LABVIEW AS A NEW SUPERVISION SOLUTION FOR INDUSTRIAL CONTROL SYSTEMS

O.O Andreassen, F. Augrandjean, E. Blanco Vinuela, D. Abalo Miron, M. F. Gomez De La Cruz, A. Rijllart, CERN, Geneva, Switzerland

Abstract

To shorten the development time of industrial control applications, CERN has developed the Unified Industrial Control System (UNICOS) framework, which standardizes the programming of the front-ends and the configuration of the Supervisory Control and Data Acquisition (SCADA) layer. At CERN the SCADA system of choice is WinCC OA, but for some specific projects (small or initial prototypes not connected to accelerator operation or not located at CERN) a more customisable supervision using LabVIEW is an attractive alternative. Therefore, a system called UNICOS in LabVIEW (UiL), has been implemented. It provides a set of highly customisable re-usable components, devices and utilities. Because LabVIEW uses different programming methods than WinCC OA, the tools for automatic instantiation of devices on the supervision layer had to be re-developed, but the configuration files of the devices can be reused. This paper reports how the implementation was done, it describes the first project implemented in UiL and an outlook to other possible applications.

INTRODUCTION

The CERN Unified Industrial Control System (UNICOS) project proposes a standardized architecture for both, supervision and control layers. It also defines a set of tools for automatic instantiation of devices, in the two mentioned layers, employing re-usable components, devices and utilities.

However, for smaller lab-size solutions e.g. systems which are only expected to exist for a short time or when users do not have the time and resources to invest in a complete SCADA as WinCC Open Architecture (WinCC OA), a LabVIEW based solution would fill this gap [1][2].

UNICOS in LabVIEW (UiL) gives the user much of the same look & feel and functionality of WinCC OA as Human Machine Interface (HMI).

UiL generates automatically the supervision objects from the same configuration file provided by UNICOS Application Builder (UAB), which was developed to generate the code and configuration files of the different components of the UNICOS project (e.g. WinCC OA, Siemens PLCs etc.) [3][4]. These objects are the counterpart instances of the control objects deployed in the PLCs in the control layer.

Once the objects are instantiated in the UiL application, they can be used in the application synoptic by means of widgets, a reduced but still meaningful vision of its status, and faceplates, a full view of the object status and the place where the operator will interact with the control system.

In addition to the widgets and faceplates, UiL provides a trending tool, an alarm and an event list. All the components, widgets and user interaction in UiL are designed to resemble the UNICOS look and feel as much as possible; making sure that the transition from one SCADA tool to the other is effortless.

UNICOS

UNICOS was initially developed in 1998 as a need to develop the LHC cryogenics control system, but became a CERN de facto standard for industrial control systems. As shown in the figure 1, the framework connects, instantiates and interacts with both the supervision and control layers with commercial industrial components [1][5].

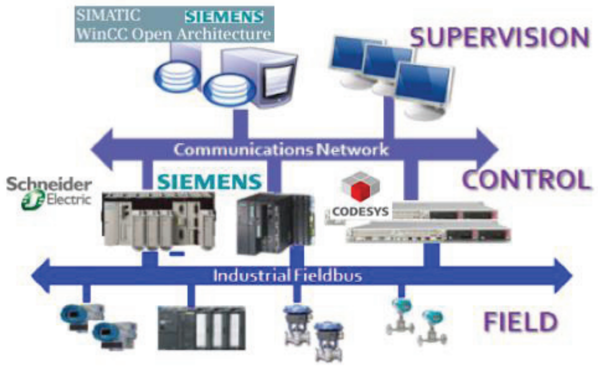


Figure 1: UNICOS architecture.

Currently UNICOS uses WinCC OA as a commercial SCADA system and both Siemens, Schneider, as Programmable Logic Controllers (PLC) [1][6]. A third possibility using Codesys extends the usage to industrial computers (IPC)

The possibilities and the application of the UNICOS framework extend to monitoring and supervision applications (e.g. Magnets alignment, Collimators, Quench protection systems...) where the control layer could be not based on UNICOS.

The UNICOS framework is based on an object oriented concept consisting of standard object classes featuring process control objects, controllers, valves, pumps, heaters, inputs, outputs etc. It also establishes how these objects interact and formalizes the control unit definition together with the way the specific control logic is implemented.

ISBN 978-3-95450-148-9

UNICOS IN LABVIEW

Although deploying a UNICOS/WinCC OA SCADA solution is easier than starting from scratch with WinCC OA, for some applications this solution can still be too difficult, and/or too costly in terms of needed expertise knowledge. In some cases, the developers lack the time, knowledge, manpower and/or resources to invest in a complete SCADA as WinCC OA. Many times the developers are already familiar with LabVIEW as well. These are the basic justifications for the UNICOS in LabVIEW (UiL) project [1][7].

The objective of the UiL project is to offer a simple and affordable solution including most of the core functionality and tools found in UNICOS. The combination of a simplified UNICOS functionality and LabVIEW's graphical programming interface, makes it more suited for lab-size and prototype applications with a substantial reduction on engineering costs (Figure 2). In addition, the site-licensing scheme adopted at CERN for LabVIEW reduces the cost of ownership for smaller SCADA systems.

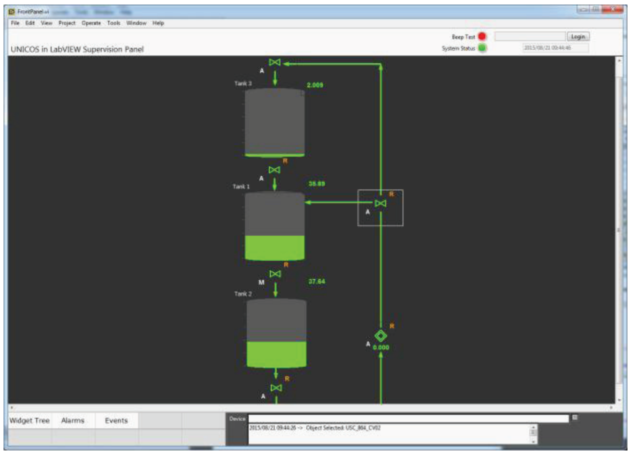


Figure 2: UiL example application

Architecture and Components

UiL is implemented using several commercial off-the-shelf components as LabVIEW, LabVIEW Datalogging and Supervisory Control Module (DSC) and a third party Python bindings application interface to MongoDB, an open-source document database designed to ease the development and scaling [8].

The general module architecture can be seen in Figure 3. The application revolves around a client, which subscribes, caches and re-serves data throughout the application using LabVIEW's internal notifiers and queues. The most resent updates are stored and broadcasted through an internal notifier, which can be read (by reference) in any sub module of the application. The larger data sets (arrays and vectors for trends and graphs) are stored in an internal queue, and indexed by object type identifiers.

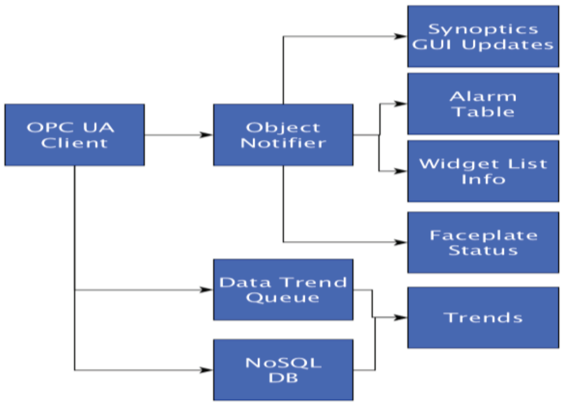


Figure 3: UiL modules.

Communication

The communication between UiL and the UNICOS based PLC's is based on OPC UA. OPC UA is a platform-independent, service-oriented architecture specification that allows the exchange of data in the industrial automation space [9]. The UNICOS based PLCs targeted are Siemens and Schneider. S7 and Modbus over Ethernet are used by those PLCs respectively. Moreover, they implement a CERN in-house protocol, the so-called Time Stamp Push Protocol (TSPP). This protocol is an event-based protocol; data is transmitted towards the supervision layer once it changes and it is sent together with the source time stamp [10].

A dedicated OPC UA server, encapsulating the TSPP protocol, has been developed at CERN. Having such server, any OPC UA client can connect to it and get the PLC data out, as it is the case of UiL, which includes an OPC UA client. This solution is very convenient as it simplifies the communication having only a single mechanism independent of the PLC (Figure 4).

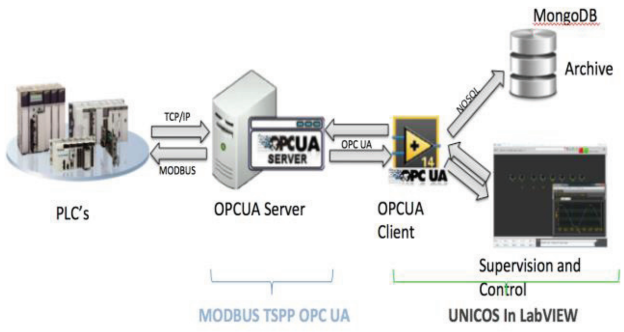


Figure 4: UiL communication.

Data Flow

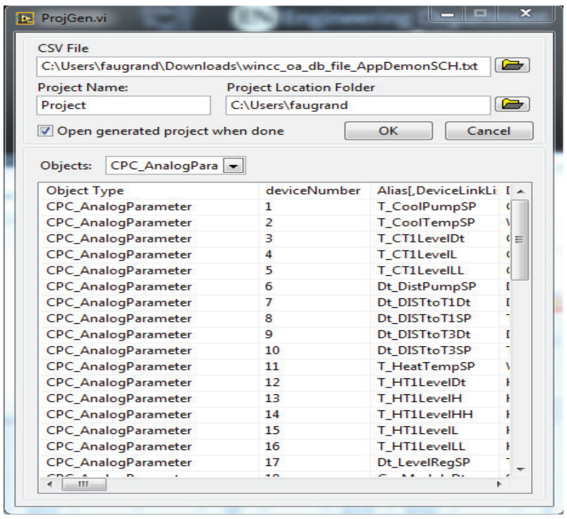
Data in UiL is primarily stored and accessed through the use of internal LabVIEW notifiers. For trends, the data is queued, in order to accurately display all data from the PLC without losing data points in the graph. The trends and events also access the NoSQL database for the viewing of historical data (Figure 4).

Archiving

UiL has the capability to archive all live data and settings. When the user selects this option in the UNICOS specification file for a particular device, the data is automatically cached and stored. UiL archives its data in MongoDB, a cross-platform document-oriented database. Classified as a NoSQL database. MongoDB eschews the traditional table-based relational database structure in favour of JSON-like documents with dynamic schemas, making the integration of data in applications such as UiL easier and faster [1][4]. The method used to archive data is class based and intended to support multiple sinks through class overrides in the future. This would make data retrieval possible directly from a WinCC OA archive, hence facilitating integration in between these to supervisors.

WORKFLOW AND DEVELOPMENT

The starting point for UiL configuration is the UAB generated configuration file [3][4]. This file contains the front-end, typically a PLC, parameters and the devices to instantiate. The user navigates to the file, selects a project name and location where to save the project (Figure 5). The IP address or hostname of the PLC is retrieved from the configuration file when the program runs after creation. The user can also manually set this if one wishes to connect to a simulated device.



CSV File: C:\Users\faugrand\Downloads\wincc_oa_db_file_AppDemonSCH.txt

Project Name: Project

Project Location Folder: C:\Users\faugrand

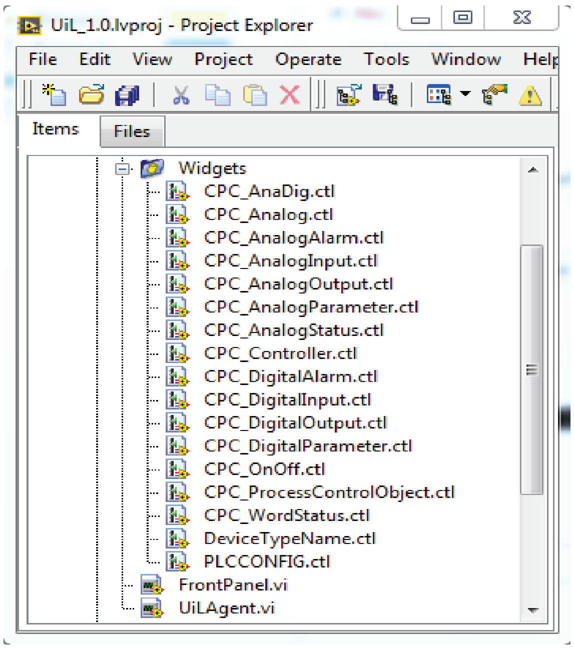
☑ Open generated project when done

Objects: CPC_AnalogPara

Object Type	deviceNumber	Alias[,DeviceLinkLi
CPC_AnalogParameter	1	T_CoolPumpSP
CPC_AnalogParameter	2	T_CoolTempSP
CPC_AnalogParameter	3	T_CT1LevelDt
CPC_AnalogParameter	4	T_CT1LevelL
CPC_AnalogParameter	5	T_CT1LevelLL
CPC_AnalogParameter	6	Dt_DistPumpSP
CPC_AnalogParameter	7	Dt_DISTtoT1Dt
CPC_AnalogParameter	8	Dt_DISTtoT1SP
CPC_AnalogParameter	9	Dt_DISTtoT3Dt
CPC_AnalogParameter	10	Dt_DISTtoT3SP
CPC_AnalogParameter	11	T_HeatTempSP
CPC_AnalogParameter	12	T_HT1LevelDt
CPC_AnalogParameter	13	T_HT1LevelH
CPC_AnalogParameter	14	T_HT1LevelHH
CPC_AnalogParameter	15	T_HT1LevelL
CPC_AnalogParameter	16	T_HT1LevelLL
CPC_AnalogParameter	17	Dt_LevelRegSP

Figure 5: UiL Project generator.

Building the synoptic for a UiL application is drag and drop based. In the LabVIEW Project window, the 'Widgets' folder contains all object types as well as other graphical indicators and decorations. During the generation of the widgets, all items have been given a type identifier, which is used internally by the UiL's scripting engine. This facilitates all animations and user interactions with the widget.



Widgets
- CPC_AnaDig.ctl
- CPC_Analog.ctl
- CPC_AnalogAlarm.ctl
- CPC_AnalogInput.ctl
- CPC_AnalogOutput.ctl
- CPC_AnalogParameter.ctl
- CPC_AnalogStatus.ctl
- CPC_Controller.ctl
- CPC_DigitalAlarm.ctl
- CPC_DigitalInput.ctl
- CPC_DigitalOutput.ctl
- CPC_DigitalParameter.ctl
- CPC_OnOff.ctl
- CPC_ProcessControlObject.ctl
- CPC_WordStatus.ctl
- DeviceTypeName.ctl
- PLCCONFIG.ctl

FrontPanel.vi

UiLAgent.vi

Figure 6: LabVIEW UiL project window.

To place specific devices into a synoptic, the user drag-and-drops a widget (Figure 6) onto the Front Panel which triggers a window where the user can select the object the widget must be linked to, choose an icon representation, and what device reference to connect to. Once configured, the widget is animated on the applications front panel.

VALIDATION

The UiL development has been initially tested in terms of connectivity with two OPC UA servers, interfacing with one PLC and one virtual device and up to three hundred channels. The test machine was a standard HP Compaq 8000 elite with 4Gb of RAM running Windows 7 64bit. The UiL LabVIEW instance was running in 32bit mode. The test bench had 200 active channels to the two servers and all the data was logged to its MongoDB archive. No significant increase in CPU (was at 30% when starting and stayed at 35% after the application ran) or Memory (used 200MB at start and had 350 Mb after 48H run) was noticed while running the test. The test provoked a random alarm every 30 minutes on random channels, and stored all the data while running. The update frequency was set to 1 point every half second, fixed.

The results in terms of animation were also very satisfactory having several dozens of devices in the same synoptic refreshing continuously. Interaction with the devices (via dedicated buttons) in the faceplates associated to the widgets was also smooth and efficient.

The current OPC UA implementation only communicates with one PLC, and some of the status indicators and alarms rely on live data from the OPC-UA server. This can cause confusion in case of communication loss by not having a proper diagnostic in the proposed architecture.

The current UiL release only supports a single database archive per application instance. This limits the cross application interaction although this will be addressed in next releases.

FUTURE PLANS

A consolidation phase is the first step to make the UiL a solid and deployable solution. Up to now all the concepts have been validated during the development and test phases. The plans include the development of a full palette of widgets and faceplates for the different sensors, actuators and control devices. The alarm and event managers will incorporate filtering capabilities and the diagnostics of the front-ends will be properly interpreted and shown to the user.

Support of multiple connections to PLCs will be added, but we still have to decide if this will be done at the OPC-UA server level or with the UiL client connecting to multiple servers.

Cross communication between different instances of UiL backends will be implemented in the next release.

CERN services will be included in future releases, these comprise the interface with LASER, the CERN central alarm system, the DB Logging, the CERN long term database and the CERN middleware (CMW) to exchange data from the UiL to a third party applications.

With the current class based data subscription model and in combination with the RADE framework, this could be done without much effort [7][11].

CONCLUSION

UiL has been successfully developed and is being used for prototypes at CERN. The combination of LabVIEW's intuitive drag and drop-based interaction, together with the similar look and feel of WinCC OA makes UiL a good choice for small to medium sized UNICOS applications as a cost effective supervisor. UNICOS is a widely used and supported standard for industrial control systems at CERN. Using UNICOS instead of a custom solution will benefit CERN users, as it will speed up development time and their support is ensured

UiL has most of the core UNICOS features, such as the generator, widgets, faceplates, contextual buttons, the alarm and event list, the trend tool and panel navigation tools.

OPC UA has been chosen as the communication layer between LabVIEW and the PLCs. Our tests show that both the Siemens and Schneider PLCs communicate well using the OPC UA interface and scales well with UiL [2][5]. This is accomplished without needing to do any changes to the UiL OPC UA client.

Our performance tests show that UiL can handle several hundred widgets running simultaneously without any significant load to the CPU.

REFERENCES

[1] P. Gayet et al. “UNICOS a framework to build industry like control systems”, ICALEPCS 2005, Geneva, Switzerland, (2005)

[2] Siemens “SIMATIC WinCC”, (1996), https://en.wikipedia.org/wiki/WinCC

[3] I. Prieto Barreiro et al. “UAB CONTINUOUS INTEGRATION FOR AUTOMATED CODE GENERATION TOOLS”, ICALEPCS 2013, San Francisco, USA, (2013)

[4] I. Prieto Barreiro “UAB Bootstrap”, CERN 2012, Geneva, Switzerland, (2012)

[5] R.Barillère et al. “A HOMOGENEOUS APPROACH FOR THE CONTROL OF THE LHC EXPERIMENTS GAS SYSTEMS”, ICALEPCS 2003, Gyeongju, Korea, (2003)

[6] B. Farnham et al. “Migration from OPC-DA to OPC-UA”, ICALEPCS 2011, Grenoble, France, (2011)

[7] O. Ø. Andreassen et al. “The LabVIEW RADE framework distributed architecture”, ICALEPCS 2011, Grenoble, France, (2011)

[8] K. Banker “MONGODB IN ACTION”, p. 375, ISBN 9781935182870, (2011)

[9] W. Mahnke “OPC UNIFIED ARCHITECHTURE” OPC UA, (2009), https://library.e.abb.com/public/75d70c47268d78bfc 125762d00481f78/56-61 3M903_ENG72dpi.pdf

[10] J. Ortolá Vidal et al. “AN EVENT DRIVEN COMMUNICATION PROTOCOL FOR PROCESS CONTROL: PRFORMANCE EVALUATION AND REDUNDANT CAPABILITIES”, ICALEPCS 2013, San Francisco, USA, (2013)

[11] A. Dworak et al. “THE NEW CERN CONTROLS MIDDLEWARE”, CHEP 2012, New York, USA, (2012)