# J-PARC 加速器 PPS 情報収集・表示系の更新(II):GUI の自動生成

# UPGRADE OF ACQUISITION AND MONITORING SYSTEM FOR J-PARC ACCELERATOR PPS(II) : AUTOMATIC GENERATION OF GUI

渡邉和彦#, A), 仁木和昭 B), 高橋博樹 C), 山本昇 B), 地村幹 C), 福田真平 A)
Kazuhiko Watanabe#, A), Kazuaki NikiB), Hiroki TakahashiC), Noboru YamamotoB), Motoki ChimuraC), Shinpei FukutaA)
A) Mitsubishi Electric System & Service Co., Ltd.
B) High Energy Accelerator Research Organization
C) Japan Atomic Energy Agency

*Abstract*

The monitoring of J-PARC Personnel Protection System (PPS)[1] is performed using by acquisition and monitoring system (PPS Data System[2]), which consists of data acquisition PCs and information display PCs installed in Central Control Building, and PLCs distributed in the three Accelerators and Central Control Building by an independent network for PPS (PPS Network). The PPS Data System is being upgraded to an EPICS[3]-based system in phases from FY2022. In this issue, we report mainly on a major GUI update by EPICS. Control System Studio(CSS)[4] is used to create the GUI. Therefore, we considered creating and using a program to perform automatic generation. On the other hand, there is no information on where to place display objects on the display screen, which is necessary for automatic GUI generation. Therefore, we devised a method to automatically generate display objects for each signal, place them on the screen, extract their coordinates, link them to signal information, and generate a GUI. In this presentation, we report on the automatic generation of the GUI and the operation check of the generated GUI.

## 1. はじめに

J-PARC 加速器の Personnel Protection System(以下、PPS という)では、中央制御棟に設置されたデータ収集用 PC 群と情報表示 PC 群で構成される情報収集・表示系(以下、PPS Data System という)と、3 つの加速器施設及び中央制御棟に分散配置された PLC 群を、PPS 用の独立したネットワーク (PPS Network)により接続することで、中央制御棟でのデータ表示、収集を行っている。

PPS Data System では J-PARC 加速器の運転開始当初から横河電機製 SCADA ソフトウェア「ASTMAC」(以下、ASTMAC という)を使用してきたが、サーバ 1 台当たり 1000 点という信号点数の制限がある為、多くの信号を監視対象から外している。そのような状況にありつつ、加速器の状況に合わせて信号は増加する為、対応が難しくなってきている。また、ASTMAC には対応 OS が Windows7 までという制限があることや販売が終了していることもあり、別ソフトウェアによる新たな PPS Data System の構築が必要となった。そこで J-PARC PPS Gr.では J-PARC 制御システムでも長年用いられた実績がある EPICS を採用して、信号点数の制限も、OS の制限もない新システムを構築することとした。そして、新システムに移行するにあたり、OPC[5]デバイスサポートを用いた EPICS IOC を採用することにより、旧システムの持つ OPC サーバ機能を活用して ASTMAC の持つ情報を EPICS の世界で使用可能とすることで、新旧システムを共存させ、旧システムを止めることなく、新システムの開発・動作試験を可能にした。新システムへの移行は 4 つの段階に分けて移行する計画となっている。まず、第 1 段階では OPC デバイスサポートを用いた EPICS IOC の構築を行う。続いて第 2 段階では Control System Studio (以下、CSS という)を用いて GUI を作成。第 3 段階では Archiver Appliance を用いたロギングシステムの構築。最後の第 4 段階では NetDev[6]を用いた IOC を構築し、PLC から直接情報を取得するよう変更する。以上の 4 段階の経ることで ASTMAC から EPICS へ完全に移行する。今回は第 2 段階をメインに報告する。

# kaz@post.j-parc.jp

## 2. これまでの PPS Data System GUI について

これまでの PPS Data System では、ASTMAC の機能を用いて作成した GUI を使用してきた。ASTMAC の GUI を Fig. 1 に示す。

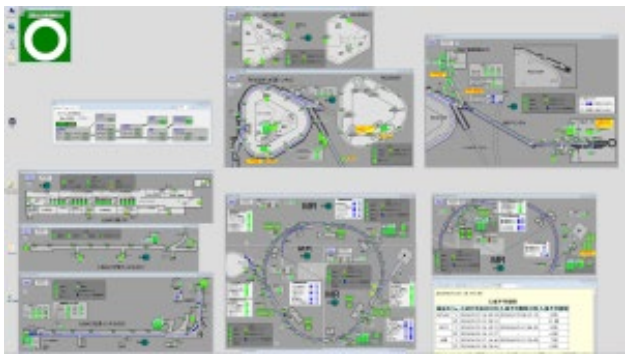

Figure 1: ASTMAC GUI.

ASTMAC には WYSIWYG な GUI 開発用アプリケーションが備わっており、オブジェクトを貼り付けていくだけで GUI を作成できる。また、オブジェクトと表示したい信号との紐付けも、一覧から信号名を選択してプロパティに設定するだけで良く、PLC モジュールの種類やアドレ

スなどを気にする必要はない為、非常に簡単に GUI を作成することができる。

一方で、表示する信号点数が多くなると WYSIWYG な操作が逆に煩わしくなってくる。新規の信号を追加する際に、既存の信号で類似のタイプの信号があればオブジェクトのコピー＆ペーストは可能だが、信号との紐付けは信号リストから信号を探して選択し、プロパティに設定するという操作を行う必要がある。しかも、どのオブジェクトにどの信号が紐付けられているかは、オブジェクトを選択してプロパティを確認する必要がある為、一度に多数の信号を追加しようとすると設定ミスが起きやすいという欠点もあった。さらに、1 画面当たりの推奨オブジェクト数制限があり、画面を分割せざるを得なかった。

その為、次期システムの GUI 作成の為のプラットフォームとしては、多数の信号を用いるのに耐えうる GUI であり、また、作成もデバッグも容易に行えるものが望ましい。

## 3. 次期 PPS Data System GUI について

PPS Data System を ASTMAC から EPICS 環境に移行する為、EPICS を用いて情報表示が行えるソフトウェアで GUI を作成しなおす必要がある。また、新しい GUI 作成ソフトウェアは OS や信号点数の制限がないものが望ましい。

EPICS 環境で使用されている GUI 作成用のツールとしては J-PARC で広く普及してきた Control System Studio(以下、CSS という)がある。

CSS は統合開発環境 Eclipse をベースとして開発された EPICS 向けツールキットで、OS が Windows に限定されず、信号点数の制限がなく、ASTMAC と同様に WYSIWYG なユーザーインターフェイスを備えている。ASTMAC とは異なり、作成したファイルは XML 形式となっている為、テキストエディタで直接編集することも可能である。

XML 形式であることにより、PPS のどの信号にどのオブジェクトを割り当てるかが定義できれば、オブジェクトを自動的に生成するプログラムを作成することにより、工数を大幅に削減することができ、その後のメンテナンスも容易になるのではないかと考えた。そこで、次期 PPS Data System 用の GUI 作成ツールとして CSS の Ver.4.6 を採用することとし、自動生成手法の検討を行った。

## 4. オブジェクトの自動生成

CSS での GUI 作成作業とは多くの場合、オブジェクトの選択、EPICS レコード名との紐付け、サイズと位置の指定である。今回作成したい PPS 用の GUI に限れば、オブジェクトの種類とサイズは信号の種類によって固定される。つまり信号の種類からオブジェクトの種類を決定し、信号情報からオブジェクト生成に必要な情報をオブジェクトのプロパティに埋め込むという作業をプログラムとして作成すれば、信号リストをベースにオブジェクトを自動生成することができる。

プログラムを作成する前に、まず、PPS Data System で使用したいオブジェクトを CSS で作成する。それをテキストエディタで読み込み、オブジェクトの記述部分だけを別のテキストファイルとして保存する。そこから変更したいプロパティ(オブジェクト名、信号名等)を空にしたものをテンプレートとする。作成したテンプレートの例を Fig. 2 に示す。

```
<widget typeId="org.csstudio.opibuilder.widgets.Label" version="1.0.0">
  <actions hook="false" hook_all="false" />
  <auto_size>false</auto_size>
  <background_color>
    <color red="255" green="255" blue="255" />
  </background_color>
  <border_color>
    <color red="0" green="128" blue="255" />
  </border_color>
  <border_style>1</border_style>
  <border_width>1</border_width>
  <enabled>true</enabled>
  <font>
    <opifont.name fontName="メイリオ" height="9" style="0" pixels="false">Default</opifont.name>
  </font>
  <foreground_color>
    <color red="0" green="0" blue="0" />
  </foreground_color>
  <height>15</height>
  <horizontal_alignment>0</horizontal_alignment>
  <name></name>
  <rules />
  <scale_options>
    <width_scalable>true</width_scalable>
    <height_scalable>true</height_scalable>
    <keep_wh_ratio>false</keep_wh_ratio>
  </scale_options>
  <scripts />
  <text></text>
  <tooltip></tooltip>
  <transparent>true</transparent>
  <vertical_alignment>1</vertical_alignment>
  <visible>true</visible>
  <widget_type>Label</widget_type>
  <width>85</width>
  <wrap_words>false</wrap_words>
  <wuid></wuid>
  <x></x>
  <y></y>
</widget>
```

Figure 2: Scripts of CSS object.

本プログラムは、信号リストからレコード名やコメント等の情報を、テンプレートの該当する部分に挿入することで、オブジェクトの生成を行うものである。また、PPS ではほとんどの信号が 2 重化され、A 系と B 系に分かれている。自動生成の際には 2 系統をペアにして、必要に応じて信号名のラベルを付けたものをグループ化して生成することとした。これにより、見た目の統一感を出し、配置作業を容易にすることができる。Figure 3 に自動生成したオブジェクトの例を示す。これは非常停止ボタンを等間隔で整列させたものを自動生成し、CSS により表示したものである。

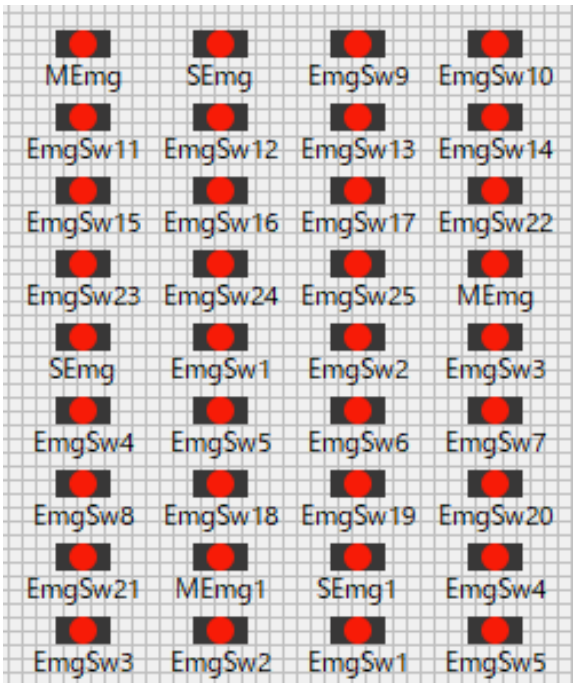


Figure 3: Examples of automatically generated objects.

長方形の中に赤丸のアイコンは二つのアイコンが隣り合って並んだ状態であり、中央から左側が A 系、右側が B 系となっている。非常停止ボタンの場合は、その下に

信号名ラベルを付けたものをグループ化している。このようにオブジェクトを生成した後に、J-PARC のマップを背景としたウィンドウに各ボタンのアイコンを手動で貼り付けていくことで GUI の作成を行った。自動生成した非常停止ボタン等をマップ表示に貼り付け、作成した GUI を Fig. 4 に示す。

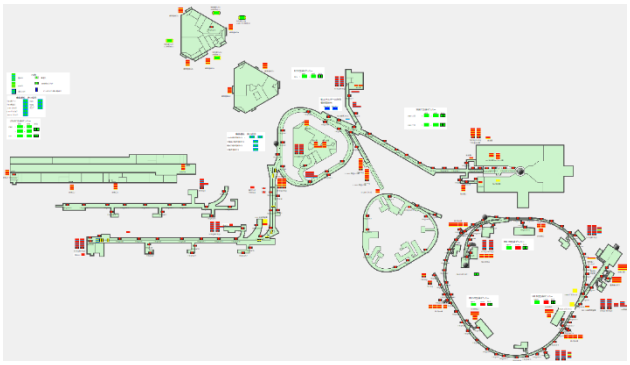

Figure 4: Created GUI.

## 5. オブジェクト座標情報移行の自動化

4 章まででオブジェクトの生成は可能となったが、ここで問題となるのがオブジェクトの配置である。

4 章ではオブジェクトを手動で配置し、GUI の作成を試みた。オブジェクトの生成がプログラムにより自動化されるだけでも、全て手動で行うよりはかなりの工数削減になる。しかし、オブジェクトの GUI 上の座標は実際に配置し、調整して決めていくことになる為、やはり、多くの工数を必要とする。

そこで、この問題を解消する方法がないか試行錯誤したところ、実は ASTMAC にも GUI をテキスト形式のファイルにエクスポートする機能があることが分かった。

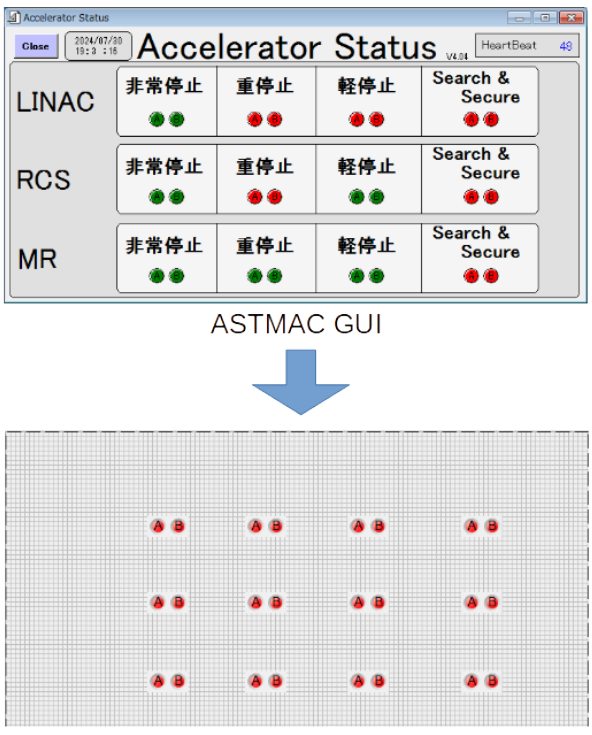


Figure 5: Example of automatically generated GUI.

この機能を用いてエクスポートされた ASTMAC のファイルからオブジェクトの座標を抽出することでオブジェクトの配置についても自動化ができないかと考えた。

ASTMAC の GUI をテキストファイルとしてエクスポートして生成されたテキストファイルはオブジェクトの始まり（Begin タグ）と終わり（End タグ）があり、その間にプロパティと値が対になって記述されるという構造になっている。グループ化されれば、グルーピング用のタグでネストされ、複数回グループ化すればその分階層が深くなる。グループ化されたオブジェクトの座標はグループ内の相対座標となっており、グループ自体の座標に加算する形で絶対位置を求める。

このようにして算出した座標を信号名と紐づけたデータとし CSS のプロパティとして組み込めば、座標情報も含めた GUI の自動生成が可能となる。Figure 5 に ASTMAC GUI から抽出した座標情報を用いて自動生成した CSS GUI の例を示す。

## 6. GUI の統合

5 章で座標情報も含めた GUI の自動生成を可能にするプログラムを作成したが、実はまだ課題を残している。

ASTMAC では 1 画面中のオブジェクト数を限定的にすることを推奨していた為、これまでの GUI は 1 画面中にオブジェクト数が多くなりすぎないよう、分割して作成していた。その為、無駄に画面が分割され、多くなってしまった。CSS では特にその制限がない為、全てを一つにまとめることもできるが、その場合、ASTMAC の GUI から取得した座標は絶対座標として扱うことができない。その為、ASTMAC の 1 画面単位をグループ化して画面上に配置することで GUI を作成することを検討中である。CSS では二つの GUI 間でオブジェクトのコピーが可能で、複数オブジェクトをグループ化しておけば、相対座標を維持したまま貼り付けや移動が可能なので、ASTMAC の 1 画面単位分の CSS オブジェクトを自動生成し、グループ化したものを、例えば全体のマップを背景とした GUI 上に貼り付けることで、ASTMAC の複数画面分のオブジェクトを一つの画面に集約することができる。

## 7. 動作試験

まず、動作試験環境として、実機とは切り離された試験用 PC を用意し、実機と同じレコード名のソフトレコードを持つ IOC と、新たに作成した CSS 版の GUI をインストールする。

全ての試験用ソフトレコードは 1 秒周期で変化するよう作成し、その環境上で CSS 版 GUI を実行して、異常が出ないかを確認する。その後、約半年程度 ASTMAC 版と並行して CSS 版 GUI を実機環境上で動作させ、安定的に動作することが確認された後に CSS 版 GUI に完全移行する予定である。

## 8. 今後の計画について

今回作成したプログラムは、完全な自動生成を行うことはできなかったが、GUI の大部分を生成することができ、大幅な省力化となった。GUI 完成後は完成した GUI か

ら絶対座標の抽出を行うことで、必要なデータはそろえることができる。今後、CSS の仕様変更や他のソフトへ移行することがあっても、そのファイルがテキストファイルであれば、テンプレートやプログラムの一部の変更のみで対応が可能である為、次回のバージョンアップやメンテナンスの際には強力なツールとなるはずである。

今回の動作試験が終了し、CSS に移行することで、PPS Data System は Windows7 から解放され、OS 及び PC を更新することができるようになる。これは一連の更新作業の大きな目的の一つが達成されたということである。しかし、最終的な目的は ASTMAC から EPICS 環境への移行であり、まだ、第 3 段階、第 4 段階の作業が残っている。

次の段階である第 3 段階の作業はロギングシステムの構築である。現在は VDS のログ機能を利用しているが、これを Archiver Appliance を中心としたロギングシステムに切り替え、より柔軟で高機能なロギングシステムの構築を目指す。

その後、OPC サーバ経由ではなく、NetDev を用いて PLC と EPICS IOC が直接通信する構成に変更し、PPS Data System の更新作業は完了となる。

## 参考文献

[1] N. Kikuzawa *et al.*, “J-PARC における人的保護システムの現状”, Proceedings of the 16th Annual Meeting of Particle Accelerator Society of Japan Jul. 31 – Aug. 3, 2019, Kyoto, Japan.

[2] K. Watanabe *et al.*, “J-PARC 加速器 PPS 情報収集・表示系の更新”, Proceedings of the 20th Annual Meeting of Particle Accelerator Society of Japan Aug. 29 – Sep. 1, 2023, Chiba, Japan.

[3] M. Clausen, L. Dalesio, “EPICS: Experimental physics and industrial control system”, ICFA Beam Dynamics Newsletter. 47. 2008, pages 56-66.

[4] K. Kasemir, “Control System Studio Applications”, ICALEPCS’07, Knoxville, Oct. 2007.
https://accelconf.web.cern.ch/ica07/PAPERS/ROPB02.PDF

[5] What is OPC? - OPC Foundation,
https://opcfoundation.org/about/what-is-opc/

[6] J. Odagiri, “ネットワーク・デバイスのための EPICS デバイス・サポート”, 2004,
https://www-linac.kek.jp/cont/epics/netdevbis/netdev-man.html