

データ収集ソフトウェアの変遷

五十嵐 洋一

KEK

20234/11/18

はじめに

ちょっと日本の DAQ の歴史を振り返りながら茶飲み話をしてみよう

自分の周りの話、知らない話も多いので言及できなかったものはすいません。

1980's

- DEC (Micro)VAX/VMS
 - KEK-PS, 理研、核研 等
- MC680x0 VME SBC/OS-9 VX-WORKS
 - 組み込み用 UNIX 風 Realtime OS
 - Belle の初期 DAQ 等
- PC-9801/MS-DOS
 - C-bus → (Q-bus) → CAMAC CC
 - 割り込み処理によるマルチタスク
 - KODAQ (小俣和夫氏)
- Transputer/occum
 - マルチプロセッサによるマルチタスク処理。
 - 静的な負荷分散、プログラムにどこでどのようにタスクが走るかが記述されている。
 - プロセッサがデータリンクを持っていて他のプロセッサと通信できる。
 - プログラムの中にあらわにデータリンクが表現される。
 - 幾つかの大型実験 で L3 トリガーなどに使われた。
 - KEK-PS での活用は部分的
- Neural network
 - トリガーに使おうとしたが、当時の技術では、解析の初期分類にしか使えなかった。

組み込みシステムといえ Realtime システムだった。

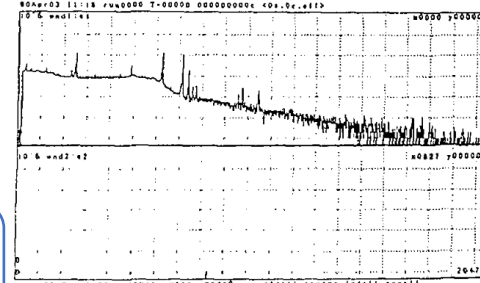


Figure 4: An example of the KODAQ display.

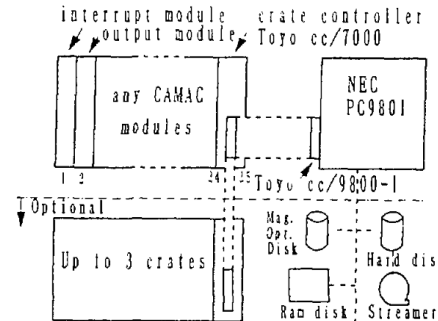


Figure 5: Minimum hardware requirement of the KODAQ.

RIKEN ARF Data Acquisition System

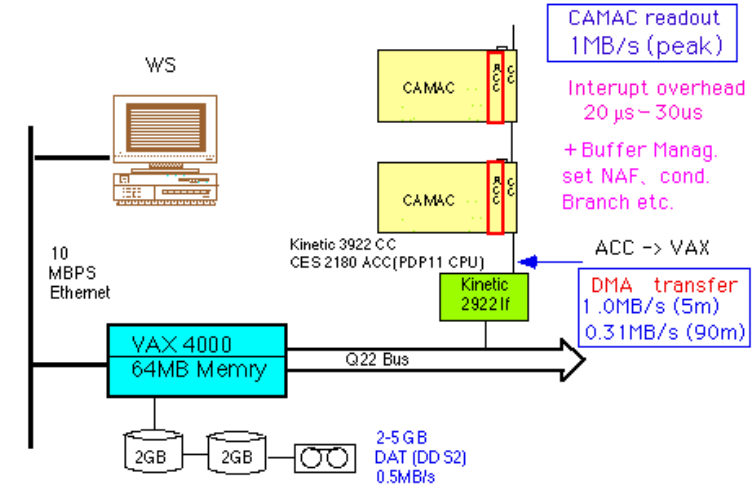
- RIKSVX:: VAX-4000 105A (RIPS counting room)
- GARISF:: VAX-4000 106A (J1 counting room)
- SMARTF:: VAX-4000 106A, Ring B2F E4 prep. room.
- RIKMW1:: Micro VAX II, Ring 1F Counting Room
- RIKMW2:: Micro VAX II, Ring B2F J1 Room
- RIKMW3:: Micro VAX II, Linac 1F Counting Room
- RIKMW4:: Micro VAX II, Ring B2F E6 prep. room
- RIKVS3:: VAX station 4000-60, Ring B2F, E6prep. room.

CES 2180 ACC (Auxiliary Crate Controller)

16 MHz J11 CPU (PDP11 CPU)
128KB S-RAM

CAMAC READ (Register Read: mov (R5) r0) 1.4 μs /16 bit
CAMAC READ (Memory Read: mov (R5) (R1)+) 2.1 μs /16 bit

CAMAC → ACC



T. Ichihara et al. IEEE Transaction on Nuclear Science, 36-5 p. 1628 (1989).

1990's

UNIX で DAQ

- Realtime system は DAQ に必須ではない。
 - 必要だったのは
 - ハードウェアに対する簡単なインターフェース
 - マルチタスク
- UNIX IPC (Inter Process Communication) によるマルチプロセス
 - Signal, Message, Shared memory
 - UNIDAQ
 - MIDAS (Stefan Ritts)
 - Online Database を中心にデザイン

時代背景

- WorkStation
 - RISC processor
 - Sun Sparc station (Sparc), DEC DECstation (MIPS Rxxxx)
- DSP によるアクセラレーション
 - ハードウェアアーキテクチャによる高速信号処理
- Web の台頭
 - NCSA Mosaic, Windows95
- PC UNIX の台頭
 - PC BSD (386BSD, FreeBSD, NetBSD, OpenBSD, ...)
 - Linux
- 次世代ネットワーク・通信
 - Token Ring, ATM, Fiber channel
- PLD から FPGA の時代に
 - 簡易言語 xxASM から HDL

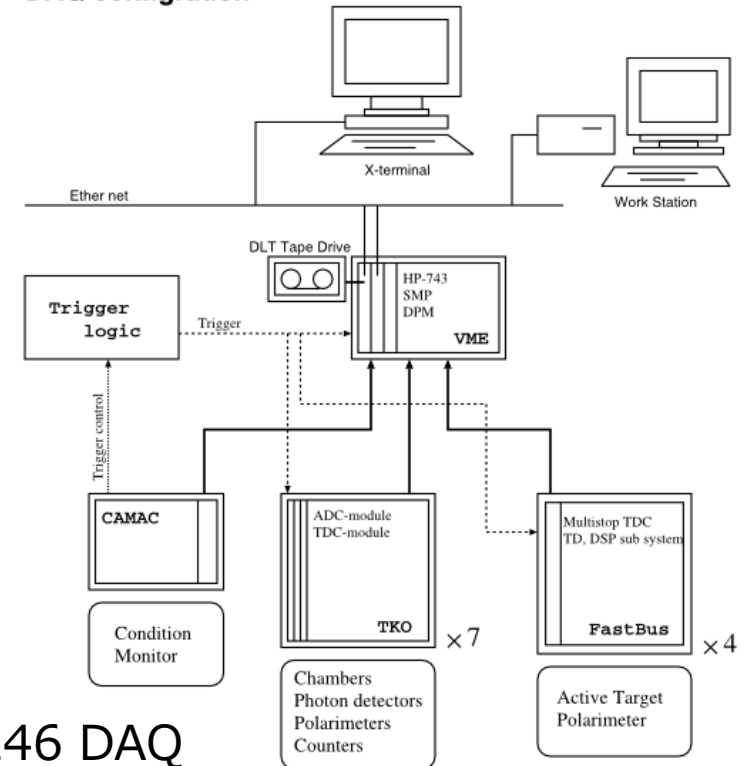
RISC vs CISC

どのようにして複数のサブラックを統合するか?
どのようにして読み出しチャンネル数を増やすか?

→ 異なる Bus をつないで統合する。

- Bus interface
- Memory interface
- Reflective memory
- G-link
- ...

DAQ configuration



KEK-PS E246 DAQ

2000's

- Belle DAQ upgrade
- Network 指向のリードアウトモジュール
 - KEK-VME, COPPER
 - Front-end でのデータバッファリング
 - Global Busy を避けて出来るだけ小さい範囲で Busy をかける
 - Event identification
- DAQ software KiNOKO (榎本三四郎氏)
 - KamLAND
- EPICS の台頭
 - 加速器への採用が広まる

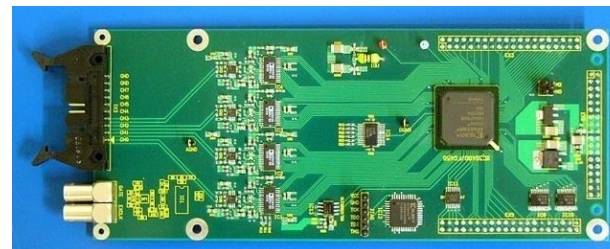
• SiTCP 開発

時代背景

- Web の時代
- Object 指向
- RISC から PC へ Intel 一強



On board PC base Readout platform module
Network をハンドリングするために PC を搭載した。



Front-end A/D mezzanine card

- 500MHz, 8bit FADC
- 65MHz, 12bit FADC
- TMC
- MWPC readout
- ...

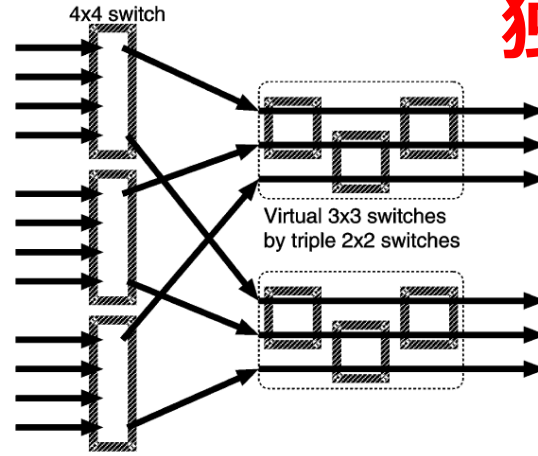
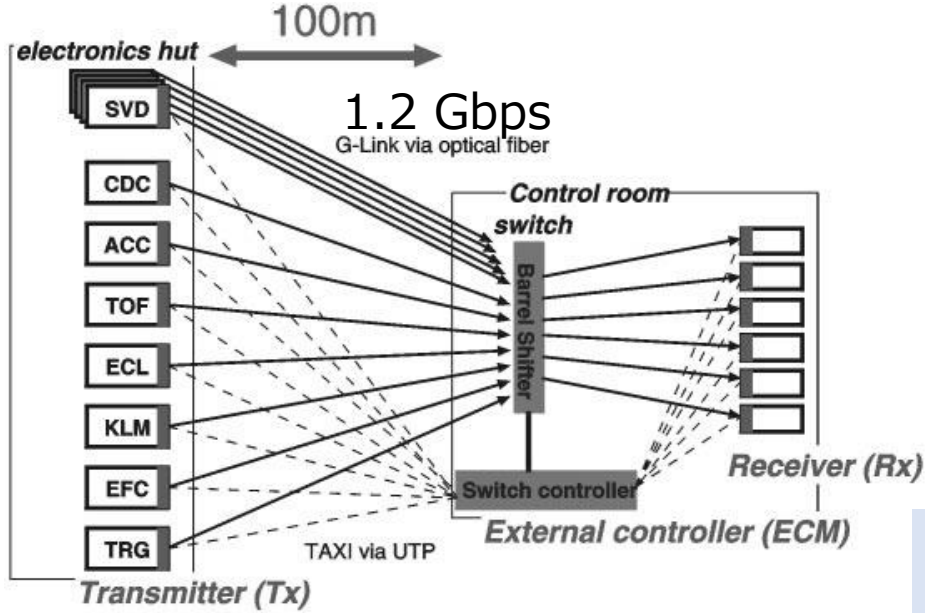


KEK-VME general I/O
(NIM/ECL/LVDS)

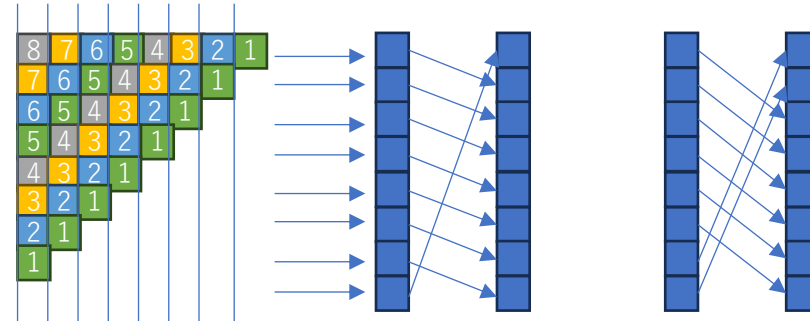
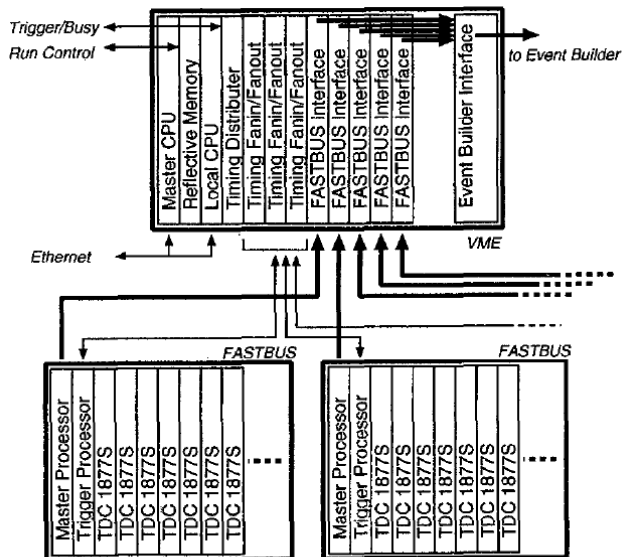
NIM の論理回路を
FPGA に置き換える

Belle Event builder upgrade 2000

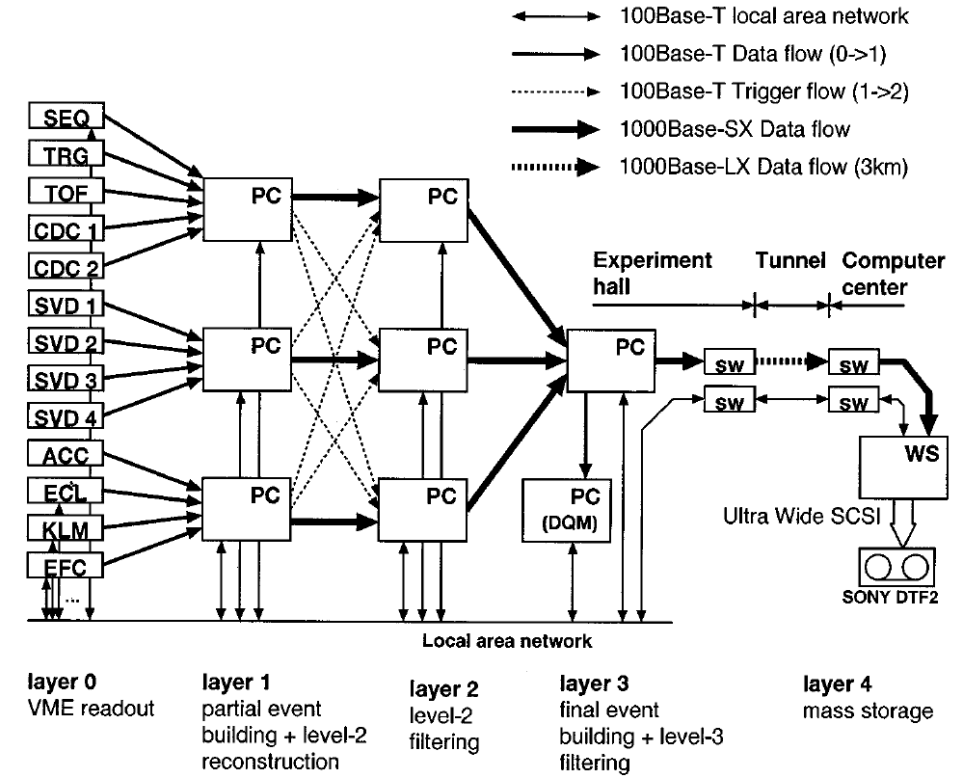
独自技術から一般技術の応用へ



4x4, 2x2 switch による
12x6 barrel shifter



Barrel shifter event building

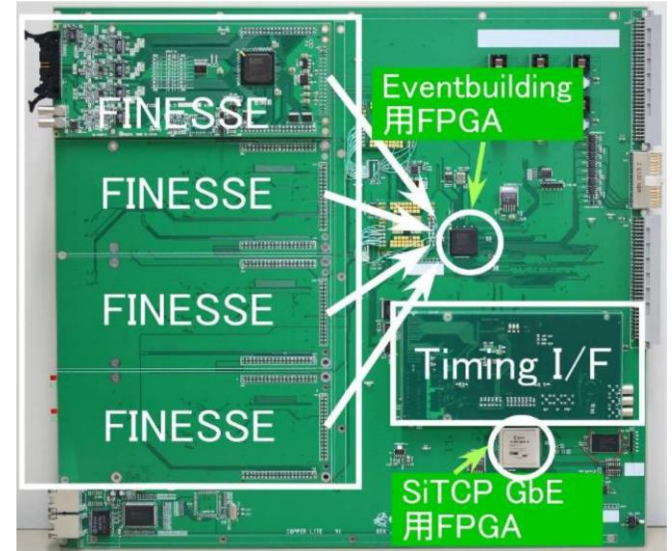


Switch-less network event building

Fig. 2. A view of a typical subsystem.

2010' s

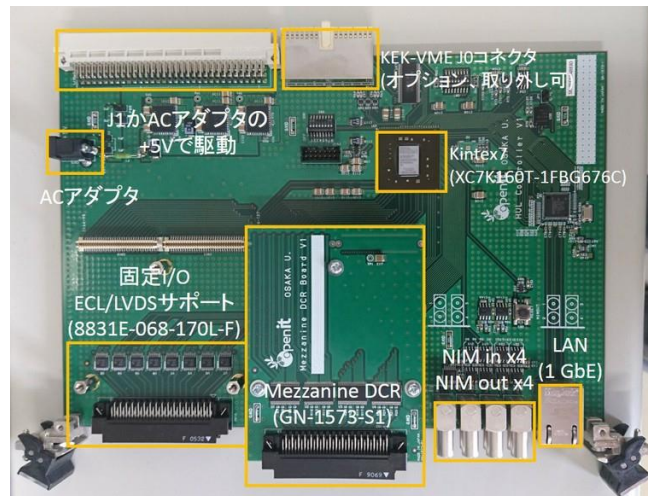
- J-PARC HD 立ち上げ
- Network 指向のリードアウトモジュール
 - COPPER-lite
 - SiTCP を搭載したリードアウトモジュール
- Network based DAQ
 - **HDDAQ**
 - **DAQ-Middleware**



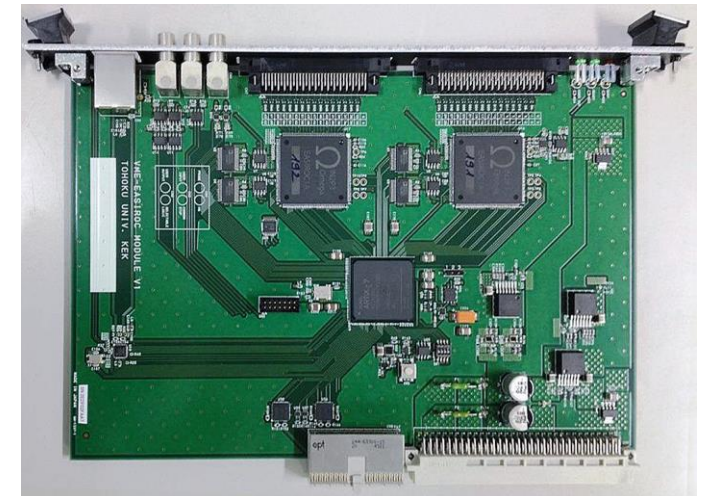
FPGA base Readout platform module

時代背景

- Web の多用途化
 - Web browser が処理系に進化
 - Java、JavaScript
- Ethernet 一強
- FPGA の時代
- スマホの台頭



HUL FPGA Readout board



MPPC Readout board

Network based DAQ system (HDDAQ)

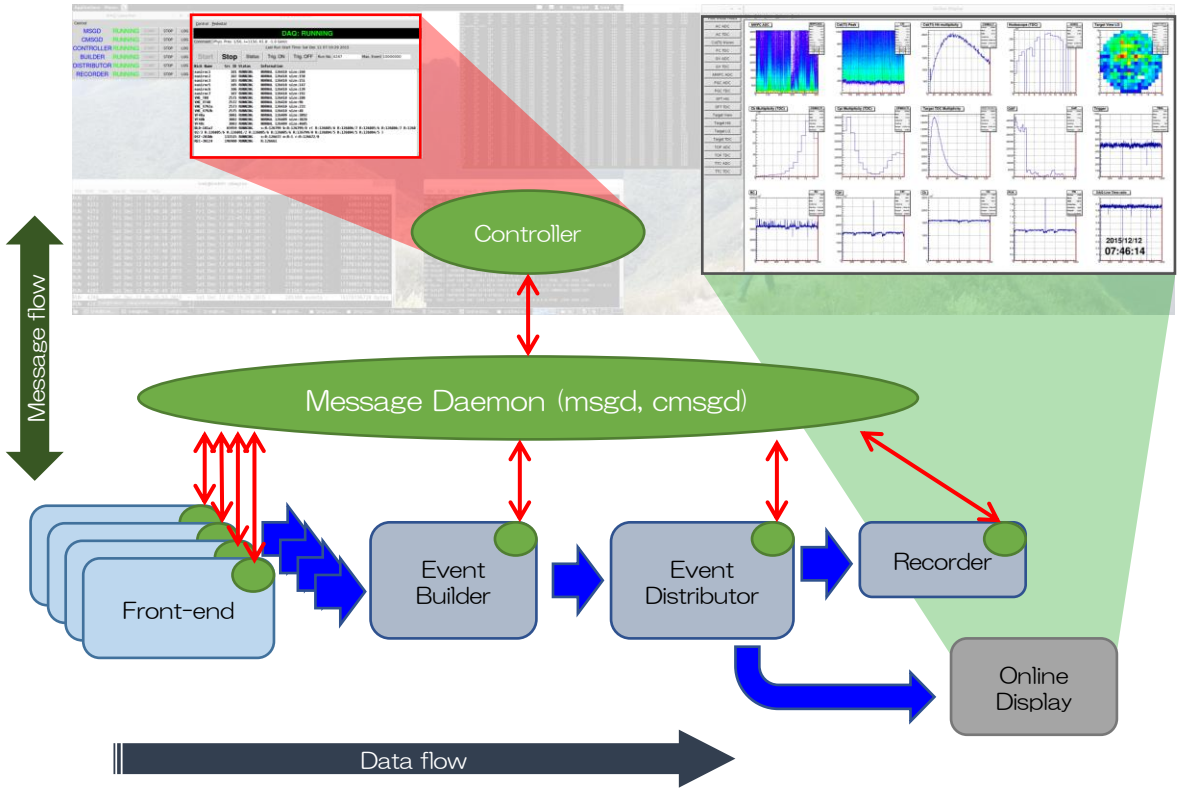
- すべての機器がネットワークと通信できるようになって来た。
- 多数で多様な機器からデータを統合して集めたい。
- 少人数でメンテナンス出来るように
 - 実験を行う人がどのように動いているか理解できるように。
 - **理解していないものは運用できない。**
- 長期の使用に耐えるような移植性
- 機器の変更、更新が容易に出来るように

- 最小のコードで必要な機能実装されたソフトウェア
 - 標準規格以外のライブラリを出来るだけ用いない
 - C++, Python, Root
- シンプルなイベント同期システム



**何がどのように動いているかが解る。
ユーザー自身によるメンテナンス、改良が可能**

Network based DAQ system (HDDAQ)



DAQ software (HDDAQ)

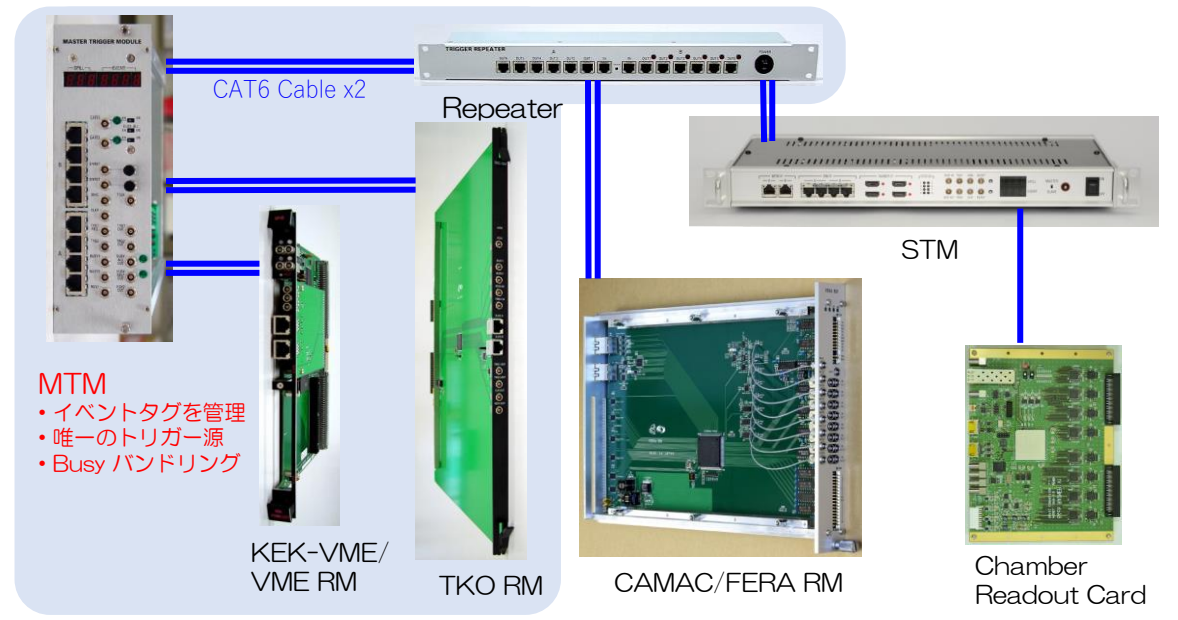
- 単機能の多数のプロセスによる協調動作
- Message Path(制御通信), Data Path(データ通信)の2種類の通信
- すべての通信は TCP/IP

主に J-PARC HD や原子核実験などで利用

Event synchronization

MTM/RM トリガー配布システム

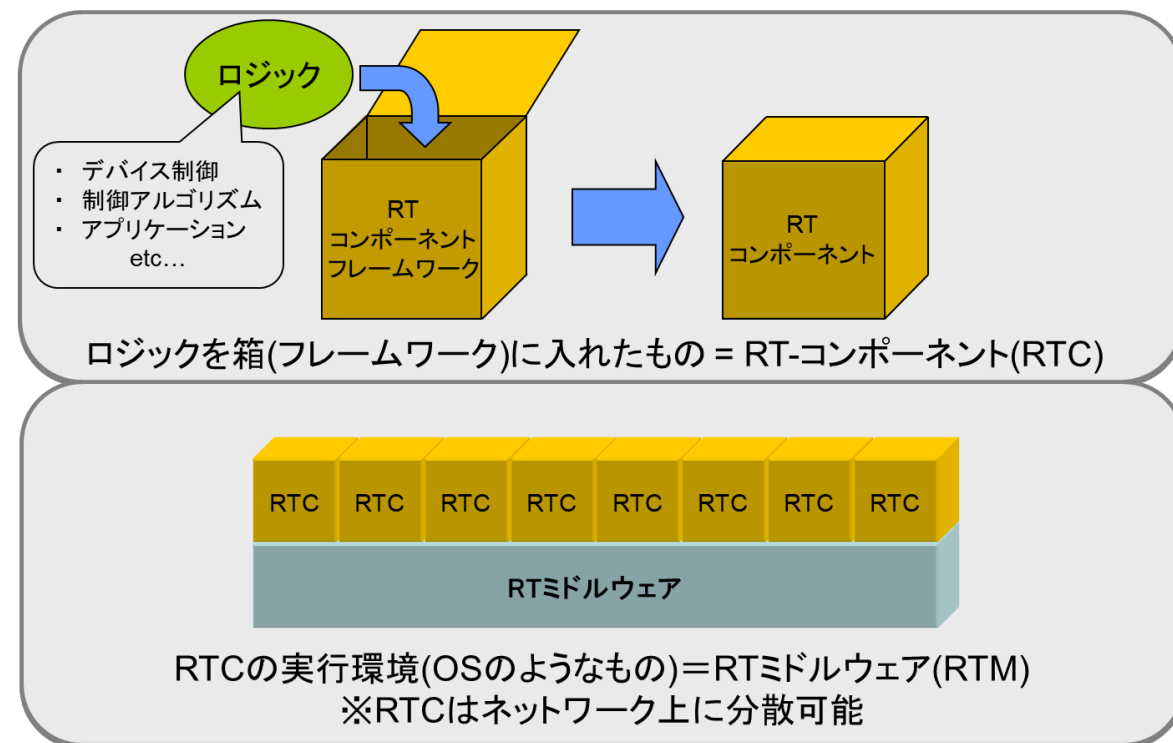
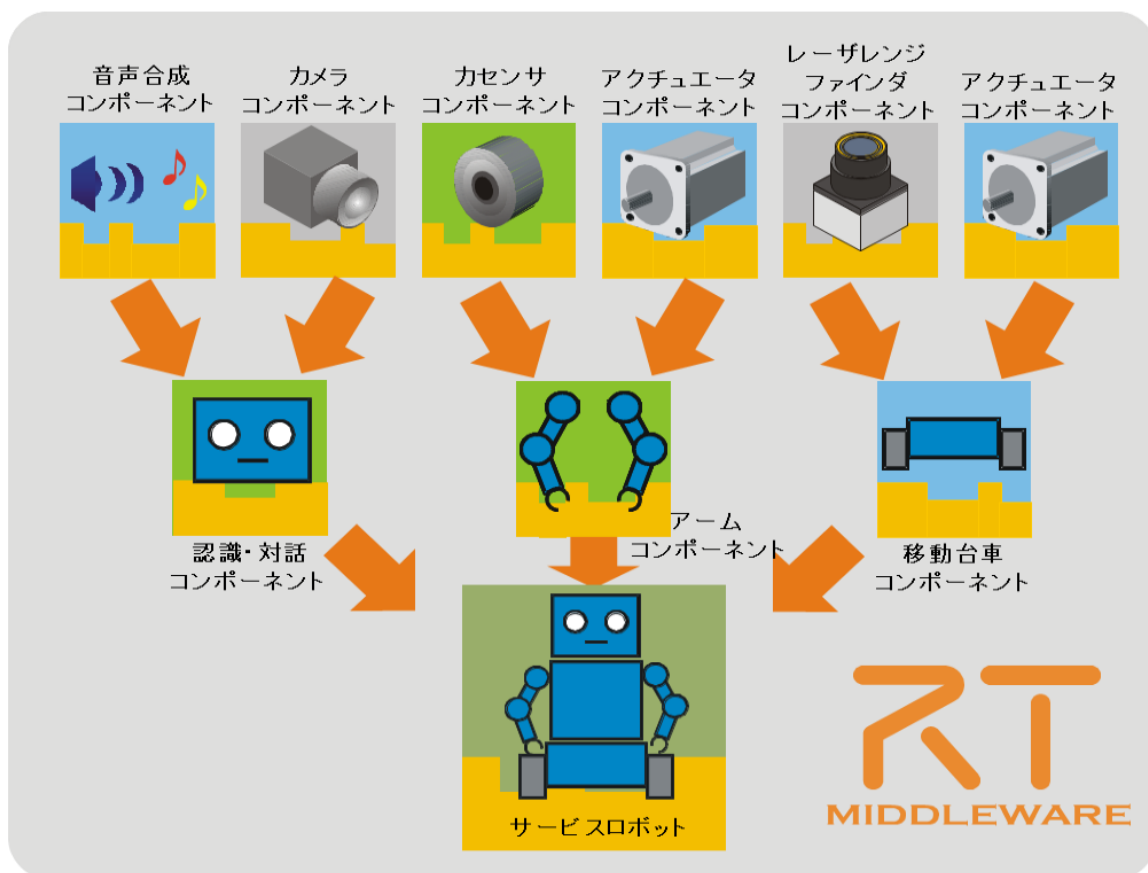
- マルチイベントバッファを持ったサブシステム間でイベントを同定する。
- Trigger/Busy ハンドシェイク
- Trigger 毎に唯一のイベントタグを各サブシステムに送る。



Robot Technology Middleware (aist)

• Robot を制御するためのミドルウェア

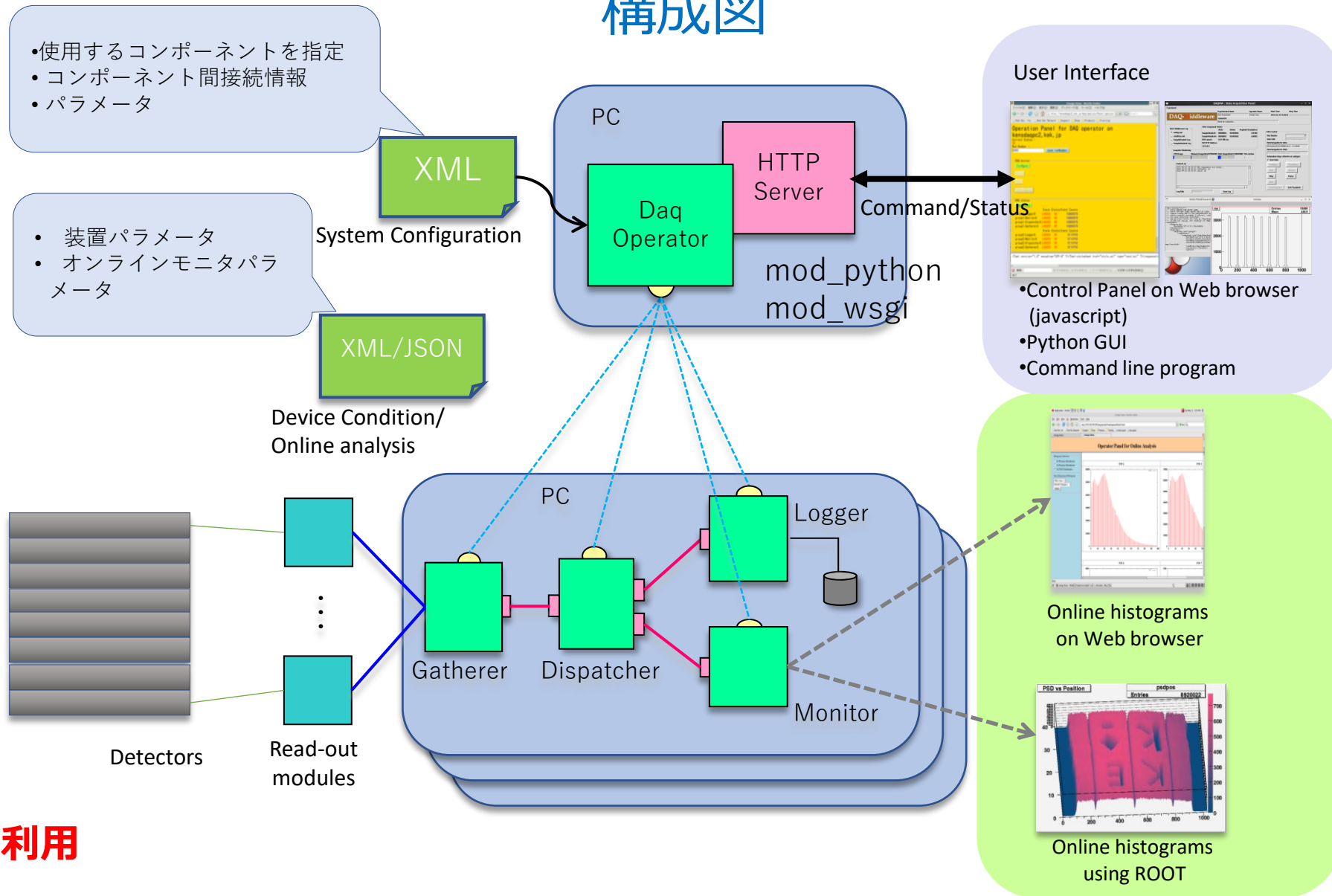
- コンポーネントどうしがオブジェクト通信を行いロボットの制御を行う。



RTミドルウェアによるRTシステムインテグレーション

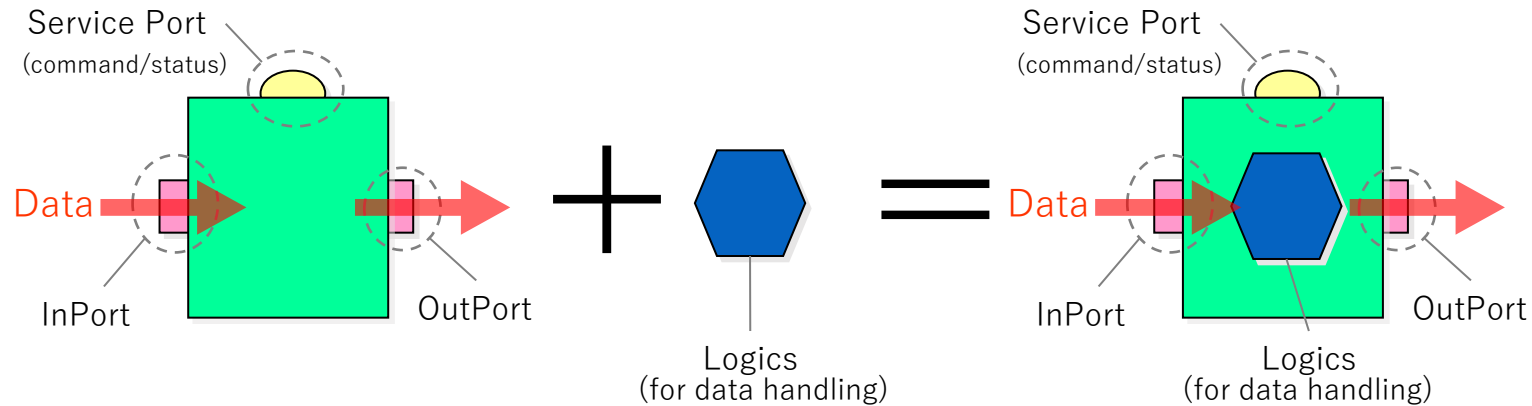
RTM の DAQ への適用 DAQ-Middleware

構成図



主にMLFで利用

DAQ-Middleware DAQ component



- DAQコンポーネントを組み合わせてDAQシステムを構築する。
- 上流からのデータを読むにはInPortを読む。
- データを下流に送るにはOutPortに書く。
- DAQコンポーネント間のデータ転送機能はDAQ-Middlewareが提供する
- ユーザーはコアロジックを実装することで新しいコンポーネントを作成できる。

コアロジックの例：

- リードアウトモジュールからのデータの読み取りロジック
- ヒストグラムの作成ロジック

DAQ-Middleware 歴史

2006 - 04 KEK測定器開発室次世代DAQプロジェクトにて開発開始

2008 - 05 MLF中性子 Day 1 (BL20 iMATERIA) で使用

2010 - 04 Open-Itに移行

2010 - 08 (1.0.0) MLF中性子依存をとりのぞいたバージョンをリリース

2011 - 06 (1.1.0) x86_64のサポート

2012 - 06 (1.2.0) Scientific Linux 6のサポート

2013 - 11 (1.3.0) インポート (データを受け取るポート)の設定を可能できるように変更

2015 - 06 (1.4.0) Scientific Linux 7のサポート

2019 - 04 (1.4.3) ライセンスをGNU LGPLに制定

2022 - 02 (1.5.0) AlmaLinux 8のサポート

2024 年度内 (1.6.0 予定) AlmaLinux 9 のサポートで更新終了

2020's

- 高速ネットワーク、分散環境
- ソフトウェアトリガー・イベント削減
 - (Belle 等規模の大きな実験では 2000年ころには実現)
- トリガーレス/Streaming DAQ
 - ずっと議論されてきたが技術が High rate な実験に追い付いてきた。
 - Over 1Gbps network, GPU, 分散コンピューティング, 強力な FPGA

ZeroMQ(通信) + FairMQ(状態遷移機械、フレームワーク) + redis(全体管理)
→ NestDAQ (Network based streaming DAQ)

- FairMQ の状態遷移機械、被制御部分を利用。
- 全体管理、制御には Redis を使用
 - キー・スペース通知
 - PUB/SUB → 制御に使える

ØMQ

Redis



NestDAQ Process 構造

NoSQL/Key-Value 型データベース **redis** ですべてのプロセスの状態管理と制御をおこなう。

FairMQ Plugin

• DAQ Service Plugin

- Run control
 - State machine の制御
 - Run number の設定
- **Service discovery**
 - 接続設定の半自動化

• Metrics Plugin

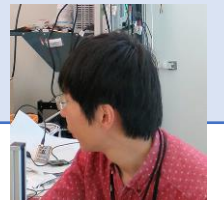
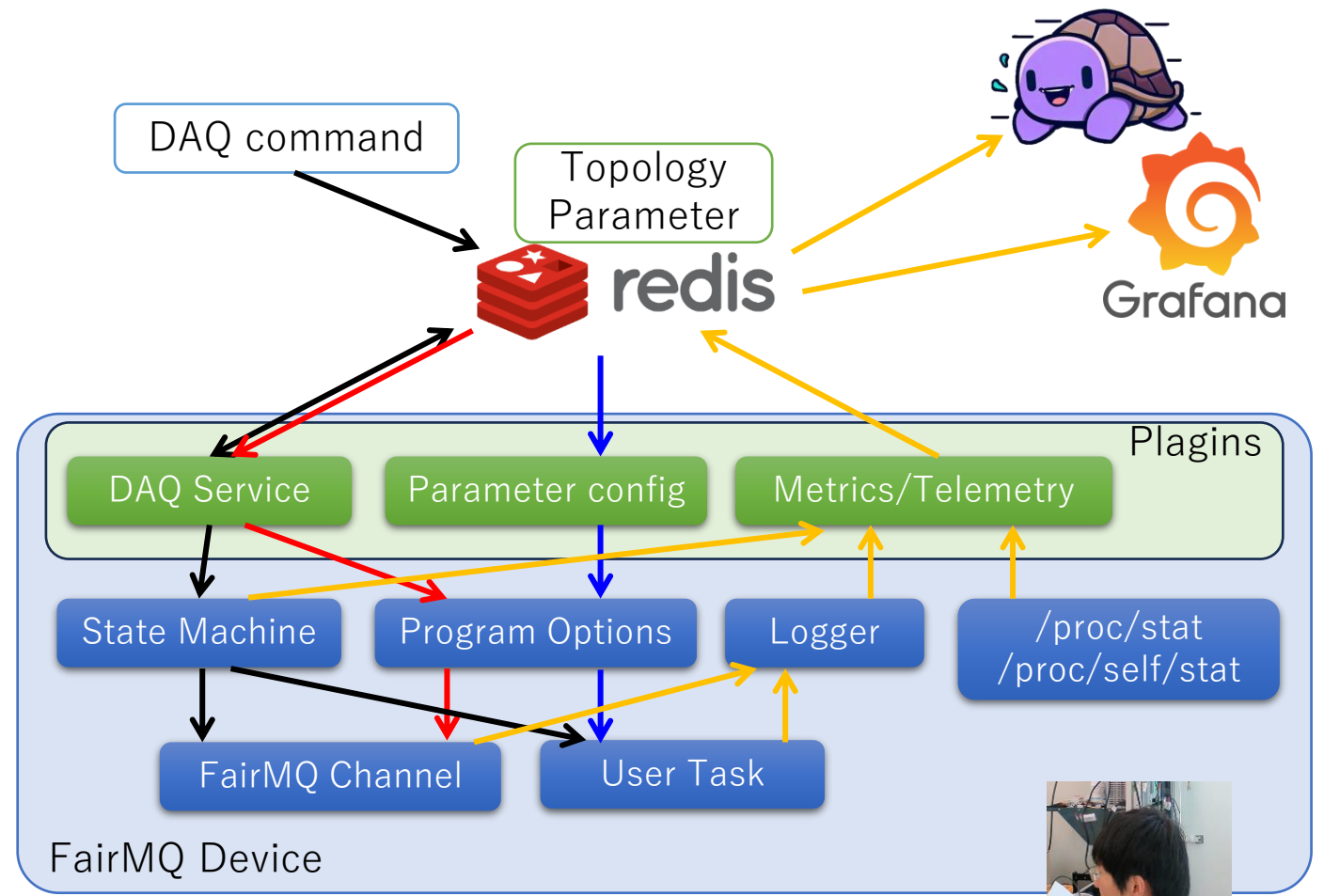
- Process の状態把握

• Telemetry Plugin

- データ処理向けのログ出力

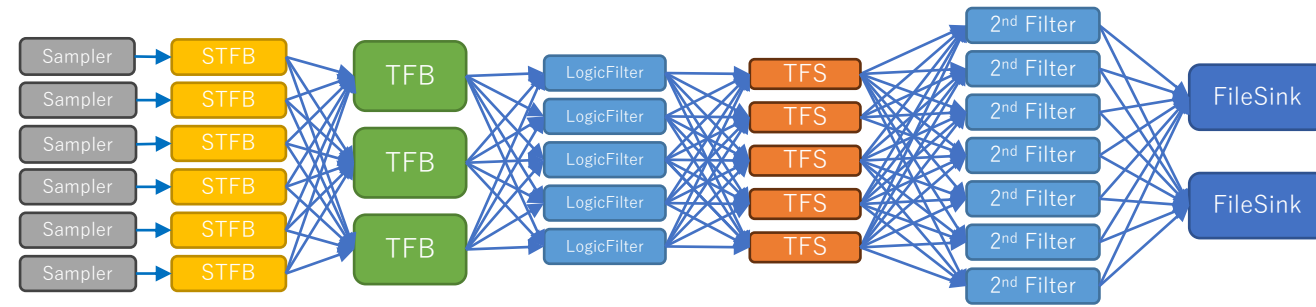
• Parameter config Plugin

- ハードウェア初期化等のパラメータをデータベースから読み込む

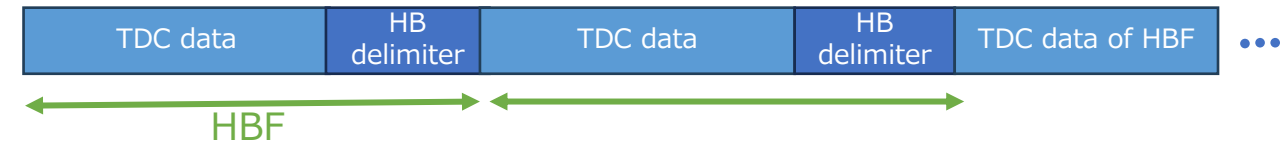


実際のデータ収集プロセス

実際の NestDAQ を使用したプロセス構成例



読み出し機器(TDC)のデータストリーム



連続的に読み出す読み出し機器

- 全部の機器の時間が同期されていること
- 長期間にわたって時間の同定が出来ること
 - 番号を持ったタイムフレーム → ハートビートフレーム(HBF)
 - 現在のハードビートフレームの長さ: 524.288 us
 - (125 MHz クロック, 16bit カウンタ)

• Sampler

- 読みだ機器からデータを読み出す。

• Sub-Time Frame Builder (STFB)

- データからハートビートフレームを切り出し、いくつかのハートビートフレームをまとめてサブタイムフレームを生成する。

• Time Frame Builder (TFB)

- 多数の STFB からサブタイムフレームを集めてきてすべての測定器の同じ時間区間のデータを持つタイムフレームを生成する。

• 第一段ソフトウェアトリガー/フィルター

- タイムフレームを処理して目的のイベントを見つけ出してその時間をTFデータに追加する。

• Time Frame Slicer (TFS)

- トリガー時間の近傍のデータを切り出してまとめる。

• 第2段ソフトウェアトリガー/フィルター

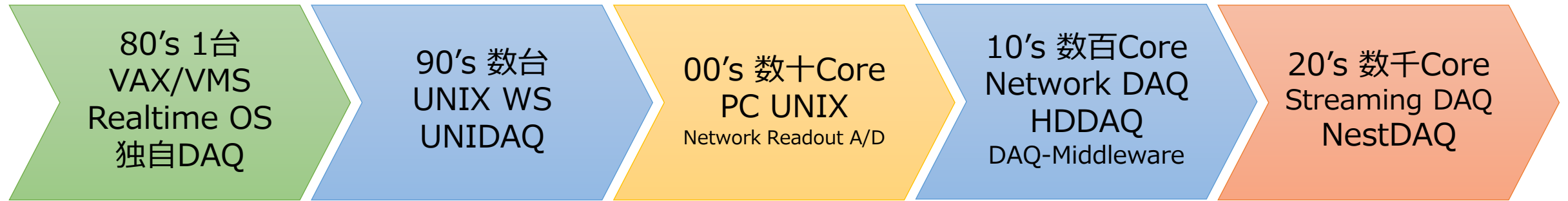
- TFS によってまとめられた狭いタイムフレーム(イベント)を処理して確度の高いイベントを抜き出す。

• FileSink

- データをファイルに書き出す。

まとめ

時代を経るにつれて DAQ に使用する計算機 (Core 数) の数は増えていく。



現在目指しているもの

- 数千 Core 環境で稼働できるような DAQ
 - スケーラビリティがあるシステム、1台から数千まで
- ある程度広い実験に適用
- 構築、メンテナンスコストを低くしたい
- Software 化された Trigger (NIM 回路 がいらぬい)

} ここはそんなに変わらない