2013-Interaction (1EXB-26)
2013/2/28

Webブラウザによるリアルタイムな協調作業のための WebSocketを用いたアドホックマルチキャスト環境

河下 夏輝[1]　吉田 恭太[2]　高田 秀志[2]

概要：近年，ブラウザのリッチクライアント化が進み，サーバとブラウザ間でリアルタイムな双方向通信が行えるようになっている．しかし，サーバとブラウザの間にあるネットワーク経路によってネットワーク遅延が大きく左右されるという問題がある．本稿では，持ち寄った端末同士で行う協調作業に着目し，WebSocket によるアドホックマルチキャスト環境を提案する．提案環境は，無線 LAN 等で繋がった端末のブラウザ間におけるアドホックネットワークの構築と，当該ブラウザ間における JavaScript の関数呼び出しの同期を可能にするものである．

Ad-Hoc Multicast Environment Using WebSocket for Real-Time Collaboration on Web Browsers

Kawashita Natsuki[1]　Yoshida Kyota[2]　Takada Hideyuki[2]

Abstract: Today, web browsers are being developed to be a rich client, which enables two-way communication between web browsers and servers. However, it has a problem that the network churn can be caused depending upon a connection path between web browsers and servers. This paper proposes an ad-hoc multicast environment with WebSocket, considering that users bring their own terminals and perform collaborative work. This environment forms an ad-hoc network among the terminals' web browsers in the same wireless LAN, and realizes the synchronization of the JavaScript function-call among terminals.

1. はじめに

近年，HTML5 や WebGL といった新しい技術が Web ブラウザ (以降，「ブラウザ」という) に標準的に備わり，ブラウザのリッチクライアント化が進んでいる．また，サーバとブラウザ間でリアルタイムな双方向通信が行える通信規格も登場してきている．しかし，代表的な通信規格である WebSocket[1] はサーバを必要とし，サーバとブラウザ間の経路によってネットワーク遅延が大きく左右されるという問題がある．

そこで本稿では，WebSocket サーバ機能を持つブラウザを利用することで，ブラウザ間のアドホックネットワークを自律的に構築するアドホックマルチキャスト環境を提案する．これにより，特別に用意したサーバを必要とせずに，協調作業を行うために持ち寄ったノートパソコンやタブレット端末などを用いて，ブラウザ間で安定したリアルタイム通信が行えるようになる．さらに，この提案環境には，協調作業アプリケーションを容易に開発できるようにするために，構築したネットワーク内にある各ブラウザ間で JavaScript の関数呼び出しを同期する機能が含まれる．また，本稿では，この環境を用いて構築した最大 4 人までが一緒に遊べるゲームアプリケーションについて述べる．

2. ブラウザを用いた協調作業

現在，Google Docs などのブラウザ上でリアルタイムな協調作業を行う Web アプリケーションが開発されている．ブラウザ上で 3D 描画が可能になるなど Web ページの表現の幅もますます広がり，今後も様々な協調作業 Web ア

[1] 立命館大学院 理工学研究科
Graduate School of Science and Engineering, Ritsumeikan University
[2] 立命館大学 情報理工学部
Faculty of Information Science and Engineering, Ritsumeikan University

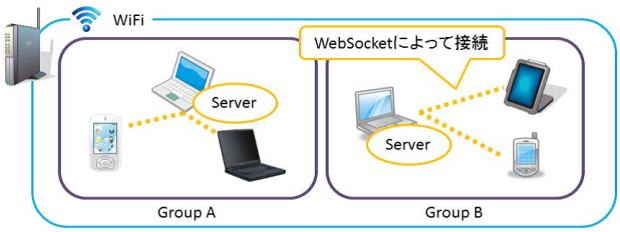

図 1 WebSocket によるアドホックネットワークの例
Fig. 1 An Example of Ad-Hoc Networks Using WebSocket

プリケーションが開発されることが予想される．

Web アプリケーション同士が直接通信を行う方法の一つに，ブラウザ上でリアルタイムコミュニケーションを行うためのフレームワークである WebRTC[2] がある．WebRTC では，カメラやマイクからマルチメディアのストリームを取り込んでリモートピアと送受信することができる．これにより，ビデオチャットアプリケーションなどを開発することが可能になり，ブラウザ上で高品質なコミュニケーションが実現できる．しかし，現状の WebRTC で利用できる API では，扱うことのできるデータは音声や映像のメディアデータだけであり，JavaScript で扱う数値や文字列といったデータをピア間で送受信することはできない．

リアルタイムな情報のやり取りを行う代表的な方法として，ブラウザとサーバ間でリアルタイムな双方向通信を可能にする WebSocket がある．WebSocket により，サーバを介してブラウザ同士のリアルタイムな協調作業を実現することができる．しかし，WebSocket ではサーバの利用が前提となるため，サーバとブラウザ間の経路によってネットワーク遅延が大きく左右されるという問題がある．対面型の協調作業に着目すると，持ち寄った端末間でネットワークを構築することで，より安定したリアルタイム通信を実現できると考えられる．

3. WebSocket を用いたブラウザ間アドホックマルチキャスト環境

本アドホックマルチキャスト環境は，対面型の協調作業を対象とし，持ち寄った端末のブラウザ間におけるアドホックネットワークの構築と JavaScript の関数呼び出しの同期を実現する．

3.1 WebSocket によるアドホックネットワークの構築

本アドホックマルチキャスト環境では，ブラウザ間のアドホックネットワークを WebSocket によって構築する．このためには，WebSocket サーバをブラウザ上で動作させ，ブラウザ同士で WebSocket の接続を行う必要がある．そこで，ブラウザの一つである Firefox の拡張機能を使い，ブラウザ上で WebSocket サーバを動作させる方法 [3] をとる．

図 1 に，構築するアドホックネットワークの例を示す．アドホックネットワークは，持ち寄った端末のブラウザの内一つが WebSocket サーバとしての役割を果たし，その他のブラウザがそのサーバに接続することで構築される．このネットワークでは，最低一つ以上の WebSocket サーバを動かすことのできる Firefox のブラウザが必要となる．サーバを動かせるブラウザがグループに一つあれば，それ以外のブラウザがサーバを動かせないものでもアドホックネットワークを構成することができる．

図 1 のようなネットワークを構築するためには，無線 LAN 等で繋がった複数の端末の中から，協調作業を行うグループの端末を識別し，さらに，グループの端末が共通のサーバに接続する必要がある．また，WebSocket サーバへの接続にはそのサーバの IP アドレスとポート番号が必要となるため，サーバの役割を果たす端末の IP アドレスとポート番号を各端末が調べる必要がある．このような条件の中で各端末がネットワークを構築するための方法は以下のとおりである．

まず，協調作業を行う端末がアプリケーションを起動した (Web ページを開いた) ときに，WebSocket サーバを起動することのできるすべての端末は，共通のポート番号を使った端末発見のためのサーバと端末固有のポート番号を持つアドホックネットワークのためのサーバを起動する．共通のポート番号とはアプリケーション毎に開発者が設定できるものであり，端末固有のポート番号とはサーバを起動したときにランダムの値が設定されるものである．

次に，協調作業を行うグループの各端末は，以下の手順によってサーバの役割を果たす端末の IP アドレスとポート番号を調べ，調べた結果をもとに各端末がグループで共通の端末一つに接続する．ここで，各端末では利用者によって設定されたグループ ID を持っているものとする．また，協調作業を行うすべての端末がアプリケーションを起動しているものとする．

(1) 各端末は一定範囲内の IP アドレスの内，接続を試みていない一定数のアドレスに対して値の小さいものから順に，共通のポート番号を使って WebSocket による接続を試みる
(2) 接続が成功した端末とはグループの ID の比較を行う
(3) もし同じ ID の場合はそのサーバ端末から固有のポート番号を受け取り，IP アドレスとともに保存しておく
(4) 一定数のアドレスを調べ終わると，保存されたサーバ情報があるかどうかを調べ，なければ手順 (1) に戻る
(5) 保存されたサーバ情報の中から最も IP アドレスの値が小さい端末情報を取り出し，自身の端末かどうかを IP アドレスを比較して調べる
(6) もし，自身の端末と同じ IP アドレスの場合はサーバの役割になり，違う場合はその端末の IP アドレスと固有のポート番号を使い WebSocket による接続を行う

この方法では，上記 (5)，(6) で最も IP アドレスの値が小さい端末に接続を行うことで，グループの各端末は共通

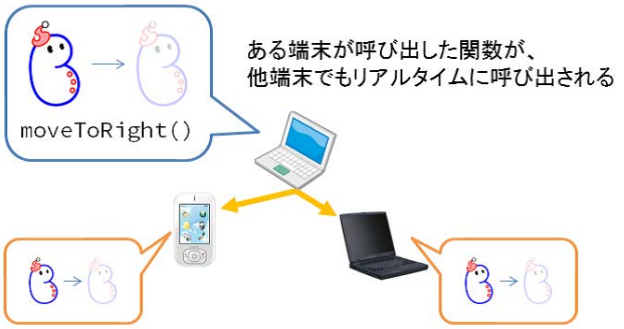

図 2 関数呼び出し同期の例
Fig. 2 Synchronization of Function-Call

の端末一つに接続することができる．上記 (1)〜(4) で，各ブラウザが IP アドレスを全探索しているのは，サーバのアドレスを登録し，それを発見するためのサービス発見の機能がブラウザには標準的に備わっていないからである．また，IP アドレスの探索範囲を一定範囲内としているのは，すべての IP アドレスを短時間に調べることが難しいからである．現状の実装では，IP アドレスの探索範囲を各端末が自動的に設定することができないため，Web アプリケーション側で予め設定しておくか，アプリケーションのユーザが利用中に設定する必要がある．

3.2 関数呼び出しの同期

本アドホックマルチキャスト環境は，JavaScript の関数呼び出しの同期を行う機能を持つ．関数呼び出しの同期とは，ある端末で呼び出された関数を，アドホックネットワークで繋がっている他の端末でもリアルタイムに呼び出す機能のことである．図 2 に，関数呼び出しの同期の例を示す．このように，関数呼び出しの同期によりブラウザ上のビューを同期することが可能であるため，協調作業アプリケーションの開発がより簡単になると期待される．

本提案環境での関数呼び出しの同期は，JavaScript の関数を置き換える機能と，プログラム中に JavaScript のコードを文字列として生成し実行することができる機能を用いて実現されている．JavaScript では，一度作成した関数を別の名前に変更したり，もともとある関数名に別の関数を割り当てることができる．これは，JavaScript での関数がオブジェクトであることによって可能になっている．また，JavaScript には文字列を受け取って JavaScript のプログラムコードとして実行できる「eval」という関数が用意されている．この関数を用いることで，プログラム中で任意の文字列を生成し，プログラムコードとして実行することができる．

図 3 に関数呼び出しが同期される流れを示す．関数呼び出しの同期は，各端末で予め登録された関数同士で行われる．本環境では，同期を行わせたい関数を登録するための関数「SetFunc」を提供する．SetFunc には，同期したい関数の名前と関数オブジェクト自体を渡す．SetFunc は，受け取った関数オブジェクトを各端末に用意されたリストに保存し，もともとの関数が呼び出されたときに同期処理を行うようにするため，もともとの関数名に同期処理を行う新しい関数オブジェクトを割り当てる．これにより，登録された関数を呼びだそうとすると，同期処理を行う関数が呼び出されるようになる．新しい関数が呼び出されると，リストの中に保存されているもともとの関数を呼び出すことのできるプログラムコードが文字列として生成される．さらに，このプログラムコードをアドホックネットワークで接続しているすべての端末にマルチキャストする．このマルチキャストを受け取った端末は eval 関数を用いてプログラムコードを実行する．これにより，端末間で同じ関数が呼び出されることになる．現在の実装では，端末間でマルチキャストされるプログラムコードが文字列であるために，オブジェクトへの参照といった端末毎に異なっているような引数を伴う関数呼び出しの同期は行うことができない．

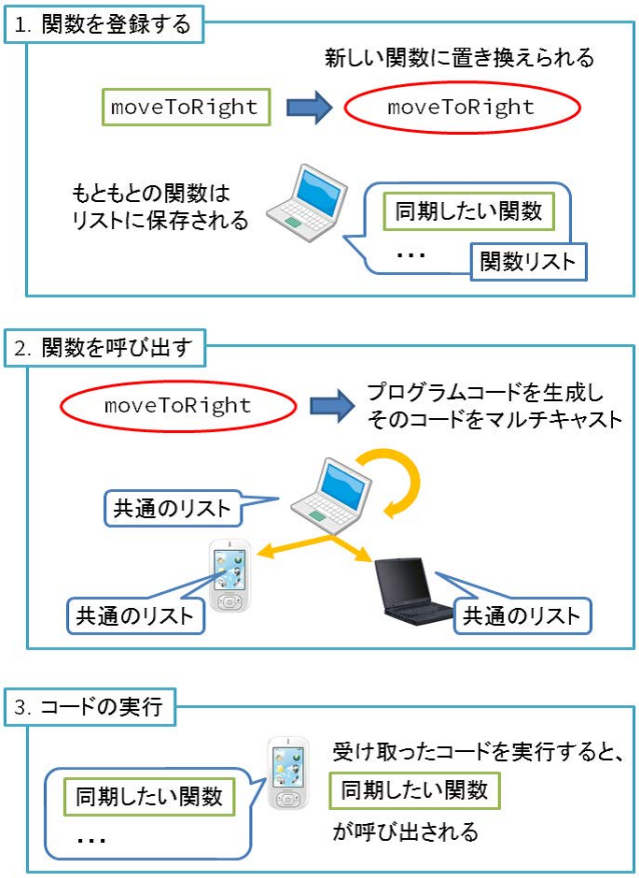

図 3 関数呼び出し同期の流れ
Fig. 3 Mechanism of Function-Call Synchronization

4. アプリケーションの実装

アドホックマルチキャスト環境を用いて，持ち寄った端末のブラウザ間で最大 4 人が同時に遊ぶことができる簡単

図 4 アプリケーション画面
Fig. 4 Application View

なゲームアプリケーションを構築した．図 4 に構築したアプリケーション画面を示す．このアプリケーションでは，無線 LAN 等で繋がった各ブラウザで動作するアプリケーションに対し，利用している端末のプライベート IP とサブネットマスク，また，一緒に遊ぶグループの ID を入力することでネットワークが自動的に構築される．ネットワークが構築されると，操作するキャラクタの動きが各ブラウザ間で同期される．

5. おわりに

本稿では，持ち寄った端末のブラウザ上でリアルタイムな協調作業を実現するために，WebSocket によるアドホックマルチキャスト環境を提案した．提案環境は，無線 LAN 等で繋がった端末のブラウザ間におけるアドホックネットワークの構築と，当該ブラウザ間における JavaScript の関数呼び出しの同期を可能にするものである．これによって，安定したブラウザ間の双方向通信と容易な協調作業アプリケーションの開発が実現できると期待される．さらに，この環境を用いて構築した，持ち寄った端末で複数のユーザが同時に遊ぶことのできるゲームアプリケーションについて述べた．今後は，提案環境がネットワークの構築や同期に必要とする時間等を調査し，ブラウザ同士の協調作業を実現する他の環境との比較を行う予定である．

参考文献

[1] World Wide Web Consortium : The Web SocketAPI (online)，入手先 ⟨http://www.w3.org/TR/websockets/⟩ (2012.11.07).
[2] World Wide Web Consortium : WebRTC 1.0: Real-time Communication Between Browsers(online)，入手先 ⟨http://dev.w3.org/2011/webrtc/editor/webrtc.html⟩ (2012.11.07).
[3] 與儀那広，城間政司，長田智和，谷口祐治，玉城史朗：WebSocket を用いた Web ブラウザ間 P2P 通信の実現とその応用に関する研究，信学技報，no.372，pp.59-62，(2011).