動画コンテンツ管理・配信システムの構築

○松能　誠仁 [a)]、浜　貴幸 [a)]
[a)] 金沢大学総合技術部情報部門

１．はじめに

2020 年頃からの新型コロナウイルス感染症拡大を契機に、オンライン授業や会議など、動画コンテンツの利用が急速に拡大した。Zoom や Webex といったプラットフォームを利用した会議だけでなく、授業においてもオンデマンド動画配信が盛んに行われるようになり、動画コンテンツの需要は増加している。

従来、本学では Wowza Streaming Engine[1] を利用して、必要に応じてその都度行う形で運用していたが、増え続ける動画コンテンツを体系的に管理するシステムがなかった。

このような状況を受け、既存の基盤は活かしつつ、誰もが簡単に動画コンテンツをアップロード、整理、検索、公開設定できる、使いやすい動画専用のコンテンツ管理機能を持つ、本格的な動画コンテンツ管理・配信システムを構築することになった。

本報告では、この新たに構築した動画コンテンツ管理・配信システムについて、詳細を説明する。

２．動画コンテンツ管理・配信システム

本システムは、Web アプリケーションとして提供され、あらゆる端末からアクセス可能である。動画ファイルをアップロードすると、配信に適した形式に自動変換され、PC やモバイル端末のブラウザから視聴できる。本学の統合認証によるアクセス制限機能を備え、クローズドな環境での動画共有も可能である。

２.１　機能紹介

基本機能に加え、運用中に寄せられた多くの要望に基づき、機能を追加実装した。以下に、その機能を紹介する。

1) 基本機能

図 1 の動画アップロード画面から動画をアップロードすると、図 2 の管理画面に反映される。公開設定を行うことで、図 3 のようにブラウザでの視聴が可能となる。

図 4 の設定画面では、個別の動画に対し、閲覧制限、LMS などへの埋め込み設定、タグ付けなどの詳細な設定が行える。

図 1　動画のアップロード画面

図 2　動画の管理画面

図 3　動画の視聴画面

図 4　動画の閲覧設定画面

2) トップページ公開機能

これまでの仕様では、各動画に 1 つの URL が割り当てられ、その URL を知っているユーザーだけが動画にアクセスできる仕組みであった。しかし、より多くのユーザーに動画を視聴してもらうため、トップページに動画を掲載できる機能を追加した。これにより、トップページから直接、特定の動画を視聴できる。

図 5　トップページ公開画面

3) ショーケース機能

アップロードした動画を 1 ページに一覧表示し、特定のコンテンツをまとめて表示できるようにした。

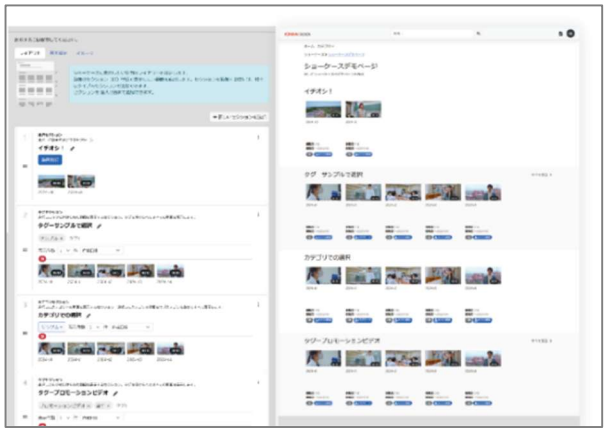

図 6　ショーケース画面と設定画面

4) 共同管理機能

当初は授業用動画を主な対象としていたが、その後、研修動画、広報動画など、様々な目的で利用される動画が増加した。その結果、動画を個人で管理するだけでなく、部署やチームなど、複数のメンバーで共同管理したいという要望を多数いただくようになりました。このニーズに対応するため、動画共同管理できる機能を実装した。

2.2　利用状況

図 7 と図 8 に、本システムの運用開始から現在までの動画のアップロード数と動画閲覧数に示す。

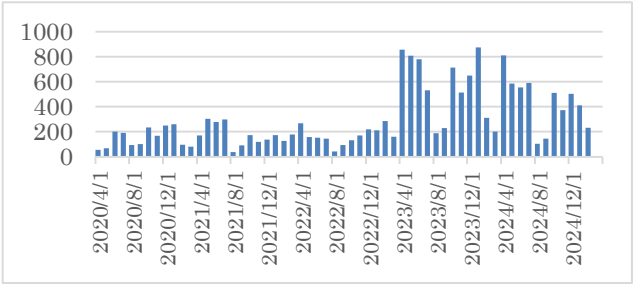

図 7　動画アップロード数

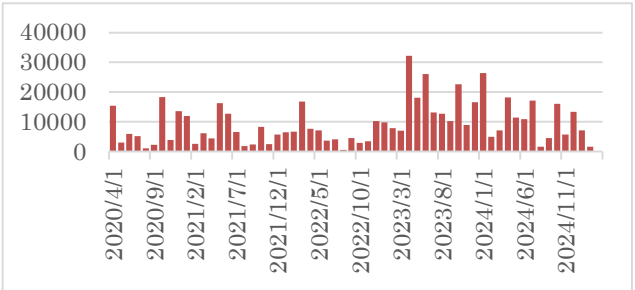

図 8　動画閲覧数

2023 年 4 月に LMS へのアップロードができなくし、本システム用に一本化されたため、一時的に動

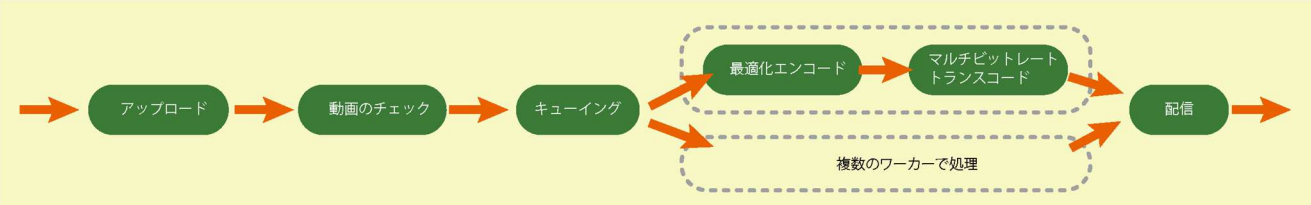

図 9　動画ファイルを配信するまでの処理の流れ

画数と閲覧数が増加した。

現在においても少しずつ数は減ってきてはいるが、オンライン教材や研修用の動画、講演・会議の記録など、幅広く活用されている。

２.３　システム構成

本システムは、NVIDIA 社製 GPU A40 を搭載した 2 台の物理サーバ上に仮想基盤に構築している。その仮想基盤上に複数の仮想マシンを作成し、各仮想マシン上でアプリケーションやコンテナを動作させる構成である。

構成するにあたって、Web サーバ、アプリケーションサーバ、データベースに加え、動画配信するサーバ、動画最適化する変換サーバ、非同期処理とマルチタスク化が必要なのでキューイングサーバなど複数の機能が必要になった。

そこで、コスト削減と機能性確保のため、動画配信サーバ以外をオープンソースソフトウェアで構成した。

システム設計・構築においては、特に次の点を考慮した。

1)　Web フロント部分

従来は、Web アプリケーションのバックエンドに Ruby on Rails[6]、フロントエンドは Rails の View を主に利用していた。

今回のシステムでは、ユーザーインターフェースの改善と、よりリッチな機能の実装を目的とし、フロントエンドに SPA フレームワークの Angular[5]を採用した。これにより、API 経由でのバックエンド(Rails)との連携を行う構成とした。

2)　動画最適化処理

動画配信の前処理には、多くの処理、時間、そしてサーバーリソースが必要である。その流れを図 9 に示す。

配信プロトコルには、幅広い端末に対応する HLS(HTTP Live Streaming) [2]を、動画コーデックには H.264/AAC を採用した。アップロードされたファイルは、様々なフォーマットである可能性があり、確認と、配信用のコーデックへの変換が必要である。

また、ネットワーク環境に応じて画質を調整するため、複数のビットレートのファイルを用意する。この変換処理には、NVIDIA GPU のハードウェア支援機能(NVENC/NVDEC)[4]と FFmpeg[3]を活用し、処理時間を大幅に短縮した。さらに、この処理をコンテナ化し、分散処理を可能にすることで、アップロードから配信開始までの準備時間を短縮した。

３．まとめ

本システムは動画公開の簡易化という初期目標を達成し、多くのユーザーに利用されている。今後は、学習効果とコンテンツ品質の向上に加え、ユーザーエクスペリエンスの向上にも取り組み、技術と機能の両面で拡張を行う。

技術面では、次世代ストリーミング技術（MPEG-DASH、HEVC、AV1）やコンテンツの暗号化に対応した動画配信を実現する。機能面では、レコメンド機能や詳細検索、字幕機能などにより、動画の発見性と視聴体験を向上させる。また、著作権保護、不適切動画監視、制作者向け分析データ提供により、コンテンツ品質向上を支援する。これらの継続的な改善により、ユーザーにとって使いやすい、持続可能な動画配信プラットフォームを目指す。

参考文献

[1]　“Wowza Stream Engine”　https://www.wowza.com/

[2]　“HTTP Live Streaming” https://developer.apple.com/documentation/http-live-streaming

[3]　“FFmpeg” https://trac.ffmpeg.org/wiki/StreamingGuide

[4]　“FFmpeg with NVIDIA GPU Hardware Acceleration” https://docs.nvidia.com/video-technologies/video-codec-sdk/11.1/ffmpeg-with-nvidia-gpu/index.htm

[5]　“Angular” https://angular.dev/

[6]　“Rails on Rails” https://rubyonrails.org/