The Rack Wizard, a graphical database interface for electronics configuration

F. Glege

CERN, 1211 Geneva 23, Switzerland Frank.Glege@cern.ch

Abstract

A substantial amount of electronics is needed in CMS to provide the required running environment, digitize the detector information and forward it to the data acquisition system. The information concerning all aspects of this electronics needs to be kept in a data base. The layout of the electronics must fulfil certain constraints. Tracking the compliance of the layout with those constraints in an evolving setup is time consuming and non trivial. A tool has been developed, called "Rack Wizard", which offers a graphical configuration of the electronics layout in a drag and drop style. That provides entry into an Oracle database. It automatically applies certain constraints and the visualization allows, for instance, the identification of hot spots.

DATABASES IN CMS

The CMS experiment at CERN will use databases to store nearly all data describing the detector, being used by the detector or being produced by the detector. The data is classified in five groups which differ in there requirements on database logistics, performance and data volume. The following paragraphs will shortly describe these groups called databases in the following. Since the interface, which is the subject of this document acts on the equipment management database, the latter will be described more in detail.

A. Constructions databases

Each sub detector has his construction database. It contains data concerning the detector construction. Information on the fabrication and testing of detector components is stored here as well as logistical information. These databases will not be used by the experiment online system. However, parts of the stored data will be copied to the databases described in the following. After the detector installation is finished, the constructions databases will mainly be used for error analysis by the sub detectors.

B. Configuration database

The configuration database will store all data necessary to put the detector in any running condition. It mainly consists of highly detector specific parameters for electronics configuration. The amount of data to be extracted per configuration varies from a few kilobytes to a gigabyte depending on the detector. The performance requirements are therefore not negligible since the configuration process should be as short as possible.

C. Conditions database

The conditions database will hold all data describing the running conditions of the detector. The main part of the conditions data will be necessary for physics event reconstruction. Other data will be used for error tracking.

D. Equipment management database

The equipment management database will hold all data describing the physical layout of the detector system. This includes the sub detector parts as well as the off detector electronics. A history of all item locations will be kept to allow for asset tracking. It is assumed that the contained data will only be used in the initial setup of the detector electronics after switching on the power. Therefore performance is not an issue for this database. The current implementation is done in ORACLE[3] since it is the supported database system at CERN and seems to fulfil all requirements.

1) Content

Actual examples of items described in this database are racks, crates, boards, channels, cables, detector parts, etc. The underlying data model is based on the idea of slots and instances to fill them. The slots describe the geometry of locations to be occupied by the item classes as well as there relations. The instances describe there specific parameters as well as there actual size and location in terms of slots of there parents. The smallest slot is currently a cable connector. The cables provide the connection between the otherwise distinct hierarchical trees of detector parts on one side and the detector electronics on the other side.

2) Procedures

The database write access is restricted by an authentication mechanism. Each item belongs to a user group and only members of the appropriate group are allowed to change the properties of an item.

The database keeps a history of all item locations. This allows calculating an irradiation dose for each item depending on its position and the time it was at this position. An electronic logbook contained in the database offers to comment each location change of the items. This produces a functioning/repair history for all items which allows to find quality problems.

3) Item properties

The geometry and location of each item is contained in the database. This allows for visualization as well as for checking of inconsistent placement, e.g. overlapping. Signal path lengths can be calculated as well using this information. For electronics components the power requirements (supplied power and dissipated power) are stored. This allows for calculating the overall power and cooling needs and helps finding possible hot spots.

THE DATABASE INTERFACE

Creating and maintaining the electronics configuration of an experiment of the size of CMS represents a large amount of work. After entering the hundred thousands of parameters, they have to be checked for consistency and maintained over the lifetime of the experiment. Although it might have been possible to store this amount of data in a simple persistent storage like an excel file, the checking and the maintenance would have been virtually impossible. The need for a database as storage medium is therefore obvious. The database contains integrity checks and it provides sophisticated mechanisms for data maintenance. To prevent the users from having to know about these mechanisms, a user interface "Rack Wizard" has been developed which translates the use of these mechanisms into intuitive, visual dragging and dropping.

A. The concept

One requirement for the implementation has been to separate the visualization from the content to make the tool as universal as possible. The rack Wizard therefore provides a kind of container, which offers the infrastructure for the data handling. The hierarchical organization and navigation is also provided. Plug in like components handle specific configurations of item classes and provide the information about the direct relation of instances to the hierarchy mechanism. The separation of structure and items allowed for an object oriented design and to profit from the mechanisms of an object oriented programming language.

B. The implementation

JAVA has been chosen as programming language for the Rack Wizard since it is independent of the operating system. For many users the tool is therefore usable without any additional configuration which increases the acceptance. The communication protocol between the interface and the database is XML (Extensible markup language)[1] over http (hypertext transfer protocol)[2]. Since the http is used in the same way as for usual web access, it is possible to use the Rack Wizard even from behind firewalls. http is fully supported by JAVA and the ORACLE application server (the ORACLE web server). XML is available in the application server and therefore only a simple xml handler had to be developed in JAVA, to keep the application footprint small. The application is separated in a loader that only loads the main application from a web page and starts it and the main application. Therefore any change to the main application does not require any more action by the user than restarting the Rack Wizard to become active which eases the maintenance. The reflection mechanisms in JAVA allow furthermore assembling the main application from classes distributed in different places. This allows for a personalization of the tool by only providing a private class in a separate location. An adaptation of the Rack Wizard to other experiments has been very simple that way.

C. The main container

The main container provides the basic, item content independent infrastructure (see Figure 1). The frame title shows the logged in user. A selection mechanism for different item hierarchies and their visualizations including the navigation mechanism is shown on the left. The right part is dedicated to the specific item visualization, a rack in the example of Figure 1. Two other examples are shown in Figure 2 and Figure 3.

1) The database access

The infrastructure provided by the main container comprises a common database access which decouples the components from the database implementation. The Rack Wizard components request data sets from the database interface, which then provides those as a graphical table. This table provides easy access to the contained data and can be visualized immediately. Additions to the data structure of an item class do not require therefore any change of the application to become visible.

The user can change the parameters in the table through the table visualization. To update the database, the table can then be sent back to the database interface, which will apply the changes.

2) The template container

Each item class can create a template container. The container will automatically load the item templates from the database and provide a visualization of those. By dragging templates from the container and dropping them into a parent item, an instance of the template is created. The instances can either be independent or references of to the template. In the latter case the instances themselves can not be configured separately but the configuration of the template is applied to all its references.

3) Labels

Each item in CMS will have a unique identifier called label. Although it's up to the specific item handlers to create this label, since the rule for the label creation depends on the item class, the infrastructure for reading and printing labels will be provided by the main container.

👙 Rack compositor - User: Geoffrey	Hall					
File Edit Tools Help						
🚹 🔛 🖲 Move mode 🛛 Copy mo	ode	Memory usage:	69%			
Racks 🗸 📥		CMS - S1 - S1C03 -				
Ϙ -⊡ ¹ CMS	56-	Rack properties Crate properties				
🗣 🛄 Building 32	54-	Rack	S1C03	Crate		
P □ S1	52- Turbine 4U	Detector	Tracker			
S1A01	50- Heat exchanger	Usage	FED TOB	Power supply		
© STAU2	48-	Responsible	GHALL			
●	46-	Comment		PC		
● 🗂 \$1A05	44-	Last change	9/19/2003 10:24:54			
🗣 🚍 S1A06	42-	Power supplied [kw]	0	Datab papal		
● 🗂 S1A07	40-	Power dissipation [kw]	4.68	Pater		
🗢 🗂 S1A08	38- Ean tray 20	Power dissipation to air [kw]	0.0	Heat exchanger		
• 🗖 \$1A09	36- Heat exchanger			Turbing 411		
• S1A10	34-			Turbine 40		
S1A11	32-			Fan trav 2U		
	30-			Air deflector 20		
● 1 S1A14	28-			All defiector 20		
	26- Datch papel					
🗣 🗂 S1B02	24- Constrait 20					
👁 🗂 S1B03	22- Heat exchanger					
🗢 🗖 S1B04	20-	Comment history:				
🗢 🗂 S1B05	18-	-> 19.09.2003 10:24 (GHALL) Added a heat exchanger to the top of the rack, to cool the air				
S1B06	16-	flow going down the rack side.				
●	14.	-> 13.06.2003 18:44 (GHALL) full -> 13.06.2003 18:44 (GHALL) Put in patch panels to make room for optical cabling. Remo				
	12- Detabased	ved heat-exchanger from above top FED crat	te.			
	10-					
• • • • • • • • • • • • • • • • • • •	8- Heat exchanger					
• 🗂 S1B12	6- Air deflector 2U					
🗣 🗂 S1B13	4					
💁 🗂 S1B14	2.					
• 🖬 S1C01 📃						

Figure 1: Rack Wizard main container



Figure 2: Visualization of a detector end cap

A	В	с	D	E	F	G	н
S1AD1	S1801	S1C01	S1D01	S1E01	S1F01	S1G01	S1H01
Tracker	DAQ	DAQ	Tracker	Opt. Cpl.	Preshower	DT	Picel
Power supply	DAQ	DAQ	FEC	Opt. Cpl.	FEC	RO/SC	Controls
GHALL	RACZ	RACZ	GHALL	VARELA	BARNEYD	WILLMOT	HORISBER
S1AD2	S1802	S1CO2	S1D02	S1E02	S1F02	S1002	S1H02
Tracker	Tracker	Tracker	Traoker	TTC	DT	Pixel	Tracker
Power supply	FED TIB TID	FED TOB	FEC	TTC	track finder	FEC	Power supply
GHALL	GHALL	GHALL	GHALL	TROSKAJK	ERCUR	HORISBER	GHALL
S1A03	S1803	S1C03	S1D03	S1E03	S1F03	S1003	S1H03
Tracker	Tracker	Tracker	RPC	TTC	DT	Pixel	Tracker
Power supply	FED TIB TID	FED TOB	Trigger	TTC	track finder	FED	Power supply
GHALL	GHALL	OHALL	RANIERI	TROSKAJK	ERCUR	HORISBER	GHALL
S1A04	S1804	S1C04	S1D04	S1E04	S1FD4	S1004	S1H04
Tracker	Tracker	Tracker	RPC	Global	DT	Pixel	Tracker
Power supply	FED TIB TID	FED TOB	Trigger	Trigger	track finder	FED	Power supply
GHALL	OHALL	OHALL	RANIERI	TAUROK	ERCUR	HORISDER	OHALL
S1AD5	S1805	S1C05	S1D05	S1E05	S1F05	S1605	S1H05
Tracker	DAQ	DAQ	RPC	Calorimeter	CSC	DAQ	Tracker
Power supply	DAQ	DAQ	Trigger	Global Trigger	track finder	DAQ	Power supply
GHALL	RACZ	RACZ	RANIERI	HEATH	TYLING	RACZ	GHALL
S1ADB	S1800	S1C00	S1D05	S1E00	S1F06	S1606	S1H08
Tracker	Tracker	DAD	RPC	TTS	CSC	CSC	Tracker
Power supply	FED PCs	DAQ	Trigger	TTS	track finder	FED	Power supply
GHALL	GHALL	RACZ	RANIERI	RACZ	TYLING	TYLING	GHALL
S1AD7	S1807	S1C07	S1D07	S1E07	S1F07	S1007	S 1H07
Tracker	DAQ	DAQ	RPC	TTS	DAQ	CSC	Tracker
Power supply	DAQ	DAQ	Trigger	TTS	DAQ	FED	Power supply
GHALL	RACZ	RACZ	RANIERI	RACZ	RACZ	TYLING	GHALL
S1AD8 GLEGE	S1808 Tracker FED TEC- GHALL	S1C08 Tracker FED TEC GHALL	S1D08 RPC Trigger RANIERI	S1E08 LHC BPTX WSMITH	S1F08 RPC endoap HV RANIERI	S1008 CSC FED TYLING	S1H08 Tracker Power supply GHALL
S1AD9 OLEGE	S1809 Tracker FED TEC- OHALL	S1C09 Tracker FED TEC OHALL	S1D09 RPC Trigger RANIERI	S1E09 LHC LHC WSMITH	S1F09 RPC endoap HV RANIERI	S1009 CSC FED TYLING	S1H09 Tracker Power supply OHALL
S1A10 CSC HV TYLINO	S1810 Tracker FED TEC- GHALL	S1C10 Tracker FED TEC GHALL	S1D10 RPC Trigger RANIERI	S1E10 WSMITH	S1F10 RPC endcap- HV RANIERI	S1G10 DT HV WILLMOT	S1H10 Tracker Power supply GHALL
S1A11 CSC HV TYLING	S1B11 DAQ DAQ RACZ	S1C11 Tracker Controls GHALL	S1D11 WSMITH	S1E11 RPC barrel HV RANIERI	S1F11 RPC endcap- HV RANIERI	S1G11 DT HV WILLMOT	S1H11 Tracker Power supply GHALL
S1A12	S1B12	S1C12	S1D12	S1E12	S1F12	S1012	S1H12
CSC	Preshower	Tracker	ME1/1	RPC barrel	RPC barrel	DT	Tracker
HV	FED	Controls	HV	HV	HV	HV	Power supply
TYLING	BARNEYD	GHALL	WSMITH	RANIERI	RANIERI	WILLMDT	GHALL
S1A13	S1813	S1C13	S1D13	S1E13	S1F13	S1013	S1H13
CSC	Preshower	Tracker	ME1/1	RPC barrel	RPC barrel	DT	Tracker
HV	FED	Controls	HV	HV	HV	HV	Power supply
TYLING	BARNEYD	OHALL	WSMITH	RANIERI	RANIERI	WILLMDT	GHALL
S1A14	S1814	S1C14	S1D14	S1E14	S1F14	S1014	S1H14
CSC	DSS	Tracker	DSS	RPC barrel	DSS	DT	DSS
HV	DSS	Controls	DSS	HV	DSS	HV	DSS
TYLING	CSCHAFER	OHALL	CSCHAFER	RANIERI	CSCHAFER	WILLMOT	CSCHAFER
-passageway-	разза дежау	passageway	plassa geway	passageway	plassaig енкау	passageway	passageway

Figure 3: Visualization of an experiment zone

Different label formats can be defined per item class. From the main container it's then possible to choose a list of instances of a certain item class and labels of the items in the list will be printed in a format to be selected by the user. A bar code reader facility will allow visualizing the component described in the label encoded in the bar code.

D. Item specific components

To illustrate the current implementation and its features, two items of the electronics hierarchy and their specific item visualizations shall be discussed. The electronics hierarchy looks as follows:

Zone

- Rack
- Crate
- Board
- Channel
- Cable

1) Zone

As shown in Figure 3, the visualization of an experiment zone consists of a synoptic representation of the rack positions in the zone. The rack representations contain information about the identifier of the rack, the detector it belongs to, its usage and the responsible person. The rack color depends on the dissipated power of the rack. This allows for easy detection of hot spots. A whole rack configuration can be copied to another rack by using drag and drop.

2) Rack

The rack visualization consists of a synoptic view of a rack with its crates. The crate color depends on its heat

dissipation. The template container on the right side of Figure 1 can contain generic templates as well as sub system specific ones. To insert a crate into the rack it is sufficient to drag and drop a template into an empty slot in the rack. Most of the rack parameters are shown as well as the crate parameters. The user is asked to comment each change he applies and the list of all comments concerning the displayed rack is shown.

Before saving the rack, certain configuration rules are checked. In case any rule is violated, the rack Wizard shows an error message, explaining the violated rule and doesn't allow saving the configuration. However, the synoptic representation and the visual configuration help to prevent already certain errors like empty slots in between crates, which would disturb the airflow.

CONCLUSIONS

The rack Wizard is currently used in CMS to insert the electronics racks configuration into the equipment management database. It has been proven to be a useful tool and is used by the electronics and cooling services as primary source on statistical information about power usage and dissipation of the racks. Soon it will also be used to follow the installation process of detector items. The bar code scanning facility will make it a complete solution for asset tracking. The possible use of the Rack Wizard in ATLAS and LHCb is under investigation.

References

- [1] http://www.w3.org/XML/
- [2] http://www.w3.org/Protocols/
- [3] http://www.oracle.com/