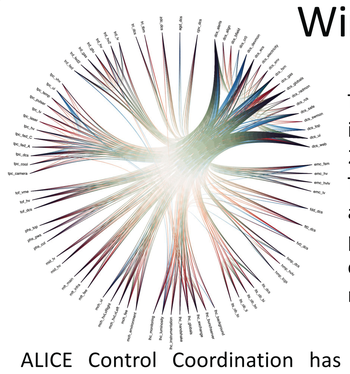
VISUALIZATION TOOLS TO MONITOR STRUCTURE AND GROWTH OF AN EXISTING CONTROL SYSTEM

ALICE

CERN

WinCC projects interdependence

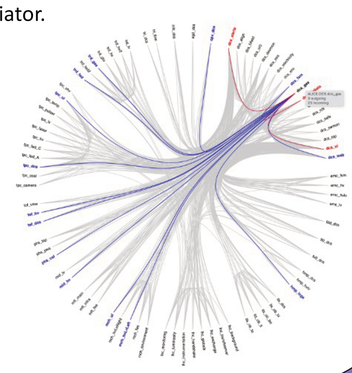


The ALICE core control system is running on 200 computers, including 85 WinCC OA nodes, distributed among 15 detectors and 2 main central systems.
Thanks to the native WinCC OA DIST mechanism, it is very easy for a project to access datapoints on a remote project. DIST is powerful but WinCC lacks tools to control and monitor the status of connections. In particular, as connections are symmetric, it is not easy to determine the initiator.

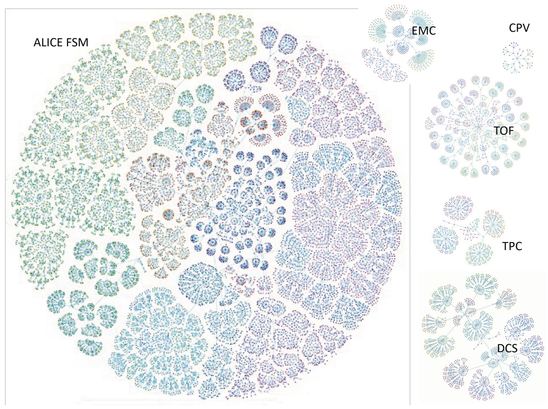
ALICE Control Coordination has defined a policy to allow connection between nodes of the same detectors, and to/from the central systems.
The pictures show authorized (left) vs active connections (right):

- snapshot of the active connections is extracted from WinCC
- the diagram of authorized connections is maintained in a central configuration file

Comparison of the two sets will reveal any unwanted and missing connections.
Realized with *javascript* and *D3.js* library.



The Finite State Machine



Hierarchical control of ALICE and its detectors is accomplished with the help of a Finite State Machine (FSM). Realized in SML/SMI++, this logical structure allows to simplify the operations through simple keywords representing the device states and the actions to be performed.
The ALICE FSM is the skeleton of DCS operations performed in the control room: through this hierarchy the DCS shifter can interact with detectors' parts and devices, even without a specific knowledge of the details.

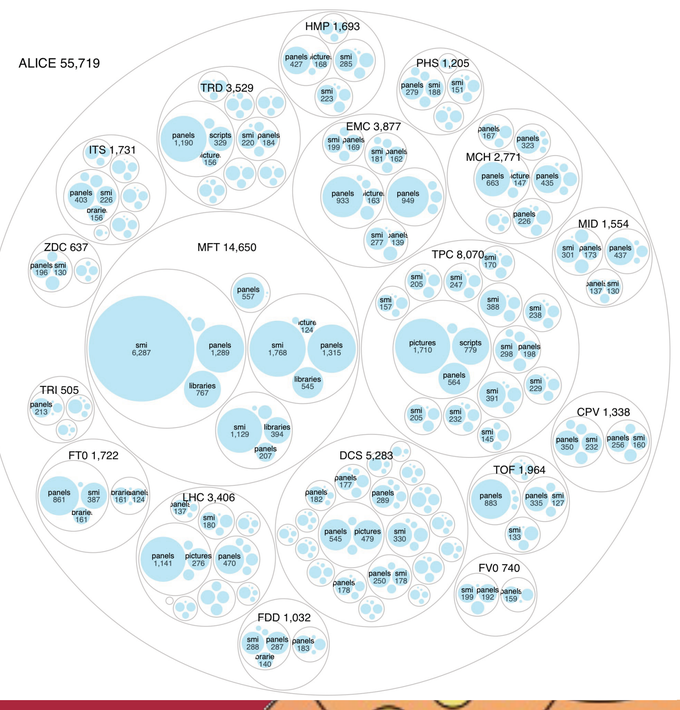
To represent the full ALICE FSM in a graphical way, we have extracted all nodes and branches in a *json* file using the FwFSM API, and visualized it in a flower-like way, with *javascript* and the *D3.js* library . In the above Figure we have more than 17000 nodes, 11500 of which are hardware devices (the leaves). Three systems represent 70% of the whole set, and their complexity is sometimes source of problems in the overall operation. By looking at this figure, and evaluating the structure of such complex branches, we were able to assist developers and address them to some optimizations.

WinCC projects complexity

The control system of each of the 15 detectors forming part of ALICE is developed on one or more worker nodes, depending on the complexity and partitioning of the detector. WinCC and other basic software is installed on each node, and detector experts develop custom files as:

- Panels: XML (or WinCC PNL) files representing the graphical interface with hardware and procedures.
- Pictures: graphical widgets and maps included in panels.
- Libraries: C-like code (WinCC ctrl++ language) containing reusable detector specific code.
- Scripts: C-like code (WinCC ctrl++ language) executing on-demand or automatic procedures.
- and SMI codes: the Finite State Machine logics regulating the operational hierarchy.

Besides the local file system, each detector group can also use a network file system shared between all its nodes, and reachable from the central projects, meant as the final repository.



Using Microsoft Powershell scripts, the number of files developed by the experts in each group was evaluated, both in the local system and in the shared file system. The picture is generated with *javascript* and the *D3.js* library.

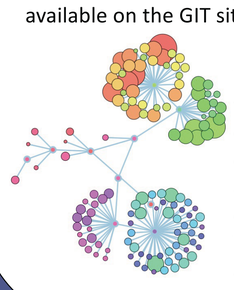
The bubble picture shows that some of the more complex detectors are actually controlled with a limited amount of software, while some of the newer detectors have developed enormous amounts of software.

The use of the shared system is also not optimal, despite the advantages it offers. These observations indicate to us that a new campaign of information and specific instructions shall be distributed during the coming winter shutdown, in order to improve the file system occupation.

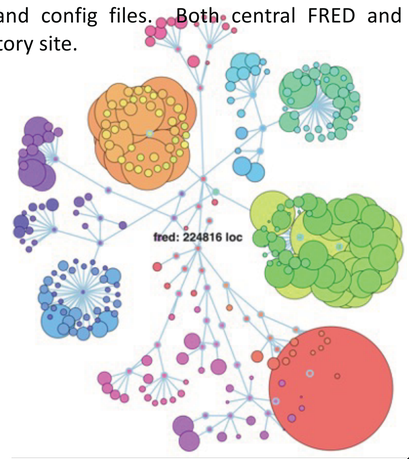
Front-end software

ALICE upgrade to the online-offline O^2 readout system has required the development of a new board, called Common Readout Unit (CRU), able to manage physics and condition data on optical lines, and interfaced by a new driver (ALF) and a frontend software called FRED. FRED is developed and maintained by central DCS, and allows access to the CRU without the need for an extensive software development. Detectors' customization is facilitated by the use of API and config files. Both central FRED and detectors' extensions are maintained in a common GIT repository site.

In order to check the extent of customization by detectors, a comparison was made between the lines of code developed for the FRED core (left) and the detector branches (right) available on the GIT site.



Through a *python* script, we scrolled through the entire software repository, computed the number of lines of code of the different file types in a *json* file, and realized a graphical representation of the GIT site.
The picture is generated in *javascript* and *CodeFlower*, based on the *D3.js* library.

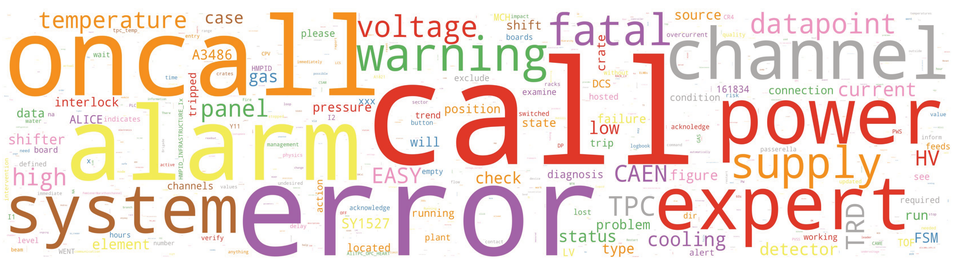


Documentation for operators

Operators in control room supervise alarms and other events generated by detectors' nodes, and consult specific help files where they should find instructions to intervene on the faulty system.

Using *python* and *matplotlib* we analyzed the documentation prepared by the experts for DCS shifters. We realized a *word cloud* according to the occurrence of the words themselves. It is interesting to notice that the most common words refer to the intervention of experts ("call", "oncall", "expert", ..), more than giving instructions to be realized directly by the shifter.

In carrying out this word analysis, we also noticed the massive presence of expert names and telephone numbers, which often turned out to be obsolete.



O. Pinazza[†,1,2], A. Augustinus[2], P. M. Bond[2], P. Chochula[2], A. Kurepin[3], M. Lechman[2,4], D. Voscek[2]
[†]corresponding author; [1]INFN, Sezione di Bologna, Italy; [2]CERN, Geneva (CH);
[3]Affiliated with an Institute Covered by a Cooperation Agreement with CERN, Geneva (CH);
[4]University of the Witwatersrand, Johannesburg, South Africa.