XDAQ based software for the SDHCAL Data acquisition Status and news

L. Mirabito,C.Combaret

IPN Lyon, IN2P3, CNRS

September 10, 2010

Introduction

Introduction

Constraints DIF based readout of SDHCAL

- Synchronous readout and data storage several DIF per plane
- Distributed readout and computing

Introduction

Constraints DIF based readout of SDHCAL

- Synchronous readout and data storage several DIF per plane
- Distributed readout and computing

Needs

- Configuration framework
- Communication framework
- Analysis framework

Introduction

Constraints DIF based readout of SDHCAL

- Synchronous readout and data storage several DIF per plane
- Distributed readout and computing

Needs

- Configuration framework
- Communication framework
- Analysis framework

Use existing frameworks

- XDAQ CMS DAQ
- MARLIN ILC analysis

XDAQ

Configuration

- Executive : Web server dynamically loading application
- Each application appears as a servlet
- XML description of all executives and applications

XDAQ

Configuration

- Executive : Web server dynamically loading application
- Each application appears as a servlet
- XML description of all executives and applications

Communication

- Inside an Executive : Zero-copy
- Inside a PC: unix sockets
- Between PCs: tcp/ip sockets

XDAQ

Configuration

- Executive : Web server dynamically loading application
- Each application appears as a servlet
- XML description of all executives and applications

Communication

- Inside an Executive : Zero-copy
- Inside a PC: unix sockets
- Between PCs: tcp/ip sockets

User interface

- Browser access of each application (CGI based)
- Web2 technology (AJAX, GWT)
- SOAP binding of the application

XML configuration

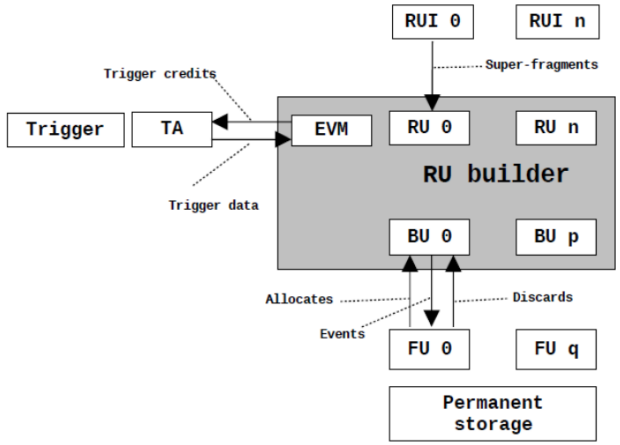
```
<xc:Partition


<!-- Binary Network definition -->
  <i2o:protocol xmlns:i2o="http://xdaq.web.cern.ch/xdaq/xsd/2004/I2OConfiguration-30">
    <i2o:target class="DIFSupervisor" instance="0" tid="130"/>
    <i2o:target class="DIFSupervisor" instance="1" tid="131"/>
    <i2o:target class="BackupSaver" instance="0" tid="170"/>
    <i2o:target class="RUCollector"  instance="0" tid="37"/>
    <i2o:target class="LocalManager"  instance="0" tid="38"/>
    <i2o:target class="rubuilder::ru::Application"  instance="0" tid="41"/>
    <i2o:target class="rubuilder::bu::Application"  instance="0" tid="42"/>
    <i2o:target class="rubuilder::evm::Application"  instance="0" tid="43"/>
    <i2o:target class="rubuilder::fu::Application"  instance="0" tid="44"/>
    <i2o:target class="MarlinAnalyzer"  instance="0" tid="45"/>
    <i2o:target class="pt::atcp::PeerTransportATCP"  instance="0" tid="47"/>
  </i2o:protocol>

<!-- One executive definition -->

  <xc:Context url="http://lyoac20:10000">
    <xc:Endpoint hostname="lyoac20" network="dhcalatcp" port="31805" protocol="atcp" service="i2o" />

 <!-- DIF supervisor #0-->
    <xc:Application class="DIFSupervisor" id="30" instance="0"  network="local">      <properties xmlns="urn:xdaq-application:DIFSupervisor" xsi:type="soapenc:Struct"
<UseBackup xsi:type="xsd:boolean">false</UseBackup>
<UseShm xsi:type="xsd:boolean">true</UseShm>
<UseCCC xsi:type="xsd:boolean">true</UseCCC>
<UseDB xsi:type="xsd:boolean">false</UseDB>
<ASICType xsi:type="xsd:integer">2</ASICType>
<ASICHeaders xsi:type="xsd:string">1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
<DIF_Identifier xsi:type="xsd:string">FT101002</DIF_Identifier>
      </properties>
    </xc:Application>
...
<!-- Library to load -->
 <xc:Module>/opt/xdaq/lib/libptatcp.so</xc:Module>
 <xc:Module>/usr/local/lib/libftd2xx.so</xc:Module>
 <xc:Module>/data/online/opt/dhcal/lib/libRUCollector.so</xc:Module>
 <xc:Module>/data/online/opt/dhcal/lib/libDIFSupervisor.so</xc:Module>
...
```

The CMS Event Builder

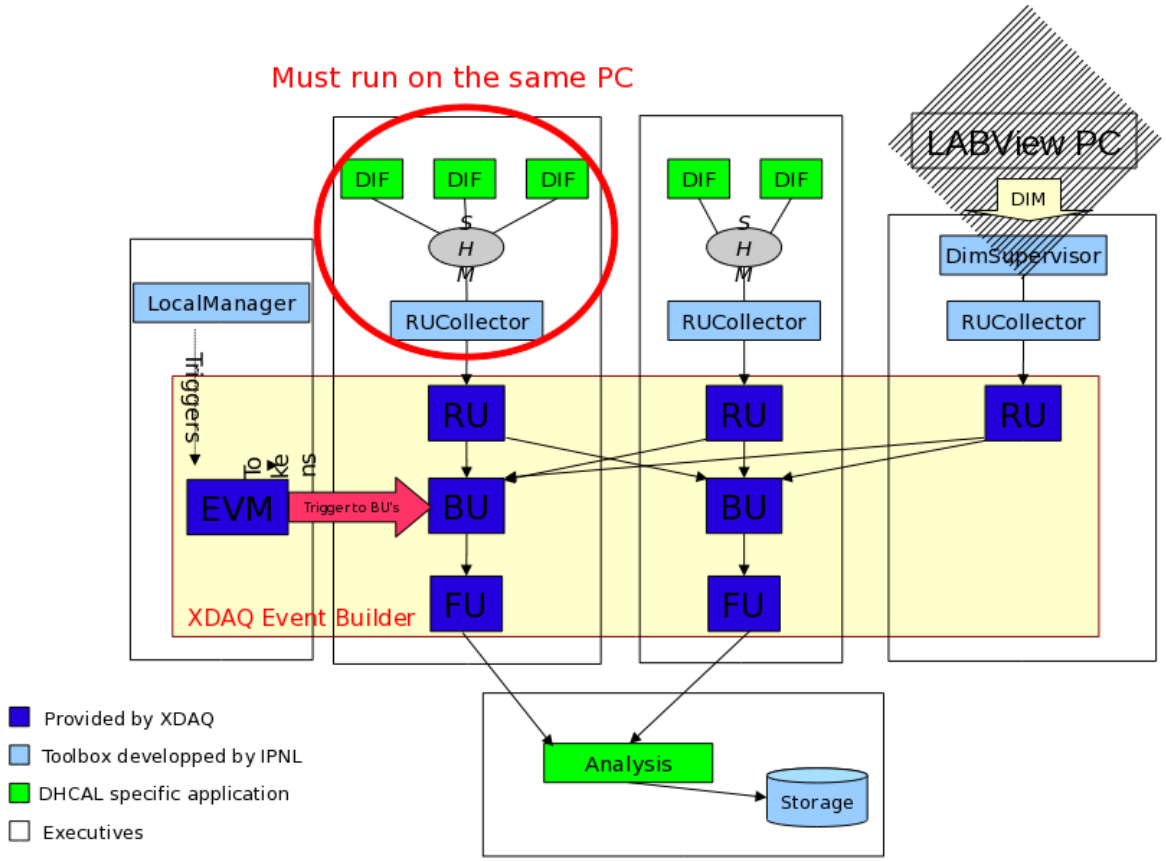
See http://cms-ru-builder.web.cern.ch/cms-ru-builder/RUBUILDER_G_V1_6_0.doc

Aysnchronous collection of data source corresponding to the same trigger.

- One trigger is seen
- Each R*eadout*U*nit*I*nput* collects its fragments and pushes it to the RU
- The T*rigger*A*ccepter* sends trigger data to the EV*ent*M*anager*
- The **EVM** sends an event Id to the **B***uilder***U***nit* that will request its first buffer to each **RU** and build the event
- The event is sent to the registered **F***ilter***U***nit* that can make data coherence checks, analysis and data storage



The DHCAL case



Including the analysis

To do

1. Data coherency and event building
2. LCIO storage
3. Monitoring

Including the analysis

To do

1. Data coherency and event building
2. LCIO storage
3. Monitoring

Implementation

- Separe FU (CMS) application to DHCAL ones: Event forwarded to an *LCIOAnalyzer* or *MarlinAnalyzer*
- Custom LCIO Event building and storage (DHCalAnalysis library)
- Monitoring using an online interface to MARLIN (MarlinAnalyzer)

Using share memories

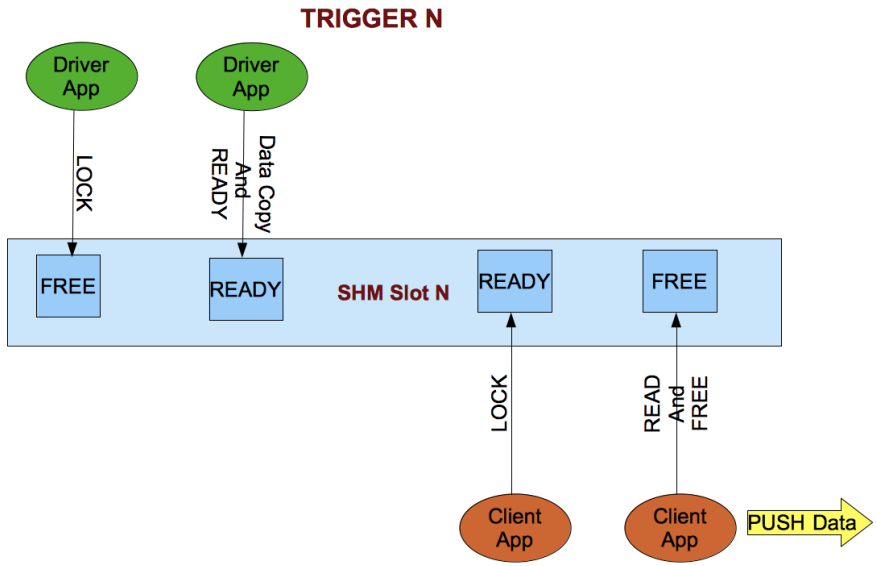
class RUShare

- Creation of a named Share memory (server) or attachment to it
- Mapped to 100 Slots of 512kBytes
- Slot status FREE,READY,LOCK

Using share memories

class RUShare

- Creation of a named Share memory (server) or attachment to it
- Mapped to 100 Slots of 512kBytes
- Slot status FREE,READY,LOCK
- Server (data source) looks for FREE slot, lock it and write data and mark it READY
- Client (consumer) look for READY slot , lock it, read data and mark it FREE



Using share memories

Servers

Each *DIFSupervisor* is a data source: it creates and fills one share memory, one trigger to one slot

Using share memories

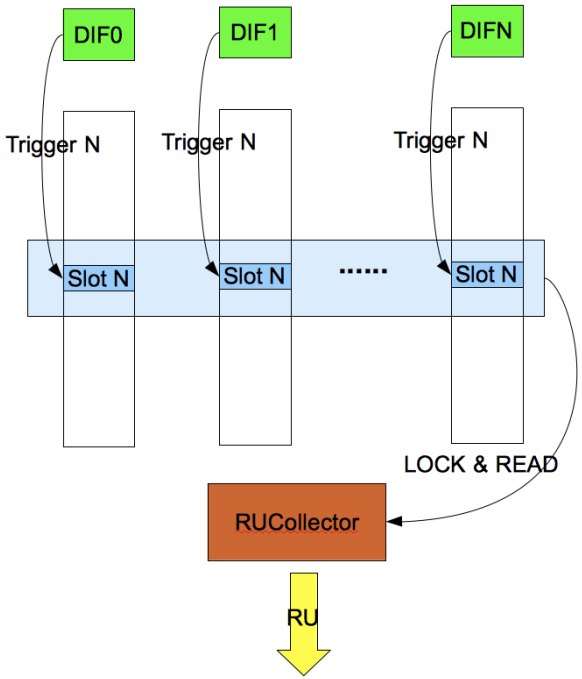
Servers

Each *DIFSupervisor* is a data source: it creates and fills one share memory, one trigger to one slot

Client

The *RUCollector* is the unique client of a set of DIFSupervisor

- it loops on the 100 slots and looks for READY slots on ALL share memories
- it copies data from the same slot(same trigger) and FREE the slots
- it pushes the collected data to the Event Builder



Several PC's

Constraint

- DIFSupervisor's should run on the PC where the DIF are connected
- one RUCollector per PC reading DIF

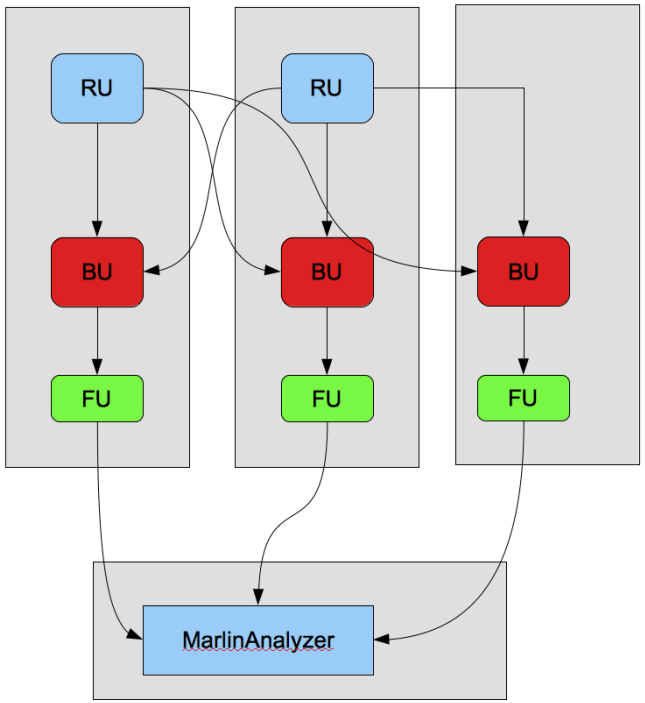
Several PC's

Constraint

- DIFSupervisor's should run on the PC where the DIF are connected
- one RUCollector per PC reading DIF

Distributed Event building and analysis

Any number of BU/FU can be added on the same or on other PCs
The whole configuration is described in an XML file and the reorganisation of the applications is easy (*to move analysis to a new PC for example*)



Data storage

Collections

Two collections are built and stored in LCIO format

- RU_XDAQ: a list of GenericObject (vector of Int) containing raw data from RU
- DHCALRawHits: a collection of RawCalorimeterHits containing all hits seen in hardRocs. Thresholds are encoded in the amplitude, position and time in CellIDs.

Slow control data are stored as parameters in a new RunHeader when read from the DIFs

Data storage

Collections

Two collections are built and stored in LCIO format

- RU_XDAQ: a list of GenericObject (vector of Int) containing raw data from RU
- DHCALRawHits: a collection of RawCalorimeterHits containing all hits seen in hardRocs. Thresholds are encoded in the amplitude, position and time in CellIDs.

Slow control data are stored as parameters in a new RunHeader when read from the DIFs

DHCALAnalysis

A standalone library handling:

- Creation of LCIO file
- handling of RU buffer
- Creation of RawCalorimeterHits from raw data
- Analysis framework

Data analysis

Standalone

Using DHCALAnalysis library

- Abstract Analyzers that can be registered (analog to Marlin, no DLL, Factory to be written)
- Can be run online

Data analysis

Standalone

Using DHCALAnalysis library

- Abstract Analyzers that can be registered (analog to Marlin, no DLL, Factory to be written)
- Can be run online

Marlin

- Port of Lyon online analysis
- Inclusion in a XDAQ application *MarlinAnalyzer* is done:
  - Peformances analog to the standalone version *LCIOAnalyzer*
  - Possibility to add full reconstruction online

Monitoring

Based on ROOT. Still preliminary.
Few tools to handle histograms and display them online.

DCHistogramHandler

Tool class handling a hashmap of TH1 and TH2 histograms with structured names:
/TOP/DIF21/HitMap ...

- Direct access on name to the histograms
- Query of histograms on regular expressions
- Canvas/ Images creation based on those queries
- Histograms storage follows the structure

Monitoring

Based on ROOT. Still preliminary. Few tools to handle histograms and display them online.

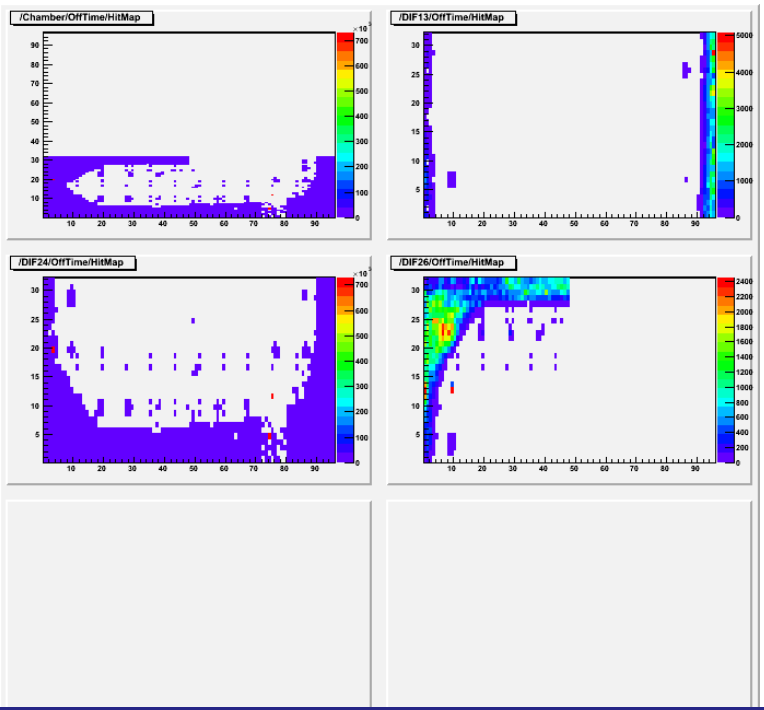
DCHistogramHandler

Tool class handling a hashmap of TH1 and TH2 histograms with structured names:
/TOP/DIF21/HitMap ...

- Direct access on name to the histograms
- Query of histograms on regular expressions
- Canvas/ Images creation based on those queries
- Histograms storage follows the structure

Toy web interface in LCIO/MarlinAnalyzer

- CGI handling of regular exp. or of the histogram structure
- Canvas creation and display inside web page



Spring 2010 Beam tests

May 2010 (T9 PS)

First use of the Event Builder in beam test

- Two $1m^2$ chambers , 6 DIFs
- One additional trigger board (TSC) to control the trigger
- First tests with the CCC

Spring 2010 Beam tests

May 2010 (T9 PS)

First use of the Event Builder in beam test

- Two $1m^2$ chambers , 6 DIFs
- One additional trigger board (TSC) to control the trigger
- First tests with the CCC

June 2010 (H2 SPS)

- $1/6m^2$ chamber (1 DIF, 24 HR2)
- B= 3 T field
- Power pulsing mode
- No trigger control anymore
  - back pressure from DIF busy signal
  - hardware veto for EVB overflow

Laboratory tests

Speed

- The main limitation comes from the USB readout of the hardrocs. Typical readout speed is 500 Hz
- More complex analysis can slow down heavily the performances and requires adjustements in the EVB structure and size.

Laboratory tests

Speed

- The main limitation comes from the USB readout of the hardrocs. Typical readout speed is 500 Hz
- More complex analysis can slow down heavily the performances and requires adjustements in the EVB structure and size.

Calibrations

The EVB structure allows an easy implementation of any calibration loop

- TA block the trigger(veto) and change parameters (Message to DIF)
- TA send N trigger with the new params and wait for data collection
- Pedestals and HR2 calibration with injection are already implemented

$1m^3$ prototype

New Hardware

Integration of DCC and LDA in the framework. It should be easy with the use of RUCollector mechanism

Performances

Need to define CPU and storage capabilities required

Monitoring

Main issue. The current online monitoring is not suited for large system. Recent MARLIN adaptation should allow new developpers to be involved.

Other developments needed

Configuration database

A first prototype of a configuration database (MySQL) is being tested. It handles a versioned set of all DIF and HardROC parameters of a given setup. First usgage during next week H4 beam test. Compulsory for $1m^3$ prototype.

Other developments needed

Configuration database

A first prototype of a configuration database (MySQL) is being tested. It handles a versioned set of all DIF and HardROC parameters of a given setup. First usgage during next week H4 beam test. Compulsory for $1m^3$ prototype.

Condition database

Both data taking conditions and errors need to be logged in a Condition database accessible offline. No implementation yet.

Other developments needed

Configuration database

A first prototype of a configuration database (MySQL) is being tested. It handles a versioned set of all DIF and HardROC parameters of a given setup. First usgage during next week H4 beam test. Compulsory for $1m^3$ prototype.

Condition database

Both data taking conditions and errors need to be logged in a Condition database accessible offline. No implementation yet.

Process configuration

Currently, the process creation is manual and the application configuration is controlled by the *LocalManager* via SOAP messages. The system is well suited for standalone setup (1 Partition). Larger system will require to use a separate framework (CMS RCMS?)

Other developments needed

Configuration database

A first prototype of a configuration database (MySQL) is being tested. It handles a versioned set of all DIF and HardROC parameters of a given setup. First usgage during next week H4 beam test. Compulsory for $1m^3$ prototype.

Condition database

Both data taking conditions and errors need to be logged in a Condition database accessible offline. No implementation yet.

Process configuration

Currently, the process creation is manual and the application configuration is controlled by the *LocalManager* via SOAP messages. The system is well suited for standalone setup (1 Partition). Larger system will require to use a separate framework (CMS RCMS?)

Slow Control

All slow control application (HV/LV control, environmental probes) are currently standalone. A common framework and an interface to configuration and condition database is needed.

Summary

Status

SDHCAL acquisition software is based on XDAQ framework for data collection and MARLIN for online analysis. The event builder of CMS is also used for coherent data collection.
The system has been tested both in laboratory test and in 2 beam tests this year.

Futur

Several developments are on-going to adapt this daq to the 1 m^3 scale: LDA & DCC integration, Configuration DB and monitoring

Try it

XDAQ may look complex but it is not a so heavy framework.
Software is available on SVN. We are ready to help new group to adapt their DAQ.