卒業論文　　2015 年度（平成 27 年）

Meetup における
プレゼンテーション記録システムの設計と実装

慶應義塾大学 環境情報学部

氏名：髙橋 俊成

卒業論文要旨 - 2015 年度（平成 27 年度）

Meetup における
プレゼンテーション記録システムの設計と実装

本研究では、Meetup におけるプレゼンテーションを記録するために、専用のプレゼンテーション記録システムを設計、構築した。

Meetup とは、共通の趣味やテーマで緩やかに繋がるオンラインコミュニティのメンバーが、知見共有や人間交流を目的として行うカジュアルな勉強会・交流会である。Meetup におけるプレゼンテーションでは、コミュニティにとって有意義な情報交換が行われている一方で、その記録作業は参加者個々人の自主的・献身的な活動に頼りきっており、成果をオンライン上のコミュニティに十分に還元できていない。

本研究では、プログラマーコミュニティによる Meetup を技術系 Meetup と定義し、そこで行われるプレゼンテーションをオンラインで共有しやすい形で記録するための要件を整理した。次にそれを満たす専用のプレゼンテーション記録システムを設計し、WebRTC や Electron 等の技術を用いて実装した。終わりに、その評価のために実証実験を実施し、本システムが技術系 Meetup におけるプレゼンテーションの記録に有用であることを確認した。

キーワード：
1. Meetup　2. プレゼンテーション，　3. オンラインコミュニティ，　4. WebRTC，　5. Electron

Abstract of Bachelor's Thesis

Design and implementation of a presentation record system for meetup

目次

図目次

表目次

第1章　はじめに

1.1.　背景

オンラインコミュニティは 1990 年代から始まった情報通信技術の発達のなかで登場し、今日まで発展してきた。古くはメーリングリストや IRC で、現在では SNS で、活発な非対面・テキスト主体のコミュニケーションが行われてきた。これらは同じ趣味やテーマによって緩やかに繋がる同質性の高い実質的なコミュニティである。

オンラインコミュニティのメンバーは時々、より密なコミュニケーションを求めて Meetup と呼ばれるオフラインイベントを開催することがある。Meetup は、知見共有・人間交流を目的とした対面・リアルタイムのカジュアルな勉強会・交流会である。Meetup の形態はコミュニティごとに様々だが、とくにプログラマーコミュニティによる技術系 Meetup では毎回設定されたテーマに沿って参加者のうち数人が発表者としてプレゼンテーションを行うセミナースタイルが主流である。技術系 Meetup の特徴は、オンラインコミュニティの活動の中でほぼ唯一のオフラインな活動だという点である。Meetup 開催前後のやりとりも全て WEB で行われるのが通例で、Meetup だけがオフラインでのリアルイベントである。

Build Insider が 2015 年 11 月に行ったアンケート調査 [1]によって、技術系 Meetup に参加する理由は WEB 上の資料だけでは得られない良質な知識や最新情報、議論、人脈を得たいからだと考察されている。プログラマーコミュニティでは得られない情報が技術系 Meetup において共有されているのであれば、それらを記録しプログラマーコミュニティに還元することで、プログラマーコミュニティ全体の情報量が増し価値が高まる（図 1)。

しかしながら技術系 Meetup の多くは個人が趣味的に開催するため、企業主催のカンファレンスのように発表や質疑応答に専任の撮影者や速記者が用意されることはまれである。現状、技術系 Meetup におけるプレゼンテーションの記録手法は、聴衆によるボランティア的な文字起こしか、そもそも全く記録していないかのどちらかである。聴衆による文字起こしとは、有志がマイクロブログサービス等にプレゼンテーションの内容やその後の質疑応答を同時進行的にテキストに書き起こし、まるで実況中継のようにマイクロブログサービスへ投稿する手法である。マイクロブログサービスはその性質上、投稿ひとつひとつが独立しており、また時間がたつと投稿内容を参照することが難しくなる [2]ので、さらにマイクロブログまとめサービスを用いて投稿された文字起こしを編纂することもある。

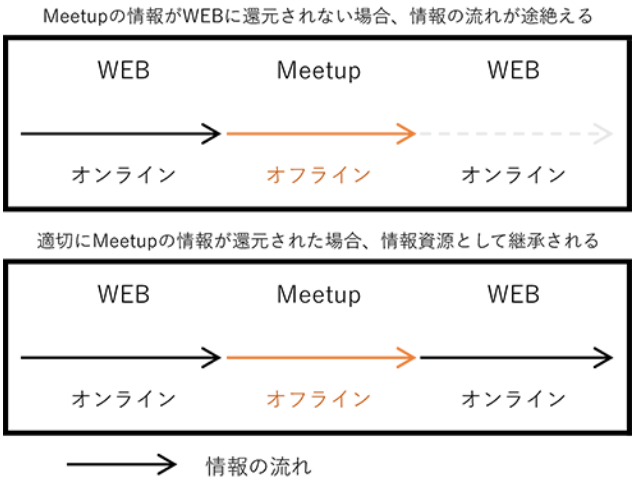

図 1　プログラマーコミュニティにおける情報の流れ

この手法の問題点は、文字起こしの負担が大きい点と、聴衆の自主的・献身的な活動に頼りきっている点、そして記録の品質が保証されない点である。リアルタイムに書き起こして投稿する作業はそこそこのタイピング能力と集中力を求められ、しかもそれは聴衆の自主的・献身的な活動に支えられている。文字起こしができ、かつやりたいという聴衆が名乗り出ない限り記録作業が行われない。また、運よく献身的な聴衆が現れたとしても、その記録フォーマットに統一されたものがないために、個々人の能力や主観によって記録内容が欠けたり、変化したりする恐れがある。

1.2.　本研究の課題

本研究では、技術系 Meetup のプレゼンテーションにおいて専任の記録担当が用意されないために、そこで共有された知見を正確に記録しオンラインコミュニティにきちんと還元できていないという問題点に着目する。

この問題を解決するために、技術系 Meetup 専用のプレゼンテーション記録システムを構築し、参加者の献身性に頼ることなくプレゼンテーションを記録しオンラインコミュニティに還元できる環境を実現する。

1.3.　本研究の目的

本研究では、Meetup におけるプレゼンテーションの記録を目的とする。そのためのアプローチとして、プレゼンテーションのリアルタイム WEB 配信機能を備えた専用のプレゼンテーション記録システムを構築する。

このプレゼンテーション記録システムは、Meetup におけるプレゼンテーションを全て WEB 上で配信しながら記録する。これにより、オフラインで行われている Meetup を文字起こし等の変換作業なしに記録できる。記録データは、発表者のスライド資料に参加者の属性情報や活動を付加したテキスト主体の「Meetup プレゼンテーション記録フォーマット」とすることで、WEB で活動するコミュニティが参照しやすくする。

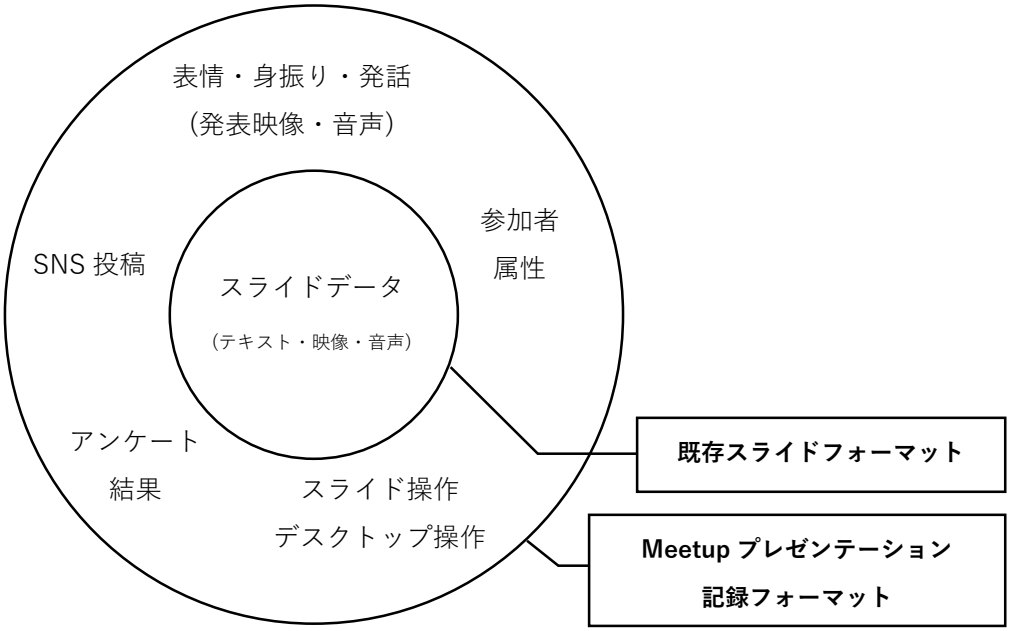

図 2　Meetup プレゼンテーション記録フォーマットの概念図

1.4.　用語定義

本節では、本論文中で用いられる主な用語を定義する。

- Meetup（ミートアップ）
  オンラインコミュニティのメンバーが、知見・意見共有と人間交流を目的として行うカジュアルな勉強会や交流会。参加者のうち数人が発表者となりプレゼンテーションを行う。発表内容は、後日発表者によって任意にオンライン公開され、オンラインコミュニティに共有されることがある。

- オンラインコミュニティ
  インターネットを介して活動する実質的なコミュニティの総称。共通の趣味・テーマなどの情報共有・意見交換・人間交流を主な目的に、緩やかに繋がっている。特定

の場所や組織に実際に行かなくともネットワークを経由して知的資産を得られる場として機能する。

- プレゼンテーション
  発表者がスライド資料を用いて聴衆に対して情報を掲示する行為と、それに付随する参加者同士の議論や意見交換。

1.5. 本論文の構成

2 章では技術系 Meetup におけるプレゼンテーションを分析し、問題解決のためのアプローチを提案する。3 章で既存のスライドフォーマットとその問題点について分析する。4 章で技術系 Meetup におけるプレゼンテーション記録システムを設計し、5 章で実装について述べる。6 章では本システムを用いた実証実験について述べ、7 章でその評価を行う。8 章では結論と今後の課題について述べる。

第2章　技術系 Meetup におけるプレゼンテーションの分析

本章では、技術系 Meetup におけるプレゼンテーションについて分析し、プレゼンテーションを記録するためのアプローチを述べる。

2.1.　技術系 Meetup の分析

本節では、技術系 Meetup の定義し、その特徴を述べる。

本研究では、技術系 Meetup を「プログラマーコミュニティによって開催されるカジュアルなソフトウェア技術の勉強会」と定義し、プログラマーコミュニティを「主にソフトウェア開発プロジェクト共有 SNS やプログラマー向け知識共有 SNS を活用して交流しているプログラマーのオンラインコミュニティ」と定義する。

図 1 に、プログラマーコミュニティにおける情報の流れを示す。技術系 Meetup は、1 人以上のコミュニティメンバーが発起人となり、イベント開催支援サイトを介して告知・募集される。興味を持った他のコミュニティメンバーが参加し、同じ日時に同じ場所に集い Meetup が開催される。開催後は Meetup 内で使われた発表資料や簡単なレポートがコミュニティ内外に投稿されることがある。

技術系 Meetup の流れを図 3 に示す。技術系 Meetup では、発表者の数だけプレゼンテーションを繰り返し、終わりに懇親会が開かれる。プレゼンテーションは知見共有を目的とした発表や質疑応答で、懇親会は人間交流を目的とした食事や飲み会である。懇親会の価値は 2 者間の関係性にあり、それは極めて属人的であるため、本研究における記録対象には含まない。

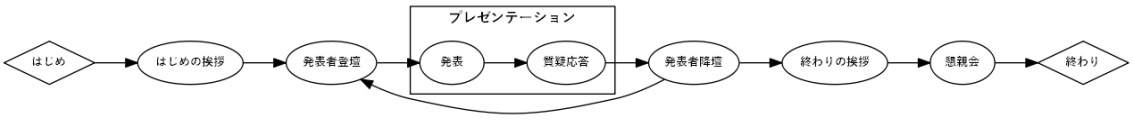

図 3　技術系 Meetup のタイムライン

2.2.　技術系 Meetup におけるプレゼンテーションの特徴

本節では技術系 Meetup で行われるプレゼンテーションの特徴を述べ、記録システムの設計に必要となる各要素の特徴について考察する。

2.2.1.　技術系 Meetup におけるプレゼンテーションの登場人物

本研究では、プレゼンテーションの進行管理を行う者を司会、プレゼンテーションで登壇し発表する者を発表者、それを視聴する者を聴衆と定義する。また、聴衆の SNS 投稿を閲覧することで間接的にプレゼンテーションを視聴するオンラインコミュニティのメンバーを遠隔の聴衆と定義する。

技術系 Meetup におけるプレゼンテーションの構成を図 4 に示す。技術系 Meetup におけるプレゼンテーションでは、司会と発表者、聴衆の 3 種類の参加者が、同じ会場に同じ時間に存在する。また遠隔の聴衆がインターネット越しに同じ時間を共有する。プレゼンテーションはセミナー形式で行われる。司会は発表者や聴衆を兼ねることがあり、また発表者は自身のプレゼンテーション以外では聴衆として振る舞う。

2.2.2.　技術系 Meetup におけるプレゼンテーションの流れ

技術系 Meetup におけるプレゼンテーションの流れを図 5 に示す。

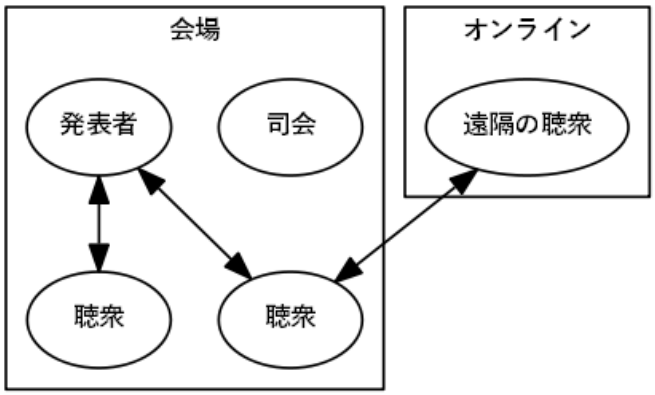

図 4　技術系 Meetup におけるプレゼンテーションの登場人物

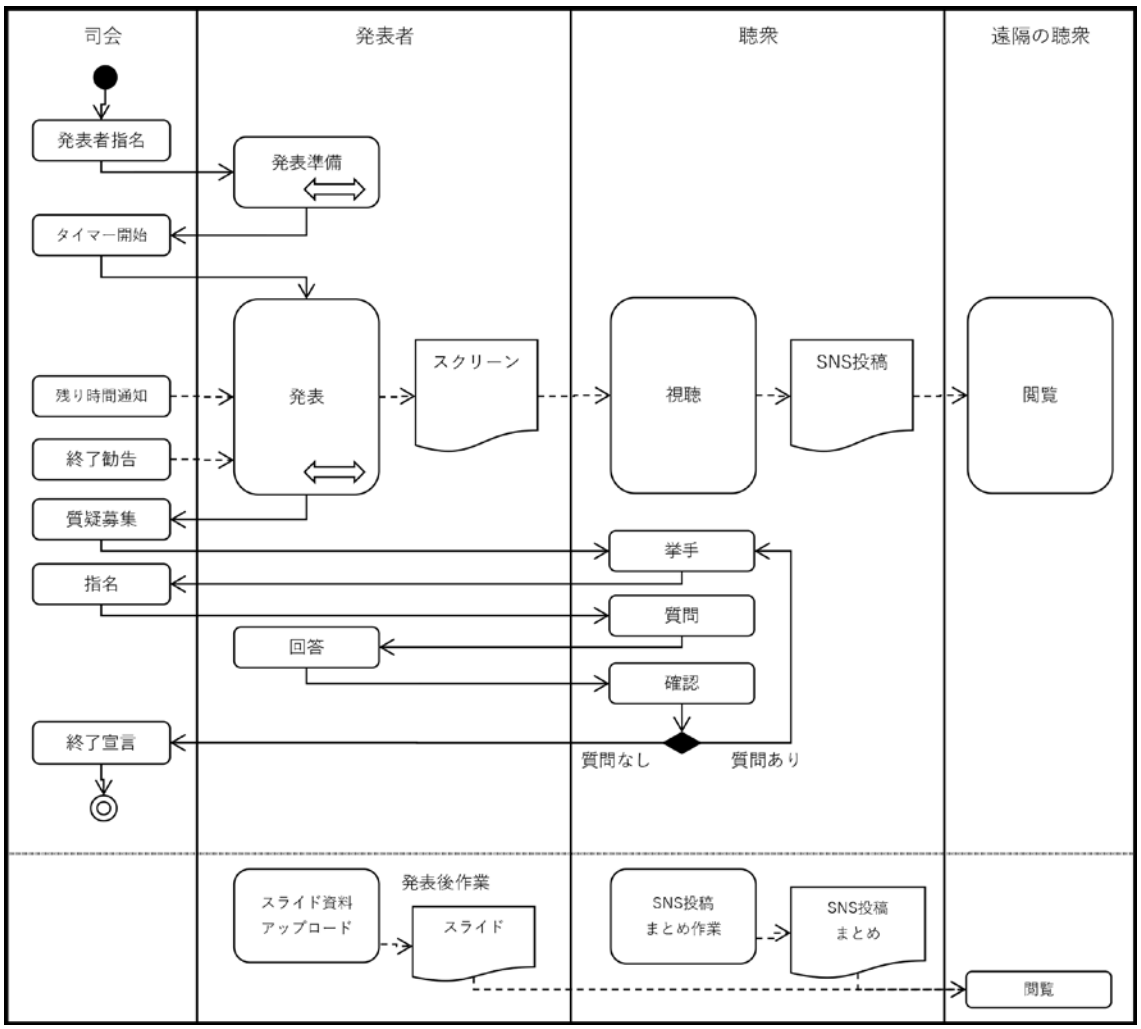

図 5　技術系 Meetup におけるプレゼンテーションの流れ

2.2.3.　技術系 Meetup におけるプレゼンテーション発表

表 1 に、プレゼンテーション中の発表者の行動を示す。発表者は、ラップトップ型コンピューターをプロジェクター等に繋ぎ、スライド資料等を大画面に映しながら発表する。発表内容によっては途中でデモンストレーションやライブコーディング等の実演をするため、スライド資料のプレゼンテーションモードを中断してデスクトップ映像に切り替えたり、またプレゼンテーションモードを再開したりと、切り替え作業を行うことがある。また発表者が聴衆の嗜好や属性を把握するために、挙手等を呼び掛けてアンケートを行うことがある。発表終了後は質疑応答が行われ、数人の聴衆が質問や感想を口頭または SNS 投稿で述べ、発表者がそれらを拾い口頭で答える。発表に使ったスライド資料は、プレゼンテーション終了後にスライド共有サイトを通じてオンラインコミュニティに共有される。

表 1　発表者の行動

行動	説明
発表	資料を見せながら発話する
スライド操作	スクリーンにスライド資料を映し操作する
ポインタ操作	スライド資料の特定座標を指し示す
実演/デスクトップ操作	スクリーンにデスクトップを映し操作する
質問/アンケート	聴衆に質問をして回答を促す
指名	質疑応答で質問者を指名する
回答/議論	割り込みや質疑応答の質問に対して回答する

なお、技術系 Meetup ではハンズオンと呼ばれる体験型発表が行われることもある。ハンズオンは発表者の指導のもと聴衆が実習をこなして技術への理解を深める。発表と質疑応答の区別は曖昧で、実習中は発表者と聴衆との間で多くのやり取りがなされる。本研究ではハンズオンはプレゼンテーションとは異なるものとして扱い、記録対象からは外す。

2.2.4.　技術系 Meetup におけるプレゼンテーション視聴

表 2 に、プレゼンテーション中の聴衆の行動を示す。聴衆は、ラップトップ型またはタブレット型コンピューターを操作しながらプレゼンテーションを視聴する。一部の聴衆は発表や質疑応答の内容や感想をリアルタイムにマイクロブログサービスへ投稿し、遠隔の聴衆がそれを閲覧する。プレゼンテーション終了後、マイクロブログサービスの投稿は聴衆によって編纂され、個人ブログやマイクロブログまとめサービスを通じてオンラインコミュニティに共有される。

表 2　聴衆の行動

行動	説明	具体例
非言語反応	発表内容への非言語的な反応	頷く、拍手、笑い声、首を傾げる
割り込み	発表を遮って発言する	「そのグラフの縦軸は何を示していますか？」
聴衆間会話	聴衆同士で会話する	「いまさっき発表者なんて言った？」「さあ」
回答	発表者からの質問に回答する	挙手
実況投稿	発表内容を SNS に投稿する	「仕事効率化に関する発表です #happyo15」
感想投稿	発表への感想を SNS に投稿する	「わかる #happyo15」「下手だなあ #happyo15」

質問投稿	発表への質問を SNS に投稿する	「調査に使った機種は何だろう #happyo15」
検索	発表に関する事柄を検索する	サーチエンジンの利用
質問要求	質問の意思表明をする	挙手
質問/議論	質疑応答で質問する	「結局何を言いたかったのでしょうか？」

聴衆がコンピューターを操作するのは、ハンズオン形式の発表で実習をしたり、発表中に話題に上がった物事をすぐ検索して調べたり、発表の実況や感想をマイクロブログサービス等の SNS に投稿したりするためである。遠隔の聴衆とは、リアルタイムに聴衆の SNS 投稿を閲覧することで間接的にプレゼンテーションを視聴しているかのような状況になった者をさす。

2.3.　技術系 Meetup におけるプレゼンテーションの構成要素

図 2 は、技術系 Meetup におけるプレゼンテーションの構成要素を表したものである。Meetup におけるプレゼンテーションでは、まず参加者がいて、次に発表者のスライド資料やそのコントロール情報があり、そして放送者や参加者の表情や身振りなどの視覚情報、発話や拍手などの音声情報がある。これらの基本的な構成要素に加えて、会場内外の聴衆による SNS 投稿も構成要素として挙げられる。

2.4.　本研究のアプローチ

技術系 Meetup におけるプレゼンテーションはオフラインの活動であるために、オンラインのプログラマーコミュニティに情報を還元するには内容を記録しコンピューターへ入力する作業が必要となる。

そこで本研究では、Meetup におけるプレゼンテーションを最初からオンラインで配信することで、普段のオンラインコミュニティの活動同様にプレゼンテーションへの参加がそのままコンピューターへの入力作業となり、ボランティアや専任の担当者なしに機械的にプレゼンテーションを記録できる環境を実現する。スライド資料や SNS 投稿等最初からデジタルデータなものはそのまま配信し、発話や身振り等は録画して配信する。

第3章　既存スライドフォーマットとその問題点

本章では、スライド資料における既存フォーマットを分析し、それぞれの問題点を整理する。

3.1.　既存のスライドフォーマットと記録可能なデータ

スライド資料に用いられるフォーマットと表現可能なデータの組み合わせの可否を表 3 に示す。

表 3　スライドフォーマットと記録可能コンテンツ

	OOXML	MP4	HTML
テキスト	○	△	○
静止画	○	○	△
映像/音声	○	○	×

○記録可能　△記録できるが制限がある　×記録できない

3.2.　既存のスライドフォーマットの問題点

本節では、既存のスライドフォーマットの問題点についてそれぞれ述べる。

3.2.1.　OOXML

OOXML (Open Office XML, ISO/IEC 29500:2012)は、Microsoft PowerPoint などのスライドツールで用いられる XML ベースのファイルフォーマット。

テキストや画像、映像などのマルチメディア資料に対応し、表現力も普及度も随一のスライドフォーマットである。しかし標準化されてから日が浅く、また高機能ゆえに非常に複雑で、OOXML フォーマットを扱えるアプリケーションは少ない。WEB ブラウザーも OOXML を直に扱うことはできないため、スライド共有サイトでは画像や HTML に自動変換される。

3.2.2.　MP4

MP4 (MPEG-4 Part 14, ISO/IEC 14496-14:2003)は、映像や音声、各種メタデータを格納できるメディアコンテナである。

規格としては動画、音声、テキストに対応するが、全てを完璧に再生できるシステム少ない。例えば WEB ブラウザーでは、MP4 ファイルのうち映像が H.264 フォーマットで、音声が AAC または MP3 フォーマットで記録されているものだけが再生できる。メタデータに含まれる字幕等のテキストデータは WEB ブラウザーでは解釈されないので、オンラインコミュニティに対して MP4 でスライド資料を提供するとき、それは実質的に映像と音声のみとなる（ただし別途 WebVTT 等のトラックファイルを用意し video 要素で MP4 ファイルと紐づけることで、映像と同期したテキストは実現できる)。映像はビットマップデータのため、作成時の縦横比で常に固定となる。またインタラクティブな要素は含められない。

3.2.3.　HTML

HTML は、WEB で最も普及している文書フォーマットである。構造化されたテキストデータなので内容の検索性が高い。ハイパーリンクや video タグを用いればマルチメディアも参照できるが、あくまで参照であり単体での表現力には欠ける。画像は Base64 エンコーディングを用いて文字列化することで埋め込める。

HTML に CSS や JavaScript、そしてリンクされたメディアファイル等を組み合わせて作られたスライド資料は WEB スライド（Web-based slideshow [3]）と呼ばれ、機能も実装も統一されていないが WEB ブラウザーで閲覧できる点は共通している。レスポンシブに作成することで様々な画面比率に対応でき、またオンラインコミュニティとの相性も良い。しかしファイルがバラバラであるため、Web 以外の手段でデータを送りあう場合は zip 等でひとつのファイルにまとめるなどの工夫が必要になり面倒である。

第4章　技術系 Meetup におけるプレゼンテーション記録システムの設計

本章では、これまでの分析を踏まえた上で、Meetup におけるプレゼンテーションの記録を実現するために必要なシステムを設計する。

4.1.　記録システムの要件定義

本節では、技術系 Meetup におけるプレゼンテーション記録システムに必要な要件を整理する。本システムに求められる機能要件を以下に示す。

- 2.3 で述べた技術系 Meetup におけるプレゼンテーションの構成要素を記録できる
  - 参加者の属性を取得し、一意に識別できる
  - 発表情報（発表概要や発話、資料、その操作）を全ての聴衆に配信できる
  - 聴衆が発表に対してコメントでき、参加者全員がそれを見られる
  - 発表者がアンケートを行い、聴衆が回答でき、参加者全員が結果を見られる
- 参加者は各自のコンピューターからプレゼンテーションに参加できる
- 配信情報から自動的にプレゼンテーションの記録を作成できる
- 作成した記録をプログラマーコミュニティへ容易に共有できる
  - プログラマーコミュニティにとって扱いやすい形式で出力できる
  - 記録の一部をレポートとしてプログラマーコミュニティに投稿できる

本システムで記録対象とするプレゼンテーション構成要素を表 4 に示す。スライド資料は、スライド共有サイトの利用が一般化していることを鑑みて、データそのものを持たずに外部 URL への参照という形で記録する。

表 4　プレゼンテーションの記録対象とデータ形式

記録対象	データ形式	付随するメタ情報
スライド資料	テキスト(URL)	発表者
参加者属性(SNS アカウント)	テキスト	

スライドコントロール（ページめくり、発表時間）	テキスト	
カメラ映像	ビデオ	動画コーデック、撮影対象、撮影者
SNS 投稿	テキスト(URL)	投稿者、投稿内容、投稿時刻、ページ番号
アンケート	テキスト	質問内容、質問時刻、選択肢ごとの回答者数

4.2. 記録システムの全体設計

本節では、前節で整理した機能要件を満たす設計について述べる。

4.2.1. プレゼンテーション記録システムの全体構成

プレゼンテーション記録システムは、発表者用のクライアントアプリケーション、聴衆用のクライアントアプリケーション、各クライアントを繋ぐサーバーアプリケーションの 3 種で構成する。このうち聴衆用クライアントは、オンラインコミュニティが利用しやすいという要件を満たすために WEB ブラウザー上で動作する WEB アプリケーションとする。

4.2.2. システム化プレゼンテーションフローの設計

図 6 で、本システムを用いたプレゼンテーションの大まかな流れを表す。プレゼンテーションを全てオンライン上で行うというアプローチを実現するために、発表前にシステムに発表情報を登録する「発表準備」を行う。そのため、発表に使うスライド資料は発表前に共有を済ませておく必要がある。「発表」では発表者の操作が全ての聴衆にブロードキャストされ、全ての参加者が常に同じ画面を見ながらコメントや質問を投稿できるようにする。発表を終了すると「質疑応答」に移り、発表中や発表後に投稿された聴衆のコメントを発表者が閲覧しながら回答する。プレゼンテーション中の活動は全て記録される。プレゼンテーション終了後に「ログ保存/アップロード」でプレゼンテーションの記録を保存または SNS 投稿できるようにする。

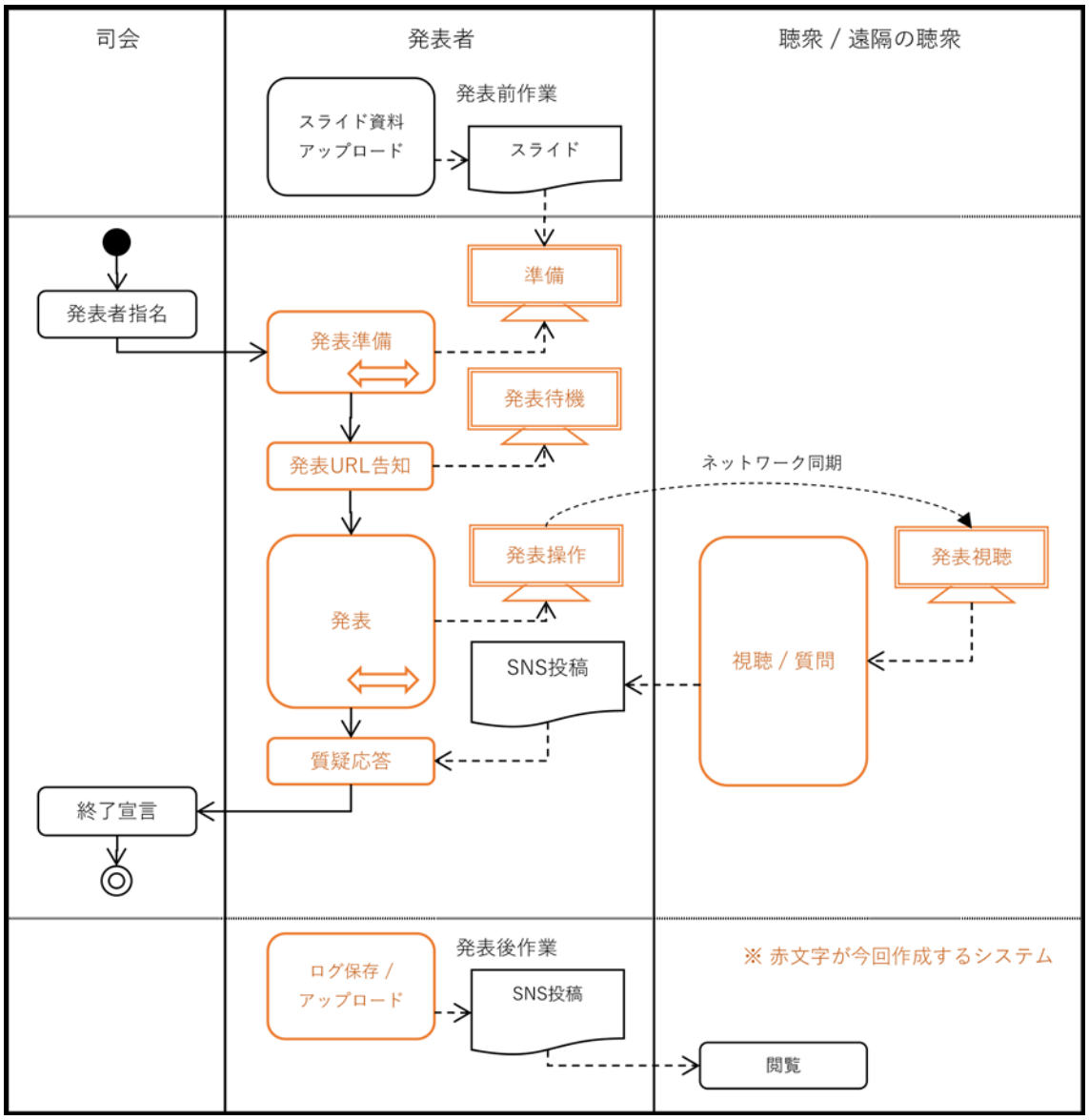

図 6　システム化プレゼンテーションフロー全体図

4.3.　発表者用クライアントアプリケーションの設計

本節では、本システムのうち発表者用クライアントアプリケーションの設計について、機能一覧表と画面構成図を示しながら述べる。

4.3.1.　発表者用クライアントアプリケーションの振る舞い

本項では、本研究で実装する発表者用クライアントアプリケーションの機能及び動作の概要を述べる。まず、表 5 で本アプリケーションの機能の一覧を示す。

表 5　発表者用クライアントアプリケーションの機能一覧

No.	大機能	小機能	機能概要
P1	発表者登録	発表者情報取得	保存された発表者の属性情報を取得する
P2		GitHub 認証要求	サーバーから GitHub アカウントの認証用 URL を取得し、標準ブラウザーに表示する
P3		GitHub PIN 入力	GitHub PIN コードの入力を受け付ける
P4		GitHub 認証	入力された GitHub PIN コードをサーバーに送り、GitHub アカウント情報を取得する
P5		Google 認証要求	サーバーから Google アカウントの認証用 URL を取得し、標準ブラウザーに表示する
P6		Google PIN 入力	Google PIN コードの入力を受け付ける
P7		Google 認証	入力された Google PIN コードをサーバーに送り、Google アカウント情報と API アクセストークンを取得する
P8		Twitter 認証要求	サーバーから Twitter アカウントの認証用 URL を取得し、標準ブラウザーに表示する
P9		Twitter PIN 入力	Twitter PIN コードの入力を受け付ける
P10		Twitter 認証	入力された Twitter PIN コードをサーバーに送り、Twitter アカウント情報を取得する
P11		認証エラー	認証や情報取得の失敗時にエラーを通知する
P12		発表者情報保存	発表者の属性情報を保存する
P13	発表準備	発表情報入力	発表概要の入力を受け付ける
P14		スライド手動設定	スライド資料を URL 手動入力で設定する
P15		Google スライド一覧表示・設定	発表者の Google スライドに登録されているスライド資料の一覧を表示し、指定されたものを資料として設定する
P16		SlideShare スライド一覧表示・設定	任意の SlideShare アカウントに登録されているスライド資料の一覧を表示し、指定されたものを資料として設定する
P17		カメラ検出/設定	発表者 PC に接続された Web カメラの配信有無を設定する
P18		発表情報送信	入力された発表情報を WEB サーバーに送信する
P19	発表告知	視聴用 URL 取得	Web サーバーから視聴ページの URL を取得する
P20		視聴用 URL 表示	視聴ページの URL を表示する
P21		視聴用 URL 投稿	発表者の Twitter アカウントに視聴ページの URL を投稿する

P22	発表操作	スライド操作	スライド資料の表示ページを変更する
P23		ポインタ操作	ポインタでスライド資料の一点を指し示す
P24		デスクトップ配信切り替え	スライド操作とデスクトップ配信を切り替える
P25	発表配信	スライド操作配信	発表者のスライド操作を WEB サーバーに送信する
P26		ポインタ操作配信	発表者のポインタ操作を WEB サーバーに送信する
P27		Web カメラ配信	発表者 PC に接続された Web カメラ映像をメディアサーバーに送信する
P28		デスクトップ配信	発表者 PC のデスクトップキャプチャ映像をメディアサーバーに送信する
P29	進行管理	タイマー設定	発表の制限時刻を設定する
P30		タイマー通知	発表の残り時刻または経過時刻を表示する
P31	質疑応答	コメント受信	聴衆からのコメントを受信する
P32		コメント一覧表示	聴衆からの全コメントを新着順に一覧表示する
P33		スタンプ受信	聴衆からのスタンプを受信する
P34		スタンプ表示	聴衆からのスタンプをスライド周辺に表示する
P35		質問通知	聴衆からの質問コメントの総数を通知する
P36		質問一覧表示	聴衆からの質問コメントを時系列順に表示する
P37		コメントリンク表示	コメントへのリンクを関連するスライド資料の特定ページ・特定座標に表示する
P38	アンケート	新規作成	2〜3 択のアンケートを作成し、選択肢の入力を受け付ける
P39		送信	作成したアンケートをサーバーに送信する
P40		終了通知	アンケートの回答〆切を受け付け、サーバーに送信する
P41		結果表示	アンケートの集計結果をサーバーから受け取り表示する
P42	記録	ログ受信	Web サーバーからログデータを受信する
P43		ログ保存	ログデータを保存する
P44		ログ変換	ログデータを Markdown 形式に変換する
P45		Gist 投稿	ログデータ(Markdown)を発表者の Gist に投稿する

発表者用クライアントは、Meetup におけるプレゼンテーションの構成要素のうち発表者が発信する情報を処理し記録用 Web サーバーに送信する機能と、Web サーバーから受信した参加者の SNS 投稿を表示する機能を持つクライアントアプリケーションである。

発表者があらかじめスライド共有サイトに投稿しておいたスライド資料の URL を本アプリケーションに入力することで、即座にプレゼンテーションの配信と記録が可能になる。プ

レゼンテーションを開始するとそのプレゼンテーション固有の URL が生成される。ほかの視聴者がこの URL を Web ブラウザーで開けば、後述する聴衆用 Web アプリケーションですぐにプレゼンテーションへの参加が可能となる。

発表用クライアントの画面にはスライド資料と自身のビデオ映像、そして参加者の SNS 投稿が表示され。マウスやキーボードでスライドコントロールが可能である。

4.3.2.　発表者用クライアントアプリケーションの流れと画面

図 7 で、本システムの発表者用クライアントの画面遷移を示す。

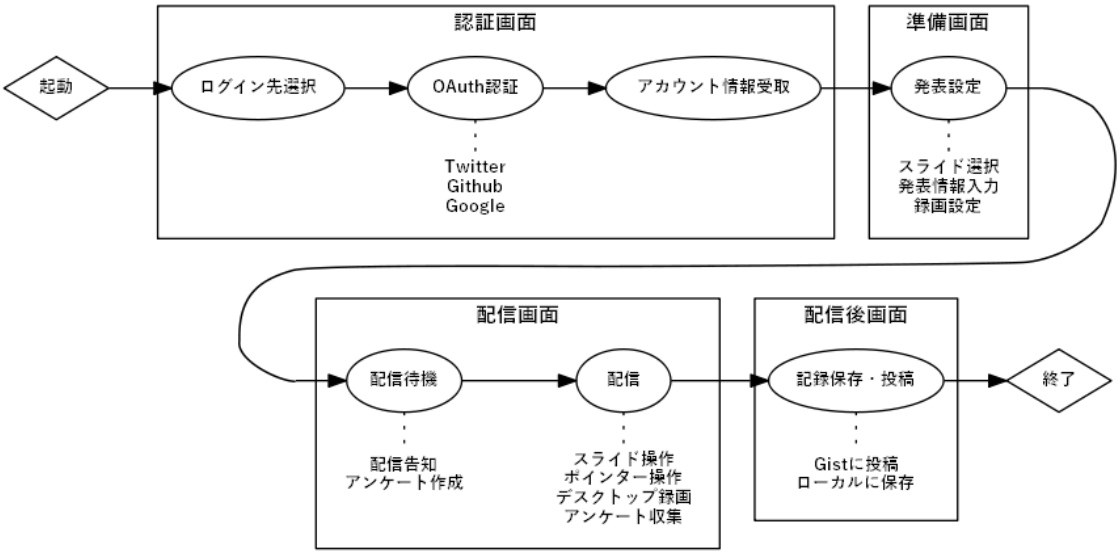

図 7　発表者用クライアントアプリケーションの画面遷移

本システムの発表者用クライアントは、認証画面、準備画面、配信画面、配信後画面の 4 つの画面によって構成される。本論文では、画面を画面レイアウトの単位とする。それぞれの画面は 1 つ以上のステップで構成され、0 個以上のビューを内包する。本論文において、ステップはユーザーインターフェース（入力可能なユーザー操作の組み合わせ）の単位とし、ビューはユーザー操作をきっかけに同一画面にオーバーレイ表示されるコンテンツの単位とする。

4.3.2.1.　認証画面

発表者のユーザー情報を入力する画面で、3 つのステップで構成され、3 つのビューが含まれる。図 8 に認証画面のレイアウトを示す。

番号	名称	機能/備考
1	ヘルプ	
2	Twitter 認証/情報	
3	GitHub 認証/情報	
4	Google 認証/情報	
5	プライバシーポリシー	
6	連携せずに準備画面へ	
7	準備画面へ	
8	PIN コード入力	
9	入力キャンセル	
10	入力決定	

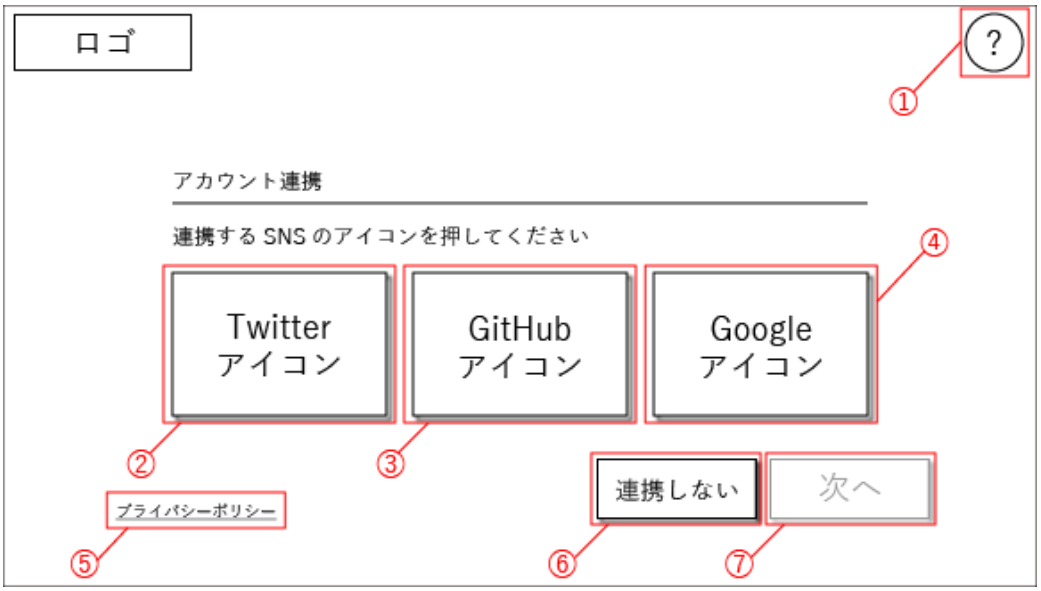

図 8　発表者用クライアント 認証画面レイアウト 1 (初期状態)

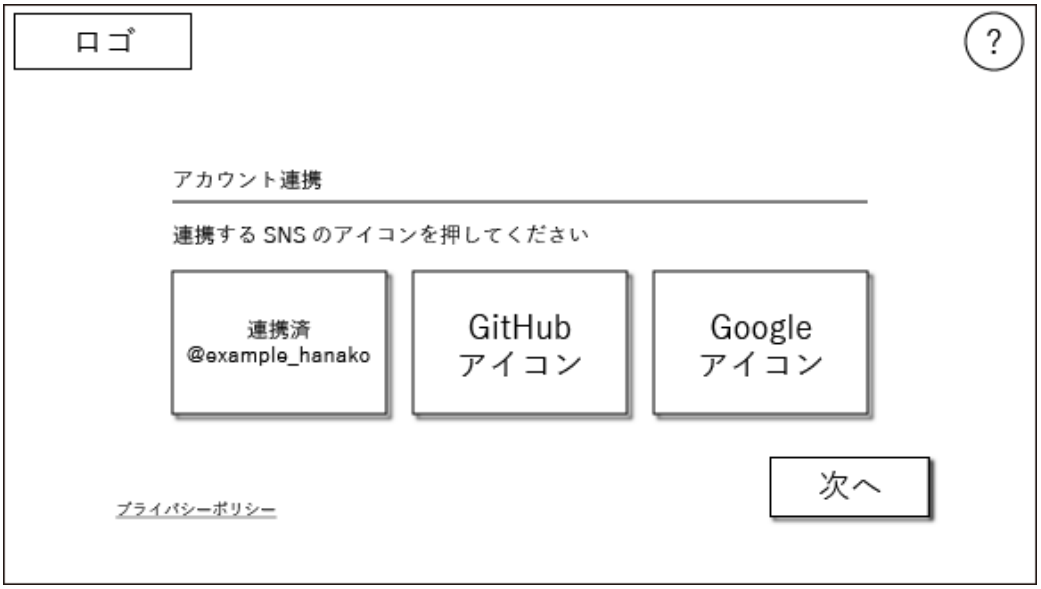

図 9　発表者用クライアント 認証画面レイアウト 1 (連携後)

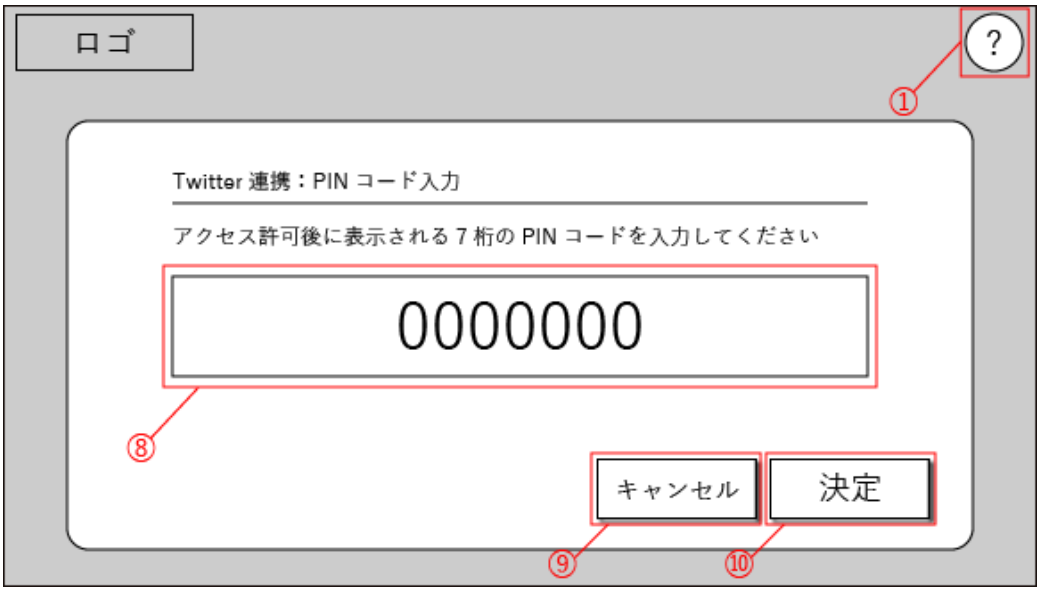

図 10　発表者用クライアント 認証画面レイアウト 2

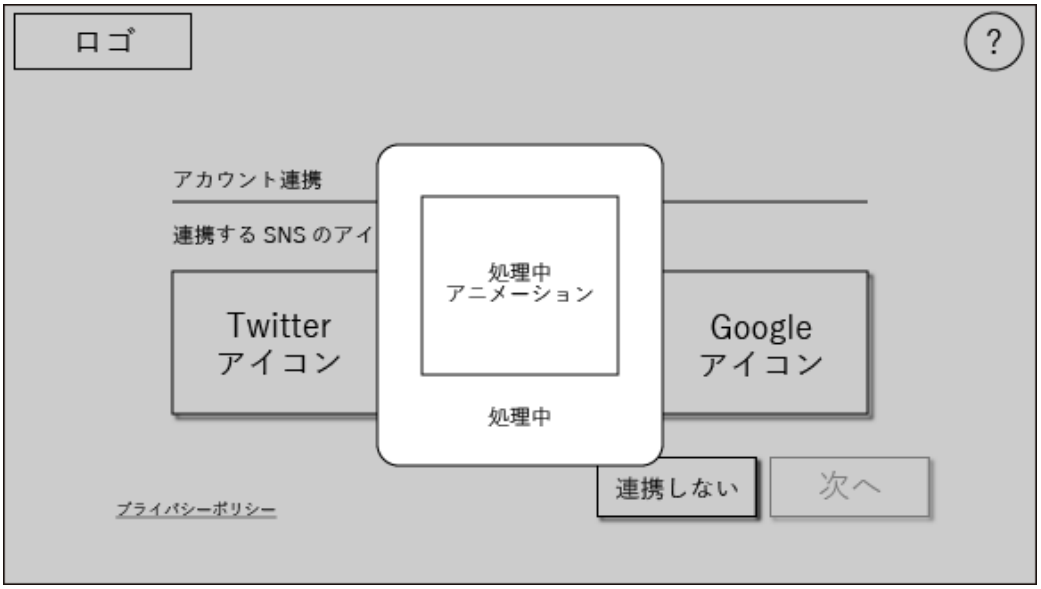

図 11　発表者用クライアント 認証画面レイアウト 3

4.3.2.2.　準備画面

(画面レイアウト)
発表者が配信のための準備をする画面で、3 つのビューが含まれる。

4.3.2.3.　配信画面

(画面レイアウト)
発表者が配信を行う画面で、2 つのステップで構成され、6 つのビューが含まれる。

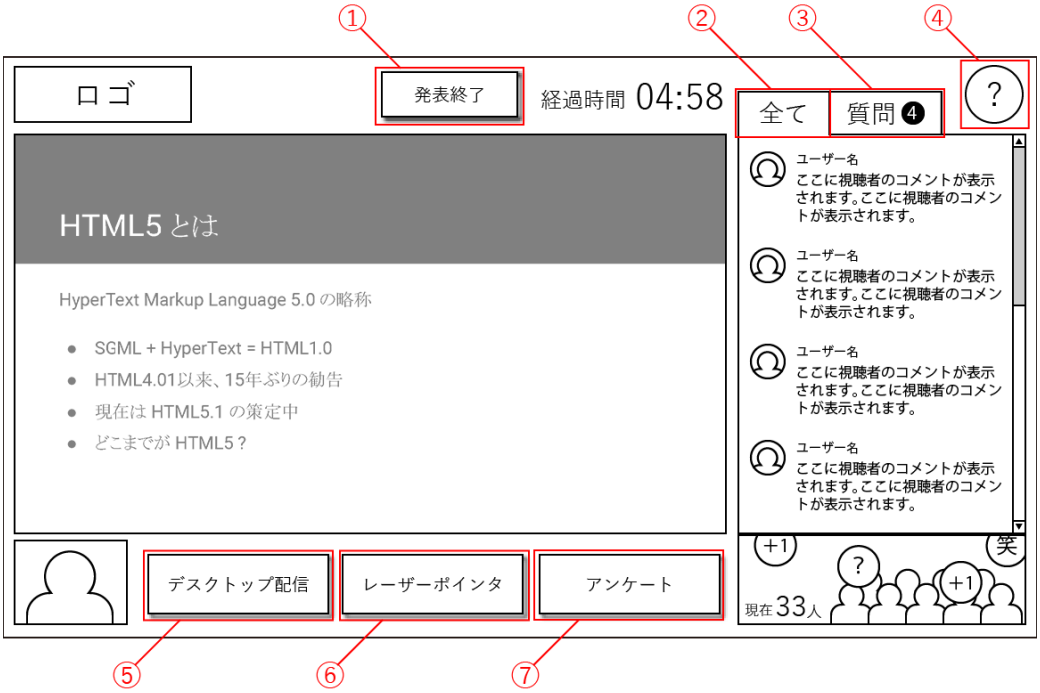

図 12　発表者クライアント 配信画面レイアウト 1

4.3.2.4.　配信後画面

(画面レイアウト)

発表者が配信の記録を保存や共有するための画面で、2 つのビューが含まれる。

ああああ

4.4.　聴衆用クライアントアプリケーションの設計

本節では、本システムのうち聴衆用クライアントアプリケーションの設計について述べる。

4.4.1.　聴衆用クライアントアプリケーションの振る舞い

本項では、本研究で実装する聴衆用クライアントアプリケーションの機能及び動作の概要を述べる。まず、表 6 で本アプリケーションの機能の一覧を示す。

表 6　聴衆用クライアントアプリケーションの機能一覧

No.	大機能	小機能	機能概要
A1	認証	聴衆情報取得	聴衆のアカウント情報を取得する
A2		Twitter 認証要求	サーバーから Twitter アカウントの認証用 URL を取得し、遷移する
A3		Twitter 認証	リダイレクトを解析して得たアクセストークンをサーバーに送り、Twitter アカウント情報を取得する
A4		認証エラー	Twitter 認証や情報取得に失敗したときにエラーを通知する
A5		発表者情報保存	発表者のセッション情報、Twitter アカウント情報を保存する
A6	入退室	入室	発表 URL に紐づいた配信への接続をサーバーに要求する
A7		入室エラー	配信に接続できないときにエラーを通知する
A8		退室	視聴画面を閉じたときにサーバーへ離脱を通知する
A9	視聴	発表情報取得	発表情報を取得する
A10		スライド資料表示	発表に紐づいたスライドデータを表示する
A11		スライド同期	発表者のページ更新通知をサーバーから受け取り同期する
A12		スライド別窓表示	スライドを別窓で、聴衆が任意に操作可能な状態で開く
A13	コメント	コメント表示	サーバーからコメント通知を受け取り同期・表示する
A14		コメント入力/投稿	入力されたコメントを表示中のスライドページまたはその特定座標(矩形)と紐づけてサーバーに送る
A15		スタンプ選択/投稿	選択されたスタンプを表示中のスライドページまたはその特定座標(矩形)と紐づけてサーバーに送る
A16		マイク録音/投稿	聴衆端末のマイク音声を録音し、表示中のスライドページまたはその特定座標(矩形)と紐づけてサーバーに送る
A17		スライド領域選択	スライドの特定座標(矩形)を選択する
A18	ポインタ	ポインタ同期	サーバーからポインタ通知を受け取り同期・表示する
A19	アンケート	アンケート表示	サーバーからアンケート通知を受け取り表示する
A20		アンケート受付	サーバーから〆切通知を受け取るまで、回答を受け付ける
A21		アンケート送信	アンケートの回答をサーバーに送信する
A22		アンケート結果表示	アンケート結果をグラフにして表示する
A23	映像	Web カメラ表示	サーバーから発表者の Web カメラ映像を受信し表示する
A24		デスクトップ表示	サーバーから発表者のデスクトップ映像を受信し表示する

聴衆用アプリケーションは、Meetup におけるプレゼンテーションの構成要素のうち聴衆の SNS 投稿を処理しサーバーに送信する機能と、発表者の発表情報と他の参加者の SNS 投稿をサーバーから受け取り表示する機能を持つアプリケーションである。

発表者から伝えられたプレゼンテーション URL を Web ブラウザーで開くと、この聴衆側アプリケーションで即座にプレゼンテーションに参加できる。聴衆用アプリケーションの画面にはスライド資料と発表者のビデオ映像、そして参加者の SNS 投稿が表示され、マウスやキーボード、タッチインターフェースで SNS 投稿やアンケートへの回答が可能である。

4.4.2.　聴衆用クライアントアプリケーションの流れと画面

図 13 で、本システムの聴衆用クライアントの画面遷移を示す。

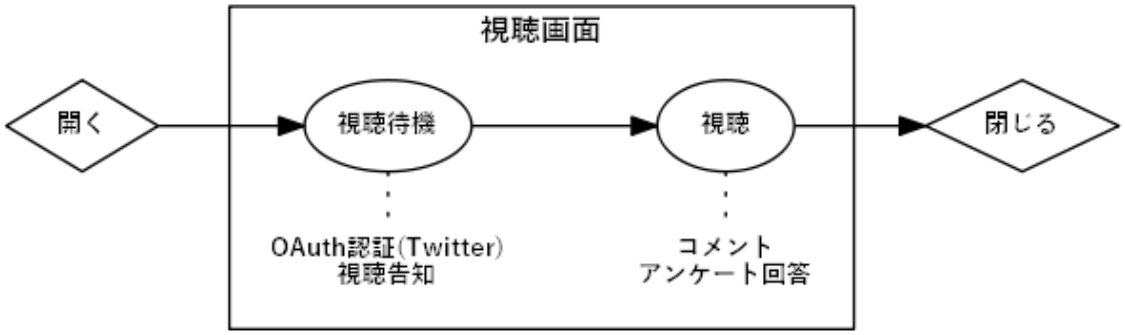

図 13　聴衆用クライアントアプリケーションの画面遷移

本システムの聴衆用クライアントは、視聴画面のみによって構成される。

4.4.2.1.　視聴画面

(画面レイアウト)
ああああ

4.5.　サーバーアプリケーションの設計

本節では、本システムのうちサーバーアプリケーションの設計について述べる。

4.5.1.　サーバーアプリケーションの振る舞い

本項では、本研究で実装するプレゼンテーション記録システムのサーバーアプリケーションの動作概要を述べる。まず、表 7 で本アプリケーションの機能の一覧を示す。

表 7　サーバーアプリケーションの機能一覧

No.	大機能	小機能	機能概要
S1	発表者認証	GitHub 認証開始	発表者クライアントに GitHub 認証用 URL を渡す
S2		GitHub 認証	発表者クライアントから GitHub PIN コードを受け取り、生成したアクセストークンで GitHub アカウント情報を取得し返す
S3		Google 認証開始	発表者クライアントに Google 認証用 URL を渡す
S4		Google 認証	発表者クライアントから Google PIN コードを受け取り、生成したアクセストークンで Google アカウント情報を取得し返す
S5		Twitter 認証開始	発表者クライアントに Twitter 認証用 URL を渡す
S6		Twitter 認証	発表者クライアントから Twitter PIN コードを受け取り、生成したアクセストークンで Twitter アカウント情報を取得し返す
S7	聴衆認証	Twitter 認証開始	聴衆クライアントに Twitter 認証用 URL を渡す
S8		Twitter 認証	聴衆クライアントからアクセストークンを受け取り､Twitter アカウント情報を取得し返す
S9	発表告知	Twitter 共有	各クライアントからの要求に応じて､発表タイトル及び URL を Twitter アカウントに投稿する
S10	発表管理	発表生成	発表者クライアントから発表情報を受け取り、一意の発表 URL を生成して返す
S11		発表接続	発表 URL に応じて、発表者クライアントと聴衆クライアントを接続する
S12		発表終了	発表者クライアントからの終了通知を受け取り、関連する接続の切断と発表 URL の無効化を行う
S13	発表配信	基本情報同期	接続した聴衆クライアントに発表情報を送信する
S14		スライド同期	発表者クライアントからのページめくり信号を関連する全聴衆クライアントにブロードキャストする
S15		ポインタ同期	発表者クライアントからのポインタ信号を関連する全聴衆クライアントにブロードキャストする
S16		映像配信	発表者クライアントからのリアルタイム配信映像を、関連する全聴衆クライアントにブロードキャストする

S17		映像切り替え同期	発表者クライアントからの映像切り替え信号を関連する全聴衆クライアントにブロードキャストする
S18	アンケート	アンケート開始	発表者クライアントからのアンケート要求とその内容を関連する全聴衆クライアントにブロードキャストする
S19		アンケート終了	アンケートの締め切り通知を関連する全聴衆クライアントにブロードキャストする
S20		アンケート回収	聴衆クライアントからのアンケート回答を受け取り、集計する
S21		アンケート集計	締め切り時点での集計結果を関連する全てのクライアントにブロードキャストする
S22	コメント	コメント同期	聴衆クライアントからコメントを受け取り、関連する発表者クライアントと他の聴衆クライアントにブロードキャストする
S23		コメント投稿	Twitter 認証された聴衆クライアントからコメントを受け取り、聴衆の Twitter タイムラインにコメントを投稿する
S24		音声コメント	聴衆クライアントから録音を受け取り、関連する発表者クライアントと他の聴衆クライアントにブロードキャストする
S25	発表記録	コメント記録	同期した全てのコメントデータを記録する
S26		映像記録	配信した全ての映像データを記録する
S27		スライド記録	スライド資料の URL 及び操作情報を記録する
S28		ログ作成	記録されたコメントデータとスライドデータを元にプレゼンテーション記録フォーマットでログを作成し、発表者クライアントに送信する
S29		Gist 投稿	発表者クライアントからの要求に応じて、ログからレポートを作成して発表者の Gist に投稿する
S30		ログ URL 通知	発表者クライアントにログ及び映像データをダウンロードできる URL を通知する

サーバーアプリケーションは、本システムにおいてクライアント間の情報の受け渡しとその記録を担う。Meetup における全てのプレゼンテーション構成要素を発表者用・聴衆用クライアントから受け取り、サーバー内に記録したうえで参加者全員に同じ情報をブロードキャストして伝える。

4.6.　プレゼンテーション記録フォーマットの設計

本節では、本システムが記録するプレゼンテーションデータのフォーマット設計につい

て述べる。技術系 Meetup におけるプレゼンテーションの記録データはオンラインコミュニティでの利用が想定されるため、WEB と親和性の高い JSON をコンテナフォーマットとして採用する。

4.6.1.　プレゼンテーション記録フォーマットの属性一覧

本項では、JSON をコンテナとする記録フォーマットの設計について述べる。各属性の詳細な構造を表 8 に示す。この記録フォーマットでの記録例を図 14 に示す。記録データには、プレゼンテーションのタイトルや時刻などの基本情報に加え、発表者情報、聴衆のリスト、スライド資料と各ページの URL、スライド資料以外の添付資料の URL、そして時系列順に整理された発表中のアクティビティのリストが含まれる。

表 8　プレゼンテーション記録フォーマット(JSON)の属性詳細

属性名	型	親要素	必須	説明/備考
title	string	-	Y	発表者が指定した発表タイトル
description	string	-	N	発表者が指定した発表の説明文
time_begin	integer	-	Y	プレゼンテーションの開始時刻(ミリ秒単位の UNIX 時刻)
time_end	integer	-	Y	プレゼンテーションの終了時刻(ミリ秒単位の UNIX 時刻)
presenter	object	-	Y	発表者情報を格納するオブジェクト
name	string	presenter	Y	発表者の氏名またはニックネーム
twitter	string	presenter	N	発表者の Twitter スクリーンネーム
google	string	presenter	N	発表者の Google ID
github	string	presenter	N	発表者の GitHub ID
audience	array<object>	-	Y	聴衆情報を格納する配列。index は聴衆の ID と同義
name	string	audience	N	聴衆の氏名またはニックネーム
twitter	string	audience	N	聴衆の Twitter スクリーンネーム
slide	array<string>	-	Y	スライド資料とそのページごとの URL を格納する配列。index はページ番号と同義。ページごとに固有の URL が割り振られていないスライド資料の場合は要素が重複する場合がある
attachment	array<object>	-	N	スライド資料以外の添付資料を格納する配列。index は添付資料の ID と同義

type	string	attachment	Y	添付資料の MIME Type
url	string	attachment	Y	添付資料の参照先 URL
name	string	attachment	N	添付資料の名称
activity	array<object>	-	Y	プレゼンテーション内のイベントを時系列順に格納する配列。index は要素の ID と同義
time	integer	activity	Y	イベントの発生時刻（ミリ秒単位の UNIX 時刻）
type	string	activity	Y	イベントの種類。begin, end, comment 等がある
page	integer	activity	N	イベントと紐づいたページ(slide 配列の index)
area	object	activity	N	ページの特定座標・領域を示す場合に用いる
x	integer	area	Y	スライド資料の横幅を 1.0 としたときの X 座標
y	integer	area	Y	スライド資料の縦幅を 1.0 としたときの Y 座標
width	integer	area	N	スライド資料の横幅を 1.0 としたときの領域幅
height	integer	area	N	スライド資料の縦幅を 1.0 としたときの領域高さ
text	string	activity	N	聴衆のコメント
url	string	activity	N	イベントに紐づいた URL
audience_id	integer	activity	N	イベントに紐づいた聴衆の ID。0 以上
attachment_id	integer	activity	N	イベントに紐づいた添付資料の ID。0 以上
activity_id	integer	activity		イベントに紐づいた他のイベントの ID。0 以上
question	object	-	N	アンケート情報
type	string	question	Y	アンケートの形式。single または multiple
text	string	question	Y	アンケートの質問文
option	array<object>	question	Y	アンケートの選択肢を格納する配列。2 要素以上
text	string	option	Y	アンケート選択肢の文章
number	integer	option	Y	アンケート選択肢の選択数。0 以上

```
{
        "title": "プレゼンテーションのタイトル",
        "description": "このプレゼンテーションはサンプルです。",
        "time_begin": 14518146780000,
        "time_end": 14518163010000,
        "presenter": {
                "name": "山田太郎",
                "twitter": "@example_yamada",
                "google": null,
                "github": "exampleyamada"
        },
        "audience": [{
                "name": "竹田花子",
                "tiwtter": "@example_hanako "
        }, {
                "name": "たまちゃん",
                "tiwtter": "@example_azarashi"
        }],
        "slide": [
                "http://example.com/slide.html",
                "http://example.com/slide.html#2",
                "http://example.com/slide.html#3"
        ],
        "attachment": [{
                "type": "video/webm",
                "url": "http://example.com/recerd.webm",
                "name": "録画映像"
        }],
        "activity": [{
                "time": 14518148620000,
                "type": "slide_update",
                "page ": 1
        }, {
                "time": 14518149240000,
                "type": "comment",
                "page": 1,
                "area": null,
                "text": "よろしくー",
                "audience_id": 0,
                "url": "http://twitter.com/status/9999999999"
        }, {
                                        中略
        }, {
                "time": 14518182720000,
                "type": "finish"
        }]
}
```

図 14　プレゼンテーション記録フォーマット(JSON)の例

4.6.2.　記録データの利用方法

本項では記録データの利用方法について述べる。この記録フォーマットを Markdown に変換しての利用例を図 15 で示す。記録データには発表概要と参加者情報、そして参加者のスライドページごとのコメント一覧が含まれる。簡易レポートの形式をとるため情報量よりも読みやすさを重視した形で記述する。参照情報のあるものはハイパーリンクを施す。

```
# プレゼンテーションのタイトル

## 発表概要
* 発表時刻：2016/01/02 11:32 〜 11:40
* [スライド資料](http://example.com/slide.html)
* [添付資料 1 - 録画映像](http://example.com/recerd.webm)

## 参加者
### 発表者
山田太郎 ([Twitter](https://twitter.com/example_yamada),
[GitHub](https://github.com/exampleyamada))
### 聴講者
* [竹田花子](https://twitter.com/example_hanako)
* [たまちゃん](https://twitter.com/example_azarashi)
ほか 4 名

## コメントログ
* [2 ページ目](http://example.com/slide.html#2)
  * [よろしくー (竹田花子)](https://twitter.com/exmple_hanako /status/9999999999)
                                後略
```

図 15　プレゼンテーション記録フォーマット(Markdown)の例

第5章　技術系 Meetup におけるプレゼンテーション記録システムの実装

本章では、前章で設計したプレゼンテーション記録システムの実装について述べる。

5.1.　実装環境

本研究で実装した Meetup におけるプレゼンテーション記録システムの実装環境を表 9 に示す。設計段階ではサーバーは 1 台であったが、メディアサーバーによる映像配信の処理が重くなり HTTP サーバーや WebSocket サーバーの動作が不安定になることが多々あったため、これらを物理的に分割し 2 台構成とした。

表 9　本研究のプレゼンテーション記録システムの実装環境

名称	詳細
WEB/WebSocket サーバーOS	Ubuntu 14.04.3 LTS
WEB サーバー	Apache 2.4.7
WebSocket サーバー	Node.js 5.2.0 + ws 0.8.1
サーバーアプリケーション使用言語	Haxe (Node.js)
メディアサーバーOS	Ubuntu 14.04.3 LTS
メディアサーバー	Kurento Media Server 6.2.0
ネイティブクライアントエンジン	Electron 0.35.3
ネイティブクライアント使用言語	Haxe (JavaScript), HTML, CSS
WEB クライアント使用言語	Haxe (JavaScript), HTML, CSS

5.2.　実装の前提となる技術

5.2.1.　Haxe

Haxe [4]は静的型付きのオブジェクト指向プログラミング言語である。Flash コミュニティを出自とする言語のため、ActionScript と似た構文を持つ。Haxe コンパイラを通して JavaScript や ActionScript、C++、C#、Java、PHP、Python コードを出力することが可能なマルチパラダイム言語で、その出力先の多さからマルチプラットフォーム開発を目的として使用されることが多い。本システムでは altJS（JavaScript および Node.js の代替言語）として利用する。

5.2.2.　WebSocket (RFC5455)

WebSocket [5]は、WEB アプリケーションのための TCP 上に構築された双方向通信プロトコルである。IETF によって RFC5455 として策定・標準化されている。従来のクライアント・サーバー間のリアルタイム通信に用いられていた XMLHttpRequest の欠点を解決する技術として開発されたため、より軽量かつ多機能で、ロングポーリング等を行わずともサーバー側からのプッシュ配信を実現できる。専用のスキームとして ws および wss が割り当てられている。なお、WebSocket API [6]は W3C によって策定されている。本システムでは、アプリケーション間のリアルタイム通信と、WebRTC のシグナリングのために WebSocket を用いる。

5.2.3.　WebRTC

WebRTC [7]は、W3C が提唱する Web ブラウザー用のリアルタイムコミュニケーション API で、特別なプラグインなしでサーバーレスのピアツーピアの映像伝送を可能とする（ただし最初の接続時は伝送経路を決めるために WebSocket 等を用いてシグナリングを行う必要がある）。映像や音声のリアルタイム伝送のためのメディアチャンネルと、欠落のないバイナリデータ伝送のためのデータチャンネルの 2 種類の通信方式がある。メディアチャンネルは DTLS の助けを借りた SRTP over UDP プロトコルを用いて映像や音声を暗号化したうえでリアルタイム伝送する。データチャンネルは SCTP over DTLS over UDP プロトコルを用いてバイナリデータを暗号化したうえで伝送する。本システムでは、発表者映像のリアルタイム配信と記録のために WebRTC を用いる。

5.2.4.　Kurento Media Server

Kurento Media Server [8]は WebRTC を用いたグループコミュニケーションのためのメ

ディアサーバーである。Kurento Technologies によりオープンソースで開発されている。Kurento を用いることで、ピアツーピア特有のフルメッシュで複雑な構造を回避し、サーバーに繋ぐだけで全てのクライアントと通信ができるようになる。本システムでは発表者クライアントと無数の聴衆クライアントを効率的につなぐための WebRTC SFU (Selective Forwarding Unit)として用いる。

5.2.5.　Electron (旧称 Atom Shell)

Electron [9]は、GitHub 社が開発するデスクトップアプリのためのクロスプラットフォーム実行環境である。Chromium ブラウザーを内蔵し、Node.js/HTML で記述したアプリケーションを Windows、Max OS X、Linux 上で動かせる。本システムでは発表者用クライアントにおいて WEB アプリケーションのクロスドメイン制約を回避するために用いる。

5.2.6.　Let's Encrypt

Let's Encrypt [10]は、非営利団体 Internet Security Research Group が運営する無料ドメイン認証 TLS サーバー証明書プロジェクト、並びにその発行サービスである。本システムでは、Chromium 系ブラウザーにおいて getUserMedia メソッドが HTTPS 環境でしか動作しない仕様に対応するために用いる。

5.3.　実装の概要

本節では、本研究で実装したプレゼンテーション記録システムの実装の概要を述べる。

実装の概要を図 16 に示す。本システムの実装は、発表者が使用するネイティブクライアントと、聴衆が使用する Web クライアント、それらから発信される情報を受け取り記録・他の参加者に仲介する Web サーバー・Kurento メディアサーバー、ならびに各アプリケーションが協調して動作するための通信プロトコルに大別される。

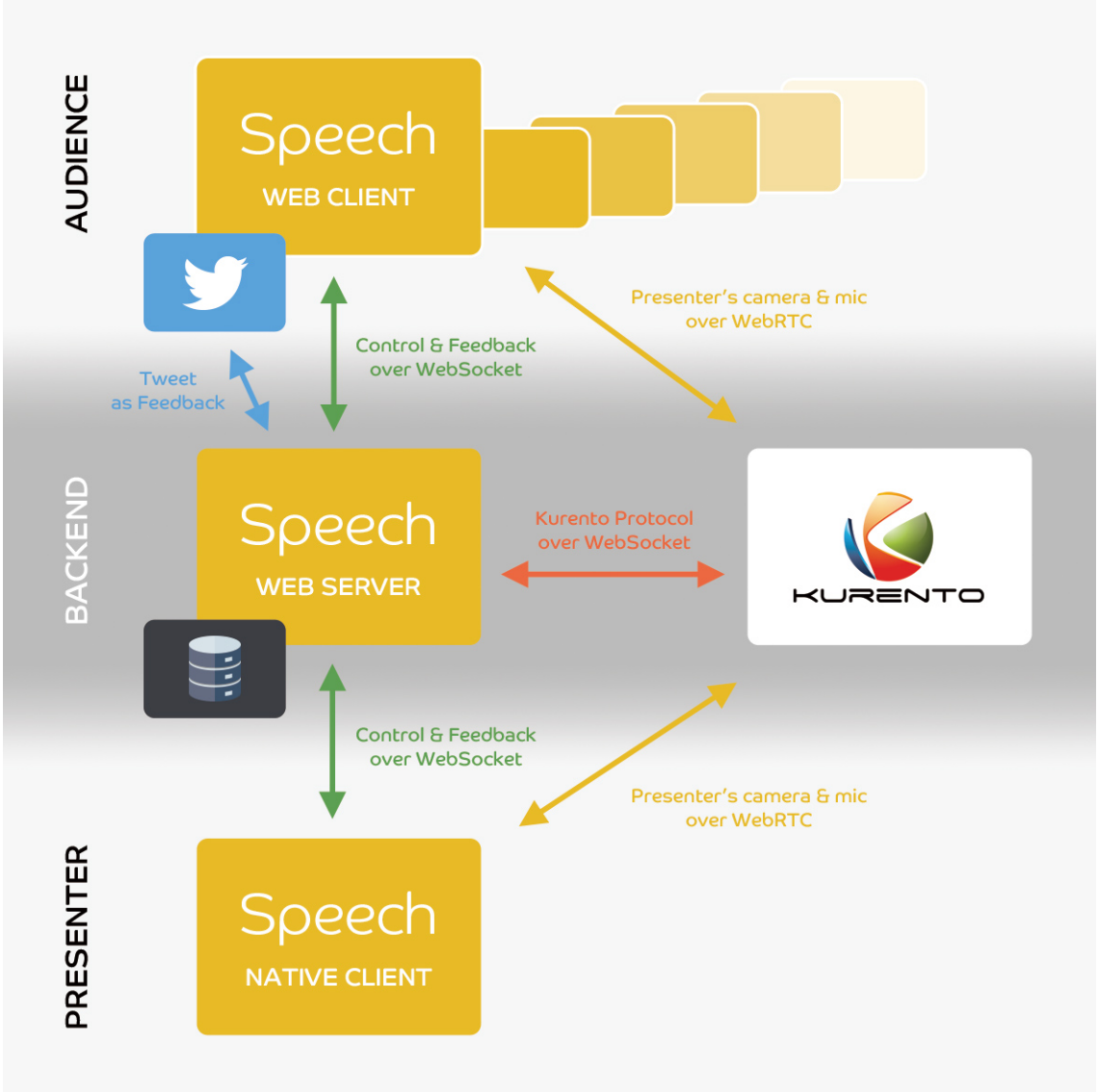

図 16　Meetup におけるプレゼンテーション記録システムの実装概要

5.3.1.　発表者用ネイティブクライアントの実装

発表者用クライアントアプリケーションは、プログラマーコミュニティができる限り利用しやすいよう WEB アプリケーションとしたかったが、iframe タグなどのクロスドメイン制限でスライドのページ送り等をシステムで検知することが難しかったため、Electron を用いてネイティブクライアントとして実装した。Electron は、WEB アプリケーションをデスクトップアプリケーションとしてパッケージングできるアプリケーションフレームワークである。Electron には iframe のクロスドメイン制限を無効とした webview という独自

のタグがあり、今回はこれを用いて WEB 上にあるスライド資料の表示と操作を実現した。このネイティブクライアントは別途構築したインストーラーで Windows、Mac OSX にインストールできる。

表 5 に挙げた機能のうち、実証実験に必要な最低限の機能（P13、P14、P17、P18、P19、P22、P25、P27、P30、P31、P32、P33、P34、P35、P36）を実装した。

5.3.2.　聴衆用 WEB クライアントの実装

聴衆用クライアントアプリケーションは、プログラマーコミュニティが利用しやすいよう WEB アプリケーションとした。HTML や CSS のコードは発表者用クライアントとできる限り共通化することで、見た目の差異を小さくし、また実装コストも減らした。利用を想定するブラウザーは、Chrome M48 以降、ならびに Firefox 43 以降とした。

表 6 に挙げた機能のうち、実証実験に必要な最低限の機能（A1、A6、A7、A8、A9、A10、A11、A13、A14、A15、A23）を実装した。

5.3.3.　サーバーの実装

サーバーアプリケーションは、WEB/WebSocket サーバーと、メディアサーバーに分けられる。両者ともに Ubuntu OS 上で動作するよう実装したが、処理負荷の都合から前者と後者は別のサーバーコンピューター上で稼働させた。WEB サーバーは Apache で、WebSocket サーバーは Node.js/ws で、メディアサーバーは Kurento Media Server を用いて構築した。

メディアサーバーは、全てのクライアントと接続し、発表者用ネイティブクライアントから配信された映像を記録しながら各聴衆用 WEB クライアントに中継する役割を担う。WebRTC は本来 P2P 通信なので発表者と聴衆を直接接続すればメディアサーバーは不要であるが、映像を記録したり、発表者用ネイティブクライアントにかかる負荷を減らしたりする必要があったために、今回 WebRTC SFU [11]として Kurento Media Server を導入した。SFU とは選択的配信ユニット

表 7 に挙げた機能のうち、実証実験に必要な最低限の機能（S10、S11、S12、S13、S14、S16、S22、S23、S25、S26、S27、S28）を実装した。

5.3.4.　JSON over WebSocket による RPC の実装

各アプリケーションが協調して動作するための RPC（遠隔手続き呼び出し）を実装した。表 10 に遠隔手続きの一覧を示す。この RPC は WebSocket 上で JSON フォーマットを用

いてやり取りされる。

表 10　実装した RPC

呼び出し名	方向	備考
ICE_CANDIDATE	Server <--> Client	WebRTC 経路情報の交換
COMMENT	Server <--> Client	コメントの投稿
UPDATE_SLIDE	Server <--> Client	スライドページの更新
ON_ACCEPT_STREAM	Server -> Client	メディアサーバーへの接続完了
ON_UPDATE_AUDIENCE	Server -> Client	参加人数の更新
ON_ERROR	Server -> Client	プレゼンテーションの異常終了
ON_FINISH	Server -> Client	プレゼンテーションの正常終了
WILL_STOP_STREAM	Server -> Client	映像配信の終了予告
ON_STOP_STREAM	Server -> Client	映像配信の終了
ON_ACCEPT_PRESENTER	Server -> Client(Native)	発表者の接続承認
ON_CREATE_LOG	Server -> Client(Native)	ログの生成完了
ON_ACCEPT_AUDIENCE	Server -> Client(WEB)	聴衆の接続承認
CAN_CONNECT_STREAM	Server -> Client(WEB)	映像配信の存在
JOIN_PRESENTER	Client(Native) -> Server	発表者の接続要求
LEAVE_PRESENTER	Client(Native) -> Server	発表者の切断要求
START_STREAM	Client(Native) -> Server	映像配信の開始要求
STOP_STREAM	Client(Native) -> Server	映像配信の終了要求
REQUEST_LOG	Client(Native) -> Server	ログの生成要求
JOIN_VIEWER	Client(WEB) -> Server	聴衆の接続要求
LEAVE_VIEWER	Client(WEB) -> Server	聴衆の切断要求
CONNECT_STREAM	Client(WEB) -> Server	映像配信への接続要求
DISCONNECT_STREAM	Client(WEB) -> Server	映像配信からの切断要求

第6章　実証実験

本章では、実装したシステムを用いて行った実証実験について述べる。

6.1.　実証実験の概要

村井純研究室 Arch グループの有志で行っている「モダンオペレーティングシステム輪講会」において、本システムを用いたプレゼンテーション記録の実証実験を行った。村井純研究室 Arch グループは、通常 GitHub や Slack 等の SNS を用いて交流し、週に 1 度のオフラインミーティング並びに輪講会においてより密な情報共有を行っているという意味において、プログラマーコミュニティと技術系 Meetup と同様の構図であることから、今回の実証実験の舞台に相応しいと判断した。

発表の構図を図 17 に示す。2016 年 1 月 14 日(木)に行われたモダンオペレーティングシステム輪講会は、村井純研究室学部生部屋を会場に、1 人の発表者が書籍「モダンオペレーティング 第 2 版」の一部内容を、スライド資料を用いて紹介・解説した。発表者はあらかじめスライド資料を Google スライドで共有し、本システムの発表者用クライアントを用いて発表した。また発表者のコンピューターは外部ディスプレイに接続し、発表画面をメイン会場全体から見えるようにした。聴衆は、各自のコンピューターから本システムの聴衆用クライアントを用いて発表に参加した。聴衆のうち 1 人は、遠隔環境を再現するために別室に隔離した。

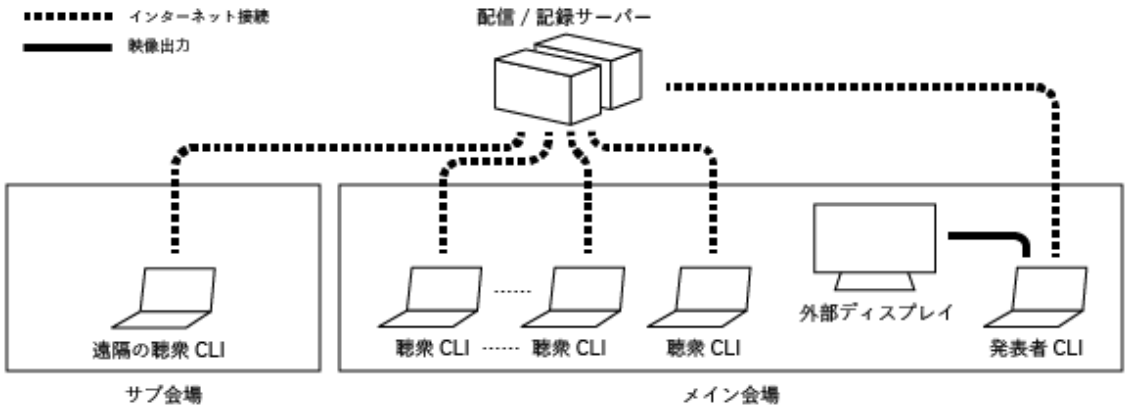

図 17　実証実験におけるプレゼンテーションの構図

6.2.　プレゼンテーション記録結果

プレゼンテーション内で行われた言動と、そのうち記録できたもの

表 11　本システムによって記録された要素

記録対象	詳細	数
発表タイトル	タイトル	1
発表開始	時刻	1
発表終了	時刻	1
発表者属性	名前、IP アドレス	1
聴衆属性	名前、IP アドレス、入室順序	22
発表スライド	ページ数、ページ順序、ページごとの URL	44
発表者映像	MIME タイプ、保存先 URL	1
入退室	時刻、聴衆番号	29
コメント	時刻、コメント種別、テキスト、ページ番号、聴衆番号	59
ページ送り	時刻、ページ番号	84

大分類	小分類	回数
発表		
質問		
感想/意見		
補足		
その他		

第7章　評価

本章では、本研究で設計・実装したプレゼンテーション記録システムの評価を行う。

7.1. 評価概要

- 映像配信時の参加者のスケーラビリティ
  WebRTC を用いて映像記録・配信を行った際に安定して動作する同時参加者数の上限値。

- 対応するスライド共有サイトの数
  プレゼンテーションを行う際に、スライド資料参照先として利用可能なスライド共有サイトの数。

7.2. 計測

7.2.1. 映像配信時の参加者のスケーラビリティ

15 秒間に一人ずつ参加者が増える環境を構築し、新規参加者が映像を受信できなくなったときの同時参加者数を計測した。これを 20 回繰り返し、その平均値を結果とした。計測の結果を図に示す。

7.2.2. 対応するスライド共有サイトの数

著名なスライド共有サイトならびに個人サーバー上に置いた異なるスライドデータの URL を発表者用アプリケーションに入力し、実際に配信が可能だったか否かを記録した。計測の結果を表に示す。

7.3. 考察

第8章　結論

本章では、結論として本研究の成果を明らかにするとともに、今後の課題について述べる。

8.1.　まとめ

本研究では、技術系 Meetup におけるプレゼンテーションの記録を目的とした。その問題解決のために、技術系 Meetup におけるプレゼンテーション記録システムを設計し、実装した。

技術系 Meetup におけるプレゼンテーションは、オンラインのプログラマーコミュニティにとって唯一のオフラインイベントである。こうした環境を踏まえ、本研究ではプレゼンテーションの構成要素を配信しながら記録する WEB システムを実装して、実証実験を行った。実証実験では、研究室で行われた輪講会において記録の実証実験を行った。

実証実験の結果より、本研究で構築したプレゼンテーション記録システムでプレゼンテーションが正確に記録されることを確認し、本研究のアプローチが有効であることが確認された。

8.2.　今後の課題

実証実験の結果から、本研究のアプローチは技術系 Meetup におけるプレゼンテーションの記録に有効であることが明らかになった。しかし実証実験で構築した環境には幾つかの問題が見受けられた。以下に今後の課題を述べる。

- スライド資料が外部参照であるため、単体では完全な記録とならない
- コミュニティに顔出ししたくない発表者を考慮できていない

謝辞

参考文献

[1] Build Insider, “なぜ勉強会・カンファレンスに参加するのか？ 良かった勉強会・残念だった勉強会 - Build Insider,” 21 12 2015. [オンライン]. Available: http://www.buildinsider.net/hub/survey/201511-engineerreal. [アクセス日: 5 1 2016].

[2] Twitter, Inc., “ツイートが消えてしまった場合 | Twitter ヘルプセンター,” 28 9 2015. [オンライン]. Available: https://support.twitter.com/articles/277716. [アクセス日: 5 1 2016].

[3] “Web-based slideshow - Wikipedia, the free encyclopedia,” 25 7 2015. [オンライン]. Available: https://en.wikipedia.org/wiki/Web-based_slideshow. [アクセス日: 5 1 2016].

[4] Haxe Foundation, “Haxe - The Cross-platform Toolkit,” 17 1 2016. [オンライン]. Available: http://haxe.org/. [アクセス日: 17 1 2016].

[5] Internet Engineering Task Force, “RFC 6455 - The WebSocket Protocol,” 9 12 2015. [オンライン]. Available: https://tools.ietf.org/html/rfc6455. [アクセス日: 17 1 2016].

[6] W3C, “The WebSocket API,” 24 10 2015. [オンライン]. Available: https://www.w3.org/TR/websockets/. [アクセス日: 17 1 2016].

[7] Google Chrome team, “WebRTC Home | WebRTC,” 14 1 2016. [オンライン]. Available: https://webrtc.org/. [アクセス日: 17 1 2016].

[8] Technologies, Kurento, “Kurento,” 6 10 2015. [オンライン]. Available: http://www.kurento.org/. [アクセス日: 17 1 2016].

[9] GitHub, Inc., “Electron,” 16 1 2016. [オンライン]. Available: http://electron.atom.io/. [アクセス日: 17 1 2016].

[10] Internet Security Research Group, “Let's Encrypt - Free SSL/TLS Certificates,” 13 1 2016. [オンライン]. Available: https://letsencrypt.org/. [アクセス日: 17 1 2016].

[11] 岩. 義昌, “WebRTC におけるサーバーソリューションの決め手とは？— WebRTC Conference Japan 基調講演 | HTML5Experts.jp,” 9 2 2015. [オンライン]. Available: https://html5experts.jp/iwase/12585/. [アクセス日: 17 1 2016].

[12] “Meetup とは,” 19 12 2015. [オンライン]. Available:

http://help.meetup.com/customer/ja/portal/articles/637187-meetup%E3%81%A8%E3%81%AF.

[13] 高木義和, “情報文化 VIII. インターネットコミュニティの特性,” 28 5 2015. [オンライン]. Available: http://www.nuis.ac.jp/~takagi/JB/2015/ic08.pdf. [アクセス日: 20 12 2015].

[14] T. NGO, “OFFICE OPEN XML OVERVIEW,” 23 10 2006. [オンライン]. Available: http://www.ecma-international.org/news/TC45_current_work/OpenXML%20White%20Paper.pdf. [アクセス日: 21 12 2015].

[15] 総務省, “デジタルアーカイブの構築・連携のためのガイドライン,” 26 3 2012. [オンライン]. Available: http://www.soumu.go.jp/main_content/000153595.pdf. [アクセス日: 20 12 2015].

[16] 市川裕康, “『ソーシャル&リアルがポイント。今求められる新しい出会い、学び、コミュニティの形〜「ミートアップ」とは』 | ソーシャルビジネス最前線 | 現代ビジネス [講談社],” 2 5 2011. [オンライン]. Available: http://gendai.ismedia.jp/articles/-/3422. [アクセス日: 21 12 2015].

[17] 一. 三井, 誠. 内田 , 晋. 白山, “SNS における情報伝播に対するコミュニティの影響,” 横断型基幹科学技術研究団体連合, 2005.

[18] H. Rheingold, The Virtual Community: Homesteading on the Electronic Frontier, Cambridge, Massachusetts: The MIT Press, 2000.