VACUUM CONTROLS CONFIGURATOR: A WEB BASED CONFIGURATION TOOL FOR LARGE SCALE VACUUM CONTROL SYSTEMS

CERN, 1211 Geneva 23, Switzerland

ABSTRACT

The Vacuum Controls Configurator (vacCC) is an application developed at CERN for the management of large-scale vacuum control systems. The application was developed to facilitate the management of the configuration of the vacuum control system at CERN, the largest vacuum system in operation in the world, with over 15,000 vacuum devices spread over 128 km of vacuum chambers. It allows users to easily integrate or modify the vacuum devices within the control system via a web browser. It automatically generates data for the configuration of the communication between vacuum devices and the supervision system, the generation of SCADA synoptics, long and short term archiving, and the publishing of vacuum data to external systems. VacCC is a web application built for the cloud, dockened, and based on a microservice architecture. The application is divided into 4 microservices: front end, validation & persistence, exporter, and synchronizer.

The front end microservice provides the user interface of the application. It allows users to be abstracted from the complexity of vacDB, enabling them to modify vacuum machine parameters that are required for the export of SCADA and PLC configuration files.

The single page application is organized following the React model, where web elements such as pages and their elements (buttons, tables, forms, etc.) are hierarchically organized into components. The components interact with backend services (validation & persistence, exporter, and synchronizer microservices) using REST and WebSockets. WebSockets are used in special cases of long lasting microservices.

The validation & persistence microservice is responsible for providing the interface between other microservices and vacDB. It contains all RESTful APIs that allow other microservices to indirectly perform CRUD operations on the database.

- API: exposes REST endpoints for other services to interact with vacDB.
- Service Layer: handles requests from the API layer, performs data validation, and when necessary, combines results from multiple Data Access Object (DAO) operations to serve API requests.
- Data Access Objects: provides objects that allow direct access to vacDB. This layer is implemented with Spring Data and Hibernate.
- Auditing: Records modifications in the configuration of the vacuum control system.
- Security: Provides authentication and authorization services for the entire application, ensuring that only users with the required privileges are able to perform database operations.

The exporter microservice is responsible for generating the configuration files for both the PLCs and for the SCADA.

For each PLC, the exporter generates function block calls for each vacuum device connected to it, along with device datatodos; these contain all relevant information that will allow PLCs to connect and interact with device controllers.

For the SCADA, the exporter microservice generates configuration files with the data that will allow the configuration of all datapoints for every vacuum device. Each datapoint will be configured with the information that will allow PLCs to connect and interact with device controllers.

The purpose of the synchronizer microservice is to automatically import vacuum data from the Layout DB into vacDB, ensuring that the official, approved layout of the vacuum system is reflected in vacDB.

The Layout DB is a CERN-wide database that models the architecture of CERN's accelerators. It contains data concerning most accelerator subsystems, including RF, beam instrumentation, magnets, cryogenics, and vacuum.

Users can trigger a differential analysis between vacDB and the Layout DB. The differences detected in the analysis are based on the create, update, and delete operations made on the Layout DB that are not reflected in the Machine DB, concerning vacuum sections and equipment.

Every change made to the code repository passes through a pipeline that will build, test, and lint the code. In case of errors, the pipeline will stop and the developer will be alerted. Commits pushed to the master branch of the repository that pass the build, test, and lint stages are automatically deployed to the staging environment, a replica of production, where developers can perform additional testing. After validation in the staging environment, a tag of the master branch is created, and developers can trigger an automatic deployment to production.

The adoption of a microservices architecture in vacCC brought several advantages. It allowed to split a big problem into smaller, independent, and more easily manageable pieces of software, where software developers are able to work simultaneously in the different system components. Future upgrades of vacCC to new technologies can now be automatically performed in vacDB and the Layout DB, allowing the developer to focus on the application development.