Archiver Appliance データの Grafana での可視化機能の改善とアラート機能の導入

IMPROVEMENT OF VISUALIZATION AND INTRODUCTION OF ALERTS FOR ARCHIVER APPLIANCE DATA ON GRAFANA

佐々木信哉#
Shinya Sasaki #
High Energy Accelerator Research Organization (KEK)

Abstract

Archiver Appliance Datasource is the Grafana plugin for visualizing data archived in Archiver Appliance (AA). The plugin has been updated to match the latest version of Grafana, and the new alerting and visualization features have been implemented. The alerting feature allows us to periodically check the data in AA and send an alert when the retrieved data meets the alert condition. In the visualization part, the plugin supports the array data. The array data can be converted into 3 types of table structures. These provide different visualization options depending on the purpose. In addition, stream feature and live feature have been implemented to update the visualized data in near real time or real time. This paper presents the details of the new alerting and visualization features.

1. はじめに

Archiver Appliance Datasource は Archiver Appliance (AA) のデータを Grafana 上で可視化するために開発された Grafana プラグインである[1]。本プラグインを導入することで、柔軟な可視化および解析を可能とする Grafana 上で、AA のアーカイブデータを表示するダッシュボードを Web から作成および閲覧することが可能となる。本プラグインは GitHub 上で公開されており[2]、PF や cERL・SuperKEKB・KEK 電子陽電子入射器・SLAC の LCLS において利用されている[1, 3, 4]。

プラグインの開発を始めた 2019 年当時は 6.0 だった Grafana のバージョンは 2023 年 8 月現在 10.0 までバージョンアップが行われ、機能の拡充や推奨するプラグイン開発ツールの変更がなされている。本プラグインにおいても、Grafana のバージョンアップに対応するためにアップデートを続けており、それにあわせて機能追加を行ってきた。本稿では 2021 年以降に追加された機能の内、バックエンドコンポーネントの追加によるアラート機能の導入と可視化機能の改善に関して報告する。

2. アラート機能の導入

2.1 Grafana のアラート機能の概要

Grafana はデータを可視化する機能の他に、警告を通知するためのアラート機能を備えている[5]。Grafana のアラート機能では、ユーザーによって指定されたデータソースのデータを定期的に確認し、異常と判断された場合にはメールやチャットツールなどで通知される。

Grafana のアラート機能はバックエンドプラグインのみが利用できる機能である。バックエンドプラグインは Grafana 7.0 からサポートされるようになった新しい動作形式のプラグインである。従来のプラグインはフロントエンドプラグインと呼ばれ、クライアント側のブラウザ上で動作していた。一方、バックエンドプラグインは Grafana サーバー上で動作するバックエンドコンポーネントを有し、HTTP 以外のプロトコルでデータを取得できるようになり、アラート機能の利用も可能となった[6]。

2.2 バックエンドコンポーネントの導入

本プラグインは Grafana 6.0 の頃に開発を始めたため、フロントエンドプラグインとして開発を進めていた。アラート機能を利用するためにはバックエンドプラグインである必要があったため、バックエンドコンポーネントの導入を行った。バックエンドプラグインであっても、クエリの編集画面などフロントエンドのコンポーネントも必要となるが、本プラグインではフロントエンド部分は大きく変更せずそのまま利用することができた。また、既存のダッシュボードに影響を及ぼすことのないように、従来のフロントエンドからのデータ取得処理とバックエンドからのデータ取得処理のどちらをデータ表示に利用するかをデータソースの設定画面から選択できるようにしている。

バックエンドコンポーネントは SLAC の Nolan Brown 氏が実装したものを統合することによって導入した[7, 8]。しかし、フロントエンドにおいて利用できた機能の一部は未実装であった。そのため、未実装であった既存機能のバックエンドへの適用を v1.4.0 から進め、v1.4.4 ではフロントエンドで利用できる機能の大部分をバックエンドにも適用した。

2.3 multi-dimensional alerting への対応

Grafana 8.0 ではアラート機能の改善が行われ、ダッシュボードと必ず結びついている必要があったアラートを、単独で作成および編集ができるようになった。それに加え、multi-dimensional alerting が利用可能となった[9]。multi-dimensional alerting は 1 つのアラートルールにおいて複数のアラートを作成するものである。従来は 1 つのアラートルールでは 1 つのアラートしか作成できなかったが、multi-dimensional alerting を利用する事で、例えば類似する複数の PV に対して同じアラートルールを適用

shinya.sasaki@kek.jp

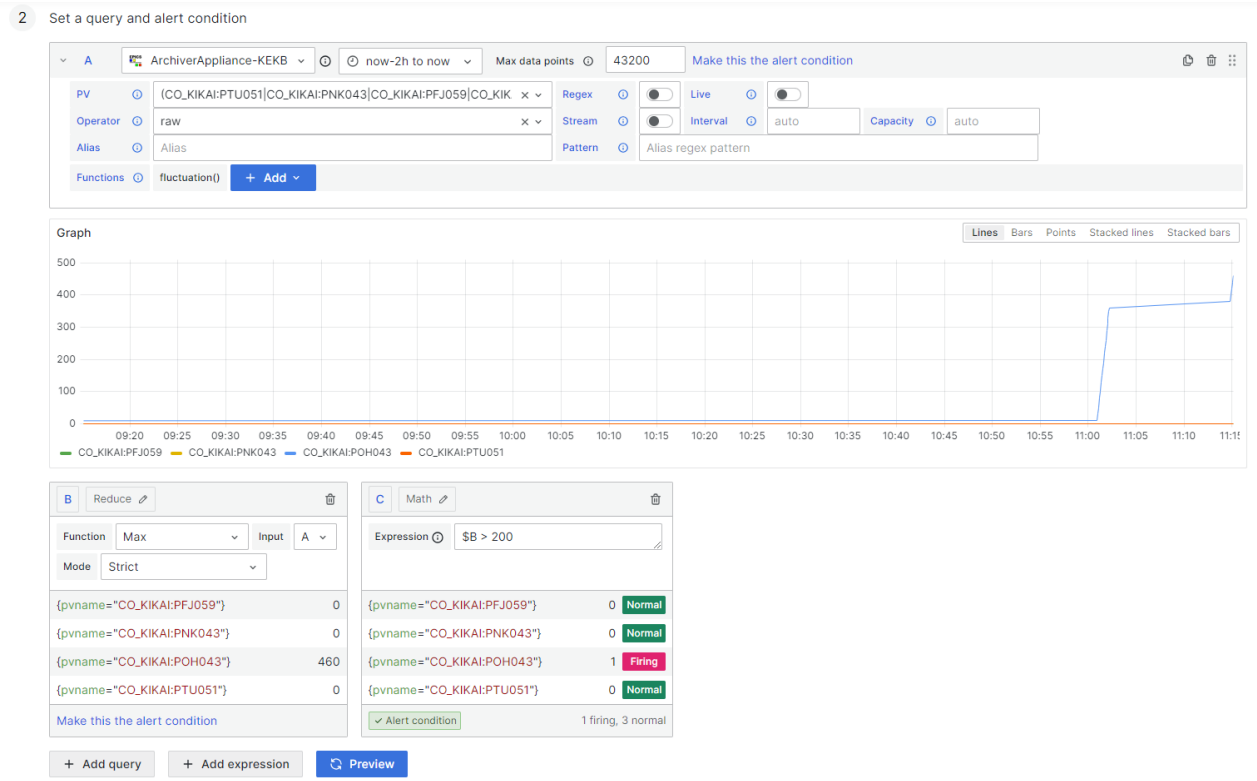

Figure 1: Example of the alert rule for the cooling water supply at SuperKEKB. This rule notifies the alert when the cooling water is supplied over 200 litters in 2 hours.

することが簡単に出来るようになった。1 つのクエリで取得した複数の時系列データを multi-dimensional alerting として扱うためには、それぞれのデータへ一意に識別できるラベルをつける必要がある。本プラグインでは各時系列データのラベルに PV 名を付与することで識別可能にしている。

2.4 アラートの設定例

アラートの発報条件の設定例として、SuperKEKB における冷却水の供給量に関するアラートの設定を Fig. 1 に示す。アラートの発報条件はアラートルールと呼ばれ、1 つ以上のクエリと expressions で構成される[10]。Figure 1 の例におけるアラートルールでは、以下の手順で発報するかどうかを判別する。

A) 現在から過去 2 時間の冷却水の供給量を取得。
B) A) で取得した時系列データを Reduce の expression によって単一の値に集約。ここでは Function に Max を指定することで各時系列データ中の最大値の値に集約される。
C) Math の expression において「$B > 200」と指定し、B)の値が 200 を超えるとアラートとみなす。

Figure 2 に Fig. 1 で設定したアラートのメール通知の例を示す。このメールの例では、アラートに設定された説明文やラベル、上記のアラートルールにおける各 expression での値が記載されている。

3. 可視化機能の改善

3.1 array データのサポート

AA は scalar データのみでなく Waveform のレコードに代表されるような array データも記録することができる。本プラグインでは v1.2.0 から array データをサポートしている。また、v1.4.0 からは arrayFormat という Function によって、array データのテーブル構造を目的に応じて変

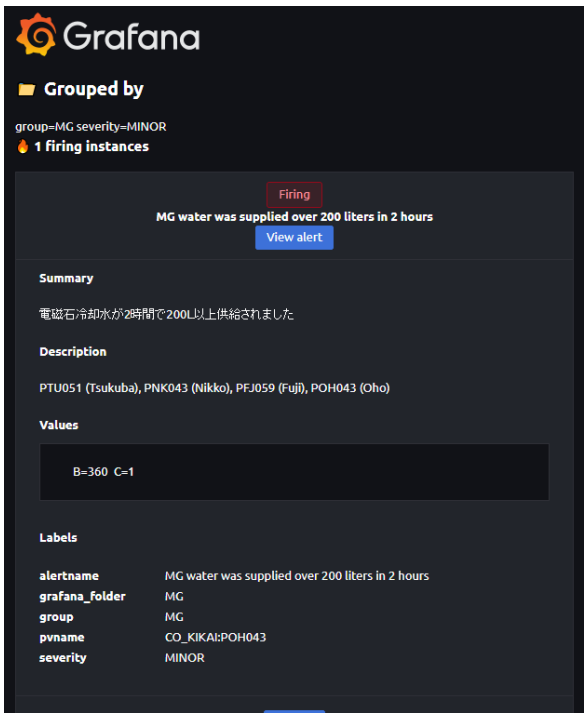

Figure 2: Example of the notification mail from Grafana alert manager.

更できるようにしている。利用可能なテーブル構造は timeseries と index、dt-space の 3 つである。Function によってテーブル構造を指定しない場合は、timeseries のテーブル構造として扱われる。それぞれのテーブル構造に関して要素数が 3 で EX:PV という名前の array データを例にして説明する。

3.1.1 timeseries のテーブル構造

timeseries のテーブル構造の例を Table 1 に示す。このテーブル構造では 1 列目にサンプリング時間である time を有し、2 列目以降は array データの各要素の値が格納される。2 列目以降の列名は EX:PV[0]のように、対象の PV 名の末尾に array の要素番号を付与した名前が付けられる。1 列目に time を配置しているため、Time series パネルで表示することが可能で、array データの各要素がそれぞれ独立した時系列データとして扱われる。

Figure 3 に timeseries のテーブル構造のデータを Time series パネルで表示した例を示す。Array データの各要素の値が、それぞれ時系列データとして表示されている。

Table 1: Table Structure of Timeseries Array Format

time	EX:PV[0]	EX:PV[1]	EX:PV[2]
2023-01-01 00:00:00	1	2	3
2023-01-01 00:01:00	4	5	6
2023-01-01 00:02:00	7	8	9

Figure 3: Visualized array data with timeseries format on Time series panel. Only the last 9 elements in the array are shown in this plot.

3.1.2 index のテーブル構造

index のテーブル構造の例を Table 2 に示す。このテーブル構造では 1 列目は要素番号を、2 列目以降は各サンプリング時間の値を格納する。他のテーブル構造と異なり time の列を含まないため、Time series パネルでは可視化することが出来ない。したがって、Table パネルを利用して表データとして表示するか、Grafana 10.0 においてベータ版として提供されている XY Chart もしくは Trend のパネルによって表示する。このテーブル構造は array データを波形として確認したい場合に利用することを想定している。

Figure 4 に index のテーブル構造のデータを Trend パネルで表示した例を示す。横軸が要素番号となり、サンプリング時間ごとの波形データを確認することができる。

Table 2: Table Structure of Index Array Format

index	2023-01-01T 00:00:00.000Z	2023-01-01T 00:01:00.000Z	2023-01-01T 00:02:00.000Z
0	1	4	7
1	2	5	8
2	3	6	9

Figure 4: Visualized array data with index format on Trend panel. The index format allows to visualize the waveform of each sampled array data.

3.1.3 dt-space のテーブル構造

dt-space のテーブル構造の例を Table 3 に示す。このテーブル構造では array データを 1 つの時系列データに変換する。そのため、array データ内の 2 要素目以降のデータは、本来のサンプリング時間に要素番号分の時間をミリ秒単位で追加した時間を新しいサンプリング時間とする。このテーブル構造は Time series パネルにおいて array データに変化があった日時を分かりやすく可視化するために開発された[11]。サンプリング間隔が短く、要素数が大きい場合はデータが重複する場合もある。

Table 3: Table Structure of dt-space Array Format

time	EX:PV
2023-01-01 00:00:00.000	1
2023-01-01 00:00:00.001	2
2023-01-01 00:00:00.002	3
2023-01-01 00:01:00.000	4
2023-01-01 00:01:00.001	5
2023-01-01 00:01:00.002	6
2023-01-01 00:02:00.000	7
2023-01-01 00:02:00.001	8
2023-01-01 00:02:00.002	9

Figure 5 に dt-space のテーブル構造のデータを Time series パネルで表示した例を示す。複数の array データが単一の時系列データとして扱われており、データが更新された時点が比較的分かりやすく表示することができる。dt-space において波形を確認したい場合には、Fig. 6 のように表示時間の範囲を狭めて表示する。

3.2 Stream 機能

Stream 機能は v1.3.0 において追加された、一定時間ごとに表示データを更新する機能である。Grafana 自体にも自動更新機能が備えられており、定期的に表示データを更新することが可能であるが、更新のたびに表示範囲内の全てのデータを取得し直してしまう。Stream 機能を利用した場合は、前回更新した日時以降のデータのみを取得するため、より効率的に表示の更新を行うことが出来る。

Stream 機能を有効にした際の動作に関して説明する。Stream 機能を有効にすると、まず表示データを格納するためのリングバッファが作成される。つぎに、指定された時間幅のデータを通常通り取得し、リングバッファに格納する。この時点でパネルではリングバッファのデータが表示される。その後は、前回取得した日時以降のデータを定期的に取得してリングバッファに格納していく。リングバッファの更新に合わせてパネルの表示も更新される。

Figure 5: Visualized array data with dt-space format on Time series panel. This plot shows when the array data were updated.

Figure 6: Visualized array data with dt-space format in short time range. This plot roughly shows waveform of the array data. However, it cannot show the index number of each element.

Stream 機能を有効にする際は、データ更新の頻度を決定する interval と、データを格納するリングバッファのバッファサイズである capacity を指定する。明示的にこれらを指定しない場合、interval はパネルの横幅のピクセル数と表示している時間範囲によって自動的に決定され、capacity は始めに取得したデータ数から自動的に決定される。

Stream 機能はバックエンドコンポーネントを有効化していてもそれを利用せず、従来のフロントエンドのコンポーネントからのデータ取得を利用する。また、Functions によって実行される後処理は、更新のたびにリングバッファの全データに対して実行される。

3.3 Live 機能

3.3.1 Live 機能の概要

Live 機能は v1.6.0 において追加された、EPICS PV の値をモニターして、リアルタイムに表示を更新する機能である。Live 機能では Grafana 8.0 から導入された Grafana Live [12]を利用する。Grafana Live はリアルタイムメッセージエンジンであり、これを利用する事で再描画するためのデータをバックエンドからフロントエンドへリアルタイムに送信できるようになる。バックエンドとフロントエンドの間のデータのやり取りには WebSocket が利用される。

3.3.2 PVWS の採用

対象の PV のデータ更新をリアルタイムにフロントエンド側へ送信するためには、バックエンド側でその PV をモニターする必要がある。そのため、Grafana サーバーとは別に PV Web Socket (PVWS) [13]サーバーを立ち上げることを前提としてバックエンドコンポーネントを実装した。

PVWS は EPICS PV に WebSocket によってアクセスすることを可能とする WebSocket サーバーである。PVWS は CS-Studio で作成された制御画面を Web 上から閲覧するための Display Builder Web Runtime [14]での利用を主目的として開発された。ALS においては EPICS PV のディレクトリサービスおよび複数の上位サービスを統合する Web インターフェイスとして開発された PV Info [15]においても利用されている。PVWS は CS-Studio Phoebus で利用される PV アクセスレイヤーを基に開発されているため、Channel Access (CA)や PV Access の他、ローカル PV やシミュレーション PV にも対応している。

EPICS PV に対して WebSocket や Server-Sent Events によって接続するための実装は他にもあるが、CA と PV Access の両方に対応しており、PV Info のような CS-Studio とは別のプロジェクトでの採用例もあることから PVWS を選択した。また、WebSocket や Server-Sent Events といった Web 関係のプロトコルを使用せず、バックエンドコンポーネントから直接 CA や PV Access によって PV と接続する方法も考えられた。しかし、Go 言語で開発しているバックエンドコンポーネントにおいては、Web 関係のプロトコルを利用した方が開発を容易に進められることから WebSocket による接続を選択した。

3.3.3 Live 機能の動作と実装

Figure 7 に Live 機能によるデータ更新を行うためのシステム構成図を示す。Live 機能を有効化している場合で

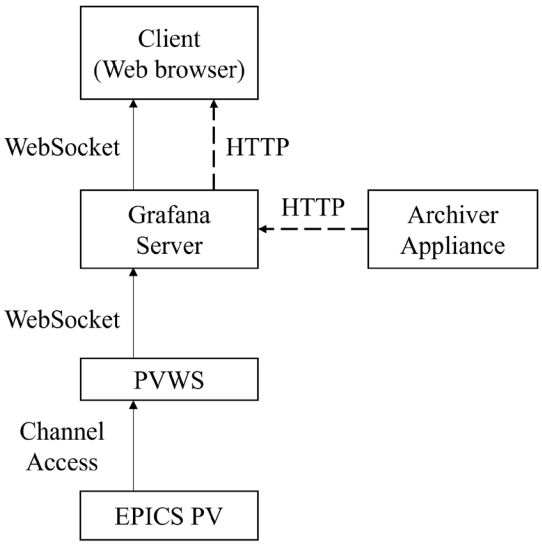

Figure 7: Block diagram of the live feature. The client first retrieves the data in the specified time range from Archiver Appliance via HTTP. Then Grafana server pushes the updates of PV values to the client from PV Web Socket server via WebSocket.

も、指定した時間範囲のデータを AA から HTTP で始めに取得する。そして、AA から取得したデータが描画された後に Live 機能によるデータの更新が行われる。

Live 機能によるデータ更新は次のように行われる。まず対象の PV のデータの購読要求がクライアントからバックエンドコンポーネントへ送られる。つぎに、バックエンドコンポーネントが PVWS に対して WebSocket で接続し、その PV のモニターを開始する。その後は、バックエンドコンポーネントが PVWS から値の更新を受け取る度に、その値を Grafana Live を通してクライアントに伝達する。

クライアントからの購読要求は Grafana Live のチャンネルによって識別される[12]。チャンネルは Scope と Namespace、Path の 3 つから構成される。データソースプラグインにおけるデータストリーミングでの利用の場合、Scope は ds という文字列、Namespace は購読要求先のプラグインの Unique ID (UID)と決まっている。したがって、チャンネルの指定においてプラグインで自由に指定可能な部分は Path のみとなる。本プラグインでは Path 部分でモニターする PV 名を指定する。しかし、PV 名でよく利用されるコロン(:)は Path の中で使用することが出来ない文字だった。そのため、Path の中ではコロンをイコール(=)に変換して指定するようにして対応している。例えば EX:PV という名前の PV は EX=PV という Path となる。PVWS に PV のモニターを要求する際に再びイコールをコロンに変換する。イコールは PV 名として使用可能な文字であるため、イコールを名前に含む PV はモニターできなくなってしまう。しかし、コロンに比べるとイコールの利用頻度は低いと考え、上記のような実装とした。

Live 機能は Stream 機能と異なり、リアルタイムにデータを更新することが可能である。一方で、Stream 機能と異なり Functions 機能によるデータの後処理を利用することが出来ないことや、フロントエンド側で利用されるリングバッファの大きさを制御できないという欠点もある。

4. まとめ

Grafana のデータソースプラグインである Archiver Appliance Datasource に機能を追加し、アラート機能の追加と可視化機能の改善を行った。アラート機能を利用するために必要だったバックエンドコンポーネントはフロントエンドプラグインで利用できた大部分の機能を備え、データ表示にもアラート機能にも使用することが出来るようになった。また、可視化機能の改善として、プラグイン内で array データを利用できるようにしたほか、Stream 機能や Live 機能の実装によって表示画面の効率的な更新も可能となった。

参考文献

[1] S. Sasaki and T. Obina, “Development of Grafana plugin to visualize archive data on Archiver Appliance”, Proc. 17th Annual Meeting of Particle Accelerator Society of Japan (PASJ2020), Online, Japan, Sep. 2020, pp. 504-508.
[2] GitHub repository of Archiver Appliance Datasource, https://github.com/sasaki77/archiverappliance-datasource
[3] I. Satake *et al.*, “Introduction of network monitoring system using Grafana in the KEK electron/positron injector linac”, Proc. of the 19th Annual Meeting of Particle Accelerator Society of Japan (PASJ2022), Kitakyusyu (Online meeting), Japan, Oct. 2022, pp. 305-308.
[4] M. Shankar, “EPICS Archiver Appliance Update”, presented at the EPICS Collaboration Fall Meeting 2020, Online, Oct. 2020.
[5] Documentation of Grafana alerting, https://grafana.com/docs/grafana/v10.0/alerting/
[6] Documentation of Grafana backend plugin, https://grafana.com/docs/grafana/v10.0/developers/plugins/introduction-to-plugin-development/backend/
[7] N. Brown, “A System for User-created Alerts with Grafana and the EPICS Archiver Appliance”, presented at the EPICS Collaboration Spring Meeting 2021, Online, Jul. 2021.
[8] Merged pull request for the backend component, https://github.com/sasaki77/archiverappliance-datasource/pull/24
[9] Updates of Grafana alerting in Grafana 9, https://grafana.com/blog/2022/06/14/grafana-alerting-explore-our-latest-updates-in-grafana-9/
[10] Queries and conditions of Grafana alert rules, https://grafana.com/docs/grafana/v10.0/alerting/fundamentals/alert-rules/queries-conditions/
[11] History of the dt-space array format development, https://github.com/sasaki77/archiverappliance-datasource/issues/83
[12] Grafana Live, https://grafana.com/docs/grafana/v10.0/setup-grafana/set-up-grafana-live/
[13] PV Web Socket, https://github.com/ornl-epics/pvws
[14] Display Builder Web Runtime, https://github.com/ornl-epics/dbwr
[15] PV Info, https://github.com/channelFinder/pvinfo