マルチディスプレイ・レスポンシブウェブシステム

坂井成道[†1]　峰松美佳[†1]　川添博史[†1]　会津宏幸[†1]

スマートフォンやスマートテレビをはじめ，ネットワークに接続可能なディスプレイ搭載機器が普及し，ユーザはそれらを複数種類所持するようになった．今後，ユーザがそれぞれの機器の特色を活かしながら同時に組み合わせて利用することが想定される．一般に，複数の機器を連携させるアプリケーションは単体の機器上で動作するアプリケーションよりも設計や実装が難しい．我々は，複数の機器のブラウザが連携して動作する新しい Web 体験「マルチディスプレイ・レスポンシブウェブ」を提案する．本稿では，マルチディスプレイ・レスポンシブウェブシステムの実現における課題と解決を示し，試作したシステムについて述べる．

Multi Display Responsive Web System

NARUMICHI SAKAI[†1]　MIKA MINEMATSU[†1]
HIROSHI KAWAZOE[†1]　HIROYUKI AIZU[†1]

1. はじめに

スマートフォンやスマートテレビ [1)]のように，ユーザがネットワーク対応機器を利用する機会が増加している．さらに，ディスプレイを搭載した機器が多数普及し，それらの低価格化などにより一人のユーザあたり複数の機器を所持することで，ユーザが従来よりも多くのディスプレイを通して情報収集や情報発信，コンテンツの享受をすることが多くなっている．これにともなって，機器を同時に複数組み合わせて利用するマルチ機器（マルチスクリーン）のユースケースが想定されている．たとえば，テレビで動画を見ながら動画の感想をスマートフォンで SNS に投稿するなど，機器によって役割は違うものの，ユーザの一連の行動がコンテンツを楽しむというひとつの目的の元にある，といった例は現在もよく見られる．

一般に，マルチスクリーンアプリケーションの開発は，シングルスクリーンアプリケーションの開発よりも困難になる．例えば，複数のスクリーンを自由にレイアウトできる場合や使用する機器の数が多い場合，それらが自由に参加離脱することや，機器の種類や数に応じてレイアウトを適切に変更することを想定してアプリケーションを設計しなければならない．

一方，スマートフォンに代表されるように，機器のディスプレイ解像度は多岐にわたっている．従来のシングルスクリーンアプリケーション開発においても，開発者はこれらのさまざまなディスプレイ解像度に合わせてアプリケーションを開発し，テストを行う必要があった．ディスプレイ解像度のバリエーションへ対応する技術として，Web の世界では「レスポンシブウェブ」という概念を用いることがある．これは HTML5[2)]の柔軟性を利用して，機器のディスプレイ特性に合わせて表示を行う．例えば，PC に搭載されたブラウザで表示した場合には 3 ペイン構成になっている Web ページを，ディスプレイの幅の小さなモバイル機器などのブラウザでは 1 ペインに構成する．我々はこの概念をマルチディスプレイに拡大した，「マルチディスプレイ・レスポンシブウェブ」を提案する．

本稿では，マルチディスプレイ・レスポンシブウェブに関して課題と解決を示し，試作の詳細について述べる．

2. マルチディスプレイ・レスポンシブウェブの提案

マルチディスプレイ・レスポンシブウェブでは，複数のディスプレイのレイアウトや機器の用途，性能，特徴により，ページのレイアウトをそれぞれのディスプレイで変更して表示させ，さらにそれらのディスプレイの表示を連携させる．マルチディスプレイ・レスポンシブウェブの元となるレスポンシブウェブの概念と合わせて以下に説明する．

2.1 従来のレスポンシブウェブ

レスポンシブウェブとは，様々な解像度のディスプレイ，ブラウザ，バージョンなどの条件に対して，動的に Web ページの表示形式を変更してユーザに見せる技術あるいは概念である．例えば，図 1 のように，携帯端末などの縦長のモバイル機器ブラウザでは 1 ペイン構成，PC ブラウザでは 3 ペイン構成になるように，ページ読込時にスタイルを設定する．ディスプレイやブラウザが多岐にわたる現在では，多くの Web サイトがレスポンシブウェブを取り入れている．

単体の Web ページソースを元として，ページ中の

†1（株）東芝 研究開発センター ネットワークシステムラボラトリー
Network System Laboratory, Corporate Research & Development Center

JavaScript™ あるいはスタイルシートの機能 (Media Query[3]) が上記の条件を参考にしてページ要素のスタイルを変更することでこれを実現する．

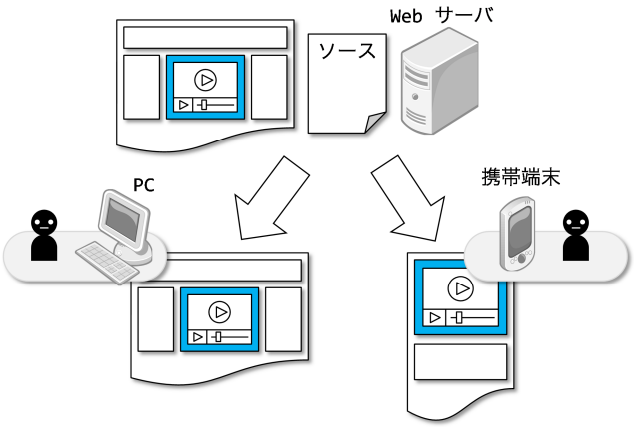

図 1：レスポンシブウェブのイメージ

2.2 マルチディスプレイ・レスポンシブウェブ

我々が提案するマルチディスプレイ・レスポンシブウェブは，複数のディスプレイにひとつのコンテンツがまたがり，ディスプレイ解像度や機器の種類，それらの組み合わせや台数などの条件に合わせて，コンテンツの表示を動的に変更するものである．レスポンシブウェブではコンテンツの表現が単体の機器内に限られているのに対して，マルチディスプレイ・レスポンシブウェブでは複数の機器にまたがってコンテンツを動的に展開する．

例えば，図 2 のように，本来はひとつのページ上に動画とそのコントロール UI が一緒に存在するコンテンツを，テレビに動画を，スマートフォンにコントロール UI を分けて表示することで，手元のスマートフォンでテレビの動画を制御できるようになる．また，機器ごとに最適な UI となるように表示を変更することで，テレビにフルスクリーン動画を，スマートフォンにはボタンが大きく操作しやすいコントロール UI を提供することも可能となる．他の例として，左右に配置したディスプレイに対して，ひとつの Web ページを中央で分割してそれぞれ表示することで，擬似的に横幅の広いブラウザ上のページとして表示する．

上記のような，マルチディスプレイ・レスポンシブウェブを実現するためには，シングルスクリーンアプリケーションとしての Web ページを分割し，分割したページの状態を複数のブラウザで共有する必要がある．具体的には，ページの DOM 要素の状態，イベント，JavaScript™ の実行状態などを共有する．

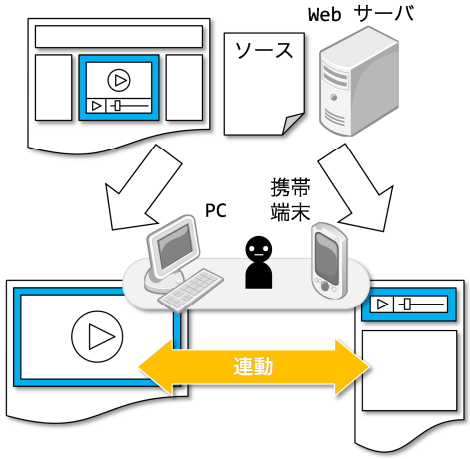

図 2：マルチディスプレイ・レスポンシブウェブのイメージ

3. 方針検討

3.1 前提条件・要件

マルチディスプレイ・レスポンシブウェブを実現するに当たり，いくつかの前提条件・要件がある．ユーザが保持する複数の機器に特別なソフトウェアやプラグインなどのインストール作業を要求することは，ユーザの負担になるため，避けるべきである．また，開発コストやメンテナンスの面からも，多種多様の機器のために独自にブラウザやそのモジュールを個別に開発するのは望ましくない．既存の Web サイトにもスムーズに適用できるよう，Web 管理者あるいはページ作成者に多くの作業を発生させるべきではない．性能面においては，少なくとも従来のブラウザにおける従来のページ利用の応答性を確保することが望まれる．

3.2 実現方針

上記の条件を満たし，ひとつのページから複数のページを生成し，それらをひとつのページのように振舞わせるために，我々は Web プロキシを用いた．Web プロキシを利用することで，従来のサーバ側，ユーザ側を改変することなく利用可能という利点がある．Web プロキシを用いずにサーバ側でマルチディスプレイ用のコンテンツをあらかじめ用意しておく方法 [4]もあるが，要件を満たさない．プロキシの役割を図 3 に示す．この方式では，ユーザ機器のブラウザから HTTP 要求を受信した Web プロキシが，Web サーバから Web ページを取得し，ページの内容を解析し，ディスプレイのレイアウトや各機器の情報にしたがってページを分割・生成し，各機器へ転送する．また，ブラウザのページ上で発生したイベントを各機器で共有するために，ブラウザが自身の持つページ上で生じた操作イベントを検知すると，ブラウザはそれらをプロキシ経由でその他の機器上のブラウザに対して転送する．

* JavaScript は，米国 Oralce Corporation およびその子会社，関連会社の米国およびその他の国における登録商標または商標です．

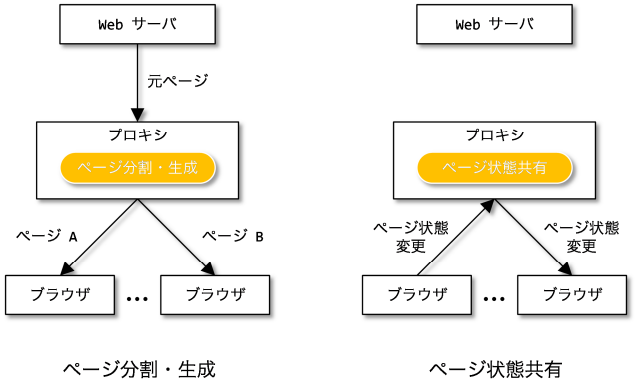

図 3：提案システムにおけるプロキシの役割

4. 課題と解決方法

4.1 課題

上記方針に従ってマルチディスプレイ・レスポンシブウェブの基本的なシステムを実現するにあたり，以下の課題が挙げられる．

- ページの分割配布と状態共有
  ひとつのページを複数の部品へ分割し，それぞれの機器のブラウザへ配布，それらの状態を共有する．
- 途中参加機器への対応
  近年は，ブラウザ上でページの構成が動的に変化するページが増えている．これらの動的ページに対応するため，途中から別の機器がページ表示に参加した場合にも，現在のページの状態を参加した機器のブラウザに対して反映する．
- 操作レスポンスの低減防止
  複数の機器のブラウザにまたがったページをひとつのコンテンツとしてユーザに提供するに当たり，少なくとも通常の Web ページを利用している時と同等の操作レスポンスを提供する．

4.2 解決方法

上記の課題に対する解決を以下に述べる．具体的な動作に関しては後述する．

- ページの分割配布と状態共有
  プロキシが機器上のブラウザから HTTP リクエストを受けて Web サーバからページを取得すると，プロキシはオリジナルのページに対してイベント共有機能とページ変更機能を埋め込む．イベント共有機能はページ上で発生したユーザからのイベントをプロキシに送信する機能で，ページ変更機能はプロキシからの要求により逐次ブラウザ上のページ状態を変更するための機能である．これらの機能によりページを複数のブラウザに分割し，それらの状態を共有することができる．なお，ページをどのように分割するかは本稿では扱わない．
- 途中参加機器への対応
  途中参加機器に対して，すでにページを表示しているブラウザと同じページ状態を共有させるため，プロキシにてページ全体の状態を保持する．新たな機器参加時に最新の状態を該当機器上のブラウザへ転送することで，新規機器上のブラウザは，その他の既存のブラウザと同じページ状態を持つことができる．
- 操作レスポンスの低減防止
  ブラウザにて発生したイベントをプロキシに対して通知する際，同時にブラウザローカルでイベントを反映する．こうすることで，プロキシからのページ状態の変更結果を待つことなく，ブラウザへ変更を反映することができる．

5. システム構成

マルチディスプレイ・レスポンシブウェブの試作システムの構成について，図 4 を用いて説明する．

ブラウザの搭載されたユーザ機器と Web サーバの間にプロキシを配置する．プロキシには，一般的な Web プロキシに搭載されている HTTP 転送モジュールの他，ページにイベント共有機能とページ変更機能を組み込むためのモジュール（機能組込モジュール）を置く．また，プロキシはブラウザに発生したイベントを解釈して自身が保持するページマスタへ反映するモジュール(マスタ管理モジュール)を持つ．このページマスタには，各ブラウザからプロキシに送信されてくるイベントを反映する．試作では，マスタ管理モジュールは既存のブラウザ機能を利用した．

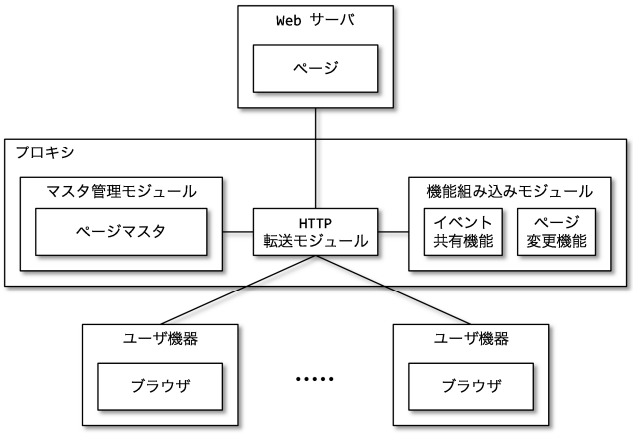

図 4：試作システムの構成

6. 動作説明

Web ページ閲覧操作に関し，試作したマルチディスプレイ・レスポンシブウェブシステムの動作を説明する．

6.1 ブラウザによる初回ページ読込時

図 5 にブラウザによる初回ページ読込時のフローを示す．機器上のブラウザが HTTP リクエストをプロキシへ送信する．プロキシはリクエストを受信し，Web サーバへ HTTP リクエストを転送する．Web サーバから HTTP レスポンス（ページのソース）を受信すると，プロキシはイベント共有機能とページ変更機能をページに対して埋め込む．プロキシは機能を埋め込んだページをマスタに保存した上でブラウザに対して送信し，ブラウザはページを描画する．

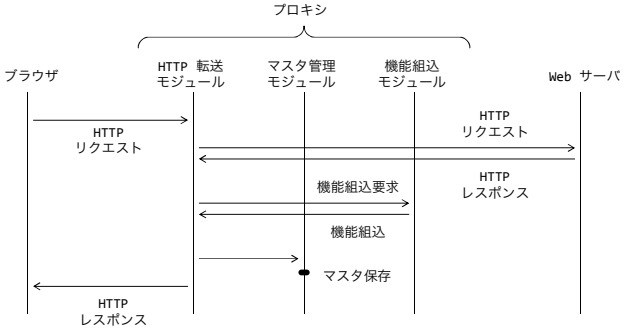

図 5：ブラウザによる初回ページ読込時

6.2 追加ブラウザの参加時

図 6 に追加ブラウザの参加時のフローを示す．最初にプロキシに対してページを要求したブラウザと合わせて，他のブラウザで同じページを分割表示する場合を述べる．二つ目以降のブラウザが HTTP リクエストをプロキシに対して送信する．新規の機器のブラウザがページ表示に参加した場合，プロキシは新たに Web サーバに対して HTTP 要求を送信せず，自身が現在マスタに保持している最新のページ状態をブラウザに対して転送する．ブラウザは受信したページを描画する．

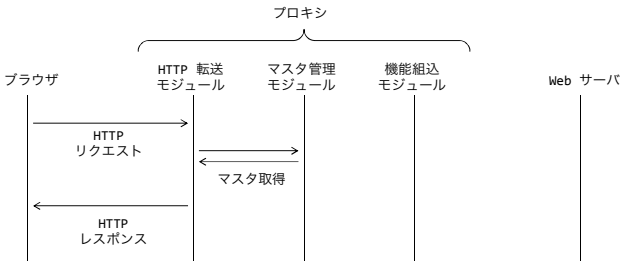

図 6：追加ブラウザの参加時

6.3 ブラウザでイベント発生時

図 7 にブラウザでイベント発生時のフローを示す．ユーザ操作によりブラウザ上のページにイベントが発生すると，まずブラウザは自身の保持するページに変更を加え，同時にプロキシへ同イベントを送信する．このイベント送信は，ページに埋め込まれたイベント共有機能による．プロキシはマスタページに対してイベントを適用し，ページを変更する．

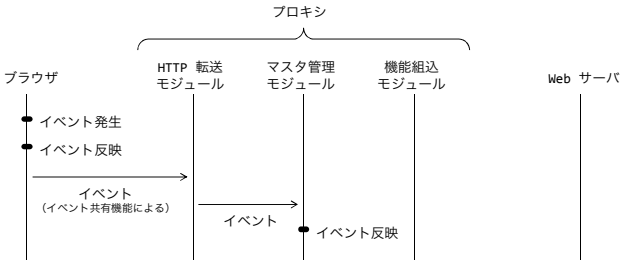

図 7：ブラウザでイベント発生時

6.4 ページ状態変更時

図 8 にページ状態変更時のフローを示す．上述のイベント反映等によりマスタページの状態に変更が生じると，プロキシはページ変更に関連するすべてのブラウザに対して変更内容を含むページ変更通知を送信する．通知を受信したブラウザは，自身の保持するページに対してページ変更を反映させる．このページ変更反映は，ページに埋め込まれたページ変更機能による．

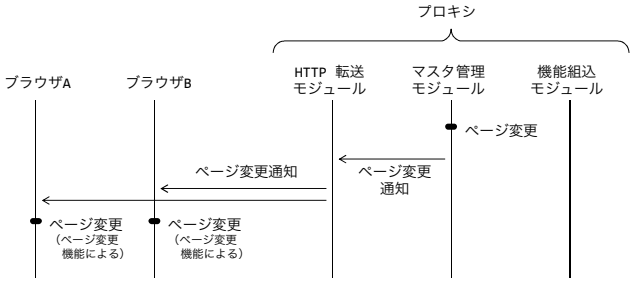

図 8：ページ状態変更時

7. 考察

プロキシ上でページに対してイベント共有機能とページ変更機能を追加することで，分割したページの状態共有が実現可能であることが確認できた．また，端末上のブラウザで生じたイベントを，その他の端末のブラウザだけでなく，プロキシが保持するマスタにも反映する．こうすることで，あとからページを取得した端末のブラウザも，過去のイベントが反映されたページを取得可能であることが確認できた．また，ページのマスタは必ずしもプロキシが保持しなければならないわけではなく，マスタへのイベントの反映とページ変更の転送が可能であることを常に保証できれば，任意のユーザブラウザがマスタを管理する役割を担ってもよい．

ブラウザにて生じたイベントは通常の Web ページと同様に即座にローカルのブラウザへ反映するため，プロキシが保持するマスタの変更を待つための遅延を含まず，応答性が損なわれることがない．

各端末へのページ変更通知として，発生したイベントそのものを通知してもよいし，あるいは，ページ変更の影響を受けたページの一部分を差分として通知してもよい．前者は通知対象が生じたイベントだけなので，イベントごとに発生する通信が低く抑えられるが，イベント反映のため各機器がページ全体を保持しておく必要がある．後者はページの一部分を通知するので通信の量は多くなるが，各ブラウザは自身の関連する部分だけを保持すればよく，ページ保持に必要な端末リソースが少ない．

本試作ではプロキシに対するすべてのアクセスをひとつのセッションとして扱うが，不特定多数のユーザが利用するサービスとして展開するためには，ページを共有するブラウザのペアリングやセッション管理が必要になる．

8. おわりに

多くの機器のディスプレイを連携させて利用する Web のスタイルであるマルチディスプレイ・レスポンシブウェブを提案した．マルチディスプレイ・レスポンシブウェブでは，従来のひとつの Web ページを複数の部品に分けて構成し，複数の機器のブラウザで異なる部品から構成されたページを表示することが可能になる．さらに，それらの異なる構成のページを開いたブラウザを連携させて，ひとつのページとして扱う．こうすることで，複数の機器のディスプレイの特徴を活かした新しい Web 体験が可能となる．システムを試作により，ページの分割とページの状態共有機能をプロキシサーバが提供することで，マルチディスプレイ・レスポンシブウェブの基本的な機能が提供できることを確認できた．

参考文献

1) 坂本典哉 : W3C での Web and TV の規格化動向, 東芝レビュー, Vol.68, No.2, pp60-61 (2013)
2) HTML5, W3C (online), available from <http://www.w3.org/TR/html5/>
3) Media Query, W3C (online), available from <http://www.w3.org/TR/css3-mediaqueries/>
4) 小林 透，瀬古俊一，川添雄彦 : HTML5 によるマルチスクリーン型次世代 Web サービス開発，翔泳社 (2013)