5N-8

Meteor.js を用いたスポーツ用リアルタイム Web アプリケーションの提案と実装

田中 佑太郎　　松原 俊一　　Martin J. Dürst
青山学院大学理工学部情報テクノロジー学科

1　はじめに

様々な分野でスマートフォンや PC のアプリケーションは Web 化される傾向がある。第 1 筆者は，大学生活を通じて全日本大学バスケットボール連盟（JUBF[1]）に所属し，組織運営に携わってきた．業務の中で最も重要なのは試合のデータ集計である．シュートの成功，失敗，選手の交代など，試合中に起こったすべての出来事を記録する．この出来事のことをスポーツ業界ではスタッツと呼ぶ．スタッツ（Stats）とは，Statistics の略で，個人成績のことを指し，その記録にはスタッツ集計システム（Statistics Gathering System，SGS）を用いる．

Web アプリケーションフレームワーク Meteor.js [1] を用いて，従来の SGS （C-SGS, Current SGS）に代わる新たな SGS（R-SGS, Realtime SGS） を提案・実装し，第 2 節で述べるデータ集計における問題点を解消することが本研究の目的である．さらにこれにより，Meteor.js の Web アプリケーション開発の簡略化による開発フレームワークとしての適性を検証する．

2　問題点と解決策

2.1　環境依存とリアルタイム性の欠如

JUBF では，スタッツの記録に Windows タブレット PC のアプリケーションである C-SGS を用いて，1 試合当たり約 600 個の出来事を記録している．記録したデータは，Web サイトに公開し，マスメディアの記事作成，チームやファンの試合分析などに役立てられている．図 1 は JUBF の公式サイト[2]に公開されている，実際の試合で集計されたスタッツの一覧（BOX スコア）である．

ところが C-SGS には問題点が二つある．一つ目は C-SGS の環境依存である．OS の違いなどの異なる環境での利用が困難である．

青山学院大学　　HC：長谷川 健志

No.	選手名	S	PTS	3P FG M	3P FG A	2P FG M	2P FG A	FT M	FT A	F	REBOUNDS OR	REBOUNDS DR	REBOUNDS TOT	TO	AS	ST	BS	MIN
0	船生　誠也			-	-	-	-	-	-	-	-	-	-	-	-	-	-	DNP
1	大峰　龍盛		0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	01:18
3	小林　通太	*	2	0	0	1	4	0	0	1	2	0	2	3	1	0	0	23:40
5	高橋　貴大			-	-	-	-	-	-	-	-	-	-	-	-	-	-	DNP
7	野本　建吾		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	01:20
8	張本　天傑	*	16	0	3	5	11	6	6	2	0	6	6	3	0	0	1	36:50
11	田中　光			-	-	-	-	-	-	-	-	-	-	-	-	-	-	DNP
12	中深迫　諒太		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00:59
13	鵜　誠司	*	6	0	0	3	7	0	0	4	1	1	2	3	0	2	1	27:50
15	山崎　将也		0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	01:50
17	溝山　拓未			-	-	-	-	-	-	-	-	-	-	-	-	-	-	DNP
18	笠井　廉平			-	-	-	-	-	-	-	-	-	-	-	-	-	-	DNP
25	永吉　佑也	*	18	0	1	7	9	4	6	3	1	5	6	6	2	1	1	39:01
32	畠山　俊樹		2	0	1	1	3	0	0	2	0	1	1	1	1	3	0	27:12
56	比江島　慎	*	13	1	3	5	13	0	0	0	0	8	8	3	2	0	0	40:00
	チーム		0	0	0	0	0	0	0	0	6	7	13	0	0	0	0	
			57	1	10	22	48	10	12	12	10	28	38	19	6	6	3	200:00

図 1: BOX スコア

二つ目の問題点は，データの集計から Web サイトへの公開までのリアルタイム性の欠如である．出場選手情報の作成から，試合終了後の Web サイト公開までに，現在，合計五つのデータ形式を用い，データを扱う端末やアプリケーションを何度も入れ替えている．その結果試合データの公開が遅くなり，報道関係者や試合会場にいない人が試合の詳細を知るのは試合終了後になる．

2.2　解決策

C-SGS の代わりとなるリアルタイム性を備えた新たな SGS である R-SGS を提案する．

バスケットボールは，1 秒で展開が大きく変わる可能性があるスポーツである．そこで本論文では「リアルタイム」をデータベースに情報が挿入されると，約 1 秒以内に別のクライアントにその結果が表示されることと定義する．

Meteor.js では，サーバとクライアントが連携するリアルタイム Web アプリケーション開発が可能である．R-SGS を Web アプリケーションとして実装することで環境依存を解消し，開発には Meteor.js のリアルタイムページ更新機能を適用することでリアルタイム性の欠如を解消する．Web で一括して情報管理をすることによって，デバイス入れ替えの際の時間的遅延なども解消される．

3　Meteor.js

Meteor.js は 2012 年 4 月に公開されたオープンソースのリアルタイム Web アプリケーション開発フレームワークである．

Proposal and Implementation of a Real-time Web Application for Sports Using Meteor.js
Yutaro Tanakai, Shunichi Matsubara, and Martin J. Dürst
Department of Integrated Information Technology, College of Science and Engineering, Aoyama Gakuin University
5–10–1 Fuchinobe, Sagamihara, Kanagawa 229-8558, Japan
yutaro@sw.it.aoyama.ac.jp,
{matsubara, duerst}@it.aoyama.ac.jp

[1]Japan University Basketball Federation
[2]http://www.jubf.jp/

3.1 リアルタイムのページ更新

Meteor.js は WebSocket を活用することで，サーバとクライアント間の高速な双方向通信を実現している．それを利用し，コードやデータの変更を検知して HTML コードを再構築し，クライアントでの再描画を自動的に処理している．複数のクライアントに対してもリアルタイム更新が可能である [1] [2].

3.2 データベースとクライアントキャッシュ

Meteor.js は，サーバ側のデータベースとして MongoDB[3] を採用している．クライアント側ではキャッシュを備えている．クライアント用のデータアクセス API は Mongo API に準じており，サーバ用のコードとクライアント用のコードで，同じ API を使ってデータにアクセスできる．図 2 に，Meteor.js におけるデータフローを示す．クライアント側のキャッシュには，サーバ側のマスターデータのコピーが格納され，自動的に同期される仕組みになっている．

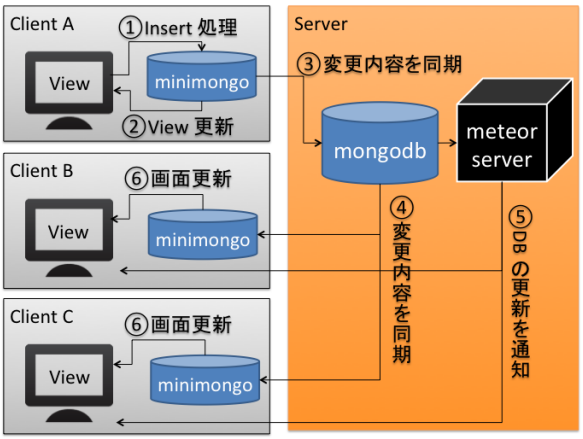

図 2: Meteor.js におけるデータフロー

3.3 Web アプリケーション開発の効率化

Meteor.js のアーキテクチャは，開発者の負担を軽減することを目的に組み立てられている．クライアントとサーバは，どちらも JavaScript で記述できる．サーバ側では Ruby や Java などを使う必要がなく，クライアントとサーバのコードを共有できる．また，パッケージ管理機能もある．実運用のサーバもあらかじめ準備され，コマンド 1 行でアップロードでき，さらにその際には，JavaScript や HTML ファイルが自動的にミニファイされる．

4 R-SGS の実装

スタッツデータ集計機能（図 3）と，チームと選手登録機能が主な機能である．その他，play by play 表示機能，集計データを用いた試合の流れの可視化機能も実装した．

[3]http://www.mongodb.org/

システムは，Team，Game，Ranking の三つのタブで構成されている．Meteor.js では，プロジェクトの中に作成できる html ファイルは一つだけであり，リンクによるページ遷移は不可能である．そのため，CSS を用いて擬似的なページ遷移を実装している．

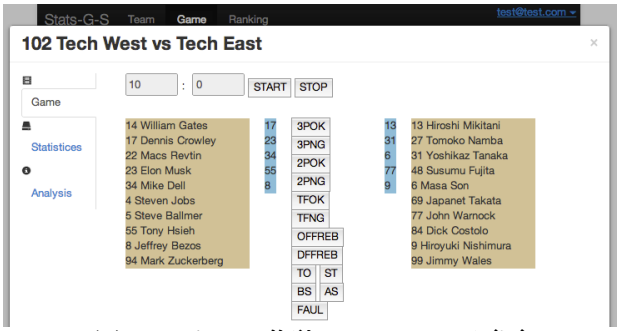

図 3: スタッツ集計システムの一画面
(選手の名前などは架空のもの)

本システムは，メールアドレスとパスワードでアカウント管理をしている．ログインしていない状態では，閲覧のみ可能である．特定のアカウントの場合はデータの追加，編集が可能になる．Web ブラウザが搭載されたデバイスさえあれば，R-SGS はどこでも使用できる．

5 まとめ

Meteor.js を用いた Web アプリケーションの実装によって，C-SGS の環境依存が解消でき，リアルタイムの情報公開が可能になった．本研究は，バスケットボールを対象に進めたが，リアルタイムデータが必要な他のスポーツにも応用できる．

また，リアルタイム通信の実装や MVC モデル化[4]，実運用環境のセットアップなどを Meteor.js のアーキテクチャが吸収することで，本来必要な作業を省略して開発ができた．

Meteor.js では，HTML，JavaScript，CSS を中心に開発をするので，Ruby on Rails などのフレームワークと比較して，WebAPI を作成するのが難しい点が今後の課題として挙げられる．

参考文献

[1] Meteor Development Group. Meteor. `http://www.meteor.com/`, 2012 年 1 月 8 日閲覧.

[2] 白石俊平. 体感！JavaScript で超速アプリケーション開発 － Meteor 完全解説. `http://gihyo.jp/dev/serial/01/meteor`, 2012 年 1 月 8 日閲覧.

[4]Model，View，Controller