WEB GUI DEVELOPMENT AND INTEGRATION IN LIBERA INSTRUMENTATION

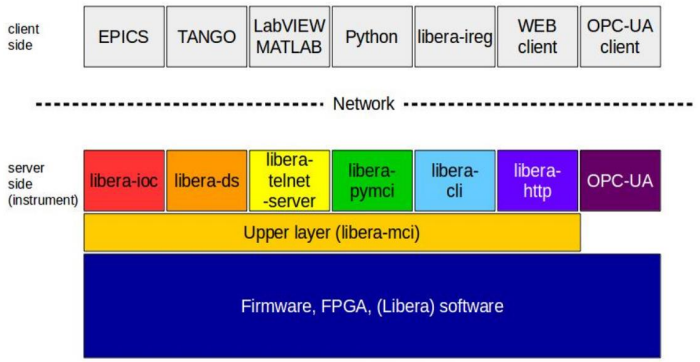
D. Bisiach, M. Cargnelutti, P. Leban, P. Paglovec, L. Rahne, M. Škabar, A. Vigali

Instrumentation Technologies doo, Solkan, Slovenia

ICALEPCS 2021

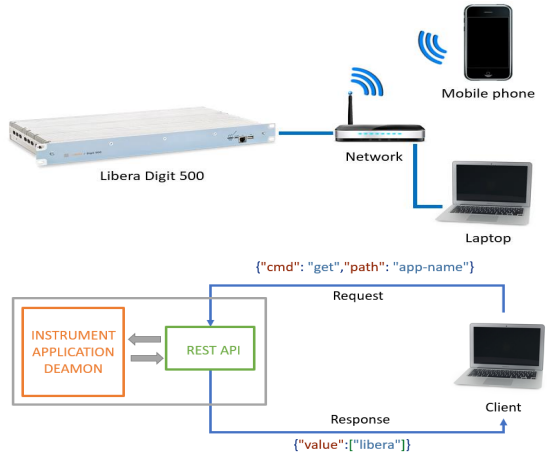
INTRODUCTION AND APPLICATION LAYER

- Allows communication between the instrument and the Software layer
- Exposes the application nodes through the Machine Control Interface (MCI)
- APIs provide high level accessibility



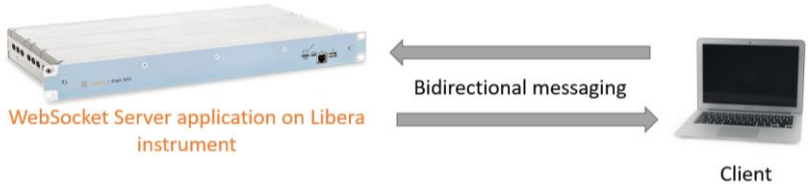
HTTP APPLICATION ARCHITECTURE BASED ON REST API

- Exposes the instrument SW to the network using HTTP API
- Integrated in the local network
- Perform trobuleshooting in an easier and more efficient way

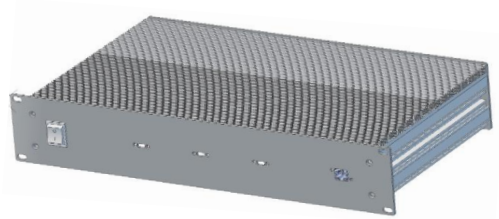


WEBSOCKET

- High performance on data streaming thanks to full duplex communication interface
- Introduced to meet the high demanding requirements on data streaming, minimum latency and system reliability



12 CHANNEL LIBERA CURRENT METER IMPLEMENTATION, RESULTS AND CONCLUSIONS



WEB GUI DEVELOPMENT AND INTEGRATION IN LIBERA INSTRUMENTATION

D. Bisiach, M. Cargnelutti, P. Leban, P. Paglovec, L. Rahne, M. Škabar, A. Vigali

Instrumentation Technologies doo, Solkan, Slovenia

ICALEPCS 2021

ABSTRACT

During the past 5 years, Instrumentation Technologies expanded and added to the embedded OS running on Libera instruments (beam position instrumentation, LLRF) a lot of data access interfaces to allow faster access to the signals retrieved by the instrument. Some of the access interfaces are strictly related to the user environment Machine control system (Epics/Tango), and others are related to the user software preferences (Matlab/Python). In the last years, the requirement for easier data streaming was raised to allow easier data access using PC and mobile phones through a web browser. This paper aims to present the development of the web backend server and the realization of a web frontend capable to process the data retrieved by the instrument. A use-case will be presented, the realization of the Libera Current Meter Web GUI as a first development example of a Web GUI interface for a Libera instrument and the starting point for the Web GUI pipeline integration on other instruments. The HTTP access interface will become in the next years a standard in data access for Libera instrumentation for quick testing/diagnostics and will allow the final user to customize it autonomously.

INTERFACE BETWEEN INSTRUMENT AND APPLICATION SOFTWARE

The access to the setting and the data provided by the instrument is allowed by the software structure reported in Fig.1. The lower layer is tightly bonded to the hardware interface and is responsible to communicate at a lower level with the FPGA and the CPU processes. The second layer called Machine Control Interface (MCI) connects all the user interfaces by providing APIs that enable the servers to access the configuration parameters, the status information, and the data acquired by the instrument.

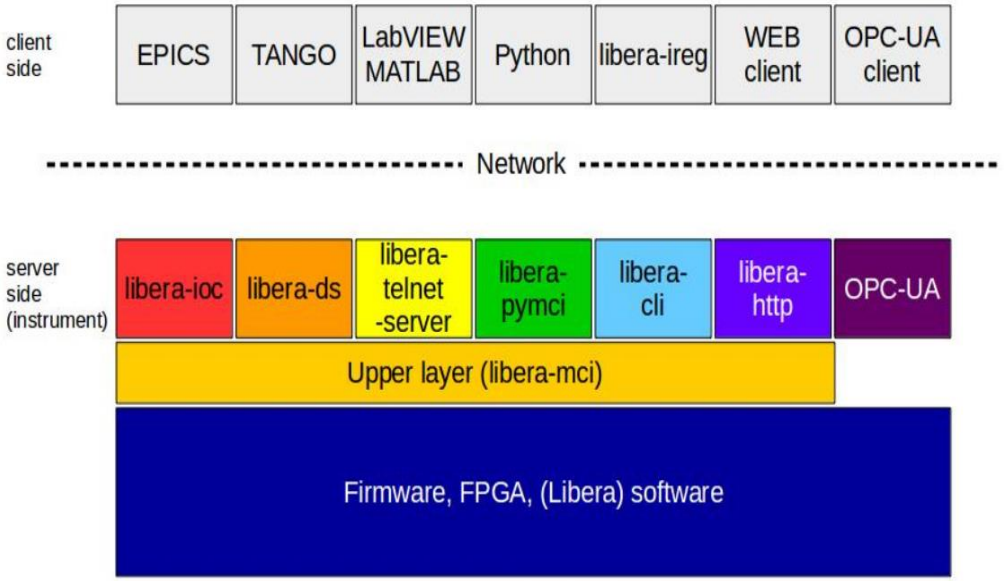


Figure 1: Application stack layer

HTTP APPLICATION ARCHITECTURE BASED ON REST API

A typical use case is reported in Fig. 2 where the instrument is accessible using the wired and wireless network.

The system architecture is based on the REpresentational State Transfer (REST) software architectural style and provides the services through Application Programming Interfaces (API) that allow the programmer to easily implement access to the instrument internal interfaces.

The requests are performed on the client-side by sending desired parameters in JSON format to the REST API. The server will then return the requested node in JSON format that will be processed by the client and presented in a human-readable format on the WebGUI.

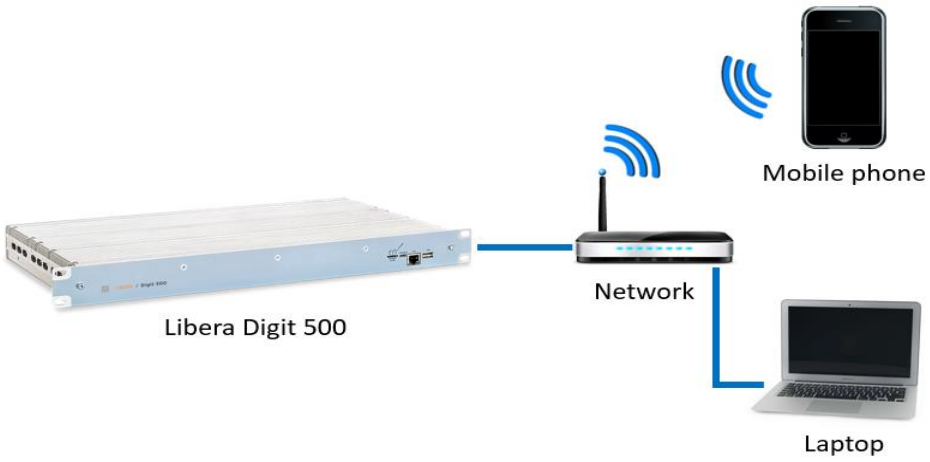


Figure 2: Access to the instrument in a local network

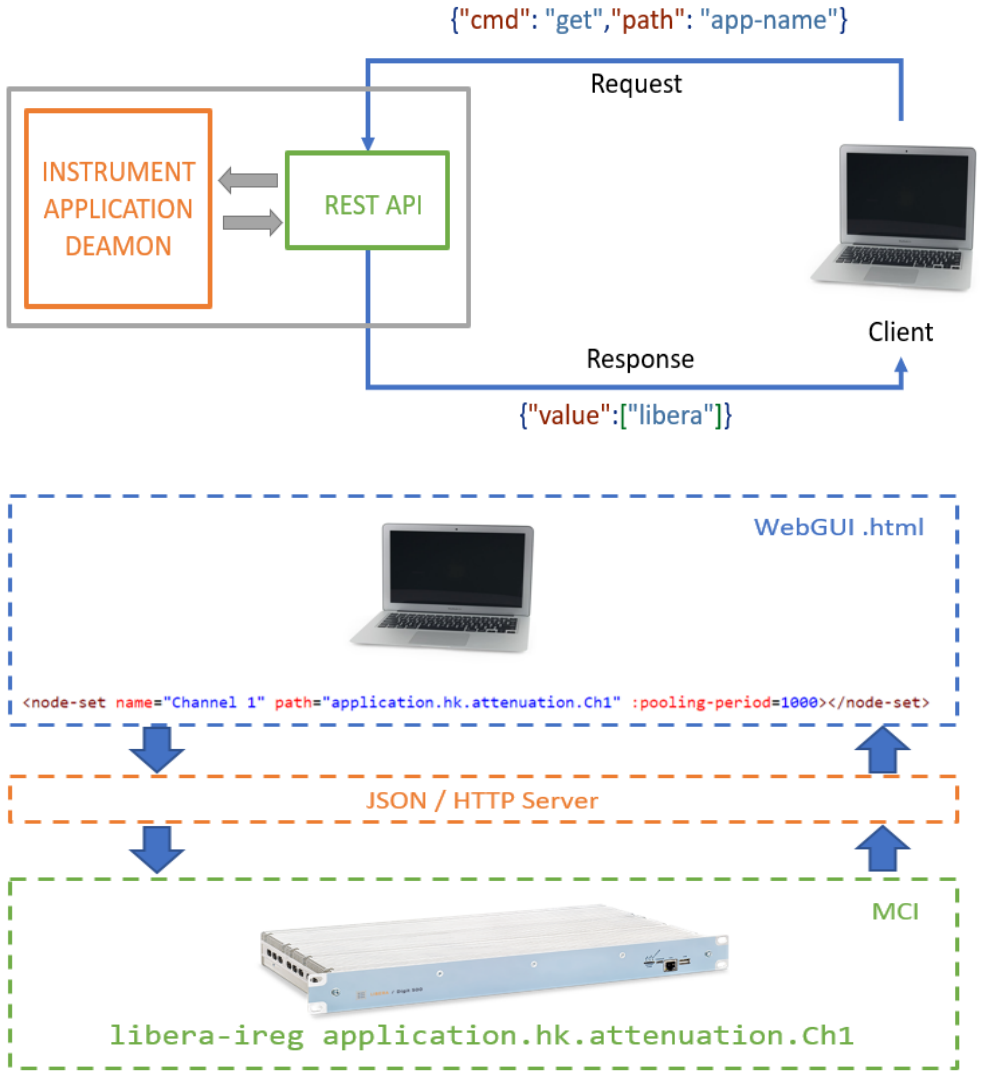


Figure 3, Figure 4: HTTP REST API application architecture.

HTTP APPLICATION ARCHITECTURE BASED ON WEBSOCKET

An additional high-performance interface based on WebSocket technology was integrated into the Libera software.

The interface is compliant with the RFC 6455 and compatible with the HTTP protocol by allowing a full-duplex communication between the client and the server running without the need to re-establish the communication at every request but by keeping it open to facilitate the real-time data transfer and data streaming over TCP. Fig.5 reports the architecture implemented with the WebSocket architecture:

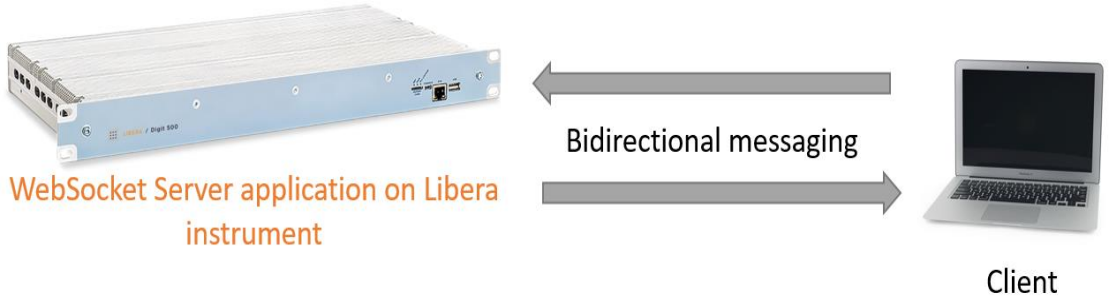
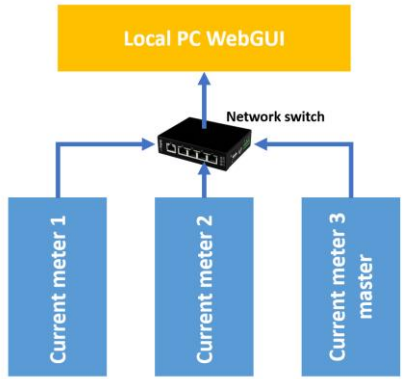


Figure 5: Access to the instrument using WebSocket

EXTENSION AND CUSTOMISATION OF THE HTTP USER INTERFACE: DEVELOPMENT AND INTEGRATION OF A 12 CHANNEL CURRENT METER

The realization of the 12 Channel Current Meter GUI interface was one very extensive customization of the user software that consisted in the implementation of really demanding features listed below:

1. Capability to perform calculations on the data ac-quired such as mean values and standard deviation for the acquired signals.
2. Continuous backup of the acquired data in .csv file format.
3. Integrate the data acquisition from 3 independent 4-channel acquisition instruments in a 12 channel device.
4. The ability of the WebGUI to run smoothly for one month operation, without any loss of data and by keeping the measurement going during the operation.



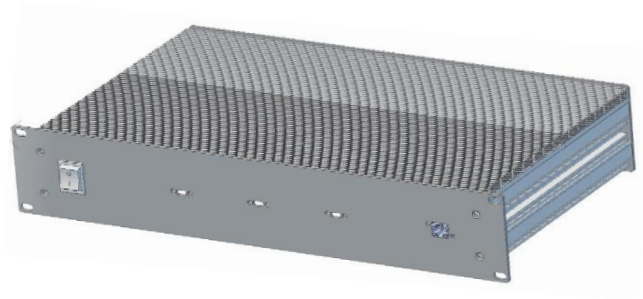


Figure 6: 12 Channel Libera Current Meter implementation.

The decision was to implement all the communication based on WebSockets.

All the 3 integrated instruments are acting as an independent device with 12 channels by keeping also the synchronization constant without acquisition timing drift between the 12 channels.

The resulting WebGUI is reported in the image below:

Figure 7: Libera 12 Channel Current Meter WebGUI

RESULTS AND CONCLUSIONS

The development of an HTTP REST API and Web-Socket interfaces required a lot of effort in terms of system implementation, integration, testing, and debugging. One of the main benefits is that such a system can be easily accessed by any final user without the need for any control system software (EPICS, TANGO) or proprietary software (Matlab, Labview) already running on the data acquisition system. These features made it very reliable and the first choice of use during system troubleshooting and installation.

The other benefit was to introduce high scalability of the interface: many instruments can benefit from this implementation since the HTTP server is available as a quick add-on for all the Libera instruments with minimal effort dedicated to porting. The performance of the WebSockets can make a difference when a large amount of data need to be retrieved from the instrument.