Purpose of the IEPLC framework

The Programmable Logic Controller s (PLCs) and other micro-controller families are essential components of the CERN control system. Together with their weaknesses and their strengths they typically present custom communication protocols and it is therefore difficult to federate them into the control system using a single communication strategy. The purpose of IEPLC is to mitigate the communication issues by providing communication interfaces in a hardware independent manner. In addition it automatically generates all the resources needed on client and controller side to implement a common and generic Ethernet communication.

The IEPLC framework is composed of a set of tools, scripts and a C++ library. The configuration tool allows the definition of the data to be exchanged and their instantiation on different controllers within the control system. The scripts generate the resources necessary to the final communication while the library eventually allows the application on the client side to send and receive data to/from the different controllers. Finally, a diagnostic tool is proposed to validate the whole configuration without implementing a single line of code.

Context: The Three-tiers control system

The control system splits over three different responsibility tiers which communicate among them via TCP/IP protocol over Gigabit Ethernet connection:

- The presentation tier, the middle tier and the resource tier

Both equipment controllers and FEC belong to the resource tier. In the IEPLC model, FEC acts as bridges between the presentation tier and the physical layer by standardising the integration of industrial components within CERN's global infrastructure.

Keywords:
- Front-end computer (FEC), VNEM/PCI platforms, Debian/RT-Linux, FLESA framework.

The parameters file

The library that is instantiated by a client application running on the FEC (FLESA server, python scripts, C/C++ software, etc.) first loads the parameters file(s) which reflects the exact mapping of the class deployed on the target controller. IEPLC allows the same equipment class to be instantiated over different heterogeneous controllers as well as deploying different classes on the same controller.

Keywords:
- Classes, VNEM/PCI platforms, Debian/RT-User.

The client library

The C++ library handles the communication between the FEC and the controller(s) hiding all the communication details through a very simple and high-level API. It implements the low-level protocols of the supported PLCs and micro-controllers including communication protocols, high-level communication parallelisms, diagnostics, etc. Use of native and standard protocols of the industrial ethernet coupler and polling mode for data transmission simplifies considerably the communication process and makes the service fully generic, and very flexible and scalable.

Keywords:
- Native protocols: SAC/Modbus-TOC, NI-CN.

The device model

IEPLC provides a non-intrusive solution by generating only data structures and no process code, not even for the communication tasks. Different memory mapping can be adopted to optimize the network traffic. The user relies on the manufacturers software and tools to upload the controllers with the generated code.

Keywords:
- Step 7 STU/SCI, Unity Pro XPC, C-code.

The generation process

The generation, fully automated by a set of python scripts, generates two kind of resources:

- The parameters file: an XML document used to provide the library with all information required to communicate with the predefined controller (hostname, protocol, mapping, ...)
- The controller sources: the code to be uploaded on the controller itself

A checksum mechanism ensures the consistency of each IEPLC configuration.

Keywords:
- Python, CICL2, NFS, SVN.

The diagnostic tool

A graphical tool provides the user the possibility of interacting with hardware adopting the IEPLC service for diagnostic and testing purposes. Full control is given in sending and receiving data from the tool, so that specialist can validate his complete configuration before operational delivery.

What about the future?

Both IEPLC and POGA technologies could be used to speed-up and standardise the integration of controllers’ custom hardware within the framework. Proposal is to extend the IEPLC framework in order to be able to communicate with generic WAGO/ON or ABB Allen-Bradley. POGA will be used as a bridge between the SPECIFIC ID protocol and the Ethernet connection using protocols developed by OSI, University of Bristol and ONERA, respectively called Etherbone and IPOP, bridge.

Keywords:
- SPECIFIC ID, Etherbone, IPOP.