Web 技術を用いた EPICS Channel Access から HTTP への単方向ゲートウェイの提案

PROPOSAL FOR A UNIDIRECTIONAL GATEWAY FROM EPICS CHANNEL ACCESS TO HTTP USING WEB TECHNOLOGIES

山田秀衛 *
Shuei YAMADA*
High Energy Accelerator Research Organization (KEK) / J-PARC Center

Abstract

Control systems for large accelerators such as J-PARC, KEKB and RIBF are built using EPICS. EPICS is a framework for building a networked distributed control system, but the Channel Access (CA) protocol used in EPICS is assumed to be used in the internal network for accelerator control. On the other hand, there is a demand to acquire the status of various equipment related to accelerator operation from networks outside the institute, such as at home or other institute, whilst it shall be prohibited to operate accelerator equipment from outside the institute. In this paper a unidirectional gateway from CA to HTTP using web technology is proposed, as well as a web application which is available even from tablets and smartphones.

1． はじめに

J-PARC では 2015 年から web ページを用いた加速器の運転状況を所内 LAN(JLAN と呼ばれる) に提供している。Figure 1 に web ページのスクリーンショットを, Fig. 2 に加速器制御 LAN から JLAN への情報の経路を示す。加速器制御 LAN と JLAN は直接接続されてはおらず, 間に DMZ を介している。また, 加速器制御 LAN と DMZ, DMZ と JLAN の間にはファイアウォールが設置されている。制御 LAN から JLAN に提供するのは画像ファイルであること, web サーバは読み取り専用でファイルサーバにアクセスすることで, 制御 LAN に接続されている加速器の機器を JLAN から操作することがないようにしている。この web ページはあくまでも要約された情報でしかない点, 1 分に 1 回程度しか更新されない点, 画像データであるために数値を利用しにくい点, スマートフォン等の画面の小さな端末からは見にくい点などにおいて不満があった。

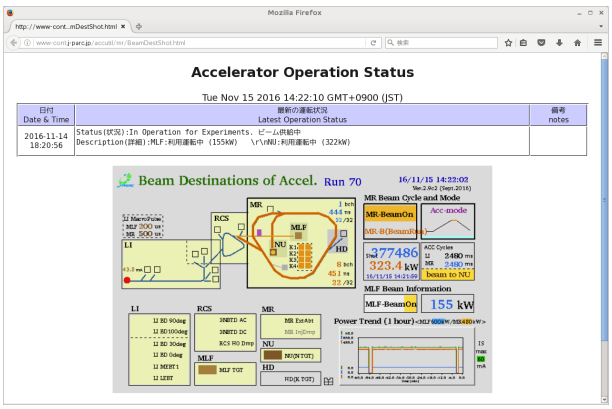

Figure 1: A web page which shows J-PARC accelerator status.

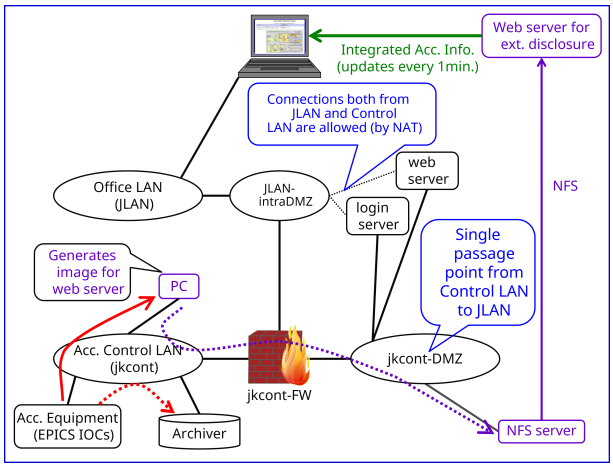

Figure 2: Initial pathway of data transmission based on files.

2018 年には, J-PARC 加速器が制御システムに採用している EPICS [1] の Channel Access (CA) プロトコル [2] を加速器制御 LAN から JLAN へ転送するゲートウェイシステムを構築した [3]。Figure 3 に制御 LAN から JLAN への情報の経路を示す。加速器を JLAN から操作することを確実に禁止するため, 制御 LAN-DMZ 間を中継するゲートウェイと DMZ-JLAN 間を中継するゲートウェイの 2 段構成とし, いずれのゲートウェイも読み込みアクセスのみを許可するようにした。さらに, これらゲートウェイ間は CA 以外では直接通信ができないようファイアウォールで制限した。これにより JLAN に接続された居室の PC でも加速器の運転に用いられるものと同じ上位制御系アプリケーション [4, 5] をインストールして利用することが可能になった。しかしながらスマートフォンやタブレットといった端末から利用するのは困難であった。また、CA プロトコルを所外のネットワークに流すのは不適切であるため, 自宅や出張先といった所外から利用するのも困難であった。

現在, PC からスマートデバイスまでの様々な端末上で

* shuei@post.kek.jp

利用可能であり, かつ別途プラグインやサードパーティのソフトウェアをインストールしなくとも利用可能なソフトウェアは Web ブラウザである。EPICS 界隈において web ブラウザ上で上位制御系アプリケーションを実行する試みの歴史は 15 年以上前に遡り, [6] に挙げられているだけでも 10 を超える。これまでに様々な web 通信技術・web サーバ側開発言語・クライアント側 web 技術の組み合わせが提案されており, web ブラウザから加速器を操作可能なものもあれば読み込み専用なものもある。また, 精力的な開発・保守が継続されているものもあれば, 既に開発が放棄されてしまったものあり, どれか一つを選択したとしても今後の動向は明らかではない。

そこで CA を HTML ストリームへ変換するゲートウェイを独自に構築し, J-PARC 加速器の詳細な運転情報をどこからでも・どんな端末からでも利用可能な web アプリケーションの可能性を検討することにした。

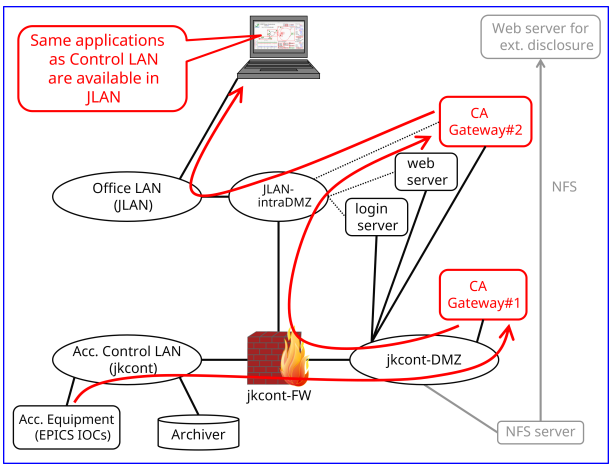

Figure 3: Transmission of data using CA protocol through two CA gateways.

2． EPICS と CHANNEL ACCESS

EPICS はネットワーク分散型の制御システムを構築するためのフレームワークで, いわゆる出版-購読型のネットワーク通信モデルを採用している。EPICS では制御点を Process Variable (PV) あるいはレコードと呼ぶ。クライアントとサーバの間では CA プロトコルを用いて PV を読み書きする。CA は C/C++ 言語で記述されたランタイムライブラリとして実装されている。上位制御系アプリケーションや, I/O Controller (IOC) と呼ばれるフロントエンド計算機を開発したり利用したりするだけなら, EPICS の利用者が CA の詳細を目にすることはない。

通常は IOC が CA のサーバとなり、クライアントである上位制御系アプリケーションに制御対象機器への I/O アクセスを提供する。クライアントが IOC から機器の状態を取得する場合, クライアントはまず IOC への接続を確立し、その後は IOC から値が通知されるのを待ち受ける。IOC が個々の PV の値を取得するには機器からの割り込みまたはポーリングを用いるが, クライアントから見ると IOC からは非同期に PV の値を通知される。また, IOC の再起動やネットワーク障害等でクライアントと IOC の接続が切断された場合には, クライアントが自動的に IOC への再接続する。

3． ゲートウェイに用いるプッシュ技術の選択

2 節で述べたように CA による通信は非同期な出版-購読型である。Web ブラウザからは Web サーバとして動作し, IOC に対しては CA のクライアントと動作するようなゲートウェイを考えたとき, web サーバから web ブラウザへはプッシュする技術を使うのが自然であろう。

一般的な web ブラウザ[1]で利用可能なプッシュ技術としては

- ロングポーリング
- Server-Sent Events
- WebSocket

が挙げられる。本稿で提案するゲートウェイでは Server-Sent Events (SSE) を採用したが, 以下にこれらの技術の概要と選択の理由を示す。

3.1 ロングポーリング

ロングポーリングは HTTP 通信上で疑似的なプッシュ機構を実現する手法である。通常の HTTP 通信の場合, クライアントが web サーバに情報をリクエストするとサーバは直ちにレスポンスを返して HTTP 通信を完了し接続を切断する。一方ロングポーリングの場合は, クライアントが web サーバに情報をリクエストしても サーバはすぐにはレスポンスを返さず, 接続を維持したまま保留する。サーバに新しい情報が現れたらサーバはクライアントにレスポンスを返し, HTTP 通信を完了して接続を切断する。サーバからの応答を受け取ったら, クライアントはサーバに新たにリクエストする。

ロングポーリングを採用した場合,

- サーバからの応答を受け取って HTTP 通信が完了してから新たにリクエストするまでの間に更新された情報を取りこぼす可能性がある
- HTTP 通信では同時に利用できる接続数に制限があるため, 複数の PV からの情報を単一の接続に重畳する必要がある。HTTP 通信が完了するたびに重畳される全ての PV に対して再接続する必要がある
- HTTP 通信の接続と CA 通信の接続をそれぞれ独立して管理しようとすると, ゲートウェイの設計が複雑になる

ことから, 採用を見送ることとした。

3.2 Server-Sent Events (SSE)

SSE は web サーバから断続的に HTTP 通信でデータを送信し続ける技術で, HTML5 で API が標準化されている [7]。通信に HTTP/1.1 を利用するところは通常の HTML 通信と同様であるが, メディアタイプに Content-Type:text/event-stream を使用し, web サーバは通信を切断することなくデータを送信し続ける。

JavaScript にもイベント駆動型の API が用意されており, 一般的な Web ブラウザから利用可能である。クライ

[1] ここでは Firefox, Safari, Chrome, Edge を指す。Internet Explorer は含まない。

アントは一度だけ web サーバにリクエストし, その後はサーバから断続的に送信されるデータを随時受信することができる。Web サーバからクライアントへの単方向通信で, データはテキストのみである。データの形式として JSON を使うことで複雑なデータ構造を記述することが可能であり, クライアント側の JavaScript への受け渡しも容易である。SSE ではイベントに名前を付けることが可能で, 単一の HTTP ストリーム上で web サーバから異なる種類のイベントを送信することができる。接続が切れたらクライアントが自動的に再接続する。接続を終了するにはクライアント側で明示的に切断するか, サーバ側で接続を拒否する必要がある。

3.3 WebSocket

WebSocket は HTML5 の一部として提案されたプロトコルである [8]。HTTP と同じ 80 番・443 番ポートを使って TCP 上で双方向通信を可能にする技術であるが, HTTP とは異なるプロトコルである。

しかしながら,

- 必ずしも双方向通信は必要ない
- Web サーバのほかに WebSocket サーバが必要
- JLAN・加速器制御 LAN のファイアウォールやプロキシサーバを混乱させる可能性がある

といった点から WebSocket の採用は見送ることとした。

4. ゲートウェイの実装

ゲートウェイは導入済みの Web サーバ資源を活用すべく, camonitor を流用して CGI プログラムとして実装した。camonitor は CA ライブラリの用例とも言える EPICS の CLI アプリケーションで, コマンドライン引数にひとつ以上の PV 名を渡して実行すると, 各 PV の値が更新されるたびに PV 名とその値を標準出力に表示し続ける。出力を Server-Sent Events の名前付きイベントの形式に整形して CGI プログラムとして実行すれば, 最小限の変更で CA から HTTP へのゲートウェイとして動作させることができる。

Listing 1 に, ゲートウェイが HTTP ストリームに流す名前付きイベントの例を示す。イベントの名前を指定する “event” フィールドを PV 名とすることで, 値が更新された PV をクライアント側で容易に判定できるようにした。“data” フィールドには PV の値のほかにタイムスタンプや PV の状態などを示す属性等を JSON 形式で記述した。

Listing 1: Example of named events representing structure of a PV.

```
event: EPICS:PV_NAME1
data:{"time":"2022-09-06 12:34:56",
      "type":"DBR_TIME_DOUBLE",
      "val":0.938,
      "stat":"HIHI",
      "sevr":"MAJOR"}
```

Figure 4 に加速器制御 LAN から JLAN へ, CA から HTTP へのゲートウェイを経由した情報の経路を示す。ゲートウェイには PV への書き込みを実装しないこと, またゲートウェイは読み込み専用の CA ゲートウェイから PV の値を取得するようにすることで, web ブラウザから制御 LAN に接続されている加速器の機器を操作することがないようにしている。

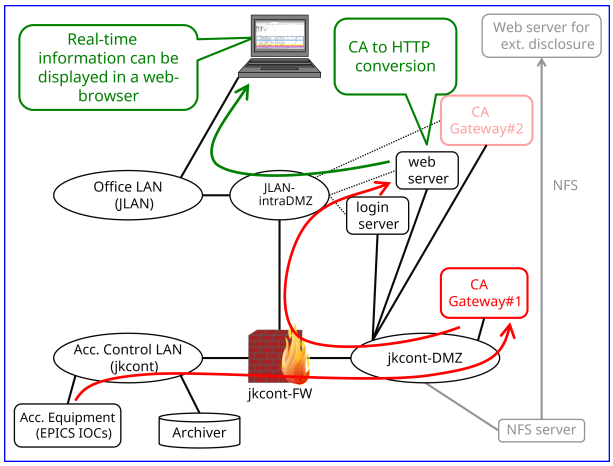

Figure 4: Transmission of data through CA to HTTP gateway.

5. ウェブアプリケーションの試作

Figure 5 は camonitor と同様の機能を持った試作 web アプリケーションのスクリーンショットである。テキストエリアに値を取得したい PV 名を入力して画面内のボタンを押すと各 PV の値が表示され, リアルタイムに更新される。

Listing 2 は, クライアント側で実行される JavaScript コードの断片である。クライアントはまずテキストエリアに入力された PV 名をクエリ文字列としてゲートウェイの URL に渡して HTTP 接続を確立する。Server-Sent Events でサーバが通知してくるイベント名に対応するコールバック関数を登録し, コールバック関数では Web ページ内の適切な HTML 要素を動的に書き換えるようにしておく。ここで受け取るイベントはすなわち PV の情報であるから, JavaScript がイベントを受け取るたびに web ブラウザ上で表示されている PV の値や矩形領域の色が更新されることになる。

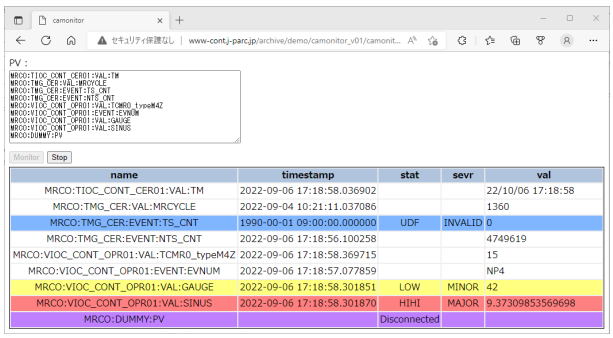

Figure 5: A preliminary web-application displaying values of several PVs.

Listing 2: Code fragment of JavaScript that connects to the gateway URL and waits for named events in a handler.

```
mon = new EventSource(url);
mon.addEventListener("EPICS:PV_NAME1",
    (event) => {
        data=JSON.parse(event.data);
        id1.innerHTML=...
    }
);
```

6． まとめと課題

EPICS で用いられる CA プロトコルから HTML のストリームに変換するゲートウェイを実装し, 一般的な web ブラウザ上で PV の値の変化をリアルタイムにテキストとして表示するアプリケーションを試作した。PV の値を視覚的に表示できる表示灯やメーター, あるいはトレンドグラフといったウィジェットの開発と、これらウィジェットを利用するのに適した JavaScript の API の定義が今後の課題である。

また, J-PARC では 100 を超える上位制御系アプリケーションを加速器の運転に利用している。これらのアプリケーションを web 版に移植するのもマンパワーの観点において困難が予想される。既存の画面を web 版へと自動的に変換する技術の開発が望まれる。

参考文献

[1] EPICS - Experimental Physics and Industrial Control System; http://epics.anl.gov/
[2] Channel Access; https://epics-controls.org/resources-and-support/documents/ca/
[3] S. Yamada, “Real-time and Detailed Provision of Accelerator Operation Information from the J-PARC Accelerator Control LAN to the J-PARC Office LAN”, Proceedings of the 15th Annual Meeting of Particle Accelerator Society of Japan, WEP101, pp.613 (2018).
[4] CSS – Control System Studio; http://controlsystemstudio.org/
[5] S. Yamada *et al.*, “Deployment of Control System Studio at J-PARC Main Ring”, Proceedings of the 8th Annual Meeting of Particle Accelerator Society of Japan, WEP103, pp.543 (2015).
[6] https://github.com/JeffersonLab/epics2web/wiki/Similar-Projects
[7] HTML Standard, 9.2 “Server-Sent Events”; https://html.spec.whatwg.org/multipage/server-sent-events.html
[8] RFC6455 “The WebSocket Protocol”; https://www.rfc-editor.org/rfc/rfc6455