RETROFIT PROCESS PLANT DIGITALIZATON

The Code of Your Process Plant

IntelliFlux Controls, Inc.

Subir Bhattacharjee

IF CTRL

Contents

Preface

Retrofit Process Plant Digitalization is a series of articles on Digital Transformation in the Process Industry. Most of these articles are transcripts of my presentations at Universities, Conferences/Trade Shows, or Webinars during the past three years. As my 2020 travel came to a halt, I decided to put these scripts in the hands of my marketing team at IntelliFlux Controls as precursors for their marketing writeups. I thought that our marketing experts will consider these quite dense (none of my writeups get a Flesch Readability score above 30). I was surprised when they suggested I post them on the Company Website as "position papers" or "thought articles". Well, as long as these are not printed on paper, and it only hurts electrons to transmit these, I am not averse to posting these. I hope that someone in the process industry, and anyone interested in exploring digitalization, may find these articles informative. I hope that professionals facing challenges with digitalization or grappling with the digital transformation drive around the world will find nuggets of interest in the experiences I share about retrofit digitalization.

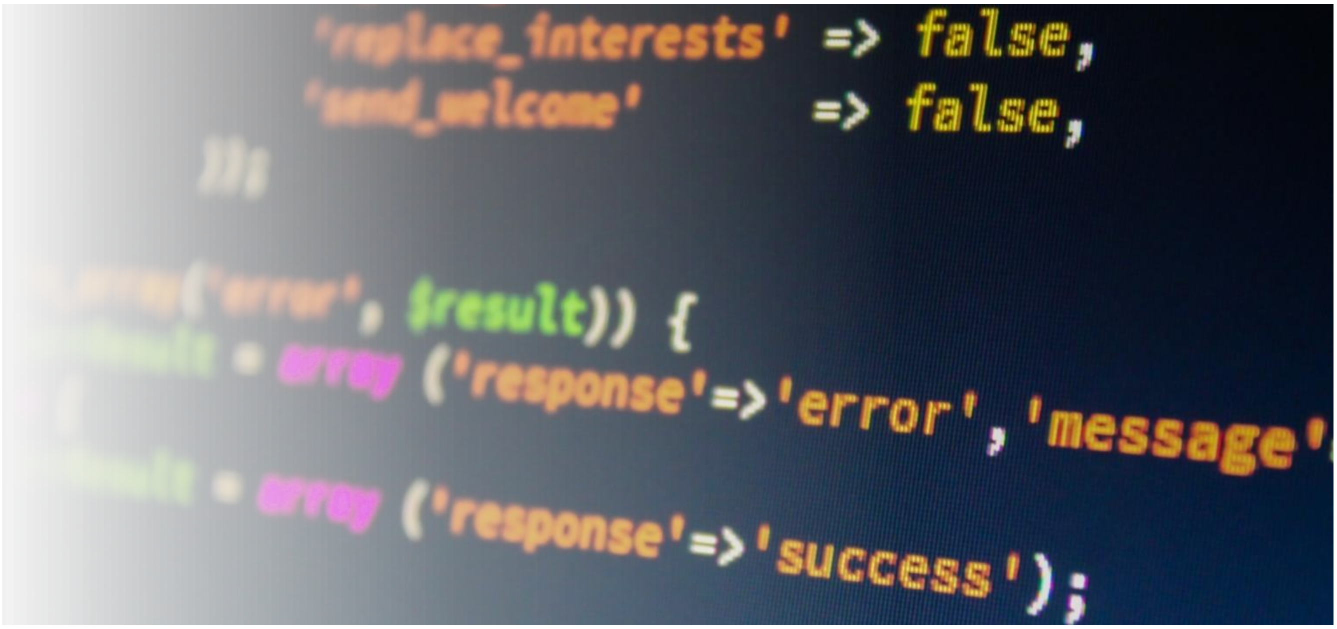
The present article "*The Code of Your Process Plant*" discusses one of the most overlooked topics related to digitalization in the process industry. We have rarely seen process industry experts and leaders being aware of the state of automation in their plants, and about the programs that run these plants.

The first step toward digital transformation of your process plant could be to direct attention to an invaluable asset that already exists at your plants – the automation programs and codes that currently run them. The process industry already has a digital automation infrastructure in place. Therefore retrofit digitalization may not be too onerous for this industry. This article provides an overview of where these programs reside, what these programs are, and how to capitalize on them when you are planning on your plant's digital transformation.

I am grateful for the tremendous support from IntelliFlux Controls, our stakeholders, our customers and well wishers. As the founder and CEO of IntelliFlux Controls, I do see everything through the lens of my IntelliFlux journey. So, my technical writeups cannot be strictly classified as "impartial and academic" anymore. That said, I still appreciate the freedom my company gives me to write the way I like to write – objectively. Finally, if there are any errors or omissions in this article, the responsibliity is entirely mine.

Subir Bhattacharjee
CEO
IntelliFlux Controls, Inc.

June 2020



The Code of your Process Plant

Your process plant was built as a system through efforts of a project team that involved, among other professionals, chemical (process), civil, electrical, instrumentation, and mechanical engineers.

The core manufacturing process was first envisioned by process developers and engineers, who performed laboratory tests, feasibility analysis, scaling, and piloting. They conceptualized the actual sequence of processes and operations that will convert the raw materials into the products you manufacture. This conceptual plant design was then implemented layer by layer through a group of professionals who pieced together the foundation, the mechanical design, sizing and specifying the various unit operations and components, and even the power, water, and other resource requirements. Then the plant design was converted to reality through a capital project which included procurement of the components, assembly, piping, mechanical, electrical, and control integration.

And then there was commissioning, and the plant went into production…

From concept to realization, the plant was transformed by the efforts and expertise of a wide range of professionals, with every step of the process documented in detailed designs, specifications, diagrams, layouts, instructions, mathematical models, and operating instructions. A critical component that one often misses out in the above list are the **programs or codes that integrated, synchronized, and automated the operations of the various processes of the plant – the so-called automation layer** of the plant. The plant is essentially run through its lifespan using these programs and codes.

Importance of the Code

Why is the automation code of your existing process plant important? What is in the code? These are important questions, and their answers can probably unlock considerable value you can additionally derive from your process plant.

The code of an existing plant - how the plant is currently automated and programmed, how the data flows, and how the information is collected and analyzed - become important considerations when implementing modern **digitalization upgrades to a plant.** The code of your existing plant is a foundation on which you can develop your **"smart plant" upgrade**.

The automation platform acts as the nervous system of the plant. On one hand, it is the sensory system, providing feedback to the operators about how the plant is performing, or indicating any fault in a component. On the other hand, it is a controller of actions, repeatedly and reliably guiding the human operator and the plant machinery through cycles of startup, production, maintenance, and shutdown.

The automation programs and codes are the blueprints of the brain and nervous system of the plant. The plant automation code is a single repository of information and critical knowledge about your plant's design and its operating characteristics. It encapsulates the plant's operational protocols, the operating philosophy of each process component, operating conditions, and equipment performance limitations.

The automation programs are the instructions to process the stimuli and provide response – through adjustment of setpoints and conditions. These programs are the software fingerprints of your plant. These codes give life to the plant and keep it functioning. **The automation programs define the "personality of the plant"**.

Retrofit digitalization often requires building on the existing automation framework of the plants. We call this a **bottom-up approach** of retrofit digitalization. This approach requires a thorough understanding of the processes and automation steps of the existing plant. This assessment of the existing automation and process control framework of a plant is important to avoid conflicts between the existing operational framework and the new digitalization systems to be implemented.

The value addition through retrofit digitalization is strongly correlated with the capabilities and sophistication of the existing automation framework of a process plant. A thoughtful investment in this bottom-up approach of implementing a digitalization retrofit can significantly improve capacity utilization, efficiency, reliability, and sustainability of existing process plants.

Process Plant Automation Architecture

The code of the plant is embedded primarily in the plant automation layer or within the industrial control system (ICS) [1]. To understand where the code exists, let us briefly review the general layout of how automation is implemented in these plants, and what are the hardware components that are programmed. Figure 1 depicts the layout of a conventional distributed control system (DCS) architecture implemented at process plants.

Most plant hardware (sensors, gauges, motors, mixers, blowers, valves, etc.) are sensory or motor devices whose operations can be automated. Collectively, these sensory and motor components are termed as remote terminal units (RTUs). There is an embedded programmable electronic converter component in each hardware device. This converter can convert the physical interactions of sensory devices with their environment into calibrated electrical signals. For instance, the temperature measured by a temperature sensor, or tank level measured by a level gauge can be converted to an electric current of magnitude between 4 – 20 milli-Amperes. The electric signals and instructions received by converters in motor devices alter their action on the environment. For example, the frequency of an electrical signal in a variable frequency drive (VFD) of a pump motor can be changed to alter the motor speed, and consequently the pump flow rate.

Individual RTUs are connected to a single focal point within a process or equipment. This focal point contains a more general-purpose microcontroller that can process these signals and uses a set of instructions (programs or codes) to operate the individual sensory and motor components of the process. These general-purpose microcontrollers are often built using programmable logic controllers (PLC). The PLCs can be programmed, and the code reads the sensory signals as inputs and utilizes its programmed logic to send output signals to the motor components.

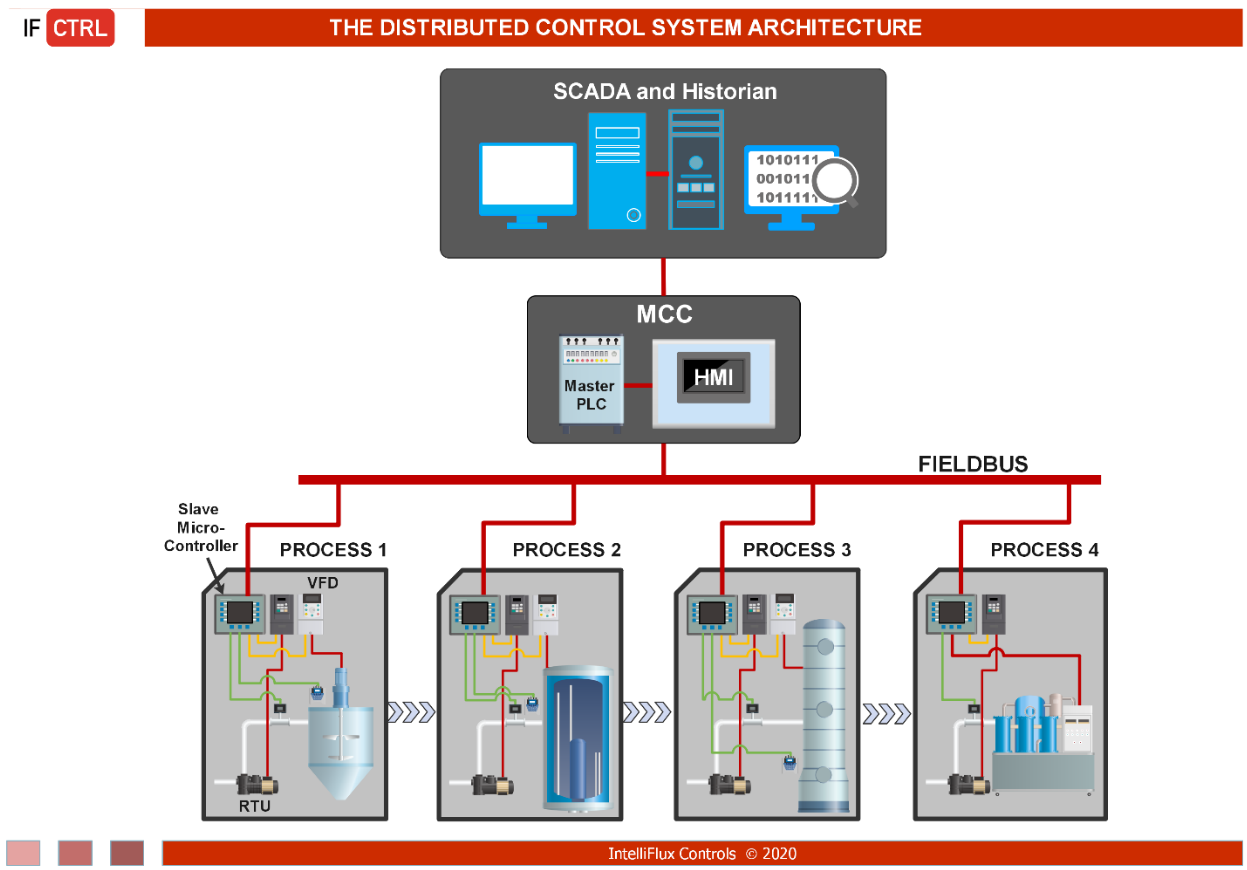


Figure 1: Layout of a Distributed Control System (DCS) at a typical process plant.

In a complex process plant, individual processes are often procured from different vendors, who design and integrate these processes using their own programming codes and conventions. Individually, each process is programmed to operate with a certain level of automation that is managed at the process level.

When a plant consisting of multiple processes is integrated, the microcontrollers and PLCs of the individual processes are connected to a central control center at the plant, the Master Control Center (MCC). At this MCC, the individual machinery or process equipment PLCs are connected to a larger microcontroller, which is usually a more powerful PLC. The PLC residing in the MCC acts as the master controller that can receive signals from the individual process equipment PLCs (slaves) and can transmit instructions to them. This configuration is often referred to as the Master-Slave configuration. The MCC is the location at a plant where operators and engineers can monitor the overall performance of all the individual processes at the plant and can send instructions to the individual process level PLCs to change the setpoints and operating variables. The operator interacts with the plant components through a human machine interface (HMI) located at the MCC, although HMIs can also exist at individual process level control stations.

On top of the MCC, complex plants often have a supervisory control and data acquisition (SCADA) layer, which accumulates all the process conditions and settings, including time traces of the operational data. The SCADA has a historian, which is a database that collects all the plant data. The database is often some form of a relational database (typically some variant of sequential query language, or SQL). At the SCADA layer, operators or engineers can monitor the plant performance and analyze historical data to observe the performance trends or troubleshoot operations. Sophisticated SCADA systems can provide instructions to the Master PLC or individual process PLCs; however, in process plants, most SCADA systems are typically used as monitoring and historizing systems. Generally, engineers monitoring a plant operation at the SCADA level contact the on-floor engineers and operators when they detect any anomaly, and the process is modified at the master or slave PLC level by the operators. Communication between the various PLC layers and the SCADA layer are often done through a multitude of communications protocols such as Profibus, OPC, Modbus, TCP/IP, *etc*.

Where Does the Plant Automation Code Reside?

IntelliFlux Controls has primarily focused on digitalization of municipal and industrial water treatment, food and beverage processing, textile processing, chemical manufacturing, upstream oil and gas, and mining operations. Aggregating information from all these plants, we have compiled a snapshot of where most of the code for a process plant is found, and what are the languages in which such codes or instructions are written. Figure 2 summarizes these findings.

At the plants IntelliFlux has worked on, about 86% of the primary process control, monitoring, and automation code resides at the PLC level, usually distributed between the master and slave PLCs. Some SCADA level codes are also encountered (about 14%). The PLCs at the MCC are the locations at which most of the plant data from sensors converge as input signals. The master PLC communicates with an HMI, through which the operator interacts with the plant by changing its setpoints. This master PLC also serves as the gateway of information to the SCADA layer. The SCADA level codes involve historizing and trending of the process information, and in some cases, the SCADA level codes provide alarms and analytics that can be used by service providers (Call centers/engineers) to alert the plant operator when needs arise. The primary role of the SCADA layer is that of a decision support system for the operations team, and in some cases, avenues for business management to gather plant performance information such as uptime, throughput, inventory, and operating conditions.

The individual process level PLCs often contain more detailed codes and instructions controlling the automation and operation of the individual processes. These are frequently programmed by the individual process system providers when proprietary process control steps and logic are involved. In some cases, the component technology provider does not pre-program these process components, but the plant automation integrator programs these local process PLCs along with the master PLC.

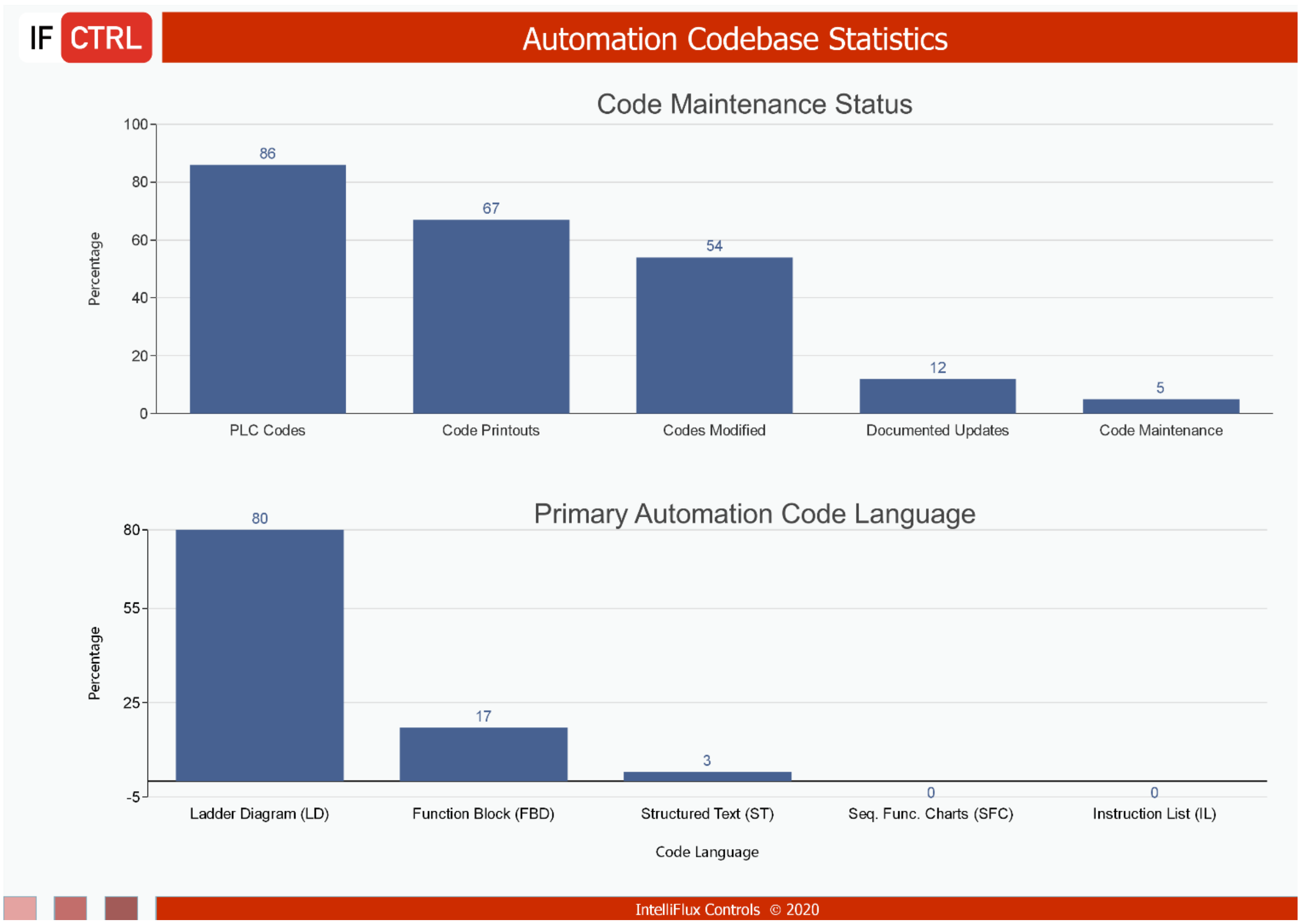


Figure 2: Top: The percentage distribution of the process automation codes locations, documentation history and maintenance status. Bottom: Distribution of the programming languages used for process automation codes.

Although many RTU devices can be programmed, for instance, the VFD of a pump, or a programmable dosing pump, system integrators often avoid directly programming at the RTU level. Instead they connect the RTU devices to the slave or master PLCs and perform all the automation coding at the PLC level to control the automated operation of the RTUs.

The PLC code is typically written at the time of commissioning of a plant. In many cases (about 67%), we only had access to a printout of the original code for the as-built plant. However, for many of these plants, the codes were modified at times (typically when there were additions to the plant, hardware changes, or process modifications). Many of these changes or updates to the PLC code were not properly documented (we only found 12% of the changes properly documented), and only about 5% of the automation codes at these plants were subject to some type of regular maintenance and update.

The Coding Language

PLC programming is performed using five programming languages. These are, (i) Ladder diagrams (LD), (ii) Function Block Diagrams (FBD), (iii) Structured Text (ST), (iv) Sequential Function Charts (SFC), and (v) Instruction Logic (IL). Among these, ladder diagrams appear to be the most common language for programming PLCs. 80% of plant automation codes encountered by IntelliFlux were developed using ladder diagrams.

It will be interesting to note that most conventional software programmers familiar with high level computer programming/scripting languages like C, Java, Python, or even the venerable Fortran or Pascal will not be familiar with most of the PLC programming languages (barring some vague familiarity with structured text). Someone familiar with assembly level programming may also find Instruction Logic (IL) somewhat familiar. To the best of our knowledge, no mainstream university curriculum teaches PLC programming. Polytechnics, community or technical training colleges and institutes, or vocational training institutions offer courses on PLC programming. Typically, PLC programmers are familiar with specific platforms of PLCs, and most PLC programmers have learnt these programming languages on the job or through courses offered by PLC hardware vendors or their distributors. PLC programs are written using software integrated development environments (IDE) specific to each type of PLC hardware (for instance, Simatic WinCC for Siemens, or Studio 5000 for Allen Bradley brand PLCs).

The State of the Process Plant Code

The programs and codes that run the automation layer of the plant, particularly at the SCADA, PLC and RTU level, are often poorly understood by software developers and IT professionals outside the realm of automation hardware professionals. The software coding tasks to implement automation is frequently outsourced by engineering and design firms. Their testing is mostly done to only ensure functionality. The codebases are quite fragmented and platform specific. Their maintenance, upgrade, and documentation are largely neglected, or even forgotten after initial plant commissioning. Over the lifetime of a plant, any modification to these codes are usually done as patchwork, frequently associated with additions or alterations to the plant hardware; many such modifications are not even properly documented. We have rarely found a consolidated, well documented, updated repository of the programs running a plant.

PLC or microcontroller programming is done on a separate computer using an IDE software provided by the PLC manufacturer, and the program is uploaded to the memory of the microcontroller. When an update to the program is needed, the old code from the microcontroller is downloaded by the programmer to a computer, and after alterations are made, the program is loaded back to the microcontroller memory. Documentation of such modifications is highly variable in quality, frequently unsatisfactory. New features and functionalities are added to the code without comprehensive review of the prior code and its logical constructs. In some cases, we found password locked function blocks in the code, which could not be readily retrieved as the customer could not find the password set up by the PLC programmer who coded the program originally.

PLCs are nowadays made by several manufacturers, and they embed different types of microprocessors, utilize manufacturer specific programming languages, data handling features, firmware, drivers, and function libraries. There are also issues about the software and firmware versions when it comes to upgrading PLC codes. Certain PLC brands and models lack backward compatibility of their programming software for writing PLC codes. Hence, many old codes require finding the older versions of the IDE software to open the program.

It is important to realize that PLC codes developed and delivered at the time of plant commissioning should be reviewed following proper guidelines. In fact, IEC 61131 [2] has standard guidelines related to PLCs, code language standards, validation and testing procedures, and factory acceptance of automation codes. However, practical implementations of these factory acceptance testing, which occur between the commissioning team and the process automation programmers, are mainly based on point by point check of the programmed hardware functions. We have seen disconnects between the intended process dynamics and control originally envisioned by the process designers, and the PLC implementation of the process control. One common aspect we note is the way sensor data are aggregated without regard to sensor physics and dynamics. We have also encountered PLC codes where the intended Proportional-Integral-Derivative (PID) control constructs specified by the process engineering and design team were substituted by an on-off control around a fixed setpoint.

It is also worth noting that sophisticated control strategies such as auto-tunable PID control, fuzzy logic implementation, MIMO, etc., are not best suited for implementation at the PLC level, as PLCs are not high powered memory intensive computing platforms, but are extremely precise and robust timing devices. The PLC level programs can only handle simple instructions on single rungs of a ladder logic program and are best suited for coding logical instructions. However, in many smaller plants, we have encountered PLC level implementation of process control constructs like PID feedback/feedforward loops, temperature/pressure correction of fluid properties through complex equations and data interpolations, data lookup, iterative calculations, and even model-based process control. Probably such PLC level implementations were outcomes of value engineering, but such approaches of packing in heavy weight computations into a PLCs ladder logic code can adversely affect time critical non-linear processes (specifically chemical reactions and thermal processes). One cannot envision implementing complex and iterative numerical or statistical algorithms at the PLC level.

Coding of a more complex nature can certainly be done at the SCADA level. In fact, most modern digitalization system implementations that perform tasks such as remote monitoring, reporting, process visualization, predictive analytics, or even machine learning, can be viewed as SCADA layer constructs. Classical SCADA systems (such as, GE Digital's Proficy or Cimplicity, or Rockwell Automation's Factory Talk) have evolved over the years to provide sophisticated layers of automation programming and connectivity to many industrial sectors. Many of our customers have experimented with a diverse range of SCADA layer tools available at different costs (for example OSI PI, Schneider Ecostruxxure, or Inductive Automation's Ignition). The problem is not with the availability of the software frameworks for SCADA programming, but rather how these are more generic IDE type tools, with their own function libraries, proprietary firmware, and restricted set of hardware platform support. Fundamentally, even with these tools, the responsibility of writing the exact codes and implementing efficient process control programming still rests with the programmers and system integrators that design, build, and program the plant. This is where the key challenge exists. The process engineering, instrumentation, and process control programming teams should have a close coordination during implementation of the operations logic for the plant. When this synchronicity does not exist, there are gaps in the implementation of even the higher levels of automation and digitalization.

Considerations for Implementing Digitalization to Existing Process Plants

To implement a bottom-up digitalization framework, it is important to assess the capabilities of your plant's existing automation framework, clearly articulate the goals to be achieved by the added digitalization features, key performance indicators (KPIs), expected returns on investment (ROI), perform a gap analysis, and then embark on an improvement plan. Fundamentally, one has to first ask: ***"What does digitalization offer that is not already in my current plant automation layer?"***

Digitalization purports to making manufacturing "smart", and this has already been implemented in several manufacturing sectors (automotive, for instance). However, in the process industry, where one deals with fluids, complex chemical reactions, phase change physics, and intricate multi-physics dynamics between reactive material and energy flows, such smartness was not readily built into the automated operations. The control systems in the process industry were traditionally designed to rely quite substantially on the process engineers at the plant to address process related perturbations and interventions. There are certain Achilles heels of the existing automation framework of process plants that hinder a simple implementation of the full extent of digital transformation and autonomous control. One of this is the lack of understanding of how your current automation framework is programmed, and how the process engineering constructs can be implemented in such control programming framework.

The core technical implementation of digitalization centers around developing a closed Physical-to-Digital-to-Physical (PDP) loop [3]. This involves (i) collecting and mapping sensory data from a physical plant to a digital realm, (ii) performing analysis at various levels of complexity in the digital domain, including basic statistical analysis, correlations, predictive analytics, and machine learning, followed by (iii) transmitting the assimilated information from the digital realm back to the physical plant for actions to optimize and enhance the performance of the plant.

One might be inclined to think that replacing the entire existing automation hierarchy (RTUs, PLC, SCADA) with a modern digitalization framework is more desirable. However, the existing plants had an automation strategy already in place, and through the PLC/SCADA codes, already had a working blueprint of the operational philosophy of the plant. In this respect, your existing automation framework could already contain certain components and enabling elements of digitalization. If properly utilized, these already existing assets can significantly speed up your digitalization journey and reduce the cost burden of implementing digitalization. It is discernible that **utilizing these existing codes as the prima facie digital twins of the plant could lead to a quicker and economically viable approach to implement digitalization** to existing process plants. IntelliFlux benefited tremendously by adopting this approach for the process industry.

The benefits of digitalization in a retrofit process plant upgrade scenario depend on the existing state of automation of the plant, the desired digitalization goals, and the cost of upgrading the plant to achieve the digitalization goal. The primary benefits of converting a process plant into a smart plant can be broadly categorized into **economic benefits, reliability and operational flexibility enhancement, and achieving organization wide sustainability and circular economy goals**. The customers of IntelliFlux mostly adopt digitalization primarily as a means to achieve: (i) lower operating costs, (ii) increase process throughput or yield, (iii) enhance monitoring framework for workforce deployment to remote plant locations, and (iv) explore adaptive optimal control as an approach to improve reliability. The last factor is embodied in requests such as “can you make our sea water reverse osmosis plant operate through algae blooms?”, or “Can you make the drying process adapt to the different feedstock qualities?”, or “Can you make our water treatment plants work through influent water quality fluctuations?”. The answer is a resounding “yes” in each case. The approach we take to achieve these goals is to first assess how the existing plant is already programmed, and how flexible the programs are to allow such improvements. Our approach is to deliver retrofit digitalization through the existing automation framework and the existing codebase of the plant, without disrupting any of the functionality of the pre-existing automation framework.

Digitalization of Process Industry Through the Lens of IntelliFlux Controls

IntelliFlux Controls has implemented its retrofit digital transformation stack in many process plants around the World within a cross-section of the chemical process industry. This allowed us to assess the complexities involved in digitalization of existing process plants. We have analyzed the multilayered industrial automation architecture of these plants, evaluated the bandwidth and reliability of the communications channels, and deployed various levels of digital transformation providing decision support and decision automation to these plants without disrupting their existing automation architecture and cybersecurity frameworks.

One of the first tasks we perform for any plant is the **Plant and Process Digitalization Assessment**. This assessment considers the process automation layer architecture at a plant in conjunction with the process it is automating. The evaluation encompasses the original plant and process design, plant details, process & instrumentation diagrams (P&IDs), dynamics of individual processes, existing automation hardware including sensors, and the existing codes of the plant. This focus on the process engineering details along with the DCS features of the plant provides us a unique ability to clearly quantify the potential improvements that can be achieved with digitalization at each plant.

The following list highlights our salient experiences with digitalization of process plants so far. Every plant we have retrofitted differed in its baseline automation layout, reflecting multiple combinations of the following factors:

1. Widely varying processes, operating cultures, standard operating protocols (SOPs), and operational priorities.
2. Different levels of automation in similar plants and technologies.
3. Strong leaning toward operator mediated process interventions, and mistrust of autonomous control.
4. Diverse range of sensors, with different protocols for connecting to the automation framework.
5. Different brands, models, and capability limits of PLCs.
6. Different PLC programming languages, interfaces, drivers, firmware, and programming practices.
7. Different communication protocols between the various layers of automation (MODBUS, PROFIBUS, TCP/IP, OPC).
8. Practically non-existent to quite sophisticated SCADA hardware and software.
9. Different regulatory and cybersecurity frameworks (such as “air gaps” and segregation of business and control networks).
10. Widely varying pre-conceptions and pre-dispositions about digitalization, its benefits, risks, and consequences.

While the PLC codes generally implemented the existing automation strategy quite adequately at most of the plants, they were not programmed to adjust setpoints or institute any operational changes in response to sensor data. Any change in process set points were done by operators. There was no scope of deploying smart plant paradigms, automated process optimization, or adaptive control through the existing programming architecture.

In most cases, reviewing and utilizing the PLC automation codes allowed us to construct a fairly accurate digital twin of the process, which could be refined further during digitalization. Availability of the PLC / SCADA codes also facilitated the implementation of many decision support and digitalization steps. Finally, in the plants served by IntelliFlux, the existing DCS framework was already operating within an implemented cybersecurity and information security management system (ISMS) which often conformed to ISO/IEC 27000 [4,5]. Using the existing DCS framework to deploy digitalization did not disrupt any of the existing cybersecurity policies of the customers.

Summary

There are considerable benefits in retrofit digitalization for process plants, since it increases the longevity of existing plants, and makes production more efficient without significant capital burden and disruption in operation.

Most process technologies are quite mature, and significant step changes in process efficiency cannot be expected from new technologies. Consider water treatment, for example. Biological treatment processes are quite mature and are probably some of the least expensive approaches for treatment and reuse of municipal and industrial wastewater. Membrane processes are also quite mature. Sea Water Reverse Osmosis (SWRO) costs have been driven down remarkably over the past two decades. Most recent improvements in RO technology are primarily adaptive process control improvements fueled by automation advancements. These technologies work perfectly well when they are implemented as intended by their designers and developers. The minimal and arcane PLC level automation of the plants coupled with lack of timely operator guidance provided by these automation systems are often the key reasons why these technologies do not always perform as advertised. Converting these plants to "Smart Plants" by superimposing a digitalization layer on top of the existing process automation layer through addition of primarily software modifications can add tremendous value to the process industry with minimal disruption. Considering this, extending the life of a plant, while making them smarter and more efficient through digitalization is perhaps a judicious sustainability move for the process industry. If such a move is made, taking account and control of your existing process automation codes is a very pertinent starting point.

Abbreviations

DCS	Distributed Control System
FBD	Function Block Diagram
HMI	Human Machine Interface
ICS	Industrial Control System
IDE	Integrated Development Environment
IEC	International Electrochemical Commission
IL	Instruction Logic
IP	Internet Protocol

ISMS	Information Security Management System
ISO	International Standards Organization
LD	Ladder Diagram
MCC	Master Control Center
MIMO	Multiple Input Multiple Output
OPC	Open Personal Computer
PDP	Physical to Digital to Physical
PID	Proportional Integral Derivative
P&ID	Process and Instrumentation Diagram
PLC	Programmable Logic Controller
RO	Reverse Osmosis
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SFC	Sequential Function Charts
SQL	Sequential Query Language
ST	Structured Text
SWRO	Sea Water Reverse Osmosis
TCP	Transmission Control Protocol
VFD	Variable Frequency Drive
VSD	Variable Speed Drive

Reference

1. Keith A. Stouffer, Victoria Y. Pillitteri, Suzanne Lightman, Marshall Abrams, Adam Hahn, Guide to Industrial Control Systems (ICS) Security, Special Publication (NIST SP) - 800-82 Rev 2, National Institute of Standards and Technology (2015).

2. IEC 61131: 2020 SER Series (contains IEC 61131 – 1:2003, IEC 61131 – 2: 2018, and IEC 61131 – 3: 2013), Programmable Controllers, International Electrochemical Commission (2020).

3. Tay, S. Te Chuan, L., Aziati, A. A., Ahmad Nur A. An Overview of Industry 4.0: Definition, Components, and Government Initiatives. Journal of Advanced Research in Dynamical and Control Systems. 10. 14**,** (2018).

4. ISO/IEC 27001:2013, Information technology — Security techniques — Information security management systems — Requirements, (2019).

5. ISO/IEC 27002:2013, Information technology — Security techniques — Code of practice for information security controls, (2019).

IF CTRL

email
info@ifctrl.com

phone
+1 949 630 0940

2081 Business Center Drive, Suite 285
Irvine, CA, 92612, USA