Notes and Logbook

Sinan DAROUKH

4th May 2019

Table of contents

Table of figures and illustrations

1 Internship context

1.1 The CERN

The Frenchman Louis de Broglie, who was awarded the Nobel Prize in physics in 1929, was behind the creation of CERN. In 1949, he proposed the creation of a European scientific laboratory in order to breathe new life into scientific research following the Second World War. In 1952, with the support of UNESCO, the European Council, with the support of the United Nations Educational, Scientific and Cultural Organization (UNESCO), decided to set up a European scientific laboratory for Nuclear Research (CERN), it is the result of an agreement between eleven European governments. The municipality of Meyrin, located near Geneva, was chosen to host this laboratory. In 1954, the first construction work on the site began and on the 29 of September the CERN Convention was signed by twelve European States. It was at this date when the research centre, then called the European Organisation for Nuclear Research, was officially established.

Today CERN is very well known thanks to its experiences and more particularly its particle accelerator. However, it should be noted that many accelerators have succeeded one another on the laboratory site. The first was the Synchro-Cyclotron a protons inaugurated in 1957. The accelerators being larger and larger, an agreement was made with France in 1965 to expand the site on French territory. Then in 1981 it was decided to build an accelerator in a tunnel with a circumference of 27 kilometers located 100 meters deep underground. This tunnel, which is housed in first the LEP (Large Electron Positron Collider). This one was replaced by the current LHC (Large Hadron Collider) in 2008. CERN now has 23 member countries and around 2,500 employees and more of 17,500 researchers who come to the Meyrin site to carry out experiments on the premises of the research centre. The budget of such an organisation amounts to more than 1 billion Swiss Francs per year. This funding is fully covered by the Member States.

1.2 The LHC - Powerful particle accelerator

The LHC, or Large Hadron Collider, is the most powerful particle accelerator in the world built to date. It is the last link in a huge particle accelerator complex, which accelerates protons or heavy ions such as lead and creates collisions. Particles in the LHC reach a speed of 99.9999991% the speed of light. The collisions take place at four points on the accelerator where the four LHC experiments are located: ALICE, ATLAS, CMS and LHCb.

1.3 The LHCb Experiment and Internship goals

I am currently working at the LHCb experiment departement which is one of experiment of the LHC. The main goal of my internship is to streamline the connection process to the monitoring system. Actually the LHCb departement use multiples ssh connection to monitor the experiment and its results. In order to simplify this workflow, I have been investigating on special components on WinCC-OA to set up pre-existing project online.

2 Technologies

2.1 What is WinCC-OA ?

SIMATIC WinCC is a Supervisory Control And Data Acquisition (aka SCADA) and human-machine interface from Siemens. SCADA systems are used to monitor and control physical processes involved in industry and infrastructure on a large scale and over long distances. SIMATIC WinCC can be used in combination with Siemens controllers. WinCC is written for the Microsoft Windows operating system, but it could be use on Linux operating system. WinCC-OA aims are mainly to control and acquire data from the sensors from reals controls on experiments. You may have heard about PVSS, it was a SCADA system, made by ETM. Siemens now owns ETM and rebranded PVSS as WinCC-OA, but it's still the same tool. It's a tool for building SCADA applications. WinCC-OA is the SCADA system chosen by JCOP (CERN).

2.2 What is JCOP ?

JCOP stands for Joint COntrols Project which is a collaboration between the LHC experiments, the PH Departement and EN-ICE, the Controls Group in the Engineering Departement. JCOP aims to reduce the overall manpower cost required to produce and run the experiment control systems. The JCOP Framework provides all the components required for WinCC-OA tool. Basically, it's a layer of software components and shared tools that might be useful for modelling LHC Experiment.

Around the end of 1997, a common project, the Joint Controls Project (JCOP), was setup between the four LHC experiments and a Controls group at CERN, to define a common architecture and a framework to be used by the experiments in order to build their Detector Control Systems (DCS). The JCOP Framework adopted a hierarchical and highly distributed architecture providing for the integration of the various components in a coherent and uniform manner. The Framework was implemented based on a SCADA (Supervisory Control And Data Acquisition) system called WinCC-OA (formerly PVS-SII). While WinCC-OA offers most of the needed features to implement a large control system, it was felt that a tool for implementing higher-level logical behavior was missing. For this reason, the JCOP project was complemented by the integration of SMI++; a

toolkit for sequencing and automating large distributed control systems, whose methodology combines three concepts: object orientation, Finite State Machines (FSM) and rule-based reasoning.

2.3 HTTP Server - Addons

The HTTP Server integrates Internet technologies directly into WinCC OA. Implemented as an add-on for the Control Manager, possible applications are many and diverse. The HTTP Server can be used as a web server for static HTML pages including download facility or to generate dynamic HTML pages. This allows you to make data from WinCC OA available to many users at the same time. The complete WinCC OA programming language Control is thus available for server-side scripting tasks.

Advantages of the HTTP Server :

- Provide custom HTML pages incorporating data from WinCC OA for intranet / Internet.
- WEB alert screen with acknowledge function and free choice of filters.
- Diagnostics page displaying hard disk capacity, RAM, processor usage, alert throughput, users logged in and more.
- "HTML references" allow you to replace $parameters at runtime without any special knowledge of HTML, that is, you can query any data points through the Internet.
- Querying WinCC OA data using SQL
- Transfer static HTML pages
- File transfer (HTTP download).
- HTTP Server uses WinCC OA specific tags in HTML.

2.4 ULC UX - Addons

The ULC UX is a User Interface client for WinCC OA based on the VNC technology and HTML5 technology. It is used as a new alternative to the WinCC OA UI (native UI on Windows and Linux) and the WinCC OA Web Client (Plug-in-based web solution). The ULC UX can use already engineered panels without any special adaptions, and it provides a very high compatibility to the native UI functionality

Benefits : The ULC UX combines following benefits in a single UI solution:

- No installation is required on client side
- Communication is secured by using SSL encryption
- User management can be performed by using SSO
- Usage of WinCC OA panels without changes to the existing control scripts

Performance Requirements : There are parameters which have an influence on the performance of the ULC UX in the web browser.

- Bandwidth
- Latency
- Dynamic panel content
- CPU performance of the web server
- Available memory on the web server

Because the performance and the possible ULC UX connection to one web server are highly addicted to the panel and the scripting inside the panel we cannot guaranty any fixed number of possible connections. This will be highly customer panel dependent and the customer will need to test his own panels to get a possible ULC UX connection. If the performance of one web server is not sufficient load balancing can be used with multiple web servers.

Architecture The basic architecture displayed in the figure below gives an overview for the involved WinCC OA managers or software modules for the WinCC OA ULC UX.

When a browser tries to connect to the ULC UX - URL of the WinCC OA HTTP Server, the HTTP server returns the ULC UX web page and automatically starts a local WinCC OA UI manager. This server side UI manger transfers displayed information of the UI into HTML 5 interpretable data chunks.. A Java Script library at Client side afterwards interprets those data chunks and draws the graphics in the browser.

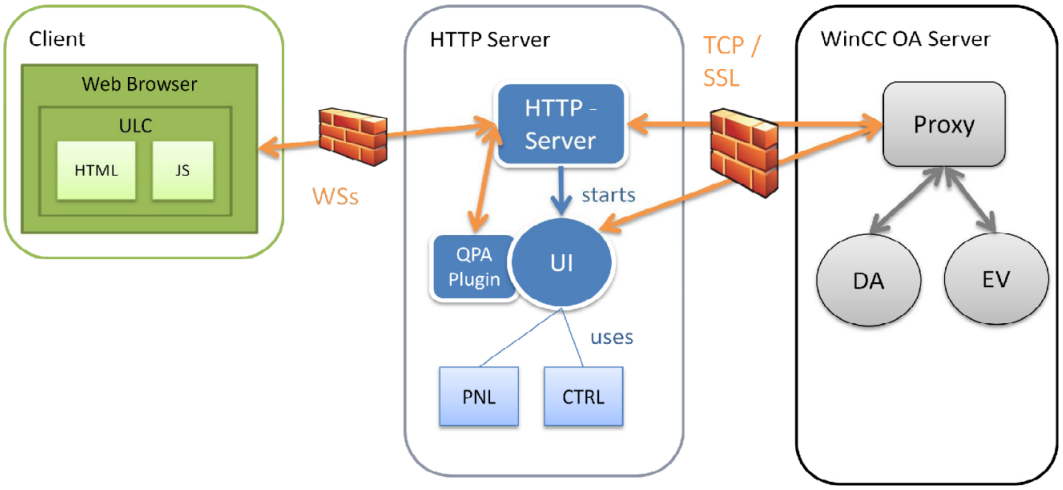

Figure 1: ULC UX Architecture

Appendices

A Logbook

A.1 Iteration from Monday 4th May to Friday 15th May

Objectives of the iteration : The main goal of this iteration was to get more familiar with WinCC-OA and the JCOP Framework. Also the main idea was to think about how to make the LHCb experiments monitor more accessible from the web without changing the WinCC-OA pre-existing project drastically.

Tasks done & notes :

- Reading documentations about WinCC-OA.
- Setting up the main project.
- Reading/Writting emails to the ISIMA supervisor (Mr HILL) and the CERN one (Mr Luis Granado Cardoso).
- Helping and exchanging with Loann, who is working on the same technology as me.
- Going through the tutorial slides.
- Experimenting through Exercice 1,2,3 and 4.
- Facing some difficulties with the CTRL system.
- Reading documentation about HttpServer
- Watching videos about WinCC-OA on Siemens and KAASM's youtube channel.
- Doing daily meetings with Luis to track the progess.

A.2 Iteration from Wednesday 15th May to Sunday 20th May

Objectives of the iteration : The aim of this iteration was to look at all the documentation about the WebServer components in WinCC-OA, how to navigate properly between panels, how to manage users access permissions and restrictions, and how to well organise configs files in a project. I was sick during a part of this iteration, so I did not work effictively on Monday

Task done :

- Reading all the documentation

Notes : I have been reading the documentation about the configuration files :

- config.level is for managers configurations, and basically, loading libs through this file.
- config.redu is for redundancy, that we are not seeking for, at the moment
- config.http is a premade file from the WinCC-OA folder
- config.webclient is an additional file for configuration Desktop and Mobile UI add-ons... (Not what we are looking for)

I have tested multiple stuffs on those. Nothing really can simplify the actual config (root) file. I guess there's another option not yet tested which is the own configuration file, I don't know if we can use more than once.
For the navigation fluidify and simplification issues, I have been testing stuffs and reading about the differences between Modules, Embedded Modules , Child and Root Panel.
To navigate properly, I think I should take a look at the Topologies panels components...
The embedded modules through one root module, may be the best layout option.

A.3 Iteration from Wednesday 20th May to Wednesday 27th May

Objectives of the iteration :

- User permissions, one can connect to view, another can connect to administrate
- Play with the alarm Screen, make a shortcut to it
- (FSM) Implement one, how it goes on the ULC components

Tasks done :

- Login panel has been implemented (worked on both ULC and Regular WinCC)
- Can access on user admnistration panel through it : Module > SysMgm > Permissions > User Admnistration, which on root access list all users and permissions
- Groups > Admnistrate > Permissions : You can change the group permissions but also, create new groups with new permissions

- Permissions are logged in an Authorization Bits system. The first five bits are already define, they are predefined and un-changeable, but you can change the description if you want and texts of it :
  - 1 : Visualisation: Visualize only
  - 2 : Normal operator authorization: permits the opening of child panels.
  - 3 : Advanced operator authorization: permits execution of commands, explicit setting of replacement values, input of correction values as well as changes to all value range types.
  - 4 : Administration: permits the use of the PARA.
  - 5 : Acknowledgement: permits acknowledgment of alerts.
  - 32 : Allows SSO for one work station
- Can also access on login statistics panel to see whose connected : Module > Login Statistics
- We can manage the Components access thanks to the boolean getUserPermission() function : CONTROL > Control functions > G > getUserPermission()
- We can check on the Main Panel, by Login via Guest or via Root
- Auto login done, with inactivity (Glitch with inactivity, security one) (UI number changed some time)
- Alarm Panel : I have worked on it, but there is a glitchy features, the windows appears behind the Alarm Screen
- I also haven't the time to experiment on the FSM.

A.4 Iteration from Wednesday 27th May to Tuesday 2nd June

Objectives of the iteration :

- Adding and testing the FSM, on the Web app, it actually works but... it took me a lot of times.
- Create Shortcut for opening differents kinds of panels, I had issues with the DEN Panel... But now it works
- Didn't check on the Alarm Screen and the Users Permissions, as I had issues with the DNS part of the tuturials.

A.5 Iteration from Thurday 3rd June to to Monday 8th June

Objectives of the iteration :

- Look at the documentation about Distribution Manager, Distribution Configs Files, Distributions Systems etc...
- Test to implement the following panel :
  (Location : /localdisk/wincc/prod/["LBECSINFO","ECS"]/panels/)
  - lbAlarmHandling/lbAlarmScreen.pnl
  - FarmUsagePlot.pnl
  - lbECS/lbOPCMonitor.pnl
  - lbTriggers/lbTriggersOverview.pnl
  - lbTrending/lbTrending.pnl
- Use the current project as the main client, connected to the ECS and LBECSINFO projects.

I have read a lot of the documentation about the distributed managements, and distributed systems. I have learnt that you can configure them through an existing wizard only if you create a new project. Else you should do it via the configuration files. I have tried to launch script from the copied project, but without any success.
The Test Project configuration file should look like this :
[general]
distributed = 1
[dist]
distPeer = "dist_name_ECS" 1130
distPeer = "dist_name_LBECSINFO" 1140

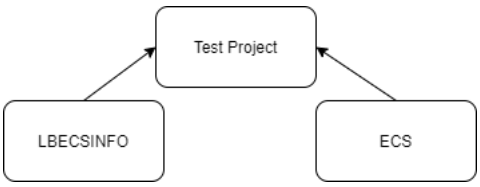

Figure A.1: Example of distributed projetcs

A.6 Iteration from Monday 8th June to Wednesday 10th June

Objectives of the iteration : The meeting with Luis helped me debugging the and connecting the two distributed projects. I am currently trying to launch differents panels

from the ULC web components. I have succeed to open them, but... the data point seems to not be well-connected or at least are connected to the wrong project. I achieved this by updating the config files, which is I think not the cleanest way to do it, but the fastest (Testing first) We can explore two options which are : copying all the project in a special repos, or we can try with the addSymbol() functions, or we can also try to copy all the paths.

A.7 Iteration from Friday 12th June to Friday 19th June

A.8 Iteration from Friday 19th June to Wednesday 24th June

Objectives of this iteration :

- Create a new main page for the app, which should be more similar than the one already in use.
- Create shortcuts to the following panels :
  - LHCb Top FSM - Without the possibility to click anywhere
  - LHCb - LHC - Big Brother
  - Alarms Screen
  - Trending panels
- Also check if we can have to access to other stuffs than WinCC-OA panels onto the ULC UX components.

B Real Gantt chart

C References