

KEK 入射器における WebSocket を用いた運転情報表示パネルの開発 DEVELOPMENT OF OPERATION INFORMATION DISPLAY PANEL USING WEBSOCKET AT KEK INJECTOR LINAC.

工藤拓弥^{#, A)}, 草野史郎^{A)}, 佐藤政則^{B)}, 古川和朗^{B)}
Takuya Kudou^{#, A)}, Shiro Kusano^{A)}, Masanori Satoh^{B)}, Kazuro Furukawa^{B)}

^{A)} Mitsubishi Electric System & Service CO., LTD.

^{B)} High Energy Accelerator Research Organization

Abstract

In the KEK injector Linac, the operation information has been displayed on the Web since 1994. The system developed in PHP or CGI cannot be displayed in real time. For this reason, we develop a new operation information display panel using WebSocket. New panel can be used to display real-time operation information. In this paper, we present the status of the new operation information panel in detail.

1. はじめに

KEK 入射器 (以下, 入射器) の, 主な機器情報や運転情報は, 利用者の利便性を考慮し Web で配信している。この情報は, 定期的に生成される静的なものか, CGI や PHP を用いて生成される動的なものである。これらは, リアルタイム性に欠けるため, 機器の速い変化に対応した表示が困難であった。そのため, Web サーバおよびクライアント間での双方向通信が可能な WebSocket プロトコルを用いて, リアルタイム性の高い運転情報表示パネルの開発をおこなった。本稿では, 新規開発中の運転情報表示パネルの詳細について報告する。

2. 既存の Web 情報

各加速器機器の操作および表示には, Python/Tkinter あるいは Tcl/Tkinter を用いて開発した, X-Window システム上で動作するパネルが用いられてきた。これらのパネルを使用するには, 当然ながら X-Window システムの動作環境を整備する必要がある。しかしながら, 各機器担当者は居室において機器情報を参照したい場合が多く, そのためだけに X-Window システムの動作環境を整備するのは, 大きな負担であった。そのため, 入射器においては, 動作環境整備などの負担が少ない Web を利用して, 各機器情報の配信をおこなっている。

Web で配信する情報は, Linux/Unix 計算機汎用のスケジューラーである Cron を用いて定期的に HTML を生成する, あるいは CGI および PHP を用いて動的な生成をおこなっている。しかし, これらの情報はリアルタイム性が低く, 機器の速い変化情報を表示することが困難であった。

リアルタイム性の高い Web 情報として, クライアント側に Java applet, 通信部分に CORBA を使用したシステムを検討¹⁾したが, CORBA でのシステム開発およびメンテナンスが困難となり, 導入を見

送った。現在, 新たな試みとして, Ajax と呼ばれる JavaScript による非同期通信を用いたビーム軌道情報の配信をおこなっている。約 1 秒周期での更新は実現できているが, サーバ, クライアントともに負荷が高く, 多くの機器情報を配信するには至っていない。

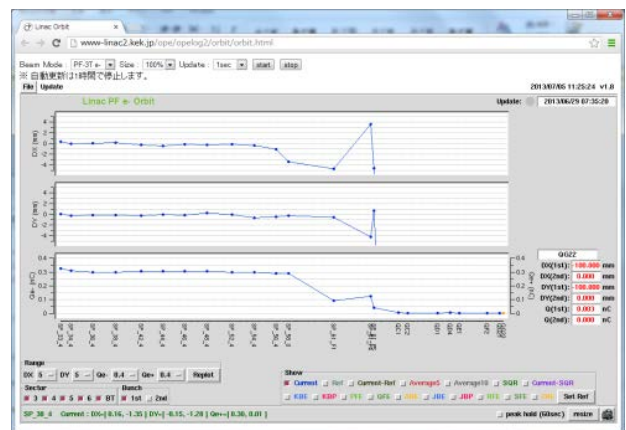


Figure 1: Web application using Ajax.

3. 新運転情報表示パネルの概要

3.1 Web アプリケーションによる構築

新運転情報表示パネルは, 最新の Web 技術を用いた Web アプリケーションとして構築されている。Web アプリケーションは, 対応するブラウザが動作可能な計算機環境であれば OS の種類を問わず動作可能である。また, ブラウザ上で動作するため, X-Window システムの動作環境整備など, 従来クライアント側でおこなう必要のあった環境設定が不要となり, 利用者の利便性を大きく向上させた。

2.1 WebSocket プロトコルの採用

リアルタイム性の高い Web アプリケーションには, サーバ側からクライアント側へデータを送信するサーバプッシュを実現することが必要である。近

[#] kudoh@post.kek.jp

年の Web 技術では、JavaScript による非同期通信を用いて定期的にサーバと通信をおこなう Ajax と呼ばれる技術や、Ajax を用いて擬似的にサーバプッシュを実現する Comet と呼ばれる方法により、リアルタイム通信を擬似的に実現することが可能である。しかしながら、これらの手法では、短い間隔での定期的な通信が必要であるため、サーバにかかる負荷が高くなってしまふ。

この問題を回避するため、サーバおよびクライアント間において、効率的な双方向通信を実現するための仕組みである WebSocket を採用することとした。WebSocket は、サーバとのコネクションを確立すると、以降の通信はそのコネクションを用いておこなうため、Ajax や Comet のように接続処理を繰り返しておこなう必要が無く、サーバへの負荷を抑えることができる。また、双方向通信をおこなうため、サーバプッシュを容易に実装することが可能である。

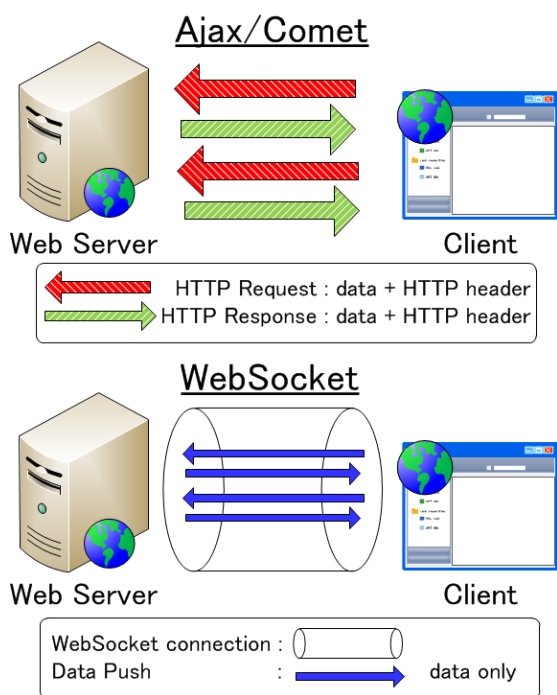


Figure 2: Comparison between Ajax/Comet and WebSocket.

4. 新運転情報表示パネルの構成

4.1 サーバサイド

新運転情報表示パネルのサーバサイドは、オープンソースのサーバサイド JavaScript アプリケーションフレームワークである Node.js^[2]およびライブラリである Socket.IO^[3]を用いて開発した。通常のサーバサイドアプリケーションは、接続ごとに1プロセスを消費するため、大量の接続があった場合はリソース不足に陥り、性能が低下する。Node.js は、Figure3 に示すような非同期 I/O を用いて1つのプロセスで多数の接続を効率良く処理するため、接続数が増加した場合においても性能が低下しにくいという利点がある。また、近年顕著であるスマートフォンの急速な普及と非 Flash 対応デバイスなどの出現

により、JavaScript の需要性が増してきており、JavaScript を用いたサーバサイドプログラムの実行が可能で Node.js は、今後の発展が非常に期待できる。

近年、入射器制御システムでは、Experimental Physics and Industrial Control System (EPICS) の導入が精力的に進められており、入射器の全パラメータは、EPICS 経由での制御が可能となっている。

Node.js から EPICS への通信には、理化学研究所・内山暁仁氏により開発された NodeCA アドオン^[4]を使用した。このアドオンは、EPICS 通信プロトコルである Channel Access (CA) の基本的な機能の caGet, caPut, caMonitor が実装されている。

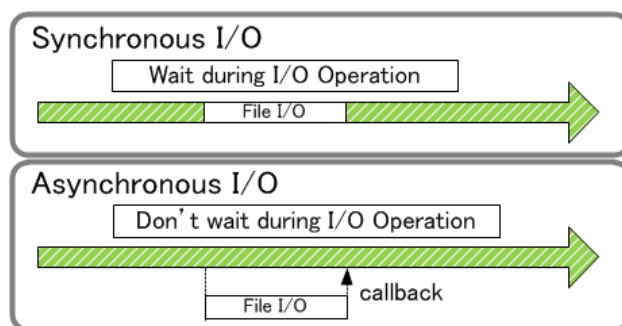


Figure 3: Asynchronous I/O.

4.2 クライアントサイド

クライアントサイドにおいては、HTML を用いて記述したページを基に、サーバサイドから受信したデータに応じて動的に DOM (Document Object Model) 操作することにより、リアルタイムな表示を実現している。DOM 操作には、オープンソースの JavaScript ライブラリである JQuery^[5]を用いている。EPICS IOC (Input/Output Controller) の情報変化から、クライアントサイドの表示を更新するまでの概略を Figure 4 に示す。

JQuery は、JavaScript の仕様解釈の違いに起因する Web ブラウザごとの挙動の違いを吸収してくれるため、クロスブラウザなパネルを簡便に実現することが可能であるのみならず、多数のプラグインが公開されており、迅速なアプリケーション開発が可能である。

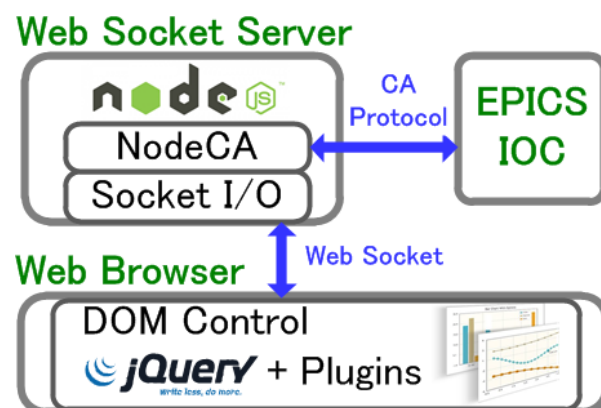


Figure 4: Configuring the server.

4.3 新パネルの画面構成

新パネルは、一目で現在の運転状況が把握できるように、複数の情報を画面内に表示しており、運転概要表示部、放射線モニタおよび電荷制限装置表示部、各機器が配置されたレイアウト図表示部から構成されている。

運転概要表示部は、入射器で導入している PostgreSQL をバックエンドとした電子ログブックシステム^[6]と連携し、現在の運転状況の概要を表示している。この情報は、オペレータが運転の状況を見て電子ログブックシステムに手入力する情報であるため、不定長である。そのため本パネルでは、画面レイアウトを崩さないように JQuery のプラグインを用いて横にスクロールさせて表示している。

放射線モニタおよび電荷制限装置表示部には、JQuery のプラグインである JqPlot^[7]を用いてグラフ化している。データ更新時には、アニメーションを伴ってグラフが再描画されるため、閲覧者はデータ更新を意識しやすくなっている。

レイアウト図表示部は、機器の異常時には、異常機器の表示部が赤く点滅するため、閲覧者は異常に気付きやすい。また、各機器表示部は実入射器の配置を意識して描かれており、迅速な異常対処の一助となることを期待している。

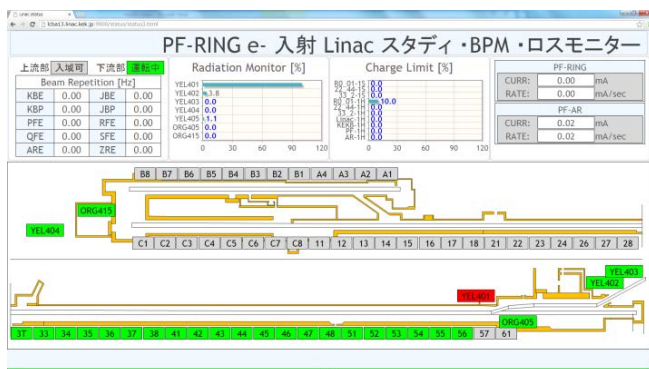


Figure 5: New panel under development.

5. 課題

5.1 セキュリティリスクの解消

本パネルの表示リアルタイム性は、X-Window 上で動作するものと比べても遜色がない。サーバサイドは、CA による値の変更である caPut も実装しているため、本アプリと同様の構成で、各機器の操作パネルを作成することも技術的に可能である。

しかし、本構成は Web アプリケーションであるため、X-Window 上で動作するパネルに比べ、悪意ある操作者に不正な操作をされるリスクが高く、操作パネルの開発/導入には至っていない。現在セキュリティリスクを排除すべく、SSL を用いたサーバ、クライアントの相互認証や、利用者認証の仕組みを検討中である。

5.2 情報視覚化方法の検討

本パネルは、入射器棟制御室に新設された 50 イ

ンチの大型ディスプレイにおいても表示する予定である。天井から吊るされた大型ディスプレイは、閲覧者から数メートル以上離れており、フォントサイズを大きくしなければ視認することが困難である。大きいフォントサイズでも、より多くの情報を表示するためには、異常があった個所をズームして表示するなど、情報の重要度に応じた視覚化をおこなうことが必要である。現在、HTML5 を基盤としたリッチコンテンツ開発用フレームワークである CreateJS^[8]を用いて、より良い情報視覚化方法を検討中である。

6. まとめ

入射器では、機器情報や運転情報を Web 配信するため、CGI や PHP にて開発された Web アプリケーションの運用をおこなってきた。しかしながら、これらの Web アプリケーションは、リアルタイム性が低く、機器の速い変化を表示できないなどの問題もあった。

このため、サーバおよびクライアント間通信に WebSocket プロトコルを用いた新運転情報パネルを開発した。本パネルは、HTML で記述されたページを、サーバから受けた情報に応じて動的に操作することにより、入射器ビーム運転および機器情報のリアルタイム表示を実現することが可能である。本パネルは、現在運用試験を重ねており、2013 年秋には通常運転に導入する予定である。

7. 謝辞

本開発を進めるにあたり、貴重な助言をいただきました。NodeCA アドオンの開発者である 理化学研究所・内山暁仁氏に感謝致します。

参考文献

- [1] S.Kusano et al., "Real-time Display of Accelerator Status Using JAVA and CORBA", Proceedings of the PCaPAC'99, Tsukuba, Jan.1999, KEK-Proceedings 98-14
- [2] <http://nodejs.org>
- [3] <http://socket.io>
- [4] A.Uchiyama et al., "DEVELOPMENT AND IMPLEMENTATION OF EPICS CHANNEL ACCESS CLIENT WITH REAL-TIME WEB USING WEBSOCKET", Proceedings of the 9th Annual Meeting of Particle Accelerator Society of Japan, Osaka, Aug. 8-11, 2012
- [5] <http://jquery.com>
- [6] T.Kudou et al., "UPGRADE OF ELECTRONIC LOGBOOK SYSTEM AT KEK INJECTOR LINAC", Proceedings of the 8th Annual Meeting of Particle Accelerator Society of Japan, Tsukuba, Aug. 1-3, 2011
- [7] <http://www.jqplot.com>
- [8] <http://www.createjs.com>