Interaction between the Topology Manager and the centralized AGATA Detector Database

Draft 1- 18/05/2012

Goal of the document:
To determine the way the Topology Manager, TM, interacts with the centralized AGATA Detector Data Base, ADDB, based on Oracle and hosted @ the CCIN2P3.

Important notes for this document:

1. The ADDB is hosted at CCIN2P3, with a connection that is possibly not always active. Thus, the TM at running time should not require it.
2. Objects in the ADDB have a name, type and version number. They are uniquely identified by an alphanumeric barcode. An object is set as valid / faulty to know if it can be used
3. There are two types of relations between objects stored in the ADDB. An object can be composed of other objects or connected through cables.
4. Information concerning objects (measures, characteristics, etc) done in ADDB through actions. Amongst the default fields:
    a. A quality flag (integer) with signification of all values stored in the ADDB. In case of negative number, the object is set automatically in the ADDB as faulty.
    b. A status [Reference, valid, notvalid, running]. Reference means current 'working' values.
5. For consistency of the ADDB, it is highly recommended to use the BigBrowser GUI functionalities or XML files (through BigBrowser or a command line program, CmdLineBB) to update the ADDB. In reading mode, direct SQL request could be done.

The following picture gives an overview of interactions between different actors using the ADDB.

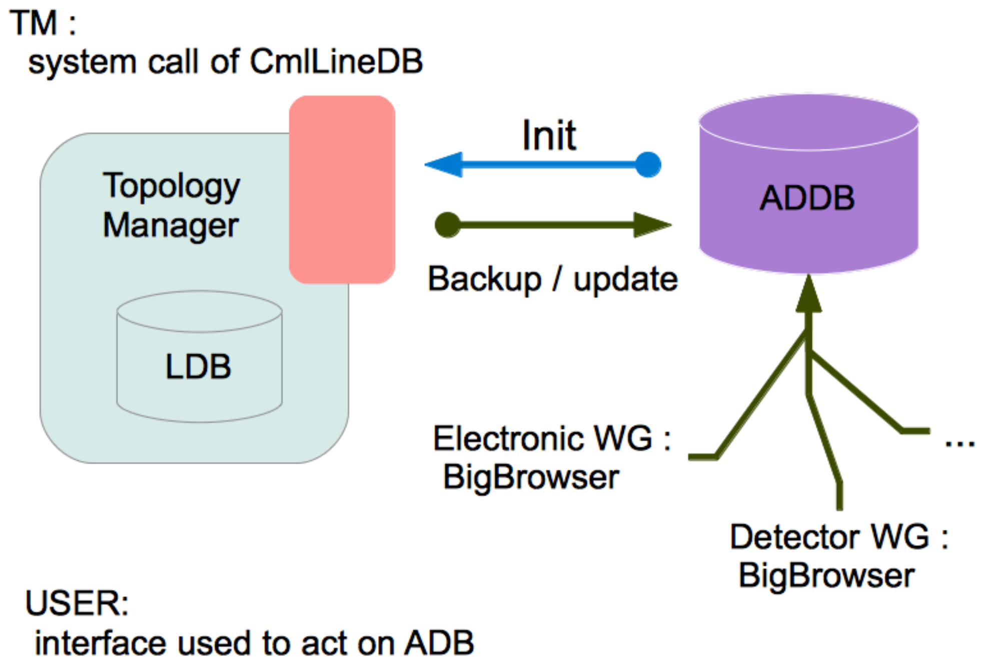

Figure 1: Interaction between TM, ADDB and other users

Here are the different tables as required by the TM and the way the information is organized in the ADDB world.
The fields are of two types static or dynamic:

- Static means the TM expects this information from the ADDB
- Dynamic means the TM is likely to change the fields but they can be modified also directly in the ADDB.

The following gives how the information exists in the TM and ADDB world and how it can be exchanged. Some fields are already in the ADDB, additional fields (not already existing in the ADDB) are stored in an action which name starts with TM_VERS_OBJECTNAME_ or if needed TM_VERS_OBJECTNAME_OBJECTTYPE.
At initialization, it would require TM to perform two consecutive requests. The first one to get all the elements and the way they are composed/connected. Then the content of the TM_ actions for all individual items should be requested.

TM world	Interface	ADDB world

Detector (AGATA … to be defined in ADDB)
Parameters

- Last_Tag: *varchar(25)* : The last tag of Topology Manager development version
- Name: *varchar(25)*
- Status : *on, off, damaged.*

Relations

- The Agata detector has 60 triple-clusters.

Name	Bar code = AGATA	OBJECT_ASSEMBLY
Status	ON = 0, OFF = 1, damaged = -1 ➥ same for all except if explicit	TM_1_AGATA_VAL
Last_Tag		TM_1_AGATA_LASTTAG
Relations	Composed of ATC, ADC	OBJECT_ASSEMBLY

Triple-cluster (extension to Double-cluster is obvious)
Parameters

- Position: *tinyint* : Position on the Agata array.
- Name: *varchar*(25)
- Satus: *on, off, damaged.*

Relations

- One triple-cluster has 3 crystals.

Name	Ex: ATC01	OBJECT_ASSEMBLY
Status		TM_1_TCLUSTER_VAL
Position		TM_1_TCLUSTER_POSITION
Relations	Composed of 3 DET_CAP_	OBJECT_ASSEMBLY

Crystal:
Parameters

- Color: *varchar(25)* : Describes the crystal type: red, green, blue. (The colors will change to A, B, C)
- Name: *varchar(25).*
- Status: *on, off, damaged.*

Relations

- One crystal has 36 crystal slices. It sends the signals from the detector segments and one core signals to 7 preamplifier modules.

Name Status Color Relations	Ex: DET_CAP_BLU_C001 With MDR cables, not yet in ADDB	OBJECT_ASSEMBLY TM_1_CAPSULE_VAL OBJECT_ASSEMBLY_TYPE LINK ? (not yet completed in ADDB)

Digitizer:
Parameters

- Mac_address: *varchar(10)* : serial hardware number.
- Name : *varchar(25)*
- Port : *integer*.
- Fpga_Name: *varchar(25)* : Name or Id of the Spartan Fpga which allows the access to the digitizer.
- Status : *on, off, damaged*.

Relations

- One digitizer home 2 main boards.

Name Status Mac_address Port Fpga_Name Relations	Ex: DPS_DIG_009 Composed of two DIGITIZERMOD	OBJECT_ASSEMBLY TM_1_DIGITIZER _VAL TM_1_DIGITIZER _MACADDRES TM_1_DIGITIZER _PORT TM_1_DIGITIZER _FPGANAME See OBJECT_ASSEMBLY

Main Board:
Parameters

- Name : *varchar(25)*
- Status : *on, off, damaged*.
- Mac Address: *varchar(10)* : Serial number of adc board.
- Kind: *segment, core* : Segment or core.
- Fpga_Name : *varchar(25)* : Name or id of Virtex2 pro Fpga.
- Firmware_version : *varchar(25)*
- Board_Version : *varchar(25)*
- Modification_state: *varchar(25)* :Hardware modification level. It is incremented when a change is made to the hardware.

Relations

- There are two types of main boards into a digitizer: segment and core. The segment main board homes 4 segment ADCs; the core main board homes 2 segment ADCs and one core ADC.

Name Kind Status Mac_address Fpga_Name Firmware Board_Version Modification_state	Ex: DPS_DIG_COR_001	OBJECT_ASSEMBLY OBJECT_ASSEMBLY_TYPE TM_1_DIGITIZERMOD _VAL TM_1_DIGITIZERMOD _ TM_1_DIGITIZERMOD _ TM_1_DIGITIZERMOD _ TM_1_DIGITIZERMOD _ TM_1_DIGITIZERMOD _ TM_1_DIGITIZERMOD _*ADC1*

As shown in picture 2 (hatching), in the ADDB there is no ADC defined. The proposition is then to have the information relative to ADC in the same action that the digitizer module it belongs to (blob or separated field with names to be defined) or to define a separate action (Ex: TMADC). As well Preampli are defined in ADDB, associated to a cluster but the information is not at all actually in the ADDB.

ADC:
Parameters

- Name : *varchar(25)*
- Status : *on, off, damaged.*
- Mac Address: *varchar(10)* : Serial number of adc board.
- Fpga_Name : *varchar(25)* : Name or Id of Virtex2 pro Fpga.
- Fpga_spare_Name : *varchar(25)*
- Kind: Segment or core.
- Board_Version : *varchar(25)*
- Modification_state: *varchar(25)* : Hardware modification level. It is incremented when a change is made to the hardware.
- Status : *on, off, damaged.*

Relations

- One ADC module receive the analog signals from 6 segment channels or 2 core channels from the preamp modules.
- One module output the digital signals of 6 segment channels or 2 core channels to one mezzanine.

Preampli:
Parameters

- Name : *varchar(25)*
- Value : *blob*
- Status : *on, off, damaged.*

Relations

- **One preamp module receive the signals from 6 segment channels or 2 core channels.**
- One preamp output the digital signals of 6 segment channels or 2 core channels to one ADC.

Fiber channel
Parameters

- Name: *varchar(25)* : Name or Number on the coaxial cable connecting the ADC output to one mezzanine. The number of coaxial cable connection is the same as detector segment level(?).
- Status : *On, off, damaged.*

Relations

- One coaxial cable forward the signals of 6 segments channels or 2 core channels coming from one ADC to one mezzanine on the carrier.

In the ADDB, the electronic cards in charge of processing a single crystal are associated into a virtual object called Pre Processing Capsule (see picture 2).

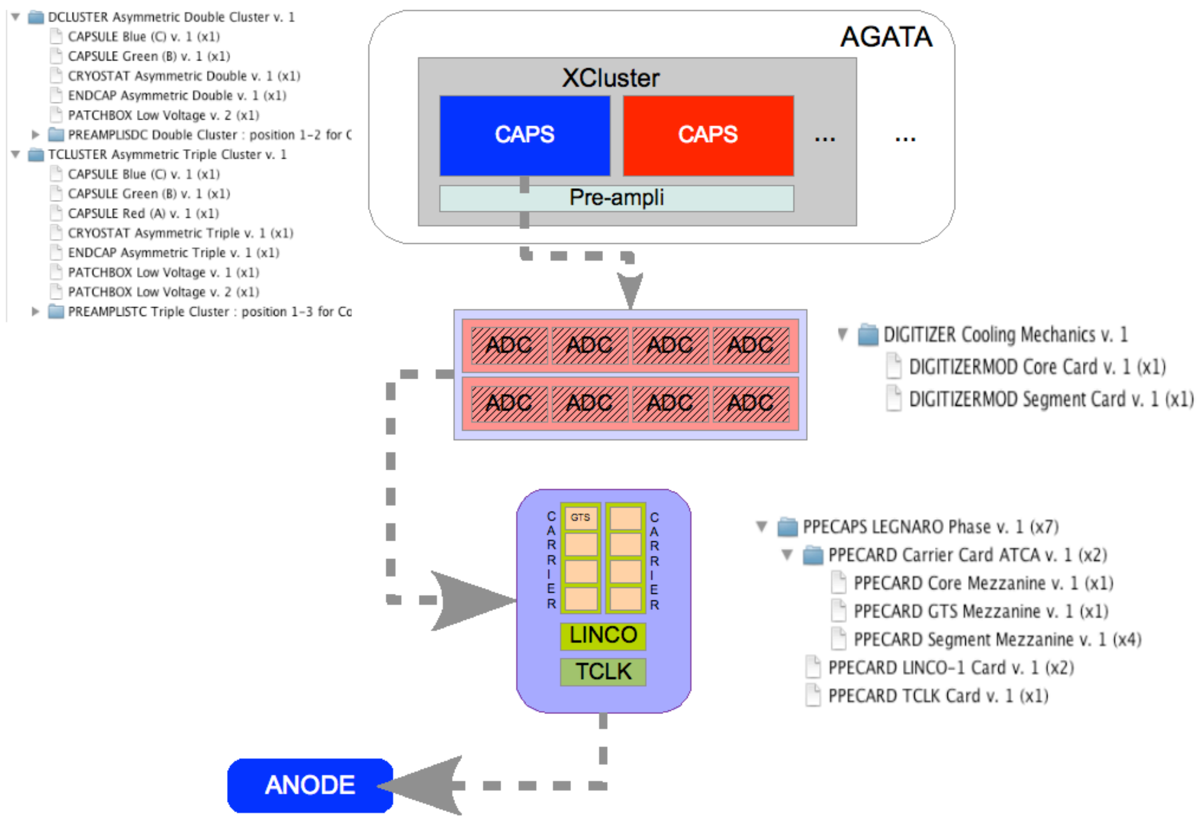
DCLUSTER Asymmetric Double Cluster v. 1
CAPSULE Blue (C) v. 1 (x1)
CAPSULE Green (B) v. 1 (x1)
CRYOSTAT Asymmetric Double v. 1 (x1)
ENDCAP Asymmetric Double v. 1 (x1)
PATCHBOX Low Voltage v. 2 (x1)
PREAMPLISDC Double Cluster : position 1-2 for C
TCLUSTER Asymmetric Triple Cluster v. 1
CAPSULE Blue (C) v. 1 (x1)
CAPSULE Green (B) v. 1 (x1)
CAPSULE Red (A) v. 1 (x1)
CRYOSTAT Asymmetric Triple v. 1 (x1)
ENDCAP Asymmetric Triple v. 1 (x1)
PATCHBOX Low Voltage v. 1 (x1)
PATCHBOX Low Voltage v. 2 (x1)
PREAMPLISTC Triple Cluster : position 1-3 for Cc
AGATA
XCluster
CAPS
CAPS
...
...
Pre-ampli
ADC
DIGITIZER Cooling Mechanics v. 1
DIGITIZERMOD Core Card v. 1 (x1)
DIGITIZERMOD Segment Card v. 1 (x1)
CARRIER
GTS
CARRIER
LINCO
TCLK
PPECAPS LEGNARO Phase v. 1 (x7)
PPECARD Carrier Card ATCA v. 1 (x2)
PPECARD Core Mezzanine v. 1 (x1)
PPECARD GTS Mezzanine v. 1 (x1)
PPECARD Segment Mezzanine v. 1 (x4)
PPECARD LINCO-1 Card v. 1 (x2)
PPECARD TCLK Card v. 1 (x1)
ANODE

Figure 2 Compositions of objects in the ADDB

From the composition of the PreProcessing Caps, carrier, GTS, mezzanine could be obtained (Question for Cecile: how to get father knowing son)

Carrier:

Parameters

- Name: *varchar(25)*
- Status : *On, off, damaged.*
- mac_address: *varchar(10)*
- Port: *integer*
- Kind : *Master or slave* . **QUESTION: master is the one with GTS ??**
- Slot : *blob* : slot on the crate
- Board_version : *varchar(25)*
- Firmware version: *varchar(25)*
- Emb_soft_version: *varchar(25)* : Embedded system version

Relations

- The master carrier has two segment mezzanines, one core mezzanine and one GTS; the slave carrier has 4 segment mezzanines.

Name Status Mac_address	Ex: DPS_PP_ACAR_018	OBJECT_ASSEMBLY TM_PPECARD_ TM_PPECARD_

Port		TM_PPECARD_
Kind	! kind different from type in ADDB[1] ?	*TM_PPECARD_*
Slot	! same comment ?	*TM_PPECARD_*
Board_version		TM_PPECARD_
Firmware_version		TM_PPECARD_
Emb_soft_version		TM_PPECARD_

Note: If the fields regarding carrier and mezzanine are similar, only one action to be defined, otherwise an action for each type of PPECARD.

Mezzanine:
Parameters
- Name: *varchar(25)*
- Status : *On, off, damaged.*
- Mac_address: *varchar(10)* : Unique identification.
- Slot: *blob* : Position on the carrier board: 1,2,3 or 4. **QUESTION: Not always the same ?? internal TM ?**
- Kind: *Segment or core.*
- Board_version: *varchar(25)*
- Firmware_version: *varchar(25)*
- Emb_soft_version: *varchar(25)* : Embedded software version

Relations
- One segment mezzanine receive 6 segment channels and one core mezzanine receive 2 core channels from one ADC.

Name	Ex: DPS_PP_SEGM_064	OBJECT_ASSEMBLY
Status		TM_PPECARD_
Mac_address		TM_PPECARD_
Port		TM_PPECARD_
Kind	! kind different from type in ADDB[1] ?	*TM_PPECARD_*
Slot	! same comment ?	*TM_PPECARD_*
Board_version		TM_PPECARD_
Firmware_version		TM_PPECARD_
Emb_soft_version		TM_PPECARD_

GTS:
Parameters
- Name : *varchar(25)*
- Status : *On, off, damaged.*
- Mac_address: *varchar(10)* : equal to the lowest byte of the IP number.
- Kind: *Leaf, root or fanin-fanout* depending of the position in the GTS Tree
- Port: *integer* : on the GTS Board for the udp/ip connection
- Slot: *blob* : Position in the carrier Board: 1,2,3 or 4.
- Board_version: *varchar(25)*
- Firmware_version: *varchar(25)*
- Host_port: integer : *varchar(25)* : the port used on the PC host.
- Emb_soft_version: *varchar(25)* : Embedded software version
- Host_soft_version: *varchar(25)* : Host software version
- *The versions of firmware and embedded software that have been used for the test(??)*

Relations
- Each TreeGTS can connect to at most 4 other GTS.

[1] To be discussed, should this field be in action or retrieve from composition?

Name	Ex: DPS_PP_GTSM_008	OBJECT_ASSEMBLY
Status		TM_PPECARD_GTS
Mac_address		TM_PPECARD_
Port		TM_PPECARD_
Kind	! kind different from type in ADDB[1] ?	*TM_PPECARD_*
Slot	! same comment ?	*TM_PPECARD_*
Board_version		TM_PPECARD_
Firmware_version		TM_PPECARD_
Host_port		TM_PPECARD_
Emb_soft_version		TM_PPECARD_
Host_soft_version		TM_PPECARD_

Crate:

Parameters

- Name : *varchar(25)*
- Status : *On, off, damaged.*
- Mac_address: *varchar(10)* : Unique identification.

Relations

- One crate home 12 carriers.

Name	Ex: DPS_PP_ATCA	OBJECT_ASSEMBLY
Status		TM_ PPCRATEPS_
Mac_address		TM_ PPCRATEPS_

Narval node:

Parameters

- Name : *varchar(25)*
- Status : *On, off, damaged.*
- Port : *integer*
- Mac_address: *varchar(10)* : Unique identification.

Relations

- One Narval node receives the data from 2 carriers.

Name	Ex: DAQ_NODE_	OBJECT_ASSEMBLY
Status		TM_COMPUTINGNODE_
Port		TM_COMPUTINGNODE_
Mac_address		TM_COMPUTINGNODE_

Note : A cable should be added between PreProcessing Capsule and the Node.

The following items don't have equivalent in the ADDB. TO BE DISCUSSED in a second step.

Mac Ip:

Parameters

- Mac_address: *bigint :* Mac_address of the pcs
- Ip_or_name: *blob :* Ip or name of the pcs.

Server:

Parameters

- Name: *varchar(25)*.
- Port: *integer*.
- kind: *Device type* of server: Carrier, etc.
- address : *varchar(128*) : server address.
- path: *varchar(25)* :Complete URL address of server.

- internal_ref: *varchar(25)* : internal identification of server.

Relations:
Parameters

- Id_origin: *integer* : (This id is only for the database) : Identification in the Topology Manager database.
- Id_destination: *integer* : (This id is only for the database) :Identification in the Topology Manager database.
- Type_origin: *Device type* of “Id_origin” device: Digitizer, triple_cluster, adc, etc.
- Type_destination: *Device type* of “Id_destination” device: Digitizer, triple_cluster, adc, etc.

The following table shows the different possibilities of relations between devices in the TM database:

Detector	**Triple-cluster**	1 – 60
Detector	**Crate**	1-??
Triple-cluster	**Crystal**	1 – ??
Crystal	**Preampli**	7
Crystal	**Digitizer**	1
Crystal	**Carrier**	2
Crystal	**Narval Node**	2
Crystal	**Gts**	1
Digitizer	**Mainboard**	2
Mainboard	**ADC**	7
Crate	Carrier	**12**
Carrier	**Mezzanine**	**7**
Mezzanine	Fiber channel	**6**
Fiber channel	**ADC**	7
Preampli	**ADC**	7
Narval Node	Carrier	2

Figure 3: TM Database relation