ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA047

CABLE DATABASE AT ESS

R. Fernandes†, S. Regnell, J. Persson, S. Gysin, European Spallation Source (ESS), Lund, Sweden
S. Sah, M. Salmič, Cosylab, Ljubljana, Slovenia
L. Johansson, OTIF, Malmö, Sweden

Abstract

When completed, the European Spallation Source (ESS) will have around half a million of installed cables to power and control both the machine and end-stations instruments. To keep track of all these cables throughout the different phases of ESS, an application called Cable Database was developed at the Integrated Control System (ICS) Division. It provides a web-based graphical interface where authorized users may perform CRUD operations in cables, as well as batch imports (through well-defined EXCEL files) to substantially shortened the time needed to deal with massive amounts of cables at once. Besides cables, the Cable Database manages cable types, connectors, manufacturers and routing points, thus fully handling the information that surrounds cables. Additionally, it provides a programmatic interface through RESTful services that other ICS applications (e.g. CCDB) may consume to successfully perform their domain specific businesses.

The present paper introduces the Cable Database and describes its features, architecture and technology stack, data concepts (or entities) and interfaces. Finally, it enumerates development directions that could be pursued to further improve this application.

INTRODUCTION

The European Spallation Source (ESS) is an international neutron research facility currently under construction and expected to operate in 2023. Based in Sweden, ESS is planned to be the most powerful neutron source in the World where a multitude of scientific experiments will take place in its 22 (foreseen) end-stations instruments. For this to happen, both the machine (i.e. accelerator) and end-station instruments are powered and controlled by hundreds of thousands of cables of different types, lengths, purposes and manufacturers. Due to this overwhelming number, an application called Cable Database [1] was developed to help manage the information of (controls) cables that the divisions of the ESS Machine Directorate (and, eventually, its in-kind collaborators across Europe) are responsible for.

The Cable Database allows users to create, read, update and delete (i.e. CRUD operations) cables both through a web-based graphical interface and EXCEL files, ideal for punctual and batch cases respectively. To complement cables-related information, the application also manages the information of cable types, connectors, manufacturers and routing points (users may perform the aforementioned operations on these as well). Moreover, routing points can be associated to cables, thus effectively forming routing paths that track where cables pass (physically speaking) and provide fundamental assistance when installing, testing and maintaining these during the construction, commissioning and operation phases at ESS.

In addition, the Cable Database allows browsing cables through personalized views (e.g. display only the system, owners and installation date fields of cables), as well as filtering cables through flexible queries based on boolean expressions (e.g. filter all cables of type A owned by person B or C and with a length between D and E). Both views and queries are saved and associated to a particular user, which he/she may apply afterwards. These features, amongst others, should guarantee a high degree of satisfaction when interacting with the application (i.e. UX) and increase users' productivity.

DESCRIPTION

The Cable Database has been under development since late 2013, and a first version was publicly released late 2014. During its development phase and first years in production, the application required around 1.5 FTE. Nowadays, it only requires 0.5 FTE mainly for maintenance (i.e. implementation of minor functionalities and bug fixes), training and supporting users. Table 1 summarizes the most important metrics about the Cable Database.

To date, about 10 (major) versions of the Cable Database have been released for production, the latest (version 2.5.6) in September 2019. It currently stores approximately 35000 cables, 290 cable types, 240 connectors, 50 manufacturers and 50 routing points. The amount of data, stored in an open-source RDBMS [2], totals more than 1.2 GB.

Furthermore, the Cable Database is part of an international collaboration called DISCS [3]. This collaboration is composed of several research facilities – ESS being one of them – with the aim of developing databases, services and applications that any (experimental physics) facility can easily configure, use and extend for its commissioning, operation and maintenance.

Table 1: Metrics about the Cable Database

Description	Value
Tables (persistence tier)	22
Constraints (persistence tier)	20
Indexes (persistence tier)	23
Lines of code (persistence tier)	0
Classes in Java (business tier)	188
Lines of code (business tier)	20579
RESTful interfaces (business tier)	13
Web pages (presentation tier)	6
Dialogs (presentation tier)	23
Lines of code (presentation tier)	5304

† ricardo.fernandes@esss.se

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA047

Features

One key feature of the Cable Database (already mentioned previously) is the capability to manage not only the information of cables *per se* but also the information that complement/surround these, namely: cable types, connectors, manufacturers and routing points. Other important features – or characteristics – worth mentioning include:

- Single sign-on: thanks to RBAC [4] (an in-house application that provides user access based on roles), the Cable Database shares single sign-on capabilities with other (web-based) applications at ESS – e.g. Naming Service [5], CCDB [6]. This eliminates the need of logging in more than once when working with these.
- User authorization: currently the Cable Database authorizes users to perform operations through one of three existing roles defined in RBAC: 1) Default (role with minimum privileges that only allows users to view stored data), 2) User (role with intermediate privileges allowing users to view stored data, create cables or update/delete existing cables that he/she owns only), and 3) Administrator (role with maximum privileges allowing users to view stored data and create/update/delete all data concepts (or entities)). These roles provide effective granular control of who can do what in the application.
- Multiple interfaces: several interfaces are provided by the Cable Database to serve different needs including a graphical interface (web-based to support the ESS business model with many users scattered across its European member states), batch data entry through EXCEL files, and RESTful [7] services for applications to consume (see subsection 'Interfaces' for additional details).
- Personalized views: by default, when browsing cables all fields are displayed (see subsection 'Data Concepts (or Entities)' for an enumeration of fields). Users may create personalized views by selecting the fields of interest and the order in which these are displayed, thus increasing users' productivity by displaying only the relevant information for a particular context – e.g. a technician pulling cables only needs to view the cable ID and routing points information (in contrast to seeing all the fields which may be cumbersome/distracting). This configuration is saved (and associated to the user in question) and may be applied later on.
- Quick response: due to the increasing number of cables stored in the Cable Database, it may take some time to start browsing (i.e. viewing) these since the entirety of the data needs to be retrieved from the persistence layer first. To overcome this, lazy-loading [8] of data is fully supported by the application. This means that only the data that actually needs to be displayed in its graphical interface – at a given moment – is retrieved, thus leading to a better response/UX.
- Flexible queries: while it is possible to filter a certain field by specifying a certain value (on the respective column), this can only solve certain use-cases. For example, if cables need to be filtered according to a given range of values, this cannot be done. The Cable Database allows the creation of queries based on user-defined boolean expressions, where sophisticated queries may be specified – e.g. a technician pulling cables may create a query to filter the cables that are not installed yet and of a certain type for the ESS Ion Source. Fig. 1 depicts the graphical interface of the queries configuration.
- Attachments: although the Cable Database attempts to capture most of the information for a proper representation of a cable, it cannot foresee neither support everything. To mitigate the situation, users are allowed to attach artefacts to cables to capture additional information (e.g. a PDF file containing the installation report of a cable).
- Traceability: the Cable Database logs all operations that were performed which changed its data (e.g. when a new connector was created). Users may browse this to have a full understanding of who has done what and when.

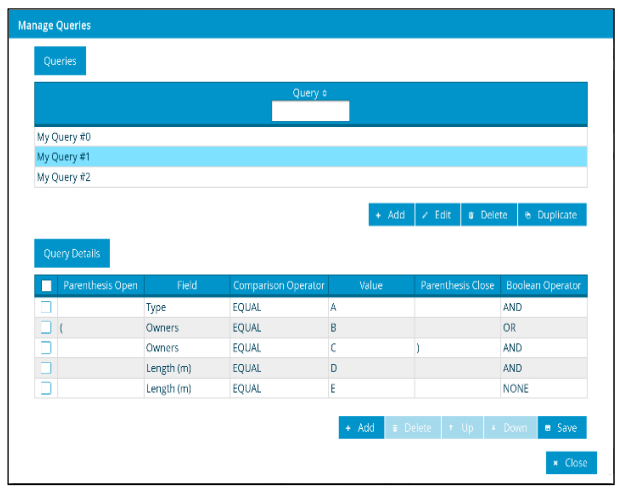

Figure 1: Graphical Interface of Queries Configuration.

Architecture and Technology Stack

The Cable Database is a distributed system based on a classical client-server model [9] where 1) users access its functionalities remotely through a web-based graphical interface and 2) external applications access its data through a programmatic interface. This model – or architecture – is composed of three tiers:

- Presentation: the layer which users interact with.
- Business: the layer which implements business logic.
- Persistence: the layer in which data is stored/retrieved.

Several technologies are employed to implement this architecture guaranteeing that the Cable Database is developed according to user requirements and expectations. Primordial criteria to select the technologies were that they had to be open-source, mature, well documented and actively maintained by the community. With these in mind, PostgreSQL, a relational database management system (RDBMS), is used to implement the persistence tier (i.e.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA047

database) of the Cable Database. The business tier is implemented in Java Enterprise Edition (Java EE) running in an application server called WildFly. It uses Hibernate (a JPA implementation) to access data from the persistence tier and JAX-RS (a Java API) to expose data stored in the Cable Database through RESTful services. It also uses JAX-RS to access data provided by external applications, namely RBAC and Naming Service. Finally, the presentation tier (i.e. graphical interface) of the Cable Database is based on PrimeFaces, a JSF implementation. Fig. 2 illustrates the architecture of the Cable Database (and the technology stack used to implement it).

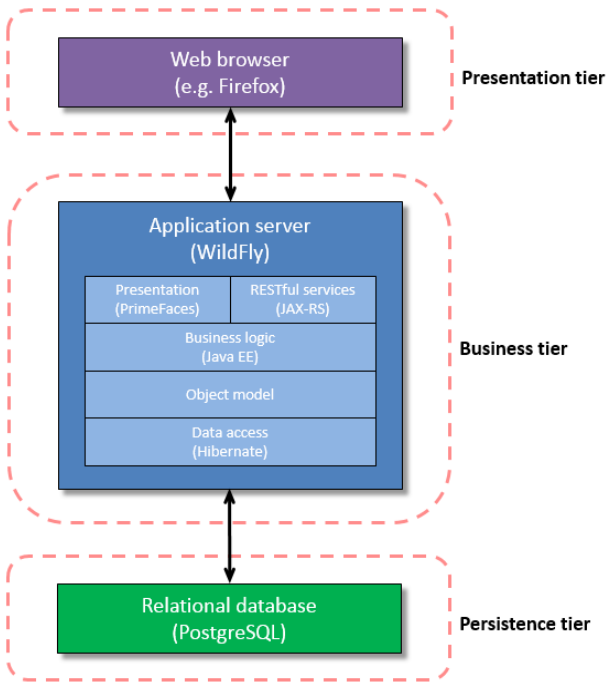

Figure 2: Architecture of the Cable Database.

Data Concepts (or Entities)

The Cable Database supports the management of cables and, complementarily, others entities that are related to these such as cable types and manufacturers. The following enumerates and (succinctly) describes all the data concepts (or entities) currently managed by the application.

- Cable: describes an instance of a cable. It is composed of the following fields: System, Subsystem, Class, Name, FBS Tag, Modified, Cable Type, Container (bundle), Electrical Documentation, From Device A, Location Device A (building), Location Device A (rack), Connector A, User Label A, To Device B, Location Device B (building), Location Device B (rack), Connector B, User Label B, Routing, Owners, Status, Installation Date, Termination Date, Quality Report, Base Length (m), Length (m), Auto-calculated (length), Comments, Revision, and History.
- Cable Type: describes the type of a cable (in an abstract way) and it may be associated to one or more cables. It is composed of the following fields: Name, Description, Service/Function, Diameter (mm), Weight (kg/m), Insulation, Jacket, Voltage Rating (V), Flammable Class, Installation Type, Radiation Resistance (mrad), Manufacturers, Status, Comments, Revision, and History.
- Connector: describes an instance of a connector and it may be associated to one or more cables to represent their terminations (or end-points). It is composed of the following fields: Name, Description, Type, Datasheet, Manufacturers, Status, and History.
- Manufacturer: describes an instance of a manufacturer of cables and/or connectors; it may be associated to one or more cable types or connectors. It is composed of the following fields: Name, Address, Phone Number, Email, Country, Status, and History.
- Routing Point: describes an instance of a point that is used to specify the routing path of a cable; it may be associated to one or more cables. It is composed of the following fields: Name, Description, Classes, Location, Length (m), Owner, Status, Revision, and History.

Interfaces

Being the cornerstone for managing cables-related information at ESS, the Cable Database provides numerous interfaces to read/write data, each tailored to cope with different needs and levels of expertise. These interfaces, available to both users and external applications, are:

- Graphical interface: users can perform CRUD operations in all data concepts (or entities) through a web-based interface provided by the Cable Database. This type of interface is ideal when users need to view information in a user-friendly way, sometimes remotely, with heterogeneous devices (e.g. desktop computer, mobile phone), or to make punctual changes to information. Fig. 3 depicts the main graphical interface of the Cable Database (displaying cables).
- EXCEL file: users needing to perform bulk CRUD operations on cables, cable types and routing points may use well-defined EXCEL files (downloadable from the Cable Database itself) for that purpose. Moreover, the same EXCEL files are used by the Cable Database to export data, which users can modify and import afterwards, thus closing the loop of bulk CRUD operations that may be performed on these data concepts (or entities).
- Programmatic interface: users can programmatically consume RESTful services provided by the Cable Database (to retrieve the data stored in it) and develop scripts and applications. This type of interface is highly effective as it enables 1) users to tackle specific needs that cannot easily be solved with neither of the aforementioned interfaces and 2) external applications to perform well and guarantee they continue being lean (by avoiding applications to store data – already in the Cable Database – in their own persistence layers).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA047

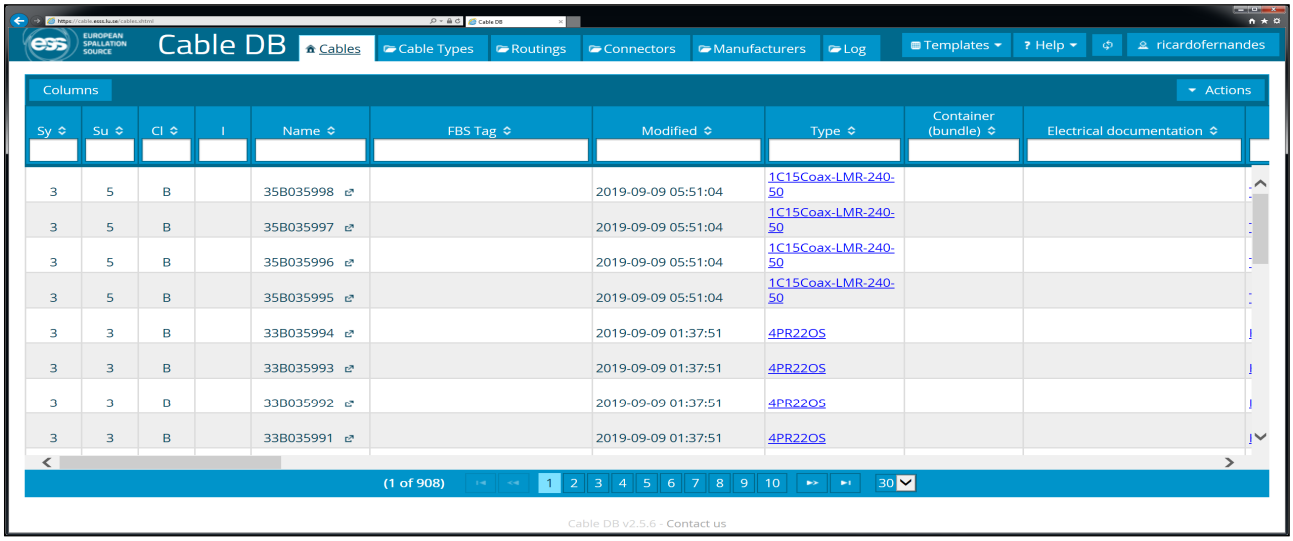

Figure 3: Graphical Interface of the Cable Database.

(Part of an) Ecosystem

Several applications have been developed (or are under development) in recent years to support both integration and controls efforts at ICS. These are producers and/or consumers of services & data that form a rich (logical) ecosystem to solve a myriad of domain (i.e. controls) specific issues such as management of IOCs, generation of PLC code, and calibration of devices. Fig. 4 shows the ecosystem, as well as the Cable Database as a producer of the CCDB and consumer of both RBAC and Naming Service.

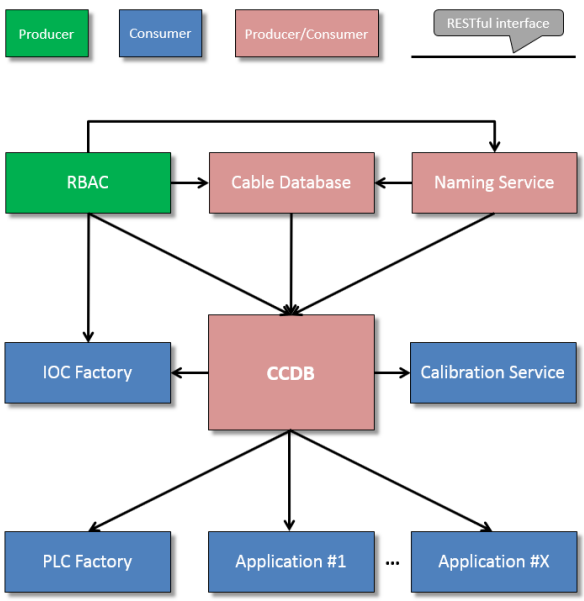

Figure 4: Overview of the ecosystem/Cable Database.

At its core, the ecosystem possesses the CCDB with the main purpose of enabling the collection, storage, visualization and distribution of (static) controls configuration data needed to operate ESS control system efficiently. In this context, the Cable Database is a consumer of data stored in the Naming Service to allow naming the extremities of cables (in other words, the devices that a cable connects from and to) in a formal and unique manner, but also a producer of data that the CCDB consumes through a RESTful interface – provided by the former – to enable it to automatically view connections between devices. This alleviates the CCDB to explicitly store information about connections in its own persistence layer, consequently reducing data duplication and (potential) inconsistencies that could emerge across different applications.

FUTURE DEVELOPMENTS

The Cable Database is used by many people at ESS of different backgrounds (e.g. technicians, cable experts, electrical engineers, integrators) with specific needs. This results in requirements for continuous developments of the application. Additionally, its user base is expected to grow further when the construction/integration phase approaches its pinnacle, which will entail many new requests for additional functionalities. The following missing functionalities have already been identified as candidates for development:

- Reports generation: at present, the Cable Database does not generate reports (or metrics) about the data that it manages. As these could provide valuable insights, the application should support the generation of reports that summarizes/aggregates its stored data – e.g. how many cables are installed, eventually per system; how many cables per subsystem are owned by a certain person; how many cables exist per class; who is the manufacturer that supplies the most cable types.
- Routing paths rendering: since the information of routing points is (also) managed by the Cable Database, it could be beneficial to extend these with spatial coordinates (e.g. x, y and z) and enable the application to display (i.e. render) cables' routing paths within a 3D space. This could help users understand where cables are passing through and facilitate a better planning of cables' installation layout.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA047

- Batch operations: while EXCEL files exist for cables, cable types and routing points, they do not for connectors and manufacturer. Therefore, EXCEL files for these two data concepts (or entities) should be provided/supported by the Cable Database to enable batch operations on these.
- Programmatic writing: currently the Cable Database provides 13 RESTful services to get (i.e. read) data stored in it. New RESTful services to put (i.e. write) data into the Cable Database should also be developed so that external applications (i.e. consumers) can programmatically insert cables in the Cable Database – e.g. instead of Eplan, a CAE tool used at ESS, exporting data in a custom EXCEL file – and forcing users to manually rearrange the data to fit in the EXCEL file provided by the Cable Database (to enable importing the file afterwards) – it could consume these new RESTful services to directly export (i.e. write) data in the Cable Database.

CONCLUSION

The Cable Database is in production since the end of 2014 and successfully fulfils its mission at ESS. Currently, this application mostly manages the information of cables of ESS High Beta Linac (HBL), Medium Beta Linac (MBL) and Spoke Linac (Spk) systems, which users may view through its graphical interface or external applications to consume in order to solve domain (i.e. integration/controls) specific issues thanks to information retrieved from it (via RESTful services).

Over the past years, many versions of the Cable Database have been released for production, and it currently manages (i.e. stores) approximately 35000 cables with 370 attached artefacts (mostly PDF files), 290 cable types, 240 connectors, 50 manufacturers and 50 routing points. The amount of data totals more than 1.2 GB and is stored in PostgreSQL, an open-source RDBMS.

Several interfaces (namely: graphical, EXCEL files importer/exporter, and RESTful) have been developed to cope with a multitude of requests so that users/external applications may achieve their goals with minimum effort imposed by the Cable Database.

By supporting, amongst others, personalized views and flexible queries, users may obtain/visualize (solely) the data of interest – rather than the entire stored data – and, hopefully, help them to stay focused and productive. Additionally, thanks to lazy-loading features incorporated in the Cable Database, users do not have to wait long until the data is displayed, leading to a significantly better experience (i.e. UX) when interacting with it.

Finally, new functionalities are being considered such as generation of reports, rendering of cables' routing paths within a 3D space, additional EXCEL files to enable batch operations of connectors and manufacturers, as well as new RESTful services to write data into the Cable Database. These will substantially improve the application and enable both users and external applications to profit even more from its usage.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to all the people that participated in discussions concerning the Cable Database, in particular Evangelia Vaena (Specialized Technical Services Group), Hinko Kocevar (Beam Diagnostics Group) and Eugene Tanke (Engineering Resources Group) at ESS.

REFERENCES

[1] Cable Database, http://openepics.sourceforge.net/cables
[2] Relational Database Management System, https://en.wikipedia.org/wiki/Relational_database#RDBMS
[3] V. Vuppala *et al.*, "Distributed Information Services for Control Systems", in *Proc. 14th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'13)*, San Francisco, CA, USA, Oct. 2013, paper WECOBA02, pp. 1000-1003.
[4] Role Based Access Protocol, http://openepics.sourceforge.net/security
[5] Naming Service, http://openepics.sourceforge.net/naming
[6] R. N. Fernandes, S. R. Gysin, S. Regnell, S. Sah, M. Vitorovic, and V. Vuppala, "Controls Configuration Database at ESS", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 775-779. doi:10.18429/JACoW-ICALEPCS2017-TUPHA156
[7] R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures", Ph.D. Dissertation, University of California, Irvine, 2000.
[8] Lazy-loading, https://en.wikipedia.org/wiki/Lazy_loading
[9] Client-server model, https://en.wikipedia.org/wiki/Client%E2%80%93server_model