

J-PARC/MLFのための DAQMWユーザガイド

DAQ-Middleware

初版 2018年3月 安 苔次

Contents

1 はじめに	5
2 DAQMWコンポーネントの現状	7
3 改良の詳細	9
3.1 RedisPubSubと関連するDAQMWコンポーネント	9
3.2 DAQMW設定の変更	11
3.3 nGEM用GathererであるGathererPushコンポーネントの追加	12
3.4 日付問題への対策	12
3.5 Logger/Kickerのアルゴリズムの改良	13
3.6 DAQ終了アルゴリズムの追加	13
4 パッケージ	15
4.1 配布	15
4.2 動作確認	15
4.3 ディレクトリ構成	16
5 参考文献	19

Chapter 1

はじめに

このドキュメントは、daqmwmlfバージョン1.4.2を下記のような改良を実施してできた新しいdaqmwmlfバージョン1.5.0について解説したものである。

改良ポイントは下記の通り。

- ・ RedisPubSubの追加と関連するDAQコンポーネントの追加
- ・ nGEM用GathererであるGathererPushの追加
- ・ 日付問題への対策
- ・ LoggerMlf/KickerMlfのアルゴリズムの改良
- ・ DAQ終了アルゴリズムの追加
- ・ MLF配布パッケージの見直し

ここで説明するDAQMWコンポーネントは、新しく作られたものや改良されたものである。その一覧は下記の通り。

- ・ 新規PublisherMlf
- ・ 新規T0EventPublisherMlf
- ・ 新規DaqInfoPublisher
- ・ 新規GathererPsdPub
- ・ 新規GathererPush
- ・ 改良LoggerMlf
- ・ 改良KickerMlf

DAQ-MiddlewareはLinux OSで動作するが、この改良ではRedHat系のOSであるCentOS7・CentOS6で動作が確認されているが、Scientific Linux (CERN)系でも動作すると考えてよい。

Chapter 2

DAQMWコンポーネントの現状

「改良の詳細」に入る前に、J-PARC/MLF用DAQMWコンポーネントの現状について簡単に紹介する。DAQMWコンポーネントはいくつかのグループに分けられる。

検出器からのデータを読み出すGathererと呼ばれるコンポーネントグループがある。例えば、GathererPsdコンポーネントはPSDからのデータを読み出す。

さらに、GatenetコンポーネントやTrignetMlfコンポーネントなどはタイミングやデバイス類の情報を扱うDAQMWコンポーネントである。

また、Gathererからのデータを後段の複数のDAQMWコンポーネントに送るDispatcherMlfコンポーネントがある。

後段のDAQMWコンポーネントには、データをファイルに記録するLoggerMlfコンポーネントやKickerMlf(以前はKicker)コンポーネントなどがある。KickerMlfコンポーネントは、LoggerMlfコンポーネントがイベントカウント情報をログ情報とする代わりに、キッカーカウント情報をログ情報とする。オンラインデータ解析を行うDAQMWコンポーネントやT0インデックスを作成しファイルに記録するT0EventLoggerMlfコンポーネントなどもこのグループに入る。

典型的な構成例をあげてみる。下記の例は、GathererPsdからDispatcherMlfへデータを送り、さらにDispatcherMlfからLoggerMlfとT0EventLoggerMlfにデータを送り、データの記録とT0インデックスの作成・記録を行う。

GathererPsd --> DisptacherMlf --> LoggerMlf, T0EventLoggerMlf

また、新しく作られた、nGEMからデータを読み出すGathererPushコンポーネントやデータをRedisサーバに送る(publishする)PublisherMlfコンポーネントの例では、下記ようになる。この例では、DispatcherMlfからのデータはLoggerMlfコンポーネントによりファイルに記録され、PublisherMlfコンポーネントによりRedisサーバに送られる。Redisサーバについては「改良の詳細」で触れる。

GathererPush --> DisptacherMlf --> LoggerMlf, PublisherMlf

PublisherMlfコンポーネントにより、以前はデータファイルから利用されてきたオンライン解析がRedisサーバから利用できるようになった。

また、nGEMからのデータも後段のDAQMWコンポーネントにデータを送ることができるようになった。以前のNgemMlfコンポーネントは直接データをファイルに書いていて、下流にデータを流していなかった。

さらに、以前より問題として指摘されていた日付問題(詳しくは「改良の詳細」で触れる)は、DAQ-Middlewareの本体とdaqmw.mlfパッケージの改良により解決された。

DAQ終了アルゴリズムの改良では、Gathererのデータ収集終了のタイミングが、後段のDAQMWコンポーネントに正確にわかるようになった。

また、LoggerMlfとKickerMlfの無限ループアルゴリズムが改良され、DAQMWコンポーネント群を管理するDaqOpertorはログ情報を的確に取得できるようになり、それらのコンポーネントはEnd情報をすぐに取得できるようになった。

Chapter 3

改良の詳細

3.1 RedisPubSubと関連するDAQMWコンポーネント

3.1.1 RedisPubSubパッケージ

RedisPubSubは、分散メッセージミドルウェアであるRedisのPUBLISHやSUBSCRIBEコマンドを利用して、いわゆるPublish/Subscribeモデルを実現させたJ-PARC/MLF用のライブラリである。Redisはオープンソースで、インメモリ型のデータストアであり、データベースやキャッシュやメッセージブローカーとして利用されている。RedisPubSubは、MLFにおける分散システムを念頭に、必要な機能を実現させるべく、C++言語で実装された。

DAQMWでは分散オブジェクト技術であるCORBA技術を元にオブジェクトをシリアル化してバイナリーデータとして転送していた。RedisPubSubでは基本的に文字列を含むバイナリーデータを転送する。バイナリーにはシリアル化されたオブジェクトも入れることができる。また文字列にはXML/JSONなどの構造化テキストをオブジェクトからシリアル化して入れることができる。従って、RedisPubSubは、オブジェクト転送と遜色ない機能を実現させるポテンシャルを持つ。

RedisPubSubには、情報を発信するpublish機能と情報を受け取るsubscribe機能がある。チャンネル名(キー名)に対応した値、キー・バリューでデータが扱われる。Publisherは特定のキーにデータを付随してRedisサーバに送ると、Redisサーバはそれを保持し、同じキー名でデータがくるたびに、キューに貯める。一方、Subscriberはあらかじめそのキー名をsubscribeして置くと、Redisサーバではデータが来ると、それが入ってきた順に同じデータを複数のSubscriberは読み出すことができる。そのキー名の特定のデータは、全てのSubscriberが読み出し終わると、Redisサーバから消去される。

このように、Redisサーバは、データの送受信を仲介するメッセージブローカーとして機能する。

3.1.2 PublisherMlfコンポーネント

このコンポーネントは、DispatcherMlf等、前段のDAQMWコンポーネントからデータを受け取り、Redisサーバにチャンネル名(キー名)を持って、そのままpublishする。publishするデータの単位は、送られてくるデータの単位と同じである。publishされたデータは、すでにsubscribeしているSubscriberに送られる。生データのキー名の命名規則は下記の通り。

AAA:edb:AAABBBBBB_CC_DDD_edb

AAAは、装置名で3文字、例えばNVA

BBBBBBはラン番号で6文字の数字、例えば012345

CCはDAQ IDで2文字の数字、例えば00

DDDはModule Numberで3文字の数字、例えば000

このコンポーネントに与えるパラメータは下記の通り。

```
<params>
  <param pid="daqId">0</param>
  <param pid="infoServerHost">localhost</param>
  <param pid="infoServerPort">6379</param>
  <param pid="isPublishing">yes</param>
  <param pid="srcAddr">192.168.0.16</param>
  <param pid="instId">NVA</param>
  <param pid="eventByteSize">8</param>
</params>
```

infoServerHostは、Redisサーバのホスト名

infoServerPortは、Redisサーバの使用するポート番号

3.1.3 T0EventPublisherMlfコンポーネント

このコンポーネントは、T0EventLoggerMlfがT0インデックスをファイルに書く機能を持つのに対して、RedisサーバにT0インデックスをpublishする機能を持つ。T0データの命名規則は下記の通り。生データのedbという文字をt0bに替えたものである。

AAA:t0b:AAABBBBBB_CC_DDD_t0b

このコンポーネントに与えるパラメータは下記の通り。

```
<params>
  <param pid="daqId">0</param>
  <param pid="infoServerHost">localhost</param>
  <param pid="infoServerPort">6379</param>
  <param pid="srcAddr">192.168.0.16</param>
  <param pid="isPublishing">yes</param>
  <param pid="instId">NVA</param>
  <param pid="eventByteSize">8</param>
</params>
```

infoServerHostは、Redisサーバのホスト名

infoServerPortは、Redisサーバの使用するポート番号

3.1.4 DaqInfoPublisherコンポーネント

このコンポーネントは、SubscriberがDaqOperatorからのBegin/Endのタイミングを取得するために作られた。Subscriberはデータを読むためにBeginの情報やデータ処理を終えるためにEndの情報が必要になる場合がある。このコンポーネントはBeginやEndの情報をRedisサーバに投げることで、Subscriberがそれらをsubscribeして情報を取得することができるので、そのタイミングを取得できるようになる。

チャンネル名(キー名)を宣言する。下記の例では、NVA:DaqInfo:string:DaqCommandがチャンネル名となる。

```
<param pid="keyNameDaqCommand">NVA:DaqInfo:string:DaqCommand</param>
```

このキー名で送られるメッセージ内容は当面beginやendなどの簡単なものである。

3.1.5 GathererPsdPubコンポーネント

このコンポーネントは、GathererPsdコンポーネントにFIFOオーバーフロー情報をRedisサーバに投げるコードが追加されたものである。publishされるメッセージのフォーマットはjson形式である。Redisサーバから取り出して処理するプロセスがPythonなら、jsonモジュールのloadsメソッドでそれらのテキストをオブジェクトに簡単に変換できる。GathererPsdには、すでにFIFO読み出しのスレッドを走らせてFIFOオーバーフローなどの情報を定期的に読み出せる仕組みがある。

FIFOオーバーフローなどの値を常時モニターするためには1分に1回とかに情報を読み出す必要がある。Stopwatchクラスで、そのタイミングを知り送り出すことが出来る。

MakeJsonクラスは、簡単なjsonストリングを作成する。それを使って、下記のようなjsonストリングを作り、Redisサーバに送る。

```
{"time":"2017-01-20T10:11:43JST","daq_id":0,"data":[[{"192.168.0.16",444444,0},[{"192.168.0.17",33333,0},[{"192.168.0.18",22222,0}]}}
```

3.2 DAQMW設定の変更

RedisPubSubの追加により、xinetdにサービスに関わる/etc/xinetd.d/bootCompsの一部の変更が必要となる。下記のように、envパラメータを追加すること。

```
service bootComps
{
    |
    env      = LD_LIBRARY_PATH=/opt/mlfsoft/daqmw/lib64
}
```

変更後は、下記のようにxinetdの再起動が必要である。

CentOS7では、

```
sudo systemctl restart xinetd
```

CentOS6では、

```
sudo service xinetd restart
```

3.3 nGEM用GathererであるGathererPushコンポーネントの追加

このコンポーネントは、中性子モニタ用検出器であるnGEMのDAQMWコンポーネントとして使われていたNgemMlfコンポーネントの置き換えである。

置き換えの理由は、NgemMLFが「一人ぼっち」コンポーネントとして下流にデータを流さないで、PublishMlfコンポーネントを使ってオンラインデータ解析ができないためである。

nGEMはNeuNETと違って、データは一度接続されると、いわゆる「垂れ流し」でデータが送られる。NeuNETはリクエストを受けて要求に応じてデータを流す。データフローの視点から、nGEMはpush型であり、NeuNETはpull型である。push型の読み出しモジュールをサポートするという意味でGathererPushという名前をつけた。

3.4 日付問題への対策

3.4.1 改良されたコンポーネント

この改良のコードを選択できるようにするために`#ifdef COMMON_DATE`文を入れた。有効にするためにはファイル`LoggerMlf/FileUtils.h`の中の、すでに挿入してある下記のコードを確認する。

```
#define COMMON_DATE
```

無効にするためにはコメント文にする。

```
//#define COMMON_DATE
```

この選択肢を設けた理由は、今までにすでにインストールされているDAQMWでも、これらのコンポーネントを走らせることができるようにするためである。新しいDAQMWへのアップデートが難しい場合や日付問題への対策を必要としない装置グループでも、これらのコンポーネントを利用できる。

3.4.1.1 LoggerMlf

基本的な変更箇所は、自分の計算機のdateを取得する部分を、DaqOperatorが提供するdateを取得するように変更したところである。主にFileUtilsが改良された。ParameterClient(LoggerMlf)/ ParameterServer(DaqOperator)機能を使って取得する。

3.4.1.2 KickerMlf

以前のKickerはもともとLoggerMlfのFileUtilsを使用していたので、ほとんど変更はない。名前はLoggerMlfとの名前の整合性のためKickerMlfとした。

3.5 Logger/Kickerのアルゴリズムの改良

従来のLogger/Kickerのdaq_run()メソッドのアルゴリズムで、無限ループを作っていた問題。以前より、「Kickerコンポーネントによって読み出されたT0カウントがIROHA2でなかなか更新されない状態があった。」理由は、これにある。DAQMWの作法で、daq_run()の中で無限ループを作るのはご法度。永遠に抜け出れなくなり、コマンドを受け付けなくなるので。つまり、T0カウントを読み出すためのコマンドが受け付けられにくい状態になっていた。

これを改良するために、無限ループをなくし、いつでもコマンドが受け付けられるようにした。

3.6 DAQ終了アルゴリズムの追加

DaqOperatorからのEndコマンドを受け取っていても、後段のコンポーネントは、Gathererの読み出し終了タイミグを知る確実な方法がなかった。従って、後段のコンポーネントは読み出しのタイムアウトを繰り返し、Gathererからのデータが来ないこととするアルゴリズムを取らざるを得なかった。しかし、データの最後を示すマークつけたデータを処理するアルゴリズムを実装することで、この問題を解決した。

GathererPushに実装したメソッドでの例を挙げると、下記の通り。

```
int set_header(unsigned char* header, unsigned daqId, unsigned ModNo, unsigned data_byte_size);  
void set_end_mark(unsigned char* footer);
```

これらを利用している。

また、LoggerMlfに実装したメソッドの例を挙げると、下記の通り。

```
bool check_end_mark(unsigned char* footer);
```

これを利用している。

この改良のコードを選択できるようにするために#ifdef END_FLAG_文を入れた。有効にするためにはファイルLoggerMlf/LoggerMlf.hの中の、すでに挿入してある下記のコードを確認する。

```
#ifdef END_FLAG_
```

無効にするためにはコメント文にする。

```
//#ifdef END_FLAG_
```

有効を選択した場合でも、データの最後を示すマークを持つデータが来なかった場合、無効を選択した場合と同じように処理される。

Chapter 4

パッケージ

改良の視点は、実験に使用されるもの、テストに使用されるもの、すでに使われなくなったものなどを区別し、検証すべきコンポーネントを明確にしようというものである。また、以前のKickerコンポーネントはパッケージに入っていなかったが、今回は含めた。以前各BL担当者から頂いた構成ファイル、config.xml/condition.xmlもパッケージに入れた。これにより、どのようなコンポーネントが実験に使用されているかがわかる。

今後も構成ファイルを更新して、現状を把握し、検証対象を明らかにし、構成を変更してゆく。

4.1 配布

茨城大学の細谷さんがサポートしているシンチレータ検出器関連のパッケージ、astraをMLFパッケージに取り込む課題がある。当面は、新しいdaqmwmlfパッケージには含めず、astraパッケージをサポートSVNに入れる。daqmwmlfおよびastraは、MLF計算環境プロジェクト管理サーバのSVNで配布される。

<https://mlfdevsv.j-parc.jp/MlfSoftware/rep/daqmwmlf>

<https://mlfdevsv.j-parc.jp/MlfSoftware/rep/astra>

4.2 動作確認

ここで述べるDAQMWコンポーネントは新しく作成したものと改良したものについてのみ触れる。動作確認には、パッケージのコンパイルとDAQMWコンポーネントを走らせて正しく動作しているかの確認がある。

4.2.1 パッケージのコンパイル

パッケージはCentOS7.3以降及びCentOS6.8以降についてmake;make installが問題なく動作したことを確認している。

4.2.2 DAQMWコンポーネントの動作検証

4.2.2.1 ビーム実験で検証されているもの。

T0EventPublisherMlfは、ビームテストを行い動作を検証した。その他のDAQMWコンポーネントはNOVA(BL21)において半年近く動作しており、不具合はなかった。

LoggerMlf

KickerMlf

PublisherMlf

DaqInfoPublisher

T0EventPublisherMlf

GathererPush

4.2.2.2 エミュレータで検証されているもの。

まだビームテストでの動作検証はなされていないが、ソフトウェア・エミュレータを使って動作の検証が行われた。

GathererPsdPub

4.3 ディレクトリ構成

パッケージはMLF標準ディレクトリである/opt/mlfsoft/daqmwで利用される。ディレクトリの構造は下記の通り。

bin/ バイナリが入る

include/ includeファイルが入る

docs/ 従来から入っていたドキュメントが入る

documents/ 新しく作られたドキュメントが入る

lib64/ src/libにあるライブラリが入る

src/ mlfソースコードが入る

experiments/ legacies/ lib/ Makefile test/

etc/ 各装置グループやサンプルの構成ファイルが入る

condition.json@ condition.xml@ config.xml@ experiments/ samples/

test/ テスト用のスクリプト等が入る

srcの中の

experiments/ 実際に実験に利用される(予定の)DAQMWコンポーネントが入る

DaqInfoPublisher/ GathererPush/ MonitorEmaki/

T0EventLoggerMlf/ DispatcherMlf/ KickerMlf/ MonitorPsd/

T0EventPublisherMlf/ Gatenet/ Kicker.orig/ ParameterClient/

TrignetCp2Mlf/ GathererPsd/ LoggerMlf/ PublisherMlf/ TrignetMlf/

GathererPsdEr/ LoggerMlf.orig/ ServiceWrap1x/

GathererPsdPub/ Makefile ServiceWrap2x/

lib/ ライブラリが入る

Makefile makejson/ mlfdag/ redispubsub/ stopwatch/
test/ テスト用DAQMWコンポーネントが入る
DataGen/ Makefile PlayerEmaki/
legacies/ すでに使われなくなったDAQMWコンポーネントが入る

Chapter 5

参考文献

1. DAQ-Middlewareホームページ : <http://daqmw.kek.jp/>
2. J-PARC/MLF計算環境 : <http://mlfuser.cross-tokai.jp/ja/compenv.html>
3. J-PARC/MLFのためのdaqmwmlfパッケージとユーザガイド
 - package : <https://mlfdevsv.j-parc.jp/MlfSoftware/rep/daqmwmlf>
 - PDF : <http://nova-info.intra.j-parc.jp/mlfsoft/daqmwmlf/usersguide/DAQMWforMlfUsersGuideA4.pdf>
 - Web : <http://nova-info.intra.j-parc.jp/mlfsoft/daqmwmlf/usersguide/index.html>
4. J-PARC/MLFのためのRedisPubSubプロジェクト
 - package : <https://mlfdevsv.j-parc.jp/MlfSoftware/rep/redispubsub>
 - PDF : <http://nova-info.intra.j-parc.jp/mlfsoft/redispubsub/usersguide/RedisPubSubUsersGuideA4.pdf>
 - Web : <http://nova-info.intra.j-parc.jp/mlfsoft/redispubsub/usersguide/index.html>
 - doxygen : <http://nova-info.intra.j-parc.jp/mlfsoft/redispubsub/doxygen/index.html>