状況を共有すること
要件 3：既存環境への影響を最小限に留めること

受講者は用意された複数の言語の中から任意の言語を選択でき，選択に応じて適切な動画に自動的に切り替わる必要がある（要件 1）. また，講義動画は各言語間で同一の内容を想定しているため，視聴状況は全ての言語の動画間で共有される必要がある（要件 2）.

さらに，多言語機能の追加により，Moodle や動画配信システムの本来の機能の信頼性を下げることがないよう，既存環境への影響は最小限に留める必要がある（要件 3）.

上記の要件を満たすため，ここでは既に現行の動画配信システムで実装されているビットレート変更機能を応用して，言語選択に伴う動画の自動切り替え，及び視聴状況の共有を実現する. このように既存システムの実装を応用することで，既存環境への影響を最小限に抑えている.

4.2. 実装

多様な学習環境やデバイスにおいてオンデマンド授業を受講できるよう，柔軟性を考慮して設計されている. そのひとつとして，現行の動画配信システムは，多様な受講様式に伴う通信環境への対応として配信動画のデータダイエットを実現するために，マルチビットレート化に対応しており，視聴者は意識的に低いビットレートを選択することができる.

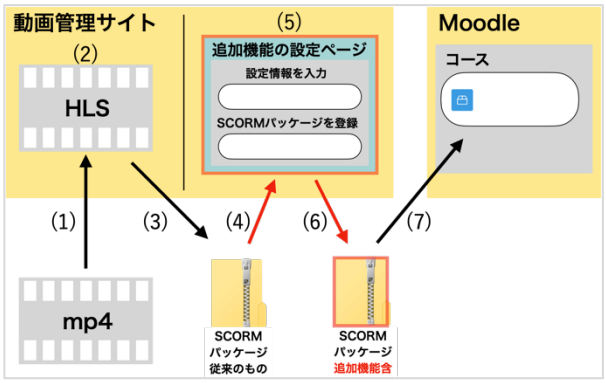

図 8. 追加機能を含む SCORM パッケージの作成手順.

このマルチビットレート化への具体的な対応は，マスタープレイリスト中でビットレートに応じたサブプレイリストへの振り分けを記述することで実現している. ビットレートの変更が行われると，動画プレイヤーが参照している動画のソースを更新した後に，続けてページ全体をリロードすることで切り替えが完了となる. また，リロード前に Cookie を利用して選択されたビットレート情報を保存しているため，リロード後も選択内容が保持された状況となる.

また，マルチビットレート化では，ユーザーインターフェースとしてラジオボタンを用い，学生が任意のビットレートを選択できるように設計されている. 一方，多言語化では，ユーザーインターフェースとしてリストボックスを用いて，学生が任意の言語を選択できるように設計した.

単一の SCORM パッケージ内の HTML ファイルに各言語に対応する講義動画の URL，暗号化キー，識別子といった動画の情報を記載することにより，動的に動画のソースを変更できる仕組みを実現した. これにより，各言語に対応した講義動画を共通の SCORM パッケージを通じて提供することが可能となり，視聴状況はすべての言語動画間での統一的な記録・共有が可能となる.

5. 動画管理サイトへの機能追加

本論文で議論した 2 つの追加機能は SCORM パッケージを通じて機能するため，既存の SCORM パッケージに追加機能の設定を含める必要がある. このための作業を教師が負担なく行えるように，追加機能を含んだ SCORM パッケージを作成するための機能を動画管理サイトの別ページに追加する.

5.1. 要件・設計

以下に追加機能を利用可能とする SCORM パッケージ作成の手順を示す（図 8）．手順（3）までが既存の SCORM パッケージを作成する手順であり，手順（4）以降が追加機能を含む SCORM パッケージを作成する手順である．

図 8 の手順の詳細は次のとおりである．(1) 教師は動画管理サイトで mp4 形式の講義動画をアップロードする．(2) 動画が HLS 形式に変換され，SCORM パッケージが作成される．(3) 変換された講義動画を再生するための SCORM パッケージをダウンロードする．(4)動画管理システムへの追加機能で必要となる情報を入力し，(3) でダウンロードした SCORM パッケージをアップロードする．(5) サーバーで追加機能を有効とした SCORM パッケージを作成する．(6) 追加機能が有効となった SCORM パッケージが作成されると，クライアントに送信される．(7) 教師は追加機能を含む SCORM パッケージを Moodle に登録する．

5.2. 実装

現行の動画管理サイトでは，Perl/CGI を利用して Basic 認証の弱点を補強した「改良型 Basic 認証」を実装し，より安全かつ効率的なユーザーログイン管理を実現している．また Perl/CGI では，環境変数を活用することでログイン中の教師の情報を取得することができるため，追加システムの実装にあたっては，現行の動画管理システムの構築で用いられている Perl/CGI を利用することで新たなログイン機能を一から構築することを避け，開発コストの削減を図っている．

5.2.1. データベース ID とトークンの入力

教師が本システムにアクセスした際に，サーバーが CGI スクリプトを実行してデータベース ID やトークンの入力機能を含む HTML ページを動的に生成・表示する．さらに，動画管理サイトで作成した SCORM パッケージをアップロードする仕組みとなっている．教師が必要事項及び ZIP ファイルをアップロードすると，HTTP リクエストを介してそれらのデータがサーバー側に送信される．なお，一度利用したトークンはサーバーの環境変数から得た教師のログイン ID の名前のディレクトリーに保存され，次回以降からは自動的にトークンが入力されるようになっている．

5.2.2. SCORM パッケージ化とその Zip ファイルダウンロード機能

教師がファイルをサーバーにアップロードすると追加機能を有効とした SCORM パッケージを作成するための CGI スクリプトが実行される．まずセッションディレクトリが作成され，このディレクトリ内でアップロードした ZIP ファイルが保存・解凍される．次に，送信されたデータを基に，本システム用の JavaScript ファイルが生成され，解凍された ZIP ファイルに追加される．最後に，更新された ZIP ファイルが再度作成され，HTTP リクエストを通じてクライアントに送信される．

6. 結論

本研究で開発した機能は，Moodle の標準機能のみを活用する設計を採用することで，システムの改変を最小限に抑え，Moodle のバージョンアップ時に伴うコストや労力の軽減を実現した．また，課題に対して適切な改良を加えることで、システムの精度と柔軟性を向上させた．この結果，教員は学生の視聴状況をより詳細に把握することが可能となった．

文　献

[1] 清水健一, 藤本茂雄, 松本暢平, 檜垣泰彦, “Moodle 用動画配信システムの内製とコロナ禍における運用・評価”, 電子情報通信学会論文誌(D), J105-D (10), pp.572-583, July 2022 DOI: 10.14923/transinfj.2021LIP0015.

[2] 渡邉 康太, 藤本 茂雄, 檜垣 泰彦, “Moodle 用動画配信システムにおける視聴履歴の可視化”, 電子情報通信学会技術研究報告, vol. 123, no. 429, pp. 13-18, March 2024.