WebSocket 通信によるブラウザ間協調動作プラットフォームの開発

小玉祥平† 市川みさ希† 太田高志† 羽田久一†

コンサートやスポーツ・イベントにおいて観衆の一体感を生成するシステムが広まりつつある．本論文では，事前の準備を必要とせずに複数の端末上のブラウザを一元的にコントロールできるシステムを提案する．本システムは WebSocket と HTML５によって構成されており，ブラウザを用いて特定の URL にアクセスするのみでデバイスを表示やコントロールに利用することが可能となる．本システムは参加型のライブパフォーマンスとして利用できる．

Multiple Browser Collaboration Platform Based on WebSocket Communications

SHOHEI KODAMA† MISAKI ICHIKAWA†
TAKASHI OHTA† HISAKAZU HADA†

Recently, a system to create an atmosphere of unity with audiences is spreading into events with large crowds such as music concert and sport games. In this paper, we propose a multiple browser collaboration platform based on HTML5 and WebSockets. This platform is a system to centrally control browsers on multiple devices without special preparations. Participants will only need to access certain URL to use their device for participation. This platform has made possible for people to easily participate in a live performance through their smartphones.

1. はじめに

近年, PC やスマートフォンの画面をマルチディスプレイ化し，協調動作させる例が多く見られるようになった．

その代表的な研究として，Pin-ch [1]と呼ばれる複数のスマートフォン上で実行しているアプリケーションの連携を動的に設定するインターフェイスの研究や，仮想ディスプレイにする端末をカメラで撮影し，ランダムに配置されていても位置特定を可能にする研究である Junkyard Jumbotron[2]などが挙げられる．しかしながら，それらの多くはネイティブアプリケーションとして提供されており，即時参加型のイベント時には向いていない．

また，国内では au のコマーシャル[3][4]であったり，ももいろクローバーZ や水樹奈々，YUKI などのライブで用いられているペンライトの FreFlow(フリフラ) [5]であったりなど，一方国外でもディズニーカリフォルニアでのショーの音楽と LED が同期するマウスの耳が着いた帽子グッズ[6]や，音楽と同期して正確なタイミングで消灯・点灯・色の変化が行える LED を内蔵したリストバンドの Xylobands[7]など，観客達が一体となって参加できるようなイベントが世界で注目されている．

そこで我々は,「参加できるマルチディスプレイコンテンツ」をコンセプトに，ブラウザ上で複数のスマートフォンの画面を統合して利用できるコンテンツプラットフォームを開発した．本システムではアプリケーションをインストールせずにブラウザ上で動作させることで，気軽に参加できる環境を目指した．

本システムは，ブラウザとの通信に WebSocket[8]を用いており，クライアントのブラウザが WebSocket に対応していれば, ブラウザで URL にアクセスするのみで実行できる．画面表示においても HTML5 で標準的にサポートされている機能のみを用いている．そのため，異なる OS で実行できる汎用性と，手持ちの端末で誰でも実行できる拡張性を併せ持っている．

本論文の構成は以下のとおりである．まず次章で関連研究として複数のディスプレイを統合的に取り扱うシステムについて述べ，3 章では本研究で提案するブラウザを用いた複数端末間での協調動作について述べる．４章ではスマートフォンをターゲットにした複数端末での協調動作を行うためのシステムの設計ならびに実装について述べ，５章では本システムを用いて行ったデモ展示について述べる．最後に本研究のまとめと今後の課題について述べて本稿のまとめとする．

2. 関連研究

複数のスマートフォン画面を利用して，一つの仮想的な表示領域を作成する試みとしては前述のとおり，Junkyard Jumbotron と呼ばれるアプリケーションや, 画面の構成をアプリケーションの実行中にも動的に変更できる Pin-ch が存在する．

† 東京工科大学 メディア学部
Tokyo University of Technology, School of Media Science

Junkyard Jumbotoron は複数の端末を空間上に配置し，それぞれがブラウザを用いて異なった２次元コードを表示させる．協調させたいすべての画面を一つの写真として撮影し，画像解析を行うことによって端末の空間的な配置を決定する．

Pin-ch は複数のタッチセンサ付き画面をもったデバイスを「つまむ」動作によって空間的な配置を決定した上で連携させることが可能となるアプリケーションであり，複数のデバイスが存在する場合には「つまむ」動作を繰り返してすべてのデバイスの位置をシステムに示す必要がある．さらに，Pin-ch は Bluetooth や Wi-Fi によるピア・ツー・ピア通信のため，端末 1 対 1 での通信は本システムよりもより安定しているが，接続端末数が増加してくると遅延が発生する問題がある．

コンサート会場などでの参加型のイルミネーションとして近年では FreFlow や Xylobands などが利用されている．これらは参加者がそれぞれ持っている専用の端末に無線を用いて信号を送ることで，端末に内蔵された LED の色調や明滅をコントロールするものである．いずれのシステムも参加者側に専用のハードウェアを用意し，会場内に設置されたシステムからの電波によってコントロールされている．参加者に専用のハードウェアを配布する必要があり，コンサートのように閉鎖された空間で行うイベントでは有効であるが，オープンスペースで通りがかった人が参加するようなタイプのイベントには向いていない．また，一方的に無線を用いてコントロール信号を送るため，端末一つ一つをコントロールするような用途には向いていない．

3. ブラウザ間協調動作プラットフォームの提案

3.1 オープンな参加型インタラクション

コンサート会場などでは参加者の一体感を高めるために，参加型のインタラクティブな LED ライトなどが利用されるようになってきている．しかしながら，オープンスペースでは端末の管理の問題もあり，同様の仕組みを利用することは困難である．また，コンサートなどにおいても数千から数万個の専用端末を配布することは一般的ではない．このような参加型のインタラクションはコンサートやスポーツの会場において重視される傾向にあるため，専用の端末を必要としないシステムが求められている．このようなユーザ参加型イベントでの双方向コミュニケーションの価値は今後ますます高まると考えられ，なるべく多くの人が事前の準備を必要とせずにメンバの一員となってインタラクティブなイベントに参加できることが望ましい．

3.2 ブラウザ間協調動作プラットフォーム

本研究では自由に参加，離脱が可能な多数の端末を用いたインタラクティブコンテンツを実現するために，複数の端末上で動作するウェブブラウザを集中してコントロールするプラットフォームを提案する．本システムは複数ブラウザ間での協調したコンテンツ提供を行うようなプラットフォームであり，それぞれの端末がブラウザを通じてサーバと通信することにより，コンテンツの一部となったりコンテンツのコントローラとなったりすることが可能である．

本プラットフォームでは利用者は HTML5 と WebSocket に対応したブラウザの搭載された端末を用意することで，協調動作のメンバとなる．本研究における協調動作とは単にディスプレイを敷き並べて動作させるのではなく，誰もが気軽に参加できるエンターテイメント性のあるマルチディスプレイコンテンツを実現させることである．

3.3 システムの概要

本システムはコントローラ用 URL へアクセスした１つのコントローラからすべてのクライアントの画面の色を一斉に変化させたり，１端末を１ピクセルとして，指で触れた箇所に対応するクライアントの画面色変更，一連のプログラムによる色変化による動きのあるコンテンツの再生などを行う．

本システムではコンサートやライブ会場で行われているような端末の LED の色を変化させるような演出と同様に画面の色を変化させる演出を行うことができるが，それにとどまらずクライアントの端末それぞれをコントロールすることで全体を一つのディスプレイとして表現できる．そのため，端末の位置関係を把握することにより多数の端末をピクセルに見立てた画像出力が可能となる．

3.4 システムのモデルと全体設計

複数のクライアントであるブラウザを同期してコントロールするために，本システムは端末同士で通信を行うピア・ツー・ピア型の通信ではなく，WebSocket を用いたサーバ・クライアント型の通信を行なっている．(図１)

Web ページはコントローラとなるページと，コントロールされるディスプレイ側のページの２種類がある．すべての状態はネットワーク上に設置されたサーバで管理されており，それぞれの端末はサーバにアクセスすることでその役割に従った動作を行うようになる．

コントローラページは排他になっており，他の誰かが同ページにアクセスしコントロールを始めると，既にコントロールしていた端末は停止状態となる．具体的な通信については次章で述べる．

3.5 端末の空間的配置の特定

本システムでは複数の端末を一度に変化させる群として扱うのみならず，空間上の特定位置に配置されている，いわゆるマルチディスプレイとして扱うことが可能である．それぞれの端末を一意に管理し，空間的な配置情報を保持することでマルチディスプレイ化を可能としている．一般

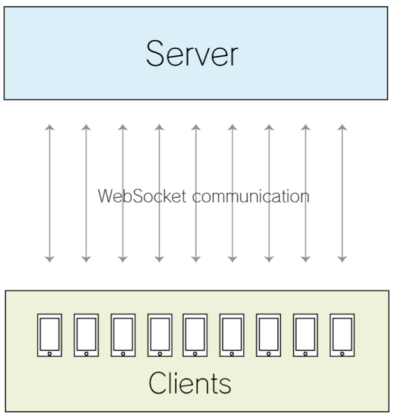

図 1　サーバ・クライアント型の通信

的に，複数の端末を大型の仮想ディスプレイの一部として扱う場合，どの端末が仮想ディスプレイのどこに当たるのか識別できなければならない．その方法として，カメラを用いて各端末の配置を画像として認識する方法や，エディタで各端末の解像度や実際の配置と同じ形を PC 上で設定する方法など幾つかの方法が存在する．本システムでは配置する端末としてスマートフォンを想定しているため，ある程度同じ画面サイズであること，さらに設定後に頻繁に場所を入れ替えることがないという前提条件をおき，縦横に何個までの端末を配置するかというフレーム情報を予めブラウザ上の GUI で設定する方法を選択した．

フレーム情報の指定が完了した後，クライアントは Web ページにアクセスし，並べた順にタップすることで一意な配置番号が端末に割り当てられる．

4. アプリケーションの実装

4.1 システムの概要

本システムに必要なものはネットワーク上にあるサーバとサーバに繋がるネットワークとスマートフォンなどの端末である．コントロールしたい端末はコントローラ用の URL に，コントロールされる側になるにはクライアント用の URL にブラウザでアクセスする．

端末の配置情報を送信する時やログイン時を除いてほとんどの場合，図２のようにコントローラから何らかの要求が送られ，その要求に対する処理をサーバで行うか，そのままクライアントへブロードキャストするという２つのパターンに分けられる．その通信にはリアルタイム性が求められる．現段階でスマートフォンのブラウザ上でリアルタイムに送受信する通信方法は WebSocket が最適だと判断し，本システムは WebSocket を用いて通信を行なっている．

4.2 WebSocket での通信

ブラウザによるリアルタイム通信の手法は Ajax や Comet などで利用される XMLHttpRequest があるが，本システムはスマートフォンで利用できて，尚且つリアルタイム性が

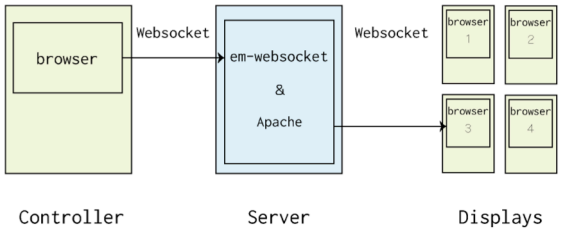

図 2　接続端末とサーバ間の構成図

高い WebSocket と呼ばれる通信規格を採用した．対応状況については，Android 標準ブラウザや Android 版 Opera mini など WebSocket に対応していないブラウザでは動作しない．

WebSocket での通信は，Ajax や Comet で使われている XMLHttpRequest とは異なり，ヘッダ情報の送信は通信開始時だけで済む．その後送受信されるメッセージにはヘッダ情報がつかないため，交換されるメッセージは小さくなる．そのため，ヘッダ情報を通信するオーバーヘッドを抑えることが出来るため本システムのようなリアルタイム性をもったインタラクティブな表現を行うのに適している．また，WebSocket コネクションが確立されると，クローズイベントが発生しない限り，任意のタイミングで送受信が可能になる．クローズイベントは様々な状態で発生する．サーバから又はクライアントからの応答がない，スマートフォンがロックされた，ブラウザがバックグラウンドへ移動したなどである．

WebSocket の仕様上，待機状態でのタイムアウトを防止するために keep-alive の送信をクライアント側から 30 秒ごとに行なっている．また，WebSocket コネクション中に一時的なネットワークの問題が原因で送信に失敗する場合がある．この場合，絵描きソフトやチャットなど過去の情報が必要なアプリケーションでは接続回復後にキューを再送信するなどの実装が必要だが，本システムはリアルタイムに上書きされるような実装の為，再送信の実装は行っていない．

4.3 協調動作サーバの実装

Web サーバは Apache HTTP Server[9]，WebSocket サーバは em-websocket[10]と呼ばれる Ruby 言語で実装された WebSocket ライブラリを用いている．WebSocket でのメッセージのやり取りは全て JSON 形式のテキストで行なっている．JSON の送信は，全てのクライアントにブロードキャストするものと，指定した配置番号のクライアントにのみ送信する２種類のメソッドを実装している．

サーバ側は主にコントローラからの要求を JSON メッセージで受け取り，要求の処理を行った後，任意のクライアントに命令を送信する処理を行なっている．コントローラ

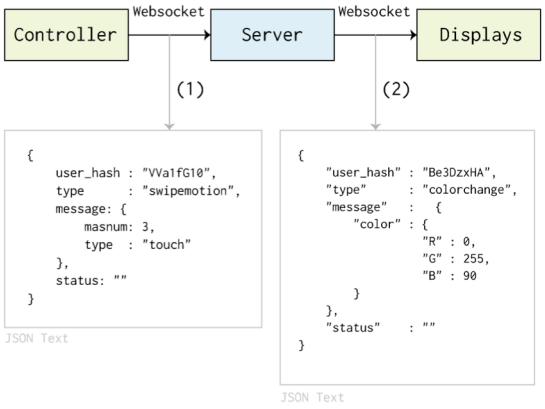

図 3　JSON メッセージの通信イメージ

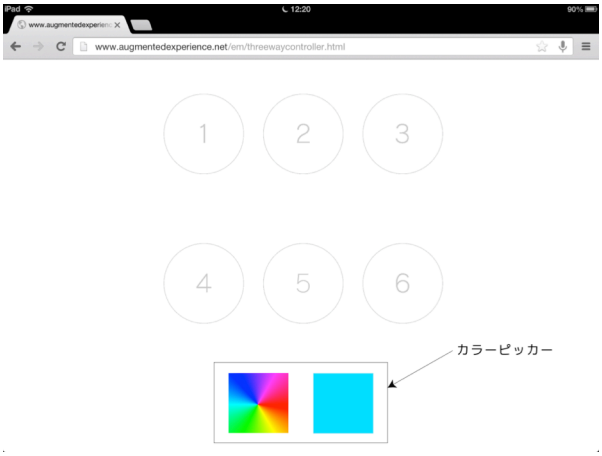

図 4　コントロール画面

からのメッセージは，ログイン時にサーバへ格納されるコントローラの固有 ID と一致するかによってクライアントへのメッセージと区別される．

図 3 は，配置番号 3 番の端末の画面色を変えるメッセージを，コントローラからサーバへ送って 3 番の端末の画面色が変わるまでの JSON メッセージのやりとりのイメージである．

コントローラが送信するメッセージは図 3 の（1）に示される部分である．内容として含まれているのはそれぞれの端末に固有の ID である user_hash とコマンドの種類を指定する type ならびにそれぞれのコマンドに必要な情報を内包する message の情報である． それぞれのメッセージに含まれる status にはクライアントからのログイン要求時には”want”が含まれ, 要求への返答時に”OK”や”NG”などの情報が含まれるが，この例では利用していない．

今回の message の内容としてはクライアントを特定するための masnum と，タッチ時の色変化なのか，タッチ終了時に背景色に戻す色変化なのかを意味する type である．

位置番号に対する端末の固有 ID の情報はサーバ側で保持しており，配置番号を受け取ったサーバは配置番号がセットしてある端末のみに命令を送る処理を行う．どの色に変えるかについては別途, 図 4 下部のカラーピッカーでの色変更に応じてコントローラから送られる JSON メッセージに従う．

4.4 クライアントの実装

クライアントソフトウェアは HTML5 と JavaScript，CSS を用いて実装している．WebSocket の送受信の動作は全て JavaScript で行なっており, サーバからメッセージが受信されると，JSON 内の type に従い，画面の色を変えたり画像を表示させたりなどの処理へ移行する．図 3 （2）にサーバから送られるメッセージを示す．コントローラからサーバへのメッセージと同様に通信先を示す user_hash とともに，コマンドの種類を示す type，その内容を示す message で構成される．現在定義されている type は色を変える命令の colorchange である． message にはどの色にするのかという情報を R,G,B それぞれ 256 階調で記述したものが含まれる．

コントローラ画面は，クライアントのディスプレイの色をコントロールするためのインターフェイスとして機能する．図 6 はクライアント 1 つずつをリアルタイムに制御するためのインターフェイスである．1 から 6 と書かれた丸はそれぞれの位置に配置されたクライアント端末を表しており，丸印をタップすることで，端末を任意の色に変化させることが可能である．端末の色は下部左側の虹色のカラーピッカーをタップして選択した色であり，現在の設定色は右側の四角形に表示されている．

コントローラ画面のインターフェイスとクライアントのディスプレイ部分の殆どは HTML5 の Canvas を用いて実装している．そのため曲線が多いデザインや複雑な形のユーザインタフェースへの変更を行うことができる．

4.5 クライアント端末の識別

端末を識別するための ID を振り，指定した ID の端末へ背景色変更命令を送信する手順は以下のとおりである．

初めに接続クライアントが Web ページにアクセスした時にブラウザ上でハッシュを生成する．生成したハッシュはタブが閉じられるまで保存されるセッションストレージへ格納する．格納されたハッシュ値をログイン要求時にサーバへ送り，サーバ側で接続クライアントリストに登録する．接続クライアントリストにはハッシュの他にその接続クライアントの配置番号や画面サイズなどが保存される．

その後コントローラから，指定した配置の端末の色を変える命令がサーバへ送られると，サーバ側は接続クライアントリストから，受け取った配置番号のハッシュを見つけて端末を割り出し，その端末のみに命令が送られる．

5. デモ展示とその結果

本システムの実装をデモ展示することにより，その有効

図 5　平面に端末を配置したデモの様子

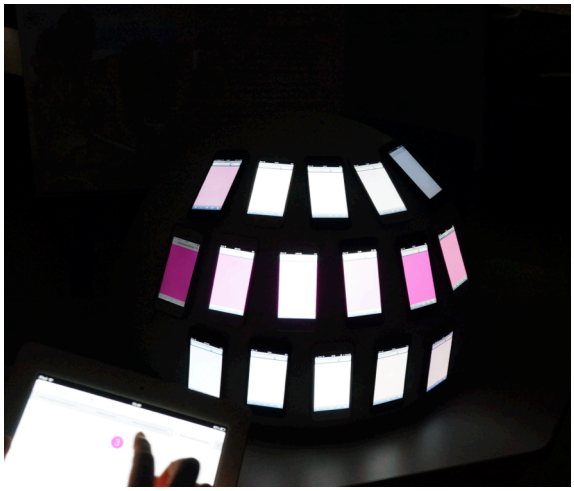

図 6　　半球面へ配置した端末によるデモの様子

性について検証を行うとともに，観覧者からの感想を聴取した．デモ展示は２回行われ，１つは卒業研究の中間発表としてであり，もう一方はオープンキャンパスにおいてである．これらの展示の特徴としては，どちらも本システムについての知識がない人たちが見学に訪れることである．展示への参加者はデモ用に設置されている端末を操作して，システムの動作を確かめるのみならず，会場に掲示された URL へとアクセスすることによって，自らのもつ端末をシステムの一部に加えて動作させることが可能である．

デモ展示の会場ではスマートフォンを接続するための専用の Wi-Fi 基地局を設置し，準備した携帯端末はすべて Wi-Fi によりインターネットに接続した．サーバはインターネット上のクラウドサービスで動作しており，Wi-Fi のみならず公衆回線を用いてアクセスすることも可能である．

図５に平面に端末を配置したデモの様子を示す．壁面に設置した透明なウォールポケットに 6 台の iPodTouch を入れて壁に固定している．コントローラとして設定した端末は iPad を利用しており，観覧者に操作してもらうことを想定した．観覧者が実験に参加する場合には，ポケットの空きスペースに端末を入れて URL へのアクセスを行うことで本システムに個人の端末を追加することが出来る．

そこでデモにおいては本システムの簡単な説明をした後，興味をもった観覧者にはデモページへの URL が載った QR コードを各自のスマートフォンで読み取ってもらい，ブラウザでアクセスしてもらう．スマートフォンによる QR コードの読み込みから，ディスプレイの一部になりコントロール可能になるまで 1 分程度で行う事ができ，ブラウザ上での実装による運用の手軽さを実証できた．

図６の展示では端末を半球面に配置することにより，端末設置の自由度を示している．このデモにおいては 16 台の端末が利用されているが，事前には 10 台のみを準備し，残りの 6 台はデモ展示中に観覧者から借り受けたものである．一般的なマルチスクリーンシステムと異なり，それぞれのスマートフォンは単体で完結した機能を持っているため，設置の自由度が高く，平面のみならず曲面にも端末を配置することが容易である．曲面への配置の場合には，端末を縦×横の矩形で示すことは困難であるが，欠番となる端末の場所を設定時に指定することで対応している．

これらの実験によって得られた観覧者の反応は以下のとおりである．

まず，平面での展示においてはブラウザでアクセスするのみで自らのスマートフォンがディスプレイの一部になっている様子を見て「まさに不思議だ」「コンサートで使ったら面白そう」「インテリアになりそう」などの肯定的な意見が寄せられた．また，本システムは停止させるまで自動でカラーバーが流れるモーションがあり，この機能上でリアルタイムに色を変えられるカラーピッカーの操作に，観覧者は特に驚きを見せていた．

半球面に配置したデモにおいては，小型の端末を並べることによる３次元的な形状の自由度に関して肯定的な意見を得ることが出来た． 球面へのスマートフォン配置は，それぞれが小型のディスプレイを持ち独立して動作することが可能なスマートフォンだからこそ実現できるマルチディスプレイの実現方法であり，これまでにない表現方法・演出が可能になるだろうとの意見が挙げられた．

6. おわりに

本論文では多数のスマートフォン上で動作するブラウザを連携させることで，あたかも画面サイズのピクセルを持った大型ディスプレイのように操作できるシステムを提案し，実装を行った．スマートフォンのブラウザで特定の URL にアクセスするだけで端末をシステムの一部として協調動作させることが可能となる．

本システムではスマートフォンや PC に搭載されている汎用のウェブブラウザを用い，WebSocket と HTML5 によりマルチディスプレイ環境を実現している．そのためネイ

ティブアプリケーションと異なり，利用者が事前準備を必要としない点が特徴であり，参加者全体を把握していないイベントや偶発的に参加者が来訪するイベントであっても参加者に連携を促すことが出来る．

本論文で紹介した壁一面に敷き並べる平面でのデモでの聞き取り調査では，主に高校生や大学生という，スマートフォンの所持者が多いと考えられる年齢層での本システムの利用可能性と参加の容易性が実証できた．また，半球面への配置のデモでは，スマートフォンだからこそできる曲線表現の可能性・設置の自由度の高さの実証できた．

本システムはコンサートやスポーツの観客がより一体感をもってイベントを楽しむための手段として用いられることが想定される．このような用途に耐えうるだけのシステムの規模性の検証として，端末を増やした場合の性能評価と端末の空間的配置の特定方法は今後の課題として挙げられる．

参考文献

1) 田中 潤, 太田 高志 : スマートフォンを利用した複数画面の連携表示と動的なレイアウト変更によるアプリケーション,IPSJ Interaction 2012
2) Rick Borovoy, Brian Knep : Junkyard Jumbotron, http://jumbotron.media.mit.edu/
3) au by KDDI 驚きを，常識に, http://www.au.kddi.com/odoroki/
4) ODOROKI, https://itunes.apple.com/jp/app/odoroki/id590209985
5) 新ペンライトシステム 「FreFlow（フリフラ）」開発秘話, http://www.musicman-net.com/report/82.html
6) mostwatchedtoday : Disney's 'Glow With The Show' LED Mickey Mouse Ears., http://www.mostwatchedtoday.com/disneys-glow-with-the-show-led-mickey-mouse-ears/
7) xylobands product infomation : product information , http://xylobands.com/glowbands-product_info.php
8) IETF : The WebSocket Protocol ， http://tools.ietf.org/html/rfc6455
9) The Apache Software Fundation : The Apache HTTP Server Project. http://httpd.apache.org/.
10) em-websocket , https://github.com/igrigorik/em-websocket