SuperKEKB での利用に向けた Kafka を基盤とする CS-Studio alarm system の性能評価

EVALUATION OF CS-STUDIO ALARM SYSTEM BASED ON KAFKA FOR SuperKEKB

佐々木信哉#, A), 中村卓也 B), 廣瀬雅哉 C)
Shinya Sasaki #, A), Takuya Nakamura B), Masaya Hirose C)
A) High Energy Accelerator Research Organization (KEK)
B) Mitsubishi Electric System & Service Co., Ltd.
C) Kanto Information Service Co., Ltd.

Abstract

We have stably operated CS-Studio alarm system that is based on a relational database (RDB) and the Apache ActiveMQ message service at SuperKEKB. Recently, CS-Studio community have developed updated product of CS-Studio named Phoebus. The alarm system has also been updated and the combination of relational database and ActiveMQ has been replaced by Apache Kafka. We conducted several tests to evaluate the Kafka-based alarm system. The tests revealed the performance of the Kafka-based system is superior to that of the original system. The Kafka-based system is simpler than the previous implementation for the combination of RDB and ActiveMQ system. Therefore, the Kafka-based system can be easily integrated with other applications. We plan to run the Kafka-based system on a long-term basis to verify a stability of the system.

1. はじめに

SuperKEKB ではリレーショナルデータベース(RDB)と Apache ActiveMQ を中心とする CS-Studio アラームシステム[1]をアラームマネジメントシステムとして利用している。CS-Studio[2]は EPICS を利用した制御システムを監視・操作するためのツールセットであり、アラームシステムも CS-Studio の提供するアプリケーションのひとつである。SuperKEKB では運転が開始された時から CS-Studio アラームシステムを利用しており、これまで安定に運用することが出来ている[3, 4]。

一方、CS-Studio の開発コミュニティでは次世代の CS-Studio として Phoebus[5]の開発が進められている。Phoebus ではアラームシステムの構成が見直され、Apache Kafka[6]を中心とするアラームシステムが提供されている[7]。CS-Studio アラームシステムにおいて RDB と ActiveMQ が担っていた役割を Kafka 一つが担うようになったことで、全体の構成が単純になっている。

Phoebus が提供する Kafka を基盤としたアラームシステムを SuperKEKB のアラームマネジメントシステムとして利用することが可能かどうか検討するため、我々はアラームシステムの動作試験および性能評価を行った。本稿ではアラームシステムの試験結果を報告する。また、SuperKEKB のアラームマネジメントシステムを、Phoebus のアラームシステムに移行する利点・欠点について報告する。

なお、Phoebus が提供するアラームシステムは Kafka ベースの CS-Studio アラームシステムと呼称される[5]。しかし、本稿では RDB と ActiveMQ を中心とした従来のシステムを CS-Studio アラームシステム、Phoebus が提供する Kafka を中心としたシステムを Phoebus アラームシステムと呼称することとする。

2. SuperKEKB でのシステム運用状況

SuperKEKB では運転開始以来 CS-Studio アラームシステムを利用し、約 15000 個のアラームを安定して監視することが出来ている[3]。

SuperKEKB ではアラームシステムのクライアントツールとして、独自に開発したアプリケーションを主に利用している[3]。例えば、現在のアラームの状況や過去のアラームの履歴を表示する GUI アプリケーションや、CSV 形式のファイルからアラームの登録内容を変更するアプリケーションを利用している。

システムが安定して稼働出来ている一方で、その運用方法には課題もある。例えば、アラームが発報した際に、オペレータがそれに対してどのように対応すべきか判断に困る場合がある。CS-Studio アラームシステムでも Guidance などの機能を利用して、アラーム毎にその対応方法を記述することは可能である。しかし、利用している独自のアプリケーションではそれらの機能を利用するための実装は行っていないため、プログラムの変更が必要になる。

また、メンテナンス中などで多数のアラームが発報している際に、対応すべきアラームを見過ごしてしまう場合もある。この課題を解決するには運用方法の見直しなどが必要となる。

独自に開発したアプリケーションの変更や運用方法の見直しを検討するのにあわせて、我々は Phoebus アラームシステムの試験・評価を行うことにした。

shinya.sasaki@kek.jp

3. Phoebus アラームシステム

3.1 Apache Kafka

ここでは、Phoebus アラームシステムにおいて中核を担う Apache Kafka について説明する。Apache Kafka は JMS（Java Message Service）のようにメッセージングサービスとして利用可能な分散ストリーミングプラットフォームである[6]。Kafka は単体のサーバーとして利用することもできる他、複数台でクラスタを構成して分散処理させることもできる。また、通常のメッセージングサービスと異なり、システムでやり取りされるメッセージをストレージに保存しておくことが可能である。

Kafka でやり取りされるメッセージはキーとバリューで構成される。キーの重複に関わらず、全てのメッセージがストレージに保存される。そのため、保存されたデータを読み出した場合に、同じキーを持つメッセージが複数回読み出される場合もある。メッセージは Kafka の Partition に書き込まれた順序に従って保存される。Kafka の読み出しクライアントである Consumer からデータを読み出す際も Partition に書き込まれた順序になる。そのため、同じキーのメッセージが読み出された場合でも、最後に読み出されたメッセージがそのキーの最新のメッセージとなる。

やり取りされるメッセージを全て残し続けると保存データが肥大化していく。そのため、Kafka では保存されたメッセージを定期的、もしくはデータが一定量に達した際に削除することが出来る。デフォルトでは過去のメッセージから順に削除される。また、キー毎に最新のメッセージを残して、それ以外のメッセージは削除する Log Compaction と呼ばれる方式も利用できる。Phoebus アラームシステムではアラームの設定情報や状態に関するメッセージを扱う Topic を Log Compaction に設定することで、最新のアラーム状態が保持され続けるようにしている。

3.2 Phoebus アラームシステムの構成

Phoebus アラームシステムの構成を Fig. 1 に示す。登録された PV のアラーム状態を監視する Alarm server、アラーム状態の表示やコマンドを送信する Alarm client、メッセージのやり取りや保存を行う Kafka で Phoebus アラームシステムは構成される。CS-Studio アラームシステムにおいて JMS と RDB が担っていた「メッセージの配信」と「アラームの設定・状態の保存」の役割を、Phoebus アラームシステムでは Kafka が担っている。

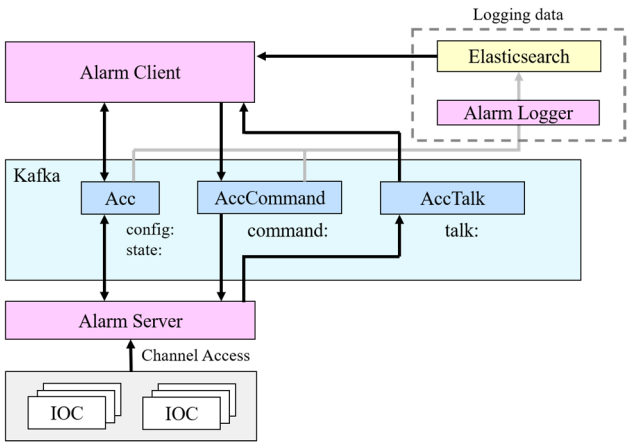

Figure 1: System diagram of the Kafka-based alarm system.

Phoebus アラームシステムでやり取りされるメッセージの種類、やり取りに利用する Topic、データ削除の方針を Table 1 に示す[8]。Topic にはユーザーが任意の名前を付与するため、ここでは Acc という名前を指定した場合の Topic 名としている。

Table 1: Message Types and Configuration of Kafka for Kafka-based Alarm System

種類	Topic	削除方針
config	Acc	compaction
state	Acc	compaction
command	AccCommand	delete
talk	AccTalk	delete

Phoebus アラームシステムでは「config」・「state」・「command」・「talk」の 4 種類のメッセージがやりとりされる。アラームの設定情報を扱う「config」や状態を扱う「state」のメッセージは Acc Topic で処理される。これらは最新のメッセージを保持し続ける必要があるため、Log Compaction が設定される。一方、Alarm client から Acknowledge を送信するのに利用する「command」のメッセージは AccCommand Topic に、アラームの音声通知のために利用する「talk」のメッセージは AccTalk Topic で処理する。これらは過去のデータを保存する必要がないため、過去のデータから削除されるデフォルトの削除方針が設定される。

アラームシステムでは監視対象の PV とそれらが構成するアラームのグループがツリー構造として管理される。Phoebus アラームシステムでやり取りされるメッセージは、メッセージの種別と、どの PV・グループに対するメッセージなのかをキーで指定する。例えば、Vac グループに属する PIn という PV に対する「config」メッセージは以下の通りとなる。

config:/Acc/Vac/PIn = {“description”: “Pressure”}

Phoebus アラームシステムでは CS-Studio アラームシステムと同じ XML 形式のファイルで設定情報のインポート・エクスポートが可能である。そのため、CS-Studio アラームシステムから Phoebus アラームシステムへのデータ移行も容易に行うことが出来る。

3.3 連携サービス

Kafka を経由して送信されるメッセージを利用する Alarm logger や Alarm config logger といったサービスが Phoebus では提供されている。これらのサービスは Phoebus アラームシステムが動作する上で必須のものではないが、アラームシステムの運用に便利な機能をもた

らす。

Alarm logger は送信されるメッセージを Elasticsearch に転送し、アラームの過去の履歴を保存するためのツールである[9]。CS-Studio アラームシステムで提供されていた JMS2RDB と同等のツールである。過去の履歴を閲覧するための GUI も Phoebus で提供されている。また、Kibana などを利用すれば、Web から過去の履歴を参照することも可能である。

Alarm Config logger はアラームシステムの設定情報のスナップショットを記録するためのツールである[10]。アラームのツリー構造はそのままディレクトリ構造に展開され、各要素の設定情報は JSON 形式のファイルに保存される。このファイル群は Git で管理され、設定情報の変更履歴を遡ることができる。

4. アラームシステムの性能試験

4.1 試験環境

SuperKEKB では約 15000 点の PV を監視対象としている。また、今後さらに登録数が増えることも考えられる。そのため、数万点の PV を監視対象とした場合でも利用可能かどうかを確かめるために、性能評価のための試験を行った。試験内容は過去に CS-Studio アラームシステムで行った試験を参考にした[11]。

過去の試験と同様に、監視対象となる 50000 個の PV を用意した。50000 個の PV はアラームシステムが動作する計算機の software IOC 上で動作する。50000 個の PV の内 200 個の PV がアラーム状態となり、別の 200 個の PV のアラームが解除される動作を 40 秒毎に繰り返す。また、PV は 500 個のグループに分けてアラームシステムに登録した。グループは 3 階層に分けられ、1 階層目が 5 グループ、2・3 階層目がそれぞれ 10 グループで構成される。PV は 3 階層目のグループに属し、1 グループに 100 個の PV が登録される。

CS-Studio アラームシステムと Phoebus アラームシステムを比較するため、両者に対して同じ計算機上で同様の試験を行った。試験は KVM を利用した仮想環境において実施した。試験に使用した計算機の詳細を Table 2 に、仮想環境の詳細を Table 3 に示す。また、Phoebus アラームシステムと CS-Studio アラームシステムにおいて使用した各ソフトウェアのバージョンを Table 4、Table 5 にそれぞれ示す。

なお、CS-Studio アラームシステムを構成する PostgreSQL では、書き込み性能を向上させるために synchronous_commit を無効にして利用した。synchronous_commit を無効にした場合、OS やデータベースがクラッシュした際にトランザクションの一部が失われる可能性がある。そのため、実際の運用で利用する際には注意する必要がある。

Table 2: Specification of Test Server

CPU	AMD EPYC 7302P 3GHz 16 コア
OS	CentOS 7.8
Memory	64 GB
Disk Type	HDD

Table 3: Specification of Virtual Machine

CPU コア数	4 コア
OS	CentOS 7.9
Memory	16 GB

Table 4: Software Versions for Phoebus Alarm System Based on Kafka

Phoebus （コミット ID）	4.6.6 (2f5de05be5f2827f095097a7963114b17a4efb31)
Java	14.0.1
Kafka	2.7.0
Elasticsearch	6.8.14

Table 5: Software Versions for CS-Studio Alarm System Based on RDB and ActiveMQ

CS-Studio （コミット ID）	4.6 (f0ca39dca14f11911d665fd7529c03da8aeede98)
Java	11.0.2
PostgreSQL	13.3
ActiveMQ	5.5.0

4.2 Alarm Config Tool

XML ファイルを利用して設定情報をインポート・エクスポートする Alarm Config Tool がアラームシステムでは用意されている。Alarm Config Tool による更新処理では、一旦全ての登録情報が削除された後に設定情報が更新される。CS-Studio アラームシステムでは Alarm Config Tool による更新処理は非常に時間がかかっていたため、SuperKEKB では設定情報のバックアップ取得や障害発生時の復旧のみに利用している。

Alarm Config Tool を利用した設定情報の更新処理に要する時間を測定した。インポートする XML ファイルを調整し、設定の追加のみ行う場合、設定の削除のみ行う場合、設定の削除と追加を続けて行う場合の 3 種類のデータの更新方法について実行時間を測定した。測定結果を Table 6 に示す。

Table 6: Execution Time of Alarm Config Tool

	Phoebus	CS-Studio
実行時間：追加	6.4 秒	52 秒
実行時間：削除	6.0 秒	35 秒
実行時間：削除と追加	6.8 秒	1 分 31 秒

Table 6 に示す通り、CS-Studio アラームシステムと比較して Phoebus アラームシステムは非常に短い時間で処

理が完了していることが分かる。

4.3 Alarm server・Alarm client 起動時間

Alarm server および Alarm client を起動するのに要する時間を測定した。Alarm server の起動時間は、起動してから監視対象の全ての PV に接続が完了するまでの時間を測定した。Alarm client の起動時間は Phoebus もしくは CS-Studio の GUI を起動してから Alarm Tree Display が全ての PV を読み込むまでの時間を測定した。それぞれの時間は、プログラムが出力するログファイルを参照して算出した。

測定結果を Table 7 に示す。Alarm server の起動時間は Phoebus アラームシステムの方が早かった。一方、Alarm client の起動時間は CS-Studio アラームシステムの方が早かった。

Table 7: Start-up Time of Alarm Server and Client

	Phoebus	CS-Studio
Alarm server 起動時間	8 秒	31 秒
Alarm client 起動時間	56 秒	20 秒

4.4 アラーム情報更新に要する時間

アラームの更新情報が Alarm client に伝わるまでの時間を測定した。Phoebus アラームシステムでは Kafka の Consumer スクリプトを利用して、全ての更新情報が受信されるまでの時間を測定した。CS-Studio アラームシステムでは、PostgreSQL のログを参照し、全ての SQL の処理が完了するまでの時間を測定した。

監視対象の PV は 40 秒ごとに 400 件のアラーム状態の変化が発生する。また、PV を動作させている software IOC を停止することで 50000 件のアラーム状態の変化を発生させることができる。試験ではこれらの更新情報が Alarm client に伝わるまでの時間を測定する。

測定結果を Table 8 に示す。どちらのアラームシステムでも許容範囲の時間で処理できている。また、Phoebus アラームシステムの方が比較的早いことが分かる。

Table 8: Time of Updating Alarm State

	Phoebus	CS-Studio
400 件の更新時間	0.2 秒	1 秒
50000 件の更新時間	1.5 秒	4.7 秒

4.5 アラームログの書き込み時間とデータサイズ

監視対象の PV を動作させている software IOC を停止し、50000 件のアラーム状態の変化が発生した時のアラームログの書き込み時間とデータ量の変化を測定した。Phoebus アラームシステムでは Alarm logger によって Elasticsearch に全てのデータが書き込まれるまでの時間を測定した。Elasticsearch の Ingest pipelines[12]を利用して、Elasticsearch にデータが書き込まれた時間をフィールドに付与することで、書き込み時間を測定した。CS-Studio アラームシステムでは JMS2RDB によって RDB に全てのデータが書き込まれるまでの時間を測定した。RDB にデータが書き込まれた時間を示す Time カラムのデータを参照して書き込み時間の測定を行った。また、アラームの音声通知のために使用される TALK の Topic は保存されないように設定した。

測定結果を Table 9 に示す。Phoebus アラームシステムはログの処理時間が短く、アラームの更新時間とほぼ同程度の時間でログの処理が完了していた。保存に要するデータサイズも Phoebus アラームシステムの方が小さいことが分かる。

Table 9: Time and Data Size of Logging Alarm Message

	Phoebus	CS-Studio
ログ書き込み時間	1.8 秒	31 秒
ログのデータ量	5.7 MB	52 MB

5. Phoebus アラームシステムを導入する利点・欠点

これまでに示した Phoebus アラームシステムの特徴や、その性能試験の結果を踏まえ、Phoebus アラームシステムを SuperKEKB のアラームマネジメントシステムとして利用する利点および欠点を検討する。

性能試験において、Alarm client の起動時間以外では Phoebus アラームシステムが良い性能を示した。また、Phoebus アラームシステムでは全てのデータが Kafka のメッセージとしてやり取りされる。そのため、クライアントアプリケーションを開発する際は Kafka とのメッセージ通信処理のみを実装するだけで済む。RDB と ActiveMQ の両方に接続し情報を取得する必要があった CS-Studio アラームシステムと比較するとクライアントアプリケーションの開発は容易になる。Logstash などのログ収集アプリケーションを利用して他のアプリケーションと連携した動作をさせることも可能である。機能向上やバグ修正のサポートにおいても CS-Studio の後継として開発されている Phoebus の方が優れていると考えられる。

一方で、CS-Studio アラームシステムと比較して Phoebus アラームシステムは開発されてから日が浅く、利用実績も多くない。そのため、バグもまだ残っており、動作の安定性の面では課題がある。実際にいくつかのバグが今回の性能試験の際に見つかり、修正されている[13, 14]。また、CS-Studio アラームシステムでは SQL を利用して条件付きの問い合わせを行い、データの絞り込みなどが可能だった。しかし、Phoebus アラームシステムにおいてデータの絞り込みなどを行いたい場合には、Kafka に保存されている全てのデータをクライアントが受信し、クライアント側でデータの絞り込みなどの処理を行わなければならない。SuperKEKB のように独自のクライアントアプリケーションを利用している場合には、それらを Phoebus アラームシステム用に変更する作業も必要となる。SuperKEKB では RDB は他のアプリケーションでも

利用されており、利用実績もある。しかし、Kafka の利用実績はないため、これまで運用実績のない Kafka を運用する必要があるという点も懸念される。

6. まとめ

SuperKEKB のアラームマネジメントシステムとして Phoebus アラームシステムが十分な性能を備えているか確認するため、アラームシステムの性能試験を行った。試験の結果、CS-Studio アラームシステムと比較して Phoebus アラームシステムの性能の方が多くの部分で優れていた。Phoebus アラームシステムは Kafka を中心とした単純な構成のため、他のアプリケーションとの連携も容易に行うことができる利点もある。一方、CS-Studio アラームシステムと比較すると利用実績が少ないため、安定性の面などで課題もある。今後は SuperKEKB 加速器の運転にあわせて Phoebus アラームシステムを試験的に運用し、長期的な試験を行うことを検討する。

参考文献

[1] K.-U. Kasemir, “The Best Ever Alarm System Toolkit”, in Proc. ICALEPCS'09, Kobe, Japan, Oct. 2009, pp. 46-48; https://accelconf.web.cern.ch/icalepcs2009/papers/tua001.pdf
[2] CS-Studio; https://controlsystemstudio.org
[3] T. Nakamura *et al*., “Operation Status of CSS Alarm System for SuperKEKB”, Proceedings of the 13th Annual Meeting of Particle Accelerator Society of Japan, Chiba, Japan, Aug. 8-10, 2016, pp. 1159-1162; https://www.pasj.jp/web_publish/pasj2016/proceedings/PDF/TUP0/TUP095.pdf
[4] M. Hirose *et al*., “Evaluation of the CSS V4 alarm system for SuperKEKB”, Proceedings of the 14th Annual Meeting of Particle Accelerator Society of Japan, Sapporo, Japan, Aug. 1-3, 2017, pp. 602-606; https://www.pasj.jp/web_publish/pasj2017/proceedings/PDF/TUP0/TUP093.pdf
[5] K. Shroff *et al*., “New Java Frameworks for Building Next Generation EPICS Applications”, in Proc. ICALEPCS'19, New York, NY, USA, Oct. 2019, pp. 1497-1500; doi:10.18429/JACoW-ICALEPCS2019-WESH1002
[6] Apache Kafka; https://kafka.apache.org
[7] K.-U. Kasemir, “CS-Studio Alarm System Based on Kafka”, in Proc. ICALEPCS'19, New York, NY, USA, Oct. 2019, pp. 1504-1508; doi:10.18429/JACoW-ICALEPCS2019-WESH2001
[8] Phoebus Alarm System; https://github.com/ControlSystemStudio/phoebus/tree/master/app/alarm
[9] Phoebus Documentation, Alarm Logging Service; https://control-system-studio.readthedocs.io/en/latest/services/alarm-logger/doc/index.html
[10] Phoebus Documentation, Alarm Configuration Logging; https://control-system-studio.readthedocs.io/en/latest/services/alarm-config-logger/doc/index.html
[11] T. Nakamura *et al*., “Evaluation of the CSS based Alarm System for SuperKEKB”, Proceedings of the 12th Annual Meeting of Particle Accelerator Society of Japan, Tsuruga, Japan, Aug. 5-7, 2015, pp. 795-799; https://www.pasj.jp/web_publish/pasj2015/proceedings/PDF/WEP1/WEP110.pdf
[12] Elasticsearch Guide, Ingest pipelines; https://www.elastic.co/guide/en/elasticsearch/reference/current/ingest.html
[13] https://github.com/ControlSystemStudio/phoebus/issues/1905
[14] https://github.com/ControlSystemStudio/phoebus/issues/1936